10 Practical aspects and testability

Is it not strange that desire should so many years outlive performance?
Shakespeare: King Henry IV

Objectives

The chapter is intended to round off and summarize much of the preceding text and to discuss some of the practical realities the designer must face. The problems of communication again receive close attention and are illustrated in the context of the 4-bit data path design.

The chapter also includes a section headed 'Ground rules for successful design' and the reader will find that most of the rules, tabulated data and performance parameters are grouped together consecutively in this section. The question of noise margins and other relevant aspects, such as CIF code and CAD tools, are also discussed.

The second half of the chapter is entirely devoted to the very important subject of testability, which must always be a key design requirement for systems of any size.

10.1 Some thoughts on performance

Two important parameters (other than 'does it work at all?') are speed and power dissipation. These factors are generally interrelated; power dissipation and area are also interrelated in MOS technology.

Take, for example, the simple case of an nMOS 8:1 inverter which may be set out with a minimum feature size pull-down transistor (i.e. $2\lambda \times 2\lambda$ pull-down gate area and a minimum width 16λ long $\times$ 2λ wide pull-up channel) giving a total resistance from V_{DD} to GND of 90 kΩ. The maximum power dissipation f›r this particular design will thus be

$$\frac{(5\text{ V})^2}{90\text{ k}\Omega} = 0.278\text{ mW}$$

An alternative form of 8:1 inverter is to use a pull-down geometry 2λ long and 6λ wide with a 6λ long, 2λ wide pull-up channel giving a V_{DD} to GND resistance of 33.3 kΩ and a consequent maximum power dissipation of

$$\frac{(5\text{ V})^2}{33.3\text{ k}\Omega} = 0.744\text{ mW}$$

that is, about three times the dissipation. However, comparing the total transistor areas for each case we have, in the first case, $2\lambda \times 2\lambda + 16\lambda \times 2\lambda = 36\lambda^2$ area and, in the second case, $2\lambda \times 6\lambda + 6\lambda \times 2\lambda = 24\lambda^2$. In other words, the 3:1 (approximate) reduction in power dissipation is at the expense of a 50% increase in transistor area.

Now consider the aspect of speed (or circuit delays), and take the simple case of one 8:1 inverter driving another similar inverter. The longest delays will occur when the output of the first stage is changing from logic 0 (Lo) to logic 1 (Hi), that is, the Δ transition of the output, and the capacitances associated with the output and the input of the next stage must charge through the pull-up resistance of the first stage as in Figure 10–1. Asymmetry is also present in CMOS devices. It is also obvious that during the complementary ∇ transition the same capacitances must be discharged through the pull-down transistor of the first stage.

For the minimum pull-down feature size nMOS 8:1 inverter, for example

$$R_{p.u} = 8R_s$$
$$R_{p.d} = 1R_s$$
$$C_{IN} = 1\square C_g$$

allow stray and wiring capacitances

$$C_S = 4\square C_g \text{ (say)}$$

Then

$$\Delta\text{ transition delay} = 8R_s \times 5\square C_g = 40\tau$$

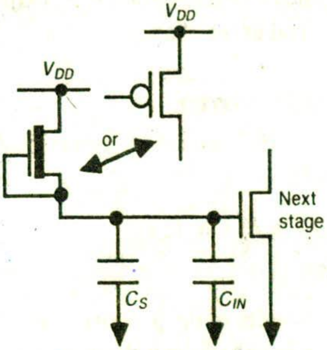


Figure 10–1 Circuit model for inverter driving an inverter on a ΔO/P transition

and

$$\nabla \text{ transition delay} = 1R_s \times 5\square C_g = 5\tau$$

For the alternative 8:1 inverter design discussed earlier, and allowing the same stray and wiring capacitances

$$\Delta \text{ transition delay} = 3R_s \times 7\square C_g = 21\tau$$

and

$$\nabla \text{ transition delay} = \tfrac{1}{3}R_s \times 7\square C_g = 2\tfrac{1}{3}\tau$$

Thus, it may be seen that a speed-up factor of about 2:1 in this case is bought at the expense of a 3:1 increase in power consumption but has the bonus of reducing area by a factor of 2:3. Similar considerations apply to the switching energy of CMOS circuits.

Therefore, as in most engineering situations, there are trade-offs to be made, and it is essential that the would-be designer have a good fundamental understanding of the discipline to be able to make sound decisions.

But remember, in the end there will always be limits imposed by the technology and some specifications will be impossible to meet.

10.1.1 Optimization of nMOS and CMOS inverters*

The approximate calculations presented here should be useful from a qualitative point of view and are intended to give the reader some appreciation of basic CMOS and nMOS circuit optimization problems.

* The authors are indebted to Professor K. S. Trivedi of Duke University for providing this material on inverter optimization.

For a more rigorous treatment of circuit optimization methods, refer to the articles cited at the end of the chapter.

10.1.1.1 The CMOS inverter

The area of a basic CMOS inverter is proportional to the total area occupied by the p- and n-devices.

$$A \propto (W_p L_p + W_n L_n)$$

where

W_p = width of the p-device
L_p = length of the p-device
W_n = width of the n-device
L_n = length of the n-device

Minimum area can be achieved by choosing minimum dimensions for W_p, L_p, W_n and L_n, that is

$$W_p = L_p = W_n = L_n = 2\lambda \text{ (minimum)}$$

Hence

$$\frac{W_p}{W_n} = 1$$

Switching power dissipation, P_{sd}, can be approximated by $C_L V_{DD}{}^2 f$ where

C_L = load capacitance at the inverter output
V_{DD} = power supply voltage
f = frequency of switching

For fixed V_{DD} and f, minimizing P_{sd} requires minimizing C_L which can be achieved by minimizing the area A since C_L is proportional to the gate areas comprising A.

Asymmetry in rise and fall times, t_r and t_f (transition times between 10% and 90% logic levels), can be equalized by using $\beta_n = \beta_p$. (Notice that t_r and t_f are proportional to the average resistance of the device which is approximately given by $\frac{2}{\beta V_{DD}}$ where $\beta = \beta_n$ or β_p.) This requires that

$$\frac{W_p}{L_p} = \left(\frac{\mu_n}{\mu_p}\right)\frac{W_n}{L_n}$$

to compensate for the lower hole mobility μ_p, compared to electron mobility μ_n

Assuming $L_p = L_n = 2\lambda$, $\frac{\mu_n}{\mu_p} \doteqdot 2$, we require $\frac{W_p}{W_n} \doteqdot 2$. This yields $t_r = t_f$.

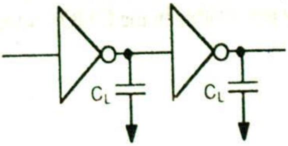


Figure 10–2 Inverter pair

Note that equalizing rise and fall times is not possible in nMOS or pseudo-nMOS inverters because of the ratio requirement.

Asymmetry in noise margins, NM_H and NM_L, can be equalized by choosing $\beta_n = \beta_p$ and hence $\frac{W_p}{W_n} \doteq 2$ for $L_p = L_n$. This yields $NM_H = NM_L$. (See Figure 10–4(b).)

Basic inverter pair delay — Consider a basic inverter pair shown in Figure 10–2 where C_L is the capacitive load driven by the two identical inverters, inverter pair delay $D(= t_r + t_f)$ is proportional to $(R_p + R_n)C_L$ where $R_p = 2/(\beta_p V_{DD})$ and $R_n = 2/(\beta_n V_{DD})$ are the average resistances of the p- and n-transistors respectively.

Also

$$C_L = C_E + (W_p L_p + W_n L_n)C_g$$

where

C_E = lumped parasitic capacitance
C_g = gate capacitance per unit area

Hence

$$D = D_0\left[\left(\frac{2}{\beta_p V_{DD}} + \frac{2}{\beta_n V_{DD}}\right)(C_E + (W_p L_p + W_n L_n)C_g)\right]$$

where D_0 is a constant of proportionality. Assuming $\frac{\mu_n}{\mu_p} \doteq 2$

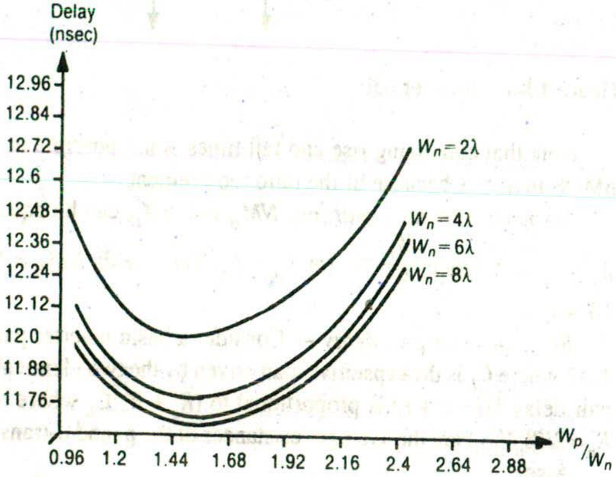
$$D = D_0\left[C_E\left(\frac{2L_p}{W_p} + \frac{L_n}{W_n}\right) + C_g\left(2L_p^{\,2} + 2L_p L_n \frac{W_n}{W_p} + L_p L_n \frac{W_p}{W_n} + L_n^{\,2}\right)\right]$$

Since D increases with L_n and L_p, for minimum D choose $L_n = L_p = 2\lambda$ (minimum). Minimizing D with respect to W_p yields a solution

$$W_p / W_n = \sqrt{2}\left[1 + \frac{C_E}{C_g L_n W_n}\right]^{1/2}$$

$$W_p / W_n \doteq \sqrt{2} \text{ for } C_E << C_g L_n W_n \text{ (normal case)}$$

However, D does not vary significantly with W_p/W_n in the range $1 \leqslant \frac{W_p}{W_n} \leqslant 2$ (see Figure 10–3). Hence simultaneous optimization of various parameters mentioned above seems to be easily achievable in the CMOS inverter, without greatly increasing the delay D.



Notes: $L_p = L_n = 2\lambda = 5\ \mu m$
Gate capacitance $C_g = 4 \times 10^{-4}\ pF/\mu m^2$
$C_\ell = 4 \times 10^{-3}\ pF$

Figure 10–3 Delay (nsec) vs. W_p/W_n for CMOS inverter

10.1.1.2 nMOS inverter

Let $Z_{p.u.}/Z_{p.d.} = \frac{L_{p.u.}W_{p.d.}}{W_{p.u.}L_{p.d.}} = k$ where the subscripts *p.u.* and *p.d.* refer to the pull-up and pull-down transistors respectively. Then area

$$A = A_0(L_{p.d.}\ W_{p.d.} + L_{p.u.}\ W_{p.u.})$$

$$= A_0\left(L_{p.d.}W_{p.d.} + kW_{p.u.}{}^2 \frac{L_{p.d.}}{W_{p.d.}} \right)$$

where A_0 is a constant of proportionality. For a fixed k, to achieve minimum A, we need $L_{p.d.} = W_{p.u.} = 2\lambda$. Minimizing A with respect to $W_{p.d.}$ yields a solution $W_{p.d.} \sqrt{k}W_{p.u.} = \sqrt{k}\,2\lambda$. Hence, using $Z_{p.u.}/Z_{p.d.} = k$, we obtain

$$L_{p.u.} = \sqrt{k}.L_{p.d.} = \sqrt{k}.2\lambda$$

This implies $Z_{p.u.} = \sqrt{k}$ and $Z_{p.d.} = 1/\sqrt{k}$. Giving

$$\text{Minimum area} = 8A_0\lambda^2\sqrt{k}$$

Static power dissipation, $P_d = P_0 \dfrac{V_{DD}{}^2}{(k+1)Z_{p.d.}}$, where P_0 is a constant of proportionality — for fixed k and V_{DD}, P_d is minimized by choosing as large a $Z_{p.d.}$ as possible. However, a large $Z_{p.d.}$ requires a large a $Z_{p.u.}$ ($Z_{p.u.} = kZ_{p.d.}$), and hence the delay D of the inverter pair increases. One has to choose the maximum $Z_{p.d.}$ possible for a given maximum allowed delay D.

If we use $Z_{p.d.} = 1$ with $L_{p.d.} = W_{p.d.} = 2\lambda$, and $Z_{p.u.} = k$ with $L_{p.u.} = 2k\lambda$ and $W_{p.u.} = 2\lambda$, we obtain

$$P_d = \frac{P_0 V_{DD}{}^2}{(k+1)}$$

$$A = 4A_0(k+1)\lambda^2$$

Inverter pair delay — Proceeding in a similar manner to the CMOS case

$$\begin{aligned} C_L &= C_E + C_g\, W_{p.d.}L_{p.d.} \\ D &= t_r + t_f = D_0(Z_{p.d.} + Z_{p.u.})C_L \\ &= D_0[Z_{p.d.}C_E(1+k) + C_g(1+k)L_{p.d.}{}^2] \end{aligned}$$

To minimize D:

1. Choose minimum $L_{p.d.} = 2\lambda$.
2. For maximum $W_{p.d.}$, choose $L_{p.u.} = 2\lambda$, as $W_{p.d.} = 2k\lambda \dfrac{W_{p.u.}}{L_{p.u.}}$ which yields $W_{p.d.} = kW_{p.u.}$

Choosing large $W_{p.d.}$ to minimize D increases A. Hence for a given area $A(= W_{p.d.}L_{p.d.} + W_{p.u.}L_{p.u.})$ with $L_{p.d.} = L_{p.u.} = 2\lambda$, we must have

$$W_{p.u.} = \frac{A}{2\lambda(k+1)} \qquad W_{p.d.} = \frac{kA}{2\lambda(k+1)}$$

With $W_{p.u.} = 2\lambda$, we have $W_{p.d.} = k2\lambda$. Hence $Z_{p.u.} = 1$ and $Z_{p.d.} = 1/k$ for minimum D.

$$\text{Minimum } D = D_0(1+k)\,(C_E/k + 4\lambda^2 C_g)$$

Table 10–1 shows the summary of optimization of the three parameters, D, A and P_d. Notice that the solution for minimum power dissipation also gives the lowest power delay product among the three designs.

Table 10–1 Optimum parameters for nMOS inverters

	$L_{p.d.}$	$W_{p.d.}$	$Z_{p.d.}$	$L_{p.u.}$	$W_{p.u.}$	$Z_{p.u.}$
Minimum *D*	2λ	$2k\lambda$	$1/k$	2λ	2λ	1
Minimum A	2λ	$2\lambda\sqrt{k}$	$1/\sqrt{k}$	$2\lambda\sqrt{k}$	2λ	$\sqrt{k}$
Minimum P_d	2λ	2λ	1	$2\lambda k$	2λ	k

	A/A_0	D/D_0	$P_d/(P_0V_{DD}^2)$
Minimum *D*	$4\lambda^2(k+1)$	$(1+k)\,(C_E/k+4\lambda^2C_g)$	$\frac{k}{k+1}$
Minimum *A*	$8\lambda^2\sqrt{k}$	$(1+k)\left(\frac{C_E}{\sqrt{k}}+4\lambda^2C_g\right)$	$\frac{\sqrt{k}}{(k+1)}$
Minimum P_d	$4\lambda^2(k+1)$	$(1+k)\,(C_E+4\lambda^2C_g)$	$\frac{1}{(k+1)}$

10.1.2 Noise margins

Noise margins have been mentioned in the preceding section and it is appropriate now to consider this factor in more detail.

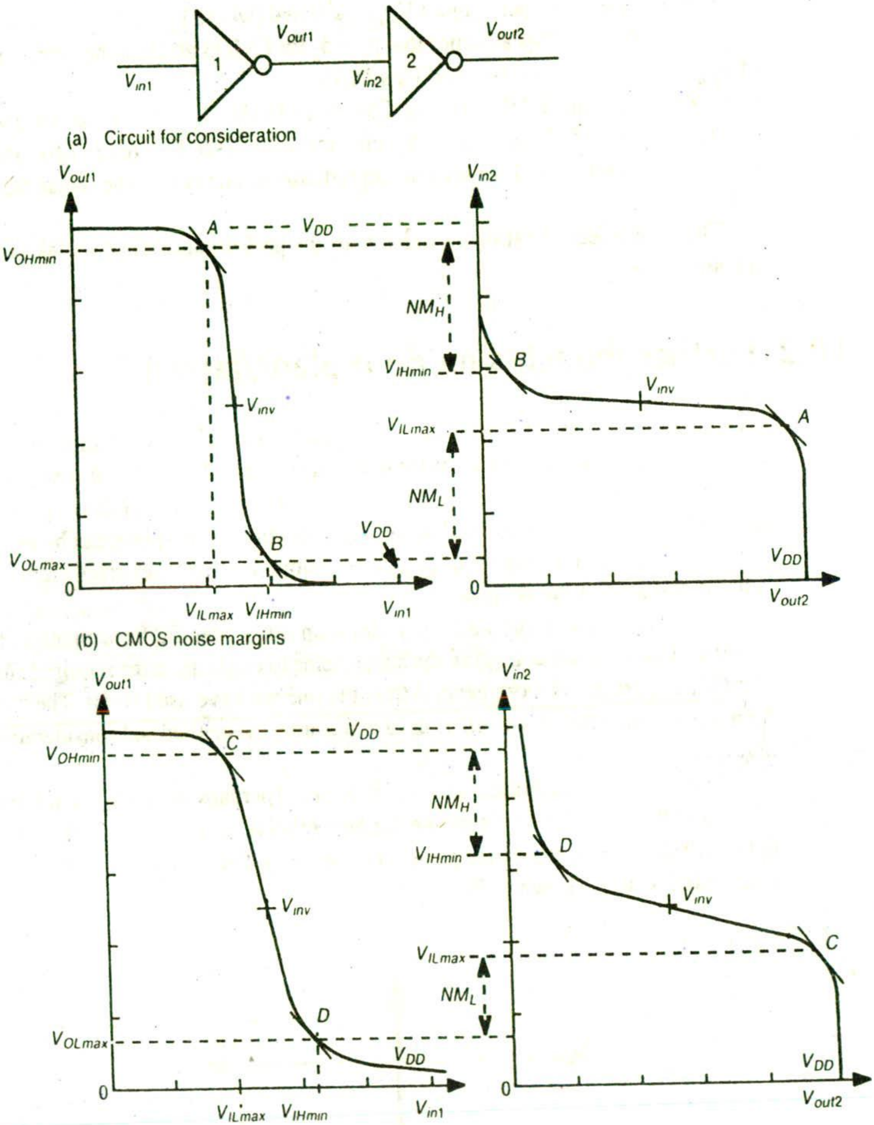
Noise margins are a measure of a logic circuit's tolerance of noise voltages in either of the two logic states; in other words, by how much the input voltage can change without disturbing the present logic output state. In order to examine this, it is convenient to consider a pair of inverters (nMOS or CMOS) and derive the noise margins for signals applied to the input of the second inverter, inverter 2, which is driven from the output of a similar inverter, inverter 1, as in Figure 10–4(a).

Referring now to Figure 10–4(b), we see the transfer characteristics (V_{out} vs. V_{in}) for a pair of CMOS inverters set out in such a way that the output voltage of inverter 1 is applied as the input voltage to inverter 2. By first considering the point at which output 1 starts to enter the transition region (the unity gain point *A*) and calling this voltage $V_{OH\,min}$ and then considering the input voltage level $V_{IH\,min}$ (point *B*) at which the transition of the output of inverter 2 commences, we are able to define the high level noise margin of inverter 2 as NM_H where

$$NM_H = V_{OHmin} - V_{IHmin} \text{ (a positive voltage)}$$

Similarly, a consideration of the low logic level conditions gives

$$NM_L = V_{OLmax} - V_{ILmax} \text{ (a negative voltage)}$$



(a) Circuit for consideration

(b) CMOS noise margins

(c) nMOS noise margins

Note: *A* and *B*, *C* and *D* are unity gain points.

Figure 10–4 Inverter noise margins

A similar approach will yield noise margins for the nMOS inverter as shown in Figure 10–4(c). It may be seen that generally the CMOS inverter will have better noise margins than the nMOS inverter, particularly for the low condition.

In both cases, symmetry about V_{inv} is assumed (where V_{inv} is the point at which $V_{out} = V_{in} = V_{DD}/2$). This assumes that $\beta_p = \beta_n$ for CMOS and that the correct ratio of $Z_{p.u.}$ to $Z_{p.d.}$ has been observed for nMOS.

Changes in the β_n / β_p ratio for CMOS or to the $Z_{p.u.}/Z_{p.d.}$ ratio for nMOS will result in a shift in the V_{out} vs. V_{in} characteristics (see Figures 2–7 for nMOS and 2–15 for CMOS) and consequent degradation of one or the other noise margin in each case.

Thus the effect of ratios on noise margins performance must be taken into account in design.

10.2 Further thoughts on floor plans/layout

In considering the layout of the four-bit data path used earlier as a design exercise, we could have waited until we knew the minimum size and disposition of connections to each functional block in order to finalize the floor plan. Indeed, this is a possible approach if communications will allow. Quite accurate floor plans can be set out at an early stage if a library of properly dimensioned and characterized elements/cells is available to the designer.

However, a better approach is to draw up quite specific floor plans at the outset and then design/configure the subsystems to conform to the required floor plan. This approach is more general than the one we have used so far. The same 4-bit processor (Figure 8–1) will be used to illustrate the method and considerations involved.

First (as before) determine an *overall strategy* (perhaps as suggested in Figure 10–5) and then use this to determine the best relative disposition of subsystems in light of data flow and control paths through the system. For the 4-bit data path, a suitable layout is shown in Figure 10–6.

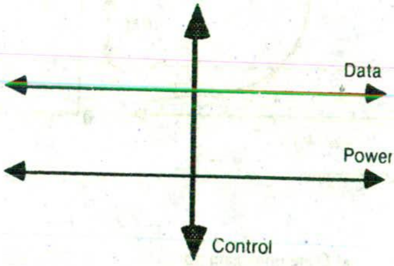


Figure 10–5 A communications strategy

When approached this way, a reasonably well thought-out floor plan can be developed before knowing any real detail of the subsystem/block areas. In the event, features of individual subsystems (Figures 7–8, 7–9 with 7–10, 8–11(b), and 9–15 with 9–17) will, in general, dominate the overall layout and other blocks may then be stretched and/or reconfigured as necessary to conform with the dominant features.

In order to do so it is essential to set out clearly the way in which data will flow on the buses. In this case:

1. Floating bus lines are envisaged.
2. All read and write operations are coincident with ϕ_1.
3. Bus A *connects the* I/O port to the register array and carries one operand (A_k) from the registers to the adder. It will also be used to carry the output of the shifter back to the register array (and I/O port). Bus A is therefore *bidirectional.*
4. Bus B connects the register array with the other input (B_k) of the adder and may also be used to carry the sum output (S_k) from the adder to the input of the shifter. Bus B is *unidirectional.*

Taking the subsystems of the 4-bit data path example (Figures 7–9 with 7–10, 8–11(b), and 9–15 with 9–18), one of the main features is the bus spacing, that is, the spacing between buses A_n and B_n and between A_n and A_{n+1}, etc., and close examination of the interconnection of designs pursued in this text will reveal that the bus spacings of the adder subsystem dominate those of the other subsystems.

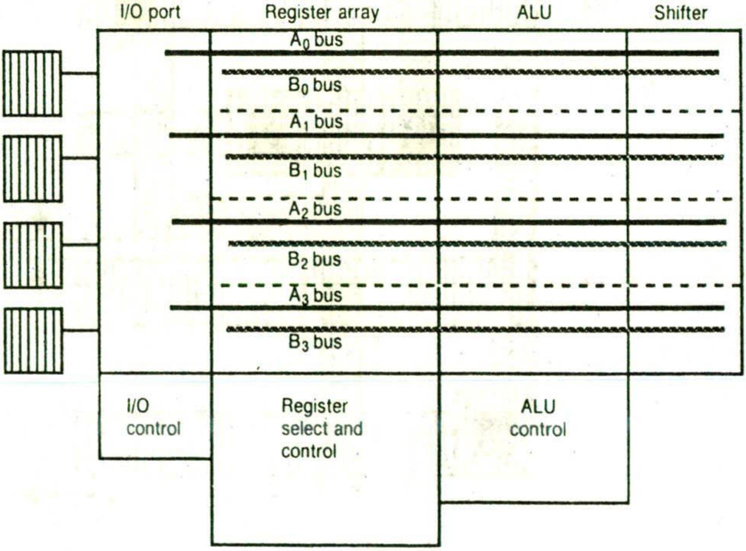
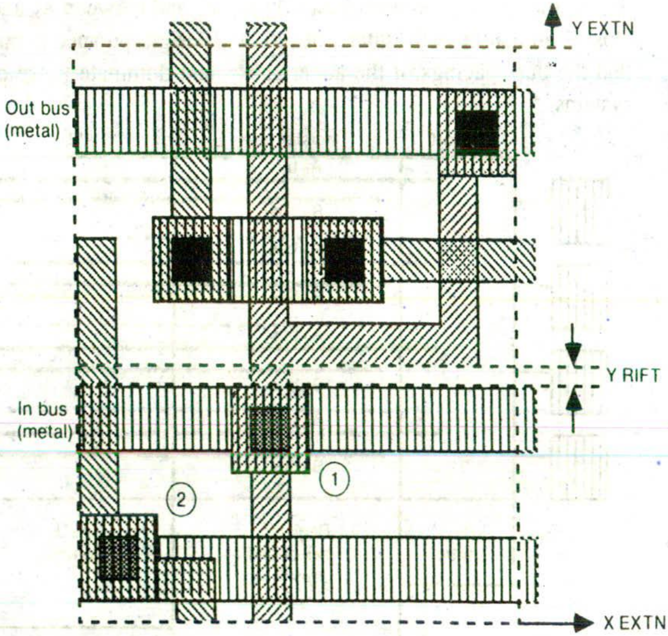


Figure 10–6 Possible floor plan for 4-bit processor

Rearrangements consequent on these considerations affect the barrel shifter (Figures 7–8 to 7–10) in particular. It is necessary to interchange the relative position of the *In* and *Out* bus lines and also make the cell stretchable to match the height of the dominant (adder) block and its bus spacing. Also, to mate with the bus structures of the other blocks, the *In* and *Out* bus lines should be in metal rather than polysilicon and diffusion, as used in our original design of Figures 7–8, 7–9 and 7–10.

The way in which this may be done is indicated in the revised standard cell layout (Figure 10–7); it is necessary to allow for rifts and extensions *and* to cope with optional features which result from the four versions (owing to optional contacts) of the standard cell required, thus ensuring generality.

The concept of the use of a Y RIFT which is extendable from 0λ minimum upward and X EXTN and Y EXTN which are extensions of the cell from 0λ upward make the barrel shifter configurable to match most bus dispositions. Note that rifts and extensions should be placed where they cut a minimum amount of simple geometry; for example, Y RIFT involves the stretching of two wires — one in polysilicon and the other in diffusion. Once such a degree of freedom is available, subsystems may be mated with a smooth flow-through of buses as suggested in Figure 10–8, which, for simplicity, shows the mask layout for an nMOS adder and a shifter which is on the right.



Note: 1 and 2 are optional contacts.

Figure 10–7 Standard cell for barrel shifter

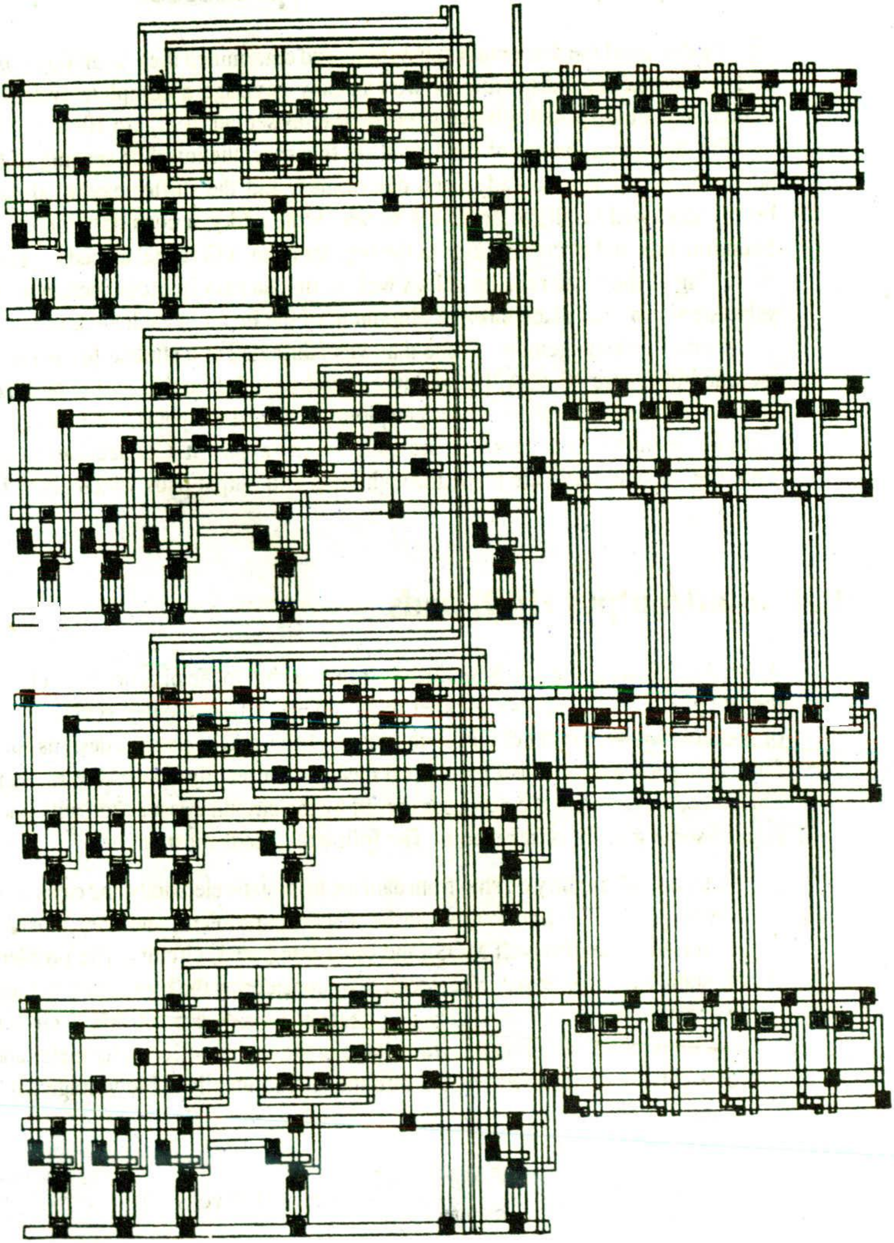


Figure 10–8 A possible interconnection of the adder and shifter subsystems

10.3 Floor plan layout of the 4-bit processor

Having designed the three main subsystems and determined their bounding boxes and interconnection dispositions, we can now envisage a complete system in which they are disposed relative to each other as set out in Figure 10–6.

The dominant feature of the layout (in this case, the interbus spacing of the adder circuit) having already been determined, and the shifter having already been redesigned to allow stretching to match the adder, a consideration of the bounding box and of connections to the register array will reveal a need for some stretching of the basic register cell as well so that an easy interconnection of the subsystem can take place through alignment of the buses in each subunit.

A possible arrangement — one that was fabricated as a student project — is included in Figure 10–9. Although layer encoding is lost in this particular black and white reproduction from a color-pen plotter of the mask layouts, the architecture and placement of the subsystems are quite readily apparent. Connections to and from the outside world are made through input and output pads which allow for bonding.

10.4 Input/output (I/O) pads

As well as allowing the bonding of leads from the chip to the pins on the package, the I/O pads cover a number of other requirements. Consequently, several types of pad are required. It is not within the scope of this text to present designs for a family of pads and, in most cases, pad designs are readily obtainable as basic library cells. However, the purposes served by the circuitry associated with pads require some general observations. The following needs must be met:

1. Protection of circuitry on chip from damage from static electricity and capacitive discharge (ESD) effects: this can be a serious problem, and care must be exercised in handling all MOS (and other integrated) circuits. The problem of 'static zap' may be put in perspective by considering the breakdown voltage of the thin oxide between gate and channel in, say, a 5 μm MOS circuit. Silicon dioxide has a breakdown voltage in the region of 10^9 volts/meter and for a gate oxide thickness of 0.1 μm, the maximum allowable voltage gate/channel is

$$V_{gc\,max} < \frac{10^9 \text{ volts}}{\text{meter}} \times \frac{0.1}{10^6} \text{ meter} = 100 \text{ volts}$$

This may sound generous in light of rail voltages of the order of 5 to 10 volts, but relatively high voltages are readily generated on one's person or on tools and handling equipment. Quite innocent pastimes, such as walking across

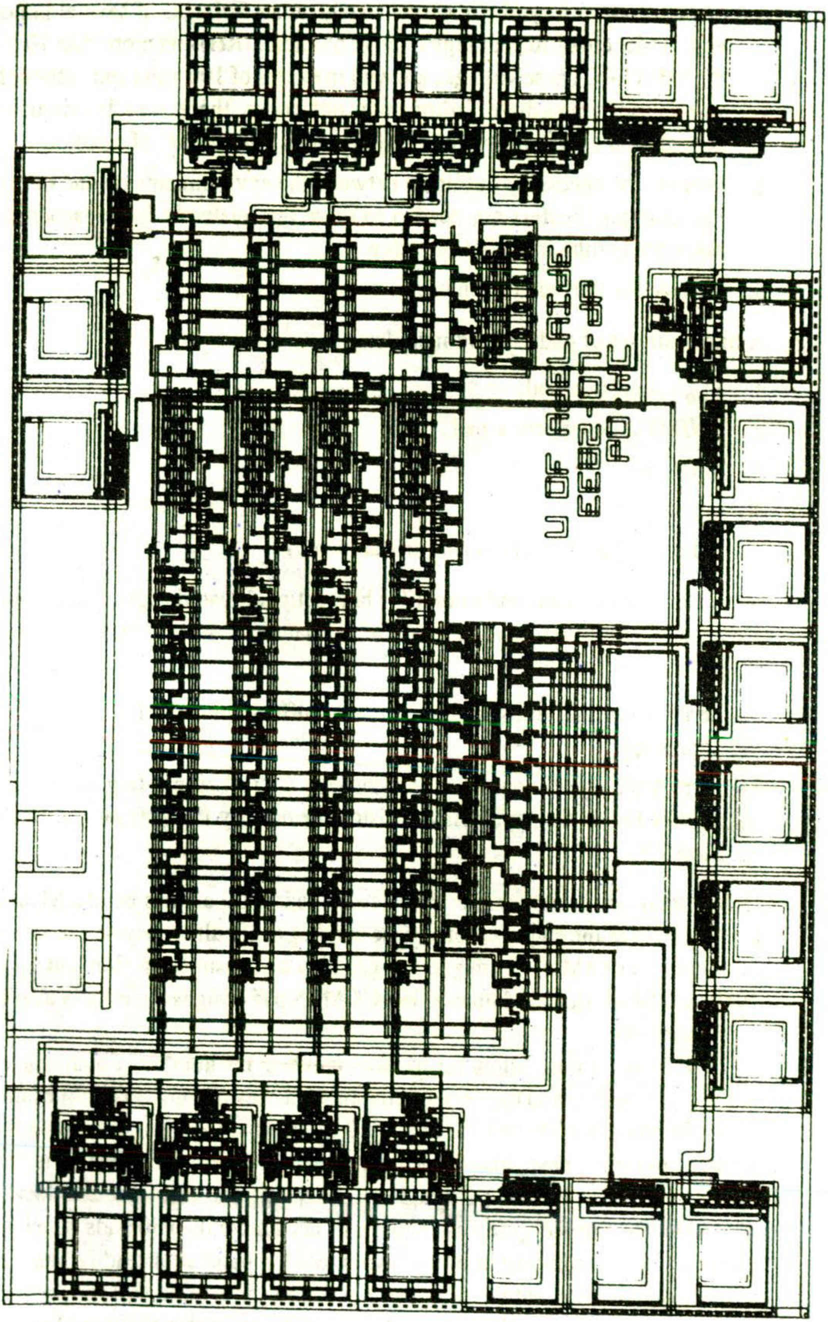


Figure 10–9 Complete layout of 4-bit data path multiproject chip

a vinyl floor or a synthetic carpet, can generate voltages of several hundred volts under conditions of high relative humidity (RH) and more than 10 kV if the RH is low. These voltages are well in excess of 100 volts and, although in some cases immediate failure may not occur, there may be significant degradation of reliability and/or life through 'wounding' of circuits.

2. Provide the necessary buffering between the environments on and off chip. For example, buffers are needed to drive the relatively large capacitances associated with circuits off the chip.
3. Provide for the connection of power supply rails.

A minimum set of pads should include:

1. V_{DD} connection pad;
2. *GND* (V_{SS}) connection pad;
3. input pad;
4. output pad;
5. bidirectional I/O pad (usually tristate logic).

In all cases when input and output (or bidirectional) pad designs from a library are used, the designer *must* be aware of the nature of the circuitry embodied in the pad design, that is:

1. be aware of the ratios/size of inverters/buffers onto which output lines are connected;
2. be aware of how input lines pass through the pad circuit (e.g. are the input signals fed in through pass transistors or do they come from inverter-like stages?).

Unless there are exceptional circumstances pads must always be placed around the *periphery* of the chip area, otherwise bonding diffficulties may be encountered. A sample set of nMOS 5 μm pad designs may be consulted in Hon and Sequin, 1980, and Newkirk and Mathews, 1984. CMOS pad designs are usually available from fabricators.

The designer must allow for the way in which the number of available pads quickly get used up and the very significant area they occupy. Take, for example, a simple processor of the type discussed in this text together with some RAM memory to form a basic microprocessor circuit. A typical arrangement is shown in Figure 10–10. Allowing for eight memory address lines (i.e. 256 locations of RAM), the complete chip as shown will need more than 30 pads which must therefore be accommodated in the layout. Such a number is readily bonded to, say, a 40-pin header, but the designer must also bear in mind that the package to be used will impose an ultimate limitation on the allowable number of pads.

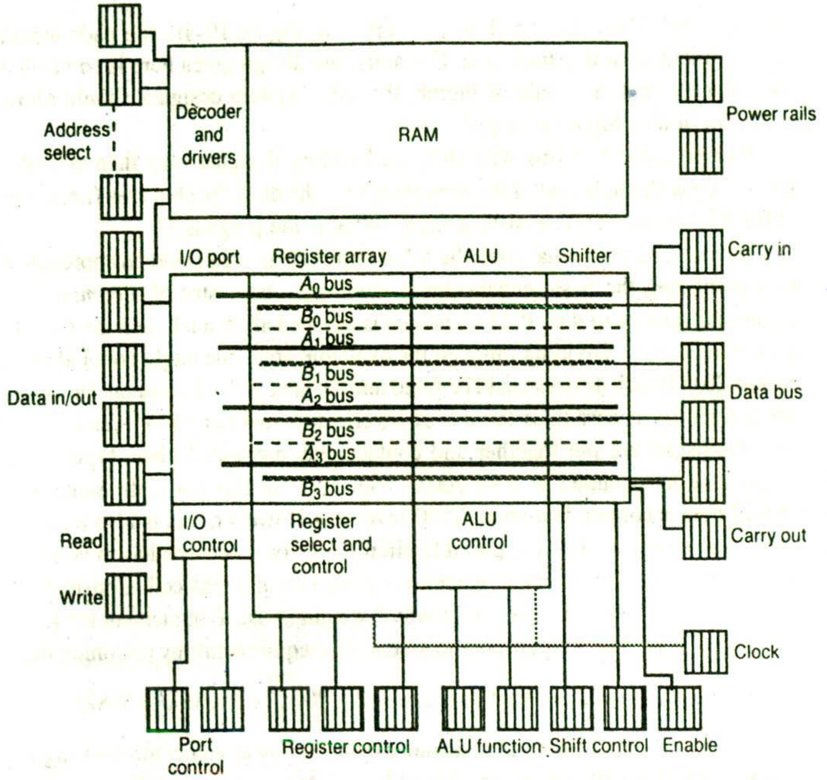


Figure 10–10 4-bit processor — pad utilisation

10.5 'Real estate'

Give me land, lots of land . . .

(words of a popular song of yesteryear)

One of the most common mistakes among beginners is to assume that phenomenal amounts of circuitry occupy very little area on the chip (VLSI = very little silicon indeed?). In order to correct such over-optimism it is necessary to consider only one or two of the practical factors which arise in system design.

For example, consider the area required by the I/O pads for the floor plan of Figure 10–10. The connections shown require 33 pads and typical standard 5 μm pad layouts require an area of 105λ by 100λ to 200λ (depending on the nature of the pad). An average pad then occupies some 105λ by 150λ, say, that is, an area of 15,750λ^2. Thus the area required for 33 pads is over 500,000λ^2. To put this into perspective, the average area allowance for each student project for a multiproject chip (MPC) design was typically somewhere in the region of 1000λ × 1000λ, that is, $10^6\lambda^2$. Thus, for the floor plan given in Figure 10–10, the pads would occupy one-half of this total area. Certainly, the design given here is somewhat pad-intensive but, as a rule of thumb, the small system designer should allow *one-third* of the chip area for pads.

Having come to terms with this, the budding designer may then consider what to do with the layout of the remaining two-thirds of the chip area (i.e. about $700{,}000\lambda^2$ for an example MPC design). What is the prognosis?

An assessment of what could be fitted into such an area could be approached by considering the basic enhancement mode pass transistor of *minimum size* occupying an area of $4\lambda^2$. If 2λ clearance is allowed all around, then the *on chip area* will be $36\lambda^2$. Dividing this into the available area, one might conclude that almost 20,000 such devices could be fitted into the area under discussion. However, MOS circuitry necessitates the use of inverters or inverter-like circuits. When two transistors are put together and contacts etc. are added, then, typically, a single inverter occupies at least $200\lambda^2$. Viewed from this point, the same area should thus accommodate about 3500 inverters. However, this is also an over-optimistic assessment of the possible circuit density, since one has to consider the significant effect of interconnections even within a leaf-cell. Consider the simple memory cell of Figure 10–11 which we might use to implement the RAM of Figure 10–10. The temptation is to assess area requirements by reasoning thus:

two inverters + three pass transistors = $2 \times 200\lambda^2 + 3 \times 36\lambda^2 = 508\lambda^2$.

However, when design rule clearances, buses, power and control wiring are allowed for, this cell can occupy $1500\lambda^2$ or more (i.e. a factor of 3:1 over the 'simple' estimates).

Now, consider the available area on the floor plan and further assume that about half this area (i.e. approximately $350{,}000\lambda^2$) is to be devoted to the RAM. This area will allow no more than 256 bits of storage elements, as in Figure 10–11, and if each RAM location must hold a 4-bit word, then the designer can be no more ambitious than a 64-word RAM. The running of extra bus lines,

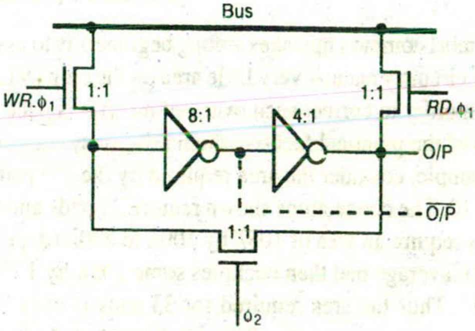


Figure 10–11 Pseudo-static memory cell

the register array, will further substantially increase the area occupied by each memory cell.

10.6 Further thoughts on system delays

10.6.1 Buses

> *He thought he saw [an operand],*
> *descending from a bus,*
> *he looked again and saw it was*
> *a hippopotamus.*
>
> (With apologies to Lewis Carroll)

The use of bus lines is a convenient concept in distributing data and control through a system. However, it is easy to lose sight of what is *really* happening and bus-derived signals tend not to be what were expected.

Bidirectional buses are convenient but conflicts must be avoided since data cannot flow in both directions at once. Clearly, in our data path design, the sum S_k must be stored and then subsequently read onto the bus, since it becomes obvious that two buses cannot carry two input operands and the sum simultaneously. A significant problem which is often underestimated is that of speed restrictions imposed by the capacitive load presented by long bus lines.

The largest capacitance (for a typical bus system) is contributed by C_{BUS} (the bus wiring capacitance), and for small chips with, say, a 1000λ long bus this can be as high as 0.75 pF for a metal layer bus in 5 μm technology. In total, then, the bus and associated circuitry for the system being considered could contribute a capacitive load of about 0.8 pF, which may be *driven* through pull-up (typically 20 to 40 kΩ 'on' resistance) and pull-down (typically 10 kΩ 'on' resistance) transistors and through at least one pass transistor or transmission gate in the series.

Therefore, sufficient time must be allowed to charge the total bus capacitance during, say, ϕ_1 of the clock. In the data path system considered here, the time required for the total bus capacitance to charge to an appropriate level (to, say, >90% of V_{DD}) is in the region of 100 nsec. Thus, it may be seen that equal ϕ_1 and ϕ_2 clock periods would result in an upper clock frequency limitation for the processor due to bus loading alone of 5 MHz. This frequency can be increased by using asymmetric ϕ_1 and ϕ_2 periods or by using BiCMOS drivers.

10.6.2 Control paths, selectors, and decoders

A basic operation of a data path is to add together the numbers stored in two registers to produce a sum and a carry at the 'carry out' pad (for cascading, etc.).

In terms of *delays* involved, and in the context of the 5 μm system considered here, the following delay mechanisms are encountered during this process:

1. *Select register* and open pass transistors (or transmission gates) to connect cells to bus. For a particular design, the select logic and associated drivers might have the equivalent circuit as shown in Figure 10–12.

   The overall *delay* of this arrangement may be assessed in terms of τ (where τ is the time constant of $1\square C_g$ charging through a minimum-size n-type pass transistor).

Element(s) contributing	*Delay*
Input pad	30τ (typical)
Three pass transistors ($n^2\tau$) = 9τ	9τ
Driver inverter pair ($\Delta A \rightarrow \nabla B \rightarrow \Delta C$) (Assuming $4\square C_g$ load at C)	34τ
Sum of delays (select register)	= 73τ

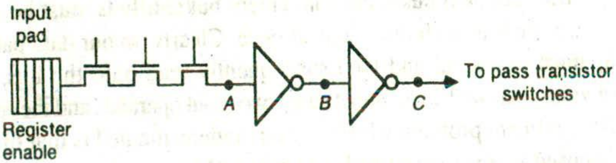


Figure10–12 Register select circuit

2. *Data propagation along bus* — This has already been calculated as 100 nsec.
3. *Carry chain delay* — The longest delay in the particular design of adder used is that of forming the 'carry out' which, in effect, propagates through all bits of the adder and then through the outlet pad as shown in Figure 10–13. Timing simulator results for a 2-bit arrangement is given as Figure10–14. It will be seen that, although the ΔC and ∇*C* delays are slightly different, an average delay of 65 nsec is a fair assumption for the 2-bit system simulated. We may also deduce the delay per bit (≑ 20 nsec) from the simulation. Overall then, a 5 μm 4-bit ripple-carry adder could be expected to have a delay of about 105 nsec.

Thus, the overall delay = select registers + bus delays + carry chain delays = (73τ) + 100 nsec + 105 nsec.

For τ = 0.2 nsec

$$\textit{Sum of delays} = 14.6 + 100 + 105 \doteqdot 220 \text{ nsec}$$

Thus, ϕ_1 of the clock must have a duration longer than 220 nsec.

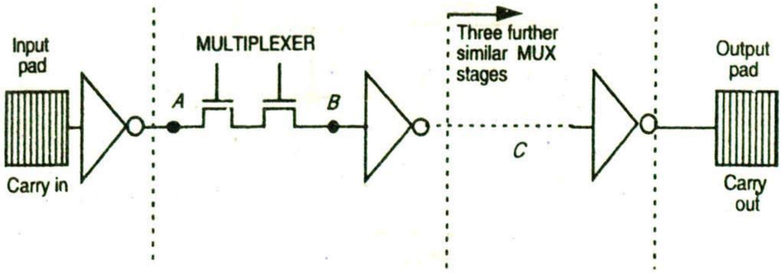


Figure 10–13 Possible carry chain circuit

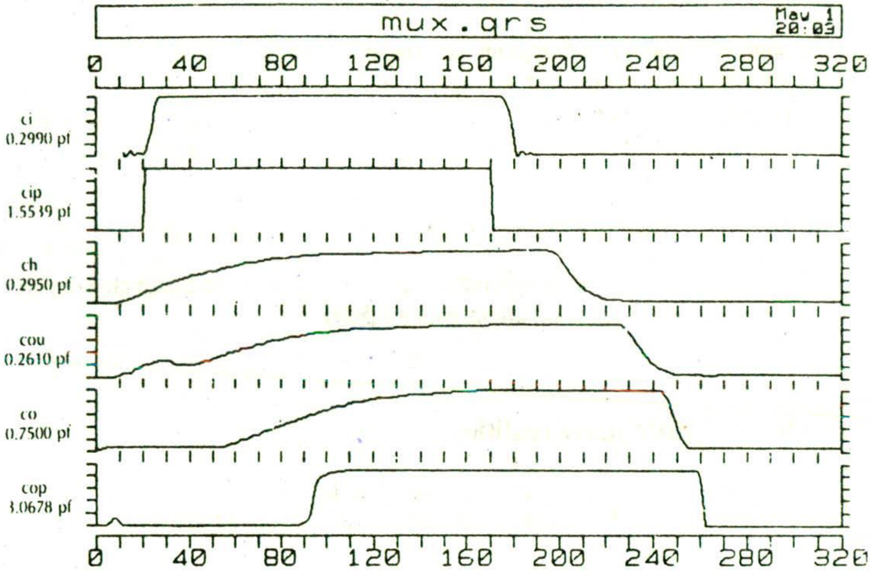


Figure 10–14 Timing simulation result for a 2-bit version of the multiplexer-based adder

10.6.3 Use of an asymmetric two-phase clock

10.6.3.1 Clock period ϕ_2

In many systems, ϕ_2 of the clock is used only to refresh memory/register cells such as that shown in Figure 10–15. From the figure it can be seen that ϕ_2 has to be long enough in duration to allow C_{in} to charge through the pull-up resistance of the second inverter and through the feedback circuit — which may be in the region of 35 kΩ. If time is allowed for C_{in} to charge to within < 10% of its final value, then refresh time ÷ 2.5 × 10τ = 25τ which, for the 5 μm system being

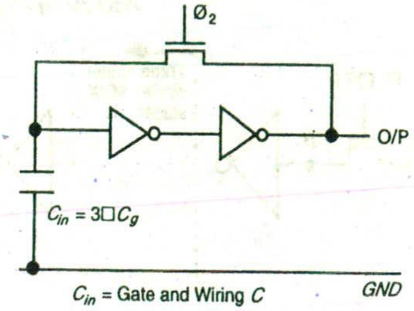


Figure 10–15 Memory cell refresh

evaluated, equates to a minimum 'on' time of 5 nsec for ϕ_2. However, ϕ_2 signals must also propagate through wiring etc., and finite rise- and fall-times must be allowed for so that some extra time should be allowed for the ϕ_2 'on' period. For safety allow, say, 50τ (i.e. 10 nsec) for the ϕ_2 'on' period and also allow 10 nsec underlap between the two phases. Thus

$$\text{total clock period} = 220 + 10 + 10 + 10 = 250 \text{ nsec}$$

Therefore, *in theory*, our simple modeling suggests that the data path chip design should operate on add instructions with a 4-MHz clock.

10.6.4 More nasty realities

Life wasn't meant to be easy.
Malcolm Fraser (former Prime Minister of Australia)

The simple calculations made on the particular processor design seem to indicate that a clock frequency in the region of 4 MHz would be possible. In practice, this may not happen. Why is this so? To answer this it is necessary to consider practical as well as theoretical realities.

From the theoretical aspect, our predictions have been made on very approximate parameter values and on very simple circuit models. We have also mostly ignored the quite significant effects of peripheral capacitance in diffusion regions and fringing field capacitances around conductors on the chip.

Although τ was assumed to be in the range 0.1 to 0.3 nsec for 5 μm technology, the value of τ measured for the fabricated chip may not be within this range.

In fact, the value of τ measured on some 5 μm MPC circuits fabricated and tested for this project was in the region of 0.6 nsec.

The designer, therefore, must be aware of, and allow for, all the significantly nasty realities affecting the performance of the design, and have a good knowledge of the parameters of the processing plant or fabrication line where that design is to be implemented in silicon.

There are two main points of difference between expectations and realization which characterize many of the designs of beginners. They are:

1. The system being designed occupies far more area than was anticipated.
2. The system when manufactured is slower than the designer had estimated.

However, if the first few designs are carefully carried out, are not over-ambitious, and are properly checked for logical and design rule errors, the beginner is usually pleasantly surprised by the fact that the system does in fact function, albeit not quite as fast as intended.

10.7 Ground rules for successful design

This section is intended to provide a convenient focus for design information. From our considerations of system design up to this point a number of ground rules; aspects of philosophy, and some basic data have emerged which help to ease the design process and ensure success. These and one or two other considerations which are important (but have not as yet been formally set out in the text) are presented or referenced here under 19 subheadings.

1. The ratio *rules* (Chapter 2)
   (a) for nMOS inverters and inverter-like stages

   $Z_{p.u.}$:$Z_{p.d.}$ ratio = 4:1 when driven from another inverter

   $Z_{p.u.}$:$Z_{p.d.}$ ratio = 8:1 when driven through one or more pass transistor(s).

   where

   $$Z = L / W \text{ for the channel in question}$$

   (b) for CMOS, a 1:1 ratio is normally used to minimize area, but for pseudo-nMOS inverters etc., a ratio $Z_{p.u.}$:$Z_{p.d.}$ = 3:1 is required.
2. *Design rules* (Chapter 3). Never bend the rules.
3. *Typical parameters for* 5 μm (λ = 2.5 μm), 2 μm *and* 1.2 μm *featurę size MOS* (Chapter 4) including guidelines for signal interconnections.

Table 10–2 (Table 4–1) Typical sheet resistances R_s of MOS layers for 5 µm, and Orbit 2 µm and 1.2 µm technologies

Layer	*R_s ohm per square*	
	5 µm	*Orbit*
Metal	0.03	0.04
Diffusion (n-type or n-active)	10–50*	20–45*
Silicide	2–4	–
Polysilicon	15–100	15–30
n-transistor channel	10^4†	2×10^4†
p-transistor channel	2.5×10^4†	4.5×10^4†

Note: In some processes a silicide layer is used in place of polysilicon.

*Times 2.5 for p-type.

† These values are approximations only. Resistances may be calculated from a knowledge of V_{ds} and the expressions for I_{ds} given earlier.

Table 10–3 (Table 4–2) Typical area capacitance values

Capacitance	*Value in pF $\times 10^{-4}$/µm^2 (relative values in brackets)*		
	5 µm	*2 µm*	*1.2 µm*
Gate to channel	4 (1.0)	8 (1.0)	16 (1.0)
Diffusion (active)	1 (0.25)	1.75 (0.22)	3.75 (0.23)
Polysilicon* to substrate	0.4 (0.1)	0.6 (0.075)	0.6 (0.038)
Metal 1 to substrate	0.3 (0.075)	0.33 (0.04)	0.33 (0.02)
Metal 2 to substrate	0.2 (0.05)	0.17 (0.02)	0.17 (0.01)
Metal 2 to metal 1	0.4 (0.1)	0.5 (0.06)	0.5 (0.03)
Metal 2 to polysilicon	0.3 (0.075)	0.3 (0.038)	0.3 (0.018)

Note: Relative value = specified value/gate to channel value for that technology.

*Poly 1 and Poly 2 are similar (also silicides where used).

Table 10–4 (Table 4–3) Typical values for diffusion capacitances

Diffusion capacitance	*Typical values*		
	5 µm	*2 µm*	*1.2 µm*
Area C (C_{area}) (as in Table 42)	1.0×10^{-4} pF/µm^2	1.75×10^{-4} pF/µm^2	3.75×10^{-4} pF/µm^2
Periphery (C_{periph})	8.0×10^{-4} pF/µm	negligible*	negligible*

* Assuming implanted regions of negligible depth.

In order to calculate the total diffusion capacitance we must add the contributions of area and peripheral components.

$$C_{total} = C_{area} + C_{periph.}$$

Standard unit of capacitance $\square C_g$

$1\square C_g$ is defined as the gate-to-channel capacitance of a MOS transistor having $W = L =$ feature size, that is, a 'standard' or 'feature size' square (the concept of $\square C_g$, originated by VTI (USA), has been adapted here).

$\square C_g$ may be evaluated for any MOS process. For example, for 5 μm MOS circuits:

$$\text{standard value } \square C_g = .01 \text{ pF}$$

or, for 2 μm MOS circuits (Orbit) :

$$\text{standard value } \square C_g = .0032 \text{ pF}$$

and, for 1.2 μm MOS circuits(Orbit):

$$\text{standard value } \square C_g = .0023 \text{ pF}$$

The delay unit τ

We have developed the concept of sheet resistance R_s and standard gate capacitance unit $\square C_g$. If we consider the case of one standard (feature size square) gate area capacitance being charged through one feature size square of n channel resistance (i.e. through R_s for an nMOS pass transistor channel), we have:

$$\text{time constant } \tau = lR_s \text{ (n channel)} \times 1\square C_g \text{ seconds}$$

This can be evaluated for any technology and for 5 μm technology

$$\text{theoretical } \tau = 0.1 \text{ nsec.}$$

and for 2 μm (Orbit) technology

$$\text{theoretical } \tau = 0.064 \text{ nsec.}$$

and for 1.2 μm (Orbit) technology

$$\text{theoretical } \tau = 0.046 \text{ nsec.}$$

However, in practice, circuit wiring and parasitic capacitances must be allowed for so that the figure taken for τ is often increased by a factor of two or three.

Taking account of resistances and total capacitances we may set out practical guidelines on signal path lengths as in the following table (10–5), noting that the figures given are conservative but safe.

4. *Inverter pair delay*

In general terms, the delay through a pair of similar nMOS inverters is

$$T_d = (1 + Z_{p.u.}/Z_{p.d.})\tau$$

and for a minimum size CMOS complementary inverter pair

$$T_d = 7\tau$$

Table 10–5 (as for Table 4–4) Electrical rules

Layer	*Maximum length of communication wire*		
	lambda-based (5 μm)	*μm-based (2 μm)*	*μm-based (1.2 μm)*
Metal	chip wide	chip wide	chip wide
Silicicide	2,000λ	n.a.	n.a.
Polysilicon	200λ	400 μm	250 μm
Diffusion (active)	20λ*	100 μm	60 μm

* Taking account of peripheral and area capacitances. n.a. not applicable.

5. *Cascaded inverters for driving capacitive load (C_L)*
The approach is to use N cascaded inverters, each one of which is larger than the preceding stage by a width factor f.

It has been shown that the number 'N' of stages required is given by

$$N = \frac{ln(y)}{ln(f)}$$

where

$$y = \frac{C_L}{\square C_g}$$

It can also be shown that total delay is minimized if f assumes the value e (base of natural logarithms); that is, each stage should be approximately 2.7* times wider than its predecessor. This applies to CMOS as well as nMOS inverters. See Chapter 4 for more details.

* *Note*: Usually $f = 3$ will do since the curve is quite flat near the minimum.

6. *Propagation delay through cascaded pass transistors or transmission gates* (Chapter 4)

$$T_d = n^2 rc\,(\tau)$$

where

n = number in series

r = relative series resistance per transistor or per transmission gate in terms of R_s

c = relative capacitance gate to channel per transistor or per transmission gate in terms of $\square C_g$.

Normally, no more than four pass transistors or transmission gates should be connected in series without buffering.

7. *Factors influencing choice of layer for wiring* (Chapter 4)

Table 10–6 (Table 4–5) Choice of layers

	Relative		
Layer	*R*	*C*	*Comments*
Metal	Low	Low	Good current capability without large voltage drop ... use for power distribution and global signals.
Silicide*	Low	Moderate	Modest RC product. Reasonably long wires are possible. Silicide is used in place of polysilicon in some nMOS processes.
Polysilicon	High	Moderate	RC product is moderate; high IR drop.
Diffusion	Moderate	High	Moderate IR drop but high C. Hence hard to drive.

Note: V_{DD} and V_{SS} (or *GND*) rails must always be run in metal, except for very short 'duck unders' where crossovers are unavoidable.

* Not often available — depending on process line.

8. *Subsystem/leaf-cell design guidelines* (Chapter 6)
   - (a) Define the requirements properly and carefully.
   - (b) Consider communication paths most carefully in order to develop sensible placing of subsystems and leaf-cells.
   - (c) Draw a floor plan (alternating with (b) as necessary).
   - (d) Aim for regular structures so that design is largely a matter of replication.
   - (e) Draw stick diagrams for basic cells, leaf-cells, and/or subsystems or enter the design in symbolic form.
   - (f) Convert to a mask level layout.
   - (g) Carefully and thoroughly check each mask layout for design rule errors and simulate circuit or logical operation. Correct as necessary, *rechecking* as corrections are made.

9. *Restrictions associated with MOS pass transistors and transmission gates* (Chapter 6)
   - (a) No more than four in series without buffering (see Point 6).
   - (b) No pass transistor gate must be driven from the output of one or more pass transistors, since logic 1 levels are degraded by threshold voltage V_{tp} (where V_{tp} can be as high as 0.3 V_{DD}).

(c) When designing switch logic networks of pass transistors or transmission gates, care must be taken to deliberately implement *both* the logic 1 and logic 0 output conditions.

Note: An *if, then, else* approach to specifying requirements will help to make sure that this is done.

10. *Storage of logic levels on the gate capacitance of transistors*

(a) Gate/channel capacitance is suitable for storing a bit, but care must be taken to allow for the finite decay time (about 0.25 msec at room temperature).

(b) It is quite allowable to construct pass transistors, etc. *under* metal layers to save space. This is often convenient and is used, for example, in some multiplexer layouts, but care must be taken with *overlying* metal wires where gate/channel capacitance is used for bit storage.

Consider Figure 10–16(a). Three such instances are illustrated here, all of which lie under metal wires. Two of these cases, T_1 and T_3, will operate satisfactorily, since for T_1 the metal wire is actually connected to the gate and for T_3 the metal wire is at a fixed, unvarying potential (that is, V_{DD} in this case). However, T_2 gate region lies under a metal bus which has no connection with the gate of T_2. If a bit is stored on T_2 gate by momentarily connecting *Control A* to the required level, then the bit will be stored but can be disturbed or destroyed by variation of the voltage on the overlying bus, as Figure 10–16(b) reveals.

(c) Restrictions also apply to logic level storage on the input capacitance of a *Nand* gate except for the input *nearest* the *GND* or V_{SS} rail. Conditions are indicated in Figure 10–17.

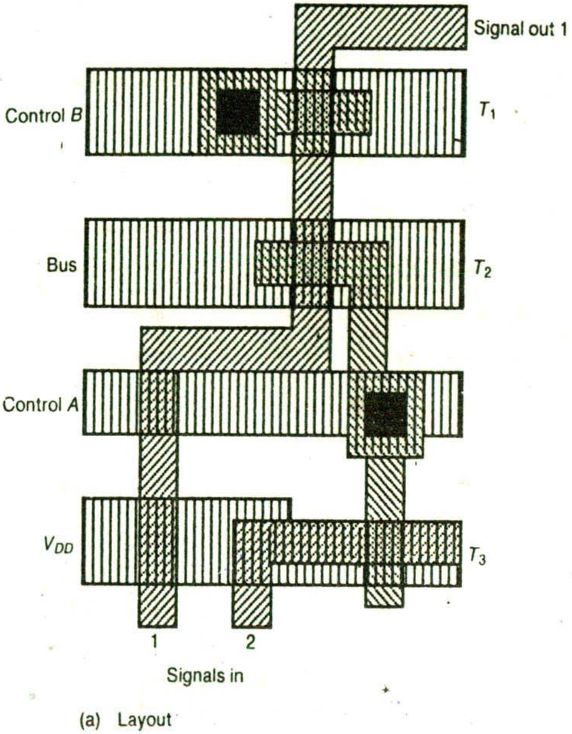
11. *Enhanced clocking*

One of the basic limitations on the use of simple MOS pass transistors (see Point 9 above) is the degradation of logic 1 levels by V_{tp} and the consequent inability of one pass transistor to drive the gate of a second (or more) pass transistor. This is particularly bothersome in clocking networks and a solution to this problem is to run all clock lines at a voltage level above V_{DD} as shown in Figure 10–18.

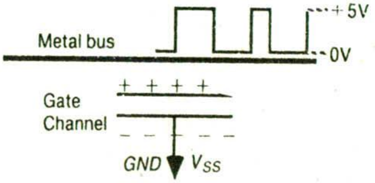
Note that the signal propagated through T_1 is V_{DD}, while that propagated through T_2 is $V_{DD} - V_{tp}$.

12. *The maximum allowable current density* in aluminum wires is 1 mA/μm^2. Otherwise, metal migration may occur (Chapter 6). Current density must be particularly carefully considered if the circuit is to be scaled down.

13. *Scaling effects:* see Chapter 5.



(a) Layout



(b) Circuit model

Figure 10–16 Pass transistors under metal wires

14. *System design process* (Chapter 7 — refer also to point 8 in this section)
    (a) Set out a specification together with an architectural block diagram.
    (b) Suitably partition the architecture into subsystems that are, as far as possible, self-contained and give interconnections that are as simple as possible.

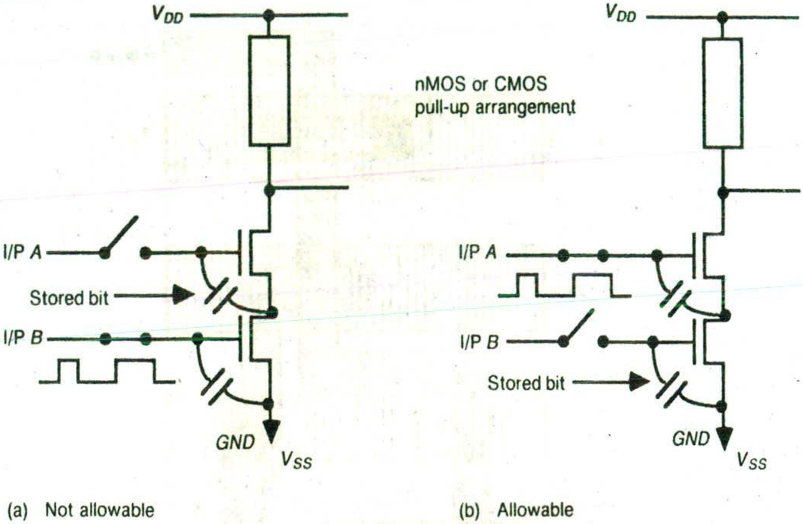


Figure 10–17 Storage nodes in gate arrangements

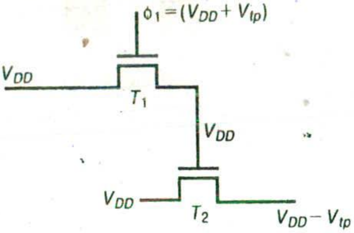


Figure 10–18 Enhanced clocking

(c) Set out a tentative floor plan showing the proposed relative physical disposition of subsystems on the chip.

(d) Determine interconnection strategy.

(e) Revise (b), (c), and (d) interactively as necessary.

(f) Choose layers on which to run buses and main control signals.

(g) Take each subsystem in turn and conceive a *regular* architecture to *conform* to the strategy set out in (d). Set out circuit and/or logic diagrams as appropriate. *Remember* that switch-based logic is such that both logic 1 and logic 0 output conditions must be deliberately satisfied (see Point 9).

(h) Develop stick or symbolic diagrams adopting suitable tactics to meet the overall strategy (d) and choice of layers (f). Determine suitable *leaf-cell(s)* from which the subsystem may be formed.

(i) Produce mask layouts for the leaf-cells *making sure* that cells can be butted together, side by side and/or top to bottom, without design rule violation or waste of space. Carefully check for any design rule errors in each standard cell itself. Determine overall dimensions of each cell and characterize in bounding box form if convenient.

(j) Cascade the replicate leaf-cells as necessary to complete the desired subsystem. This may now be characterized in bounding box form with positions and layers of inlets and outlets. External links, etc. *must* be allowed for. Check for design rule errors.

15. *Further observations on the design process* (based on Chapter 8)

    (a) First and foremost, try to put requirements into words (often an *if, then, else* approach helps to do this) so that the most appropriate architecture or logic can be evolved.

    (b) If a standard leaf-cell(s) can be arrived at, then the actual detailed design work, including simulation, is confined initially to small areas of simple circuitry.

    (c) Aim for generality as well as regularity, that is, leaf-cells, etc. should not be highly specialized unless absolutely necessary.

    (d) Communications dominate any system design.

    (e) A good library of basic leaf-cells and subsystems will speed design and allow accurate floor planning at an early stage.

    (f) A structured and orderly 'top-down' approach to system design is highly beneficial and becomes essential for large systems.

16. *Set out rules of system timing* at an early stage in design. A sample set of such rules is set out in Chapter 9 (section 9.1).

17. *Avoid bus contentions* by setting out bus utilization diagrams or tables, particularly in complex systems and/or where bidirectional buses are used.

18. *Do not take liberties with the design rules* but *do* take account of the ground rules and guidelines.

19. *Remember, IC designers should expect their systems to function first time around** and this will happen if the design concepts are correct and if the rules are obeyed.

    (We do *not* subscribe to the view 'If it works, it's out of date' (Stafford Beer), but we do contend that poorly conceived and badly designed systems may well be out of date before they work!)

* Not necessarily at optimum speed . . . this may take longer and depends on the designer's understanding of the properties of circuits produced in silicon.

10.8 The real world of VLSI design

Knowledge comes, but wisdom lingers.

Alfred Lord Tennyson

The preceding sections of this book have been intended to give the reader an understanding of the way in which system, circuit, and logic requirements may be turned into silicon and a feeling for the nature of silicon circuits. The authors believe that a sound understanding of *cause and effect* is essential if the maximum benefits are to be obtained from VLSI and the fullest range of applications opened up to VLSI realizations. Thus it is without apology that we have dwelt on the fundamental aspects of design in silicon.

From a sound foundation, a VLSI designer can operate with confidence, but must face up to the following requirements when contemplating large system designs in silicon.

1. *CAD.* The VLSI designer will need computer-aided design assistance, not only to assist in the design but also to handle the sheer complexity of the information needed to express the physical aspects of the design in a form suitable for translation into silicon.
2. *Verification tools* are essential to verify that the design is physically and logically correct and will perform correctly at the desired speed.
3. *Testability.* The designer must, from the outset, face up to the requirements of being able to test a system once it is realized in silicon.
4. *Test facilities.* Not only must testability be designed in but complex systems will need sophisticated equipment to actually test for correct operation.

Thus, it is a purpose of this chapter to present an overview of these important topics in order to put them in perspective for the budding VLSI designer. However, it is not our intention to cover comprehensively any of these topics. Although the topics are dealt with separately, it will be readily apparent that they are closely interrelated and can be significantly interdependent.

10.9 Design styles and philosophy

Style, like sheer silk, too often hides eczema.

Albert Camus

When wishing to implement a system design in silicon, various approaches are possible and, of course, a wide range of technologies is available to choose from. The designer must choose an appropriate design style, but at this point it must be stressed that in no case will the choice of style hide the lack of a competent and systematic approach by the designer. However, we may summarize the possibilities into three broad categories:

1. Full custom design of the complete system for implementation in the chosen technology. In this case, the designer designs all the circuitry and all interconnection/communication paths.
2. Semi-custom design using a library of standard leaf-cells together with specially designed circuits and subsystems which are placed appropriately in the floor plan and interconnected to achieve the desired functional performance. In this case, the designer designs a limited amount of circuitry and the majority of interconnections/communications.
3. Gate array (uncommitted logic array) design in which standard logic elements are presented for the designer to interconnect to achieve the desired functional performance. In this case, the design is that of the interconnections and communications only.

Once again the boundaries between these categories may be blurred. For example, full custom design seldom involves the complete design of the entire chip; input/output pad circuits are more or less accepted as standard components and are generally available to the custom designer.

In all cases it is desirable to take a hierarchical approach to the system design in which the principles of iteration or replication (regularity) can be used to reduce the complexity of the design task.

The designer is usually concerned with a number of key design parameters. These will include:

1. performance, in terms of the function to be performed, the required speed of operation and the power dissipation of the system;
2. time taken for the design/development cycle;
3. testability·
4. the size of the die, which is determined by the area occupied by the circuitry and in turn has a marked impact on the likely yield in production and on the cost of bonding and packaging and testing. Large die sizes are generally associated with poor yields and high costs.

Full custom design tends to achieve the best results, but *only* if the designer is fully conversant with the fundamental aspects of design in silicon so that parameters can be optimized. However, full custom design parameter optimization is usually at the expense of parameter 2, the time taken to design.

Semi-custom and gate array designs both have penalties in area and often in speed and this is contributed to by the fact that not all the available logic will be used. This is due to the need for generality in gate array and standard cell geometries. However, it may often be the case that gate arrays will be faster than a prototype full custom design in, say, MPC form and the final custom designs must often be carefully optimized.

Once the approach is chosen, there remains the design philosophy which ranges through the following general possibilities.

1. *Hand-crafted design* in which, for example, the mask layouts are drawn on squared paper with layer encoding and are then digitized to give a machine-readable form of the mask detail. Digitization can be done 'by hand', with entry of coordinates through, say, a keyboard or by more direct digitization of the drawn layout using a digitizer pad and cursor.
2. *Computer-assisted textual entry* of mask detail through a keyboard using some specially developed language employing a text editing program. Such programs may have relatively low-level capabilities, allowing the entry of rectangular boxes, and 'wires', etc. only, or may be at a higher level and allow symbolic entry of circuit elements such as transistors and contact structures.
3. *Computer-assisted graphical entry* of mask geometry through either a monochrome or color graphics terminal, again with the aid of the appropriate entry, display, and editing software.

   In cases 2 and 3 the software usually aids the processes of hierarchical system design in that leaf-cells (or symbols) can be instanced many times, each instance being placed as appropriate in the floor plan. Subsystems thus created may themselves be repeatedly instanced and placed as required to build up the system hierarchy.

   Such tools obviously encourage *regularity* and are generally used with a *generate then verify* design philosophy.
4. *Silicon compiler-based design* in which a high level approach is taken to design, and special languages, analogous to high level programming language compilers, are developed to allow the designer to specify the system requirements in a manner which is convenient and compact. The silicon compiler program then translates this input code into a mask design which will generate a circuit in silicon to meet the specified system requirements. Such programs are the subject of much research and development work at this particular time. Indeed, the work has reached a stage at which silicon compilers have been in use for some time and there are textbooks on the subject (e.g. Ayres, 1983).

10.10 The interface with the fabrication house

Knowledge without practice makes but half an artist.

Proverb

Obviously, real world designs in silicon are intended to be fabricated and there is no doubt that the learning processes associated with VLSI design depend heavily on actually designing systems in silicon, on having them *fabricated* and then on *testing* the fabricated chips. In all cases, then, good *two-way* communications between the fabrication house or silicon broker and the designer must be established.

Communication from the former to the latter usually takes the form of a set of design rules which specify clearances, widths, spacing, overlaps, etc. for the process to be used. The design rules used in this test are examples of such rules. The fabrication house will also supply design parameters relevant to its processes. These include layer resistance values, layer to layer capacitance values, etc., and typical values have been given and used in this text.

In return, the designer must communicate his mask layout designs to the fabricator in a form which is convenient and clearly understandable. Methods of expressing mask geometry are not entirely standardized, but a *de facto* standard appears to be CIF code.

10.10.1 CIF (Caltech. Intermediate Form) code

CIF is a low-level graphics language for specifying the geometry of integrated circuits (Hon and Sequin, *A Guide to LSI Implementation,* Xerox). The purpose of CIF code is to communicate chip geometry in a standard machine-readable form for mask-making. CIF code is reasonably compact and can cope with small and large system geometry. Its format is straightforward and it has the added advantage of being easily read. It has been widely used for the electronic transport of designs between universities and industrial laboratories, using such facilities as ARPANET in the United States and CSIRONET in Australia. Thus, it is appropriate to briefly examine some of the features of CIF so that the reader may appreciate general attributes of the code.

10.10.1.1 Geometric primitives

Various geometric structures such as boxes, polygons, and wires are readily defined. In general, the position, dimensions, and orientation must be specified and, also of course, the layer on which the box exists in the silicon. When examining the attributes of CIF code, the reader should be aware that CIF dimensions and positions are given in *X,Y* coordinate form but are in absolute dimension units, *not* in lambda form.

A few examples (Figure 10–19) illustrate the features of the representation.

Boxes (B) are specified as

Box Length (L) Width (W) Center (C) Direction (D)

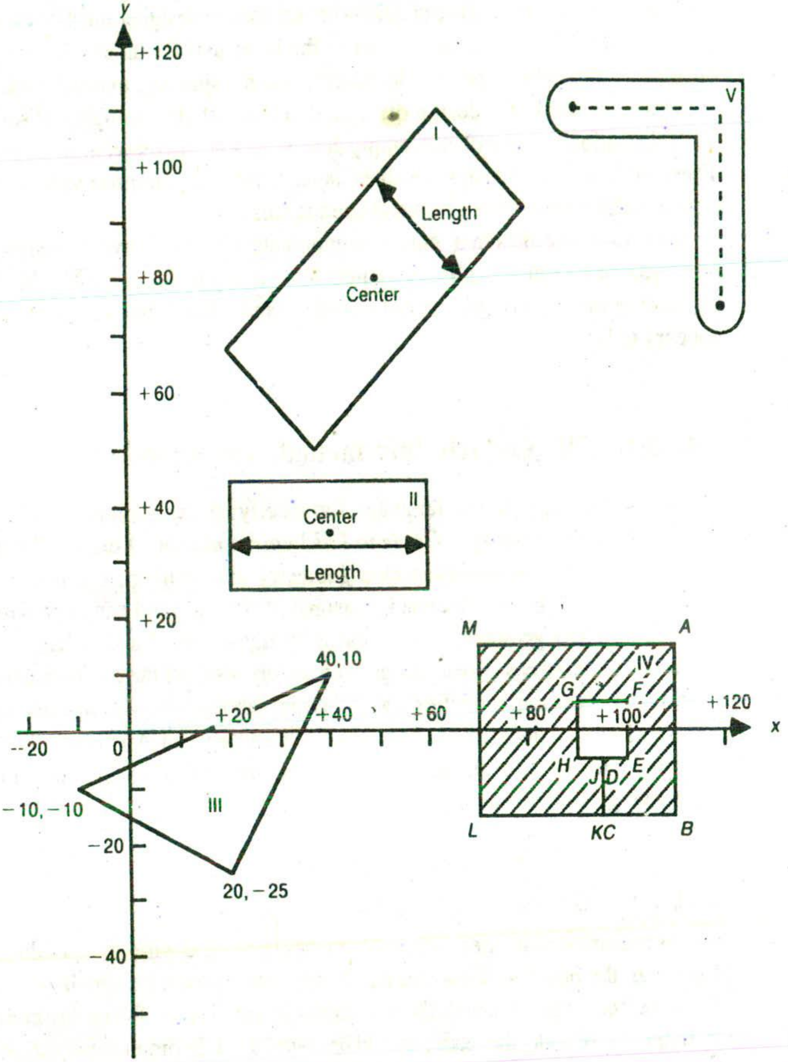


Figure 10–19 CIF Primitives — examples

Note that direction is given as a vector assumed parallel to the length. If not given, then a vector 1,0 (*x*,*y*) is assumed (that is, length will be parallel to the *x*-axis).

Boxes I and II in the diagram would therefore appear in code as

$$\begin{array}{lllll}
\text{B 25} & 60 & \underbrace{50\quad 80} & \underbrace{-10\quad 10;} & \text{(box I)} \\
\vdots & \vdots & & & \\
\text{(L)} & \text{(W)} & \text{(C)} & \text{(D)} & \text{(L, W, C, D would not appear in the actual code.)} \\
\vdots & \vdots & & & \\
\text{B 40} & 20 & \overbrace{40\quad 35;} & & \text{(box II)}
\end{array}$$

Polygons (P) are specified in terms of the vertices in order. An *n-sided* polygon requires *n vertices* and a connection between first and last is assumed to complete the boundary.

Polygon III in Figure 10–19 would therefore appear in code as

```
P - 10 - 10        40  10        20 - 25        (polygon III)
```

In order to represent areas with holes in them, as in polygon (IV), the vertices A, B, C, D, E, F, G, H, J, K, L, M would be used to specify the area.

Wires (W) are specified in terms of their width followed by the center line's coordinates of the wire's path. In Figure 10–19, wire (V) would be specified as follows:

```
W  10  90  110  120  110  120  75
```

Note that each segment of wire ends in a semicircular 'flash' which will overlap any connecting area.

10.10.1.2 Layers

Layer selection and subsequent changes are treated by mode setting prior to or during the entry of geometric primitives. Layer setting must precede the entry of the first piece of geometry and must then precede the geometric inputs on any change of layer.

For the processes in this text the layers are named as follows:

ND (nMOS diffusion/thinox)	CAA	(CMOS diff/thinox)
	CNA	(CMOS nDiff/thinox)
	CPA	(CMOS PDiff/thinox)
NP (nMOS polysilicon)	CPF	(CMOS polysilicon 1)
	CPS	(CMOS polysilicon 2)
NC (nMOS contact cut)	CC	(CMOS contact cut)
NM (nMOS metal 1)	CMF	(CMOS metal 1)
NN (nMOS metal 2)	CMS	(CMOS metal 2)
NI (nMOS implant)		
	CS or CPP	(CMOS p^+ mask)
	CW or CPW	(CMOS p-well)

NV (nMOS Via)	CVA	(CMOS Via)
NB (nMOS buried contact)		
NG (nMOS overglass cuts)	CG or COG	(CMOS overglass cuts)
	CBA	(BiCMOS p-base)
	CCA	(BiCMOS buried collector)

Layer changes are indicated by the letter L followed by the layer name.

CIF also accommodates calls (C) and rotations and translations, etc., but the elementary review given here should convey the essential features. To reinforce this, a simple cell layout is given as Figure 10–20 with the corresponding CIF code given in Table 10–7.

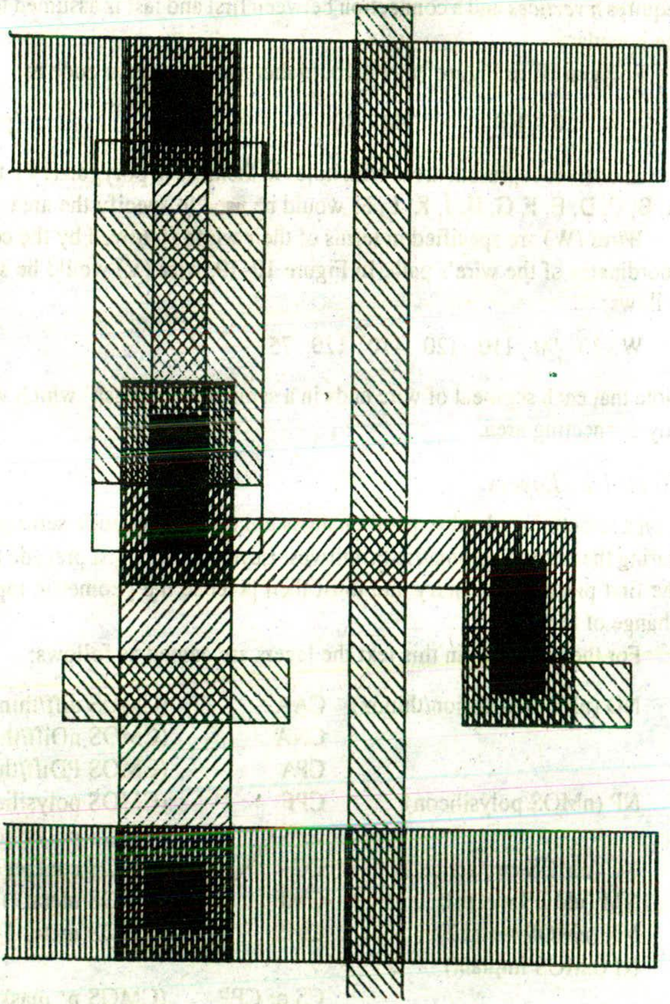


Figure 10–20 Layout of 'SRCELL' (plotted from the CIF code of Table 10–7)

Table 10–7 CIF code for SRCELL

```
25 Lambda = 250;
DS 1001;
9 SRCELL;
42 – 500, – 250 5250,7000;
      L   NM  ;
                        W 1000 0,500 4750,500;
                        W 1000 0,6250 4750,6250;
      L   ND  ;
                        B 1000 1000 1000,500;
      L   NC  ;
                        B 500 500 1000,500;
      L   NM  ;
                        B 1000 1000 1000,500;
      L   ND  ;
                        B 1000 1000 1000,6250;
      L   NC  ;
                        B 500 500 1000,6250;
      L   NM  ;
                        B 1000 1000 1000,6250;
      L   ND  ;
                        B 1000 1000 1000,3250;
      L   NP  ;
                        B 1000 750 1000,3875;
      L   NC  ;
                        B 500 1000 1000,3500;
      L   NM  ;
                        B 1000 1500 1000,3500;
      L   ND  ;
                        W 1000 1000.750 1000,3000;
                        W 500 1000.6000 1000,3500;
      L   NP  ;
                        B 2000 500 1000,2000;
                        B 1500 2000 1000,4500;
      L   NI  ;
                        B 1500 3000 1000,4500;
      L   ND  ;
                        B 1000 1000 4000,2750;
      L   NP  ;
                        B 1000 750 4000,2125;
      L   NC  ;
                        B 500 1000 4000,2500:
      L   NM  ;
                        B 1000 1500 4000,2500;
      L   ND  ;
                        W 500 1250,3000 3750,3000;
      L   NP  ;
                        W 500 4250,2000 4750,2000;
                        W 500 2750,0 2750,6750;
DF;
      C 1001 T 0,0;
End
```

10.11 CAD tools for design and simulation

Efficiency is intelligent laziness.

Arnold Glasgow, *Reader's Digest,* 1974

The design of a chip of reasonable complexity can in time be completed 'by hand' but it is both a hard and inefficient way of doing things. As far as the design of very large systems is concerned, it is *essential* to have computer aids to design so that the design can be completed in a reasonable time and, indeed, so that it can be completed at all. Whatever the size or nature of the design task, there is no doubt that well-conceived tools can make it much easier *and* do it better. Tools are therefore essential to ensure first time (and every time) success in silicon. At the very least, the designer's 'tool box' should include:

1. *physical design layout and editing* capabilities, either through textual or graphical entry of information;
2. *structure generation/system composition* capabilities, which may well be part of the design layout software implementing Point 1;
3. *physical verification.* The tools here should include design rule checking (DRC), circuit extractors, ratio rule and other static checks, and a capability to plot out and/or display for visual checking.
4. *behavioral verification.* Simulation at various levels will be required to check out the design before one embarks on the expense of turning out the design in silicon.

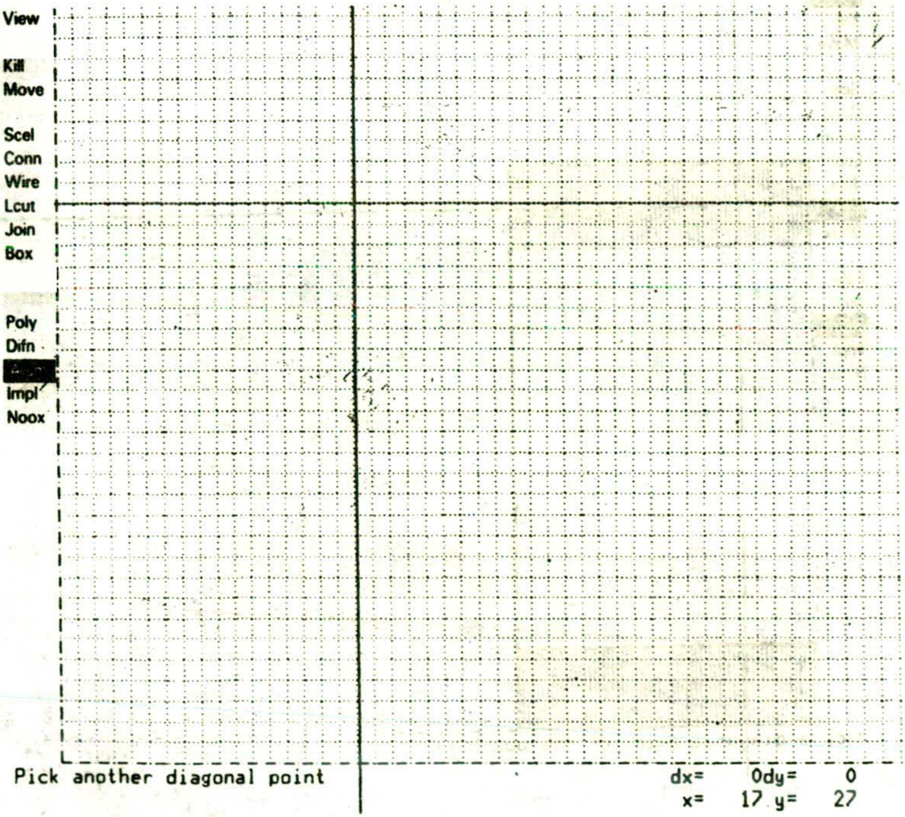
Simulators are available for logic (switch level) simulation and timing simulation. Circuit simulation via such programs as SPICE is also possible, but may be expensive in terms of computing time and therefore impractical for other than small subsystems. Recent advances in simulators have made it possible to use the software as 'a probe' to examine the simulated responses on various parts of the circuit to input stimuli also provided via the simulator. Such a facility, known as a software probe (and analogous to a CRO and associated hardware probe and signal generator), is available in various suites of design programs.

The authors can only stress that the joy of discovering that 'it does what it's supposed to' is only exceeded by the dismay of discovering that 'it doesn't work!' once a chip is fabricated (the designer having failed to carry out proper simulation testing). Some aspects of typical design tools are briefly reviewed next.

10.12 Aspects of design tools

10.12.1 Graphical entry layout

Textual entry of layouts was at one time quite widely used and special textual entry editors are in existence and may well be used for small subsystem layout. However, such tools have been virtually swept aside by a much more convenient and highly interactive method of producing layouts for which monochrome or color graphics terminals are used, and on which the layout is built up and displayed during the design process. Such systems are mostly 'menu driven', in that menus of possible actions at various stages of the design are displayed on the screen beside the display of the current layout detail. Some form of cursor allows selection and/or placement of geometric features, etc., and the cursor may also allow selection of menu items or, alternatively, these may also be selected from a keyboard. Positioning of the cursor may be effected from the keyboard in simple systems and/or cursor position may be controlled from a bitpad digitizer or from a 'mouse', etc.

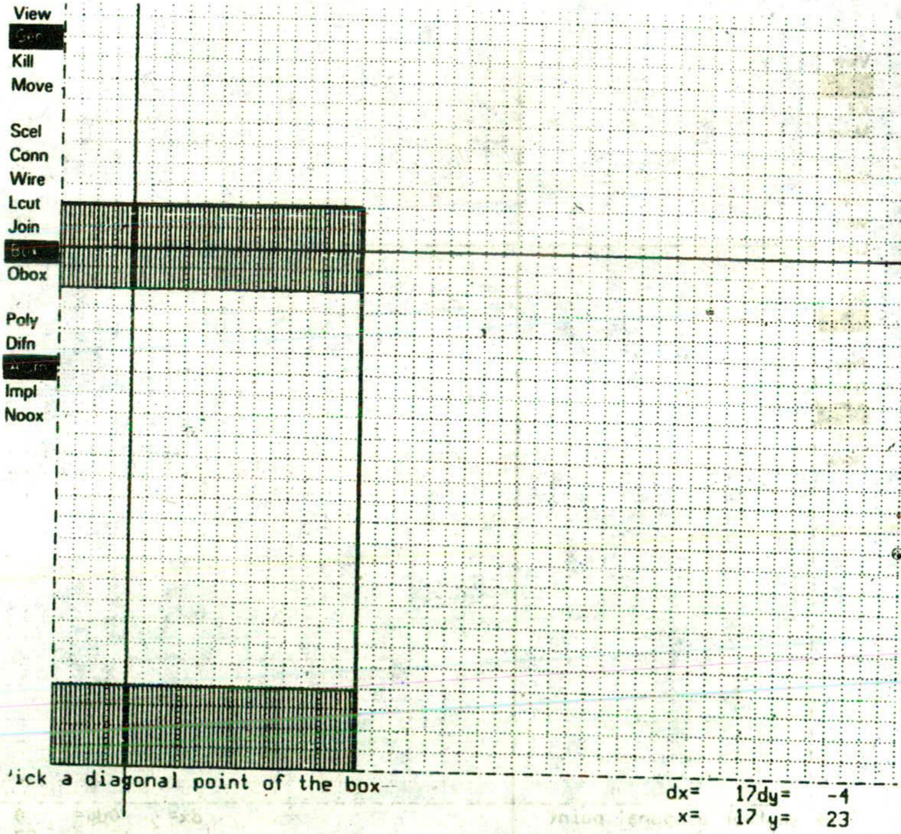


```
Pick another diagonal point                    dx=      0dy=      0
                                               x=      17 y=     27
```

Figure 10–21 Basic PLAN design environment*

* Figure shows λ grid, cross hair cursor, and menu (selected items in inverse video). *x* and *y* values of current or previous cursor position may also be displayed as shown. OBOX is selected to establish an outline (bounding) box. Then a name, (SRCL), is allocated to the enclosed cell.

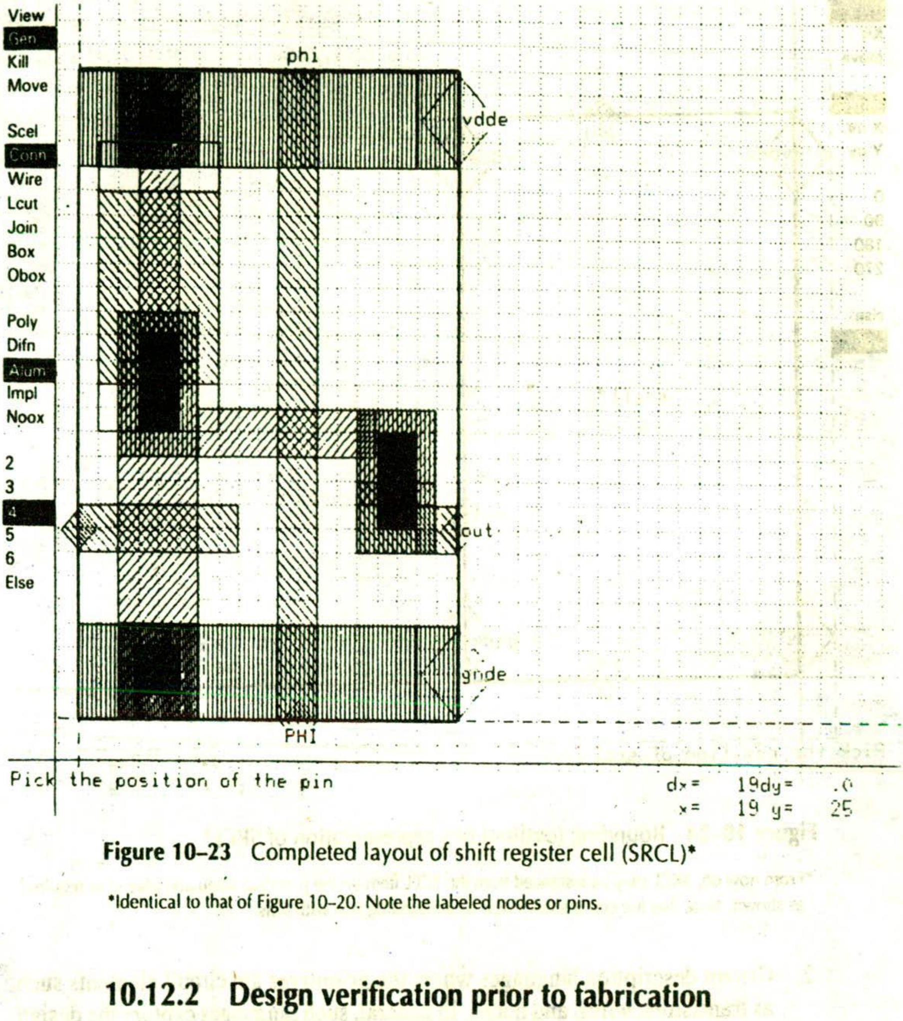
Two of the earliest available graphical entry layout packages were KIC, developed at the University of California, Berkeley, and PLAN, originally developed at the University of Adelaide. PLAN makes use of low-cost monochrome, as well as color, graphics terminals and is marketed by Integrated Silicon Design Pty Ltd, Adelaide. The use of an early version of PLAN to generate layouts is illustrated in Figures 10–21 to 10–25 and it is hoped that the inclusion of these figures, which show various stages of design, is sufficient to convey an idea of the nature of the layout process using this class of software tools.



```
'ick a diagonal point of the box--
                                              dx=   17dy=    -4
                                               x=   17 y=    23
```

Figure 10–22 Layout of metal geometry using the BOX generate feature*

* V_{DD} and GND rails have been drawn by specifying diagonally opposite corners of each box (the Alum or metal layer is selected). *x* and *y* values shown are for the last corner specified and *dx* and *dy* give the relative movement of the cursor between corners of the last box drawn.



```
Pick the position of the pin                    dx=   19dy=   .0
                                                 x=   19 y=   25
```

Figure 10–23 Completed layout of shift register cell (SRCL)*

*Identical to that of Figure 10–20. Note the labeled nodes or pins.

10.12.2 Design verification prior to fabrication

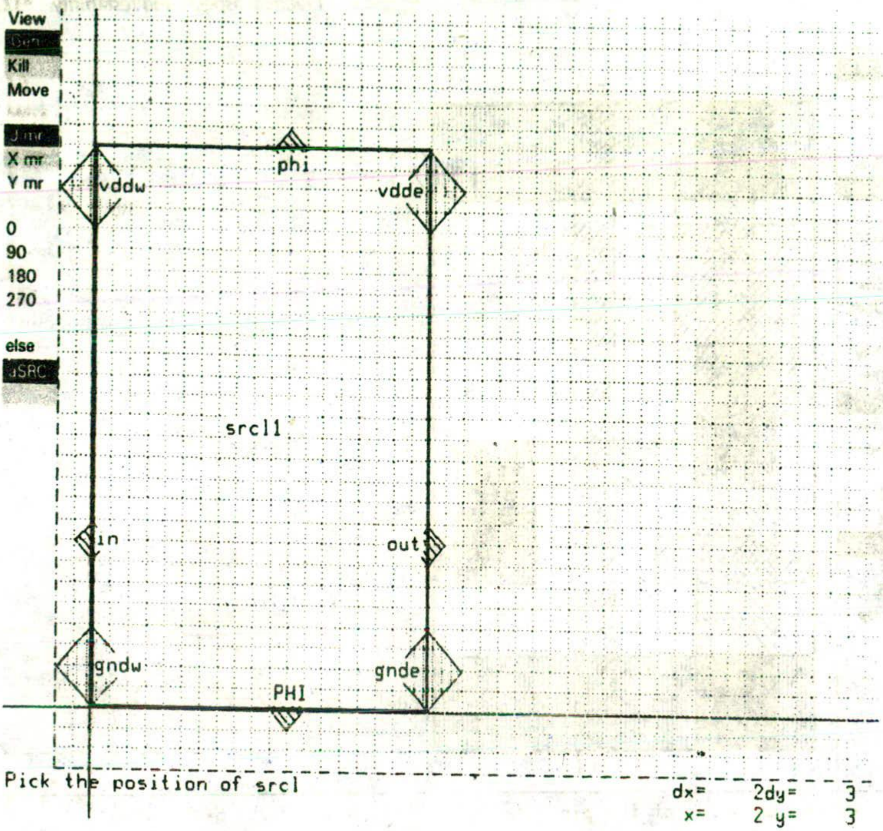
Try your skill in gilt first, and then in gold.

Proverb

It is not enough to have good design tools for producing mask and system layout detail. It is essential that such tools be complemented by equally effective verification software capable of handling large systems and with reasonable computing power requirements.

The nature of the tools required will depend on the way in which an integrated circuit design is represented in the computer. Two basic approaches are:

1. Mask level layout languages, such as CIF, which are well suited to physical layout description but not for capturing the design intent.



```
Pick the position of srcl
                                                  dx=      2dy=      3
                                                   x=      2 y=      3
```

Figure 10–24 Bounding (outline) box representation of SRCL*

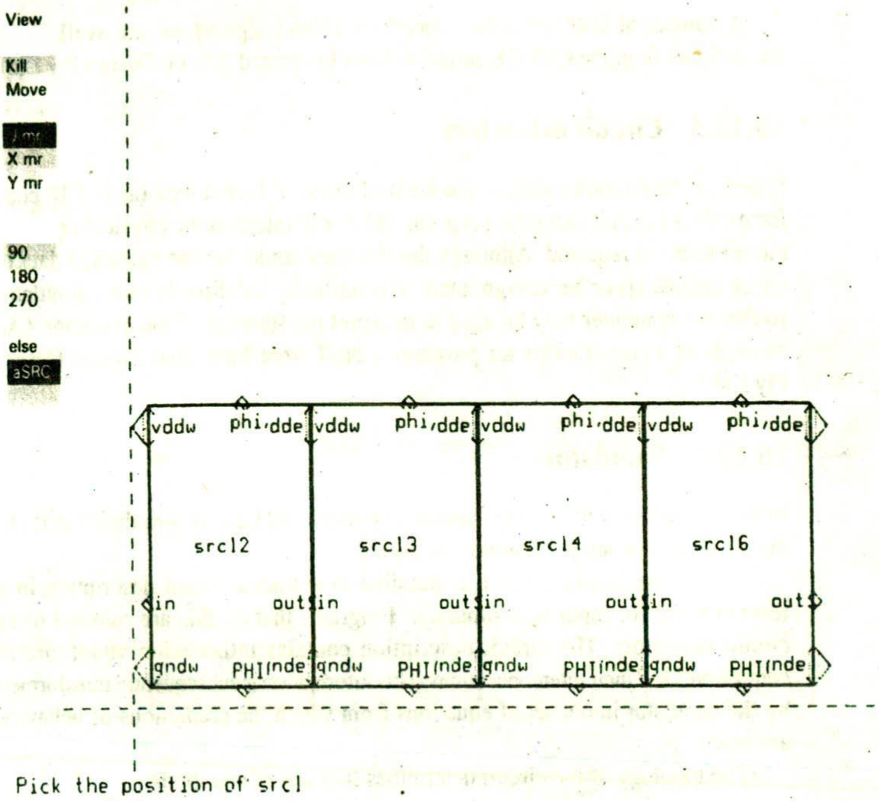
*From now on, SRCL may be instanced from the SCEL item on the previous menu and placed as required as shown. Note that the cell is shown now as a bounding box with pins.

2. Circuit description languages where the primitives are circuit elements such as transistors, wires, and nodes. In general, such languages capture the design intent but do not directly describe the physical layout associated with the design.

By and large, therefore, the designer's needs may include the following.

10.12.3 Design rule checkers (DRC)

The cost in time and facilities in mask-making and in fabricating a chip from those masks is such that all possible errors must be eliminated before mask-making proceeds. Once a design has been turned into silicon there is little that can be done if it doesn't work.



Pick the position of srcl

Figure 10–25 Instancing SRCL to form a register*

* Several instances of SRCL may be set out as shown to form a complete register (2-bits only shown) and a bounding box, and a name can be given to the whole structure.

The wise designer will check for errors at all stages of the design, namely:

1. at the pencil and paper stage of the design of leaf-cells;
2. at the leaf-cell level once the layout is complete (e.g. when the CIF code for that leaf-cell has been generated);
3. at the subsystem level to check that butting together and wiring up of leaf-cells is correctly done;
4. once the entire system layout has been completed.

The nature of physical layout verification 'design rule checking (DRC)' software may depend on whether the design rules are absolute or lambda-based, or on whether or not the layout is on a fixed or virtual grid.

A number of DRC programs, based on various algorithms, are available to the designer (e.g. the CHECK program from Integrated Silicon Design Pty Ltd).

10.12.4 Circuit extractors

If design information exists in the form of physical layout data (as in CIF code form), then a circuit extractor program which will interpret the physical layout in circuit terms is required. Although the designer could use the extracted data to check against his or her design intent, it is normally fed directly into a simulator so that the computer may be used to interpret the findings of the extractor. (An example of a circuit extractor program is NET from Integrated Silicon Design Pty Ltd.)

10.12.5 Simulators

In this section we very briefly consider the important topic of simulation prior to the VLSI design being committed to silicon.

From mask layout detail it is possible to extract a circuit description in a form suitable for input to a simulator. Programs that do this are referred to as circuit extractors. The circuit description contains information about circuit components and their interconnections. This information is subsequently transformed by the simulator into a set of equations from which the predictions of behavior are made.

The topology of the circuit determines two sets of equations:

- Kirchoff's Current Law — determining the branch currents; and
- Kirchoff's Voltage Law — determining node voltages.

The electrical behavior is defined by mathematical modeling, the accuracy of which determines two key factors:

- the accuracy of the simulation; and
- the computing power and time needed for the simulation.

We are often interested in relatively simple models to enable the highlighting of key features of performance in the design stage and to be able to observe trends as aspects of a design are changed by means of on-line interactive design.

Various types of simulators are available but generally they fall into the following groups:

- circuit simulators;
- timing simulators;
- logic level (switch level) (functional) simulators;
- system level (functional) simulators.

Circuit simulators are concerned with the electrical behavior of the various parts of the circuit to be implemented in silicon. Simulation programs such as SPICE can do this quite well, but take a lot of computing time to simulate even relatively small sections of a system and are completely impractical for circuits of any real magnitude.

Timing simulators (such as PROBE from Integrated Silicon Design Pty Ltd and QRS (developed at MCNC)) have attempted to improve matters in these respects by concentrating on active nodes and ignoring quiescent nodes in simulation. Work is proceeding in many establishments on improving the nature and performance of simulators; in particular, the way in which devices/circuits are modeled is vital. In all cases, the accuracy of simulation depends on the accuracy of the fabrication house parameters which must be fed into the simulator. In most cases, simulators attempt to predict the electrical performance with an accuracy of 20% or better. Examples of the output form of typical timing simulators have been included at several points in this book.

Timing simulators are becoming increasingly important during the design phase because of their speed and consequent interactive qualities. The structure of these tools ensures that run times are strictly linearly related to the number of devices and nodes being simulated. Speed-up is usually achieved through the use of a simple simulation cycle, a somewhat restricted network model and reasonably simple transistor models.

The simulation cycle is organized around the concept of a timestep. Each node voltage V is updated within each timestep by applying the following relation:

$$V_{new} = V_{old} + \frac{I_{ds}}{C} \Delta t$$

where

I_{ds} = drain to source current
C = node capacitance
Δt = timestep

In order to improve transistor modeling it is possible to include:

- body effect;
- channel length modulation;
- carrier velocity saturation.

The last two effects are particularly important for short channel transistors, that is, channel lengths ≤ 3 μm, and their effects should be taken into account.

Channel length modulation — for voltages exceeding the onset of saturation there is an effective decrease in the channel length of a short channel transistor. For example, the change in channel length ΔL for an n-transistor is approximated by

$$\Delta L = \sqrt{\frac{2\varepsilon_0\varepsilon_{si}}{qN_A}(V_{ds} - V_t)}$$

The resultant drain to source current I^1_{ds} is approximated by

$$I^1_{ds} = I_{ds}\frac{L}{\Delta L}$$

where I_{ds} is given by the simple expressions developed in Chapter 2.

Velocity saturation — when the drain to source voltage of a short channel transistor exceeds a critical value, the charge carriers reach their maximum scattering limited velocity before pinch off. Thus, less current is available from a short channel transistor than from a long channel transistor with similar width to length ratio and processing.

Logic level simulators can cope with large sections of the layout at one time but, of course, the performance is assessed in terms of logic levels with no or little timing information. However, there may be large sections of a system which can be satisfactorily dealt with and verified this way, provided that leaf-cell elements have been subjected to a more rigorous treatment.

When considering complete systems, logic simulators may be replaced by simulators which operate at the register transfer level.

In all cases, the designer should carefully consider the availability of all such tools when choosing VLSI design software.

10.13 Test and testability

The proof of the pudding is in the eating.

Proverb

Although this topic has been left to last in this chapter, it is by no means least in significance.

Three factors conspire to create considerable difficulties for the test engineer and, indeed, for the designer testing his or her own prototypes:

1. the sheer complexity of VLSI systems;
2. the fact that the entire surface of the chip, other than over the pads, is sealed by an overglass layer and, thus, circuit nodes cannot be probed for monitoring or excitation;
3. with minor exceptions, there is no way that the circuit can be modified during tests to make it work.

It is also essential for faults to be detected as early as possible in the manufacture of a system. A relationship, known as 'the rule of ten', tends to apply as far as test costs are concerned. This rule is concisely put as follows:

If chip test cost = \$x, then once that chip is soldered into a p.c. board with other components, test cost = \$10x. Further, once that board is integrated into a system/equipment, then the test cost escalates by a further factor of ten to test cost = \$100x. Finally, a factor which is often overlooked is that test costs may escalate by a further factor of ten when the equipment is in service in the field. It is thus essential to test at the chip level as comprehensively as possible.

Thus, chip design/fabrication mistakes can be very costly, both in terms of time and money and, for a complex chip, lack of thought at the design stage may mean that it cannot be properly tested at all. Design for testability (DFT) is an essential part of good design.

The requirements of testing must be considered at the outset and a satisfactory and sufficient measure of *testability* built into the architecture. So important is testability that many designers are prepared to dedicate 30% or more of chip area for this purpose alone.

10.13.1 System partitioning

The problems of testing, particularly at the prototype stage, are greatly eased if the system is sensibly partitioned into subsystems, each of which is as self-contained and independent as possible. To take the example of the four-bit data path chip used earlier in the text, the partitions used — namely, the register array, the adder, and the shifter — are functionally independent to a large extent and have relatively simple interconnections.

At the prototyping stage it is possible to provide special test points (by providing extra pads for probing) at the interface between the subsystems. It is also possible, in a prototype, to provide double pad and fusible link connections in key paths between subsystems. This allows these connections to be open circuited if necessary so that one system can be divorced from another as a last resort in prototype testing.

For production items, also, it helps greatly if subsystems can be checked out individually by providing the appropriate additional inlet/outlet pads for test purposes. The test requirements for exhaustive testing of large digital systems are quite prohibitive if the system is tested as a whole. Take, for example, a finite state machine realized as a mixture of combinational logic and memory elements. Let us assume n possible inputs to the combinational logic and m memory elements, and that m memory elements outputs are fed back as inputs to the combinational logic.

In this case, to fully exercise the system for every possible combination of inputs and internal states would involve the generation of 2^{m+n} test vectors. If, say, $n = 24$ and $m = 20$, the resultant number of test vectors for exhaustive testing is 2^{44} and, even if these are generated at a rate of 10^6 vectors/sec, then testing will take six months at 24 hours per day.

On the other hand, if the system is partitioned for testing, exhaustive testing can be reduced to $2^n + 2^m$ vectors, a much more reasonable proposition (and for the figures given above would result in a test time of less than 20 seconds).

10.13.2 Layout and testability

Although it is impossible to generalize on this topic, common sense and a thoughtful approach to system layout may well considerably ease the problems associated with testing. For example, the inclusion of key point test pads or pads to energize special test modes are possible when the design is evolving.

The designer should also be aware of practical factors which will reduce the likelihood of short and open circuits. In particular, for MOS circuits, it has been shown that short circuits and open circuits in the metal layer and short circuits in the diffusion layer were the dominant faults experienced. Careful observance of design rules and ground rules should help to reduce the incidence of such faults.

10.13.3 Reset/initialization

One simple but very effective aid to testing and testability is to design a reset facility into all digital systems of any complexity. This has the considerable advantage of setting all internal states to known values, and testing may then at least proceed from known conditions.

The simple expedient is quite often overlooked or omitted.

10.13.4 Design for testability*

There are two key concepts underlying all considerations for testability. They are:

1. controllability
2. observability.

Quite simply, these concepts ensure that the designer considers the provision of means of setting or resetting key nodes in the system and of observing the response at key points.

The effects of testability or lack of it are such that it has been predicted that testability will soon become the main design criterion for VLSI circuits. The alternative is to save area by ignoring testability, but the penalties are such that

* The authors acknowledge contributions embodied in this section by Dr A. Osseiran of the Swiss Institute of Technology.

even for modest complexity (e.g. 10,000 gates per chip) the test costs could rise by a factor of five to ten, compared with the same system designed for testability. Given that test is already a significant component of LSI chip costs, the effects will be quite dramatic and could well cause the test costs to exceed all other production costs by a significant factor.

Design for testability (observability and controllability) is then reduced to a set of design rules or guidelines which, if obeyed, will facilitate test.

A failure during testing at the chip level may be due to a design defect or a poorly controlled fabrication process.

The inputs of the device under test (DUT) are subjected to a test pattern (or test vector) which supplies a set of binary values, in combination and/or in sequence, to detect faults. The specification of the test vector sequences must involve the designer, while the generation and application of test patterns to a DUT are the problems faced by the test engineer. Test pattern generation is assisted by using automatic test pattern generators (ATPG), but they are complicated to use properly and ATPG costs tend to rise rapidly with circuit size.

Once the application of a test pattern has revealed a fault, the process of diagnosis must be invoked to localize the fault.

10.13.4.1 Test coverage

Detecting all the possible faults in a DUT corresponds to 100% 'test coverage'. In general it is relatively easy to detect the first 80% of faults using various classical test strategies, but when more than an 80% coverage is required, appropriate test strategies must be developed. In any case, it is not generally possible to anticipate 100% of all faults, so that we tend to talk about a set of fault hypotheses which may then be covered 100%.

Faults may be classified using different models, and three such levels of definition are:

- mathematical model
- logical model (stuck-at)
- physical model.

The latter two are most commonly used. The 'stuck-at' model has been widely used and was originaly developed in the testing of p.c. boards but is not in itself sufficient to test actual VLSI CMOS circuits.

A further set of physical fault models is also used:

- Class 0: A single physical defect such as a faulty contact or via, a transistor stuck on or stuck off, an interconnection through any layer open circuit.
- Class 1: Class 0 with a short circuit between metal lines or diffusion lines.
- Class 2: Class 1 with short circuit(s) between two lines on any layer.

10.13.4.2 Nature of failures in CMOS devices

Failures may occur after a CMOS circuit has been fabricated and has successfully passed initial testing. Such failures may be due to poor design, weaknesses introduced during fabrication, ageing, or corrosion (of metallization) mechanisms which, again, may be accelerated owing to poor quality control in fabrication. The MOSFETs used in CMOS circuitry are susceptible to performance defects if there is a change either in the specified threshold voltage or in the transconductance.

Design faults are mostly due to deviations from the design rules specified for the fabrication process and this type of fault is difficult to detect since the manifestation of the fault often occurs later in the life of the device. Such faults mainly take the form of open circuits in conductors or short circuits between conductors. Crosstalk is also a cause of faults which are generally transient and are again due to poor design.

Other failures may be due to oxide breakdown, usually the thin oxide between gate and channel regions. This form of breakdown is often related to the inadequate protection against electrostatic discharge (ESD) and may be traced back to defects in or poor design of input/output pad circuitry.

Another problem is caused by hot carrier injection which causes both threshold voltage shift and transconductance degradation due to charge accumulation in the gate oxide.

10.13.5 Testing combinational logic

The solution to the problem of testing combinational logic is to generate a set of test patterns which will detect all possible fault conditions.

The first approach to testing an N input circuit is to generate all the possible 2^N input signal combinations by means of, say, an N-bit counter (controllability) and observe the output(s) for checking (observability). This is called exhaustive testing and is very effective, but is only practicable where N is relatively small. The reason for this becomes obvious when exhaustive test times are evaluated, taking, say, a relatively high (10 MHz) clock speed to the test pattern generating counter, results for several values of N being as follows:

N inputs	=	*2^N combinations*	=	*$2^N \times 0.1$ μsec.*	=	*total test time*
32 inputs		2^{32}		$2^{32} \times 10^{-7}$sec		≥ 7 minutes
40 inputs		2^{40}		$2^{40} \times 10^{-7}$sec		≥ 30 hours
64 inputs		2^{64}		$2^{64} \times 10^{-7}$sec		≥ 574 *centuries*

10.13.5.1 Sensitized path-based testing

Many of the patterns generated during exhaustive testing may not occur during the application of the circuit. Thus, it is productive to first enumerate the possible faults and then generate a set of appropriate test vectors.

The basic idea is to select a path from the site of the possible fault, through a sequence of gates leading to an output of the logic circuitry under test. The process comprises three steps:

1. *Manifestation*: Gate inputs at the site of an assumed fault, say a 'stuck at' (SA) fault, are specified to generate the opposite value to the assumed SA value (0 for SA1, 1 for SA0).
2. *Propagation*: Inputs of other gates are determined so as to propagate the fault signal along the selected path to the primary output of the circuit. This is done by setting *And/Nand* inputs to '1' and *Or/Nor* inputs to '0'.
3. *Consistency* (or justification): This final step finds the primary input patterns to realize all the necessary values. This is done by tracing backward from the gate inputs to the primary input of the logic.

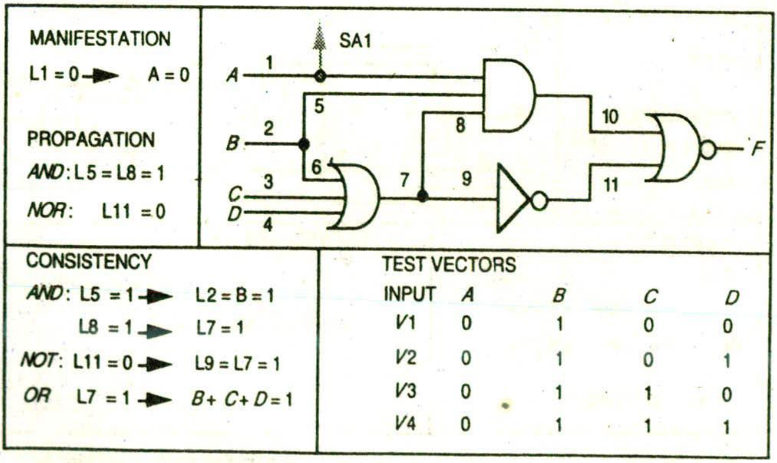
Examples will help explain the process.

Example 1 : Take an SA1 fault on line 1(L1) in Figure 10–26, then

Manifestation: L1 = 0, then A = 0. In a fault-free situation, output F changes with A if B, C and D are fixed. If L1 is SA1 then F = 0 even if A = 0.

Propagation: For propagation through the *And* gate, line 5(L5) = line 8(L8) = 1 and, since we are propagating the condition L1 = 0, then L10 = 0. So that the propagated fault manifestation can reach the output through the *Nor* gate, then L11 = 0. Output F is thus read and compared with the fault-free value F=1.

Consistency: For the *And* gate, L5 = 1 and thus L2 = input B = 1; also L8 = L7 = 1. So far, we have determined the values of inputs A(= 0) and B(= 1) and also that L7 =1. For this latter contention to be true, we may see that B + C + D = L7 = 1. If L7 is =1 then L9 = 1 and L11 will be 0, which is consistent with the



MANIFESTATION

L1 = 0 → A = 0

PROPAGATION

AND: L5 = L8 = 1

NOR: L11 = 0

CONSISTENCY

AND: L5 = 1 → L2 = B = 1

L8 = 1 → L7 = 1

NOT: L11 = 0 → L9 = L7 = 1

OR L7 = 1 → *B*+ *C*+ *D* = 1

TEST VECTORS

INPUT	*A*	*B*	*C*	*D*
*V*1	0	1	0	0
*V*2	0	1	0	1
*V*3	0	1	1	0
*V*4	0	1	1	1

Figure 10–26 Combinational logic testing — sensitized path — example1

-23

Nor gate propagation requirements. A set of test vectors, *V*1 to *V*4, may now be specified as shown in the figure.

These tests will therefore reveal the SA1 fault. However, there are some faults which will be inherently undetectable with the sensitized path approach as illustrated in the following:

Example 2: Take the same circuit as in Figure 10–26, but with a stuck at 1(SA1) fault on line L8 as in Figure 10–27.

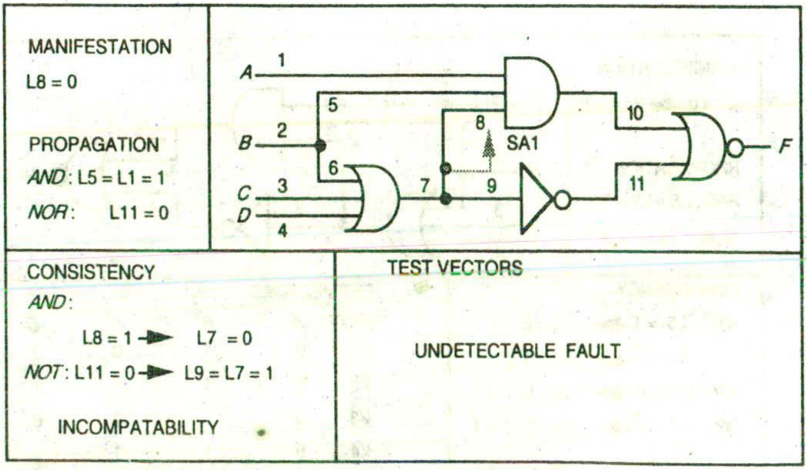
Manifestation: L8 = 0.

Propagation: For propagation through the *And* gate, L5 = L1 = 1 and, since we are propagating the condition L8 = 0, then L10 = 0. For the *Nor* gate L11= 0, which means that L9 = L7 = 1.

Consistency: For the *And* gate, L8 = 0 and thus L7 = 0. Clearly this is inconsistent since L7 cannot be set to 1 and 0 at the same time. This conflict cannot be resolved and the fault is undetectable.

10.13.5.2 *The D-algorithm* [Roth]

Given a circuit comprising combinational logic, the algorithm aims to find an assignment of input values that will allow detection of a particular internal fault by examining the output conditions. In order to do this the algorithm is based on the hypothesis of the existence of two machines — a good machine and a faulty machine. The existence of a fault in the faulty machine will cause a discrepancy between its behavior and that of the good machine for some particular values of inputs. The D-algorithm provides a systematic means of assigning input values for that particular design so that the discrepancy is driven to an output where it may be observed and thus detected. The algorithm is extremely time-intensive



MANIFESTATION

L8 = 0

PROPAGATION

AND: L5 = L1 = 1

NOR: L11 = 0

CONSISTENCY

AND:

L8 = 1 ⟶ L7 = 0

NOT: L11 = 0 ⟶ L9 = L7 = 1

INCOMPATABILITY

TEST VECTORS

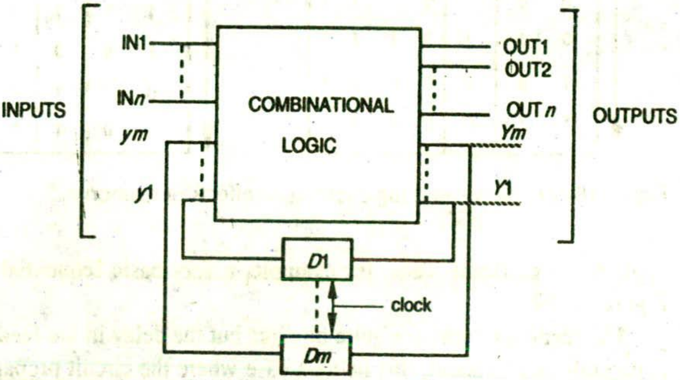
UNDETECTABLE FAULT

Figure 10–27 Combinational logic testing — sensitized path — example 2

and computing intensive for large circuits and has been the subject of several adaptions, modifications and improvements. LASAR (Logic Automated Stimulus And Response), PODEM (Path Oriented DEcision Making) and FAN (FAN-out oriented test generation) are all improvements on the D-algorithm. Further reading is cited in the reference list which follows this chapter.

10.13.6 Testing sequential logic

Sequential circuits, which may be generally represented as finite state machines, may be modeled as combinational logic with a set of delays and feedback from output to input as shown in Figure 10–28.



Dm etc. = Delay flip-flop or register

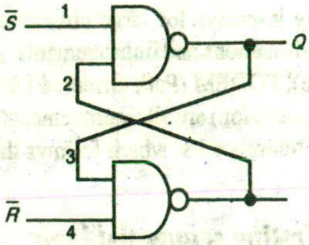
Figure 10–28 Sequential logic testing — finite state machine model

The 'm' feedback variables constitute the state vector and determine the maximum number of finite states which may be assumed by the circuit. In the most general case, the next state and the output are both functions of the present state and the independent inputs. The delay elements are generally assumed to be associated with the feedback path and, for clocked systems, the basic delay elements are flip-flops, although, in asynchronous circuits in particular, the delays may be contributed by circuit propagation delays.

The test generation for a sequential circuit is a very complicated task since the test signals must not only be logically correct but must also occur at the correct time relative to other signals.

10.13.6.1 The effect of memory

All sequential circuits exhibit a memory property since, in deciding what to do next, they take into account (or remember) previous conditions. In testing then, it is not only the test pattern but also the order or sequence in which it is



INPUTS		OUTPUT Q				
		GM	Input SA1			
$\overline{S}$	$\overline{R}$		1	2	3	4
0	0	1	0	1	1	1
0	1	1	0	1	1	1
1	0	0	0	0	0	1
1	1	0	0	0	1	1

INPUTS		OUTPUT Q				
		GM	Input SA1			
$\overline{S}$	$\overline{R}$		1	2	3	4
1	1	?	?	0	1	?
1	0	0	0	0	0	?
0	1	1	0	1	1	1
0	0	1	0	1	1	1

Figure 10–29 Sequential logic testing — effects of memory

applied is significant. Take, for example, a very basic sequential circuit as i Figure 10–29.

The feedback paths are quite obvious but the delay in the feedback path i apparently non-existent, this being a case where the circuit propagation delay contribute the necessary delay elements.

To explain the tabulations under the figure, the input test pattern is applied a shown under the $\overline{S}\ \overline{R}$ heading and working in sequence from top to bottom row The remaining columns tabulate the state of output Q first of all for a 'goo machine' (GM) and then for the 'faulty machine' (FM) for a SA1 fault on each c the four input lines (1, 2, 3, 4).

Note that in the first table with the SA1 fault on line 2 the machine matche the good response and so this particular test sequence will not detect a SA1 fau on line 2.

In the second table, the vectors are applied in exactly the reverse order. I this case '?'s appear owing to the memory property of the circuit, each '?' indicatin that Q will retain whatever value it had prior to the application of the test vecto for that row. Again, the SA1 fault on line 2 may not be detected if the latches ar reset (i.e. $Q = 0$) prior to applying the test sequence.

10.13.6.2 The iterative test generation method

An obvious way of approaching the testing of sequential logic is to 'convert' th logic into combinational logic by cutting the feedback lines, thus creating pseud inputs and outputs as well as the original primary input and output lines. For an N-state machine, this arrangement is then replicated N times so that an N-state sequential machine is converted into an N-time frame combinational machine.

The main problem of this technique is that a simple fault in the sequential machine is manifest as N multiple faults during test. This is time-consuming for circuits of any complexity. It is also necessary to describe all the initial states of the circuit, which is also time-consuming. For these reasons the iterative test generation (ITG) methods are best suited to logic with few feedback loops as in control logic for example.

10.13.7 Practical design for test (DFT) guidelines

Practical guidelines for testability should aim to facilitate test processes in three main ways:

- facilitate test generation;
- facilitate test application; and
- avoid timing problems.

The discussions in this section address these matters.

10.13.7.1 Improve controllability and observability

Design for test methods must ensure that a design is well enough covered to provide for complete and efficient testing. When a node is difficult to access from primary input or output pads, then a very effective method is to add additional, internal pads to access the desired point. These additional pads may be accessed using a prober.

If the node is a link between blocks of a circuit, as in Figure 10–30, then the most immediately obvious attributes required are to be able to observe the output of block 1 and also to provide for the control of block 2. Some additional circuitry will be required and a possible configuration is set out in the figure. If the *Normal/$\overline{Test}$* line is set to 1(*Normal*) then transmission gates T_2 and T_3 are open and T_1 is closed. Normal transmission between the blocks can take place through T_2 but a *control* input to block 2 can also be applied through T_3. When the *Normal/$\overline{Test}$* line is set to 0 (*Test*) then transmission gates T_2 and T_3 are closed, there will be no transmission between the blocks, and the output (*observe*) of block 1 can be monitored through T_1 which is now open.

This solution requires three pads and eight transistors in a CMOS environment. This technique must be complemented by other appropriate testing techniques which will depend on the internal structures of blocks 1 and 2.

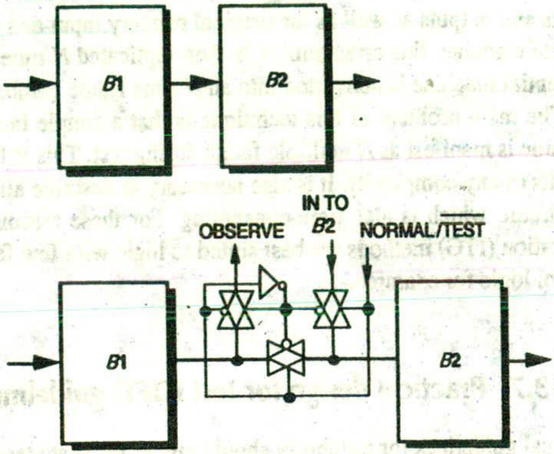


Figure 10–30 Practical DFT guidelines — controllability and observability

10.13.7.2 The use of inter-block multiplexers

Some general attributes are illustrated in Figure 10–31. This arrangement allo the bypassing of blocks. The addition of demultiplexers also improves observabili The major penalties incurred here are the numerous extra devices and the add propagation delays through the multiplexers.

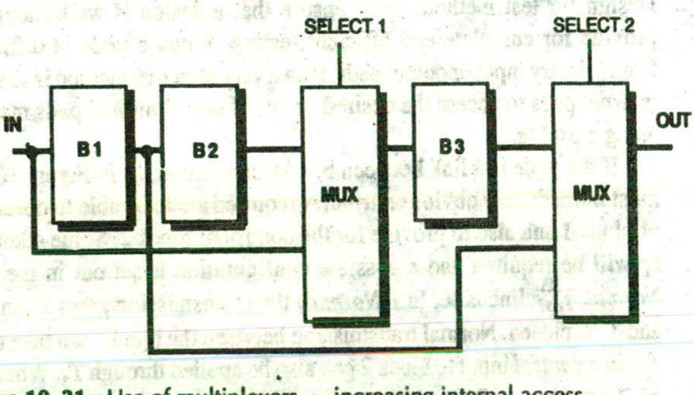


Figure 10–31 Use of multiplexers — increasing internal access

10.13.7.3 The partitioning of large circuits

Partitioning large circuits into smaller subcircuits is an effective way of reduc test generation complexity and test time. It has been shown that test generat effort for an n gate general purpose logic circuit is proportional to somewh between n^2 and n^3 (Bennetts, 1984). If the circuit is partitioned then the effor

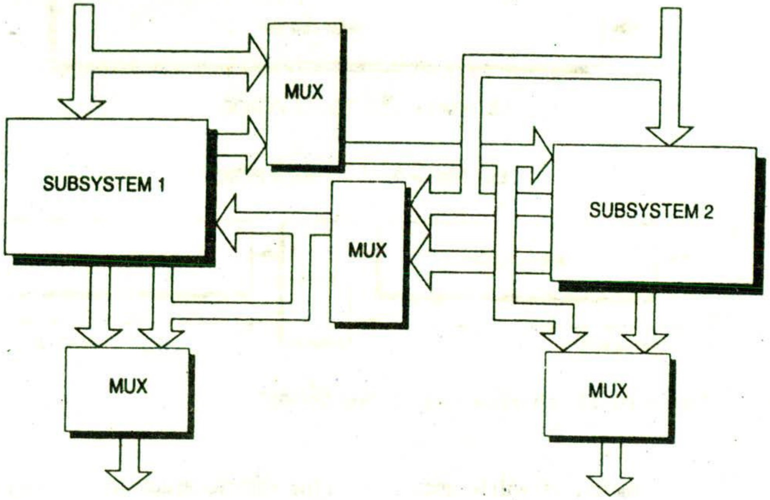


Figure 10–32 System partitioning — using multiplexers

reduced correspondingly. For example, exhaustive testing of the SN7480 adder circuit for SA1 and SA0 faults requires 2^9 (= 512) tests. If the adder is partitioned into four subcircuits, then the number of tests is reduced to 24. Clearly, partitioning should be done on a logical basis into recognizable and sensible subfunctions and can be achieved physically by incorporating clock line isolation and control facilities, reset and power supply lines. Isolation and control are readily achieved through the use of multiplexers as suggested in Figure 10–32.

10.13.7.4 Dividing long counter chains

Counters are sequential and need a large number of input vectors to be fully tested. Partitioning into sub-counters can be very effective in reducing test complexity. For example, the full testing of a 16-bit counter requires the application of 2^{16} = 65,536 clock pulses. Division of the counter into two 8-bit counters, as Figure 10–33, reduces this number to 2×2^8 = 512 clock pulses.

10.13.7.5 Initialization of sequential logic

An important problem in sequential logic testing arises at power-up time where the first state will be quite random if there is no initialization. In this case it is impossible to start a test sequence correctly. The remedy is to design the circuit using elements which have a *preset* and/or *clear* facility (e.g. JK flip-flop elements with *Pr.* and *Clr.* inputs). From a practical viewpoint, this could be very space-consuming and it may often be sufficient to initialize only the first stage. For example, if its first stage only is initialized, a serial in serial out counter will pick up a known state after a few clock pulses.

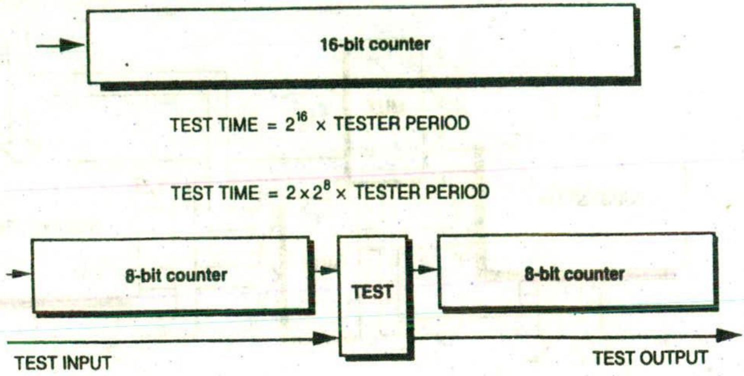

16-bit counter
TEST TIME = 2^{16} × TESTER PERIOD
TEST TIME = 2 × 2^8 × TESTER PERIOD
8-bit counter
TEST
8-bit counter
TEST INPUT
TEST OUTPUT

Figure 10–33 Dividing long counter chains

Sometimes it will be necessary for the tester to be able to override the normal initialization state of the logic and the addition of appropriate gating in the *reset/initialize* control line will achieve this.

10.13.7.6 Asynchronous sequential logic.

Asynchronous logic is driven by self-timing state transitions in response to changes of the primary inputs. This makes it generally impossible to determine when the next state is actually established and in consequence there are large problems to be faced in the timing and also in the memory effects associated with such circuits. Although asynchronous logic is inherently faster than clocked logic it has several serious disadvantages from the test viewpoint as follows:

- testing is difficult;
- sensitivity to tester skew;
- non-deterministic behavior;
- prone to races and other hazards.

The design processes are more difficult than synchronous logic and must be approached with care, taking due account of critical race and other hazard-generating conditions.

10.13.7.7 Avoiding logical redundancy

Logical redundancy may be present by design; for example, in order to mask a static hazard condition, or unintentionally as a design bug. In both cases it is not possible to make a primary output value dependent on the value of the redundant node. Thus, there are certain fault conditions associated with the node which cannot be detected, Take, for instance, the two sets of conditions outlined in Figure 10–34.

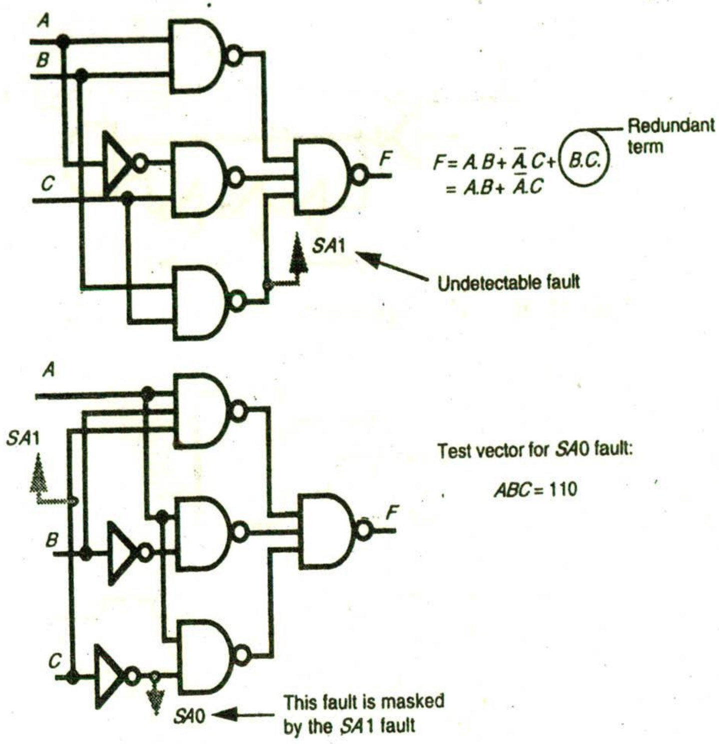

A
B
C
F
$F = A.B + \bar{A}.C + \boxed{B.C.}$
$= A.B + \bar{A}.C$
Redundant term
SA1
Undetectable fault
A
SA1
B
C
F
Test vector for SA0 fault:
ABC = 110
SA0
This fault is masked by the SA1 fault

Figure 10–34 Fault masking due to logical redundancy

10.13.7.8 Avoiding delay dependant logic

An example of a delay-dependent circuit is given in Figure 10–35. It will be seen that the presence of a pulse at the *And* gate output depends not on the logical performance of the three inverters but rather on their temporal performance. Automatic test pattern generators (ATPGs) work in the logic domain and view delay-dependent logic as redundant combinational logic. In the case illustrated in Figure 10–35, the ATPG will see the *Anding* of a signal with its complement and will therefore always compute a '0' as the output of the *And* gate — rather than a pulse.

10.13.7.9 Avoiding gating or asynchronous delays in the clock line

When a clock signal is gated with another signal, such as a *load* signal coming from a tester, then any skew (or other hazard) on that signal can cause an erroneous output from the associated logic. This is illustrated in Figure 10–36.

Further, another timing situation to avoid is that illustrated in Figure 10–37

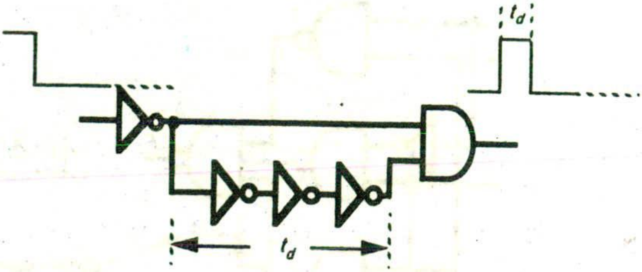


Figure 10–35 Delay-dependent logic

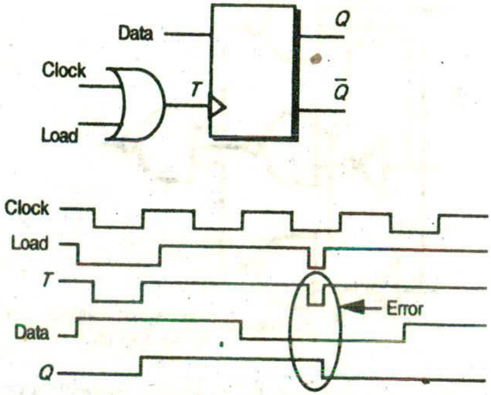


Figure 10–36 Clock line gating hazard

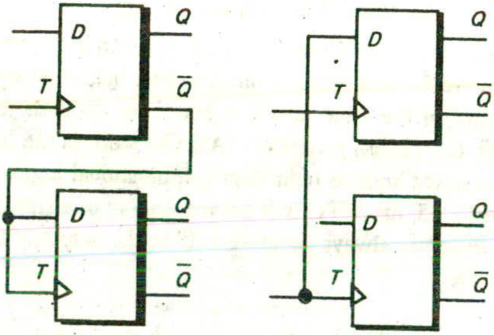


Figure 10–37 Asynchronous delays in the clock line

where the tester could not be synchronized if one or more clock is dependent on asynchronous delays (from the *D*-input to *Q*-output of the flip-flop in the figure), or when a signal is used both as data and as a clock.

10.13.7.10 *Avoiding self-resetting logic*

The problems here are akin to those in asynchronous logic, since the reset input is independent of the system clock. This can result in an erroneous value being read by the tester. The situation is indicated in Figure 10–38(a).

One solution to this problem is to allow the tester to override by adding an *Or* gate as indicated in Figure 10–38(b). This allows the tester to receive the right response at the right time.

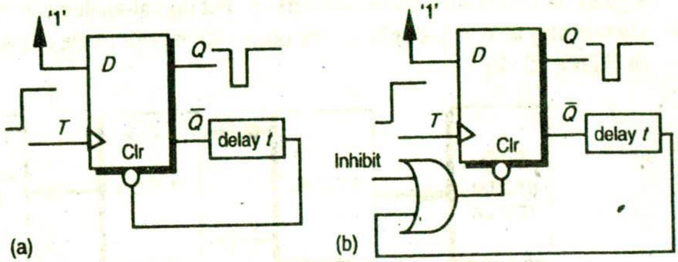


Figure 10–38 Problems associated with self-resetting logic

10.13.7.11 *The use of bused structures*

This approach is related to the partitioning technique and is very widely used for microprocessor-like circuits as illustrated in Figure 10–39.

Using this arrangement allows the tester access to all the main subsystems and other modules which the buses interconnect. The tester can then effectively disconnect any unit or module from the bus by putting its output into the high impedance state. Test patterns can then be applied to each separately.

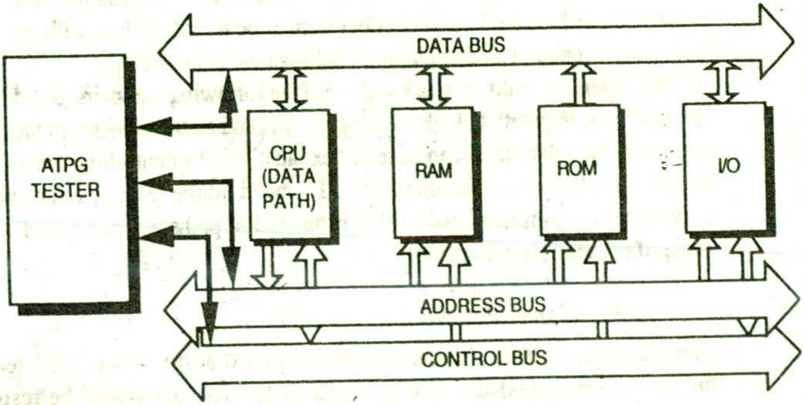
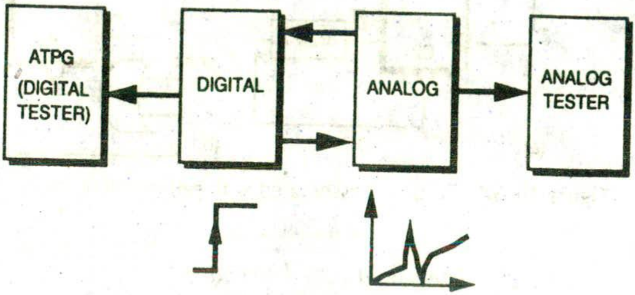


Figure 10–39 Use of bused structures

10.13.7.12 Separation of analog and digital circuits

The testing of analog circuits requires a completely different strategy from digital circuits and is therefore incompatible.Furthermore, the fast rise- and fall-times of digital signals can give rise to cross-talk problems in analog signal lines if they are in close proximity. Where it is essential to route digital signals near analog lines, then consideration must be given to balancing and shielding the digital signals.

In the case of analog-digital converters, it is better to bring out the analog signals for observation before conversion. For digital-analog conversion the digital signals may also be brought out for observation prior to the converter as outlined in Figure 10–40.



ADC TESTING: BRING OUT ANALOG INPUTS FOR TEST OBSERVE DIGITAL OUTPUT

DAC TESTING: BRING OUT DIGITAL INPUTS FOR TEST OBSERVE ANALOG OUTPUT

Figure 10–40 Separation of analog and digital signals

10.13.7.13 Bypassing techniques

Bypassing a subsystem consists of providing the facilities for propagating its inputs directly through to its outputs. The aim is to bypass the sub-system in order to directly access another subsystem to be tested, and, as with partitioning, wide use is made of multiplexers to achieve the bypassing.

Bypassing techniques work well with the following: counters, dividers, RAM, ROM, PLAs, sequential blocks, analog circuits and internal clocks. In the bypassing approach, the subsystems can be tested exhaustively by controlling the multiplexer-based interconnections in the system. To speed up the testing, some subsystems may be tested simultaneously if the propagation paths are associated with other disjoint or separate subsystems.

10.13.7.14 Some observations on DFT

The preceding sections have not presented an exhaustive list of DFT techniques but have been intended to present a set of rules which should be respected in design. Some of the guideline goals are to simplify test vector generation and application, and others are intended to avoid timing problems in the design. The references given at the end of this chapter and other related material give variants of the design guidelines for PCBs and for ICs.

10.13.8 Scan design techniques

The testability guidelines so far presented provide ad hoc methods for dealing with random logic designs. The scan design techniques which are now to be discussed are structured approaches to designing sequential circuits so that testability is 'designed in' from the outset.

The major difficulty in sequential circuit testing is in determining the internal state of the circuit. Scan design techniques are directed at improving the controllability and observability of the internal states. The approach aims to reduce the problem of testing a sequential circuit to that of testing combinational logic.

10.13.8.1 The scan path

A sequential circuit comprises combinational logic and storage elements — usually in the feedback path — as illustrated in Figure 10–41. Scan path design techniques configure the logic so that the inputs and outputs of the combinational part can be accessed and the storage elements reconfigured to form a shift register known as the scan path. Thus the internal states of the circuit can be observed and controlled by shifting (scanning) out the contents of the storage elements

The storage elements are usually 'D', JK' or 'RS' flip-flop elements with the classical structure being modified by the addition of a two-way multiplexer on the data input(s). The multiplexer is controlled by an external *mode* signal and allows the scan path reconfiguration to be effected. In Figures 10–41 and 10–42 a basic 'D' flip-flop has been shown with the added input multiplexer. This configuration is commonly known as an 'MD' (multiplexed 'D') flip-flop.

The sequential circuit containing the scan path has two modes of operation — a *normal* and a *test* mode. The configuration associated with each basic mode is set out in Figures 10–42(a) and (b) — *normal* and *test* mode respectively.

A large sequential circuit is generally partitioned into a number of subcircuits each with a combinational section and one associated scan path.. The efficiency of the test pattern generation for the overall combinational circuit is greatly improved by partitioning since its depth is reduced.

Before applying test patterns, the scan path shift register is verified by shifting in all ones then all zeroes.

A general method for testing with the scan path approach is as follows:

1. Set the mode to *test* so that the scan path is configured.
2. Verify the scan path by shifting test data in and out.
3. Set the shift register to a known initial state.

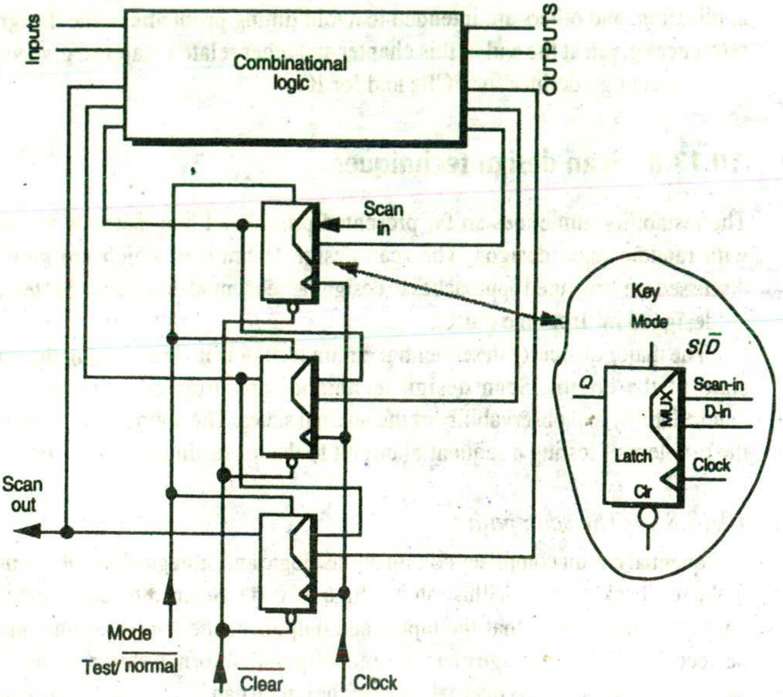


Figure 10–41 Sequential circuit configured for scan path testing

4. Apply a test pattern to the primary inputs of the overall circuit.
5. Set the mode to *normal.* The circuit then settles and the primary outputs are monitored.
6. Activate the circuit with one clock pulse.
7. Return to the *test* mode.
8. Scan out the contents of the scan path registers and simultaneously scan in the next pattern.
9. Repeat from step (4) etc.

10.13.8.2 Level-sensitive scan design (LSSD)

This is a technique, initially developed by IBM, which incorporates two aspects — level sensitivity and a scan path approach (Williams, 1986). The general arrangement is indicated in Figure 10–43.

The *level-sensitive* aspect means that the sequential network is designed so that when an input change occurs, the response is independent of the component and wiring delays within the network.

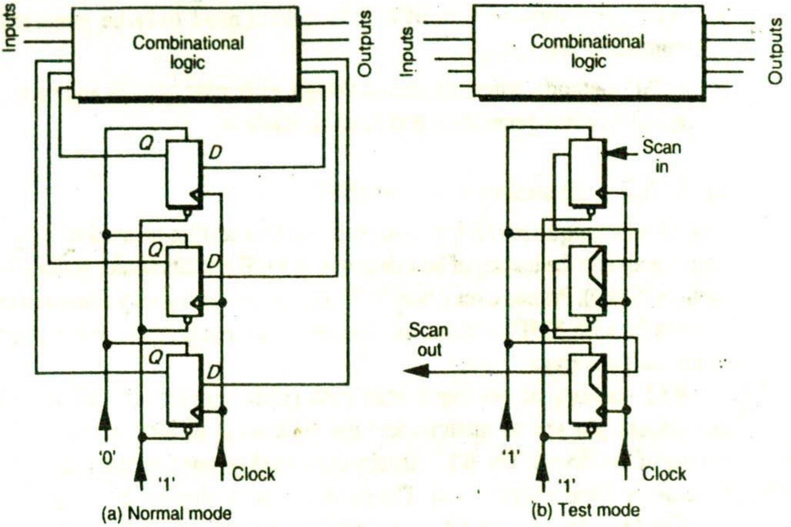


Figure 10–42 Sequential circuit showing normal and test mode configurations

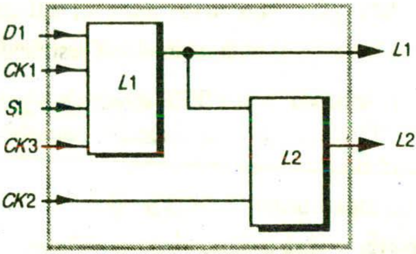


Figure 10–43 Level-sensitive scan design LSSD configuration

The *scan path* aspect is due to the use of shift register latches (SRL) employed as storage elements. In the *test* mode they are connected as a long serial shift register. Each SRL has a specific design similar to a master-slave flip-flop. It is driven by two non-overlapping clocks which can be controlled readily from the primary inputs to the circuit. Input DI is the normal data input to the SRL, clocks CK1 and CK2 control the normal operation of the SRL while clocks CK3 and CK2 control scan path movements through the SRL. The SRL output is derived at L2 in both modes of operation, the mode depending on which clocks are activated. The following advantages are claimed for the LSSD approach:

- The circuit operation is independent of the dynamic characteristics of the logic elements — rise- and fall-times and propagation delays.

- ATP generation is simplified since tests need only be generated for a combinational circuit.
- LSSD methods, when adopted in design, eliminate hazards and races; greatly simplifies test generation and fault simulation.

10.13.8.3 Boundary scan test (BST)

This is a technique involving scan path and self-testing to resolve the problems associated with the testing of boards carrying VLSI circuits and/or surface-mounted devices (SMD). Printed circuit boards (PCBs) are becoming very dense and complex, especially with SMD circuits, so that most test equipment cannot guarantee a good fault coverage.

BST consists of placing a scan path (shift register) cell adjacent to each component pin and to interconnect the cells so as to form a chain around the border of the circuit. The BST circuits contained on one board are then connected together to form a single path. The general idea is illustrated in Figure 10–44.

The boundary scan path is provided with serial input and output pads and appropriate clock pads which make it possible to:

- test the interconnections between the various chips on the board;
- deliver test data to the chips on the board for self-testing;
- test the chips themselves with internal self-test facilities.

BS techniques are grouped by the IEEE standards organization into a 'standard test access port and boundary scan architecture' (namely, *IEEE*, p. 1149.1–1990). The advantages of BST are seen as follows:

- no need for complex testers in PCB testing;
- the test engineer's work is simplified and efficient;
- the time spent on test pattern generation and application is reduced;
- fault coverage is increased.

10.13.8.4 Other scan design techniques

Many other stuctured approaches have evolved over the past few years, for example, *partial scan*, *scan/set* and *random access scan*.

The partial scan is derived from the scan path technique, but is less area-consuming. The scan path approach needs, on average, a 30% area increase for testing a whole sequential circuit. Using the partial scan approach, only faults not detected by the designer's functional vectors are selected. The test generator decides exactly which flip-flops,should be scanned.

In the scan/set method, the storage elements within the circuit are not used to implement a scan path. Instead, a separate register is added whose sole function is to scan test data in and out of the circuit. This allows for the main circuit under

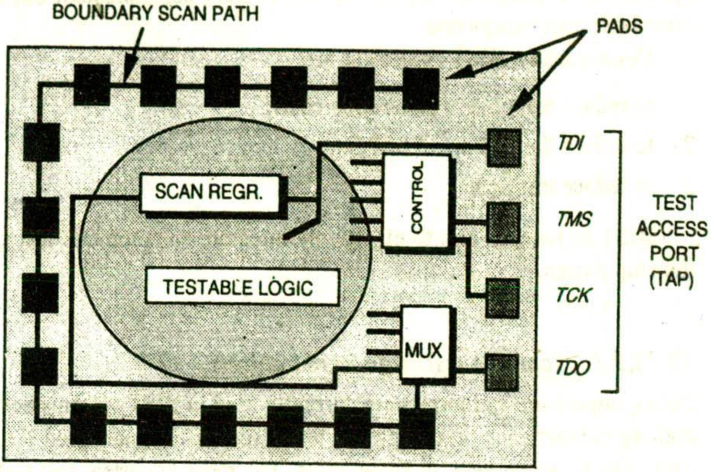


Figure 10-44 Boundary scan test (BST) configuration

test to be of any type — it is not restricted to combinational as before, and the storage elements are not restricted to particular types of latch or flip-flops. The major disadvantage of this technique is the high overhead cost in terms of additional input/output pins.

An overview of the other scan design techniques referred to here is presented in Table 10–8. However, many other scan design approaches exist but are mostly based on one or more of the methods discussed.

Table 10–8 Other scan design techniques

Partial scan	*Scan/set*
Scan: area increase	Separate shift register
Approach targets faults	No interruption to normal operation.
Selected flip-flops	No reduction to combinational test.

Random access scan

No shift register with flip-flops

Matrix of flip-flops addressed, controlled and observed

Disadvantage: The number of I/O pins

10.13.9 Built-in-self-test (BIST)

As the complexity of individual VLSI circuits and as overall system complexity increase, test generation and application becomes an expensive, and not always very effective, means of testing. Further, there are also very difficult problems associated with the high speeds at which many VLSI systems are designed to operate. Such problems require the use of very sophisticated, but not always affordable, test equipments.

Consequently, BIST objectives are:

1. to reduce test pattern generation costs;
2. to reduce the volume of test data;
3. to reduce test time.

BIST techniques aim to effectively integrate an automatic test system into the chip design.

10.13.9.1 Compact test: signature analysis

Data compression techniques are currently used in BIST systems and consist of making comparisons on compacted test responses instead of on the entire test data, which can be huge in some cases. The most important test task — is the circuit fault free? — is hence executed in an efficient manner.

The test compacting scheme currently used most is called *signature analysis.* This was developed by Hewlett Packard in the late 1970s'. Signature analysis performs polynomial division, that is to say division of the data out of the device under test (DUT). This data is represented as a polynomial $P(x)$ which is divided by a characteristic polynomial $C(x)$ to give the signature $R(x)$, so that

$$R(x) = P(x) / C(x)$$

This is summarized in Figure 10.45.

The signature from the DUT is compared with the expected signature to determine if the DUT is fault-free. The differences between the faulty signature and a good signature may also be used to indicate the nature of the fault. Signature analysis has been proved to be a reliable and attractive alternative to full uncompacted testing.

Another technique of data compression — *transition counting* — has been in use for some considerable time. This consists of counting transitions of a specified direction (0 to 1 or 1 to 0) and then comparing this count with the count obtained from the simulation model.

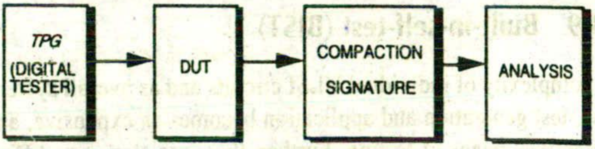


Figure 10–45 Built-in-self-test — signature analysis

10.13.9.2 Linear feedback shift register (LFSR)

The LFSR model is that of a finite state machine comprising storage elements and modulo two adders *(Xor gates)* connected in feedback loops as indicated in Figure 10–46.

LFSR techniques can be applied in a number of ways, including random number generation, polynomial division for signature analysis, and n-bit counting. LFSR can be either series or parallel, the differences being in the operating speed and in the area of silicon occupied — parallel LFSR being faster but larger than serial LFSR.

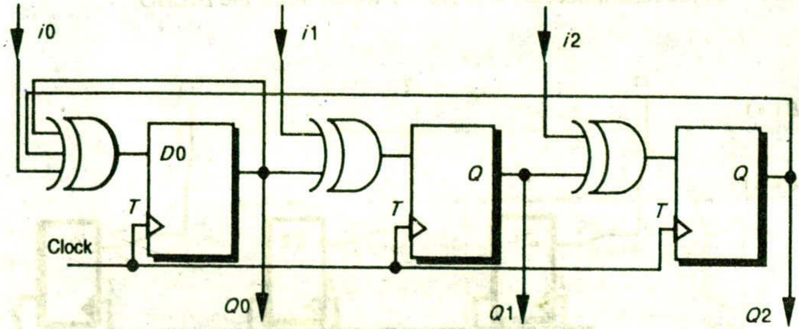


Figure 10–46 Built-in-test — linear feed-back shift register

10.13.9.3 Built-in logic block observer (BILBO)

BILBO is a built-in test generation scheme which uses signature analysis in conjunction with a scan path. It is aimed at integrated modular and bus-oriented systems, such as microprocessor and similar circuits.

The major component of a BILBO is an LFSR with a few gates. In Figure 10–47 the BILBO is controlled by two signals, B1 and B2 which define the modes.

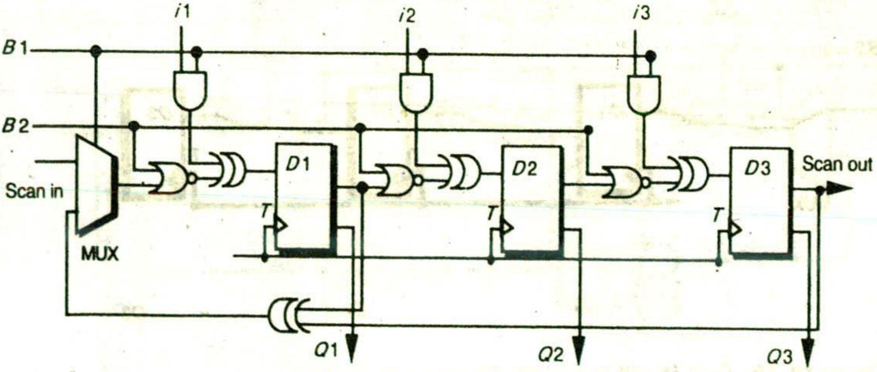


Figure 10–47 Built-in-self-test — Built-in logic block observer (BILBO)

In the *Normal* mode, B1 = B2 = 1 and the storage elements are used independently by the circuit as in Figure 10–48.

In the *Test 1* mode, B1 = B2 = 0 and the storage elements are configured as a scan path, all storage elements being connected as a serial shift register. This is shown in Figure 10–49. Test vectors are then applied to the scan-in input and responses shifted out at the scan path output. The analysis of data is then similar to that for a simple scan-path test.

In the *Test 2* mode, as in Figure 10–50, B1 = 1, B2 = 0 and the circuit is then configured in a LFSR mode and can be used either as a polynomial divider to compact data or as a random test pattern generator.

In the final mode, B1 = 0, B2 = 1 which *resets* the BILBO.

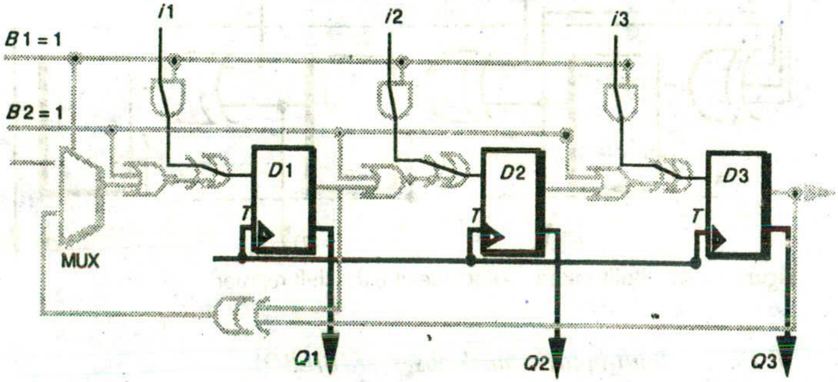


Figure 10–48 built-in-self-test — BILBO normal mode

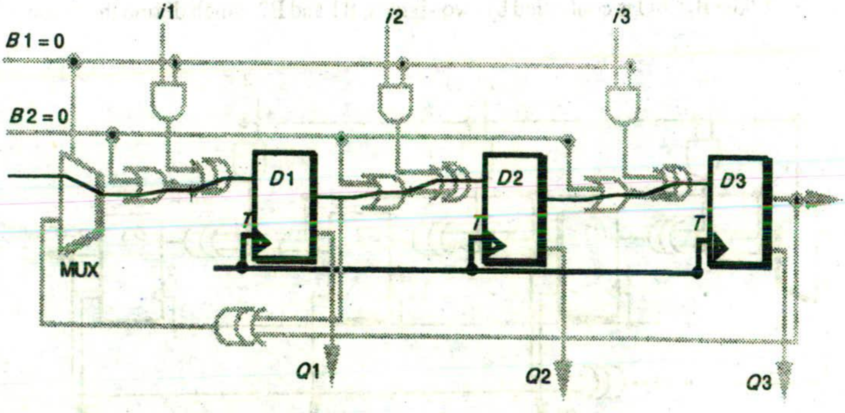


Figure 10–49 Built-in-self-test — BILBO scan path mode

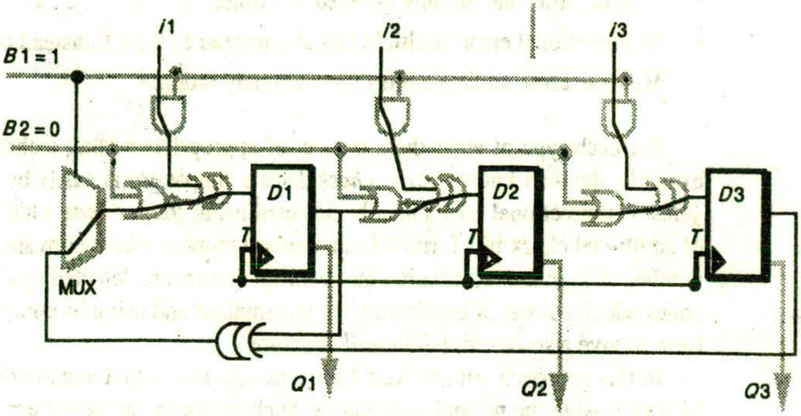


Figure 10–50 Built-in-self-test — BILBO: LFSR mode

10.13.9.4 Self-checking techniques

Data transmission in computer systems commonly makes use of coding to allow for the ready detection of errors. Such error detection techniques have been adapted and extended for built-in test purposes.

Self-checking techniques consist of supplying coded input data to the logic block under test and comparing the output in a checker designed to detect any errors. This is illustrated in Figure 10–51.

The design of logic blocks and checkers should then obey a set of rules in which the logic block is 'strongly fault secure' and the checker 'Strongly Code Disjoint'. A set of hypotheses is used in self-checking design to define the optimal design which allows a test coverage of 100% of these hypotheses.

The code used in data encoding depends on the type of errors that may occur at the logic block output. In general, three types are possible:

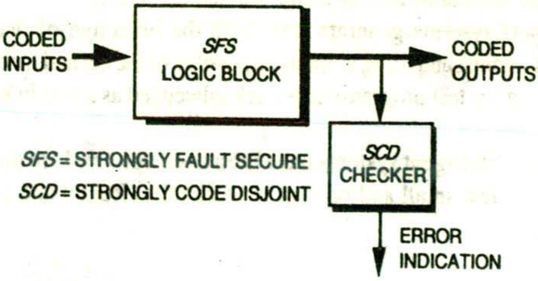


Figure 10–51 Self-checking logic: coding techniques

- Simple error: one bit only affected at a time.
- Unidirectional error: multiple bits at 1 instead of 0 (or 0 instead of 1).
- Multiple error: multiple bits affected in any order.

For each type of error there is a set of appropriate coding techniques. For example, the well-known *parity check* detects simple errors easily by using *Xor* gates. Unidirectional errors may be detected using *Berger code* which consists of additional check bits formed from a binary number which corresponds to the number of 1s in the information bits. Multiple errors are detected by duplication codes which consist of duplicating the information and using its complementary form to give a so-called *double rail* structure.

In this approach, all checkers have a double rail output and are designed to be tested using the normal code inputs. Such checkers can detect errors in their own operation.

Self-checking techniques are applied to circuits in which security is important so that fault tolerance is of major interest. Such techniques will occupy considerably more area in silicon than classical techniques such as functional testing but provide a very high test coverage.

10.13.10 Future trends

Observability and controllability are dramatically improved with new techniques using non-destructive E-beam probes which eliminate any probe capacitive loading. CAD tools are often used, and voltage contrast methods allow an immediate fault diagnosis.

Scanning E-beam microscopes have many test advantages such as the ability to function at very high clock speeds (GHz range) and at very high resolution (e.g. 0.3 um), and finally with a wide range of CAD tools. The main disadvantage is their very high cost.

Expert CAD tools for IC design are being developed and are based on the hard-earned experience of test engineers, performing test pattern generation in the same manner as an individual test engineer does.

Expert systems generate tests with the objective of detecting all possible faults of interest and using as many as possible of the normal mode circuit operations. Consequently, test programming is acknolwedged as a highly knowledge-intensive task.

Many intelligent systems have been developed and are distinguished by their abilitity to test small and/or structured circuits (Russel and Sayers, 1989).

10.14 References

(a) Chip design

Ayres, R. F. (1983) *VLSI Silicon Compilation and the Art of Automatic Microchip Design*, Prentice Hall, Englewood Cliffs, NJ.

Hon, R. W., and Sequin, C. M. (1980) *A Guide to LSI Implementation*, 2nd edn, Xerox Press, EL Segundo, Calif.

Newkirk, J. A., and Mathews, R. G. (1984) *The VLSI Designer's Library*, Addison-Wesley Publishing Co. Inc., Reading, Mass.

Sung Mo Kang, (1981) 'A design of CMOS polycells for LSI circuits', *IEEE Transactions on Circuits and Systems*, Vol. CAS-28, No. 8, 838–43.

(b) Digital systems optimization and testing

Agrawal, V. D., and Seth, S. C. (1988) *Tutorial-Test Generation for VLSI Chips*, Computer Society Press.

Amersekera, E. A., and Campbell, D. S. (1987) *Failure Mechanisms in Semiconductor Devices*, Wiley.

Antreich, K. J. and Huss, S. A. (1984) 'An interactive optimization technique for the nominal design of integrated circuits', *IEEE Transactions on Circuits and Systems*, Vol. CAS-31, No. 2, 203–12.

Bardell, P. H., McAnney W. H. and Savir, J. (1987) *Built-In Test for VLSI: Pseudorandom Techniques*, Wiley.

Bateson, J. (1985) *In-circuit Testing*, Van Nostrand Reinhold.

Bennetts, R. G. (1982) *Introduction to Digital Board Testing*, Crane Russak.

Bennetts, R. G. (1984) *Design of Testable Logic Circuits*, Addison-Wesley.

Bhattacharya, Debashis and Hayes, John P. (1990) *Hierarchical Modeling for VLSI Circuit Testing*, Kluwer Academic Publishers.

Brayton, R. K. et al. (1981) 'A survey of optimization techniques for integrated circuit design', *Proc. IEEE*, Vol. 69, 1334–62.

Breuer, M. A., (ed.) (1972) *Design Automation of Digital Systems*, Prentice Hall.

Breuer, M. A., and Friedman, A. D. (1976) *Diagnosis and Reliable Design of Digital Systems*, Computer Science Press.

Chang, H. Y., Manning, E. G. and Metze, G. (1970) *Fault Diagnosis of Digital Systems*, Wiley Interscience.

Cortner, J. M. (1987) *Digital Test Engineering*, Wiley.

Davis, B. (1982) *The Economics of Automatic Testing*, McGraw-Hill, London.

Einspruch, N. G. (1985) *VLSI Handbook*, Academic Press.

Fee, W. G. (1978) *Tutorial-LSI Testing*, 2nd edn, Computer Society Press.

Feugate, R. J., and McIntyre, S. M. (1988) *Introduction to VLSI Testing*, Prentice Hall.

Friedman, A. D., and Menon, P. R. (1971) *Fault Detection in Digital Circuits*, Prentice Hall.

Fujiwara, H. (1985) *Logic Testing and Design for Testability*, MIT Press.

Golomb, S. W. (1982) *Shift Register Sequences*, revised edn, Aegean Park Press.

Greason, W. D. (1987) *Electrostatic Damage In Electronics*, Wiley.

Healy, J. T. (1981) *Automatic Testing and Evaluation of Digital Circuits*, Reston Publishing.

Jensen, F., and Petersen, N. E. (1982) *Burn-In*, Wiley, Chichester, UK.

Karpovsky, M. G. (ed.) (1985) *Spectral Techniques and Fault Detection*, Academic Press.

Kohavi, Z. (1978) *Switching and Automata Theory*, McGraw-Hill.

Lala, P. K. *Fault-Tolerant and Fault Testable Hardware Design*, Prentice-Hall, London, UK.

Lightner, M. R. and Director, S. W. (1981) 'Multiple criterion optimization with yield maximization', *IEEE Transactions on Circuits and Systems,* Vol. CAS-28, No. 8, 781–91.

Mahoney, M. (1987) *DSP-Based Testing of Analog and Mixed-Signal Circuits (Tutorial)*, Computer Society Press.

McCluskey, E. J. (1986) *Logic Design Principles with Emphasis on Testable VLSI Circuits*, Prentice Hall.

Miczo, A. (1986) *Digital Logic Testing and Simulation*, Harper & Row.

Miller, D. M. (ed.) (1987) *Developments in Integrated Circuit Testing*, Academic Press.

Needham, W. (1991) *Designer's Guide to Testable ASIC Devices*, Van Nostrand Reinhold (International), London.

Niraj, K. J., Sandip Kundu (1990) *Testing and Reliable Design of CMOS Circuits*, Kluwer Academic Publishers.

Parker, K. P. (1987) *Integrating Design and Test: Using CAE Tools for ATE Programming*, Computer Society Press.

Pradham, D. K. (ed.) (1986) *Fault-Tolerant Computing: Theory and Techniques*, Vols I and II, Prentice Hall.

Pynn, C. (1986) *Strategies for Electronics Test*, McGraw-Hill.

Reghbati, H. K. (1985) *Tutorial: VLSI Testing and Validation Techniques*, IEEE Computer Society Press, North Holland.

Ronse, C. (1984) *Feedback Shift Registers*, Springer-Verlag.

Roth, J. P. (1980) *Computer Logic, Testing, and Verification*, Computer Science Press.

Ruen-wen-liu, (1979) *Testing and Diagnosis of Analog Circuits and Systems*, Van Nostrand Reinhold (International), London.

Russel, G., and Sayers, I. L. (1989) *Advanced Simulation and Test Methodologies for VLSI Design,* Van Nostrand Reinhold (International), London.

Singh, N. (1987) *An Artificial Intelligence Approach to Test Generation*, Kluwer Academic Publishers.

Stevens, A. K. (1986) *Introduction to Component Testing*, Addison-Wesley.

Stover, A. C. (1984) *ATE: Automatic Test Equipment*, McGraw-Hill.

Timoc, C. C. (1984) *Selected Reprints on Logic Design for Testability*, Computer Science Press.

Tsui, F. F. (1986) *LSI-VLSI Testability Design*, McGraw-Hill.

Turino, J. L. (1991) *Design to Test*, 2nd edn, Van Nostrand Reinhold (International), London.

Wilkins, B. R. (1986) *Testing Digital Circuits: An Introduction*, Van Nostrand Reinhold, Berkshire, UK.

Williams, T. W. (1986) *VLSI Testing*, North-Holland, Amsterdam.

Yarmolik, V. N. (1990) *Fault Diagnosis of Digital Circuits*, John Wiley & Sons.

11 Some CMOS design projects

You cannot create experience — you must undergo it.

Albert Camus

Practice makes perfect.

Proverb

And for instructors:

Practice what you preach.

Proverb

Objectives

The 'raison d'être' for this chapter is self-evident. The authors regard project work and the tutorial work which leads up to it as absolutely essential to effective learning. The way in which the five individual projects are approached will, it is hoped, provide further insight into VLSI design processes. The projects have been chosen for their diversity and also to provide designs for useful and practical circuits.

11.1 Introduction to project work

The design exercises tackled earlier in this text were chosen to illustrate the design processes and to introduce the reader to the type of problems suitable for introducing design in silicon and relevant to everyday applications.

Following on from the design processes introduced, it is now instructive to formally tackle CMOS design work on some complete subsystems, and to this end various projects are now tackled in this chapter.

11.2 CMOS project 1 — an incrementer/decrementer

The design to be pursued is that of a 4-bit incrementer/decrementer, but the design is general in that the standard cell envisaged can be cascaded at will to *n*-bits.

11.2.1 Behavioral description

The truth table for a binary 1-bit incrementer is shown in Figure 11–1, where C_i is the carry bit from the previous stage, *Cl* is the clock input, C_{i+1} is the carry bit output, and Q_n is the stage output.

The logic expressions for the incrementer are as follows:

$$Q_n = C_i \oplus Q_{n-1} \tag{11.1}$$

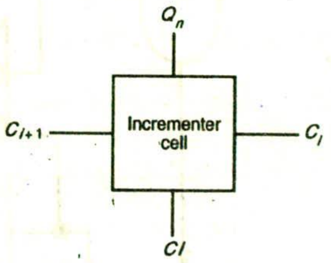
$$C_{i+1} = C_i \cdot Q_{n-1} \tag{11.2}$$

The *n* stages are isolated by the clock signal *Cl*, and it will be seen that the truth table assumes positive-edge clocking. A reset signal (*Res*) should also be provided for the incrementer to be able to start from zero at any instant in time.

For the incrementer to function as a decrementer the additional equation that needs to be implemented is as follows:

Truth table

Inputs			*Outputs*	
Cl	C_i	Q_{n-1}	Q_n	C_{i+1}
0	0	0	0	0
1	0	0	0	0
0	1	0	0	0
1	1	0	1	0
0	0	1	0	0
1	0	1	1	0
0	1	1	0	0
1	1	1	0	1



Note: Where Q_{n-1} is state of output prior to clocking.

Figure 11–1 1-bit incrementer cell

$$C_{i+1} = C_i \cdot \overline{Q_{n-1}} \tag{11.3}$$

A particular, but not the only possible, approach to designing this subsystem follows. For those readers who wish to 'fly solo' in tackling this design, the next project follows in section 11.3 on p. 367.

11.2.2 Structural description

11.2.2.1 Logic representation

An incrementer/decrementer cell is realized by direct implementation of expressions (11.1), (11.2), and (11.3) as in Figure 11–2, for example. Note that a reset control line may also be added using the *'clear'* input of the flip-flop to enable the circuit to start from zero at any time, but this is not shown in the figure. The control line which is required to set the circuit operation to that of an incrementer or a decrementer is shown in the figure.

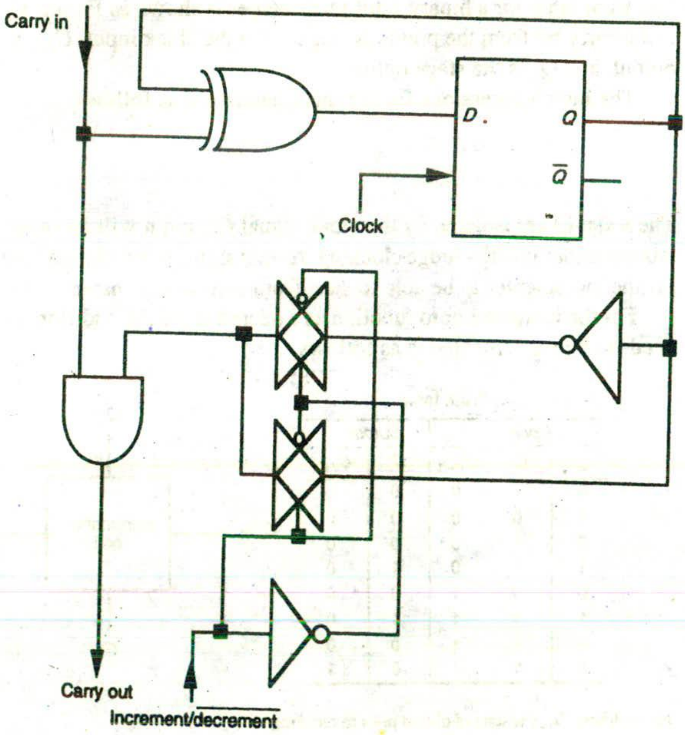


Figure 11–2 Logic diagram for an incrementer/decrementer cell

11.2.2.2 Operation of the circuit

The circuit functions like an adder or a subtractor with one of its three inputs set to zero. The cell uses its current state as one input and the carry in from the previous stage as the other input. The current state and the carry out are modified according to the two inputs on clocking.

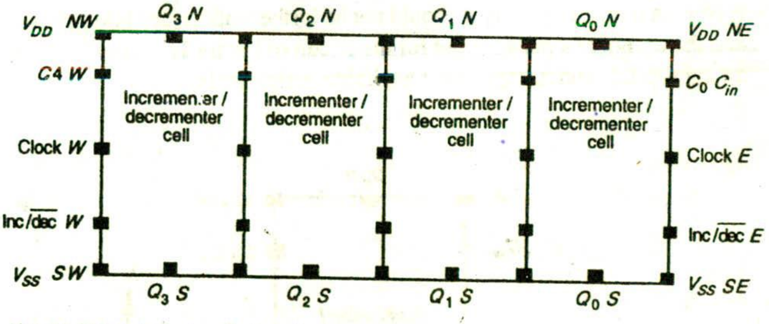
11.2.2.3 Critical paths

The critical delay in this circuit is the propagation delay of the carry bit — analogous to the adder situation. Since the circuit is clocked, the minimum allowable clock period is set by the maximum circuit delay; in this case the time that the carry bit needs to propagate from the first to the last stage. This will be reasonably fast as the carry bit passes through only one *And* gate per stage.

11.2.3 Physical description

11.2.3.1 Floor plan of a 4-bit incrementer/decrementer subsystem

The 4-bit incrementer/decrementer is realized by abutting four identical cells. The height of the complete subsystem remains constant while the width grows linearly with *n* — the number of bits. Therefore the width of each cell should be made as small as possible. The control lines run right across the whole structure and adequate driving capability should be supplied when *n* is significant. The resulting floor plan is shown in Figure 11–3.


V_{DD} NW
Q_3 N
Q_2 N
Q_1 N
Q_0 N
V_{DD} NE
C4 W
C_0 C_{in}
Incremen.er / decrementer cell
Incrementer / decrementer cell
Incrementer / decrementer cell
Incrementer / decrementer cell
Clock W
Clock E
Inc/$\overline{\text{dec}}$ W
Inc/$\overline{\text{dec}}$ E
V_{SS} SW
V_{SS} SE
Q_3 S
Q_2 S
Q_1 S
Q_0 S

Note: N, SW etc. indicate cell orientation (compass points).

Figure 11–3 Floor plan for a 4-bit incrementer/decrementer

If the width of the leaf-cell is *w*, then the width of an *n*-bit incrementer/decrementer is *nw*. This dimension must be pitch-matched to the rest of the system into which the incrementer/decrementer is to fit (e.g. a VLSI processor, etc.), which may be assumed to be of width *W*. Therefore

$$w = W/n$$

provided that

$$w_{min} \leqslant W/n$$

where w_{min} is the minimum width of a cell. In the event that $w_{min} > W/n$, then the design must be adjusted to be thinner and taller, otherwise the width W of all mating subsystems may have to be increased.

11.2.3.2 Leaf-cell floor plan

The floor plan of the 4-bit incrementer/decrementer basically determines the floor plan of the leaf-cell which is given in Figure 11–4.

The width w of each cell is set by the total allowable maximum incrementer/decrementer width W which cannot be exceeded if the circuit is to be properly pitch-matched to the rest of the system, e.g. data path, for which it is being designed. The minimum height h of the leaf-cell is set by its complexity once the width w has been fixed. The decision about the output connection and the power rail placements is made at the subsystem level (the subsystem here being the four-bit incrementer/decrementer).

In a complex design, the number of leaf-cells should be kept to the absolute minimum, which implies that the complexity of the leaf cells should be as high as possible. This greatly simplifies the global floor plan, but it must be recognized that the available design tools will determine the maximum size of leaf-cell which can be readily handled. In general, a 50 to 100 transistor leaf-cell can be readily realized with available design tools. Since the incrementer/decrementer leaf-cell is of a medium complexity it should not be further subdivided into sub-leaf-cells, and the design of a mask layout for the circuit of Figure 11–2 may be pursued for an appropriate technology using available design tools.

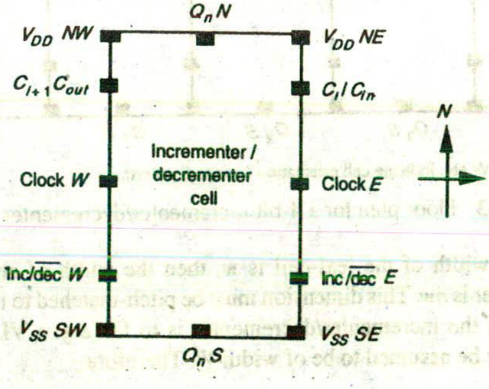


Figure 11–4 Floor plan of incrementer/decrementer leaf-cell

11.2.4 Design verification

The leaf-cell circuit was designed using 5 μm p-well CMOS technology and a mask layout arrived at. The detail present in the CIF code specification for the mask layout was extracted with a circuit extractor (NET) and then a two-bit subsystem simulated with a circuit simulator (PROBE). The simulation results are given in Figure 11–5.

11.3 CMOS project 2 — left/right shift serial/parallel register

This project is concerned with the design of a general purpose shift register cell capable of expansion to form an *n*-bit register.

11.3.1 Behavioral description

Table 11–1 defines the shift register connections that apply to Figures 11–6, 11–7 and 11–8. The logic circuit for a suitable single shift register leaf-cell is shown in Figure 11–7 and in block diagram form in Figure 11–8.

Table 11–1 Shift register control functions

Controls	*Function*	*Conditions required*
dp	parallel data input	latched when *dprl* is asserted
dprl	parallel input data control	$\overline{\mathit{left}} \cdot \overline{\mathit{right}}\ \phi_1$
qp	parallel data output	valid when *qprl* is asserted
qprl	parallel output data control	data valid on ϕ_2 of clock
ds	serial right data input	valid when *right* is asserted
qs	serial right data output	valid when *right* is asserted
right	shift right control	$\overline{\mathit{dprl}} \cdot \overline{\mathit{left}} \cdot \phi_1$
leftin	serial left data input	valid when *left* is asserted
leftout	serial left data output	valid when *left* is asserted
left	shift left control	$\overline{\mathit{dprl}} \cdot \overline{\mathit{right}} \cdot \phi_1$
fb	internal refresh control	$\overline{\mathit{dprl}} \cdot \overline{\mathit{right}} \cdot \overline{\mathit{left}} \cdot \phi_1$
ϕ_2	second clock phase	data latch to output node

11.3.2 Structural description

11.3.2.1 Logic representation

The complete 4-bit shift register is made up of single shift register cells abutted as shown in part in Figure 11–6.

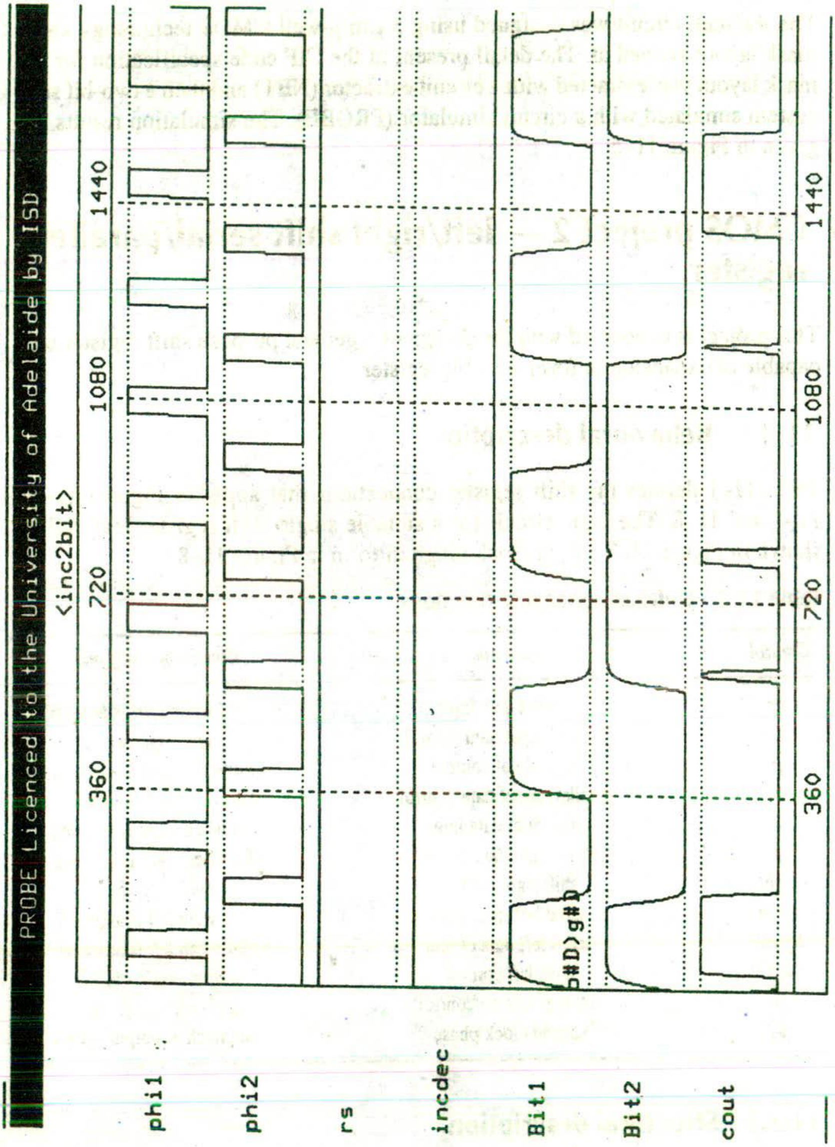


Figure 11–5 Simulation results for a 2-bit system

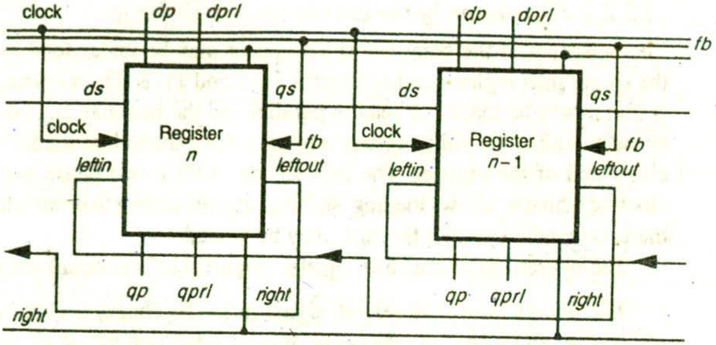


Figure 11–6 Two-bit shift register block diagram

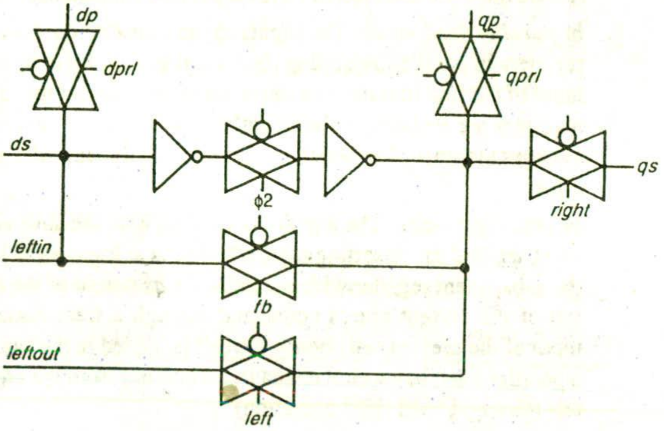


Figure 11–7 Shift register cell logic diagram

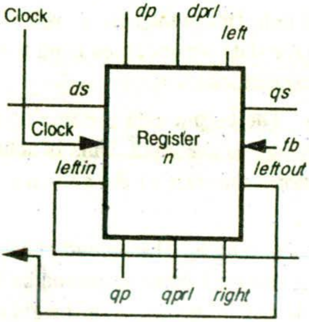


Figure 11–8 Shift register cell block diagram

11.3.2.2 Operation of the circuit

The operation of the complete shift register may be understood by considering the single shift register cell of Figures 11–7 and 11–8. The advantage of this cell is that it may be loaded or read in parallel and the bits may be shifted either left or right within the shift register and an output thus obtained in serial form at either end of the register. The register also uses a two-phase non-overlapping clock of which ϕ_1 allows loading, shifting, and refreshing to occur while ϕ_2 isolates the two inverters so that the cells may be loaded.

The operations of the shift register (Figure 11–7) in detail are as follows:

1. *The refresh loop.* The refresh signal *fb* (or feedback) occurs in coincidence with ϕ_1 and when no other control is asserted (namely *dprl, right,* and *left).* The transmission gate takes the output of the second inverter and uses it tc refresh the logic level stored on the input gate capacitances of the first inverter.
2. *In parallel load mode.* The inputs *dp* and control *dprl* are used to load the registers in parallel. Asserting *dprl* when ϕ_1 is at logic level 1 will cause the input of the first inverter to assume the state of *dp*. At this time $\phi_2 = 0$ and the inverters are isolated. Subsequently $\phi_2 = 1$ and the second inverter output assumes the state of *dp* which has been stored dynamically at the first inverter input.
3. *In shift right mode.* The signals associated with the shift right operation are *right, qs,* and *ds*. Asserting *right* when ϕ_1 is at logic level 1 effectively loads the subsequent register with *qs,* while the *qs* output of the register cell to the left of the current one is connected through a transmission gate to the *ds* input of the present cell. Hence the cell is loaded in the same manner as with a parallel load, but with the data input coming from the adjoining cell to the left (that is, a shift right operation).
4. *In shift left mode.* The signals associated with the shift left operation are *left, leftout,* and *leftin.* Asserting *left* when ϕ_1 is a logic level 1 effectively loads the previous register with *qs* via the line *leftout.* The register cell to the right of the current one has its *leftout* connected through a transmission gate to *leftin* of the present cell. Hence the cell is loaded in the same manner as with a parallel load but the data input comes from the adjoining cell to the right (that is, a shift left operation).
5. *For parallel output.* The output data is correctly read at the end of ϕ_2 when there can be no change to the input. This is achieveo oy asserting *qprl*, in which case *qp* assumes the state of the cell and all outputs are then read in parallel
6. *Isolation of the inverters by* ϕ_2. The second phase of the clock (ϕ_2) is used to isolate the inverters during a write operation so that the register array does not become 'transparent'. Consider a shift right operation but allow $\phi_2 = 1$.

Here the first inverter output would become $\overline{ds}_{i-1}$ (from the next left cell). The second inverter output would thus become ds_{i-1}. However, since ϕ_2 is logic 1, ds_{i-1} can now be passed on to cell $i+1$, since *right* is asserted and $qs = ds_{i-1}$. Hence the register has become transparent and ds_{i-1} would ripple throughout the entire array. This undesirable effect is eliminated by loading and coupling inverter pairs on separate clock phases.

11.3.2.3 Critical paths

The system is restricted to shifts of 1 bit only in either direction and hence any shifts of more bits will take proportionally more time. In this case, there is a minimum time t_1 for which ϕ_1 must be asserted to allow the data to be stored at the first inverter input gate. After this delay the data is passed to the output on ϕ_2 which must have time duration t_2 for the second inverter input capacitance charge to change its state if required. The total delay (T) is governed by the sum of t_1 and t_2 and the number of shifts n required (i.e. $T = n \cdot (t_1 + t_2)$). To reduce this delay a fast shifting cell is required.

The most critical path at the leaf-cell level is associated with the output of the second inverter which must drive four transmission gate input capacitances. For this reason the second inverter is not usually made minimum size. Note, however, that the second inverter cannot be made too large since the first inverter (which is minimum sized) must drive its input when $\phi_2 = 1$. The final sizing of the transistors may be determined after a series of simulations following circuit extraction from the mask layout.

11.3.3 Physical description

11.3.3.1 System floor plan

The 4-bit shift register may be formed by abutting four identical 1-bit register cells. The most convenient arrangement for an n-bit shift register is to have the parallel data inputs and outputs running perpendicular to the direction of the register array. The control lines are also conveniently run perpendicular to the register array but, on exiting a register cell, may be run along the array with appropriate connections made to adjoining cell control signals. The power rails must be implemented in metal and also run perpendicular to the parallel input/output data. The resulting floor plan is shown in Figure 11–9.

If the width of the leaf-cell is w, then the width of an n-bit register is nw. This dimension must be pitch-matched to the rest of the system (e.g. a VLSI processor, etc.) of assumed width W. Therefore

$$w = W/n$$

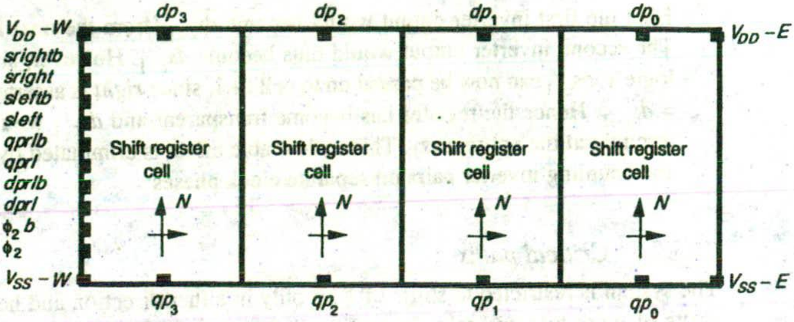


Figure 11–9 Proposed floor plan — 4-bit shift register

11.3.3.2 Leaf-cell floor plan

The floor plan of the 4-bit shifter basically specifies the floor plan of the leaf-cell. The width *w* is set by the total maximum register width, and this cannot be exceeded if the register is to be properly pitch-matched, for example, to a processor. The minimum height *h* of the leaf-cell is set by its complexity once the width has been fixed. The decision about the input/output connection and the power rail placements is made at the system's level.

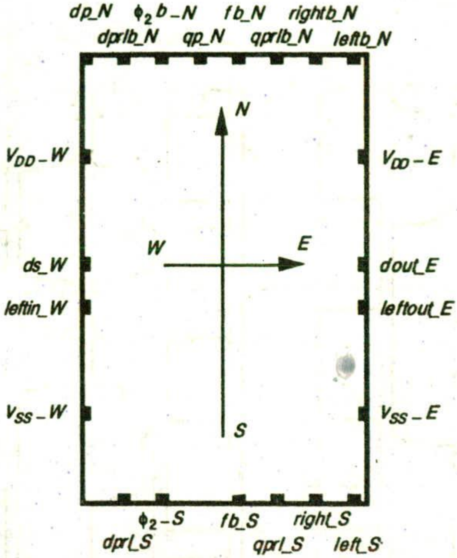
In a complex design, the number of leaf-cells should be kept to the absolute minimum, which implies that the complexity of the leaf-cells should be as high as possible. This greatly simplifies the global floor plan. As stated earlier, a 50 to 100 transistor leaf-cell can usually be readily realized with commonly available design tools. The register leaf-cell described here is of small/medium complexity and thus should not be further subdivided into sub-leaf-cells. The shift register leaf-cell floor plan is shown in Figure 11–10.

11.3.4 Design verification

Simulation results for a 4-bit register realized in 5 μm p-well CMOS technology (using PROBE software) are presented in Figure 11–11.

11.4 CMOS project 3 — a comparator for two *n*-bit numbers

This section describes the design methodology, layout strategy, and simulation results for cascadable comparator cells. A 4-bit comparator was designed using these cells, the general arrangement being as suggested in Figure 11–12.



Compass points, used to indicate orientation of cell, may also be appended to signals to indicate position as shown.

Figure 11–10 Shift register cell floor plan

11.4.1 Behavioral description

The truth table and general arrangement for a binary 1-bit comparator bit-slice is shown in Figure 11–13 where A_i and B_i are the two numbers to be compared, C_{i+1} and D_{i+1} are the inputs from outputs of the previous stage, and C_i and D_i are the outputs of the current stage. $C_i = 1$ if $A_i > B_i$; $D_i = 1$ if $A_i < B_i$; and $C_i {}_= D_i = 0$ if $A_i = B_i$.

The logic expressions for the two output signals in terms of the four input signals are as follows.

$$C_i = C_{i+1} + \overline{C}_{i+1} \cdot A_i \cdot \overline{B}_i \cdot \overline{D}_{i+1} \tag{11.4}$$

$$D_i = D_{i+1} + \overline{D}_{i+1} \cdot \overline{A}_i \cdot B_i \cdot \overline{C}_{i+1} \tag{11.5}$$

The two logic expressions may be rearranged into the form

$$\begin{aligned} C_i &= \overline{\overline{C}_{i+1} \cdot \overline{(\overline{\overline{C}}_{i+1} \cdot \overline{D}_{i+1} \cdot A_i \cdot \overline{B}_i)}} \\ &= \overline{\overline{C}_{i+1} \cdot \overline{\overline{(C_{i+1} + \overline{A}_i + B_i)} \cdot \overline{D}_{i+1}}} \end{aligned} \tag{11.4a}$$

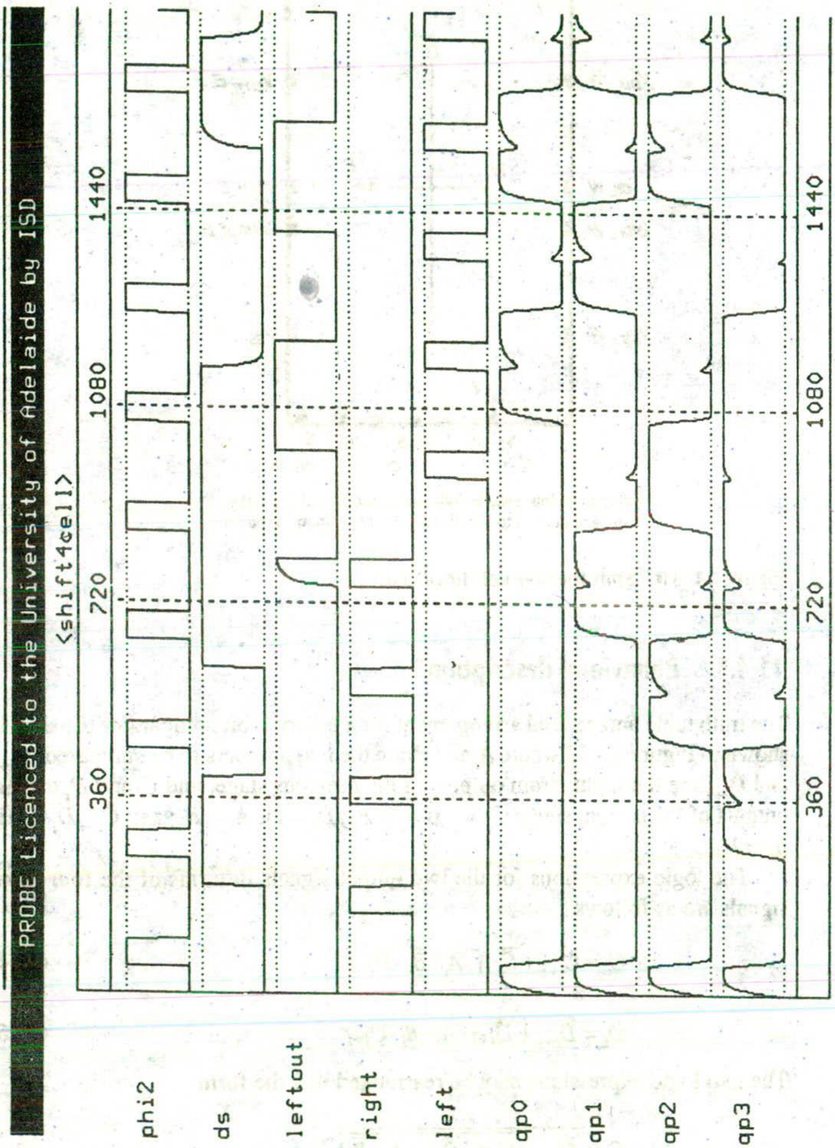


Figure 11–11 Simulation over four shift register cells

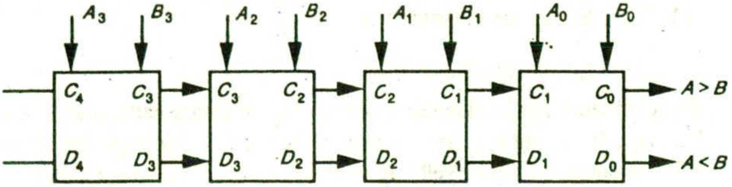


Figure 11–12 4-bit comparator — block diagram

Truth table

Inputs				*Outputs*	
A_i	B_i	C_{i+1}	D_{i+1}	C_i	D_i
X	X	1	0	1	0
X	X	0	1	0	1
0	0	0	0	0	0
0	1	0	0	0	1
1	0	0	0	1	0
1	1	0	0	0	0

X signifies 'don't care'

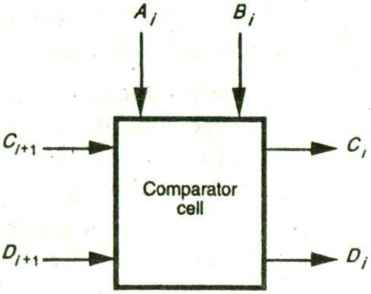


Figure 11–13 Comparator cell behavior

$$\begin{aligned} D_i &= \overline{\overline{D}_{i+1}\cdot(\overline{\overline{\overline{D}}_{i+1}\cdot\overline{C}_{i+1}\cdot\overline{A}_i\cdot B_i})} \\ &= \overline{\overline{D}_{i+1}\cdot\overline{\overline{(D_{i+1}+\overline{B}_i+A_i)}\cdot\overline{C}_{i+1}}} \end{aligned} \qquad (11.5a)$$

A further simplification may be achieved if alternate logic is used between subsequent cells. Stage i implements

$$\overline{C}_i = \overline{C}_i = \overline{C_{i+1}+\overline{D_{i+1}+\overline{A}_i+B_i}} \qquad (11.4b)$$

$$\overline{D}_i = \overline{D_{i+1}+\overline{C_{i+1}+A_i+\overline{\overline{B}}_i}} \qquad (11.5b)$$

and stage $i-1$ implements

$$C_{i-1} = \overline{\overline{C}_i\cdot\overline{\overline{D}_i\cdot A_{i-1}\cdot\overline{B}_{i-1}}} \qquad (11.4c)$$

$$D_{i-1} = \overline{\overline{D}_i\cdot\overline{\overline{C}_i\cdot\overline{A}_{i-1}\cdot B_{i-1}}} \qquad (11.5c)$$

11.4.2 Structural description

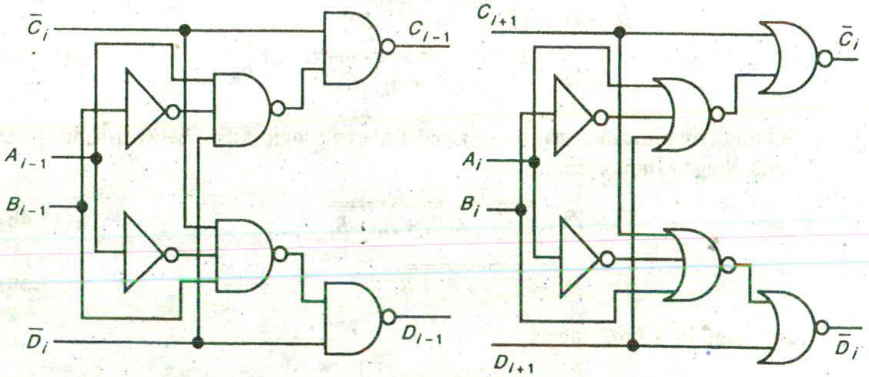
11.4.2.1 Logic representation

The comparator is implemented with complementary cells, that is, the *ith* stage has true inputs and inverted outputs while the $(i+1)th$ stage has inverted inputs and true outputs. The two cells are realized by direct implementation of expressions (11.4c) and (11.5c) (COMPCELLA) and expressions (11.4b) and (11.5b) (COMPCELLB) as shown in Figures 11–14 (a) and (b) respectively.

11.4.2.2 Operation of the circuit

The operation of the complete circuit is as follows:

- The two numbers are compared starting with the most significant bits. The outputs from this comparison are connected to the next most significant bit stage inputs etc. The two output signals C_i and D_i remain at zero as long as the two bits being compared are the same.
- As soon as a difference is detected, the two outputs are set to one of two possible states: if $A_i > B_i$ then $C_i = 1$ and $D_i = 0$; if $A_i < B_i$ then $C_i = 0$ and $D_i = 1$.
- All the remaining pairs of less significant bits then have no further effect on the state of subsequent outputs C_i and D_i.
- If all pairs of bits of the two numbers being compared are equal, then the outputs stay at zero signifying equality.



(a) Cell 'COMPCELLA'

(b) Cell 'COMPCELLB'

Figure 11–14 Comparator — logic diagram

11.4.2.3 Critical paths

The critical delay in this circuit is the propagation delay of the two outputs through all the stages. The gates passing both outputs should be sized appropriately. The delay is only one gate per stage and should not be the limiting factor on a system's scale. The final sizing of the transistors is usually determined after a series of simulations.

11.4.3 Physical description

11.4.3.1 System floor plan

The 4-bit comparator is realized by abutting cells of each type on an alternate basis. One possibility would be to have both bit inputs on the same side of a cell with the two outputs propagating at right angles to the input data path. Another possible layout would be to have the two bit inputs on opposite sides of a cell. The second approach was adopted here. The height of the comparator remains constant while the width grows linearly with n — the number of bits. Therefore the width of each cell should be made as small as possible.

A possible floor plan is shown in Figure 11–15: the inputs A_i and B_i come in at the top and bottom of each cell respectively, and C_i and D_i propagate horizontally. V_{DD} and V_{SS} rails may also propagate horizontally in global terms but may be distributed at right angles within a cell if convenient.

If the width of the leaf-cell is w, then the width of an n-bit comparator is nw. This dimension must be pitch-matched to the rest of the system (e.g. a VLSI processor, etc.) of width W. Therefore

$$w = W/n$$

11.4.3.2 Leaf-cell floor plan

The floor plan of the 4-bit comparator basically specifies the floor plan of the leaf-cells as shown in Figure 11–16. The width w is set by the total maximum

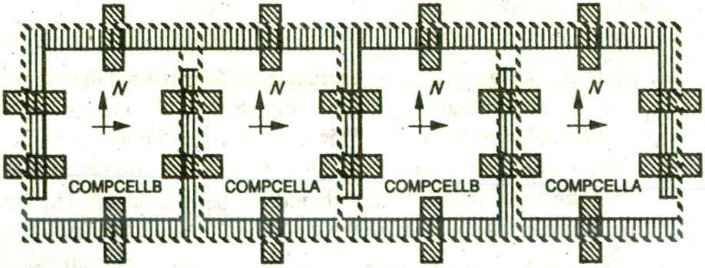


Figure 11–15 Proposed floor plan — 4-bit comparator showing shared power rails

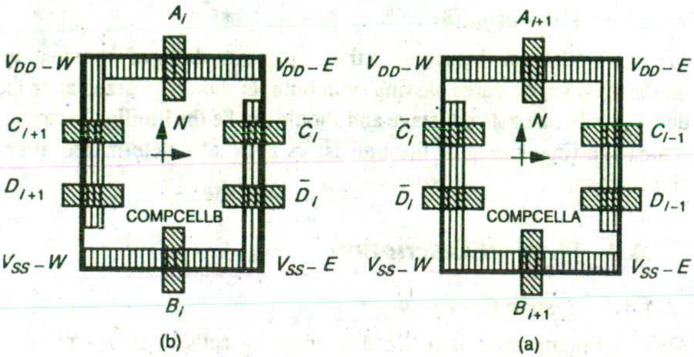


Figure 11–16 Comparator leaf-cells — floor plan

comparator width W. The minimum height h of the leaf-cell is set by its complexity once the width w has been fixed. The decision about the input/output connection and the power rail placements is made at the system's level (the system here being the 4-bit comparator).

In a complex design the number of leaf-cells should be kept to the absolute minimum, which implies that the complexity of the leaf-cells should be as high as possible. This greatly simplifies the global floor plan. A 50 to 100 transistor leaf-cell can usually be readily realized with available design tools. The comparator leaf-cell is of medium complexity and does not require any further subdivision.

11.4.4 Symbolic or stick representation to mask transformation

A mask representation is generally obtained from a symbolic form of cell specification by the process of compaction. A compactor is a tool that takes a symbolic representation of the given cell and produces a mask description of the cell according to some predefined set of process design rules. A mask description of the cell may also be obtained by direct mapping from a stick diagram using a mask level graphics editor.

A few basic rules should be observed when designing a circuit:

1. Start the design by placing an imaginary demarcation line (for p-well CMOS, this is closely related to the top edge of the well, and for n-well CMOS, the bottom edge of the well). This line separates the p-type devices, which are placed above it, from the n-type devices, which are placed below it; that is, the two types of transistors should not be intermixed. This style of design allows easy placements of the well and the p+ or n+ masks (Figure 11–17).
2. Keep the V_{DD} and V_{SS} supply rails well separated. This allows all the devices to be placed close to the required rail and be completely within the V_{DD} to V_{SS} boundaries, greatly simplifying the inter-cell connections.

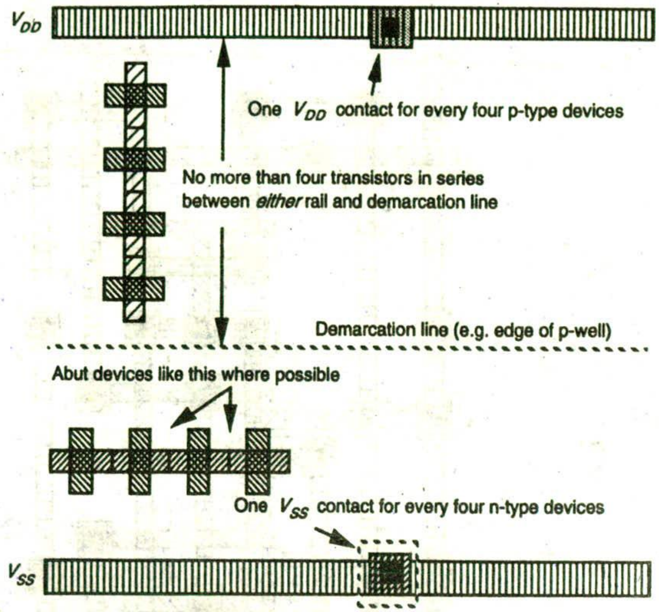

V_{DD}
One V_{DD} contact for every four p-type devices
No more than four transistors in series
between *either* rail and demarcation line
Demarcation line (e.g. edge of p-well)
Abut devices like this where possible
One V_{SS} contact for every four n-type devices
V_{SS}

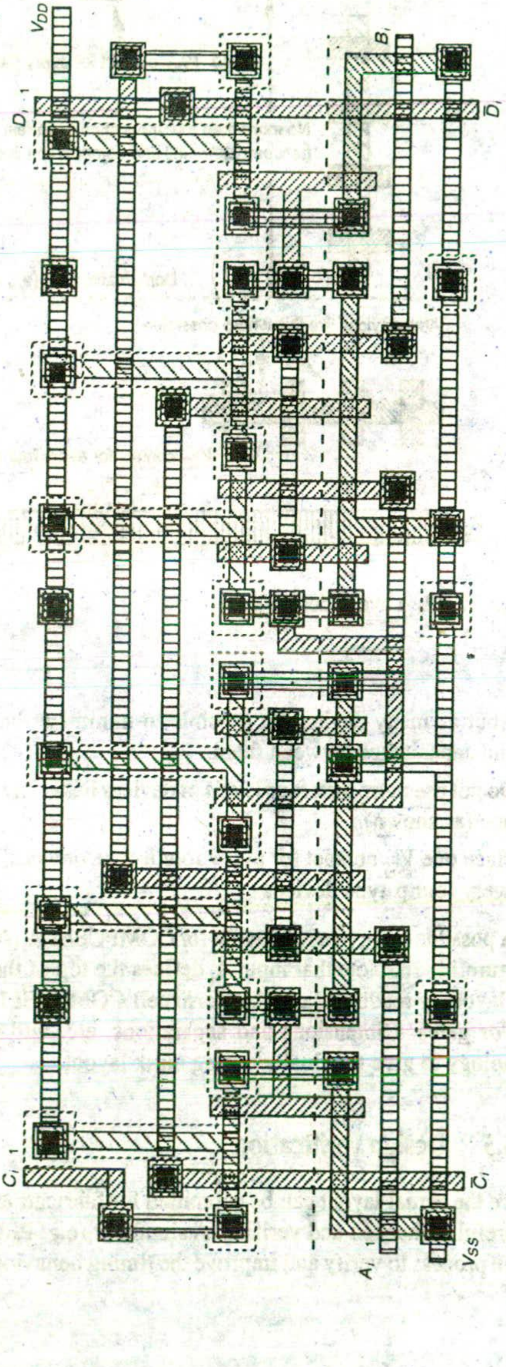
Figure 11–17 Layout design style

3. Abut as many devices as possible to minimize the interconnect resistance and capacitance between them.
4. Do not use more than four levels of devices between a rail and the demarcation line (as shown).
5. Place one V_{SS} contact for every four n-type devices, and one V_{DD} contact for every four p-type devices.

A possible embryo mask layout for COMPCELLA (*Nand* gate-based) is given in Figure 11–18. Note that input A_i defines the top of the cell and B_i the bottom. This layout is readily adapted to form cell COMPCELLB by exchanging *Nand* for *Nor* gates. Dimensions and separations, etc. will be fixed by the chosen technology to give the final working mask layout.

11.4.5 Design verification

Before the actual layout can be submitted for fabrication, the whole design must be carefully checked and verified. A simulator (e.g. PROBE) is used during the design process to verify and improve the timing behavior of each leaf-cell. When



Note: Cell orientation in relation to the floor plan (Figure 11–16).

Figure 11–18 COMPCELLA mask layout

the layout is completed, it must be passed through a design rule checker to verify its compliance with the design rules of the fabrication process to be used and an electrical rules checker to test for the number of V_{DD} and V_{SS} contacts etc. The necessary steps are:

- Pass the design through a design rule checker and correct any design rule errors.
- Check the circuit for any electrical rule errors using an electrical rules checker. Correct as necessary.
- Extract the circuit from the mask layout. This produces a file which is a circuit description containing connectivity information obtained from the CIF description.
- Simulate the cell and make any necessary changes to the transistor dimensions etc. to improve the performance. Remember to apply correct capacitance loadings to the output terminals when simulating. Simulation results obtained in this case for 5 μm technology are presented in Figures 11–19 and 11–20. Correct as necessary.
- Recheck for design rule errors.
- Carry out the final simulation at cell level.
- Assemble the complete 4-bit comparator using a suitable editor. Simulate and design rule check the complete subsystem.
- Place the input and output pads around the circuit as suggested in the floor plan in Figure 11–21. Note that the overall system, including the pads, should now have its operation checked by simulation.

11.5 CMOS/BiCMOS project 4* — a two-phase non-overlapping clock generator with buffered output on both phases

This project differs from the previous three since it is not concerned with the design of a cascadable architectural subsystem bit-slice. Rather, it is a control signal (two-phase clock) generator and bus driver which must perform well when driving relatively large capacitive loads.

* The work reported here was carried out by Brenton Cooper as part of a final year project at the University of Adelaide.

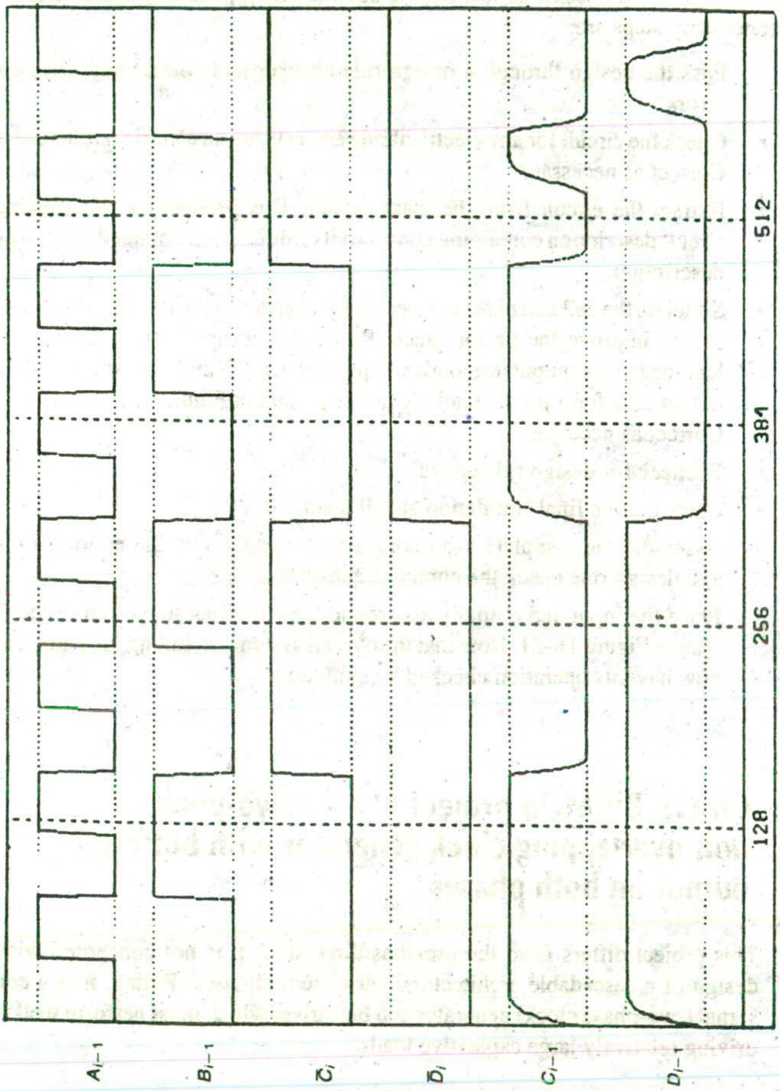


Figure 11–19 PROBE simulation results — COMPCELLA

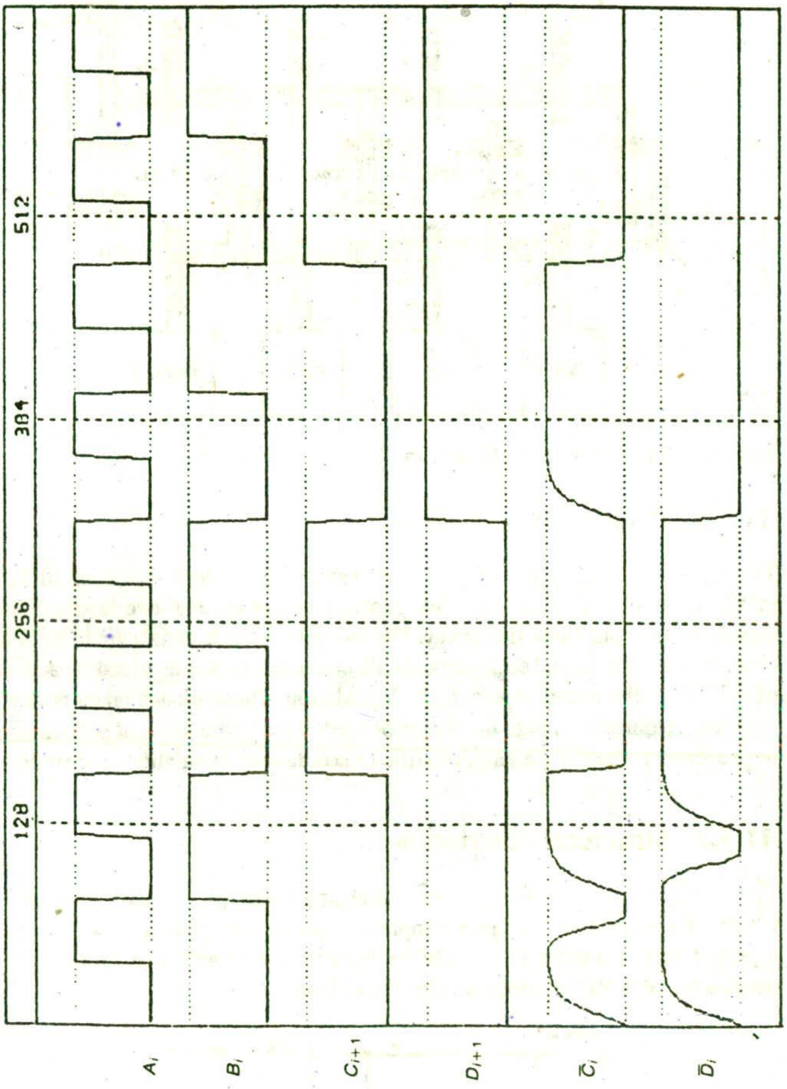


Figure 11–20 PROBE simulation results — COMPCELLB

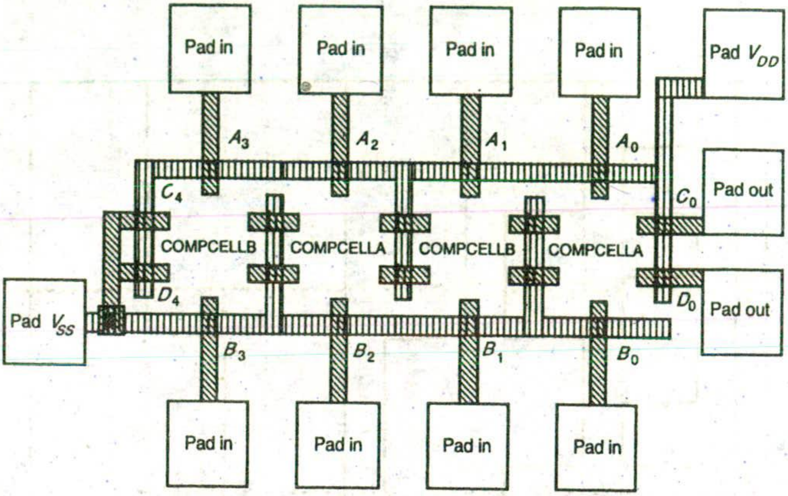


Figure 11–21 4-bit comparator with illustrative placements

11.5.1 Behavioral description

The circuit is required to accept a single-phase input clock signal of 10 MHz maximum frequency and, from this, generate two-phase non-overlapping clock signals at the input clock frequency. The two-phase clock signals are to be good, clean 'square' waves, and each phase should be capable of driving a load capacitance of 0.33 pF without undue waveform degradation. The approach taken is one of circuit development through the design of mask layout, simulation of performance, improvement where necessary, modified mask layout, simulation, and so on.

11.5.2 Structural description

The structure of a suitable basic circuit arrangement, previously introduced in Chapter 6 (Figure 6–33), is quite simple, comprising two gates and two inverters repeated here in Figure 11–22. Output buffers will be added to the two-phase outputs to cater for the specified capacitive load.

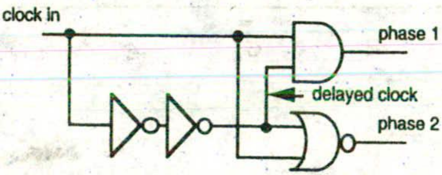


Figure 11–22 Basic two-phase clock generator logic

11.5.3 Design process

11.5.3.1 Version 1

In order to achieve the waveform requirements, it was decided to first complete a mask layout for the basic arrangement without output buffers and simulate this before proceeding further. The circuit realized is shown in Figure 11–23 and is a straightforward translation of the logic of Figure 11–22.

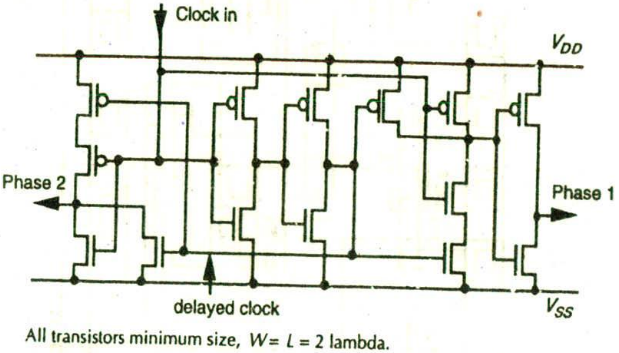


Figure 11–23 Basic two-phase clock generator circuit

The design rules used will be lambda-based with a value of $\lambda = 2\ \mu m$ for fabrication in single poly., single metal, p-well CMOS technology. All transistors are of minimum size, that is, $W = L = 2\lambda = 4\ \mu m$. The initial mask layout is given in Figure 11–24 (B&W copy of color pen plotter output) and corresponding 'H-Spice' simulation results at 10 MHz on zero external load in Figure 11–25.

The following observations are relevant:

1. The amount of 'underlap' between the phase 1 and phase 2 waveforms is barely adequate.
2. Phase 1 output rises faster than phase 2, and phase 2 peak voltage does not quite reach + 5 V owing to the time required for the *Nor* gate output to rise.
3. The square waves produced at each output are not particularly good and there is a noticeable 'glitch' on the phase 2 output.

11.5.3.2 Version 2

Clearly, all the above performance features could be improved by:

1. increasing the delay presently introduced by the two inverters in series;
2. reducing the output resistance of the final delay generating inverter so that the gate capacitance of the *Nor* gate will be charged faster, and also reducing the output resistance by widening the channels of the two p-type pull-up transistors of the *Nor* gate; and

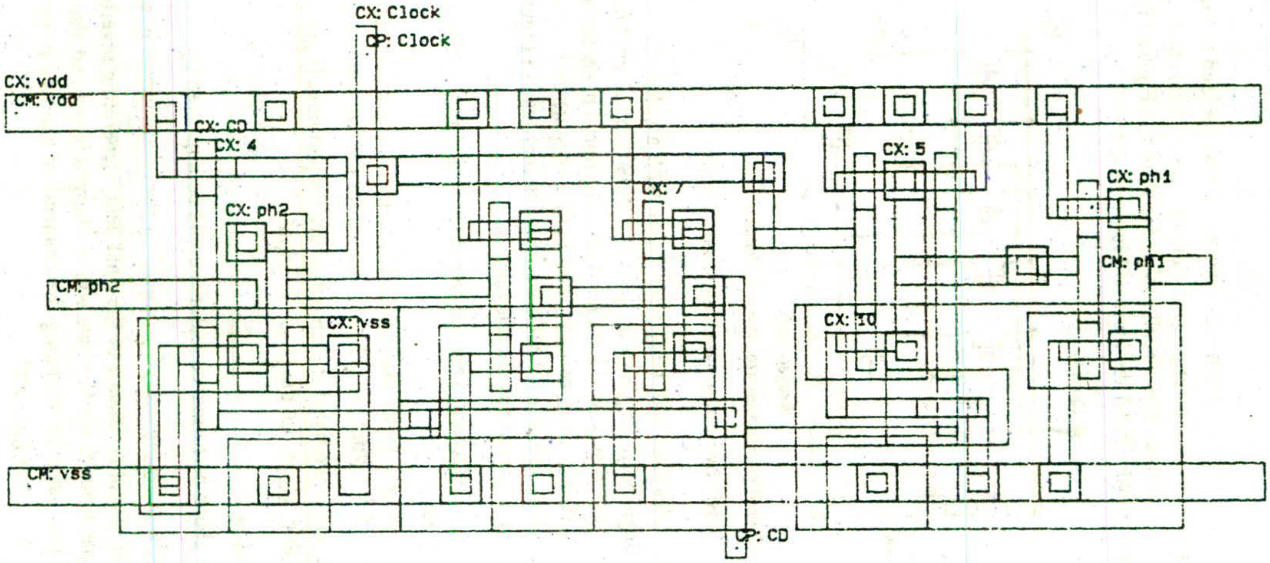


Figure 11–24 Mask layout (version 1) for two-phase clock generator circuit

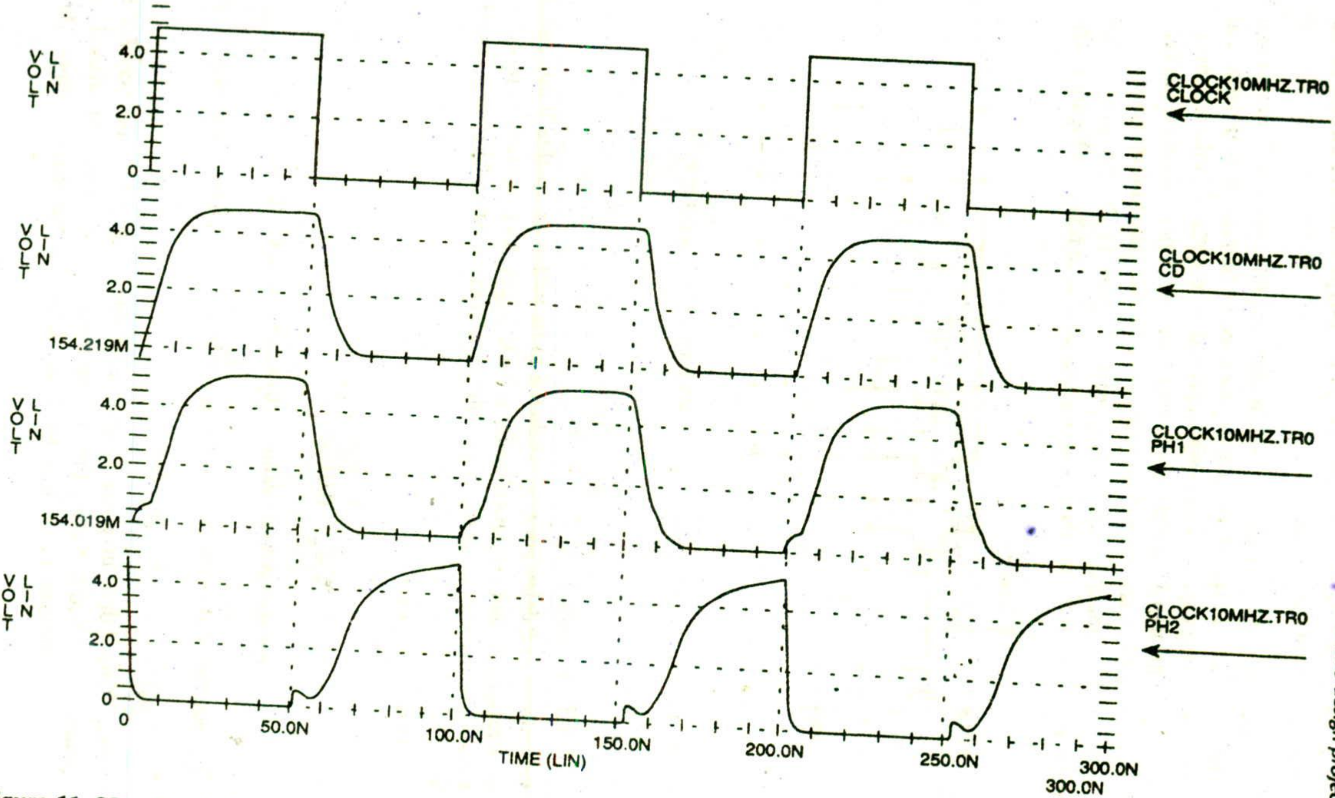
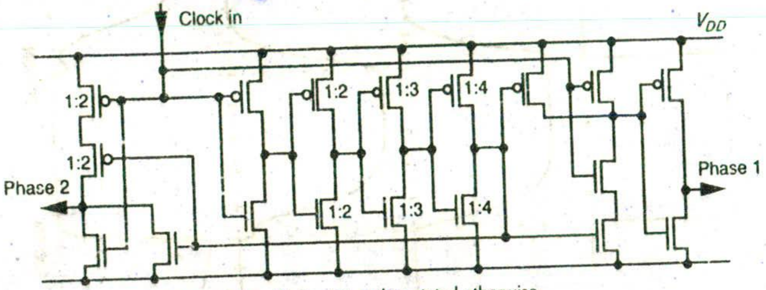


Figure 11–25 Simulation results for version 1

3. routing the delayed clock waveform to the p-type transistor closest to the output node of the *Nor* gate.

All the above points have been taken into account in version 2, noting that (1) the delayed clock waveform is now generated by four inverters in series and (2) that the driving capability has been improved by progressively decreasing the $L{:}W$ ratio for the transistor channels in each inverter. Improvement (3) has been taken account of by rearranged connections to the *Nor* gate pull-up gates.

The circuit implementation now appears as Figure 11–26, the revised mask layout as Figure 11–27 and the corresponding simulation results as Figure 11–28.



All transistors minimum size, $W{:}L$ = 1:1 unless stated otherwise.

Figure 11–26 Circuit (version 2) for two-phase clock generator circuit

11.5.3.3 Version 3

Waveforms and delay are now predicted to be within acceptable limits and it now remains to add the output buffer at each output. An acceptable approach is to cascade inverters of increasing channel width as set out in Chapter 4, section 4.8.

In this case, the ratio

$$y = \frac{C_L}{\square C_g} = \frac{0.33\ \text{pF}}{0.01\ \text{pF}} = 33$$

(An approximate value of .01 has been assumed for $\square C_g$.)

The number N of cascaded stages is given by $N = ln(y)$; thus in this case.

$$N = 3.5\ (\text{say, } N = 3).$$

Thus, we need three inverters in series, each one being ÷ 2.7 times (say 2.5) its predecessor's width. Noting the existing output inverter stage for phase 1, we need two additional buffer inverter stages to provide the phase 1 output. Three

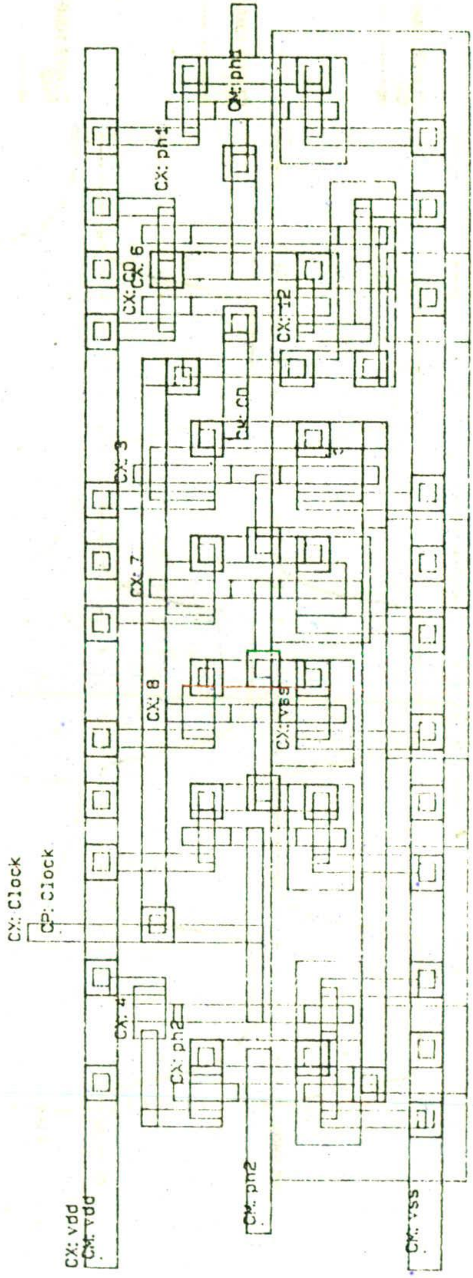


Figure 11–27 Mask layout (version 2) for two-phase clock generator circuit

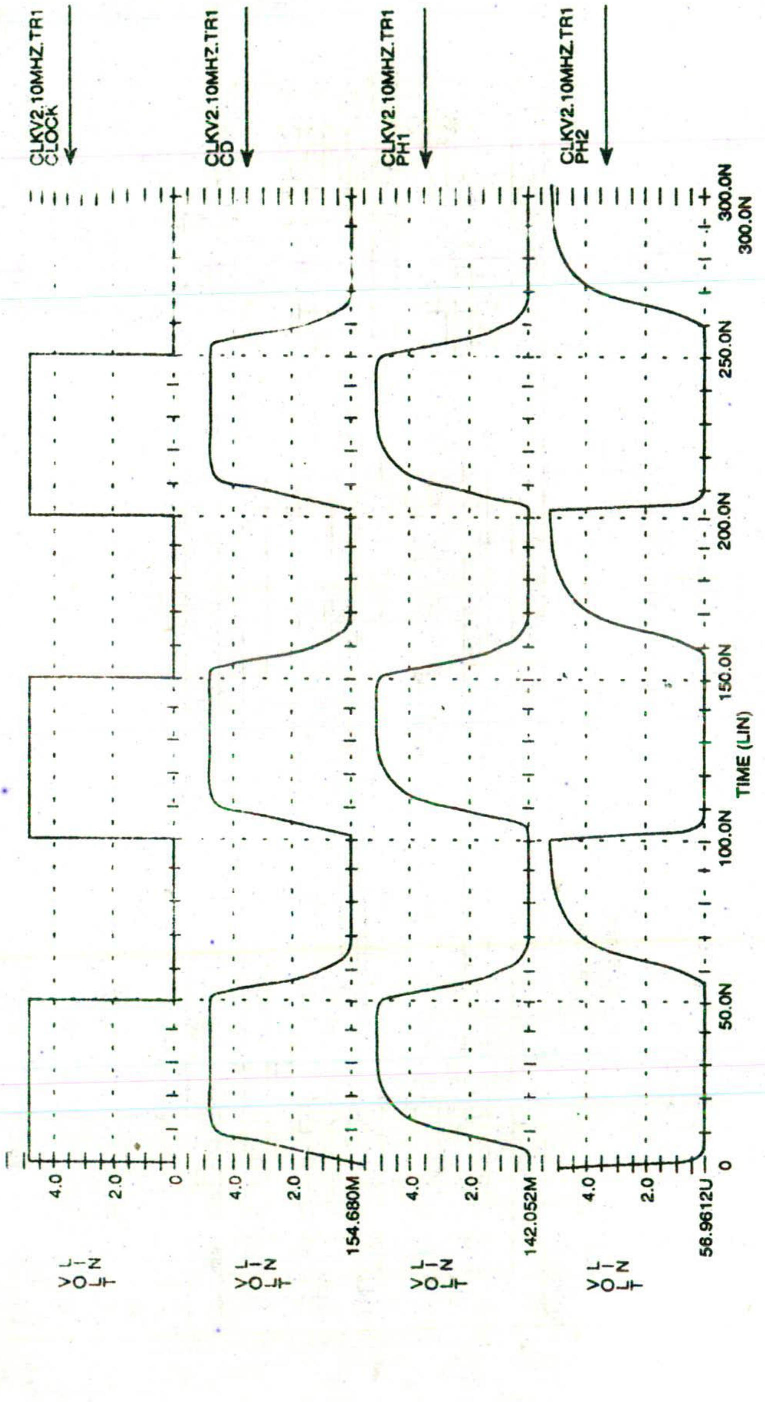
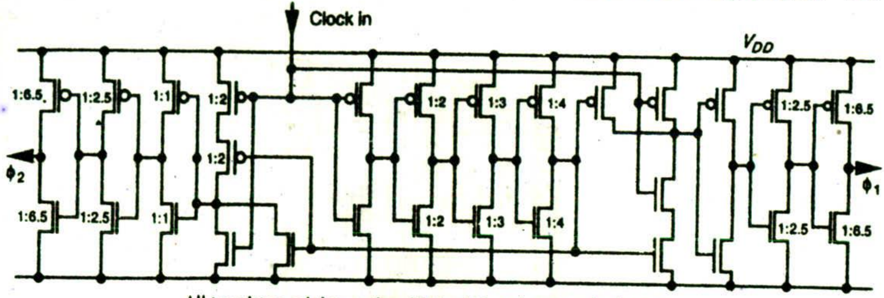


Figure 11–28 Simulation results for two-phase clock generator (version 2)



All transistors minimum size, $W:L = 1:1$ unless stated otherwise.

Figure 11–29 Circuit (version 3) for two-phase clock generator circuit

will be needed for phase 2 but a fourth is added to maintain the original phase relationship. The circuit is shown in Figure 11–29 and the modified mask layout is shown in Figure 11–30.

11.5.4 Final test (simulation) results

The final version (version 3) of the mask layout was first simulated on no external load and the waveforms generated at 10 MHz clock frequency. Near ideal non-overlapping square waves were observed as shown in Figure 11–31.

In order to assess the effect of increasing this maximum operating frequency, the input clock rate was doubled and results for a 20 MHz input clock were observed as in Figure 11–32. It will be seen that the performance of the circuit is still very good.

Finally, the outputs were loaded with load capacitance C_L which was increased in value until the slope of the clock edges began to erode the underlap between the phases. Figure 11–33 shows that acceptable waveforms are still generated even if the originally specified C_L value is exceeded by a factor of six times, indicating a very conservative design.

11.5.5 Further thoughts

The mask layouts presented here are those from which the simulation results were obtained. They should, however, be used with care since the yellow lines defining p+ areas do not copy in monochrome (B&W) form and are thus not apparent in the layouts reproduced here. Also, actual details of the appropriate technology design rules should be applied to a layout as necessary.

If larger capacitive loads are required to be driven, such as an output pad with associated off chip wiring etc., then two possibilities are:

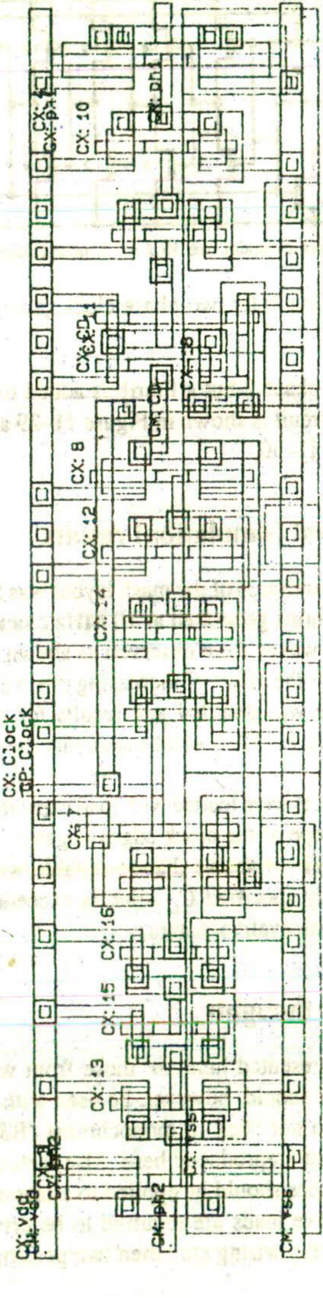


Figure 11–30 Mask layout (version 3) for two-phase clock generator circuit

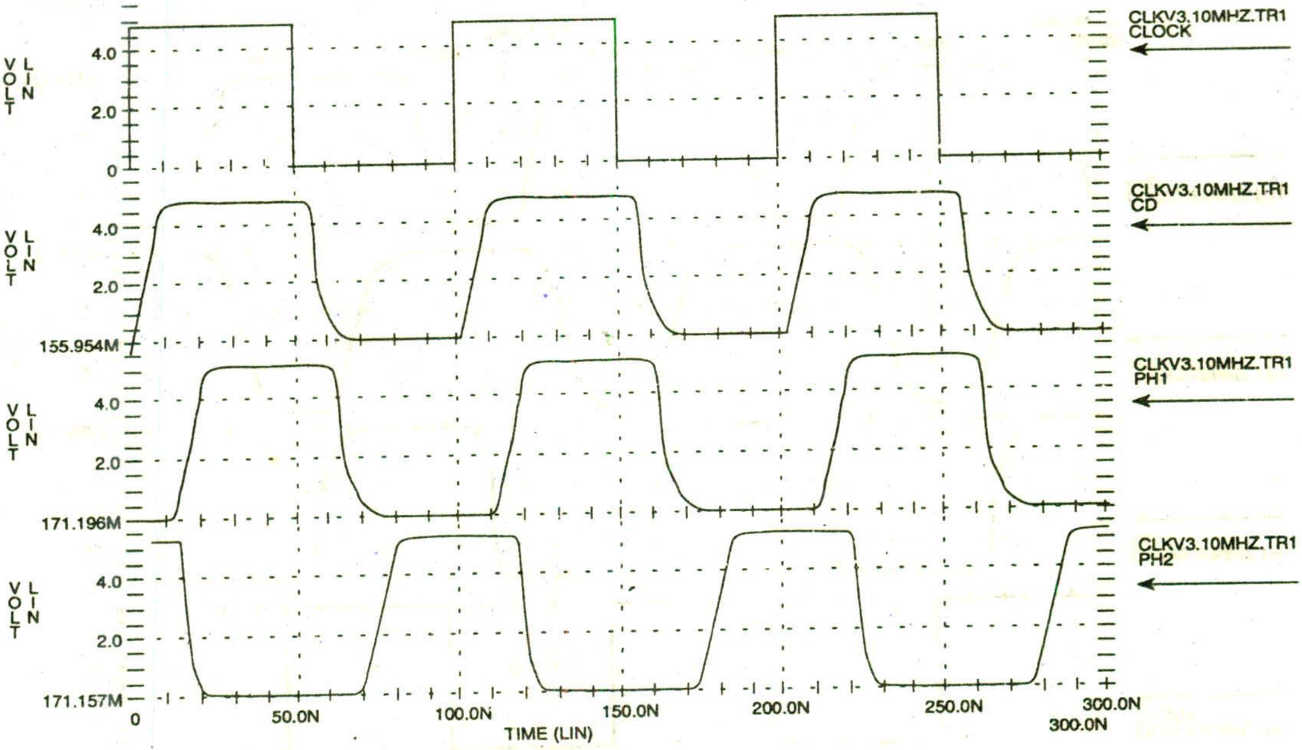


Figure 11–31 10 MHz Simulation results for two-phase clock generator (version 3)

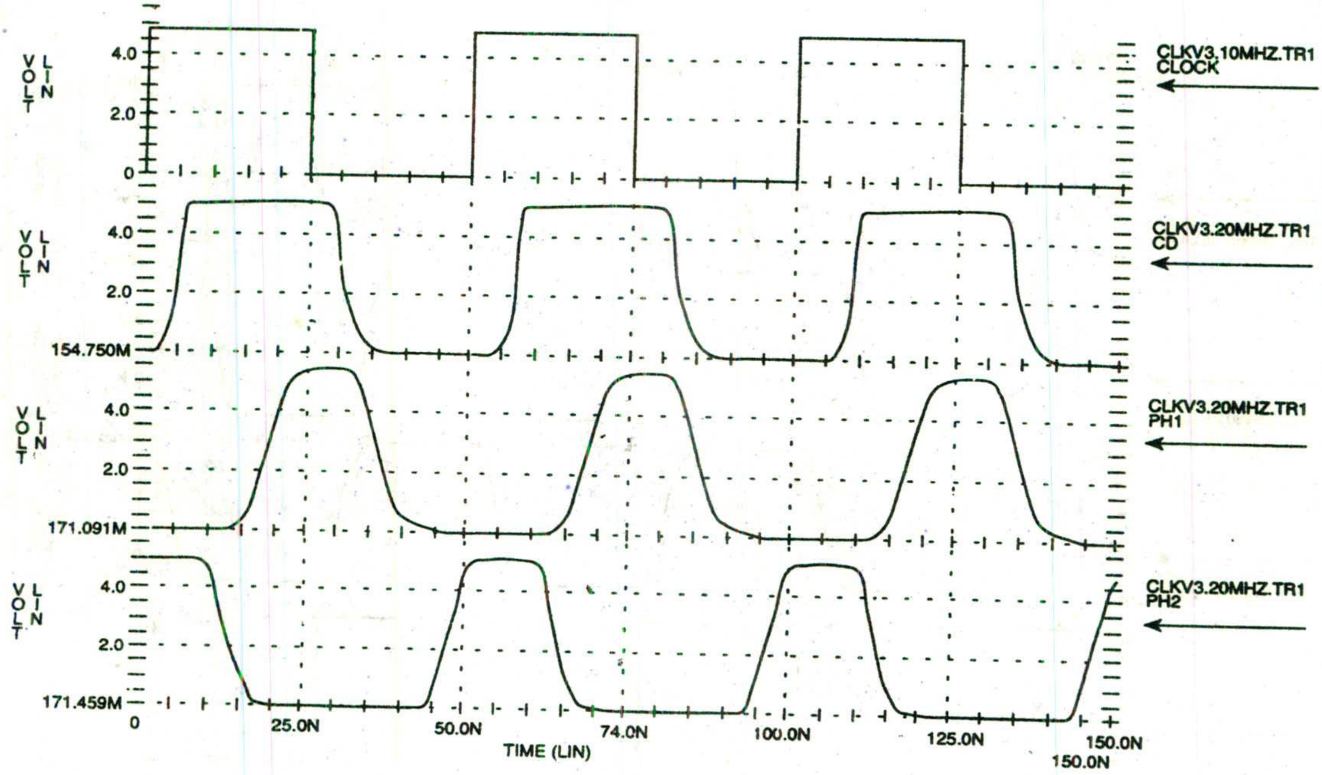


Figure 11–32 20 MHz simulation results for two-phase clock generator (version 3)

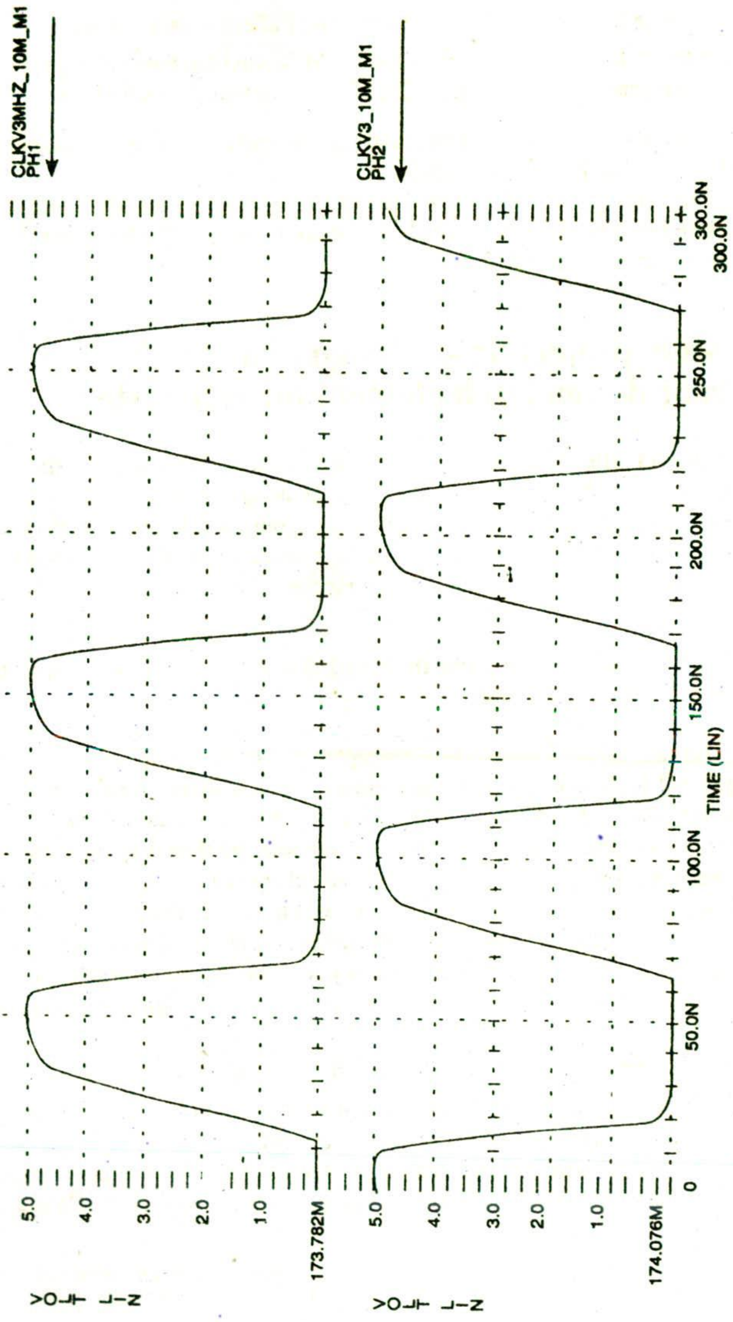


Figure 11–33 10 MHz simulation results for two-phase clock generator (version 3) with C_L = 2 pF

1. Increase the number of cascaded inverter buffers at each output.
2. If the technology in use caters for BiCMOS circuits, then redesign the two-phase generator to include BiCMOS output stages for each phase.

A further need may be for buffered complementary outputs, that is $(\overline{\text{phase 1}})$ and $(\overline{\text{phase 2}})$ to be also generated.

The mask layout for an arrangement which employs BiCMOS technology and provides four outputs —(phase 1), $(\overline{\text{phase 1}})$, (phase 2), and $(\overline{\text{phase 2}})$ — is presented as Color plate 12.

11.6 CMOS project 5* — design of a ∂*latch* — an event-driven latch element for EDL systems

This project differs from the previous four since it is concerned with the design of an *event-driven* circuit element. In fact the design to be pursued is that of an event-driven latch (∂*latch*) which is part of ongoing developments in the field of event-driven-logic (EDL). Before a behavioral description can be set out, it is necessary to acquaint the reader with some basic aspects of EDL.

11.6.1 A brief overview of event-driven logic (EDL) concepts (Pucknell, 1993)

An alternative way of approaching the representation and design of asynchronous sequential logic is to take an 'event-driven' or 'transition-based' approach. In concept, the approach taken is to define the initial conditions of a system in terms of the logic level assumed by each variable and then describe subsequent system behavior in terms of the transitions (changes in logic level, also called events) of those variables. Clearly, if all events are defined for each variable, then subsequent logic level states are also defined. In order to pursue this approach, let us first examine some of the basic features and factors associated with the concept of 'event-driven' or 'transition-based' logic and logical operations.

11.6.1.1 An event-driven or transition-based approach to logic

In formulating event-driven logic (EDL), it is necessary to adopt special operators which readily express the *transitions or events* which may occur.

Transition operators and some basic relationships. The operators proposed are an extended set of the two originally proposed (Talantsev, 1959). Considering

*The design work on this latch was carried out by postgraduate reseacher Shannon Morton at the University of Adelaide as part of the digital systems group work on the application of EDL concepts to the design of asynchronous processors.

a single line carrying a logic signal denoted 'A', then at any time 't' there are four possibilities:

1. Δ A denoting a change in A from 0 to 1;
2. ∇A denoting a change in A from 1 to 0;
3. $\overline{\Delta}$ denoting no change in A at logic 0;
4. $\overline{\nabla}$ A denoting no change in A at logic 1.

Note the operators $\Delta \nabla \overline{\Delta}\ \overline{\nabla}$ and their significance.

Possibilities 1 and 2 may be defined as *'events'* and we may write:

$$\partial.A = \Delta A + \nabla A$$

where ∂.A indicates *any event* for signal A.

Possibilities 3 and 4 may be defined as *'non-events'* and we may write:

$$\overline{\partial}.A = \overline{\Delta} A + \overline{\nabla} A$$

the negated ∂ indicating *no event* for signal A.

11.6.1.2 Some bridging rules between EDL and 'conventional logic'

Clearly, there must be some relatively straightforward rules for converting between conventional and event-driven forms of logic and EDL elements may be constructed from conventional combinational logic circuits, as is also the case for clocked sequential elements.

The basic relationships are simple and may be proved quite readily mathematically or through a process of logical reasoning. Requirements are met by the rules given in Table 11–2. To illustrate the use of these rules, we may predict the transition behavior of a simple conventional *two-input And* gate. To do this, we start with the conventional logic equations and then apply the rules of Table 11–2.

Table 11–2 Simple bridging rules

Event-driven		*Conventional*	*In words*
$\nabla A + \overline{\Delta} A$	<=>	$\overline{A}$	A becomes 0 or remains at 0
$\Delta A + \overline{\nabla} A$	<=>	A	A becomes 1 or remains at 1
$(\Delta A + \nabla A + \overline{\Delta} A + \overline{\nabla} A$	<=>	1	All possible events for A
$\Delta A (\nabla A + \overline{\nabla} A + \overline{\Delta} A)$	<=>	0	'Anding' differing events for A

Where '<=>' indicates 'translates to' and should be considered in the context of what is actually meant in conventional logic when we write, say, A = B.C or $\overline{A} = \overline{B} + \overline{C}$ etc.

$X = A.B$ becomes

$$\Delta X + \overline{\overline{\nabla}}X = (\Delta A + \overline{\overline{\nabla}}A).(\Delta B + \overline{\overline{\nabla}}B)$$
$$= \Delta A.\Delta B + \Delta A.\overline{\overline{\nabla}}B + \overline{\overline{\nabla}}A.\Delta B + \overline{\overline{\nabla}}A.\overline{\overline{\nabla}}B.$$

A little thought will reveal that this equation comprises two parts:

1. the conditions for X to change from 0 to 1

$$\Delta X = \Delta A.\Delta B + \Delta A.\overline{\overline{\nabla}}B + \overline{\overline{\nabla}}A.\Delta B$$

2. the conditions for X to remain at logic 1

$$\overline{\overline{\nabla}}X = \overline{\overline{\nabla}}A.\overline{\overline{\nabla}}B$$

Similarly, starting with the complementary form of the expression

$$\overline{X} = \overline{A} + \overline{B}$$

we may arrive at expressions for

3. $$\nabla X = \nabla A + \nabla B$$

and

4. $$\overline{\Delta}X = \overline{\Delta}A + \overline{\Delta}B$$

Taking events only:

$$.\Delta X = \Delta A.\Delta B + \Delta A\,\overline{\overline{\nabla}}B + \overline{\overline{\nabla}}A.\Delta B$$

and $$\nabla X = \nabla A + \nabla B.$$

These are the EDL equations defining the conditions for X to change from 0 to 1 and from 1 to 0 respectively. EDL equations can be written for any gate. For example, a two input *Nor* gate (inputs *A* and *B*, output *Y*) can be represented by:

$$\nabla Y = \Delta A + \Delta B \text{ and } \Delta Y = \nabla A\,\nabla B + \nabla A\,\overline{\Delta}B + \overline{\Delta}A\,\nabla B$$

Clearly, then, the behavior of simple combinational logic gates may be expressed in terms of events. Note, however, that the common combinational logic gates may well not generate simple EDL functions but it is possible to conceive a specifically designed set of EDL gates which perform straightforward EDL functions but which, in turn, may not generate simple combinational logic functions. The exception is the *Exclusive Or* gate which is the point of intersection between the two gate sets.

11.6.1.3 The inverter as an EDL element

The inverter converts one transition of its input variable (e.g. 0 to 1) into the other transition (1 to 0 for the example) at the output. It also quite clearly converts a logic level at the input to its complement at the output. The inverter will *not convert events into non-events or vice versa* unless it is faulty.

11.6.1.4 Other EDL elements

So far, this discussion has covered the EDL aspects of gate logic circuits and we may now turn attention to the application of EDL concepts to the design of storage elements. EDL storage elements will be driven and activated by events on specified control inputs. For example, the event-driven latch to be discussed here is activated by events on *pass* and *capture* control inputs.

11.6.2 Behavioral description of a *∂latch*

The circuit is required to accept a single input, pass this to a single output when any event occurs on a *pass* (*p*) control line and latch this output when any event occurs on a *capture* (*c*) control line. The basic, most general, arrangement is shown in symbolic form in Figure 11–34 and it may be seen that a delayed version of each event control line, namely *pass done* (*pd*) and *capture done* (*cd*), is presented as an output control signal. A *Clear* (*clr*) input is also required. In a particular configuration, the *pd* output provides the *c* input and the delay through the two inverters is sufficient to allow the select line of the latch input switch to go high long enough for data to propagate through the latch from input to output before it is captured. Thus, the whole latch action is controlled by events on the *p* input line. It is that version which is to be implemented here.

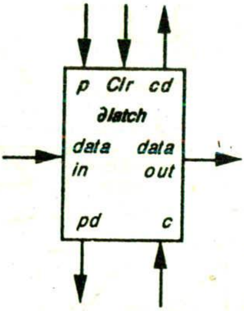
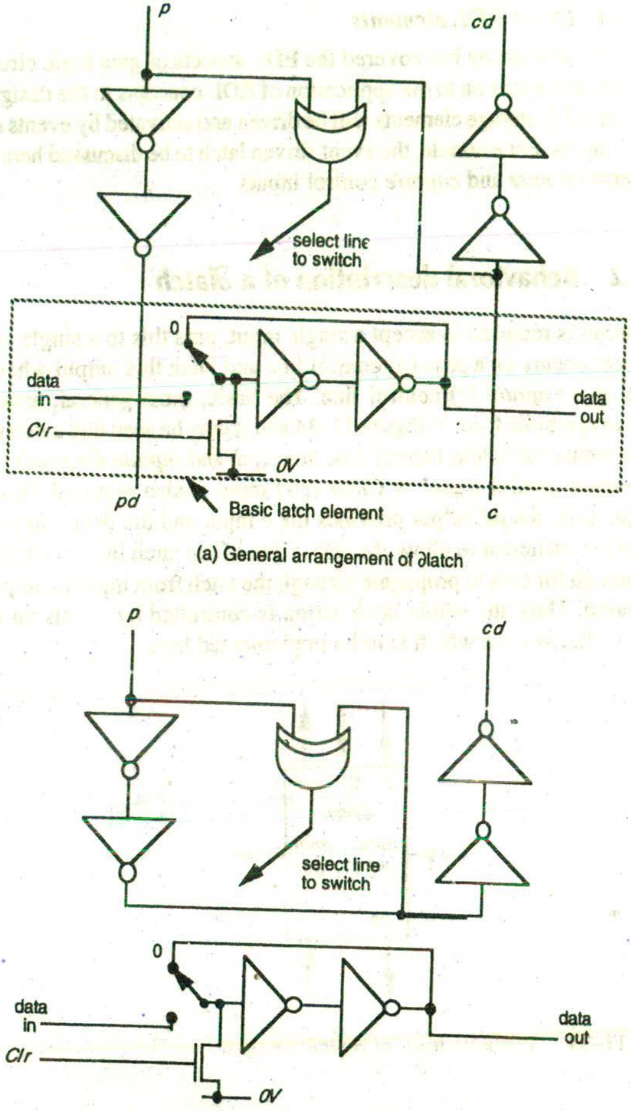


Figure 11–34 Symbolic form of *∂latch* element

11.6.3 Structural description

The structure of a suitable basic latch circuit arrangement is quite simple, comprising three inverter pairs, an *Xor* gate and a switch (multiplexer) as shown in Figure 11–35(a). Note that, in this case, *pd* and *c* will be joined, as in Figure 11–35(b), internally in the mask layout.



(a) General arrangement of ∂latch

(b) Arrangement of ∂latch to be realized here

Figure 11–35 Basic arrangement of the *∂latch* element

11.6.4 Circuit action

The select line is generated by an *Xor* gate between inputs *p* and *c*. The *done* events will occur after the select line has reached its new state which will activate the actual latching/storage part of the circuit. This consists of two pass transistor switches with a supporting pull-up transistor, a two inverter buffer/driver, and a clear transistor. If one wishes to latch more than 1-bit of data, then it is this part of the circuit alone which must be replicated, for example 16 times for latching a 16-bit word.

When the select line goes high, the input logic level is connected to the buffer/driver through one of the pass transistor switches and the output of the buffer/driver will assume the same logic level. If this is a logic 1, the logic level would be degraded by the threshold voltage of the pass transistor, but the output (logic 0) of the first inverter of the buffer/driver is used to turn on a p-type pull-up transistor which acts as a pull-up to the output of the pass transistor, thus restoring a good logic 1 level. When the latch enters the *capture* state, the select line goes low, the input pass transistor switch is turned off and the other pass transistor switch is turned on, thus connecting the input of the buffer/driver pair to its output. Thus, the data is latched.

The *clear* line is inactive when low but when enabled with a logic 1, the pass transistor switch output node is forced low and will remain low even if the select line goes high and the logic level at the input is a 1.

11.6.5 Mask layout and performance simulation

The translation of the latch circuit into a mask layout is conveniently achieved using either a symbolic entry editor or a direct mask entry editor. In either case, the technology chosen will determine absolute widths, separations and overlaps and will also determine C and R values for the various layers.

In this case, the geometry of a suitable mask layout is given in Figure 11–36 and network extraction and simulations have been carried out in both 5 μm and 1.2 μm double metal, single poly., p-well CMOS technologies. Simulation results are given in Figure 11–37 (5 μm) and Figure 11–38 (1.2 μm). Noting the differing time scales used to plot the simulation results, it can be seen that the 1.2 μm latch is faster than the 5 μm latch by a factor of approximately 5. This compares favorably with the theoretical speed-up factor = $^5/_{1.2} \approx 4.2$. Propagation times through the latch are approximately 5.4 nsec for the 5 μm design and 1.1 nsec for the 1.2 μm design.

-27

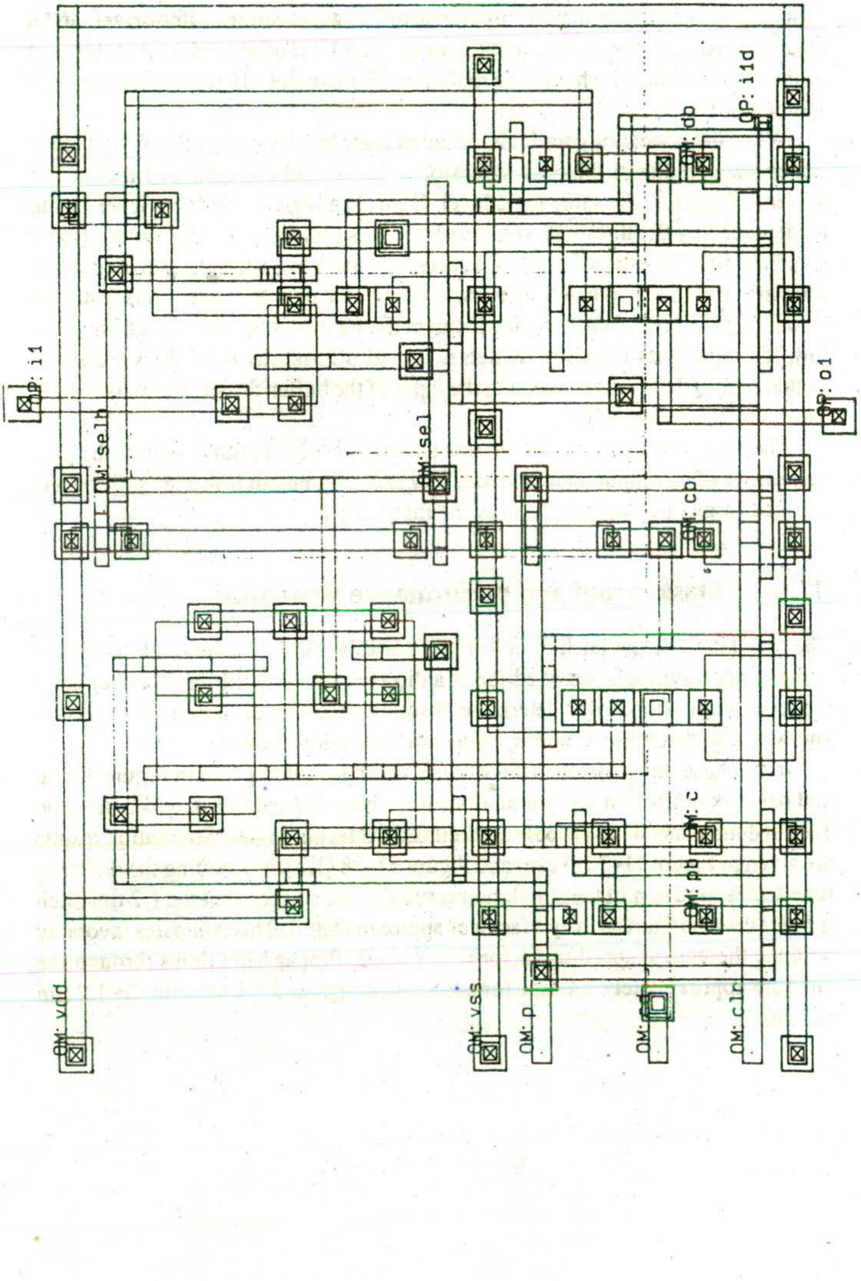


Figure 11–36 A mask layout for the ∂latch

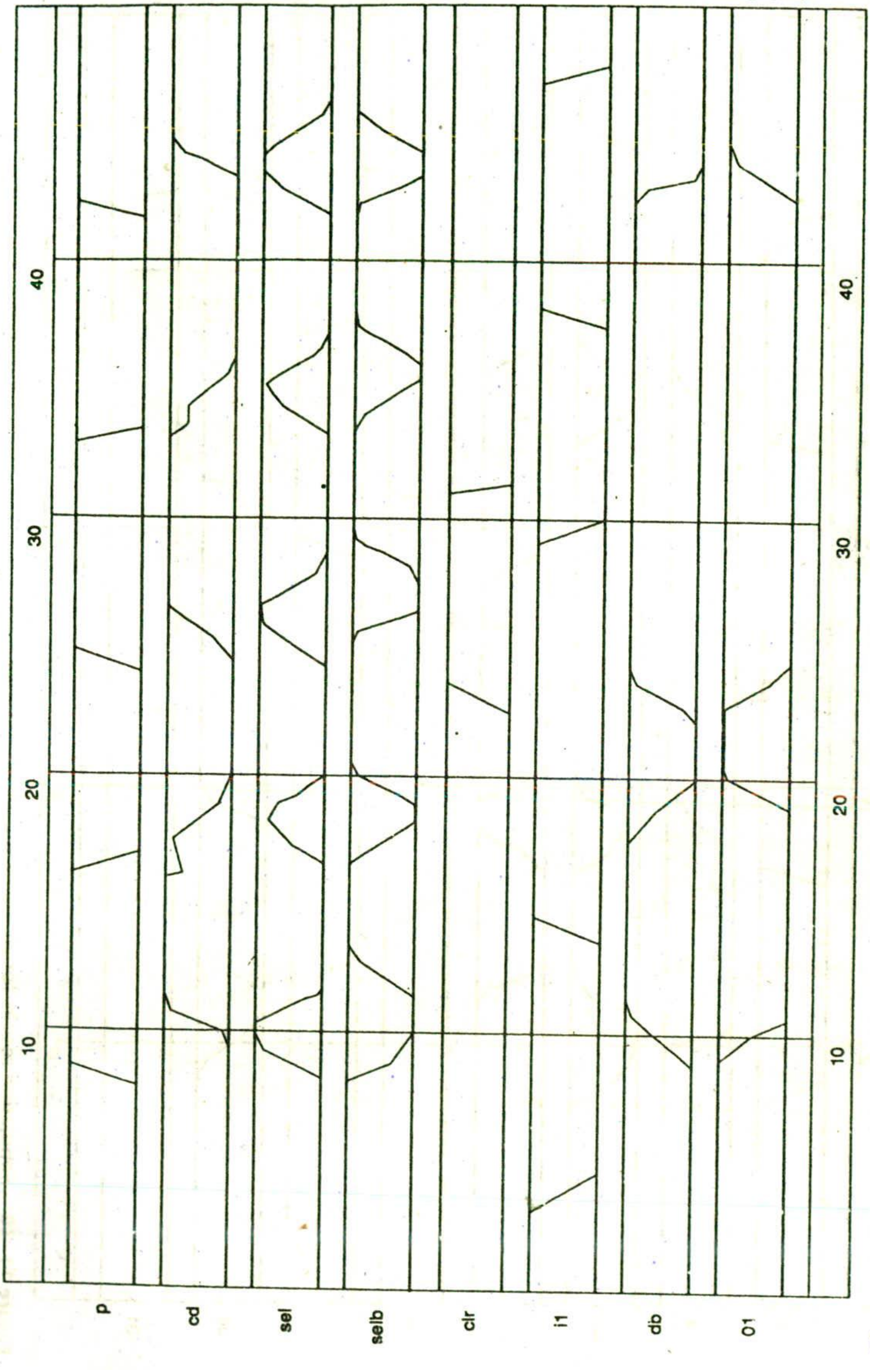


Figure 11–37 Simulation results for ∂latch in 5 µm technology

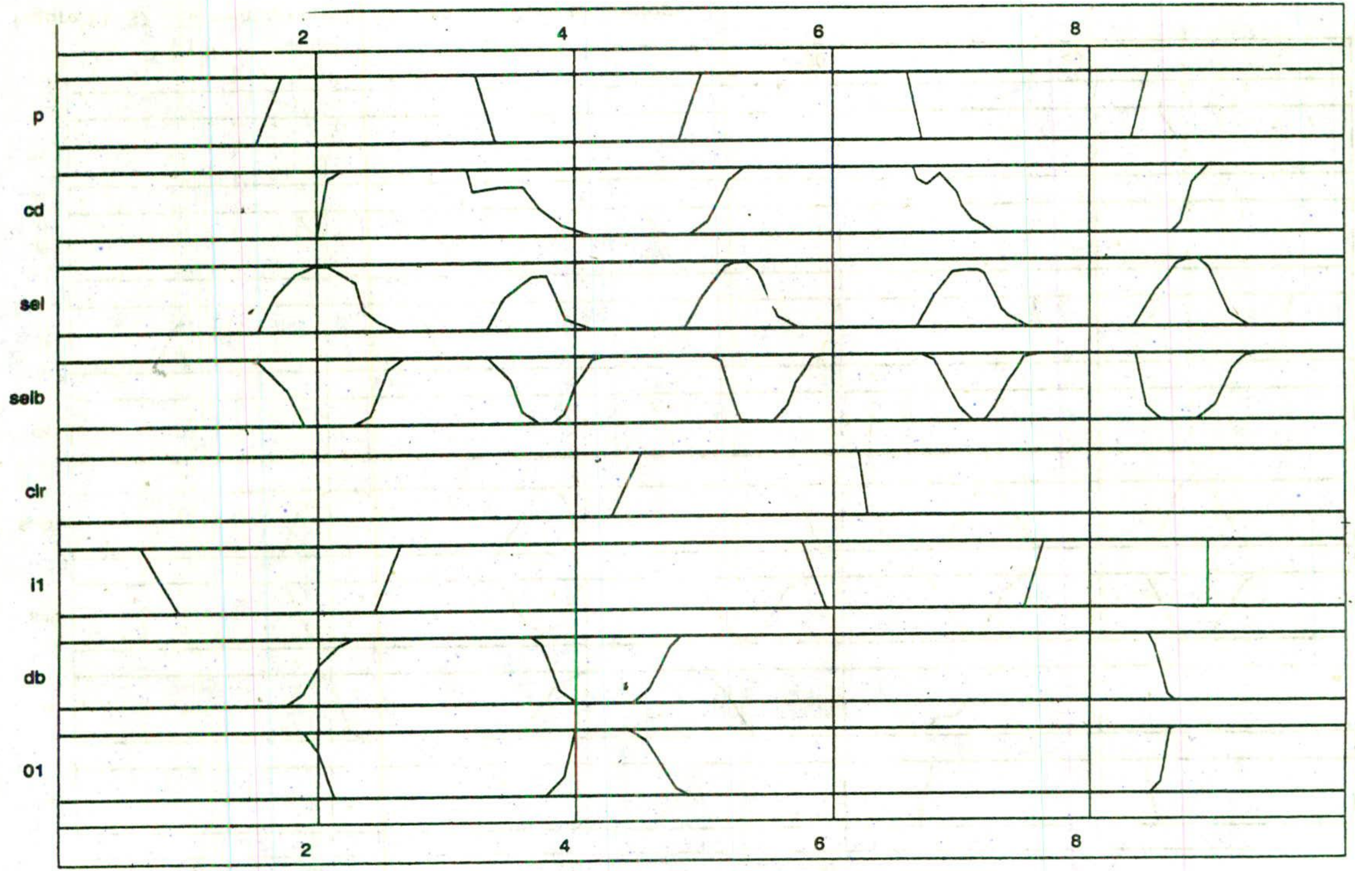


Figure 11–38 Simulation results for ∂latch in 1.2 μm technology

11.7 Observations

We have seen that the design process for the design of digital systems in silicon is a reasonably straightforward proposition, provided that an orderly, structured approach is taken. The tutorials, exercises, and project work in the text have illustrated approaches to design, and readers should by now begin to feel comfortable in their ability to tackle the design of systems of modest size and complexity. An ability to understand the characteristics of the available technologies and the design processes should enable system designers to specify an appropriate technology and, where necessary, design 'custom' digital chips.

This text has not attempted to seriously address the problems of complexity management and the design time associated with the design of large digital systems. We have also largely ignored the ever-growing need for custom-designed analog circuits in MOS technologies, both for pure analog applications and for 'on-chip' interfaces between the analog world and digital systems.

We have seen that there are factors which limit the ultimate scaling of silicon circuits and thus there are ultimate limitations on the speed of silicon circuitry. This will not be a problem in any but the fastest areas of application, but emerging needs in real-time control and in signal processing applications, to name just two, may well impose needs beyond the capability of MOS silicon systems alone. It is in such applications that other technologies, in particular gallium arsenide, will find application as fast 'front-end' processors to silicon systems. To introduce the reader to this important area, the next, and final, chapter introduces gallium arsenide technology.

11.8 References

Pucknell, D. A. (1970) 'Transition equations for the analysis and synthesis of sequential circuits', *IEE. Electron. Lett.*, 6 (23), 731–33.

Pucknell, D. A. (1993) 'An event-driven-logic (EDL) approach to digital system representation and related design processes', *IEE. Proc.-E, Computers and Digital techniques*, Vol. 140, No. 2, 119–26.

Pucknell, D. A. (1973, May) 'Sequential circuit characterisation and synthesis using a transition equation approach', *Proc. IEE.*, 120(5), 551–56.

Pucknell, D. A., and Liebelt, M. J. (1990, July) 'Aspects of event-driven logic', Proc. 9th Australian Microelectronics Conference, Adelaide, South Australia, 171–73.

Smith, J. R., and Roth, C. H. (1971) 'Analysis and synthesis of asynchronous sequential networks using edge-sensitive flip-flops', *IEEE. Trans. Comput.*, C-20, 847–55.

Talantsev, A. D. (1959) 'On the analysis and synthesis of certain electrical circuits by means of special logical operators', *Autom. and Telemech.*, 20, 895–907.

12 Ultra-fast VLSI circuits and systems — introduction to GaAs technology

There was a young lady named Bright,
Whose speed was far faster than light,
She set out one day
in a relative way,
And returned home the previous night.

Arthur Henry Buller

12.1 Ultra-fast systems

In this final chapter we will briefly review some of the limitations of silicon devices and then look at the emerging alternative for ultra-fast systems — gallium arsenide.

12.1.1 Submicron CMOS technology

Speed and smaller device dimensions are closely interrelated, and we have already touched on the fact that the foreseeable limits on channel length for MOS transistors is in the region of 0.14 μm, after which further scaling down results in unworkable transistor geometry.

In CMOS devices we have also seen that the p-transistors have inherently slower performance than similar n-transistors. This is primarily due to the lower mobility of holes compared with that of electrons. Typically

$$\mu_p \doteqdot 240\ \text{cm}^2/\text{V.sec}$$
$$\mu_n \doteqdot 650\ \text{cm}^2/\text{V.sec}$$

In long-channel devices this means a difference in current drive transition times of about 2.5:1. However, as the channel lengths are scaled down, the influence of mobility starts to diminish as the effects of velocity saturation begin to be felt.

For long-channel MOS transistors, the current/voltage relationship below saturation can be approximated by

$$I_{ds} = \frac{W\mu C_{OX}}{L}[(V_{gs} - V_t)V_{ds} - 0.5V_{ds}{}^2]$$

where

$$C_{OX} = \text{gate/channel capacitance per unit area}$$

$$= \frac{\varepsilon_{ins}\varepsilon_0}{D}.$$

This implies that current drive is proportional to mobility and inversely proportional to channel length.

Transconductance g_m is similarly influenced. When velocity saturation occurs along the entire channel length, then the current/voltage relationship is given by

$$I_{dsat} = WC_{OX}v_{sat}(V_{gs} - V_t)$$

where v_{sat} is the *saturation velocity*. Current is now independent of both mobility and channel length but dependent on the saturation velocity. Transconductance is constant and thus independent of channel length.

It should be noted that velocity saturation occurs at lower electric field strengths in n-devices owing to their higher mobility when compared with p-devices. Thus, as dimensions are scaled down, the current drive from n-transistors tends to a constant value independent of channel length while the current drive from p-transistors does not tend to a constant value until, at a shorter channel length, the holes start to run into velocity saturation. We must therefore look to other than silicon-based MOS technology to provide for the faster devices which will undoubtedly be required as the sophistication of our system design capabilities increases. An alternative technology is based on gallium arsenide.

12.1.2 Gallium arsenide VLSI technology

He that will not apply new remedies must expect new evils:
for time is the greatest innovator.

Francis Bacon

Silicon MOS technology has been the main medium for computer and system applications for a number of years and will continue to fill this role in the foreseeable future. However, silicon logic has speed limitations that are already becoming apparent in state-of-the-art fast digital system design. Paralleling developments in silicon technology, some very interesting results have emerged for gallium arsenide (GaAs)-based technology. Gallium arsenide will not displace silicon but is being used in conjunction with silicon to satisfy the need for very high speed integrated (VHSI) technology in many new and innovative systems.

Much of the development work in material technology that has paralleled that in silicon has been related to groups II–VI and groups III–V compounds, with gallium arsenide, a group III–V compound, showing the most promise.

The compound gallium arsenide was discovered in 1926. However, its potential as a high speed semiconductor was not realized until the 1960s. The high speed electron mobility of gallium arsenide with respect to silicon, a semi-insulating substrate with consequent lower parasitics, a 1.4 improvement factor for carrier saturation velocity of GaAs over silicon, its opto-electrical properties, as well as significant improvement in power dissipation and radiation hardness, have promised an ultimate system performance advantage for gallium arsenide products, given similar lithographical processes.

The developments in integrated circuit fabrication technology in the 1970s made such gallium arsenide products a possibility and finally, as the result of significant advances in ion implantation in the 1980s, GaAs VLSI technology is a commercial reality for the 1990s.

Therefore, in the sections to follow we are going to concentrate on this new material and explore the various possibilities that exist to design circuits and systems using an appropriate class of logic together with suitable design methodology for this technology.

12.2 Gallium arsenide crystal structure

Gallium (Ga), a toxic material, is produced as a byproduct in both the zinc and aluminum production processes. Similarly, arsenic (As), which is also very toxic, is produced from ores such as As_2S_3 or As_2S_4. The process entails firstly oxidation of the ores to form As_2O_3 and subsequently, through reduction with carbon, arsenic is produced.

In order to better appreciate the structure and the properties of gallium arsenide crystal, it is appropriate to focus some attention on the characteristics of the individual atoms themselves. Figure 12–1 shows Bohr's model of the atomic structures for gallium and arsenic. Similar representation for silicon is also illustrated for comparison.

Gallium possesses a positively charged nucleus of +31, while the arsenic atom's nucleus has a positive charge of +33. In each case, the total positive charge of the nucleus is equalized by the total effective negative charge of the electrons.

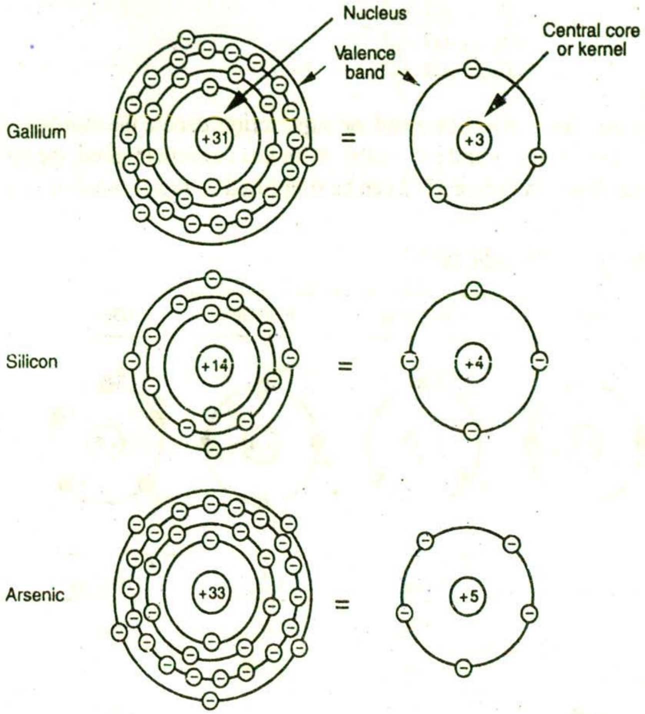


Figure 12–1 Bohr's model for silicon, gallium and arsenic

Electrons, traveling within their respective orbits, possess energy since they are a definite mass in motion (i.e. rest mass of electron is $9.108*10^{-23}$gm). This means each electron in its relationship with its parent nucleus exhibits an energy value and functions at a distinct energy level. This energy level is dictated by the electron's momentum and its physical proximity to the nucleus. The closer the electron is to the nucleus, the greater is the holding influence of the nucleus on the electron and the greater is the energy required for the electron to break loose and become free.

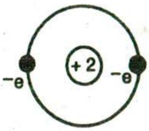
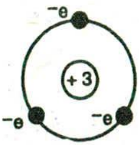
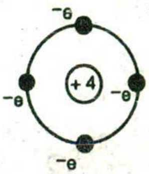
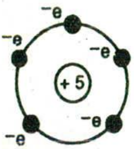
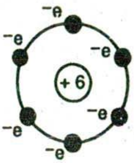
Outer orbit electrons are said to be stronger than inner orbit electrons because of their ability to break loose from the parent atom, and as a result they are referred to as *'valence electrons'*. The outer orbit in which valence electrons exist is called the *'valence band'*. It is the electrons from this band that are being considered in much of the discussions in the section to follow.

Crystal chemical bonds result through sharing of valence electrons. In materials such as Si, Ga and As, the outer-shell valence configuration can be represented by

$$\begin{aligned} &\text{Si} \rightarrow 3s^2\,3p^2 \\ &\text{Ga} \rightarrow 4s^2\,4p^1 \\ &\text{As} \rightarrow 4s^2\,4p^3 \end{aligned}$$

Here the core is not shown and the superscripts denote the number of electrons in the subshells (i.e. s and p orbitals). With this concept in mind, the structure of the atoms shown in Figure 12–2 can be simplified by representation as in Table 12–1.

Table 12–1 Periodic table

GROUP II	GROUP III	GROUP IV	GROUP V	GROUP VI
				
$Be^{4}_{9.01}$	$B^{5}_{10.82}$	$C^{6}_{12.01}$	$N^{7}_{14.008}$	$O^{8}_{16.0}$
$Mg^{12}_{24.32}$	$Al^{13}_{26.97}$	$Si^{14}_{28.09}$	$P^{15}_{31.02}$	$S^{16}_{32.07}$
$Zn^{30}_{65.38}$	$Ga^{31}_{69.72}$	$Ge^{32}_{72.60}$	$As^{33}_{74.91}$	$Se^{34}_{79.0}$
$Cd^{48}_{112.4}$	$In^{49}_{114.8}$	$Sn^{50}_{118.7}$	$Sb^{51}_{121.8}$	$Te^{52}_{127.6}$

Note: Numbers in the table refer to the atomic number and the atomic weight.

12.2.1 A compound semiconductor

Gallium arsenide is a compound semiconductor which may be defined as a semiconductor made of a compound of two elements (as opposed to silicon, which is a single element semiconductor). From Table 12–1, which shows the materials in a periodic table, it is possible to deduce the manner in which III–V semiconductors can be produced. For example, gallium, having three valence electrons, can be combined with arsenic, which has five valence electrons, to form the compound GaAs.

Figure 12–2 shows the arrangement of atoms in a gallium arsenide substrate material. Note the alternate positioning of gallium and arsenic atoms in their exact crystallographic locations. Since gallium arsenide is a binary semiconductor, special care is required during the processing to avoid high temperatures that could result in dissociation of the surface, this being one of the basic difficulties in the growth of GaAs bulk material.

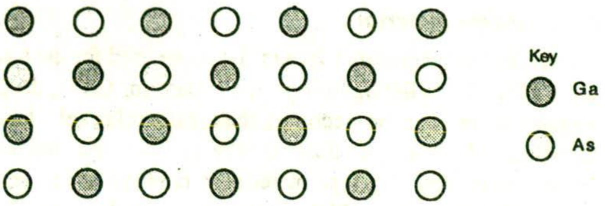


Figure 12–2 Arrangement of atoms in GaAs substrate

12.2.2 Doping process

Much as it is with silicon, it is necessary to introduce impurities into the semi-insulating GaAs material in order to facilitate the creation of switching devices. Selection of the impurity and its concentration density determine the behavior of the switching element. According to the dopant used, both n-type and p-type material can be realized.

12.2.2.1 n-type material

Group IV elements such as silicon can act as either donors (i.e. on Ga sites) or acceptors (i.e. on As sites). Since arsenic is smaller than gallium and silicon (the covalent radius for Ga is 1.26 Å and for As is 1.18 Å), group IV impurities tend to occupy gallium sites. Thus, silicon is used as the dopant for the formation of n-type material as shown in Figure 12–3.

The shrinkage of atomic radii across a given row of the periodic table (Table 12–1) can best be explained by noting that in any given period, electrons are added to s and p orbitals, which are not able to shield each other effectively from the increasing positive nuclear charge. Thus an increase in the positive charge of the nucleus results in an increase in the effective nuclear charge, thereby decreasing the effective atomic radius. This is why, for example, an As atom is smaller than a Ga atom.

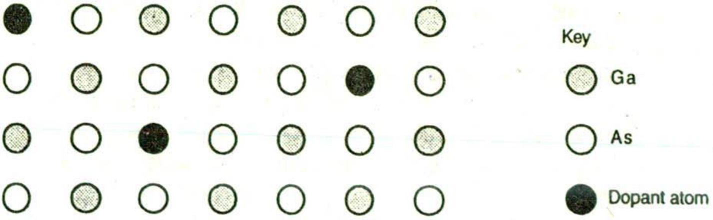


Figure 12–3 n-type material

12.2.2:2 p-type material

Beryllium (Be) or magnesium (group II) can be used for the formation of p-type material. Since Be is the lightest p-type dopant for GaAs, deep implantation of the dopant atoms can be accomplished with relatively less lattic damage. Nevertheless, Mg is also finding its way as a suitable dopant in a number of processes. Formation of p-type material is fundamental to both JFET. and CE-JFET (i.e. complementary JFET) processes, to be described in the later part of this chapter.

12.2.3 Channeling effect

The whole concept of crystal orientation becomes important during

- the etching of the crystal;
- ion implantation;
- passivation.

This introduces an 'orientation dependency' that influences the properties of GaAs field effect transistor. For example, during implantation, when a high energy ion enters a single crystal lattice at a critical angle to the major axis of the GaAs crystal, the ion is steered down the open directions of the lattice. This steering is called *axial channeling*. This implies that if a random equivalent direction is not used during ion implantation, the depth distribution will be greater than those predicted by range statistics which are used to establish penetration depth.

The channeling effect is not as dramatic in the <100> direction when compared with <110> direction. Many of the current GaAs wafers employ the <100> direction. It should be noted that the profile difference between the aligned <100> direction implant and any other direction of implant has a significant influence upon the threshold voltages of the fabricated devices.

12.2.4 Energy band structure

One of the important characteristics that is attributed to GaAs is its superior electron mobility brought about as the result of its energy band structure as shown in Figure 12–4.

Gallium arsenide is a direct gap material with valence band maximum and conduction band minimum coinciding in k space at the Brillouin zone centers. Valleys in the band structure that are narrow and sharply curved correspond to electrons with low effective mass state, while valleys that are wide with gentle curvature are characterized by larger effective masses.

The curvature of the energy versus electron momentum profile determines the effective mass of electrons traveling through the crystal. The minimum point of gallium arsenide's conduction band is near the zero point of crystal-lattice momentum, as opposed to silicon, where conduction band minimum occurs at high mojmentum. Now, mobility, μ, depends upon

- concentration of impurity, *N*;
- temperature, *T*;

  and is inversely related to
- electron effective mass, m_e.

For GaAs, the effective mass of these electrons is 0.067 times the mass of a free electron (i.e. $0.067m_e$, where m_e is the free electron rest mass). This means electrons travel faster in gallium arsenide than in silicon as the result of their superior electron mobility brought about by the shapes of their conduction bands. Electrons in the higher valleys have high mass and strong intervalley scattering and therefore exhibit very low mobility, which is very similar to conduction electrons in silicon.

Furthermore, gallium arsenide is a direct-gap semiconductor. Its conduction band minimum occurs at the same wave vector as the valence band maximum (Figure 12–4), which means little momentum change is necessary for the transition of an electron from the conduction band to the valence band. Since the probability of photon emission with energy nearly equal to the band gap is somewhat high, GaAs makes an excellent light-emitting diode. Silicon on the other hand, is an indirect-gap semiconductor since the minimum associated with its conduction

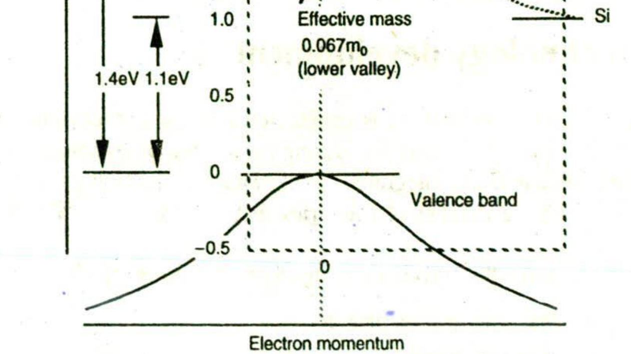


Figure 12–4 Energy band structure of silicon and gallium arsenide

band is separated in momentum from the valence band minimum. Therefore it cannot be a light-emitting device.

12.2.5 Electron velocity-field behavior

As the applied electric field, *E*, across the GaAs material is increased, the charge carriers, that is electrons in this case, gain energy from the applied field. At the same time, through collisions (i.e. optical phonon scattering) with the lattice, the electrons also lose a small portion of this energy. So long as the resultant balance is positive, the energy and drift velocity of the charge carriers increases with an increase in the applied field. However, at some point, the energy gained from the field becomes equal to the energy lost as the result of collisions. This results in the drift velocity approaching a limiting value referred to as the saturation velocity, v_{sat}.

Since gallium arsenide is a multivalley semiconductor, when the energy of lower valley electrons rises sufficiently, that is at electric fields greater than approximately 3500V/cm, electrons become 'hot'. There is a region in the electron velocity-field characteristics where some of the 'hot' electrons populate an upper conduction band that is characterized by larger electron effective mass. The resultant effect is a reduction in the number of high mobility electrons and hence the drift velocity.

In this region the drift velocity is no longer proportional to the electric field, but instead passes through a maximum of about $2*10^7$ cm/sec with increasing field, and decreases to an electric field independent saturation value of about $1.4*10^7$ cm/sec.

The velocity-field characteristics illustrating the three regions of interest are shown in Figure 12–5. For convenience of comparison, characteristics for silicon are also illustrated. From the figure it can readily be noted that in low electric field regions, silicon has a much lower mobility than gallium arsenide. This increases monotonically until the drift velocity saturates at a value of about $1*10^7$ cm/sec.

12.3 Technology development

Although this technology is confronted with similar technological problems as was silicon in the mid-1970s, during the last few years considerable progress has been made in GaAs integrated circuitry and the technology has progressed to the point where a number of foundries that provide GaAs fabrication are now in operation.

Typically, the current offerings have the following characteristics:

- less than one-micron gate geometry;
- less than two-micron metal pitch;
- up to four-layer metal;

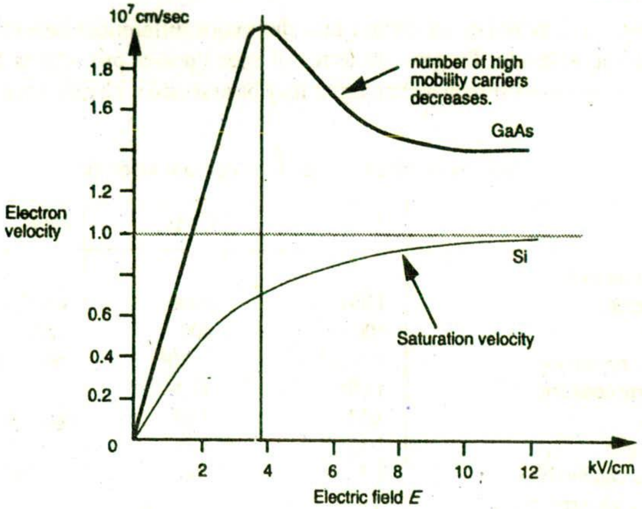


Figure 12–5 Electron velocity versus electric field for silicon and gallium arsenide

- 'ON' and 'OFF' devices;
- four-inch diameter wafers;
- suitability for clock rates in the range 1–2 GHz.

The salient features of this technology include:

- Electron mobility of six to seven times that of silicon, resulting in very fast electron transit times.
- Saturated drift velocity for GaAs and silicon are approximately equal, that is, $1.4*10^7$ cm/sec and $1.0*10^7$ cm/sec respectively. However, what is significant is that for GaAs saturation velocity occurs at a lower threshold field than for silicon.
- Large energy bandgap offers bulk semi-insulating substrate with resistivities in the order of 10^7 to 10^8 ohm.cm. This minimizes parasitic capacitances and allows easy electrical isolation of multiple devices in a single substrate.
- Radiation resistance is stronger due to absence of gate oxide to trap charges.
- A wider operating temperature range is possible due to the larger bandgap. GaAs devices are tolerant of wide temperature variations over the range –200 to +200°C.
- Direct bandgap of GaAs allows efficient radiative recombination of electrons and holes; this means forward-biased pn junctions can be used as light-emitters. Thus, efficient integration of electronics and optics becomes possible.
- Up to 70% reduction in power dissipation can be obtained over the fastest of the silicon technology such as ECL.

Table 12–2 provides an insight into the major differences between silicon and gallium arsenide. Progress in terms of speed/power projections for GaAs and commonly used silicon technologies may be assessed with reference to Figure 12–6.

Table 12–2 Comparisons between silicon and gallium arsenide

Properties	*Si*	*GaAs*	*Units*
Intrinsic mobility			
Electrons	1300	8000	cm^2/V.sec
Holes	500	400	cm^2/V.sec
Intrinsic resistivity	2.2*10^5	1*10^8	ohm.cm
Dielectric constant	11.9	13.1	
Density	2.33	5.32	gm/cm^3
Energy gap	1.12	1.43	*eV*
Thermal conductivity	1.5	0.46	W/cm° K
Effective electron mass	0.97m_e	0.067m_e	
Coefficient of thermal expansion	2.6*10^{-6}	5.9*10^{-6}	/°C
Vapor pressure (900°C)	7.5*10^{-19}	7.5*10^{-3}	mmHg
Breakdown field	3*10^5	4*10^5	V/cm
Schottky barrier height ϕ_B	0.4–0.6	0.7–0.8	V

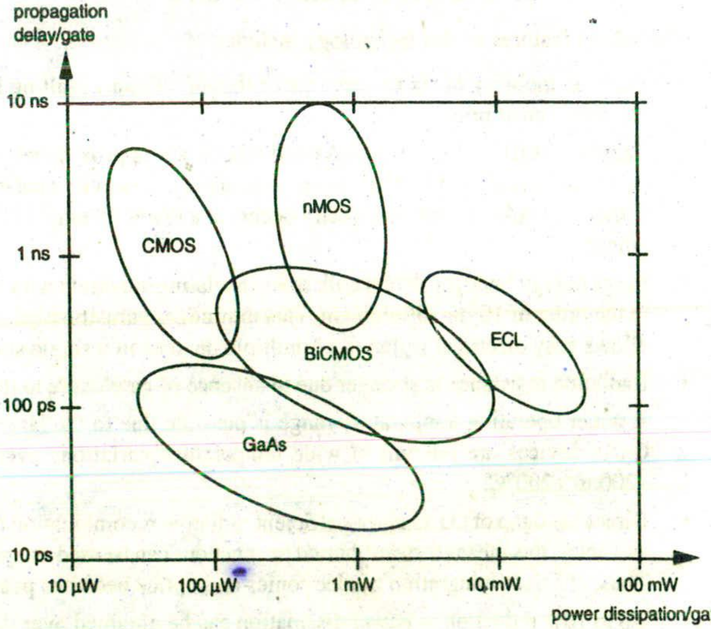


Figure 12–6 Speed/power performance projections for GaAs and Si

Table 12–3 Comparison between CMOS, bipolar and GaAs technologies

CMOS	*Bipolar*	*GaAs*
• Low dissipation	• High dissipation	• Medium dissipation
• High I/P impedance — low drive current	• Low I/P impedance — high drive current	• High I/P impedance — below ϕ_B
• High noise margin	• Medium noise margin	• Low noise margin
• Medium speed — high voltage swing	• High speed — low voltage swing	• Very high speed — low voltage swing
• High packing density	• Low packing density	• High packing density
• High delay sensitivity to load — fan-out	• Low delay sensitivity to load — fan-out	• High delay sensitivity to fan-in and fan-out
• Low output drive	• High output drive	• Low output drive
• $g_m \alpha V_{in}$	• $g_m \alpha e^{V_{in}}$	• $g_m \alpha V_{in}$
• Bidirectional	• Unidirectional	• Bidirectional possible
• Ideal switching device	• Not ideal switching device	• Reasonable switching device
• Medium f_t	• High f_t at low current	• Very high f_t
• Indirect gap	• Indirect gap	• Direct gap — good light-emitter
• Mask levels 12 to 16	• Mask levels 12 to 20	• Mask levels 6 to 10

In view of rapid developments in silicon technology itself, it is also appropriate to compare gallium arsenide with CMOS and BiCMOS. This comparison is highlighted in Table 12–3.

For very high speed operation in a semiconductor medium, three factors become significant, namely:

- carrier mobility;
- carrier saturation velocity;
- existence of semi-insulating substrate.

Gallium arsenide mostly fulfills the requirements and, together with its moderate power dissipation, provides the technology base for a new generation of circuits and subsystems.

12.3.1 Gallium arsenide devices

During the last few years a number of different devices have been developed. The so-called 'first generation' of GaAs devices includes:

- depletion-mode metal semiconductor field-effect transistor, D–MESFET;
- enhancement-mode metal-semiconductor field-effect transistor, E–MESFET;
- enhancement-mode junction field-effect transistor, E–JFET; and
- complementary enhancement-mode junction field-effect transistor, CE–JFET.

First generation GaAs gates have exhibited switching delays as low as 70 to 80 psec for a power dissipation in the order of 1.5 mW to 150 μW.

There are other more sophisticated 'second generation' devices such as:

- high electron mobility transistor, HEMT;
- heterojunction bipolar transistor, HBT.

Electron mobility in second generation transistors can be up to five times greater than in the first generation. In consequence, very fast devices are possible.

However, in the following sections we will concentrate on establishing some of the fundamental principles of GaAs design methodology for the first generation devices only, particularly the predominant MESFETs, which are now at a stage of development that enables them to be incorporated in very fast VLSI systems.

12.3.2 Metal semiconductor FET (MESFET)

The gallium arsenide field-effect transistor, a bulk-current-conduction majority-carrier device, is fabricated from bulk gallium arsenide by high-resolution photolithography and ion implantation into a semi-insulating GaAs substrate. Processing is relatively simple, requiring no more than six to eight masking stages. For the purpose of comparison, Figure 12–7 shows the evolution of process complexity in terms of mask count as function of time for both silicon and gallium arsenide technologies.

The structure of the basic MESFET as shown in Figure 12–8 is very simple. It consists of a thin n-type active region joining two ohmic contacts with a narrow metal Schottky barrier *gate* that separates the more heavily doped *drain* and *source*.

GaAs MESFETs are similar to silicon MOSFETs. The major difference is the presence of a Schottky diode at the gate region which separates two thin n-type active regions, that is, source and drain, connected by ohmic contacts. It should be noted that both D type and E type MESFETs, that is, 'ON' and 'OFF' devices, operate by the depletion of an existing doped channel. This can be contrasted with silicon MOS devices where the E (enhancement) mode transistor functions by inverting the region below the gate to produce a channel, while the D (depletion)

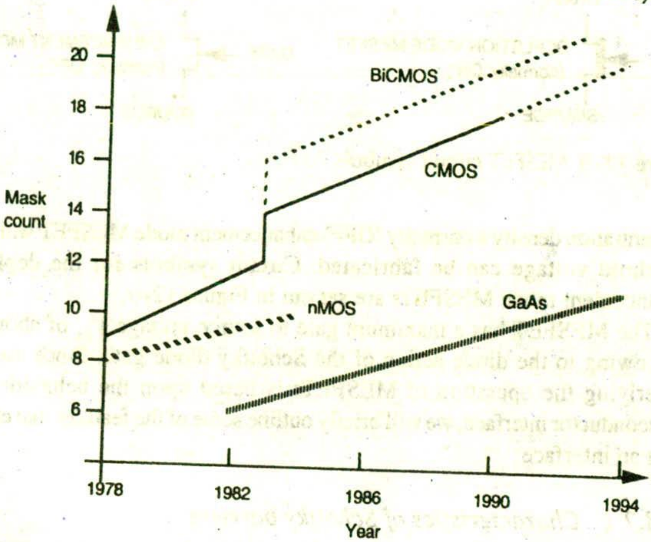


Figure 12–7 Evolution of process complexity for silicon and gallium arsenide technologies

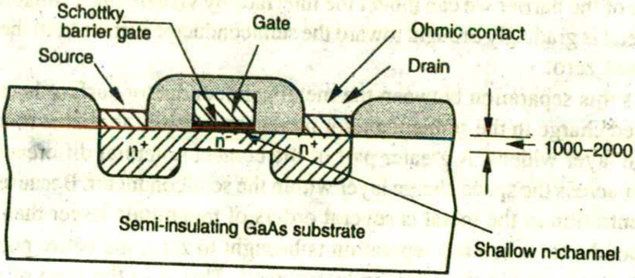


Figure 12–8 Side view for basic MESFET

mode device operates by doping the region under the gate slightly in order to shift the threshold to a normally 'ON' condition.

This similarity provides us with the basis for extending to gallium arsenide the design methodology used so successfully in silicon to simplify circuit and system design and layout issues.

The D–MESFET is normally 'ON' and its threshold voltage, V_{tdep}, is negative. The E–MESFET is normally 'OFF' and its threshold V_{tenh} is positive. The threshold voltage is determined by the channel thickness, a, and concentration density of the implanted impurity, N_D. A highly doped, thick channel exhibits a larger negative threshold voltage. By reducing the channel thickness, and decreasing the

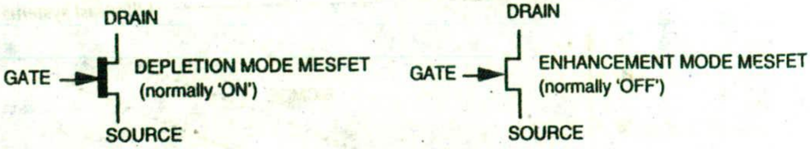


Figure 12–9 MESFET circuit symbols

concentration density a normally 'OFF' enhancement mode MESFET with a positive threshold voltage can be fabricated. Circuit symbols for the depletion and enhancement mode MESFETs are set out in Figure 12–9.

The MESFET has a maximum gate to source voltage V_{gs} of about 0.7–0.8 volt owing to the diode action of the Schottky diode gate. Since the principle underlying the operation of MESFETs is based upon the behavior of metal-semiconductor interface, we will briefly outline some of the features that characterize such an interface.

12.3.2.1 Characteristics of Schottky barriers

Wher. a metal is brought into contact with a semiconductor, an electrostatic potential barrier (refered to as Schottky barrier) is created at the interface as the result of the difference in the work function of the two materials. To appreciate the physical nature of the barrier we can model the interface by visualizing a situation whereby the metal is gradually brought toward the semiconductor surface until the separation becomes zero.

As this separation between the metal-semiconductor surface is reduced, the induced charge in the semiconductor increases, while at the same time the space charge layer widens. A greater part of the contact potential difference begins to appear across the space charge layer within the semiconductor. Because the carrier concentration in the metal is several orders of magnitude larger than that in the semiconductor when the separation is brought to zero, the entire potential drop then appears within the semiconductor itself. This is in the form of a depletion layer situated adjacent to the metal and extending into the semiconductor. A simplified view of such a transistor showing the depletion layer profile is shown in Figure 12–10 for two conditions, one when the drain to source voltage V_{ds} is zero and the other when it is greater than the saturation voltage.

12.3.3 GaAs fabrication

Although there are various approaches that are currently used, high-pressure liquid-encapsulated Czochralski (LEC) growth of gallium arsenide crystals from high purity pyrolytic boron nitride (PBN) crucibles is becoming the primary growth technique over several other methods that have emerged during the last few years.

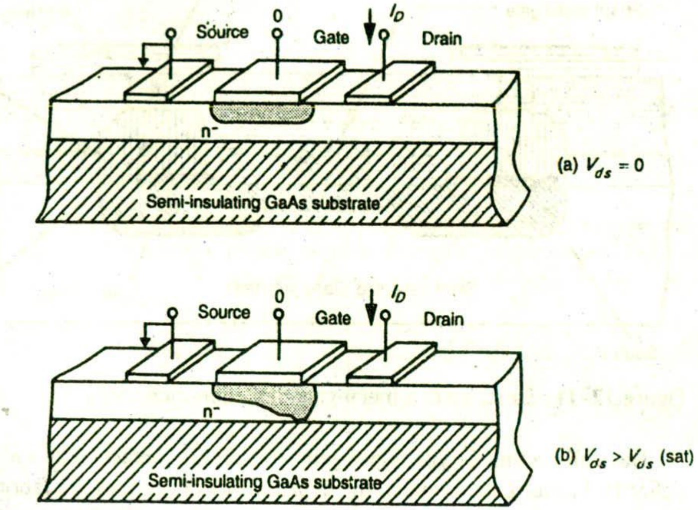

Source
0
Gate
I_D
Drain
n⁻
Semi-insulating GaAs substrate
(a) $V_{ds} = 0$
Source
0
Gate
I_D
Drain
n⁻
Semi-insulating GaAs substrate
(b) $V_{ds} > V_{ds}$ (sat)

Figure 12–10 Depletion profile of a MESFET

Since preference is usually for wafers grown in the <100> orientation, much of the success of the above method is achieved as the result of the ability to grow LEC material in the <100> direction, which produces relatively large diameter, round (100) wafers that are thermally stable and have superior semi-insulating properties.

Since the <110> cleavage planes are at a right angle, square chips can be obtained with a diamond scribe and break. This means that by adhering to the <100> growth plane many of the problems associated with cutting and subsequent handling can be alleviated.

The sequence for GaAs wafer preparation is very similar to that of silicon wafer preparation technique. The first step involves mechanically grinding the As-grown boules to a precise diameter and incorporating orientation flats. This is followed by

- wafering using a diamond ID saw;
- edge rounding;
- lapping;
- polishing;
- wafer scrubbing.

12.3.3.1 Depletion-mode MESFET

The profile for the metal gate depletion-mode MESFET (D–MESFET), the most mature of the current GaAs technologies, is illustrated in Figure 12–11.

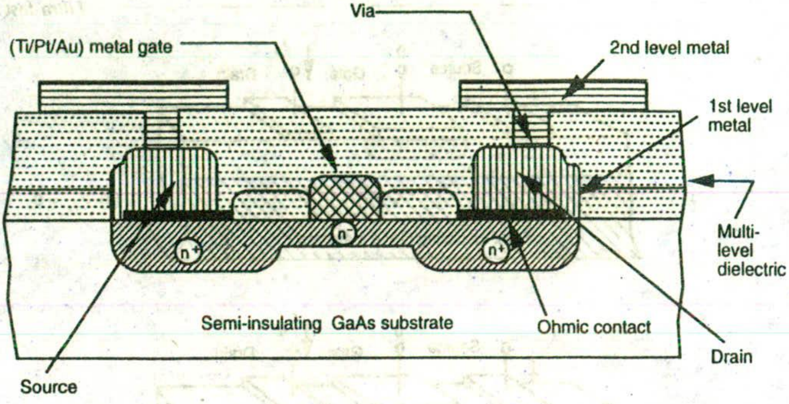


Figure 12–11 Structure of a metal-gate depletion-mode MESFET

Basically, a thin n–type region joins two ohmic contacts with a narrow metal Schottky barrier gate. Usually, the depletion-mode devices are fabricated using the planar process where n-type dopants (having concentration density typically in the range of $1*10^{17}$ cm^{-3} to $2*10^{17}$ cm^{-3}) are directly implanted into the semi-insulating GaAs substrate to form the channel as well as the more heavily doped source and drain regions. The semi-insulating substrate is ideal for all 'ion implantation' planar technology. The gate and first level interconnect metallizations are typically deposited by E-beam evaporation techniques. The gate length and its position relative to the source and drain contacts have a significant influence upon the transconductance of the device and control the performance of the MESFET. The conducting n-channel is confined between the gate depletion region and the semi-insulating GaAs substrate. By varying the channel thickness (usually in the range 1000 Å to 2000 Å) and the doping level of the active region, it is possible to vary the threshold, V_{tdep}, to the desired negative value, that is, in the range –0.5 V to –2.0 V.

12.3.3.1.1 Depletion-mode planar process flow

The driving force and indeed much of the success associated with silicon technology were brought about as the result of the presence of a stable native oxide which was readily produced through the oxidation of silicon. However, owing to the absence of a stable native oxide, GaAs technology relies on deposited dielectric films for passivation and/or encapsulation.

The fabrication process varies from foundry to foundry. However, one approach is illustrated in Figure 12–12, which entails the use of 3-inch or 4-inch liquid-encapsulated Czochralski (LEC) wafers. Initially, the GaAs substrate is coated with the first level of insulator, that is a thin layer of silicon nitride (Si_3N_4), which is sputtered on the GaAs substrate. This thin film of insulator remains on the wafer throughout the processing steps that follow, allowing the annealing of GaAs at temperatures of up to 900°C. The next step entails the formation of an n^- type active layer. This is achieved by direct ion implantation into the GaAs semi-insulating substrate through the insulating layer where the photoresist is used as the implant mask. Implantation of Si^+ ions takes place at about 220 to 230 keV to a dose of approximately $6*10^{12}/cm^2$. There are only two main implantation steps:

1. a shallow high-resistivity n^- layer for formation of the channel layer; and
2. a deep low-resistivity n^+ layer for the formation of source and drain.

The resultant channel resistance is in the order of 1000 to 2500 ohm/square, which is too high for source and drain contacts. Therefore, by keeping the surface concentration at the source, and drain regions relatively high by additional implantation, it is possible to reduce the contact resistances of these contacts.

The wafer is then coated with the interlevel dielectrics, SiO_2 by CVD (chemical vapour deposited) process. SiO_2 layer has a thickness of 400 to 500 nm and is deposited over the Si_3N_4 layer primarily to provide protection against physical damage. This is followed by an anneal in a hydrogen ambient at a temperature of about 800–850°C for approximately 30 minutes. This encapsulation phase is very important as it prevents out-diffusion of arsenic, brought about as the result of high vapour pressure associated with GaAs (Table 12–2) when subjected to temperatures over 600°C or so during the anneal step.

It should be noted that there are only a few capping materials that can be used in the process since the mechanical stability of the thin film encapsulation layer depends upon the stress that is present at the interface.

There are several sources that this stress can originate from:

- lattice mismatch;
- intrinsic stress of the encapsulation layer itself;
- thermal mismatch.

For example, the coefficients of thermal expansion for the commonly used capping materials such as Si_3N_4 and SiO_2 are:

$$Si_3N_4 = 3.2*10^{-6}/°C$$
$$SiO_2 = 0.5*10^{-6}/°C$$

This can be compared with GaAs, which has a thermal expansion coefficient of $5.9*10^{-6}/°C$. Thus, it is readily recognizable that SiO_2 has the greatest mismatch.

Since Si_3N_4 has a dielectric constant of 7, compared to 3.9 for silicon dioxide, a sandwich structure of SiO_2 and Si_3N_4 increases the effective dielectric constant of the insulator layer. Furthermore, SiO_2 was initially employed as the first-level capping material. However, it was found that Ga can diffuse through this layer.

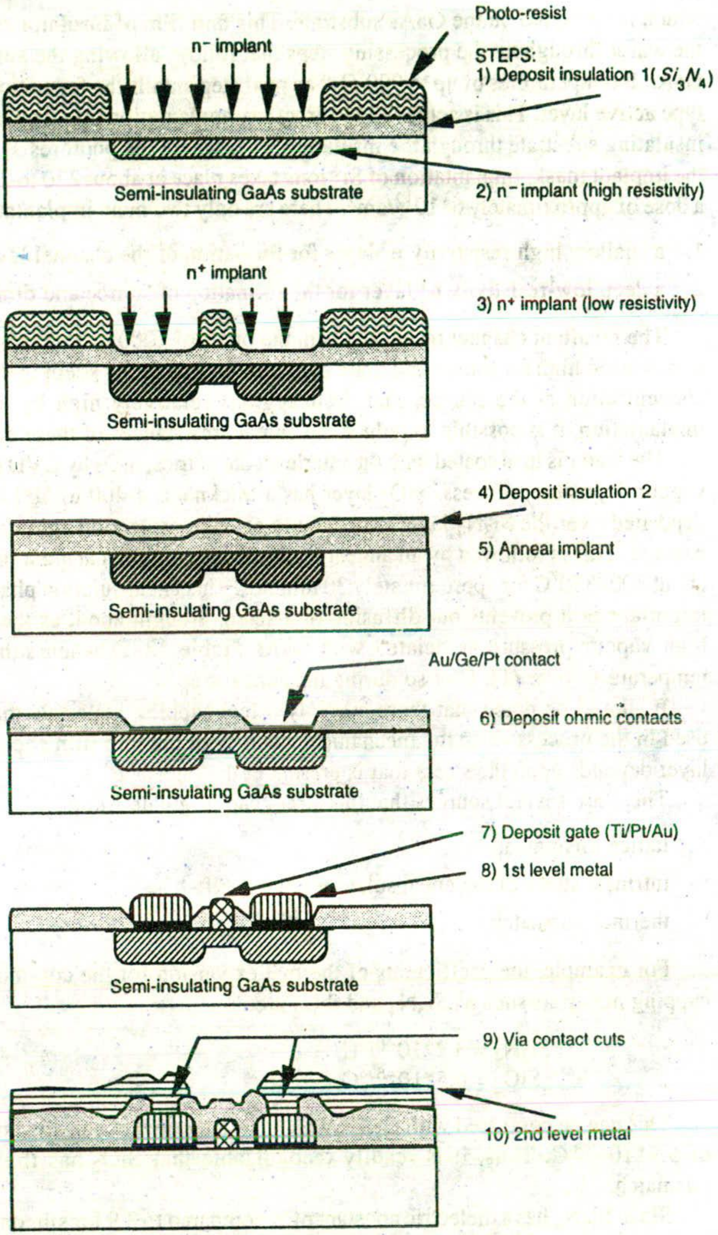


Photo-resist
n$^-$ implant
STEPS:
1) Deposit insulation 1(Si_3N_4)
Semi-insulating GaAs substrate
2) n$^-$ implant (high resistivity)
n$^+$ implant
3) n$^+$ implant (low resistivity)
Semi-insulating GaAs substrate
4) Deposit insulation 2
5) Anneal implant
Semi-insulating GaAs substrate
Au/Ge/Pt contact
6) Deposit ohmic contacts
Semi-insulating GaAs substrate
7) Deposit gate (Ti/Pt/Au)
8) 1st level metal
Semi-insulating GaAs substrate
9) Via contact cuts
10) 2nd level metal

Figure 12–12 A typical gallium arsenide DMESFET metal gate process using planar technology

This problem was subsequently alleviated by using Si_3N_4 as the first-level insulator with SiO_2 as the second-level insulator.

The next step in the process entails defining the MESFET gates, the ohmic contacts and the first-level metal interconnects. There are several points that must be considered during this phase. These are:

- The metals must be carefully alloyed to ensure reliable low resistance contacts, that is less than 10^{-6} Ω-cm^2.
- The ohmic contacts between the metal interconnect and the source and drain are deposited by evaporation using E-beam technology. A thin layer of gold-germanium-nickel (Au/Ge/Ni) or gold-germanium-platinum (Au/Ge/Pt) is alloyed on the wafer at a temperature of about 450°C to 500°C.
- One of the most critical steps in the fabrication process is the gate metallization.
- Schottky gates, together with first-level metal for interconnects, are formed by multilayer gold-refractory thin films such as titanium/platinum/gold (Ti/Pt/Au: 300 Å/400 Å/3000 Å) or alternatively titanium/tungsten/gold (Ti/W/Au) alloys deposited by E-beam evaporation. Titanium provides a good, high barrier, Schottky contact, but it has a high parasitic gate resistance. To reduce the parasitic resistance, gold is used as the top layer with platinum or tungsten as the intermediate layer. In the absence of either Pt or W layers, gold could diffuse into the GaAs surface, thus converting the Schottky contact into an ohmic one.

First-level metallization, which is about 3000 Å to 4000 Å, is accomplished by:

- delineating photoresist patterns;
- plasma etching the underlying insulator;
- deposition of the metal on GaAs wafer either by vacuum evaporation or by sputtering;
- photoresist lift-off.

The metal contacts and interconnects are precisely registered with the plasma-etched insulator windows. By fabricating the first-level metal within windows in the first-level insulator, and by ensuring that the first-level metallization thickness is close to the insulator thickness as in Figure 12–13, a more complex multilevel interconnect structure becomes possible due to the planar nature of the surface. Thus, a third-level metal and a fourth-level metal can readily be implemented.

Second-level metal is not in contact with gallium arsenide substrate; therefore platinum, which is used to prevent the interaction of gold with the GaAs surface (i.e. Au dissolves in GaAs), is usually eliminated from this step. Second-level metallization, which is about 7000 Å to 8000 Å thick, entails magnetron sputtered titanium/gold (Ti/Au: 300 Å/7000 Å) alloy only, which is followed by filling the vias between first-level and second-level metal. The sputtering process entails

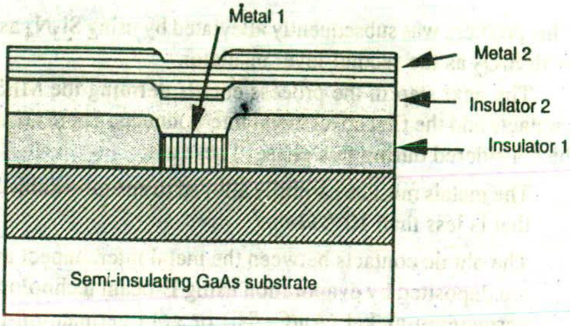


Figure 12–13 Metallization process

the physical deposition of a thin film by ion bombardment of the required material. Usually the deposition rate (i.e. thickness per unit time) depends upon the sticking coefficient of the depositing material and the nature of the sputtering equipment. The main feature of the magnetron itself is that it involves a set of powerful magnets, located behind the target surface, that provides an intense magnetic field for concentrating the plasma in the vicinity of the target.

The final step in the fabrication is passivation, used to protect against contamination and moisture. This entails a 0.4 μm to 0.5 μm thick passivation layer being deposited using a low temperature, plasma-enhanced chemical vapor deposition (PECVD) process. This is a'chemical deposition technique used for fabrication of both insulating and conducting films. The method is very similar to the low pressure CVD except that plasma excitation is provided in addition to the usual thermal energy.

Since in D–MESFETs any regions of the source or drain channel that are not under the gate are automatically strongly conducting, one does not require the precise alignments of the gate nor gate recesses to avoid parasitic source and drain resistances. However, in the metal-gate planar technology the position of the gate relative to the source and drain contacts has significant influence upon the performance of the device. Because of the very thin undepleted n$^-$ layer, the source resistance can be rather high, which subsequently causes the degradation of the transconductance, g_m.

Extension of the surface depletion layer cannot be avoided because of the presence of traps localized at the gallium arsenide surface. Subsequently, the extension of the interface depletion layer, owing to traps near the interface between the active layer and the substrate, has an influence on the drain resistance also. Hence process optimization is essential in order to minimize these resistances so that the device performance is not degraded.

It is interesting to note the close similarity between the planar implanted D–MESFET GaAs fabrication process and the Si planar process. This can readily be observed by noting that the GaAs substrate is totally protected by dielectric layers throughout the fabrication process. Cuts are made in the dielectric only where ohmic contacts, Schottky barriers, or interconnect metallizations are required. As far as process technology is concerned, the most difficult layer to control is the shallow, lightly doped high-resistance n^- MESFET channel layer. This implant layer determines the threshold voltage V_t of the MESFETs.

If some reduction in speed can be tolerated, then instead of using the exotic gold process for first-level and second-level metals it is possible to use the less costly aluminum. Thus, significant cost savings could be achieved at the expense of speed. This will be outlined in the following section on the self-aligned gate (SAG) process.

12.3.3.1.2 Ion implantation and annealing

The ion implantation and the subsequent annealing are very significant in this technology. In ion-implantation, doping is achieved by bombarding the semiconductor surface with a high-velocity ion beam. Doping density and dopants distribution in the semi-insulating material are controlled by varying the ion flux and velocity. Using this approach, crystal defects brought about as the result of ion bombardment are annealed at about 800–850°C. The advantages of ion implantation are:

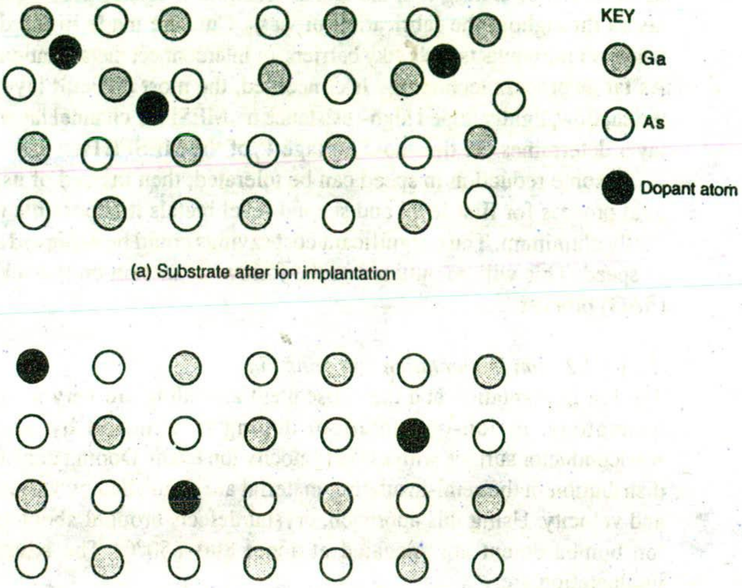
- independent control of doping level;
- independent control of doping profile;
- good reproducibility;
- ease of selective doping of selected areas.

The original arrangement of atoms on the crystal lattice was indicated in Figure 12–1. As implanted ions penetrate the GaAs substrate, they lose energy by several mechanisms, including the displacement of target atoms from the lattice sites. After ion implantation, the dopant atoms come to rest in the crystal and, as a result of interactions and collisions, the crystal lattice is disrupted as indicated in Figure 12–14(a).

Ions now occupying interstitial positions are electrically inactive. Annealing provides energy to the implanted impurities and results in moving the interstitial dopant ions into lattice positions where they become electrically active. Furthermore, the displaced substrate atoms are subsequently moved back to their crystallographic lattice locations (Figure 12–14(b)) which then gives the high electron mobility.

The extent of damage to the crystal depends on several factors, including:

- mass of the implanted ion;
- target mass;
- energy associated with the ion;
- dose;



(a) Substrate after ion implantation

(b) Substrate after annealing

Figure 12–14 Ion implantation before and after annealing

- temperature;
- displacement energies.

12.3.3.2 Enhancement-mode MESFET

The E–MESFET structure is similar to that of the D–MESFET, except for a shallower and more lightly doped channel. This means the channel is in 'pinch-off' at zero gate voltage, due to the built-in potential of the metal Schottky barrier gate. A positive gate voltage is required for the channel to begin conduction. In order to ensure that the depletion layer extends through the channel height at zero gate voltage, the gate is usually recessed into the underlying channel. The steps in the fabrication of the E–MESFET are somewhat similar to those for the D–MESFET.

12.3.3.2.1 Process details

Steps for fabrication of gallium arsenide enhancement/depletion-mode MESFETs are reproduced here to highlight some of the complexities of the process. The details of the process are:

1. Encapsulation phase:
    - wafer preparation;
    - encapsulation (deposition of first-level insulator Si_3N_4);

- alignment mark mask;
- alignment mark metallization and lift-off.

2. Ion implantation:
   - first Si^+ implant (E–MESFET) mask;
   - channel implant;
   - second Si^+ implant (D–MESFET) mask;
   - channel implant;
   - S/D implant (formation of source and drain) mask;
   - n^+ implant;
   - anneal.
3. Schottky junctions and first-level metal:
   - patterning ohmic contact mask;
   - plasma etching contact windows;
   - contact metallization (ohmic-Au/Ge/Ni);
   - contact definition and alloy;
   - H^+ implant mask;
   - H^+ implant (isolation);
   - Schottky gate mask;
   - plasma etch Schottky windows;
   - metallization (Ti/Pt/Au);
   - lift-off.
4. Second-level metal:
   - dielectric (SiO_2 sputter);
   - via cut mask;
   - metallization (Ti/Au) mask;
   - lift-off.
5. Scratch protection:
   - Si_3N_4 plasma deposition;
   - pad/scribe street mask;
   - plasma etch.

12.3.3.3 Self-aligned gate E/D process

An alternative approach in process technology is the self-aligned gate (SAG) process, which is showing promise and is beginning to emerge as a strong contender for silicon in the area of very high speed VLSI systems.

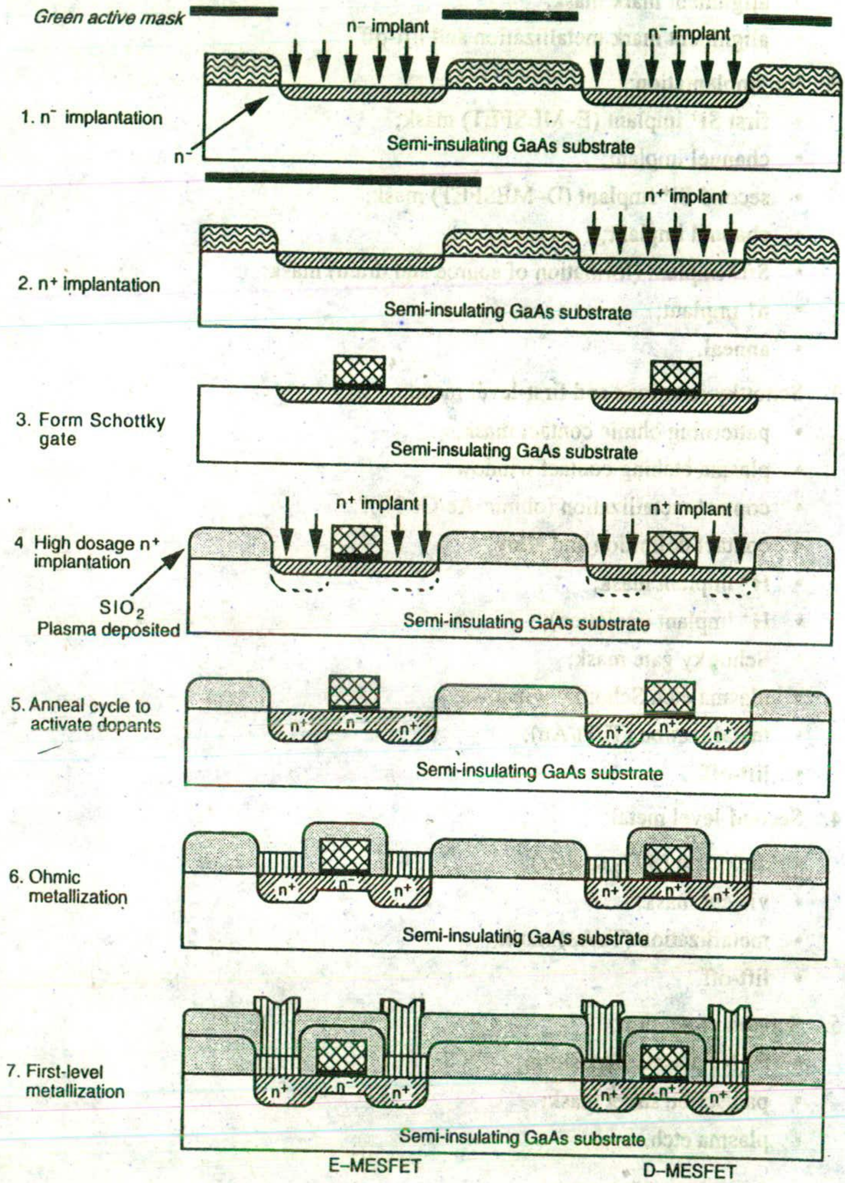

Green active mask
n⁻ implant
n⁻ implant
1. n⁻ implantation
n⁻
Semi-insulating GaAs substrate
n⁺ implant
2. n⁺ implantation
Semi-insulating GaAs substrate
3. Form Schottky gate
Semi-insulating GaAs substrate
n⁺ implant
n⁺ implant
4 High dosage n⁺ implantation
SIO₂ Plasma deposited
Semi-insulating GaAs substrate
5. Anneal cycle to activate dopants
n⁺
n⁻
n⁺
n⁺
n⁺
n⁺
Semi-insulating GaAs substrate
6. Ohmic metallization
n⁺
n⁻
n⁺
n⁺
n⁺
n⁺
Semi-insulating GaAs substrate
7. First-level metallization
n⁺
n⁻
n⁺
n⁺
n⁺
n⁺
Semi-insulating GaAs substrate
E–MESFET
D–MESFET

Figure 12–15 Self-aligned processing steps for GaAs E–/D–MESFET

Note that in the self-aligned process as shown in Figure 12–15, the n$^+$ implant regions prevent the extension of both the surface and the interface depletion layers, thus reducing the effect of the undepleted n$^+$ layer parasitic resistance which subsequently improves the performance of a device.

Process steps for a GaAs self-aligned gate are as follows:

- a n$^-$ implantation for formation of E–MESFET;
- a second n$^+$ implantation for formation of D–MESFET;
- formation of Schottky gates on an n-type GaAs layer;
- a third, n$^+$, implantation for formation of source and drain;
- an anneal cycle at 850°C to activate dopants;
- ohmic metallizations of the source and drain regions;
- interconnect metallizations.

Owing to the anneal cycle that requires a temperature up to 850°C to activate the dopants, it is necessary to choose a high-temperature-stable gate. Tungsten nitride has been found to be satisfactory as gate material. It has film resistivity ρ = 70 $\mu\Omega$-cm and Schottky barrier height $\phi_B = 0.8$ volt to n-type GaAs.

Since Schottky barrier gates on GaAs cannot be forward biased above 0.7 to 0.8 volt without drawing excessive currents, the permissible voltage swing is relatively low. This limits the noise immunity of the gate and places stringent fabrication requirements on threshold voltage control and uniformity.

As can be seen, this technology very closely resembles that of nMOS, which means it is very likely that ratio rule needs to be applied when designing logic circuits using the enhancement/depletion process shown in Figure 12.16.

In summary, the steps in the process entail defining the active areas (Green Mask) that would eventually form the E-type and the D-type MESFETs, followed by two ion implantions, that is, lightly doped for E-type and heavily doped (Yellow Mask) for D-type followed by formation of gate-metal (Red Mask).

The E-type MESFET is defined by intersection of Green and Red masks while the D-type MESFET is defined by intersection of the Green, Red and Yellow masks. This abstraction as an aid to design will be dealt with in more detail in Section 12.5.

12.3.3.4 Enhancement mode junction FET (E–JFET)

The operation of an E–JFET is similar to that of an E–MESFET, in which source and drain regions are formed by n$^+$ ion implantation while the channel is formed by n$^-$type implantation. However, in contrast to the E–MESFET, where the metal gate rests above the channel, in the E–JFET the gate is buried below the channel surface by p$^+$ implantation (using either beryllium or magnesium as the implant). Through this process, a pn junction is formed between the gate and the channel, as illustrated in Figure 12–17. The structure offers lower parasitic source and drain resistances than the E–MESFET owing to the doping of the channel region.

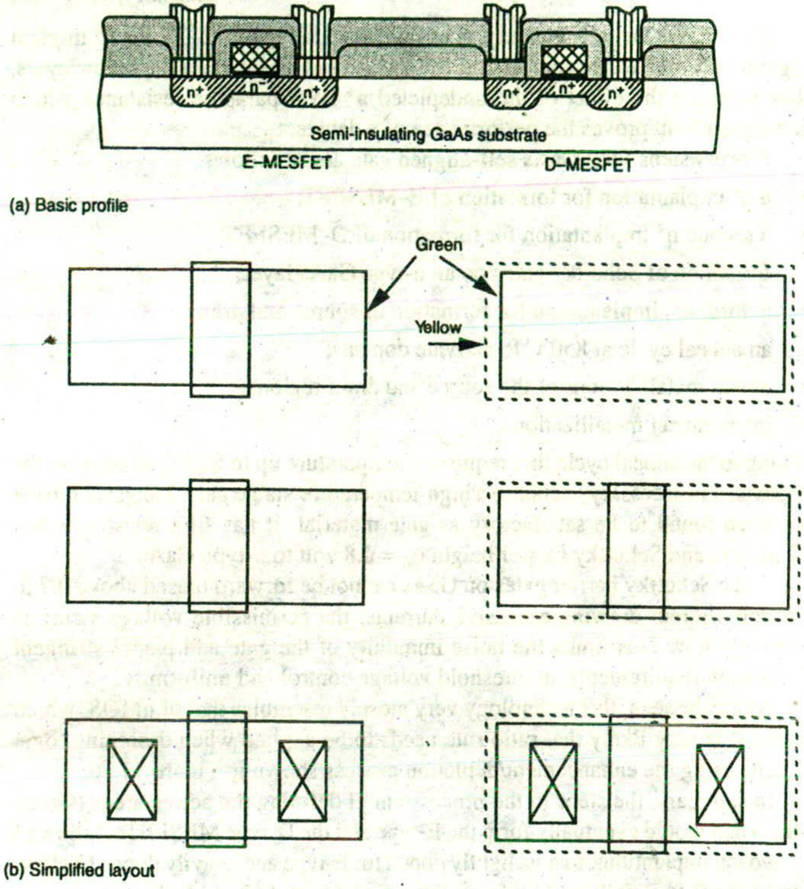


Figure 12–16 Enhancement/depletion self-aligned gate process

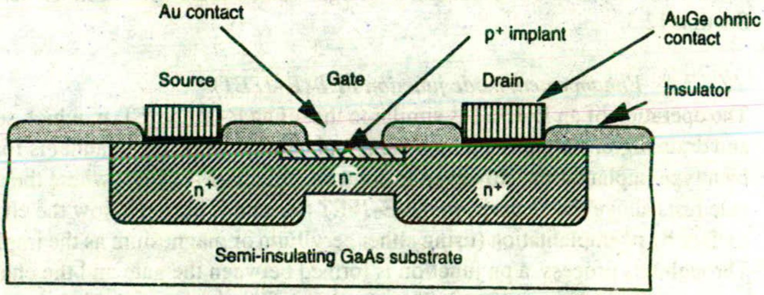


Figure 12–17 E–JFET structure

From previous discussions, the permissible voltage swing for E–MESFETs is rather low since Schottky barrier gates on GaAs cannot be forward biased above 0.7 to 0.8 volt without drawing excessive current. However, with E–JFETs, because of the larger built-in pn junction voltage, the device can be biased to about $V_{gs} = +1$ volt without incurring excessive conduction, thus alleviating some of the problems that are encountered in the control of the threshold voltage. E–JFETs are more difficult to fabricate than MESFETs primarily because of the additional p^+ implant step. It is necessary to have a precise control over the implant thickness to ensure that the desired pinch-off voltage of the device is maintained. However, here we have an additional advantage over the E–MESFET in the control of threshold voltage not only through the implant but also by adjusting the pn junction location.

A significant aspect of E–JFET technology is that complementary devices can be fabricated, whereas in MESFET technology considerable difficulty is encountered in forming Schottky barriers to a p-type implanted channel.

The presence of the additional pn junction sidewall gate capacitance makes the E–JFET slower than an equivalent E–MESFET. However, reduced power requirements together with larger logic voltage swings makes E–JFET technology a possible contender for the emerging ultra high speed VLSI systems.

12.3.3.5 Complementary enhancement-mode junction FET (CE-JFET)

A CE-JFET device, shown in Figure 12–18, is similar to the silicon complementary MOSFET. Here the nMOS depletion mode transistor, or alternatively, a resistive load, is replaced by a p-channel EJFET.

The n-channel and p-channel JFET is fabricated by a series of ion implantations into the semi-insulating GaAs substrate. The sequence entails:

- n^+ implantation;
- n^- channel implantation;

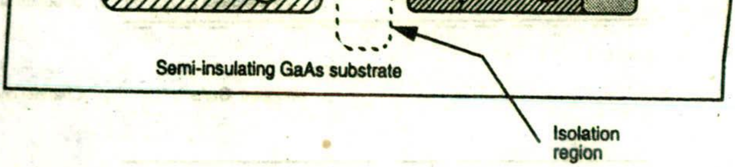


Figure 12–18 Complementary enhancement mode JFET structure

-29

- p^- channel implantation;
- p^+ implantation.

In a CE-JFET the ratio of the effective channel electron mobility μ_n of the n-channel device to that of the hole mobility μ_p of the p-channel device is given by:

$$\frac{\mu_n}{\mu_p} = 10$$

Thus, for a p-channel device requiring the same drain current I_{ds} as that of the complementary n-channel device, it is necessary that

$$W_p = 10W_n$$

where W_p = channel width for p-channel device,
W_n = channel width for n-channel device.

This means that circuits requiring equal numbers of p- and n-devices will consume large areas. Therefore, one must resort to other design methods such as precharge techniques, which require a single pull-up transistor to serve a number of n-transistors performing the logical functions.

12.3.3.6 High electron mobility transistor (HEMT)

Here, multilayered stuctures of very thin (10 Å to 100 Å) alternating layers of GaAs and AlGaAs are used. This multilayered approach is referred to as a multiquantum-well structure. The key in these structures is to place the donor atoms in a wider bandgap GaAlAs layer adjacent to an undoped GaAs channel layer, which receives the free electrons from the ionized donors. Electrons are transferred from the AlGaAs charge control layer to the undoped GaAs layer where they form a two-dimensional electron gas. Since the electrons are spatially separated from the ionized donors, they exhibit high mobility.

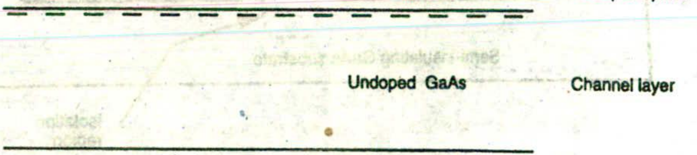


Figure 12–19 Basic structure of a high electron mobility transistor (HEMT)

Although there are variations to the processing steps, the basic structure is as illustrated in Figure 12–19 and will be seen to comprise four distinct layers:

- channel layer — GaAs;
- undoped spacer layer — AlGaAs;
- donor layer;
- cap layer.

12.4 Device modeling and performance estimation

VLSI designers, as a rule, should have a good knowledge of the behavior of the circuits they are designing. Even when large systems are being designed using computer-aided design processes, it is essential that the designs are based on a sound foundation of understanding if the system is to meet a given performance specification.

12.4.1 Device characterization

In order to preserve simplicity, the prime consideration in this section is to provide an approximate model for the MESFET which not only preserves the essential features of the device, but also assists the VLSI systems designer with performance estimations and optimization processes.

As the gallium arsenide transistor and the processes used to produce it have been introduced, it is now possible to gain some insight into the electrical characteristics of the basic GaAs MESFET circuits.

12.4.2 Drain to source current derivation

MESFETs are channel-area modulation devices, that is, they depend upon the capacitance of the Schottky barrier to control the effective charge in the channel. As for silicon MOS transistors, gallium arsenide devices have also three regions of operation:

- cut off;
- linear;
- saturation.

To appreciate some of the features that characterize I_{ds}, in the first instance, we will proceed with deriving a simple model that highlights first order effects only, and then focus attention on the more complex models, without losing our objective of simplicity which is so critical for VLSI systems designers.

Consider a typical structure (see Figure 12–20) where the majority carriers, that is, electrons, flow from source to drain. The current I_{ds}, as the result of this movement, is given by:

$$I_{ds} = -I_{sd} = \frac{\text{Charge induced in channel } (Q_c)}{\text{Electron transit time } (\tau)} \tag{12.1}$$

First, the electron tr^nsit time τ can be determined by noting

$$\tau = \frac{\text{Channel length } (L)}{\text{Channel velocity } (v)}$$

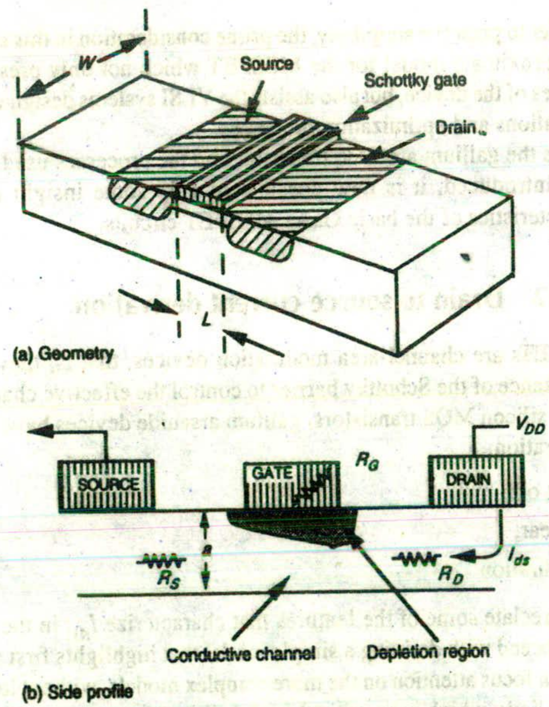
Now carrier velocity, that is the movement of electrons, is given by

$$v = \mu_n E_{ds}$$

where E_{ds} is the electric field between the drain and the source, and μ_n is the electron mobility.

The transit time τ can thus be expressed as

$$\tau = \frac{L}{\mu_n E_{ds}} \tag{12.2}$$



(a) Geometry

(b) Side profile

Figure 12–20 GaAs MESFET cross-sectional view

If we denote the average potential difference between the gate and the channel by V_{gb}, then owing to the shape of the depletion layer, this average potential can be written as

$$V_{gb} = 0.5(V_{gs} - V_t) \tag{12.3}$$

where V_{gs} is the gate to source voltage and V_t, is the threshold voltage of the device.

The threshold voltage V_t is defined as the gate voltage at which the depletion layer (Figure 12–17) just pinches off the channel, that is, the gate voltage that extends the depletion layer down to the substrate.

The average electric field E_{ds} along the length of the gate is:

$$E_{ds} = \frac{0.5(V_{gs} - V_t)}{L} \tag{12.4}$$

Upon substitution of equation 12.4 into equation 12.2, the transit time, τ, becomes

$$\tau = \frac{2L^2}{\mu_n(V_{gs} - V_t)} \tag{12.5}$$

The average electric field E_{ave} across the channel can also be approximated in terms of implant depth, a, and the voltage $(V_{gs} - V_t)$ that appears across the channel. Thus

$$E_{ave} = \frac{(V_{gs} - V_t)}{a}$$

The induced charge in terms of the device geometry and the average electric field becomes

$$Q_C = E_{ave}\varepsilon_r\varepsilon_0 WL \tag{12.6}$$

Upon substitution for E_{ave} in equation 12.6, the resultant expression for the induced charge becomes

$$Q_C = (WL)\left(\frac{\varepsilon_r\varepsilon_0}{a}\right)(V_{gs} - V_t) \tag{12.7}$$

Now, by combining equations 12.5 and 12.7 with equation 12.1, we obtain the principal result for the drain to source current I_{ds}

$$I_{ds} = \left(\frac{\mu_n\varepsilon_r\varepsilon_0}{2a}\right)\left(\frac{W}{L}\right)(V_{gs} - V_t)^2 \tag{12.8}$$

which, when rewritten, results in

$$I_{ds} = \beta(V_{gs} - V_t)^2 \tag{12.9}$$

where

$$\beta = \left(\frac{\mu_n \varepsilon_r \varepsilon_0}{2a}\right)\left(\frac{W}{L}\right).$$

β is a common parameter used in the SPICE MESFET model specification, denoted by K_p. For a typical process, K_p is in the order of 0.1 to 0.5 mA/V^2.

β may be seen to consist of a process dependent factor ($\mu_n \varepsilon_r \varepsilon_0/2a$), which contains all the process terms and a geometry dependent term (W/L), which depends on the actual layout of the transistor. The geometric terms in equation 12.8 are illustrated in Figure 12–20.

Sometimes the channel length of the MESFET is predetermined by the process, which means the designer can control the gain factor through varying the channel width of the MESFET only.

Equation 12.8 describes the behavior of the MESFET in the saturation region only. Now using a similar approach, it is possible to derive,a relation for the MESFET to represent the operation in the linear region also. The model in this region becomes:

$$I_{ds} = \beta[2(V_{gs} - V_t)V_{ds} - V_{ds}{}^2]\ ;\ V_{ds} < (V_{gs} - V_t) \text{ and } V_{gs} \geqslant V_t \qquad (12.10)$$

Special note should be made here that in GaAs the saturation of drain current, I_{ds} , with an increasing drain to source voltage, V_{ds}, is brought about by carrier velocity saturation, whereas in silicon the resultant saturation effect is due to 'channel pinch-off'.

12.4.2.1 More complete device equation

The model described by equation 12.10 unfortunately does not provide for a smooth transition between the saturation and the linear regions of MESFET operation. It is possible to modify equation 12.10 by including a hyperbolic tangent term that will facilitate this smooth transition between the two regions.

The modified model describing the behavior of a GaAs MESFET in the three regions can now be written as

$$I_{ds} = \begin{cases} V_{gs} - V_t < 0 \text{ (Cut off)} \\ \beta\left[(V_{gs} - V_t)^m + \lambda(V_{gs} - V_t)^b\, V_{ds}\right] \tanh(aV_{ds}) \\ V_{gs} - V_t > 0 \text{ (Linear and saturation)} \end{cases} \qquad (12.11)$$

Where λ is the channel length modulation factor and varies in the range 0.01 to 1.

Parameters λ and $\tanh(aV_{ds})$ are channel length modulation and hyperbolic tangent function respectively, while m and b are constants that are derived empirically. It should be noted that the hyperbolic tangent function $\tanh(aV_{ds})$ is used to describe the channel conductance at low drain-to-source voltage, V_{ds}. This effect is the result of the decrease in magnitude of the depletion region beneath the gate as the gate-to-source voltage, V_{gs}, is increased.

Usually *m* and *b* can be adjusted to suit a particular process. For example, with $m = 2$ and $b = 2$, the drain current I_{ds} as described by equation 12.11 reduces to:

$$I_{ds} = \begin{cases} V_{gs} - V_t < 0 \text{ (Cut off)} \\ \beta(V_{gs} - V_t)^2(1 + \lambda V_{ds})\tanh(aV_{ds}) \\ V_{gs} - V_t > 0 \text{ (Linear and saturation)} \end{cases} \qquad (12.12)$$

This is referred to as the *Curtice* model.

It is still possible to improve the device model further by considering the influence of velocity saturation. The new relation for the drain to source current referred to the Raytheon model is given by:

$$I_{ds} = \begin{cases} \beta\dfrac{(V_{gs} - V_t)^2}{1 + b(V_{gs} - V_t)}(1 + \lambda V_{ds})(1 - \left(1 - a\dfrac{V_{ds}}{3}\right)^3); \; 0 < V_{ds} < \dfrac{3}{a} \\ \dfrac{\beta(V_{gs} - V_t)^2}{1 + b(V_{gs} - V_t)}(1 + \lambda V_{ds}) \qquad V_{ds} \geq \dfrac{3}{a} \end{cases} \qquad (12.13)$$

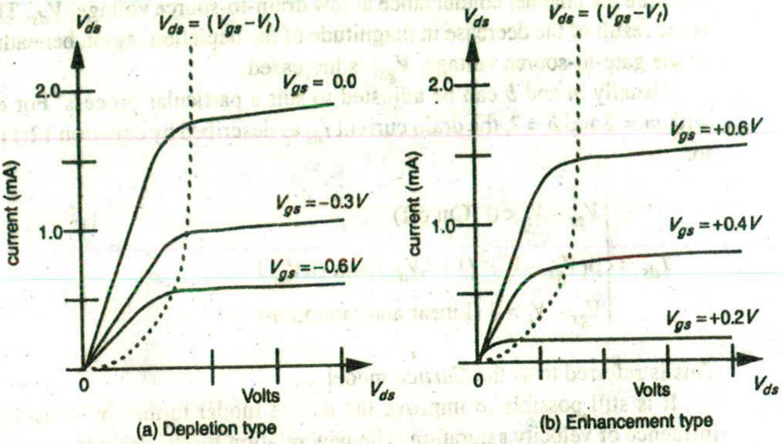
Where *b*, in equation 12.13, is an empirical term and the 'slope factor' *a* is used to take into consideration the influence of slope in both the linear and saturation regions.

12.4.2.2 V-I characteristics for GaAs MESFET

A typical voltage-current characteristic as described by equation 12.12 is shown in Figure 12–21 for both the depletion and the enhancement devices. When $V_{gs} < V_t$, the increase in the drain-to-source voltage V_{ds} above the saturation voltage $V_{ds\,(sat)}$ leads to current saturation. The saturation of drain current with increasing drain-to-source voltage is caused by velocity saturation in the high electric field in the channel.

The boundary between the linear and saturation regions defined by ($V_{ds} = V_{gs} - V_t$) is referred to as the *'knee voltage'*, and appears as a dashed line in Figure 12–21. Note that the drain current saturates at the same drain-to-source voltage V_{ds} and is independent of the gate-to-source voltage V_{gs}. This behavior can best be explained by noting that the critical electric field $E_{critical}$ in the channel is reached at the same drain-to-source voltage, V_{ds}, given by:

$$V_{ds} = E_{critical} * L$$



(a) Depletion type

(b) Enhancement type

Figure 12–21 Voltage v current characteristics for GaAs MESFET

As a matter of interest the critical electric field is in the order of 3500 V/cm.

As can be seen, the characteristic is similar to that of silicon gate technology, with the exception of the magnitude of the gate-to-source voltage V_{gs}, which is limited to about 0.8 volt. This limit is brought about by the presence of the Schottky diode at the gate region. This is illustrated in Figure 12–22 for both the E type and D type MESFETs.

Depending on whether the MESFET is operating with reverse $(V_t < V_{gs} < 0)$ or forward $(0 < V_{gs} < \phi_B)$ gate to source bias, the mode of operation is referred tc as:

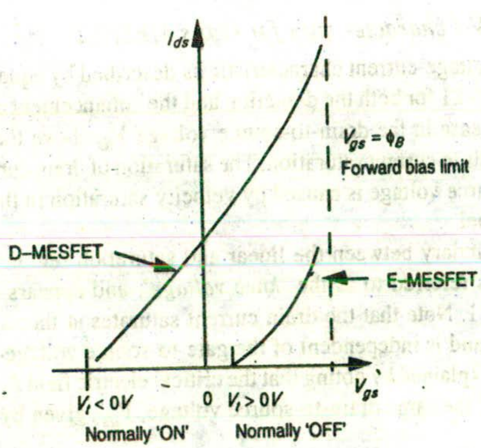


Figure 12–22 Transfer characteristics for MESFET

DEPLETION	→	REVERSE; $V_t < V_{gs} < 0$
ENHANCEMENT	→	FORWARD; $0 < V_{gs} < \phi_B$

12.4.2.3 Threshold voltage definition

The threshold voltage V_t that appears in our model has a significant influence upon the sizing of circuits. V_t is dependent upon the *pinch-off* voltage V_{po} and the barrier potential ϕ_B given by:

$$V_t = \phi_B - V_{po} \tag{12.14}$$

This relation simply means that the pinch-off voltage V_{po} is the total voltage; that is, both built-in potential and applied voltage necessary to completely deplete the channel of mobile charge carries. In other words, it is the gate voltage at which the depletion layer just pinches off the channel; that is, the gate voltage that extends the depletion layer down to the substrate as was illustrated in Figure 12–10(b). The pinch-off voltage is a function of both the channel thickness, a, and concentration density N_d and is always positive. The pinch-off voltage is:

$$V_{po} = \frac{qN_d a^2}{2\varepsilon_0 \varepsilon_r} \tag{12.15}$$

where

a = channel thickness of the n$^-$ implant
N_d = effective channel concentration density
q = electron charge ($1.6^* 10^{-19}$ Coulomb)
ε_0 = permittivity of free space ($8.85^* 10^{-14}$ F.cm^{-1})
ε_r = relative permittivity of GaAs (13.1).

This relation illustrates the difference that exists between the threshold voltage V_t and the pinch-off voltage V_{po}. This difference is somewhat significant and is brought about as the result of the built-in potential ϕ_B which can no longer be neglected as was the case with silicon. Furthermore, the threshold voltage V_t is very sensitive to both the channel thickness a (i.e. the vertical geometry) and the doping of the channel layer.

One significant aspect of the above model is that it illustrates the parameters that influence the transition of a device from being a depletion mode to an enhancement mode.

12.4.2.3.1 Threshold variation

In logic structure, the dynamic switching energy must exceed the energy stored in the load capacitor C_L. This can be written as:

$$P_g\tau_g > 1/2\,(C_L[\Delta V_o]^2) \tag{12.16}$$

where ΔV_o *is* the logic voltage swing, P_g is the gate dynamic dissipation and τ_g is the associated gate delay. To keep the dynamic switching energy small, the logic voltage swing ΔV_o must be kept small also. This requires precise control over the threshold voltages of both the D type and E type MESFETs not only between adjacent devices but also across the whole wafer.

In order to achieve such a control, it is necessary for the standard deviation of the threshold voltage σV_t to be less than 5% of the logic voltage swing ΔV_o. Thus the logic swing can be expressed as:

$$\Delta V_o > 20\,\sigma V_t$$

The logic swing for an E type MESFET is in the order of 500 mV. Above this value one can expect excessive gate current. Thus, the variation of the threshold voltage for the E–MESFETs over the chip must be better than

$$\begin{aligned}\sigma V_t &= \frac{\Delta V_o}{20}\\ &= 25\text{ mV}\end{aligned} \tag{12.17}$$

This can be compared with the D–MESFET in which the logic voltage swing ΔV_o can be larger than 1 V, which means tolerance to larger threshold voltage variation — that is, at least 50 mV, can be more readily accommodated.

Basically this implies that it becomes necessary to have a high degree of control over the threshold voltage and drain-to-source current to ensure that GaAs circuits with reasonable yields and circuit performance are produced. Also, owing to the almost exponential influence of implant depth on the threshold voltage, there is the need to control the channel thickness within ± 20 Å, and to ensure that change in impurity concentration is < ± 20% to achieve reasonable device yield across the wafer.

12.4.3 Transconductance and output conductance

The two parameters, transconductance g_m and output conductance g_o, are important since they are directly related to the gain of the MESFET. The transconductance describes the relationship between the output current I_{ds} and the input control voltage V_{gs} and is used to measure the gain of the MESFET, while the output conductance determines the slope of the output characteristics.

12.4.3.1 The transconductance parameter g_m

The transconductance g_m is derived by differentiating equation 12.12 with respect to V_{gs}, giving the principal result:

$$\left.\frac{\Delta I_{ds}}{\Delta V_{gs}}\right|_{V_{ds}} = \text{constant}$$

$$g_m = \begin{cases} = 0;\ \text{for cut off} \\ \text{or} \\ = 2\beta(V_{gs} - V_t)(1 - \lambda V_{ds})\tanh(aV_{ds});\ \text{linear and saturation} \end{cases} \tag{12.18}$$

A major difference between GaAs and Si devices to be noted is the transconductance. For GaAs the transconductance is high with very low gate capacitance. Thus a high gain, bandwidth product can be expcted.

Figure 12–23 shows typical transconductances for several types of devices, including those for silicon, primarily for comparison purposes.

It is interesting to compare the transconductance of the silicon bipolar transistor with that of a MESFET. The transconductance of the silicon bipolar transistor is given by:

$$g_m = I_c\left(\frac{q}{kT}\right)$$

where I_c is the collector current.

The expression can be rewritten as

$$g_m \propto A_E e^{V_{be}(q/kT)}$$

Here it is significant that the transconductance is independent of process and it is only slightly influenced by the transistor size. This can be contrasted with GaAs, where the transconductance is both process-dependent and size-dependent.

12.4.3.1.1 Figure of merit f_t

An indication of frequency response may be obtained from the parameter f_t. Thus, it becomes possible to predict the expected intrinsic speed of a GaAs device from a knowledge of the figure of merit commonly referred to as the gain bandwidth product f_t given by:

$$\begin{aligned} f_t &= \frac{g_m}{2\pi(C_{gs} + C_{gd})} \\ &= \frac{\mu_n}{2\pi L^2}(V_{gs} - V_t) \end{aligned} \tag{12.19}$$

where C_{gs} and C_{gd} are the gate-to-source and gate-to-drain capacitances respectively.

The current gain bandwidth product f_t illustrates that switching speed depends upon

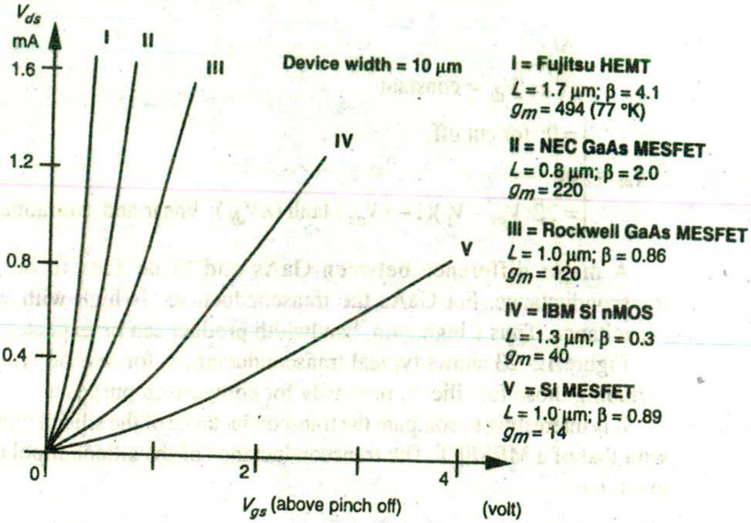


Figure 12–23 Transconductance variations for several devices

- gate length L;
- carrier mobility μ_n in the channel;
- gate voltage.

However, if we consider the limiting condition, that is, velocity saturation, the gain bandwith product may be expressed as

$$f_t = \frac{V_{sat}}{2\pi L} \tag{12.20}$$

where v_{sat} is the saturation velocity. This means for a typical 0.8 μm gate, a f_t of about 28 GHz can be expected.

12.4.3.2 Output conductance

The output conductance can also be determined by differentiating equation 12.12 with respect to the drain voltage V_{ds}. Thus

$$g_o = \begin{cases} 0;\ \text{for cut off} \\ \lambda\beta(V_{gs} - V_t)^2 \tanh(aV_{ds}) + a\beta[(V_{gs} - V_t)^2 (1 + \lambda V_{ds})] \operatorname{sech}^2(aV_{ds});\ \text{linear} \\ \text{and saturation} \end{cases} \tag{12.21}$$

In the saturation region the above relation can be simplified to

$$g_o = \lambda\beta(V_{gs} - V_t)^2 \tanh\ (aV_{ds}) \tag{12.22}$$

from which the drain-to-source resistance R_{ds} can be estimated. Thus

$$R_{ds} = \frac{V_{ds}}{\lambda\beta(V_{gs} - V_t)^2} \tag{12.23}$$

A typical characteristic illustrating the variation of R_{ds} as a function of V_{ds} is shown in Figure 12–24.

12.4.4 Logic voltage swing

In order to improve the switching speed, the options are:

- increase logic voltage swing (logic voltage swing is comparable with the gate voltage above threshold); and
- reduce gate length.

Although the former option is possible, the switching energy in this case is increased, resulting in an increase in dissipation. The dynamic dissipation P_g can be expressed in terms of the logic voltage swing ΔV_o.

Thus

$$P_g = \frac{1}{2}C_L(\Delta V_o)^2 f \tag{12.24}$$

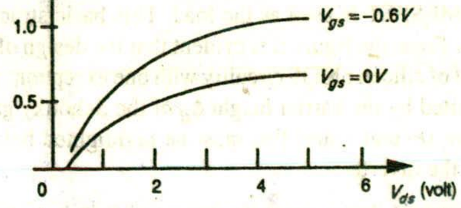


Figure 12–24 Variation of drain-to-source resistance R_{ds} as a function of V_{ds}

where

$$P_g = \text{dynamic dissipation}$$
$$C_L = \text{load capacitance}$$
$$\Delta V_o = \text{logic voltage swing}$$
$$f = \text{frequency of switching}$$

Devices must develop their transconductance at control voltages only a small logic swing above the threshold voltage in order to exhibit small dynamic switching energy.

To establish the logic voltage swing ΔV_o two conditions must be satisfied:

1. The low logic voltage level V_{low} must satisfy

   $$V_{low} < V_t$$

   which ensures that the device turns off.

2. The gate should not be driven higher than the barrier potential, ϕ_B. The logic high level V_{high} therefore should satisfy

   $$V_{high} < \phi_B$$

   Thus, the logic voltage swing can be expressed

   $$\begin{aligned} \Delta V_o &= V_{high} - V_{low} \\ &= \phi_B - V_t \end{aligned} \qquad (12.25)$$

   which is simply the channel pinch-off voltage V_{po}.

12.4.5 Direct-coupled FET logic (DCFL) inverter

A basic requirement for creating a complete range of logic circuits is the inverter. Although several options for the classes of logic are available, direct-coupled FET logic (DCFL) is chosen as this is the only class of logic that shows promise for VLSI implementation.

In this class of logic the inverter uses both the depletion mode and the enhancement mode transistors. The E–MESFET is used as the switching device, while the D–MESFET is used as the load. This basic structure is illustrated in Figure 12–25. From the figure it is evident that the design of the inverter closely resembles that of silicon nMOS circuitry with one exception: the allowable output voltage is limited by the barrier height ϕ_B of the Schottky gate diode.

Now, there several issues that must be highlighted before proceeding with the design of the inverter:

- With no current drawn from the output, the drain to source current I_{ds} for both the E type and D type devices are equal.

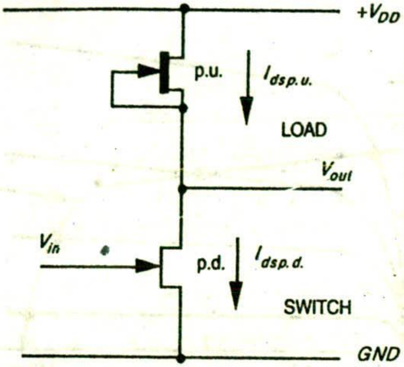


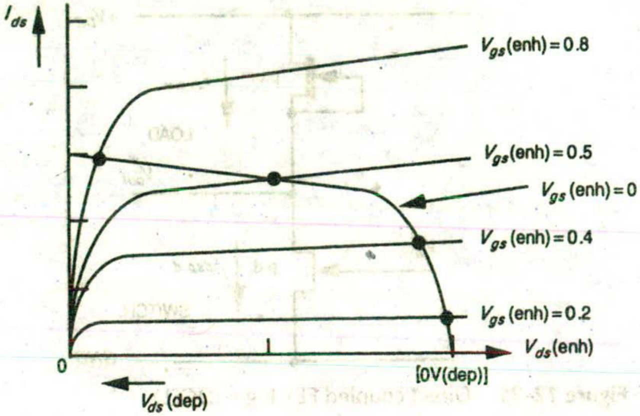
Figure 12–25 Direct coupled FET logic (DCFL)

- For the depletion mode transistor, the gate is connected to the source so it is always on and only the characteristic curve $V_{gs} = 0$ (Figure 12–21(a)) is relevant.
- In this configuration the depletion mode device is called the pull-up (p.u.) and the enhancement mode device is called the pull-down (p.d.) transistor.
- To obtain the inverter tı ansfer characteristic, we superimpose the $V_{gs} = 0$ depletion mode characteristic curve on the family of curves for the enhancement mode device, noting that maximum voltage across the enhancement mode device corresponds to minimum voltage across the depletion mode transistor.
- The point of intersection of the curves as in Figure 12–26 gives points on the transfer characteristic, which is of the form shown in Figure 12–27.
- Note that as V_{in} (= V_{gs} p.d. transistor) exceeds the p.d. threshold voltage, current begins to flow. The output voltage V_{out} thus decreases, and the subsequent increases in V_{in} will cause the p.d. transistor to come out of saturation and become resistive. Note that the p.u. transistor is initially resistive as the p.d. turns on.
- The point at which $V_{out} = V_{in}$ is denoted as V_{inv} (inverter threshold voltage). Note that the transfer characteristics and V_{inv} can be shifted by variation of the ratio of pull-up to pull-down resistances (denoted $Z_{p.u.} / Z_{p.d.}$ where Z is determined by the length to width ratio of the MESFETs).
- During transition, the slope of the transfer characteristic determines the gain

$$Gain = \frac{\Delta V_{out}}{\Delta V_{in}}$$

12.4.5.1 Determination of pull-up to pull-down ratio

Consider the arrangement as shown in Figure 12–28 in which an inverter is driven from the output of another similar inverter.



Intersection points give transfer characteristic

Figure 12–26 Derivation of DCFL inverter transfer characteristics

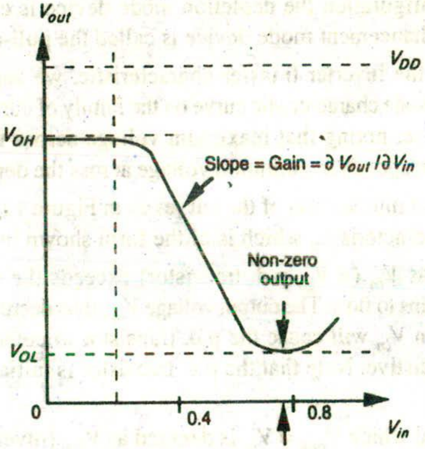


Figure 12–27 DCFL inverter transfer characteristics

In order to cascade inverters without degradation of levels we are aiming to meet the requirement:

$$V_{in} = V_{out} = V_{inv}$$

Since the logic high level is limited by the barrier potential ϕ_B, then for equal margins around the inverter threshold we set V_{inv} equal to half the logic voltage swing.

Thus:

$$V_{inv} = (\phi_B - V_t)/2 = 300 \text{ mV}$$

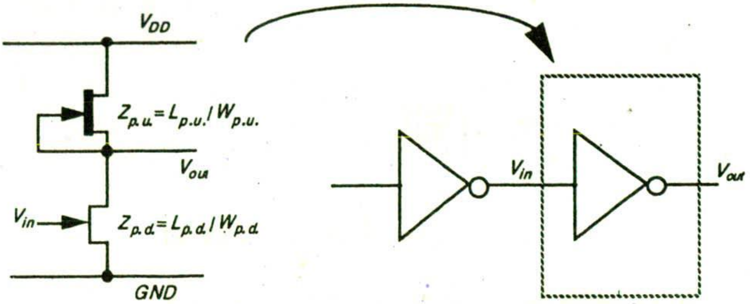


Figure 12–28 DCFL inverter driven directly from another inverter

Now assuming a supply voltage V_{DD} = +2.0 V, and with typical values for threshold voltages V_{tdep} = –700 mV, V_{tenh} = +200 mV, both the pull-up and pull-down transistors are in saturation, that is, $V_{ds} > (V_{gs} - V_t)$ for the D type and E type MESFETs. The pull-up to pull-down ratio ($Z_{p.u.} / Z_{p.d.}$) is defined as:

$$Z_{p.u.} = \frac{L_{p.u.}}{W_{p.u.}}$$

$$Z_{p.d.} = \frac{L_{p.d.}}{W_{p.d.}}$$

where $W_{p.u.}$, $L_{p.u.}$, $W_{p.d.}$ and $L_{p.d.}$ are the widths and lengths of the pull-up and pull-down transistors (i.e. the D–MESFET and E–MESFET) respectively.

The drain to source current for the pull-up transistor (D–MESFET) can be expressed by

$$I_{dsp.u.} = \beta_{p.u.}(V_{gsp.u.} - V_{tdep})^2 \tag{12.26}$$

where

$$\beta_{p.u.} = \left[\frac{\mu_n \varepsilon_0 \varepsilon_r}{2a_{p.u.}}\right]\left[\frac{W_{p.u.}}{L_{p.u.}}\right] \tag{12.27}$$

and

$$a_{p.u.} = \text{implant depth for D–MESFET.}$$

For the pull-down device (E–MESFET), the drain current is

$$I_{dsp.d.} = \beta_{p.d.}(V_{gsp.d.} - V_{tenh})^2 \tag{12.28}$$

where

$$\beta_{p.u.=}\left[\frac{\mu_n\varepsilon_0\varepsilon_r}{2a_{p.u.}}\right]\left[\frac{W_{p.u.}}{L_{p.u.}}\right] \tag{12.29}$$

and

$$a_{p.d.} = \text{implant depth for E–MESFET}$$

Now equating the two currents, and with $V_{gsp.u.} = 0$, $V_{gsp.d.} = V_{in} = V_{inv}$, we have:

$$\frac{1}{Z_{p.d.}}\left(V_{inv} - V_{tp.d.}\right)^2 = \left(\frac{a_{p.d.}}{a_{p.u.}}\right)\left(\frac{1}{Z_{p.u.}}\right)(-V_{tp.u.})^2 \tag{12.30}$$

and on rearrangement

$$\frac{Z_{p.u.}}{Z_{p.d.}} = \left(\frac{a_{p.d.}}{a_{p.u.}}\right)\left(\frac{-V_{tdep\,..}}{V_{inv} - V_{tp.d.}}\right)^2$$

whence

$$V_{inv} = V_{tenh} - \sqrt{\frac{a_{p.d.}}{a_{p.u.}}}\left(\frac{V_{tdep}}{\sqrt{Z_{p.u.}/Z_{p.d.}}}\right)$$

V_{inv} is set approximately midway between ϕ_B and ground.

Substituting typical values for the threshold voltages $V_{tdep} = -700$ mV, $V_{tenh} = +200$ mV, and with $a_{p.u.}/a_{p.d.} = 4{:}1$ and $\phi_B = 800$ mV, we obtain the principal result

$$\frac{Z_{p.u.}}{Z_{p.d.}} = \frac{10}{1}$$

For MESFETs having $L_{p.u.} = L_{p.d.}$, we have

$$\frac{W_{p.u.}}{W_{p.d.}} = \frac{1}{10}$$

However, in order to improve the packing density, as in the case for VLSI applications, it becomes necessary to use a larger gate length for the pull-up device. This will reduce the drain to source saturation current $I_{ds(sat)}$ but with appropriate optimization this may not be very significant.

It should be noted that such an approach provides us with an approximate method to size up a typical DCFL inverter and therefore it becomes essential to resort to simulation tools such as HSPICE in order to optimize a circuit.

12.5 MESFET-based design

12.5.1 MESFET design methodology

The major aim that a circuit designer is faced with is to turn circuit specifications into masks for processing. However, the physical characteristic of the gallium arsenide processing brings about statistical variations in all process parameters, including those of line width, junction depth and film thickness. The objective of this section is to develop an approach to capture the topology of the actual layout so that through a simple representation both layer information and topology can be described and at the same time interaction between signal and power buses is minimized to guard against degradation of noise margin.

12.5.2 Gallium arsenide layer representations

The advances that are taking place in the gallium arsenide process are very complex and sometimes inhibit the visualization of all the mask levels that are used in the actual fabrication process. Nevertheless, the design process can be abstracted to a manageable number of conceptual levels that represent the physical features one observes in the final GaAs wafer.

We have already seen that MESFET circuits are formed effectively on two layers:

1. green implant layer; and
2. red gate-metal layer.

If the gate-metal layer is in contact with the implant layer a transistor is formed, that is, the implant layer and the gate-metal layer interact to form the Schottky gate where they cross one another. However, if an insulating layer is introduced between the implant and the gate-metal, then there is no interaction between these layers and in this case the gate-metal can be used as an interconnect.

We have also seen that the basic MESFET properties can be modified by varying the implant concentration density. Therefore using a simple color scheme we can capture the topology of the actual layout in gallium arsenide so that simple circuit diagrams which convey both layer information and topology for different layers, including those for the E-type and D-type MESFETs, can be set out.

Through color encoding and symbolic representation of layers it is possible to remove much of the complexity associated with a given design. To convey layer information the encoding used to represent a basic transistor is:

- green (*implant*) for the active implant regions; and
- red (*gate-metal*) for Schottky gate and short interconnections.

Now to facilitate changes to characteristics of the basic transistor and to include representation of other layers, the above encoding is complemented by:

- yellow (*nplus*) for the more heavily doped shallow *n* channel implant;
- blue (*metal 1*) for first level metal; and
- dark blue (*metal 2*) for second level metal.

Transistors are formed by intersection of the green and red masks. The devices that are formed can either be enhancement mode, if no yellow implantation is provided, or depletion mode, if such an implantation is provided. Therefore, the E-type MESFETs are formed whenever the two masks red and green intersect; the D-type MESFETs are formed by intersection of green, red, and yellow masks.

It is essential that one fully understand what set of masks a particular process line uses if an interface format is to be generated. At mask level, some layers can be omitted for clarity while others are derived. The layers for a typical gallium arsenide E–/D–MESFET process are represented in Table 12–4. The following comments should assist with clarifying the color encoding used in Table 12–4.

The *green* layer mask identifies all the active regions, that is, areas that eventually form D and E type devices, active loads, Schottky diodes, and implant resistors.

Green regions that are inside the *yellow* layer mask form the more heavily doped channel of the D–MESFET.

Green regions outside the *yellow* form the lightly doped channel of the E–MESFET.

Table 12–4 Layer representation for E/D GaAs process

Layer	*Color*	*Symbolic*	*CIF*	*Comments*
Implant	Green	E–MESFET	GD	Inside is the active area, outside is the substrate. E–MESFET is formed when crossed by gate-metal.
Depletion implant n^+	Yellow	D–MESFET	GI	Defines the more heavily doped depletion MESFET.
Ohmic contact	Brown	—	GH	Used with source/drain contacts.
Gate-metal	Red	Gate-metal	GP	—
Metal 1	Blue	Metal 1	GM	—
Metal 2	Dark blue	Metal 2	GN	—
Contact	Black	Contact	GC	Source/drain and gate contacts to metal 1.
Via	Gray	Via	GV	Metal 1 to metal 2 contacts.
Passivation	White stipples	—	GG	—

12.5.3 Design methodology and layout style

Having introduced the color and encoding convention for layer representation and device formations, we are now in a position to illustrate the approach to be used to turn MESFET circuits into a mask layout.

12.5.3.1 Ring notation for GaAs MESFETs

Communication paths between cells or group of cells and organization and positioning of power (V_{DD}) and ground (*GND*) buses have significant influence upon the performance of very high and ultra high speed VLSI systems. For example, fast transitions on a signal bus could bring about significant noise on the 'Power Bus'. Thus, both the design methodology and layout will have to address the influence of coupling between buses on performance. This leads to the concept of *'ring notation'* or *'ring diagrams'*, a generic term given to a free form topological symbolic layout in which graphical symbols are placed relative to each other rather than in an absolute manner. These are subsequently interconnected by colored sticks representing mask level interconnection layers, paying particular attention to organizational aspects of 'Power 'and 'Ground' buses in relation to high speed signal carrying paths.

In this text the color coding has been complemented by monochrome encoding of the lines so that black and white copies of circuit representation using *'rings'* do not lose the layer information. The encoding is shown in Figure 12–29.

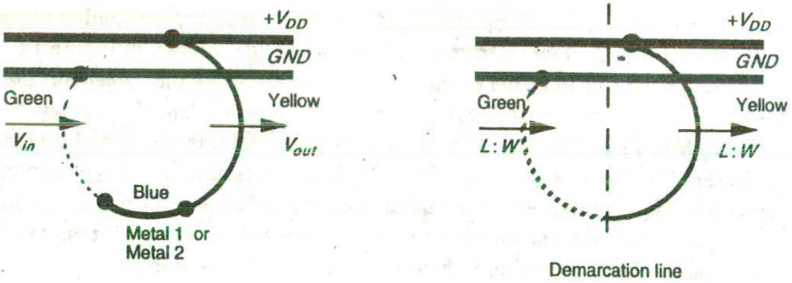
In the *ring diagram* as shown in Figure 12–30(a), the *'green'* or *'dotted'* line represents the E–MESFET while the *'yellow'* or *'solid'* line represents the D–MESFET. The two 'E type' and 'D type' features are joined together using *'blue'* metal 1. Since this rule is implicit, for simplicity of representation it is possible to remove both the metal and the cut representation at this level of abstraction and include a demarcation line as a reminder, which can be left out after gaining some layout experience. It should be noted that the missing geometries will appear when the *ring diagram* is translated into either symbolic or mask layout form. This simplification is shown in Figure 12–30(b). At this level of abstraction it is important that the length (L) and width (W) for each transistor be included.

12.5.3.2 MESFET design style

Having conveyed layer information and topology by using *ring diagrams*, the rings can then be turned into mask layout either directly or through an intermediate 'symbolic' representation stage of 'grid assignment' where *rings* are converted into circuit elements. This translation phase is illustrated in Figure 12–31. For the mask layouts produced during design to be compatible with the fabrication processes, a set of generic design rules are set out for layouts so that, if obeyed, the rules will produce layouts which will work in practice. Therefore, with the aid of *ring notation* the designer is able to layout the skeleton of a circuit quickly,

Layer	Color	CIF	MONOCHROME ENCODING
Diffusion/ implant	Green	GD	▨
Gate-metal	Red	GP	▩
n+	Yellow	GI	⬚
Ohmic contact	Brown	GH	▦
Metal 1	Blue	GM	▥
Metal 2	Dark blue	GN	▤
Contact	Black	GC	■
Via 1	Gray	GV	⊠
Overglass/ passivation	White stipples	GG	▭

Figure 12–29 Layer/feature encoding schemes



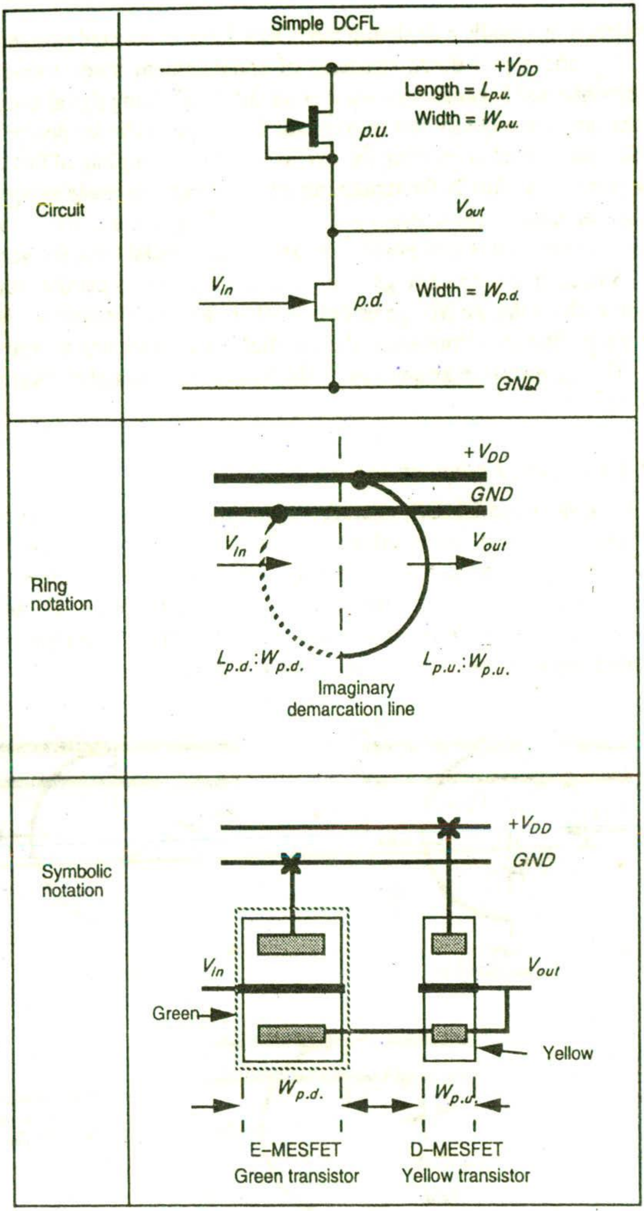
(a) Ring notation for E/D MESFETs

(b) Simplified representation

Figure 12–30 Ring diagram showing the topology of a circuit

paying particular attention to interconnects between adjacent circuitry as well as to the positioning of signal buses in relation to both the Power ($+V_{DD}$) and Ground (GND) buses.

When starting a layout, the first step is normally to draw the Metal 2 (*dark blue*) $+V_{DD}$ and GND rails in parallel and in the close proximity of one another at the top. Next, the *green* followed by *yellow* paths are drawn for inverters and inverter-based logic (such as *Nor* gates), as shown in Figure 12–31, not forgetting to make appropriate contacts. Inverters and inverter-based logic comprise a pull-



Note: Pull-down always uses minimum size gate length.

Figure 12–31 Translation of DCFL inverter circuit to ring and symbolic forms

up structure, usually a depletion mode transistor, connected from the output point to V_{DD} and a pull-down structure of enhancement mode transistors suitably interconnected between the output point and *GND*. Long signal and global control paths are conveniently run in in metal 2, parallel with the power rails with the GND bus located in between the two to reduce the coupling of fast transients into the power bus. Finally the remaining interconnects are made using either metal 1 (*blue*) or metal 2 (*dark blue*) and the control signals and data inputs added. In some processes it is also possible to use the gate metal (*red*) for very short paths.

Since, in this technology, we restrict ourselves to parallel branches in the input path — that is, *Nor* gates only — then the ring notation for *Nor* gates may be simplified by eliminating the parallel input branches as shown in Figure 12–32. The transformation to symbolic form is then straightforward, as in Figure 12–33.

12.5.3.2 Layer connections

As for nMOS and CMOS, intersections on the same layer form connections, as in Figure 12–34(a). Intersections on different layers do not form connections or transistors as shown in Figure 12–34(b). Different layers may also be connected by a contact or a via as in Figure 12–34(c). Some processes do not support *blue* (metal 1) crossing *green* (diffusion). This is primarily to reduce some of the complexities that emerge during the design phase.

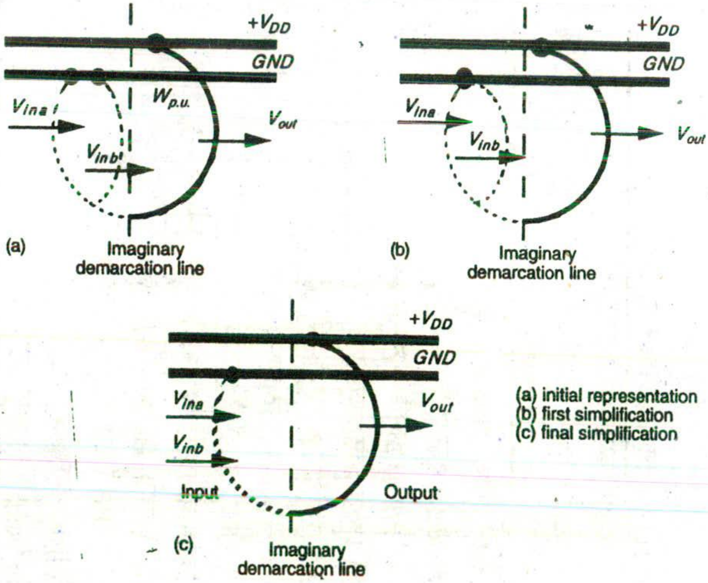
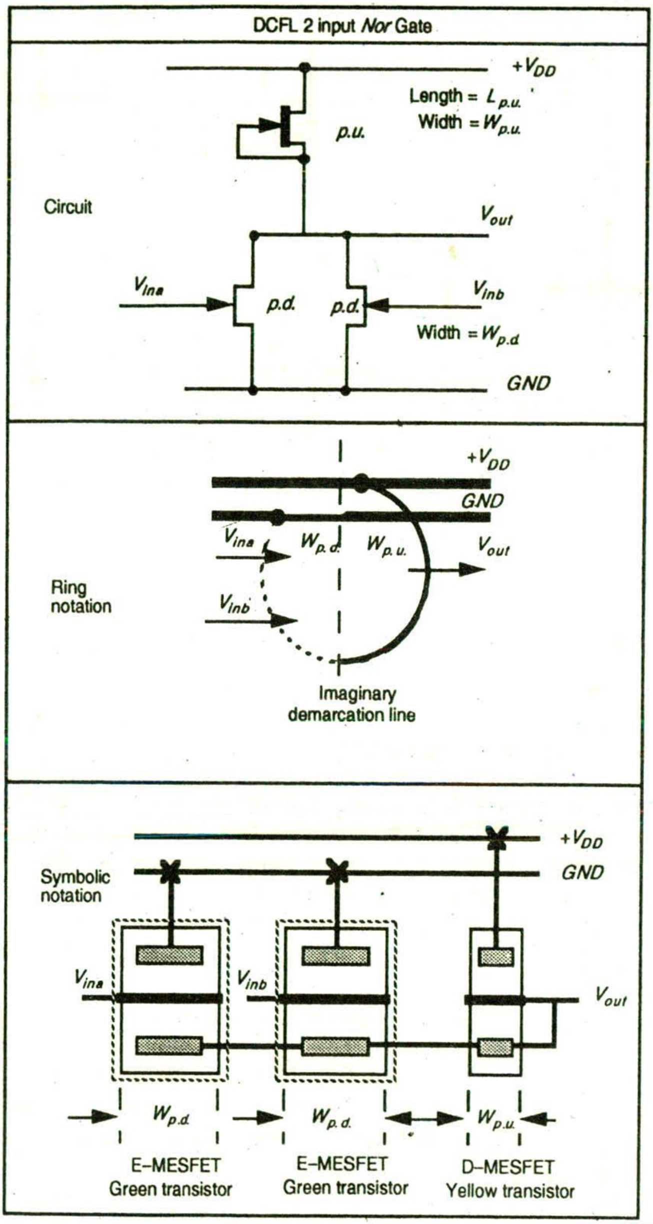


Figure 12–32 Ring notation for 2 input *Nor* gate



Note: Pull-down MESFETS are always of minimum size gate length.

Figure 12–33 Basic structure and symbolic layout for 2 input *Nor* gate

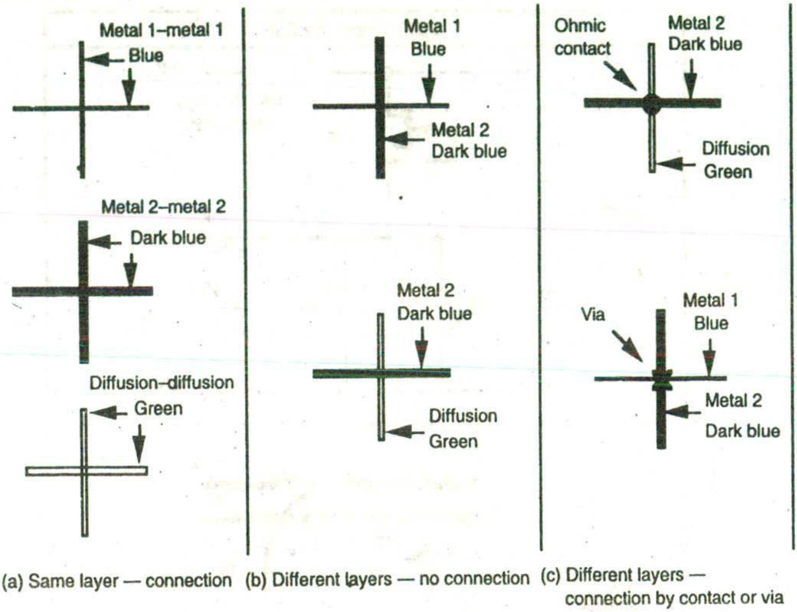


Figure 12–34 Layer connectivity*

* Some restrictions may apply for specific processes.

12.5.4 Layout design rules

Design rules, or *layout rules,* can be considered as a prescription for the preparation of the photomasks that are to be used in the fabrication of integrated circuits. The rule set provides a necessary communication link between circuit designer and process engineer during the manufacturing phase of the integrated circuit. The main objective associated with the design rules is to obtain the circuit with optimum yield in as small a geometry as possible without compromising reliability of the circuit.

Usually, the layout rules represent the best possible compromise between yield and performance. In fact, the more conservative the rules are, the more likely it is that the circuit will function. However, the more aggressive the rules are, the greater the probability of improvements in circuit performance. Such an improvement may be at the expense of yield. Design rules specify to the designer certain geometric constraints on the layout artwork so that the patterns on the processed wafer will preserve the topology and geometry of the design. What is significant is that layout rules do not represent some hard boundary between correct and incorrect fabrication, but a tolerance that ensures very high probability of correct fabrication and subsequent operation.

Circuit designers usually want tighter, smaller layouts for improved performance and decreased area. On the other hand, the process engineer calls for rules that result in a controllable and reproducible process. One important factor associated with design rules is the achievable definition of the process line equipment. Definition is determined by process line equipment and process design. For example, it is found that if a 10:1 wafer stepper is used instead of a 1:1 projection mask aligner, the level-to-level registration will be closer.

Design rules can also be influenced by the maturity of the process line. If the process is mature, then one can be assured of the process line capability allowing tighter design with fewer constraints on the designer. Layout rules address two main issues:

1. geometrical reproduction of features that can be reproduced by the mask-making and lithographical process; and
2. interaction between different layers.

Over the years several approaches have been used to describe the design rules. However, in this text we are going to concentrate on two methods that are appropriate for gallium arsenide technology. These are:

1. The *lambda-based* rule; and
2. The *micron-based* design rule.

The lambda-based design rules used earlier in the text were made popular by Mead and Conway (1980)* for silicon, and are based on a single parameter, *lambda* (λ), which characterizes the linear features as well as the resolution of the complete wafer implementation process.

Note that the degradation in circuit performance could make the lambda-based design approach unsuitable for GaAs processes. However, in this text, for simplicity, initially we will use lambda rules to illustrate principles and to familiarize the designers with the geometric features and the layout process associated with GaAs MESFETs. Then, by adopting symbolic techniques, micron rules can be applied directly.

12.5.4.1 Lambda-based rules for GaAs MESFET

Table 12–5 and Figure 12–35 are a version of a lambda-based rule set. From Figure 12–35 it can be seen that the rule set is defined in terms of:

- feature sizes; and
- separations and overlaps.

Several rule set issues require discussion.

* C. A. Mead & L. A. Conway, *Introduction to VLSI Systems*, Addison-Wesley, 1980.

Table 12–5 Lambda-based layout rules for gallium arsenide

Layer	*CIF*	*Rule feature*	*Dimension (lambda)*
Active (Diffusion)	GD	A1 minimum width	5
		A2 minimum spacing	5
		A3 minimum to n^+	5
		A4 minimum E–MESFET width	5
Depletion implant n^+	GI	B1 minimum D–MESFET gate overlap	2
		B2 minimum width	7
		B3 minimum spacing	5
		B4 minimum spacing to E–MESFET	2
Ohmic contact	GH	C1 minimum ohmic contact width	5
		C2 minimum ohmic-metal spacing	5
		C3 minimum cut overlap	2
		C4 minimum ohmic contact size	5 × 5
Gate metal	GP	D1 min. gate-metal gate extension	2
		D2 min. gate-metal length	3
		D3 min. gate-metal width	3
		D4 minimum cut overlap	2
		D5 min. gate-metal spacing	5
		D6 min. spacing to ohmic contact	3
Contact	GC	E1 minimum cut size	4 × 4
		E2 minimum cut spacing	4
		E3 minimum spacing to via	4
Metal 1 (Diffusion)	GM	F1 minimum width	4
		F2 minimum spacing	5
		F3 minimum cut overlap	2
		F4 minimum via 1 overlap	2
Via 1	GV	G1 minimum via size	5 × 5
		G2 minimum via spacing	5
Metal 2	GN	H1 minimum width	5
		H2 minimum spacing	5
		H3 minimum overlap of via 1	2

12.5.4.2 Width and spacing rules

Although diffusion, metal 1, and metal 2 can cross each other without interaction, in some processes metal 1 is not permitted to cross diffusion.

Width and separation rules given in Figure 12–35 are dependent upon the width of the photoresist.

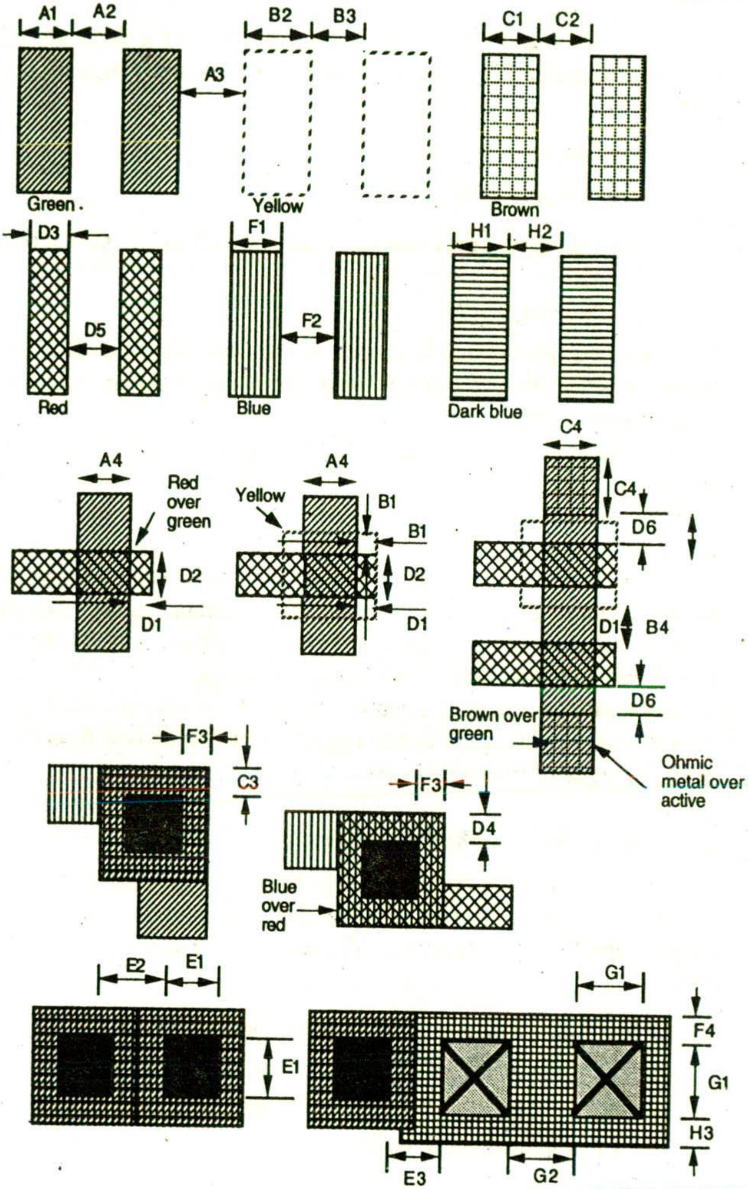


Figure 12–35 Lambda-based rules for GaAs MESFET process

As for nMOS and CMOS, we need to ensure that the depletion regions of two unrelated implants do not contact. The separation between implant is determined from:

(a) width of depletion region; and

(b) width of the photoresist.

Crossing of metal 2 over channel areas of the MESFETs should be avoided.

12.5.4.3 Transistor rules

There are two types of implants used to form the two different MESFETs. A transistor is depletion type if it is inside the n+ (*yellow*) region, otherwise it is enhancement mode.

It is essential for gate-metal (*red*) to completely cross the implant (*green*) region, otherwise the transistor that has been created will be shorted by a n⁻ path between source and drain. To ensure this condition is satisfied, 2λ of gate-metal extension is necessary. This is termed the 'Schottky gate extension'.

Orientation is an important consideration during layout. All MESFETs need to be positioned horizontally owing to the anisotropic nature of GaAs, which influences the threshold voltage of the device brought about as a result of variation in both concentration density and channel thickness.

Some processes require isolation between devices to reduce their interaction. This is achieved through lattice damage. The mask is derived from the 'logical' operation of the active layer masks.

12.5.4.4 Contact cut and via rules

Generally the size of a cut is established from the knowledge of the minimum dimensions necessary to give an acceptable resistance. The ohmic contact has a current capability in the range 0.5–1.0 mA/µm long. The rules that one may follow are:

- minimum dimensions of ohmic cut for source/drain are 5λ × 5λ;
- minimum dimensions of a cut are 4λ × 4λ;
- via dimension is 5λ × 5λ;
- metal 1 overlap of via is 2λ; and
- metal 2 overlap of via is 2λ.

12.5.4.5 Process enhancements

There are several enhancements that may be added to the GaAs processes, primarily to provide active load, capacitors and resistors, as well as to increase routability of circuits through a third metal or fourth metal level.

12.5.4.5.1 *Saturated resistor*

The saturated resistor is simply a MESFET with the Schottky gate removed. The preferred direction for layout is vertical.

12.5.4.5.2 *Capacitors*

Several of the processes provide for at least two kinds of capacitors. These are:

1. metal-insulator-metal (MIM); and
2. diode-capacitor diode (DCAP).

There is considerable complexity and variation in the approach to realize the DCAP, but the MIM capacitor structure is quite simple using metal 1 and metal 2 as the plates of a parallel plate capacitor.

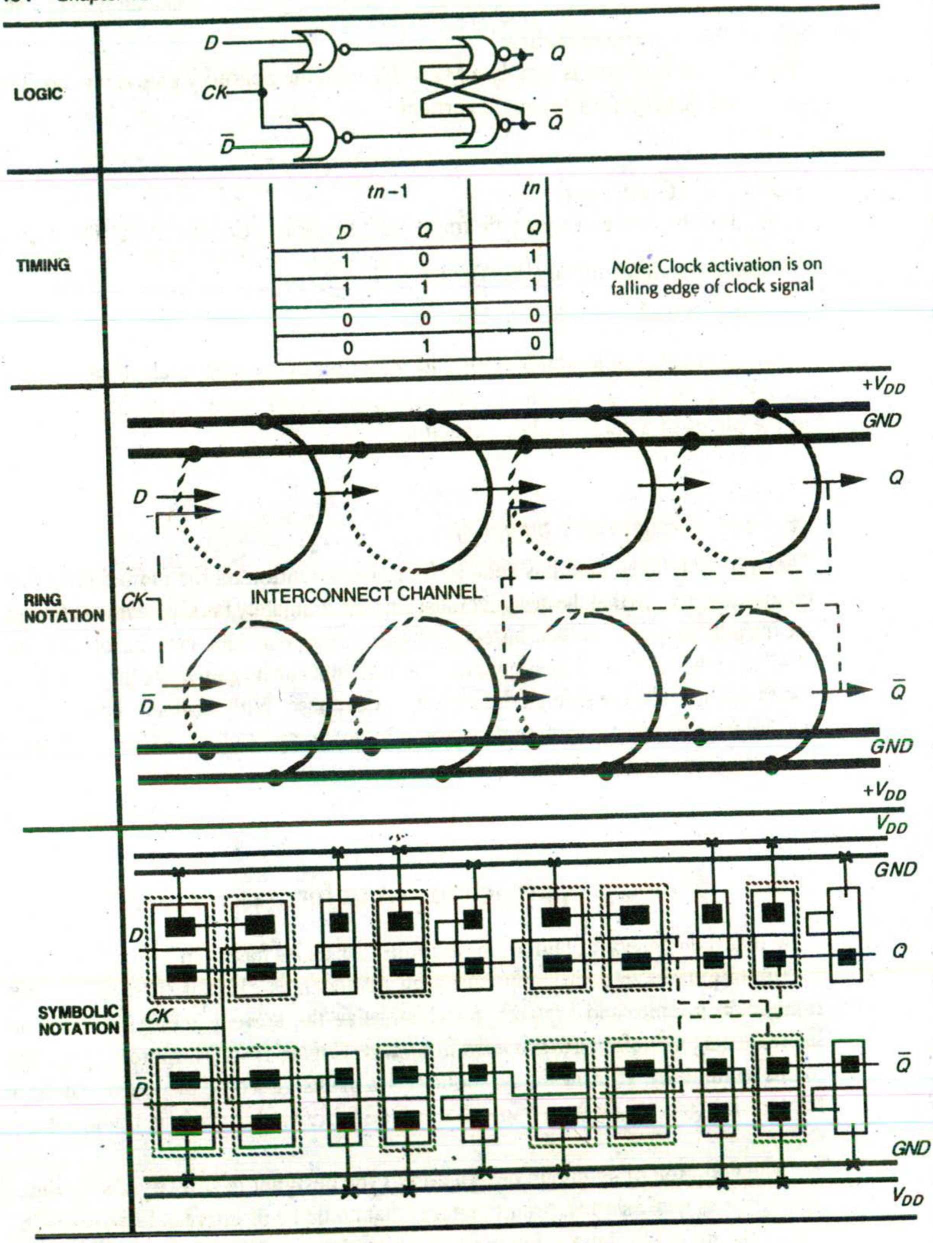
12.5.4.6 *Design rules summary*

The approach taken here has been to focus our attention on the main features of typical design rules that the designer must become intimately familiar with. Although the introduction of a lambda-based approach has no real value in terms of creating 'real' circuits, because of its simplicity, much insight can be gained for the important issues that must be considered during the layout phase. With this basic knowledge, it is not too difficult to use actual micron rules that may be obtained from different foundries for an actual design.

12.5.5 Symbolic approach to layout for GaAs MESFETs

Now that the concept of lambda-based rules for GaAs has been introduced, and its limitations have been commented on, it becomes evident that perhaps to implement circuits and systems that synthesize the correct geometry from an intermediate form referred to as *symbolic notation* (Figure 12–31) is more appropriate to this technology. This means symbolic styles of design would provide a solution for creating generic GaAs circuits that can be fabricated in various foundries or processes.

The adoption of symbolic design allows the designer to directly manipulate transistors as well as other circuit features that could be of interest. Figure 12–36 shows the symbolic layout for the D type latch using SDCFL logic (see next section) and based upon the notation of Figures 12–30 and 12–31. Translation from this level into masks once again requires the introduction of geometric detail as before, using the micron rules.



$tn-1$		tn
D	Q	Q
1	0	1
1	1	1
0	0	0
0	1	0

Note: Clock activation is on falling edge of clock signal

Figure 12–36 Symbolic representation of D type latch using SDCFL

12.6 GaAs MESFET classes of logic

There are two main approaches to logic design:

1. normally-on logic; and
2. normally-off logic.

The normally-on logic uses depletion mode MESFETs which are 'ON' devices and when used as switching elements are required to be turned OFF. Thus, a number of circuit techniques have been developed to facilitate logic turn-off. The approaches in this class of logic include:

- unbuffered FET logic (UFL);
- buffered FET logic (BFL);
- D–MESFET Schottky diode FET logic (SDFL);
- capacitor-coupled FET logic (CCFL); and
- capacitor-diode FET logic (CDFL).

The normally-off logic uses enhancement mode MESFETs as the switching element. Although several approaches have emerged during the last few years, the following structures have shown the most promise:

- direct-coupled FET logic (DCFL);
- buffered DCFL; and
- source-follower DCFL (SDCFL).

12.6.1 Normally-on logic gates

Depletion mode devices are basic switching elements for this class of circuits. Since DMESFETs are ON devices, a negative voltage is needed at the gate to facilitate turn-off. This means that two supply rails together with level shifting networks are necessary for proper circuit operation. Owing to this complexity, this class of logic is unsuitable for VLSI implementation.

12.6.2 Normally-off logic gates

The normally-off logic includes direct coupled FET logic (DCFL), buffered DCFL, and source-follower (SDCFL). The following section provides a brief outline of this particular class of logic families, with particular emphasis on the DCFL and SDCFL being the main contenders for ultra high speed VLSI systems.

12.6.2.1 Direct-coupled FET logic (DCFL)

In this class of logic both the depletion mode and the enhancement mode transistors are used. The enhancement mode FET acts as the switching device, and the depletion type device acts as the load. From the basic structure it is evident that, first, there is no need for level shifting circuitry as was the case for normally-on logic. Secondly, the design of the logic gate closely resembles that of nMOS circuitry; and finally only a single power supply is required. DCFL gate dissipation is typically 100 μW with an associated delay of about 50 ps, which is considerably less than the normally-on logic families. Thus, the logic appears as a suitable contender for very high speed VLSI systems.

The allowable output voltage is limited by the barrier height of the Schottky gate diode, which means only a small voltage swing is possible from DCFL circuits, which in turn implies relatively small noise margins.

12.6.2.2 DCFL with super buffers

DCFL circuits have weak load drive capability. This implies that the delay associated with a gate increases with an increase in both the fan-out and the interconnect line lengths. Introduction of super buffers can alleviate much of the problem at the expense of extra area. Usually the basic DCFL gate is used for light load conditions, while the super buffers are used where larger loads are to be driven.

12.6.2.3 Source-follower DCFL FET logic (SDCFL)

The source-follower FET logic (SFFL) uses both the enhancement and depletion mode devices. The basic structure for a SFFL inverter is shown in Figure 12–37. This logic family has both a power dissipation and switching delay that are comparable with the DCFL family, but with a larger noise margin which is brought

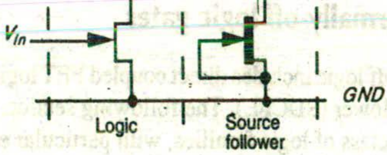


Figure 12–37 Basic inverter structure for source follower DCFL FET logic (SDCFL)

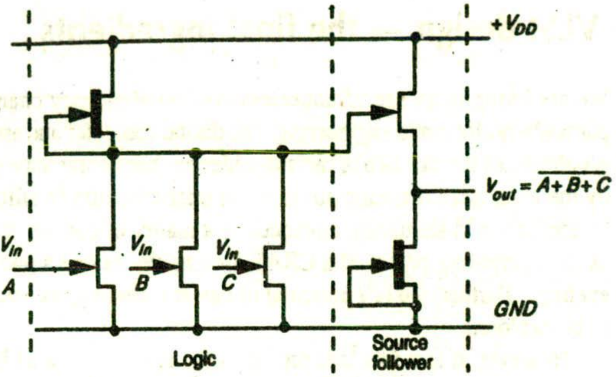


Figure 12–38 Three input SDCFL *Nor* gate

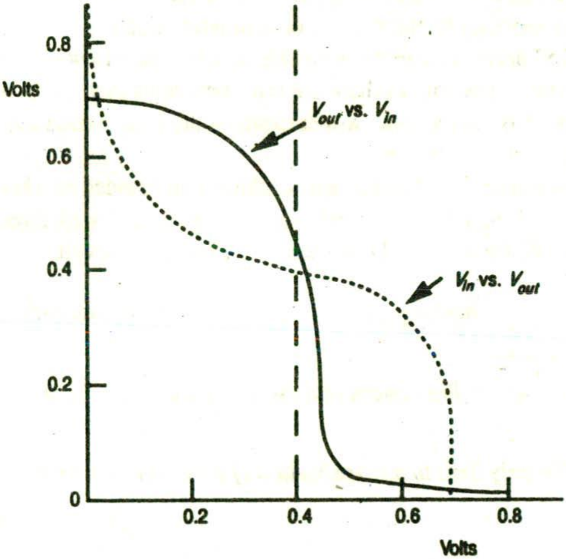


Figure 12–39 DC transfer characteristics for SDCFL

about as the result of the pull-up transistor (enhancement mode) being able to be turned off, thus permitting the source follower output to pull-down all the way toward zero voltage.

This basic structure can be extended to perform logic functions. A typical three input gate is shown in Figure 12–38. The DC transfer characteristics, showing the larger noise margin, are illustrated in Figure 12–39. This class of logic is most suitable for the realization of the *And-Or-Invert* (AOI) function which usually assists in the optimization of logical functions.

12.7 VLSI design — the final ingredients

We are living in an age of unprecedented revolutionary change in engineering, particularly electronic engineering. The digital computer and associated processing revolution of the past two or more decades has been complemented and augmented by the even more dramatic advances in microcircuitry in silicon. We have come to accept world-shattering advances as a matter of course, and predictions such as the computing power of a CRAY 1 computer in one's pocket hardly raise an eyebrow. Further, the full potential of newer technologies, such as GaAs, has yet to be explored.

However, it is a fact that unlike in the situations faced by engineers in the past, we are no longer technology-bound or limited. Indeed, we have solutions to problems that don't even exist yet! This is a situation in which the potential applications of VLSI technology are limited only by the creativity and imagination of those working in engineering or computer science.

VLSI design is also an enjoyable area in which to work. The designer has a great deal of freedom as there are few constraints associated with VLSI system designs. It is also an area which captures the imagination and it is hard not to become highly motivated.

The authors therefore recognize enthusiasm founded on a *knowledgeable base* as the final ingredient. We feel it appropriate to end with three quotations, two from R. W. Emerson and one from Franklin D. Roosevelt:

Nothing great was ever achieved without enthusiasm.

R.W.E.

The reward of a thing well done is to have done it.

R.W.E.

The only limit to our realization of tomorrow will be our doubts of today.

F.D.R.

12.8 Tutorial exercises

1. Using the electron velocity versus electric field characteristics as illustrated in Figure 12–5, compare the carrier velocity behavior of silicon with that of gallium arsenide. For a bipolar device the base region may be considered to be in the order of 0.2 μm–0.25 μm, while in the GaAs technology a typical gate has a dimension of 1.0 μm.
2. Typical values for a D-MESFET are as follows: $\mu_n = 7000\ \text{cm}^2/\text{V-sec}$, $\varepsilon_r = 13.1$, $\varepsilon_0 = 8.85^{*}10^{-14}$ F/cm, $a = 1000$ Å. Using these parameters as the base determine the gain factor for the D–MESFET.

3. For depletion mode devices, typical channel doping is in the order of $1*10^{17}$ cm^{-3} and the channel implant thickness, *a*, is about 1500 Å. Calculate the pinch-off voltage and hence the threshold voltage for this device. What conclusions can you make?
4. Using the *ring notation*, design a simple D-latch. With the aid of color encoding create a layout for this structure.