

Departement of Physics and Measurement Technology

A CMOS DESIGN MANUAL

Christer Svensson and Rolf Sundblad

Third edition, August 1982

LSI Design Center, Linköping



Linköping University Department of Physics and Measurement Technology
S-581 83 Linköping, Sweden

A CMOS DESIGN MANUAL

Christer Svensson and Rolf Sundblad

Third edition, August 1982

Contents:

1. Introduction
2. The CMOS principle
3. Some simple CMOS circuits
4. The metal gate CMOS process and design rules
5. Stick diagrams
6. CMOS logic circuitry
7. Electrical constraints
8. Design methodology
9. Some process parameters
10. The silicon gate CMOS process and design rules.

1. Introduction.

This short design manual is aimed to introduce a student to CMOS technology so that he can design simple MOS circuits in a multiple project chip environment. The manual treat only CMOS technology.

2. The CMOS principle.

CMOS technology is based on two devices, n-channel and p-channel MOS-transistors. The use of this complementary pair of devices gives CMOS digital circuits with some unique properties. They will be highly symmetric (with respect to zero and one), they will have large signal margins (the zeros and ones will be very distinct) and they will use very little power (power is used only for transitions). These properties makes CMOS a very attractive technology for VLSI circuits in the future.

In fig. 1 we show schematic pictures and current voltage characteristics of the two transistor types. The current voltage characteristic for an n-channel transistor can be written:

$$I_D = K_P \cdot (W/L) \cdot \langle u(V_G - V_T) - u(V_G - V_T - V_D) \rangle ,$$

$$u(X) = \langle (X + \text{ABS}(X)) / 2 \rangle^{**2}$$

where K_P is a constant, W/L is the width to length ratio of the transistor, V_G is the gate voltage, V_T is the threshold voltage of the transistor (a constant) and V_D is the drain voltage. The characteristic of the p-channel device is obtained by changing sign of I_D and all voltages. A very simple view is to assume that an n-channel device is open for zero gate-source voltage and shorted for a high gate-source voltage and that a p-channel device also is open for zero gate-source voltage but shorted for a high negative voltage.

The principle of the CMOS digital circuit family is clearly seen in fig. 2. When the input voltage V_{IN} is zero (V_{SS}) the n-channel device is open and the p-device is shorted. This makes the output voltage high, $V_{OUT} = V_{DD}$. Similarly $V_{IN} = V_{DD}$ makes the output voltage low, $V_{OUT} = V_{SS}$. The operation is completely symmetric with respect to V_{SS} and V_{DD} . The output signal is locked to either V_{SS} or V_{DD} . Note also that in neither state is there a current path from V_{DD} to V_{SS} , thus the circuit use no static power. Finally note that the substrate of the n-channel device is connected to V_{SS} and the substrate of the p-channel device to V_{DD} . The two substrates may therefore be used as V_{SS} and V_{DD} supply lines.

How do we find a practical way to fabricate the CMOS circuits? In fig. 3 we show the real structure of a silicon crystal with a CMOS inverter integrated in it. We can note that the p-channel MOS

transistor is made directly in the n-type silicon wafer. The n-channel device is made in a special p-well, formed by diffusing in p-dopant into the wafer in selected areas. The p-well and the n-wafer makes up a pn-diode. However as this diode always is backbiased (as the n-substrate is connected to VDD and the p-well to VSS) it will not affect our circuit at all.

3. Some simple CMOS circuits.

The simplest CMOS circuit is the inverter circuit already shown (fig. 2). This circuit can easily be extended to NOR and NAND gates, see fig. 4. In the case of the NOR gate $V_{OUT}=0$ if any of the inputs are high (shorting the output to VSS) and $V_{OUT}=V_{DD}$ only if both inputs are low (shorting the output to VDD). As in the case of the inverter no current flows in either state. Comparing with the NAND gate again demonstrate the symmetry of CMOS. The NAND gate is obtained if the NOR gate is put upside down or if the logic notation is changed from positive to negative logic.

Another circuit example is the transmission gate (pass gate) shown in fig. 5. The pass gate is used as a switch, controlled by P. Its simplicity and its property to be bidirectional makes it a very useful device.

4. The metal gate CMOS process and design rules.

As different processes have different geometry and as geometry becomes smaller with time it is convenient to relate all measures to a unit size, LAMBDA. LAMBDA can for example be the largest mask error which can occur in the process.

Let us take a closer look upon the fabrication process for metal gate CMOS circuits. In fig. 6 we show the whole fabrication process in steps corresponding to the different masks used. We show only the process for the n-channel transistor. The p-transistor fabrication is completely analogous, one just exchanges n and p diffusions and deletes the p-well and the guard.

The first process is to use an oxide mask to define the p-well area. The right hand part of fig. 6 shows the mask. The p-well is diffused into the silicon wafer through the opening in the oxide. As this diffusion is rather deep one must remember that it also diffuses under the oxide edge as shown in the figure. We will return to this.

Next step is the p-diffusion which is used for source and drain in p-transistors and for guard bands around n-transistors. Guard bands are used in the actual process to avoid so called latch-up (to be discussed in section 7). The p-diffusion is not so deep, but gives anyway some underdiffusion near the oxide edge, d. Assuming d to be less than LAMBDA allows us to put diffusions as close as 4 units of

LAMBDA without risk for shorts. This is thus the first design rule. Furthermore we will not allow the diffusion to be narrower than 4 units. We may also understand the p-well mask now. By always putting the p-well mask edge 3 units inside the outside edge of the p-guard band, its underdiffusion will be masked by the guard. Then we do not need to consider this underdiffusion any further in our design work.

Next step concerns the n-diffusion for drain and source of our n-transistor. Again, we have a small underdiffusion, d . This mask sets the important channel length of the transistor, L . As d normally is of the order of half the LAMBDA unit we may allow the masked value of L to be 3 units. The real L then becomes $3\text{LAMBDA} - 2d$. In some cases it is of interest to contact the transistor source with its guard band or its substrate. This is done by putting the two diffusions together and placing a metal contact on top. Note that the two diffusions can not be contacted directly. Without the metal short we will get a pn-diode.

We will now open holes in the field oxide for thin oxide areas. As shown in fig. 6 openings are formed both over transistor gate areas and over contact cut areas. In the transistor gate area is the field oxide opening overlapping the diffusions by 2 units. It is very important that the gate metal control the channel all the way to the n-diffusion edges. It must also control the whole thin oxide part of the gate area, thus it must overlap the field oxide edge 1 LAMBDA at the transistor edge. The field oxide opening around the contact cut shall overlap the coming contact cut by 1 unit. This is to make the contact opening easier to etch.

The contact cuts are now formed. The important point is to have the opening large enough to give reliable contacts. We have chosen a minimum opening of 4×4 units. Furthermore the opening must be at least 2 units inside a diffusion edge to avoid any shorts to substrate.

The final mask is the metal mask. For the metal we will use the following rules. Minimum width 4 units, minimum separation 3 units, minimum overlap over source and drain diffusions 1 unit and minimum overlap over field oxide edge and at transistor edges 1 unit. Over a contact cut we are satisfied by a zero overlap.

The complete design rules are given in Table 1 and are demonstrated in fig. 7.

In some cases a more complete set of guard bands are used in order to prevent surface leakage currents. This is often used when the CMOS circuit is to be operated at higher voltages, e.g. 15 V. In such a case each transistor and each diffusion is surrounded by a guard band with opposite doping. Also, the gate metal is always arranged so that it controls all area between a transistor gate edge and the nearest guard band. The additional design rules for a completely guarded CMOS process are demonstrated in Table 2 and fig. 8.

Table 1. Design rules for the metal gate CMOS process.

Rule	units of LAMBDA
Diffusions, general	
D1 Minimum width	4
D2 Minimum separation	4
Diffusions, transistor	
D3 Minimum channel length	3
D4 Minimum channel width	8
Diffusions, shorted n to p	
D5 Minimum separation	0
D6 Minimum separation n-diffusion to outer edge of p guard band	6
Contact cuts	
C1 Minimum width	4
C2 Minimum separation to diffusion edge	2
C3 Minimum overlap over diffusion	4
Metal, general	
M1 Minimum width	4
M2 Minimum separation	3
Metal, transistor	
M3 Minimum overlap over source and drain diffusions	1
M4 Minimum extension over transistor edge	2
Metal, contact cuts	
M5 Minimum overlap over contact cut	0
Bonding pads	
M6 Minimum area of pad, VDD or VSS	50x150
others	30x50
M7 Minimum separation between pads	50
M8 Minimum separation between pad and other structure	25
p-well	
W1 The p-well shall be placed 3 units inside the outer edge of the p guard band.	
Field oxide cuts	
F1 Minimum separation	3
F2 The field oxide shall be removed 2 units outside the diffusion edges in transistors, 1 unit outside a transistor edge, 1 unit outside contact cuts and 1 unit outside metal electrodes of thin oxide capacitors.	
Overglass	
G1 Overglass shall cover everything except pads, from 3 units inside the pad metal edge	
Guard bands	
A p-type guard band must surround any p-well and be connected to VSS at appropriate distances.	

TABLE 2, Design rules with full guard bands.

Rule	units of LAMBDA
Metal, transistor, M4 Minimum overlap over guard bands	3
Field oxid cuts, F2 The field oxide shall be removed 2 units outside the diffusion edges (including guard bands) in transistors, 1 unit outside contact cuts and 1 unit outside metal electrodes of thin oxide capacitors.	
Guard bands. The guard band shall be layed out so that each transistor or conductor diffusion is completely surrounded by either a guard band or a transistor channel. All guard bands of the same type shall be connected.	

5. Stick diagrams.

As demonstrated in the previos section it is quite cumbersome to make a detailed layout. Furthermore, not only the design rules must be considered, one must also handle the overall topology. One must for example handle the placement of devices and the routing of interconnections. In order to simplify this task it is convenient to divide the problem into two. First the topological problem is solved in a symbolic layout. After this is done it is quite easy to transform this symbolic layout into the final detailed layout.

We will use stick diagrams as symbolic layouts. In fig. 9 we show a circuit schematic and a stick diagram of a CMOS inverter. In the stick diagram we use a green line for n-diffusion, a red line for p-diffusion and a blue line for metal. A filled circle means a contact and an open circle a transistor gate area (thin oxide area).

The stick diagram formed in this way contains all necessary topological information. It is quite easy to make a detailed layout from it. Furthermore the stick diagram is easy enough to read so it may replace a circuit schematic.

6. CMOS logic circuitry.

Basic logical functions.

The basic logical functions like inverters, NAND- and NOR-gates have already been discussed. These are implemented using the circuits in fig 4. Mixed NAND and NOR functions can also be realized in a similar way, as shown in fig. 10. An important point is how the circuits can be realized topologically. One solution which appears to be quite efficient is to form one row of n-transistors and one row of p-transistors above each other, as shown in fig. 11. This approach can be used quite generally.

The transmission gate.

The transmission gate is a very powerful element in MOS circuitry, see fig. 5. In CMOS both the control signal and its inverse are required for proper operation. It is not too important that they overlap exactly, but when they do not overlap at all, the state of the gate may be undetermined. By combining an inverter and a transmission gate we may form a tristate inverter, Fig. 12a. Such a circuit may also be realized according to Fig. 12b.

Transistor switching networks.

More general logical functions than simple NAND or NOR functions may be realised with transistor switching networks. The principle is demonstrated in fig. 13. Starting with the truth table the idea is to make a conducting series combination to VSS or VDD depending on the output wanted for a specific input. Only n-transistors are used for VSS connections (where a high input makes the transistor conducting) and only p-transistors are used for VDD-connections (where a low input makes the transistor conducting). An example of a circuit generated in this way from the truth table in fig. 13a is shown in fig. 13b. Networks generated in this way may often be simplified. In the actual example we can for example find two n-transistor chains which are equal except for C in the first chain and \bar{C} in the second. It is quite obvious that these can be combined to one chain without the C and \bar{C} transistors. See fig. 13c. Further simplifications can be made using, for example the Carnaugh-diagram method. In fig. 13d is the Carnaugh-diagram for this circuit shown. Since the n-transistors pulls the output to zero it is natural to form the n-transistor network from minterms with zero's and according to this the p-transistor network will be formed from minterms with one's. The inputs must however be inverted in the p-transistor case since they are in "on" state when the gate is zero. From the the simplified functions we can again generate a transistor network similar to the earlier network given in fig. 13c. Finally we can generate a very nice and regular layout using a similar principle as discussed earlier, fig 13e. Again we use rows of n- and p- transistors which are crossed by inputs and gates. The layout is regular and simple enough to be automatized.

Static storage elements.

The standard element for static storage is the flipflop circuit, which in its simplest form is two crosscoupled inverters, see fig. 14a. Such a circuit has two stable states, $Q=1$ or $Q=0$. The state of the flipflop is easily "read" by connecting one of the outputs to a gate. "Writing" the flipflop, i.e. changing the state of the flipflop, is not as simple as reading it, here we can use several methods. One is to write it with "brute force". This can be done by forcing it to a new state with a strong TRI-state output, with a driving capability large enough to force the flipflop inverter output to the new state, see fig 14b. The same principle can be used in a different way to form a set-reset flipflop, fig 14c. In both these cases we have to be careful with the design in order to be sure that the set/reset transistors can force an inverter output to its opposite state.

Another method to "write" the flipflop is to break it up into two series connected inverters during the writemode. This can be done with a transmission gate, as shown in fig 14d. It is also possible to break it up only "partially", by replacing the transmission gate with a large resistance, fig. 14e, which holds the flipflop in its state. The resistance may be very large as it only needs to compensate for leakage currents. If it is large, i.e. about ten times larger than the transistor resistance, the flipflop will quite easily be turned to a new state without disturbance from the output.

A third method, finally, is to replace one or both inverters in the flipflop with NOR-gates to form a set-reset flipflop, fig 14f.

Dynamic storage and its use.

In MOS circuits it is possible to form isolated nodes with very high impedance, fig 15a. Such a node may be used as a temporary storage element (with a storage time in the order of 1 ms.) The circuit in fig 15a. is thus a simple storage element which can be used as a register cell. Register cells like the one shown in fig 15a can be combined to a dynamic shift register, fig 15b. Such a register must be driven by a twophase non-overlapping clock, P1 and P2. During the period P1 is high node 2 is set by the information previously stored at node 1. During the period P2 is high, this information is moved to node 3. The clock signals must be nonoverlapping in this case to prevent data from running straight through the chain of elements. Furthermore its period time must be short enough compared with the storage time to prevent information loss.

The clocking principle of the shift register in fig 15 can be used in other forms of logical circuits. It can, for example, be used in finite-state machines like the one shown in fig 16. The logical

network must contain dynamic storage nodes at the inputs i.e. it cannot be built with transmission gates. Usually it is built with a PLA, a gate network or a transistor switching network.

Dynamic operation can be used in any forms of MOS circuitry. In the case of transistor switching networks, they can sometimes be further simplified by using the storage capability of the output node. In a sequential machine, any transistor combination that only keeps the state from the previous state can be omitted. This means that when you are constructing the network you need only to take care of states that change the outputs from the previous state and that states where neither the p-transistors nor the n-transistors are conducting are permitted.

Clocking and clock generation.

The clocking principle discussed above is very useful and is recommended for all MOS logic design. If used strictly by the principle demonstrated in figs. 15 and 16 there will be no race problems in the design. It is however important to have clock signals of good quality with no overlap. Such signals can conveniently be generated by the circuit shown in fig. 17. In this circuit one clock signal, say at N2, can not go high unless the other signal, N3, is low due to the AND gate generating N2. The clock signals at N2 and N3 are therefore safely not overlapping. The same is true after the buffers if these have an on delay which is shorter than their off delay (as is true for a CMOS buffer inverter with equal transistor sizes). Note, that wire delays and delays in other clock buffers may degrade the clock signals if they are used to drive many gates.

Precharge circuits.

CMOS logic circuits are more complex than n-MOS circuits as one often must use two logical networks to form a logical function (see figs. 4 and 10). A clocked CMOS network may, however, be considerably reduced by using the principle of precharge.

The principle of precharge is demonstrated in fig. 18. Again we use a nonoverlapping twophase clock, P1, P2. When P1 is high (Its inverse is low), transistor T1 is off and T2 is on. Node N1 is thus precharged to a high value. This value is not transmitted further as the following tristate inverter is off (P2 is low). When P1 goes low, transistor T2 is off and T1 is on. The n-network (which forms the desired logic function by a network of n-transistors) thus becomes active and discharges the node N1 if the logic function gives a low signal for the actual input. If the logic function gives a high signal the n-network is not conducting and N1 is kept high due to dynamic storage at the node. Shortly after P1 goes low, P2 goes high, thus propagating the logic output at N1 to the output. The output value will be valid at least during the time P2

is low due to the dynamic storage at the tristate output node. The output is therefore perfectly suited to be an input to a following stage with opposite clock phases. Compare our shift register example in fig.15 with fig. 19.

The precharge principle thus allows us to replace the p-transistor network from the static circuits with a single clocked p-transistor. This is of course very important in circuits with many inputs and even more so in array circuits, like PLA's or ROM's. It also gives faster circuits as the less efficient transistor type, the p-transistor, is used only single and as the number of gates to be driven by the logic signal is halved, thus halving the input capacitance of each gate.

Another important property of the circuit in fig. 18 is that it has no problems with charge redistribution. In precharge circuits this problem easily occur. Charge redistribution occur for example in the circuit in fig. 20. Here, node N2 may be low initially and is not precharged when P1 is low. When P1 goes high the charge in node N1 have to be shared with node N2 (and with some internal nodes in the n-network). The actual voltage level in node N1 is therefore reduced according to the balance between the capacitances at the two nodes. Always be very careful with charge redistribution.

Domino logic.

Circuits of the type shown in fig. 18 may also be cascaded inside the latch functions (the tristate inverters). See fig. 21. In this case both nodes N1 and N3 are precharged high during P1. When P1 goes low, N2 is initially low due to the precharged N1. n-network 2 is therefore inactive until N1 and N2 are valid. The valid signal is thus rippling through the networks. The principle is called domino logic (compare a string of falling domino bricks). Note that this principle prevents us from using any extra inversion between the two n-networks. This is an important limitation to the logical functions which can be efficiently realized in this structure. Another important limitation is the problem of charge redistribution in the second (and following) n-network. As the signal at N2 sometimes change after the precharge cycle is finished, the charge at N3 may be redistributed inside n-network 2. The only case when charge redistribution do not occur is if n-network 2 is a pure parallell array of transistors.

The domino structure may also be realized without the inverters between the n-networks if each second network is replaced by a p-network configuration according to fig. 22.

Array logic.

With array logic we mean logic built from twodimensional arrays of equal or nearly equal cells. Such arrays are often very efficient

and easy to design. They are well suited for structured design and automatic design. We will discuss two types of array logic below, programmable logic arrays (PLA's) and read only memories (ROM's). Another important array type is read write memories (RWM's, often termed RAM's).

Programmable logical arrays (PLA's).

A PLA is usually derived from two-level AND-OR functions as shown in fig. 23. Since any simple logical operation, in most technologies gives an inverted output signal, the network is converted to a NAND-NAND or a NOR-NOR network with a simple boolean operation, fig. 23b. This network may be physically realized with two arrays, one for the input gates and one for the output gates, fig. 23c. Each array is formed as a block of gates. As seen above it is possible to choose between AND and NOR gates. In CMOS technology NAND structures are preferable since the n-transistors are connected in series and they have about two times better current capability than p-transistors. In fig. 23d is the symbolic layout for the complete PLA given.

The static realization shown in fig. 23d is not very efficient due to the demand for two transistors, one n-transistor and one p-transistor, in each cell. This also calls for a well edge in each cell. Much more efficient solutions are obtained using precharge logic.

In fig. 24a we demonstrate a PLA using two level domino logic, with alternating n and p networks. The symbolic layout is considerably simpler than the static version of the same PLA in fig. 23d. Note that each array contains only one type of transistors, which gives a very dense layout (no well edges are needed in the array). See also a cell layout comparison in fig. 24b. In our example in fig. 24a we have chosen to use a series combination of n-transistors as input array (NAND function) as n-transistors are less sensitive to series connection. In the output array we use a parallel combination of p-transistors (again a NAND-function). As this is a second level in domino configuration we also avoid charge redistribution in this way. Also compare fig 23d. A drawback of the proposed structure is that it still may be slow as quite many n-transistors in series must discharge a node with large load. Faster structures are obtained with inverters between the arrays (as buffer amplifiers) or by avoiding the domino technique, that is by dividing the PLA function into two clock phases by having tristate inverters between the two arrays. In such a case both arrays may be parallel transistor arrays.

Read only memories (ROM's).

A read only memory is only a special case of a PLA. If the input array of the PLA is formed as a complete decoder for the input word

we have a ROM. See fig 25. As the input array thus is fixed only the output array is programmed in this case.

7. Electrical constraints.

So far our description has been only qualitative in that we have not discussed transistor sizes, switching speeds etc. The above description is however sufficient to design CMOS circuits that works. The CMOS principle is nice enough to work for any transistor dimensions and any fanout. The speed of the circuit may however be very bad if transistor sizes and fanouts are not considered. In this section we will discuss the speed problem as well as some other electrical constraints.

Switching speeds.

The time needed for switching a signal between two logical states is given by the time needed for the driving transistor to charge or discharge the actual node capacitance. Let us study the inverter in fig. 26. All capacitances connected to node N2 is summed to give C2. In fig. 26 we show what capacitances may make up C2. Note that the gate-drain capacitances are doubled because of the Miller effect. The current driving capacity of the n-channel pulldown transistor is of the order of:

$$IDN = KPN * (WN/LN) * \langle (VDD - VTN) ** 2 \rangle$$

where we have used the formula for the transistor saturation current with its gate voltage equal to VDD. In the same way we may estimate the driving capability of the p-transistor:

$$INP = KPP * (WP/LP) * \langle (VDD - VTP) ** 2 \rangle$$

Normally KPN/KPP is equal to about 2.5 because of a difference in mobility of electrons and holes. However, the high doping of the p-

well makes the n-transistor less efficient so the actual difference in current is smaller, in our case a factor of 2.0. The delays of the inverter can now be expressed as follows:

$$T_F = k \cdot C_2 \cdot V_{DD} / I_{DN}, \quad V_{IN} = 0 \text{ to } V_{DD}$$

$$T_R = k \cdot C_2 \cdot V_{DD} / I_{DP}, \quad V_{IN} = V_{DD} \text{ to } 0$$

Or, if we define the delay for two inverters, the pair delay:

$$T_D = k \cdot C_2 \cdot V_{DD} \cdot (1/I_{DN} + 1/I_{DP})$$

The factor k takes care of the fact that the transistor current do change during the swithing and the actual definition of the delay. The most natural way to define inverter pair delay is to measure it inside a long chain of identical inverters. The reason for the long chain is to simulate the waveform occuring in a real system. An appropriate value for k for such a definition is 2.

As can be seen from the above formulas, the delay depends strongly on the capacitive load. For a large capacitive load it is therefore reasonable to increase the driving capability of the transistors. This is easily done by increasing the transistor widths, W_N and W_P . In order to obtain symmetric delays (the same value of T_F and T_R) one may make W_P 2.0 times larger than W_N , thus compensating for the difference in K_P .

The same formalism as given above can be used for NAND and NOR gates. Two conducting transistors in parallell will have their currents added. Two transistors in series will be equivalent to a transistor with double gate length, they will thus have only half the current capability of a single transistor (this can again be compensated by increasing their widths).

Also the transmission gate may be handled by the same formalism, normally assuming only one of the transistors active (the n-channel device when the transmitted signal is zero and the p-channel device when it is equal to V_{DD}).

Power consumption.

The power consumption in CMOS-circuits consists mainly of the current used to charge(discharge) the load capacitors. When a signal node with the capacitance C_2 goes high, it will consume the charge Q_D from V_{DD} :

$$Q_D = C_2 \cdot V_{DD}$$

This gives a approximate maximum current consumption, for one gate, of:

$$I_{DD} = F_{MAX} \cdot C_2 \cdot V_{DD}$$

where F_{MAX} is the maximum operating frequency to be used (Note that

$I_{MAX} < 1/TD$). It is important to analyze this current and design the VDD and VSS supply lines so that they always can support the current. In logical circuits we may allow a maximum voltage drop of $0.1 * VDD$ in the supply lines. For a large number of gates or a complete chip it is normally not necessary to sum all maximum currents. A gate consumes current only when it is switched and only a small fraction, say USAGE, of all the gates are active simultaneously. Thus, for N gates we may use the following expression for the current consumption:

$$IDD(N) = USAGE * N * IDD$$

An appropriate value for USAGE must be estimated by the designer. The chip power consumption, finally, is given by:

$$P = VDD * IDD(N)$$

The chip power must normally be limited to about 500mW, in order to not heat the chip too much.

Conductors etc.

When designing circuits, not only the transistor parameters are of importance. Also the properties of conductors, like metal or diffusion wires, are important. These are characterized by two parameters, capacitance per surface area and surface resistivity. The meaning of capacitance per surface area is obvious. Surface resistivity have the dimension OHMS, sometimes named ohms/square, and the resistance of a wire with the length L and the width W is given by:

$$R = RS * L / W$$

where RS is the surface resistivity.

Latch-up.

CMOS circuits have a special problem called latch-up which must be considered by the designer. The problem is that the pnpn-structures which occur along the silicon surface may act as thyristors. Such a structure may be a p-transistor source, n-substrate, p-well and an n-transistor source. These thyristors may turn on as a result of some transient signal, causing the whole circuit supply to be shorted. To prevent the pnpn-structure to turn on we recommend the designer always to connect the p-guard to VSS at short distances (about 100 LAMBDA apart). By such an action there will always be a small resistance (less than 2 kohms) between the thyristor p-gate and VSS, thus preventing the gate-cathode diode to become forward biased.

Supplies.

Finally we need to discuss the supplies, VDD and VSS. The central supply network could be designed in many ways. We recommend that

two heavy metal lines for VSS and VDD surround the hole chip, inside the pads, and that cells are supplied from these lines by metal wires. Inside cells, diffusion lines may be used for the supplies. It is quite convenient to use guards (or diffusions of same type as the actual substrate) for this purpose. Especially for the fully guarded process we will get supply lines automatically in this way. One must however be very careful with the series resistance in the supply lines. Cells which are operated frequently must have a supply resistance of less than 10 kohms at 5 V or 3 kohms at 10 V supply voltage. Note also that any p-well must be connected to VSS. Finally the n-substrate must be connected to VDD at some points in the chip area.

8. Design methodology.

When designing smaller integrated circuits, with 10 to 100 devices, special design methods are not really needed. Normally such a circuit is designed by a circuit diagram, which may be transformed into a layout directly by the designer. The layout is made by drawing or by using a graphical editor on a computer. It can be checked manually for design errors. Pure geometric design rule violations can also be checked by a computer program. It is obviously very important that the design is correct before fabrication. The fabrication step is too expensive to be used as a mean of checking the design. Still, it may be not be known if the circuit diagram is correct, that is if a "correct" circuit (according to the circuit diagram) perform as wanted. Again, this can not be checked by fabrication but may instead be checked by simulation. Simulation of a small circuit is easily done with an analog simulator like SPICE. An analog simulator is a computer program which simulates the the electrical behavior of the circuit by using mathematical models of the devices in the circuit.

Our goal is to learn how to design large integrated circuits, with 100 to 100000 devices. Then, a new methodology is needed as all the obvious methods discussed above, fail. A system with thousands of devices is too large to be handled by direct layout from a circuit diagram. This is experienced in several ways, the design time will be very large, direct layout control and analog simulation will give too long runtimes on the computer.

In the following we will discuss a design methodology, well suited to handle large systems, termed "structured design". This term contain several modern methods. The basic idea is to divide a large problem into subproblems in an ordered way. Other important ideas are topdown design and correctness by construction. Topdown design means that the work starts with a functional description which then is transformed into a more detailed description, for example a block diagram. The block diagram is then transformed to next level of description and so on, until a complete layout is defined. In fig. 27 is some common levels of design demonstrated.

Strict topdown design is normally impossible since a lot of interaction between different levels is needed. Correctness by construction means that the design is done with very strict design rules (which can not be violated) so that errors can not occur. Correctness by construction can for example be obtained in a design step which is fully automatic (through a computer program).

Structured design.

The structured design methodology for integrated circuit design was pioneered by Carver Mead and Lynn Conway and is presented in their wellknown book "Introduction to VLSI systems" (Addison Wesley, New York, 1980) It contains two important parts, hiarchy and regularity. Hiarchy means that the design is partitioned into parts in a hiarchical manner. If the whole design is considered to be cell 0, this cell contains other cells, say cell 1 to 5, which contains other cells ... until we reach the lowest level cells (often called leaf cells), see fig. 28. Regularity considers the wiring strategy. If no wiring strategy is used (random wiring), the wiring normally takes more space than the logical cells. Also, wiring normally takes needs a lot of design time and gives rise to many errors. Regularity in wiring often means that the overall wiring is done before the cell layout. Regularity also means use of regular structures like Read Only Memories, Programmable logical arrays etc. Structured design is conveniently supported by algorithmic methods, that is by using programming languages, compilers and computers in the design work.

Hiarchy.

A hiarchical method of design is absolutely necessary in large designs, but is very useful also in small designs. An important principle is that when a structure, for example a transistor, is designed and checked, the same structure can used also in other parts of the design without redesign and without rechecking. If each structure is simple enough the number of design errors is drastically reduced due to the ease of control. Hiarchy is used in all levels of design (subsystems, blocks, gates, transistors, masklayers). Let us study some examples from the CMOS cell library CLIB2. Fig. 29 illustrates the use of hiarchy in the layout level. Fig. 29a is a transistor gate region, containing diffusions, field oxide cut and metal. All lambda measures are correct for a standard transistor and we need not to check this measures any more. Fig 29b is a contact hole with all necessary layers. Combining a few cells of those types with two metal wires and an extra diffusion for guard band gives an inverter cell, fig 29c. Note that at each level the amount of new information is limited, which makes control easy. Also each subcell is chosen so it fits well to other subcells and have a welldefined function. The inverter is now a complete logical cell which can be combined with other cells, see fig. 29d.

It is very useful to divide the hierarchy in cells containing rectangles (subleaf cells and leaf cells) and in cells containing only other cells and wires (composition cells). Transistors and contact holes are thus subleaf cells, the inverter is a leaf cell and the combined cell in our example is a composition cell, see fig. 29c and 29d. The whole hierarchy described here may be described by the layout language CIF which is hierarchical in itself. As composition cells contain only cells and wires, they can alternatively be described in more simple layout languages (the language does not need to contain rectangles, polygons, field oxide etc.) An example of such a language is CHICO.

It is very important to note that the hierarchy discussed above deals not only with the layout description of the design but with other descriptions as well. A leaf cell is for example described by a functional description (for example "inverter") or a circuit diagram. A composition cell can always be described by a circuit, logic or block diagram containing other composition cells (blocks) or leaf cells (logic gates or transistors), see fig. 30.

The hierarchy should be chosen so that any cell is well defined in function and interfacing properties. The partition of the problem into cells must be done with great care.

The hierarchy also supports the use of standard cell libraries. A standard cell library is a library of cells which is predesigned by specialists in leaf cell design and made available to the chip designer. The cells must be well proven, debugged and characterized. For each cell electrical data, logical delays etc. is given. It is often possible to use only standard cells from the library in a new design, again making design easier and with less errors. Standard cells may of course be used together with own leaf cells.

Regularity.

As mentioned above, regularity means a regular wiring strategy. Such a strategy use less wiring space, it is faster to design and it gives less errors. The basic idea is very simple. See fig. 31. Let all long range wires go straight through large parts of the chip and straight through the cells. Such wires are for example supplies and clocks but may also be buses and controls. Two sets of long range wires may normally be used, one vertical and one horizontal. In metal gate CMOS one set is made in metal and the other in diffusion. Short range wires should occur only between adjacent cells, preferably formed by direct abutment of the cells.

This basic idea is accomplished in two ways. First, in any layout work, the designer should start with a global wiring strategy, putting down all long range wires in two directions on his floor plan. Apart from supplies and clocks it is often useful to put

"data" wires and "control" wires perpendicular to each other. It may even be worth while to redesign the logic block diagram at this point, in order to obtain a more regular wiring. Even better is of course to consider the wiring strategy from the beginning, at the system design stage. The ultimate goal is to be able to lay out all cells, and have the wiring done by that, as suggested in fig. 31. Second, the cells to be used must be designed into the wiring strategy used. As mentioned above the goal is to have all cells fitting each other so that no external wiring is needed. In practice this is accomplished in two ways. For cells which is used many times in similar or same environment, it is often worth while to design special leaf cells for the actual design. This also means that an overall architecture which uses regular structures (many equal cells in rows or arrays) is very efficient both in design and performance. An extreme example of this principle is a ROM. For cells which is used only very few times in the same environment, library cells are preferable. Such cells may still fit into our wiring strategy if they are designed in a proper way. It is reasonable to have supplies, clocks and inputs on all cells as wires straight through the cells and in the same directions in all cells. A cell strategy example from the library CLIB2 is shown in fig. 32.

Algorithmic methods.

It is quite obvious that computing techniques can be of great use in design of integrated circuits. In this section we will not discuss conventional computer aids in design, such as special programs like simulators or graphical editors. We will instead discuss the direct use of computing techniques in the design.

One way of using computing techniques in integrated circuit design ~~is to describe the circuit design~~ is to describe the circuit directly in a programming language (for example PASCAL). A listing of this program is then the circuit definition. Editing the program means changing the circuit. Executing the program means that a mask description is produced. If this principle is used for the layout the program is often termed LAP (LAYOUT Program). The idea may be extended also to other descriptions than layout, so that a program execution for example simulates the circuit.

If we use a LAP to describe our leaf cells we may introduce parameters of of these cells. We may for example define a NOR-gate cell with n inputs. Each time the cell is used, a certain n is defined and the program produces a layout description of the appropriate gate upon execution. We may also have flexible cells in our library, we may for example give the distance between two inputs as a parameter. This can be very useful if we want direct abutment of cells as discussed in the section "regularity" above.

The idea of parametrized leaf cells can be expanded into larger parametrized cells. We may for example define a parametrized PLA-cell with size and truth table as parameters. This can be further extended to simple finite state machines and the program may include logical optimization etc. When the LAP idea is extended like this the system is often called "Silicon Compiler" instead of LAP. The idea is that a high level description of a circuit, ~~like the truth table of a circuit~~, like the truth table of a finite state machine, is directly compiled into a layout description in a way similar to the compilation of a high level programming language into executable code.

9. Some process parameters

We will here list some parameters of the ASEA-HAFO metal gate CMOS process.

Transistors:

KPN=21E-6 A/(V**2)
 KPP=5.4E-6 A/(V**2)
 VTN=1.4V
 VTP=-1.7V

maximum currents for W/L=8/3 IDN=0.33 MA at 5V 5.6MA at 15V
 IDP=0.15 MA at 5V 2.1MA at 15V

Resistivities:

P-diffusion RDP=70 ohms/SQ
 N-diffusion RDN=10 ohms/SQ
 metal RM=0.02 ohms/SQ

Capacitancies

P-diffusion CDP=0.08 MF/SQM (millifarads)
 N-diffusion CDN=0.2 MF/SQM
 Metal on thick oxide CM=0.03 MF/SQM
 Metal on thin oxide CMG=0.4 MF/SQM

Current density recommendations

Maximum current in 8 um(4 LAMBDA) metal wire 8MA
 Maximum current in 8 * 8 um^2 (4*4 LAMBDA) contact 3MA
 Note that the contact window periphery is the limiting factor, thus use several small windows instead of one large window if necessary.

10. The silicon gate CMOS process and design rules.

A more modern process for CMOS manufacturing is the silicon gate process. It is quite different from the metal gate process and need therefore quite different design rules. Also, this process exist in many different forms. Hopefully the different forms leads to similar rules so that we need only to consider one form here. We have chosen to describe a process using selective oxidation and a p-well.

The starting material is n-type silicon. It is first coated with a silicon nitride layer which is then patterned and etched to define device and field oxide regions. See fig 33. During subsequent oxidation a thick oxide can be thermally grown in those regions on the wafer not covered with nitride, whilst oxidation is prevented in other areas.

A thick layer of photoresist is then spun on the wafer and patterned to define the p-well regions. The well dopant is implanted through the holes in the resist layer (and through the nitride). After have removed the photoresist the well dopant is driven into the required depth by diffusion and the field oxide is grown in those regions not covered by nitride.

The nitride is now etched off and a thin oxide is grown in its place. This thin oxide will be the active gate oxide in our transistors. A layer of polysilicon is deposited over the wafer, patterned and etched to form transistor gates and the first level of interconnect.

Separate masks is then used for the p-channel and n-channel source and drain regions. These masks only give the approximate area of p and n-doping. The exact edges of doping is given by the polysilicon and field oxide edges by using implant energies which are too small for the dopant ions to penetrate these layers. In such a way the dopings are "selfaligned" with these edges. Implantation of p and n-dopants is followed by a short anneal to activate the dopants.

Following source and drain dopings, the wafer is again oxidized, now forming a new oxide over the polysilicon layers. Contact holes are cut through the oxide layers to make connections to the underlying doped silicon or polysilicon. Finally aluminum is evaporated over the wafer and patterned to form the second interconnection layer and the whole wafer is covered by a protective glass layer.

Note that the polysilicon interconnect layer can not make contact directly to a source/drain region in this process. Also note that any doping crossed by polysilicon forms a transistor. Therefore any connection of polysilicon to a silicon doping must be done via metal and any doping interconnection under polysilicon must be done via a metal bridge.

The design rules for this process are given in Table 5 and illustrated in fig. 34.

Also stick diagrams will be somewhat different in the Si-gate process. In fig. 35 we give a number of examples of stick diagrams, corresponding to the diagrams given earlier for the metal gate process (figs. 9, 11, 13 and 23).

The design rules defined here is also applicable to the silicon gate silicon on sapphire (SOS) CMOS process. The main difference between the two processes is that the n and p type transistors are made in different thin silicon islands which is made on insulating sapphire. This means that there is never a latchup problem. Also any n and p diffusion can be close together without risk for substrate shorts, for example for making contact between them.

Table 5. Design rules for the Silicon gate CMOS process.

Rule	LAMBDA
Field oxide cut (Epi in SOS)	
F1 min width	2
F2 min separation	2
F3 min separation to p-well outside p-well	5
F4 min separation to p-well inside p-well	3
Polysilicon, general	
P1 min width	2
P2 min separation	2
P3 separation to field oxide cut	1
Polysilicon, transistor	
P4 gate extension at transistor edge	2
P5 min separation to field oxide inside field oxide cut	2
Polysilicon, contact cuts	
P6 min overlap over contact cut	1
Contact cuts	
C1 min size	2x2
C2 min separation	2
C3 min separation to field edge inside field cut	2
C4 min separation to polysilicon	2
C5 min size p to n diffusion contact	2x6
C6 No contacts over gate area	
C7 No contact over both polysilicon and field oxide cut	
Metal	
M1 min width	2
M2 min separation	2
M3 min overlap over contact cut	1
M4 min spacing to parallel polysilicon	1
M5 min spacing to parallel field oxide cut	1
p-well	
W1 min width	2
W2 min separation	8
p and n diffusions	
D1 Overlap over field oxide cut	=1

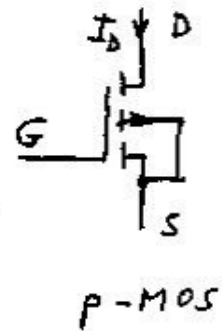
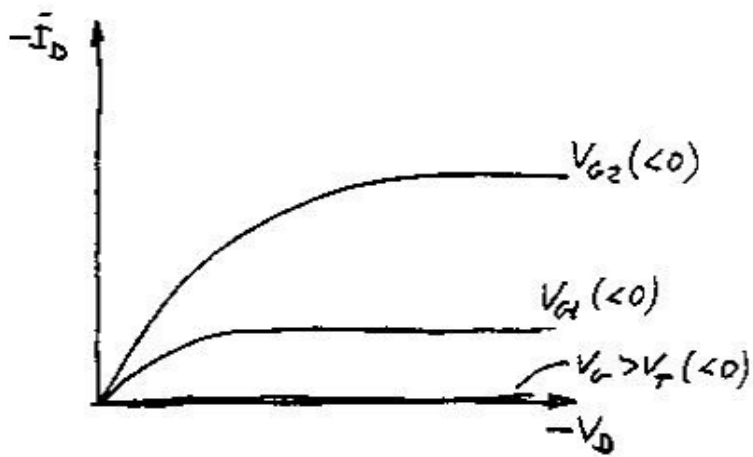
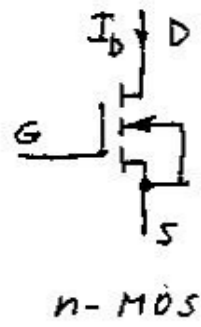
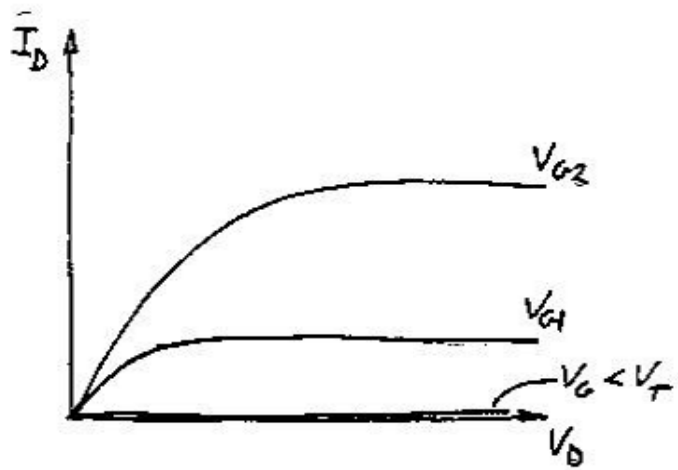


Fig. 1

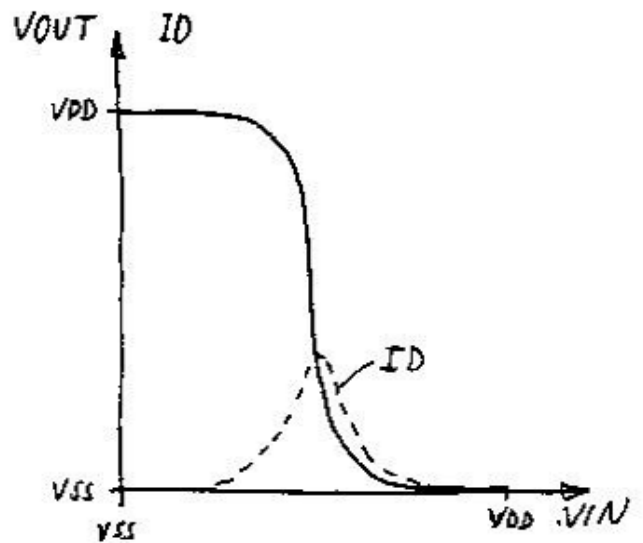
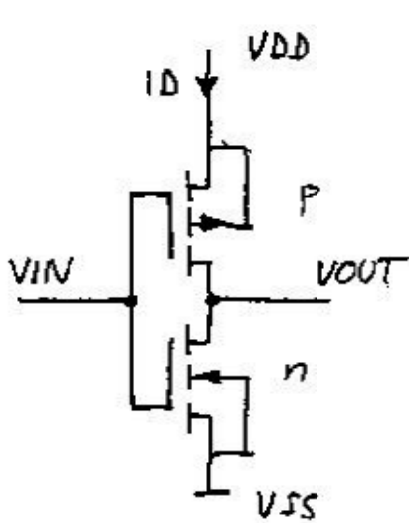


Fig 2.

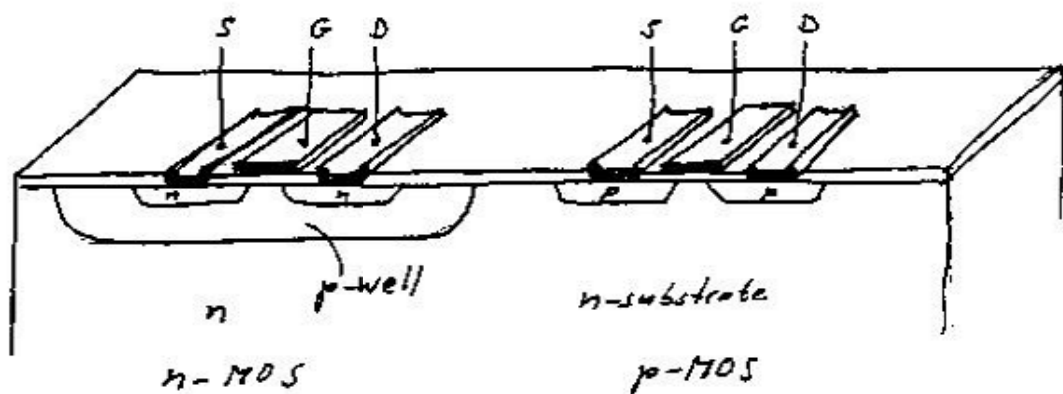


Fig. 3

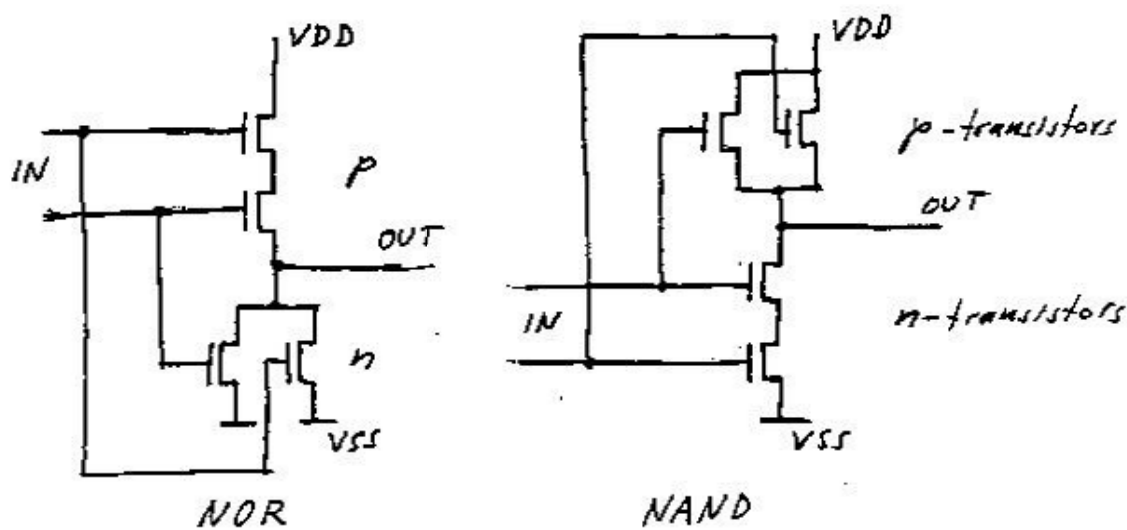


Fig. 4

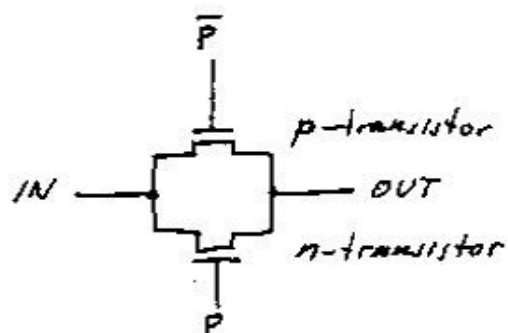


Fig. 5

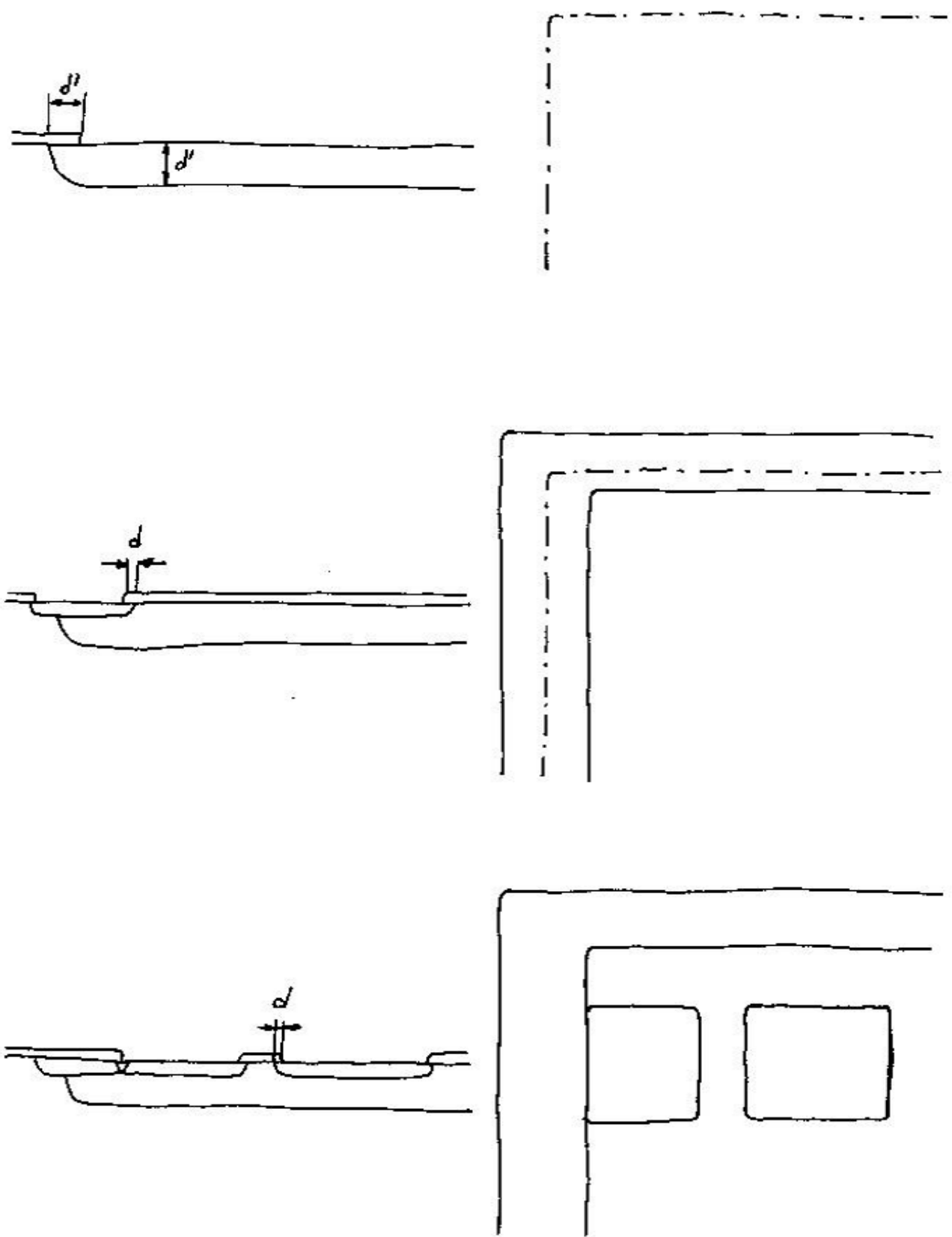


Fig. 6

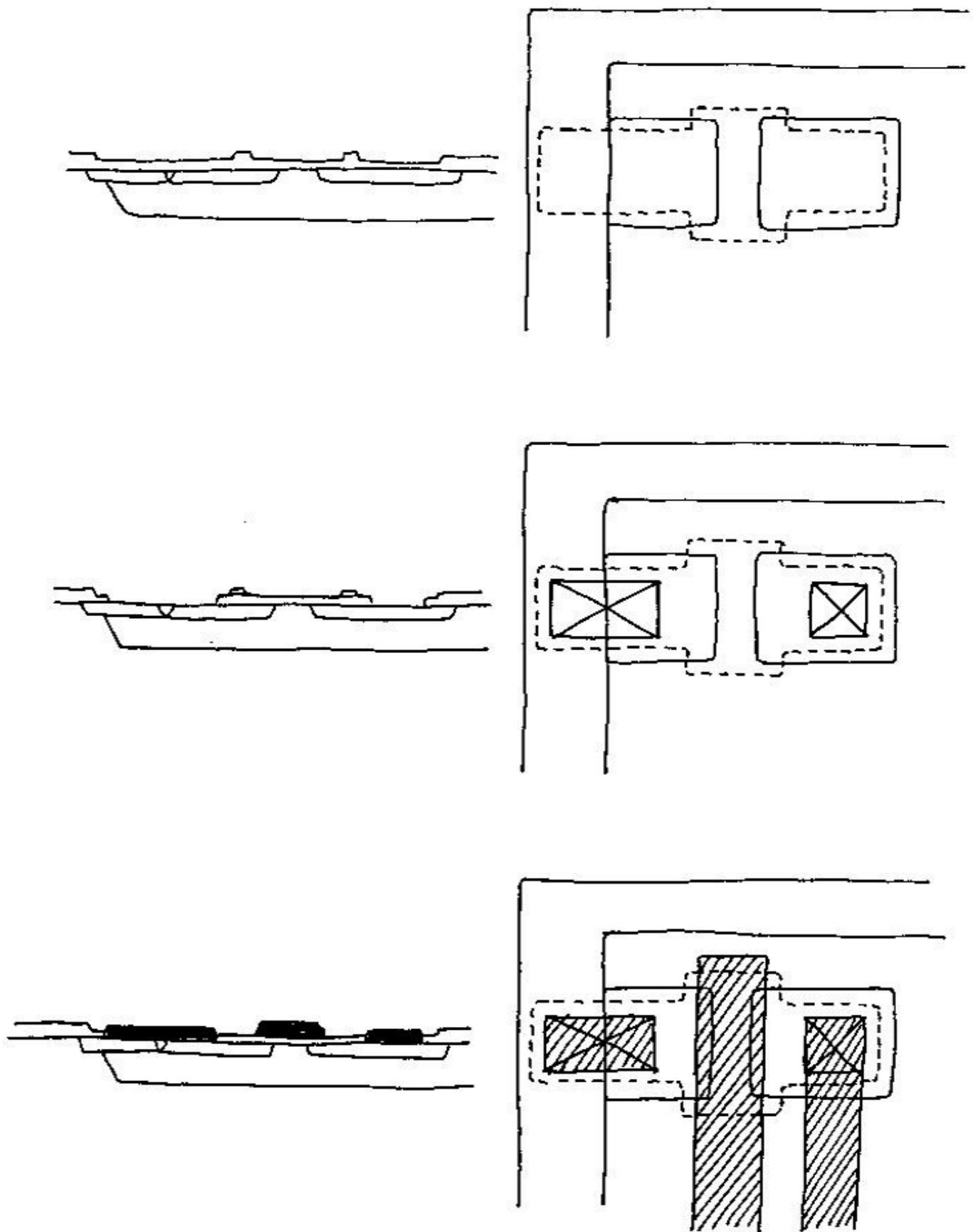
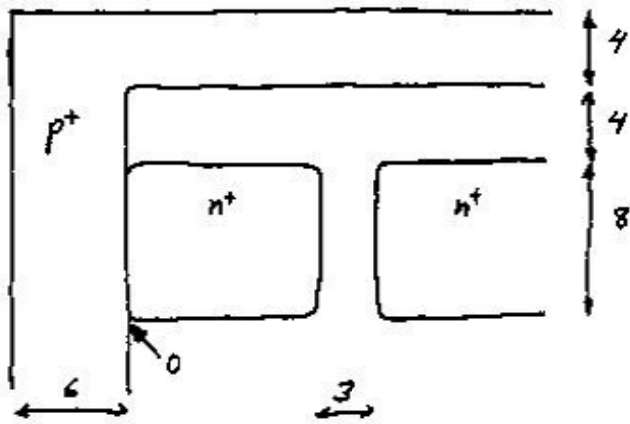
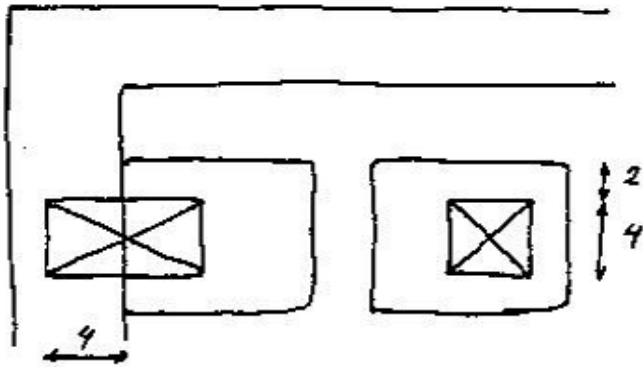


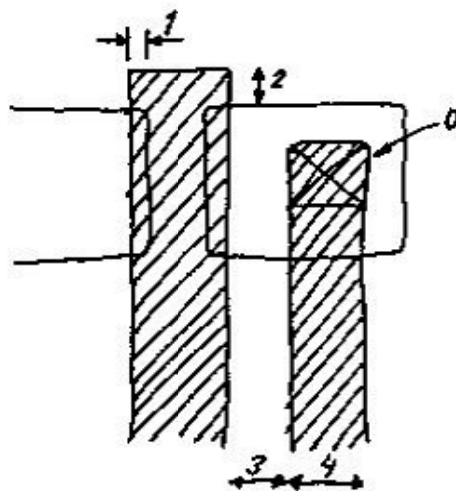
Fig. 6 contd.



Diffusion rules

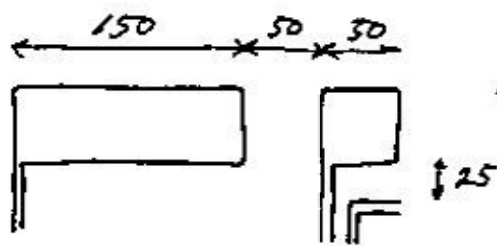


Contact cut rules

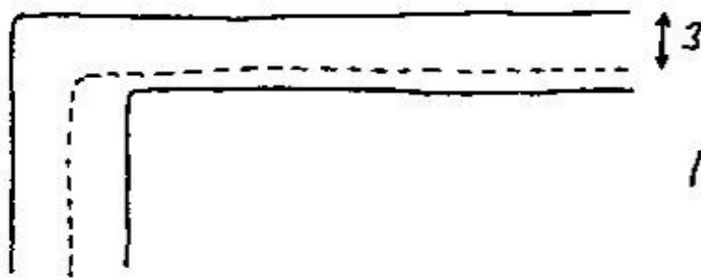


Metal rules I

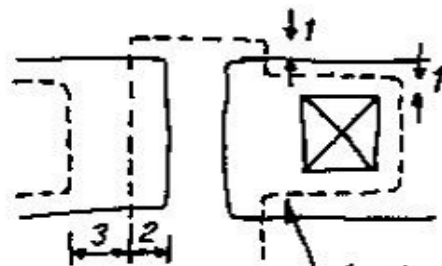
Fig. 7



Metal rules II

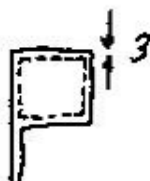


p-well rule



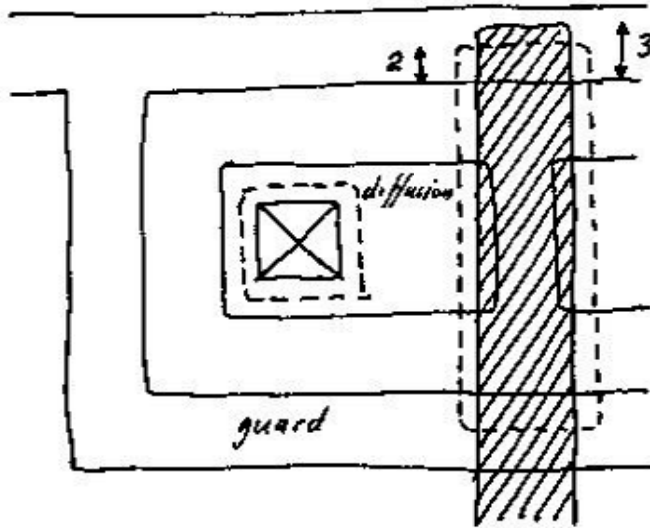
Field oxide rules

field oxide cuts combined to avoid small separation



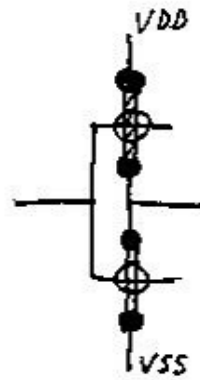
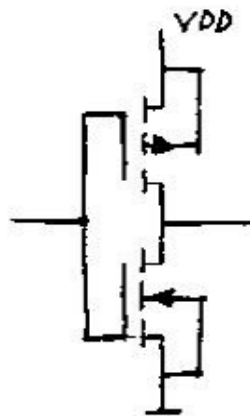
Overglass rule

Fig. 7 contd.



Full guard bands

Fig. 8






-  p-diffusion
-  n-diffusion
-  metal

Fig. 9

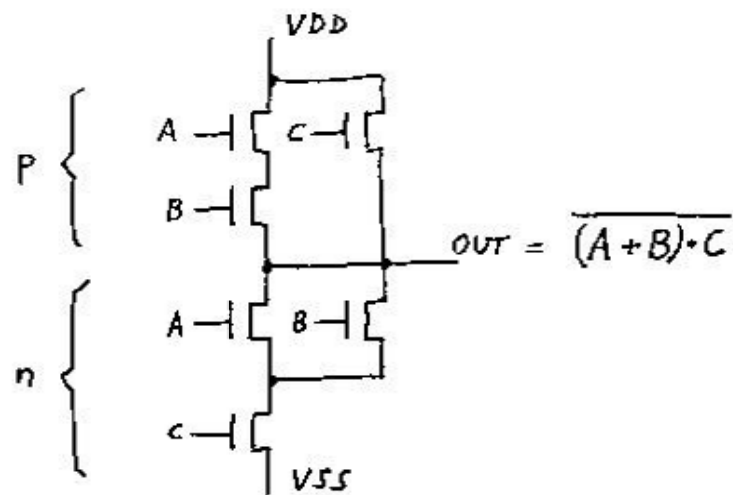


Fig. 10

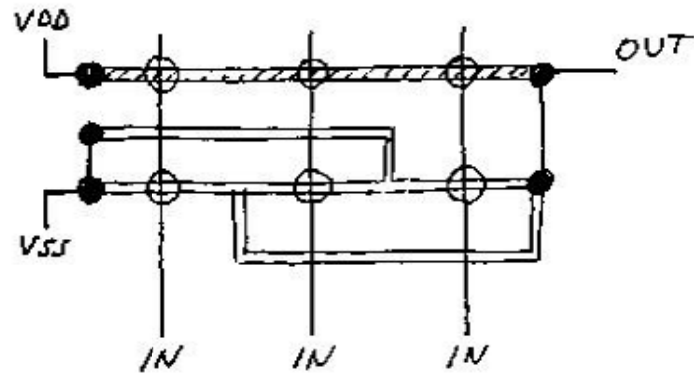


Fig. 11 3 input NOR gate

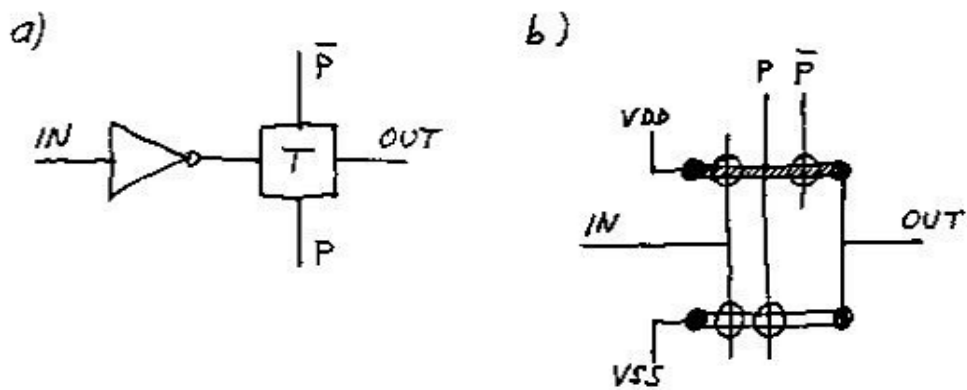
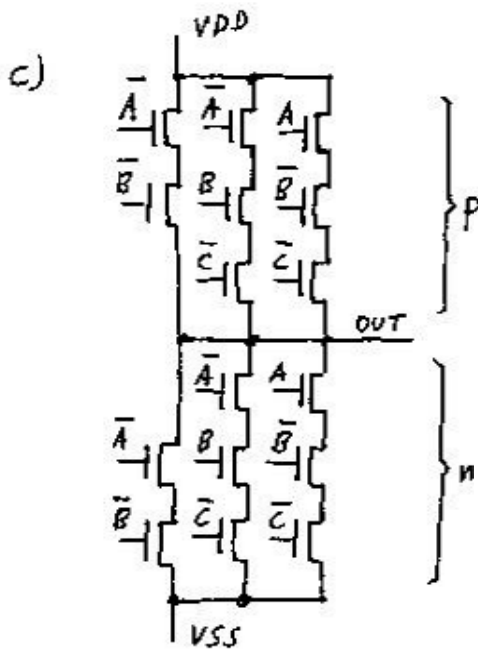
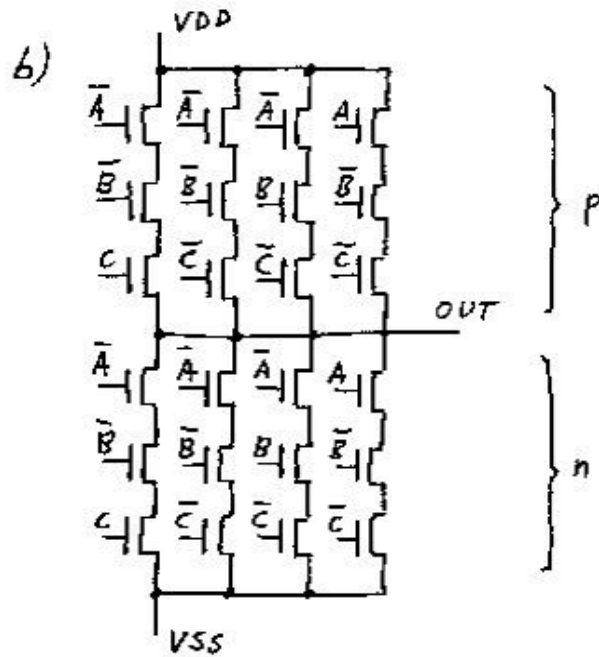


Fig. 12

a)

A	B	C	OUT
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1



d)

A	BC			
	00	01	11	10
0	0	0	1	0
1	0	1	1	1

$$OUT_N = \bar{A} \cdot \bar{B} + \bar{A} \cdot \bar{C} + \bar{B} \cdot \bar{C}$$

$$OUT_P = \bar{A} \cdot \bar{B} + \bar{A} \cdot \bar{C} + \bar{B} \cdot \bar{C}$$

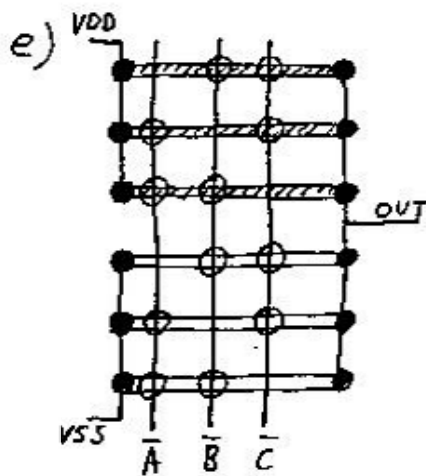


Fig. 13

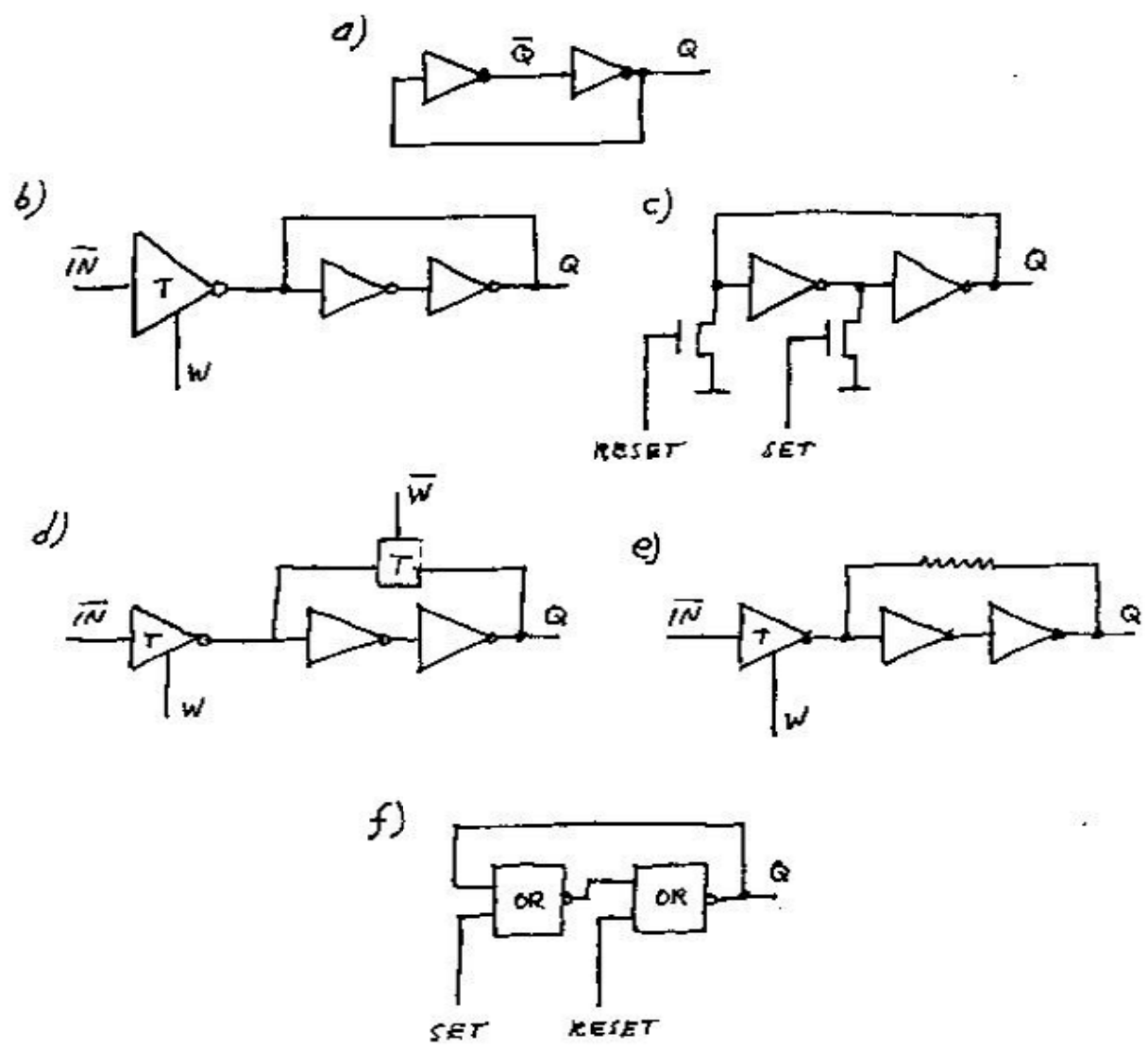


Fig. 14

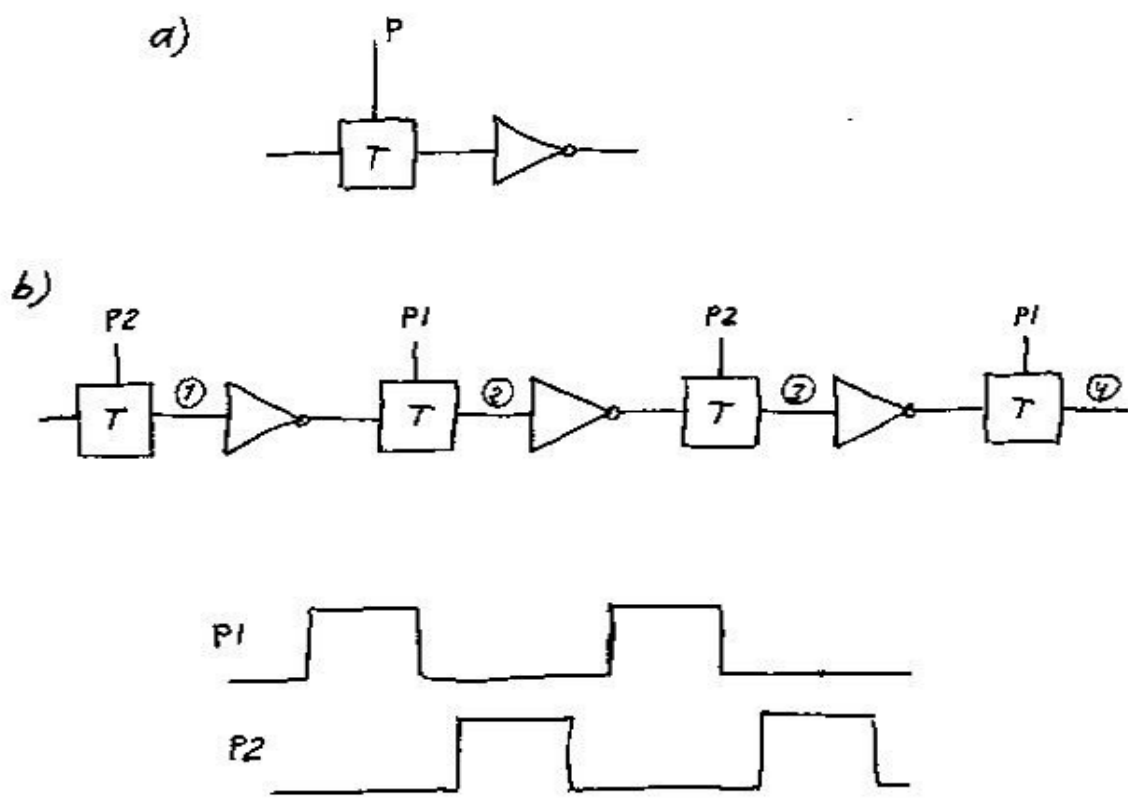


Fig. 15

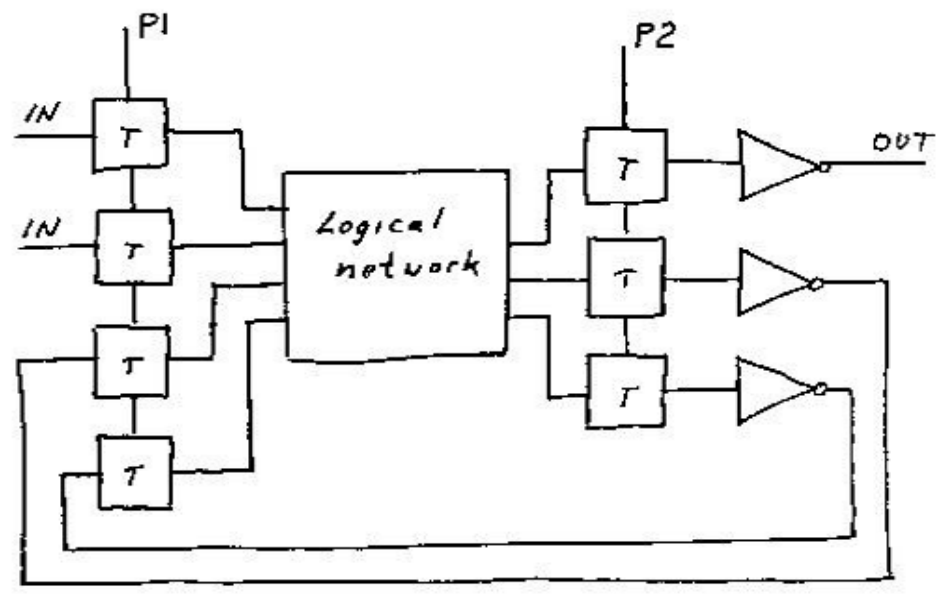


Fig. 16

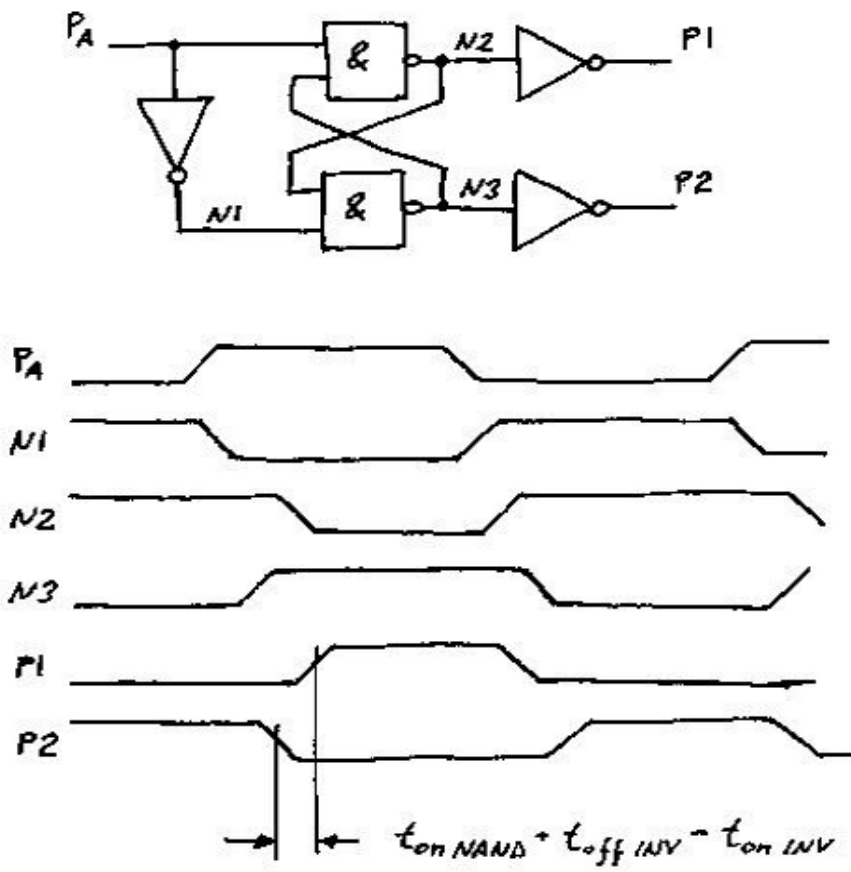


Fig. 17

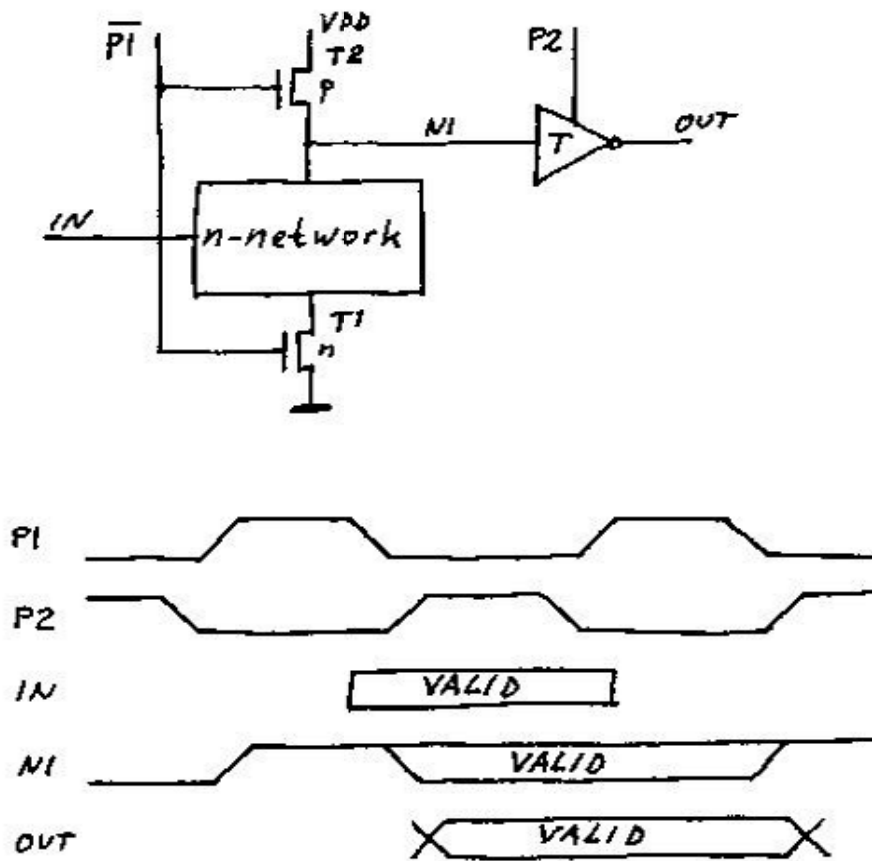


Fig. 18

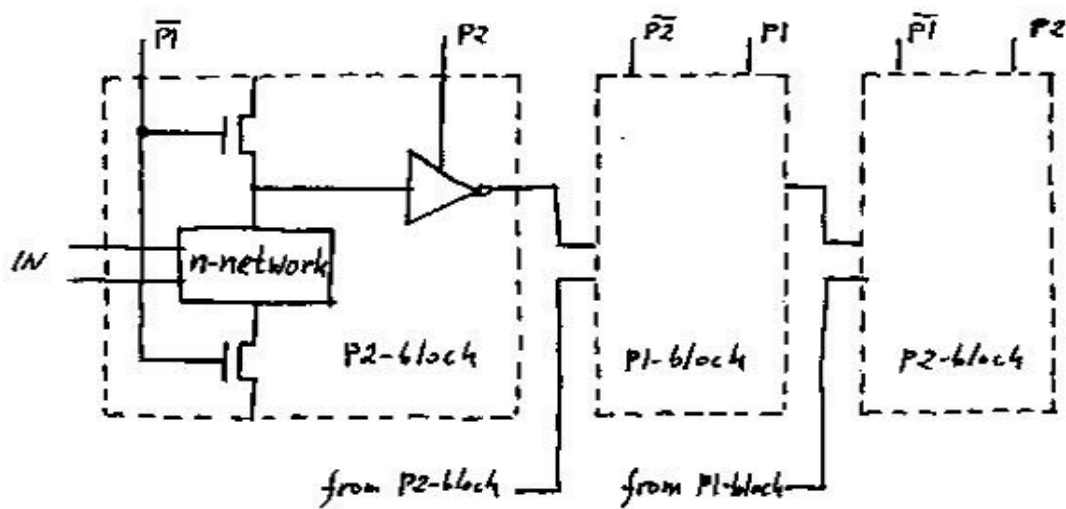


Fig. 19

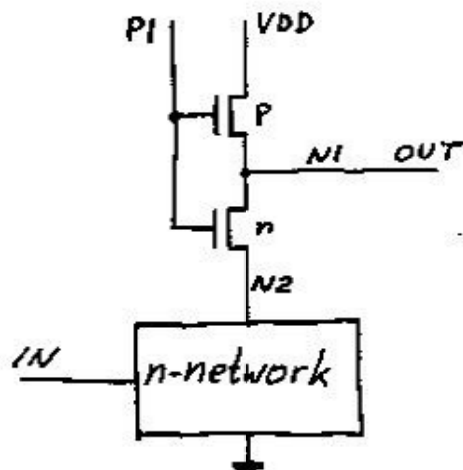


Fig. 20

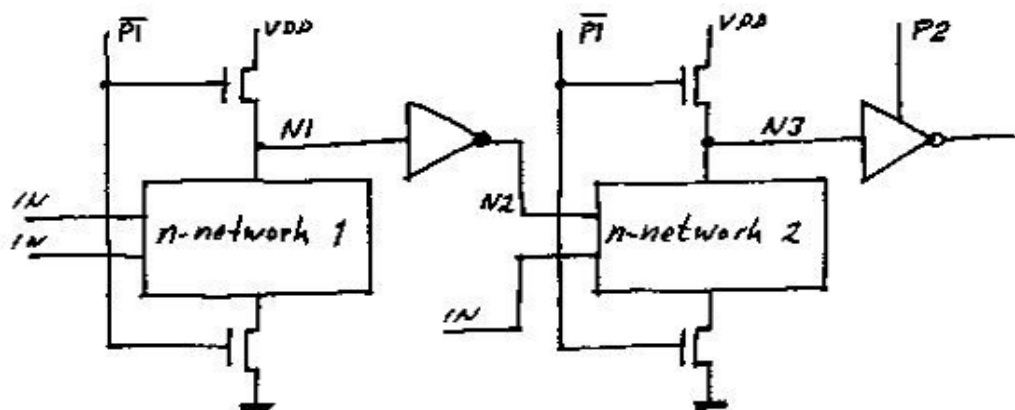


Fig. 21

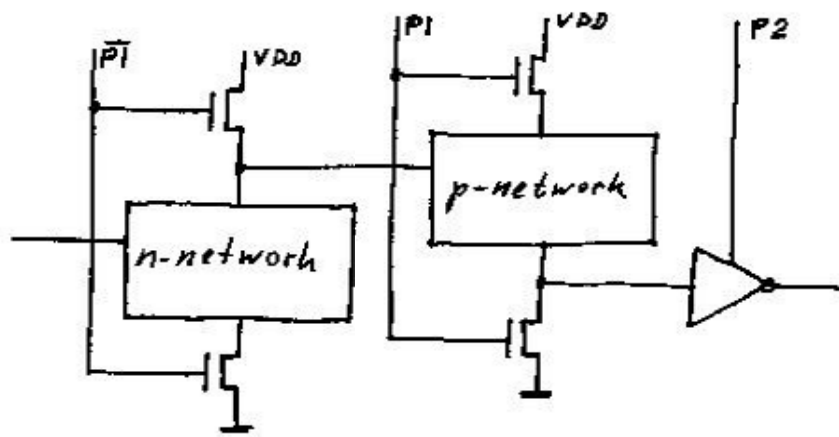
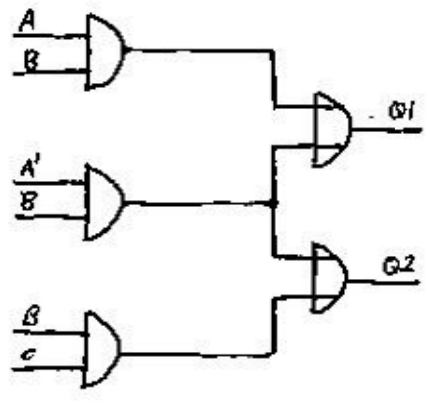
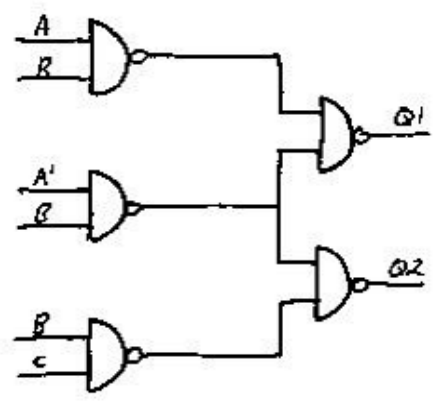


Fig. 22

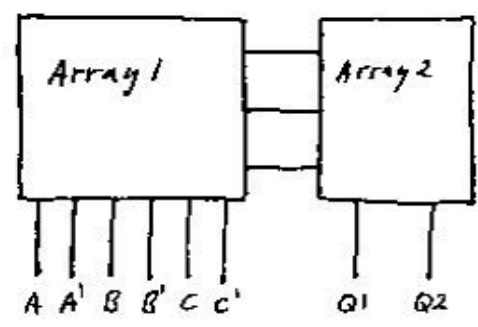
a)



b)



c)



d)

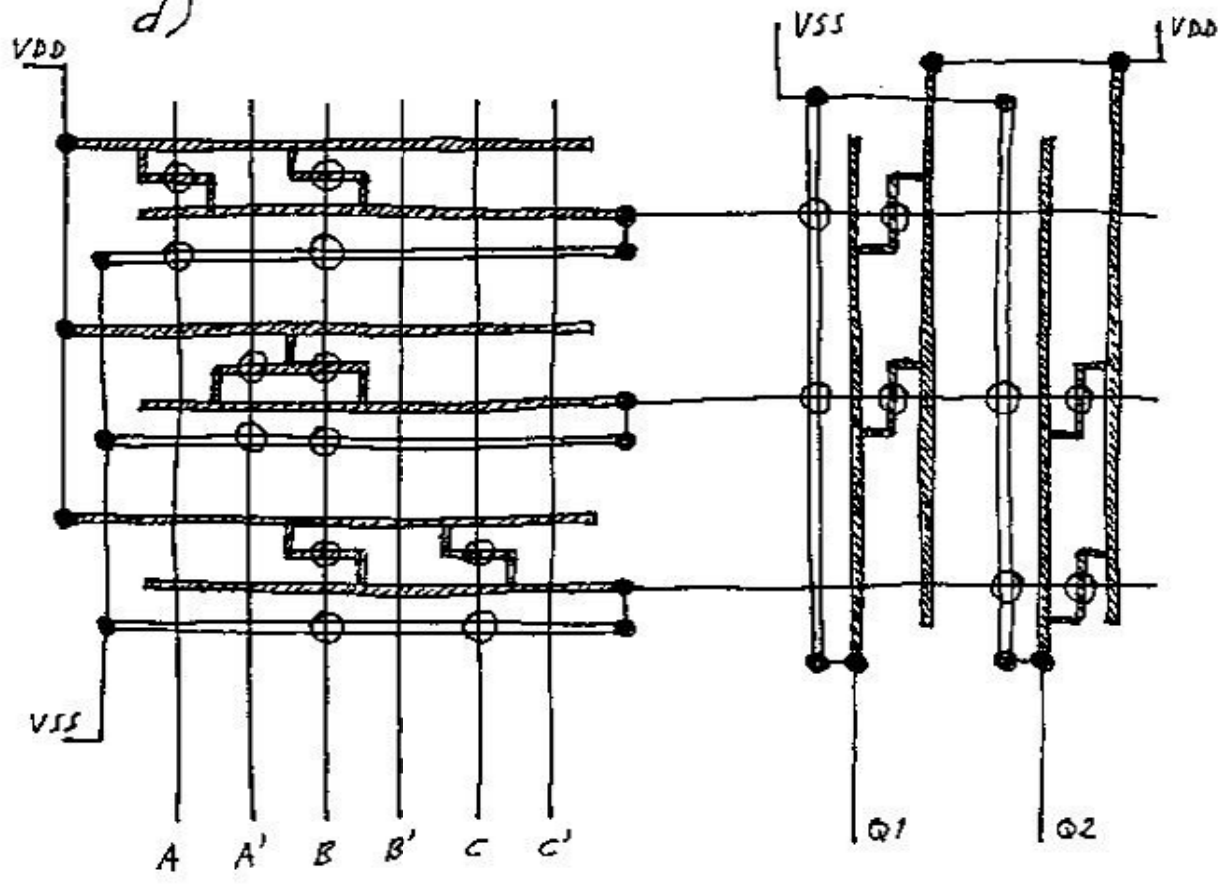


Fig. 23

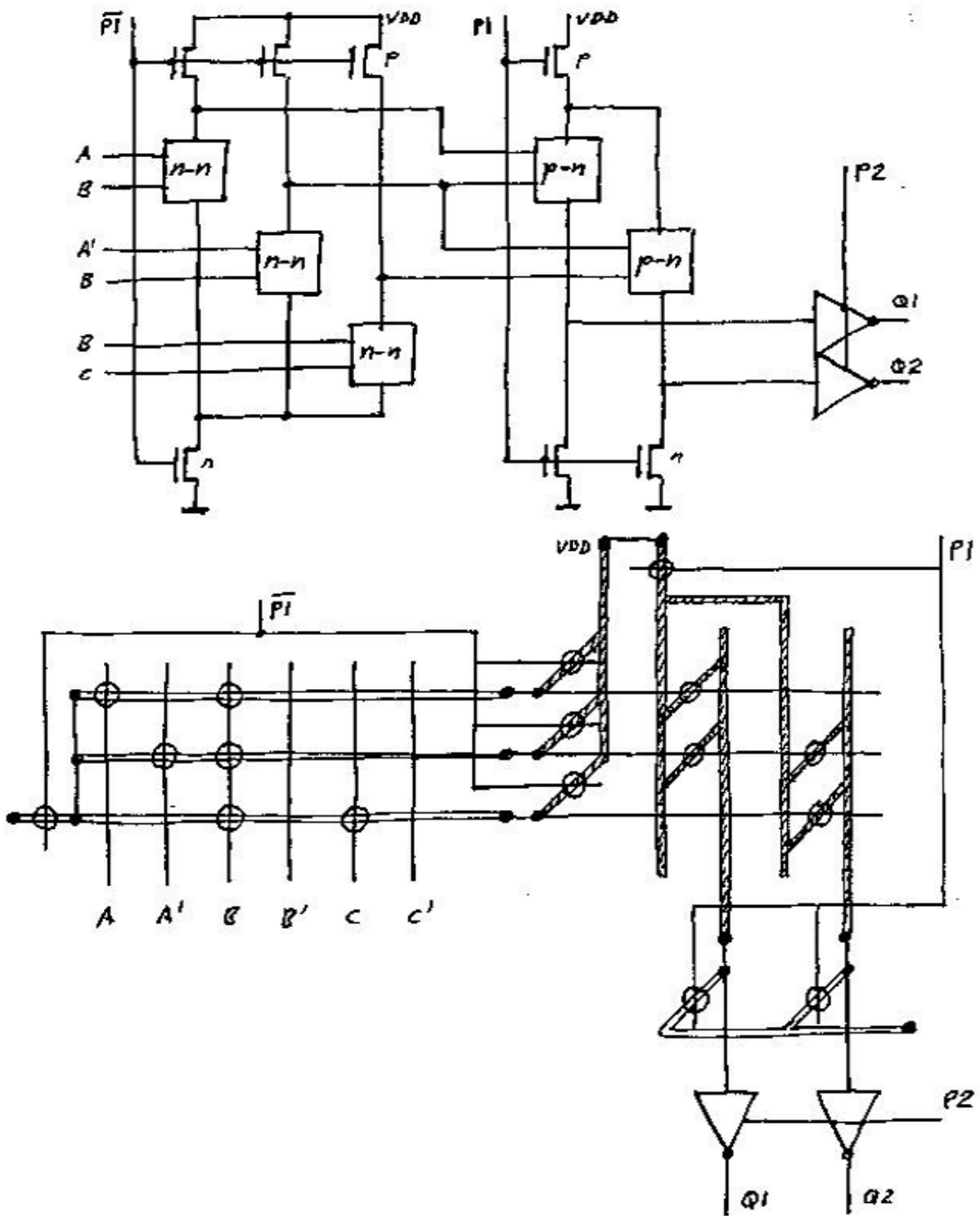


Fig 24a

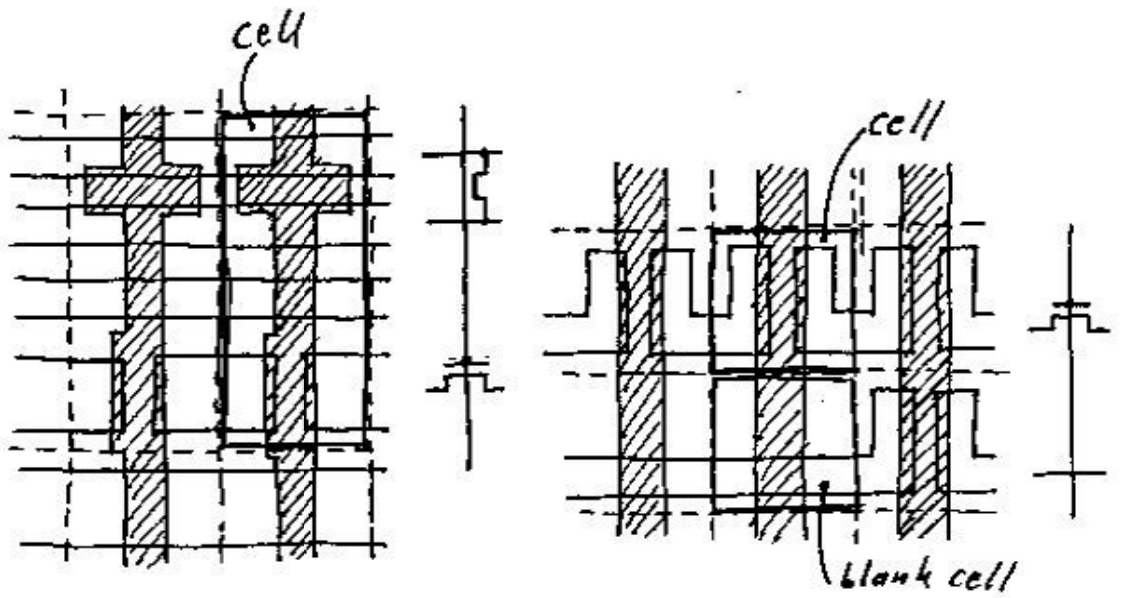


Fig 24b

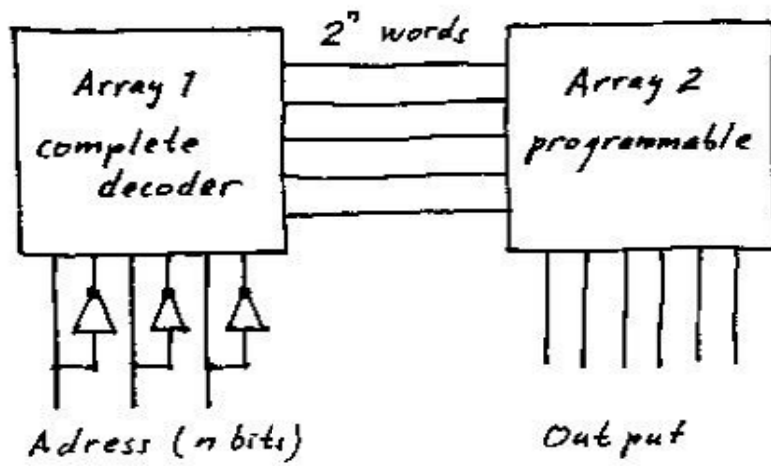
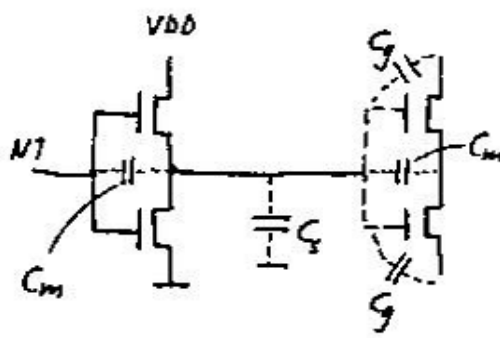


Fig. 25. ROM configuration



$2G$ gate capacitances
 $2 \cdot 2C_m$ "Miller" capacitances
 C_s stray capacitances

 $\Sigma C2$

Fig. 26

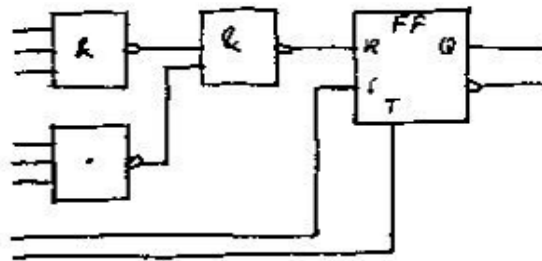
Functional description



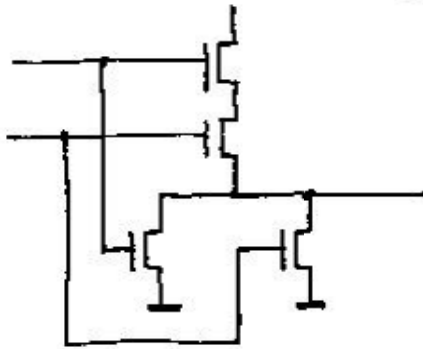
Block diagram



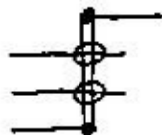
Logic diagram



Circuit diagram



Symbolic layout



Layout

Fig. 27 Levels of design.

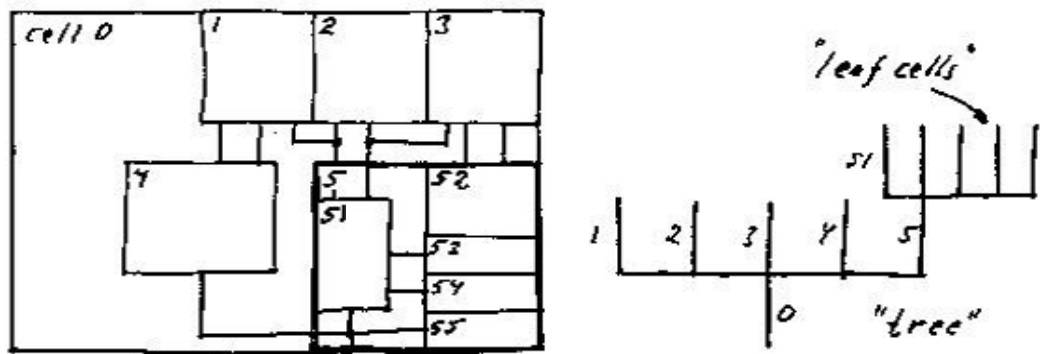
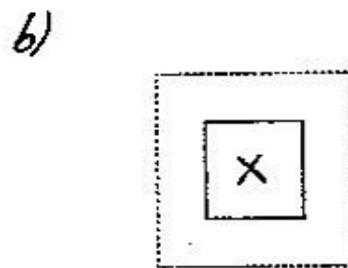
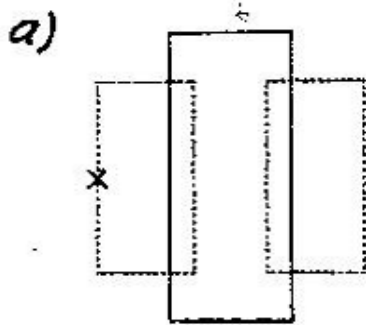
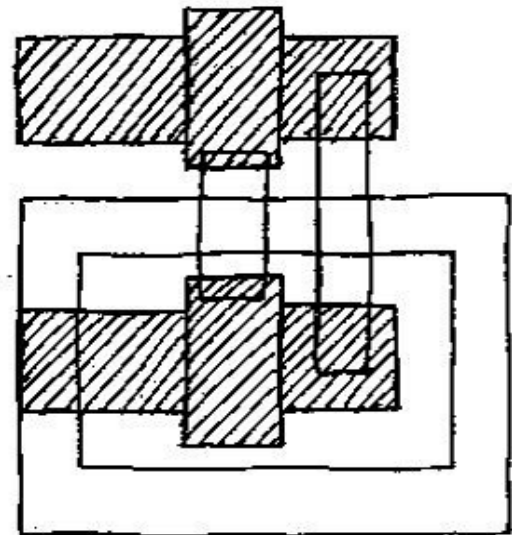
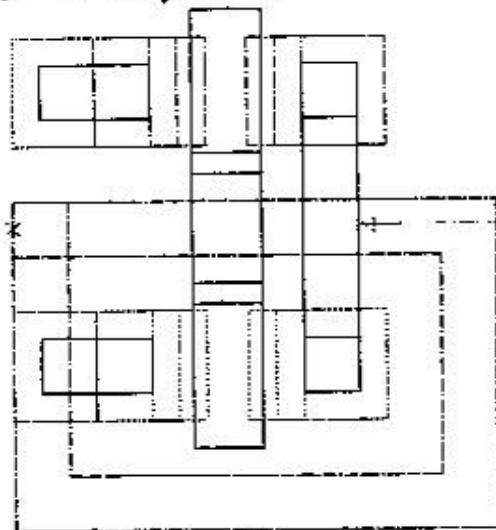


Fig. 28 Hierarchy

Displayed layers:



c) A leaf cell



d) A composition cell

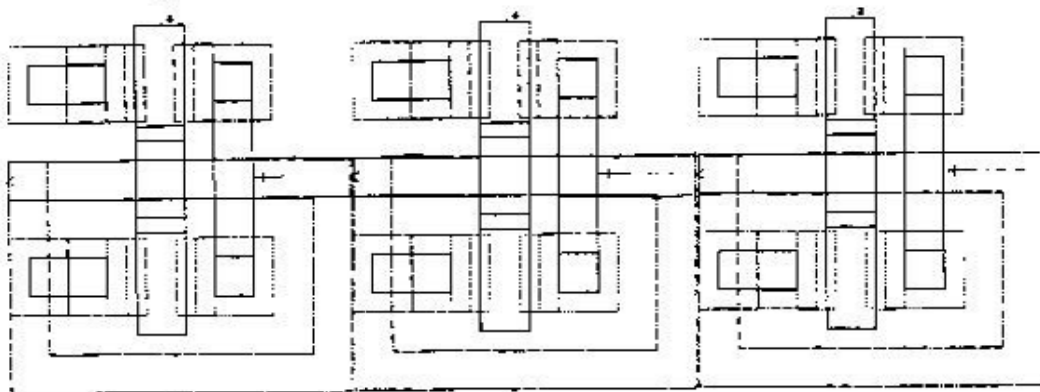
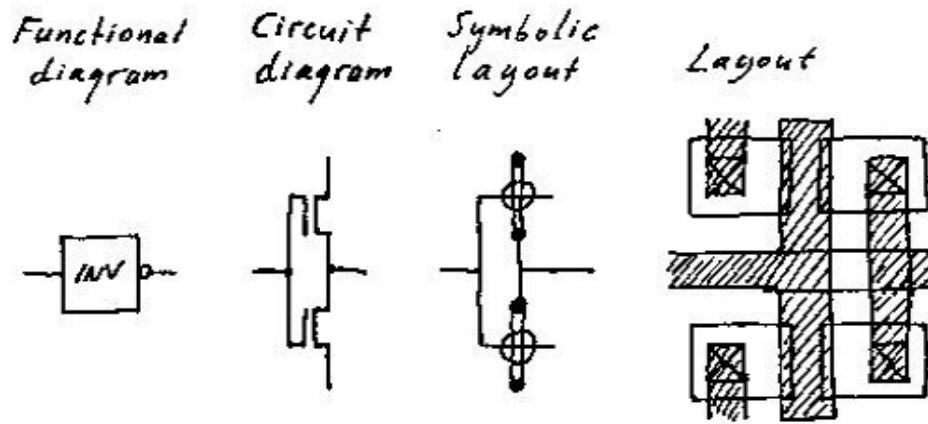


Fig. 29 Hierarchy in layout

Leaf cell descriptions



Composition cell descriptions

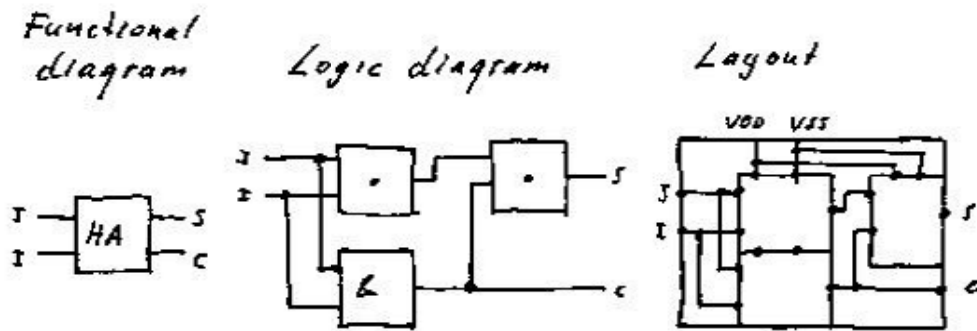


Fig. 30 Examples of cell descriptions

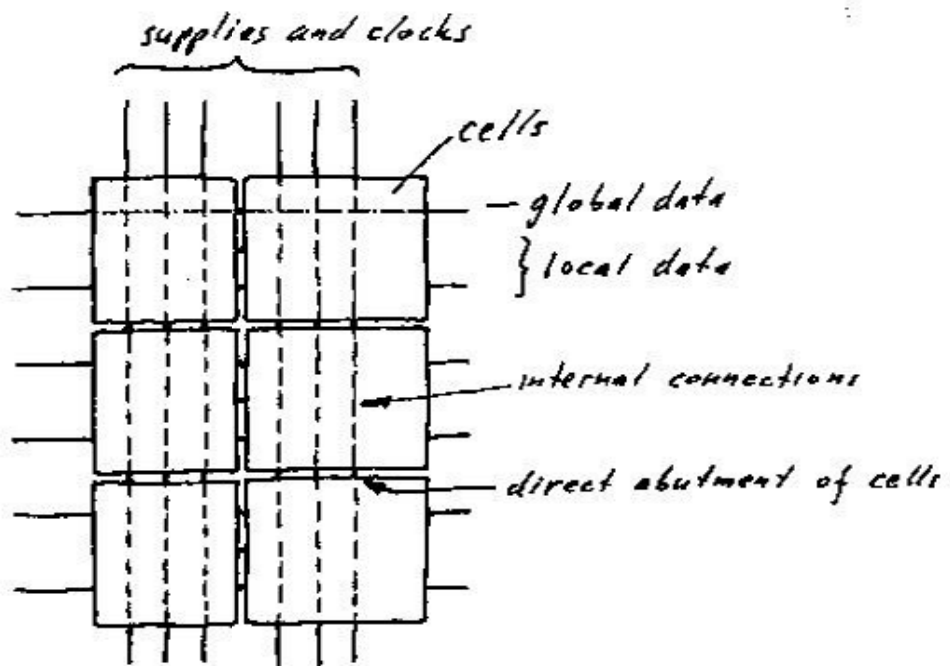


Fig. 31 Regular wiring.

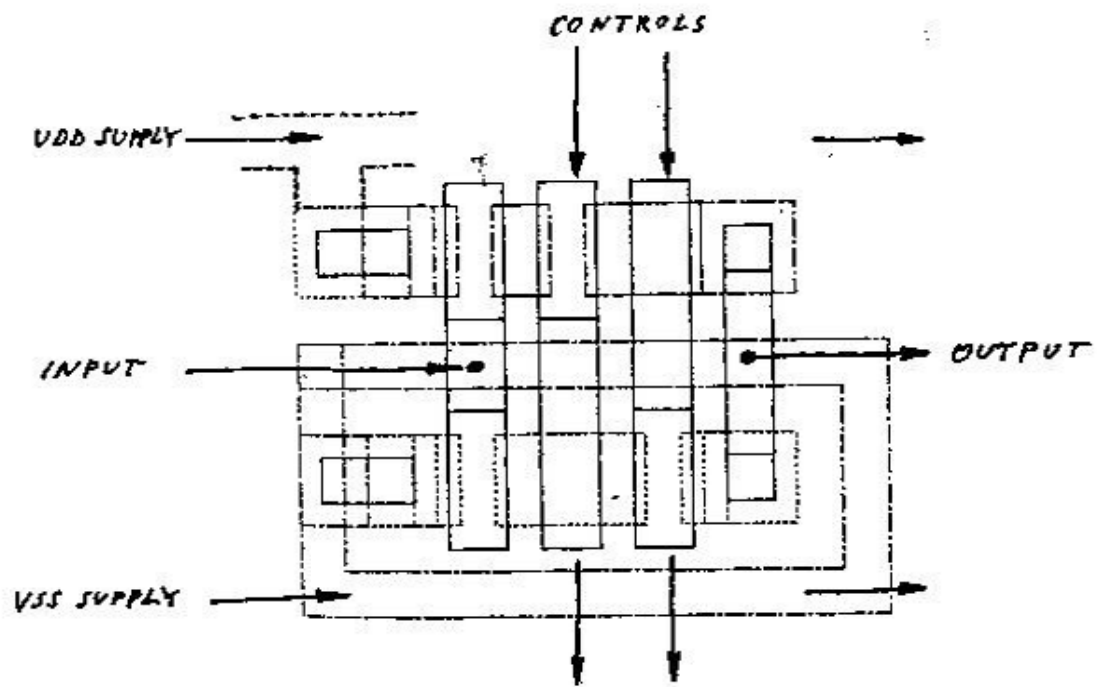


Fig. 32 Tristate inverter from CLIB2.

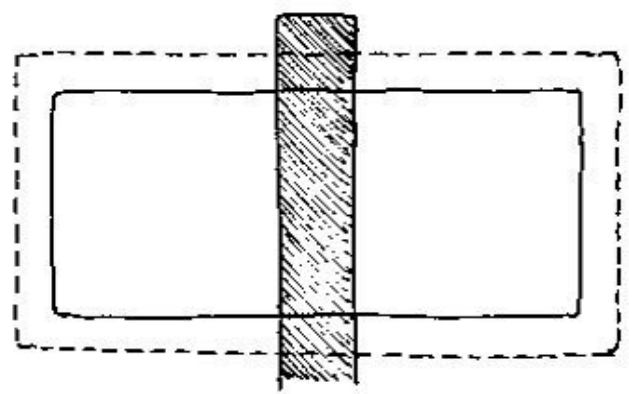
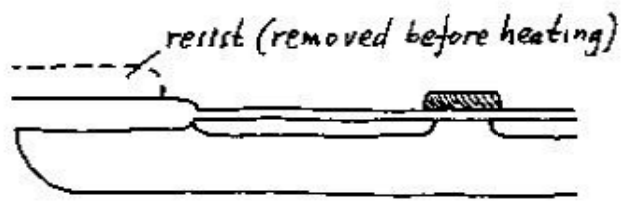
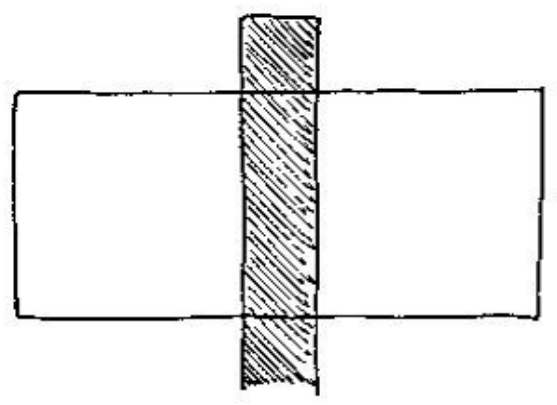
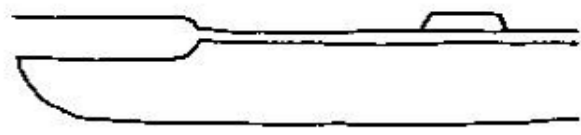
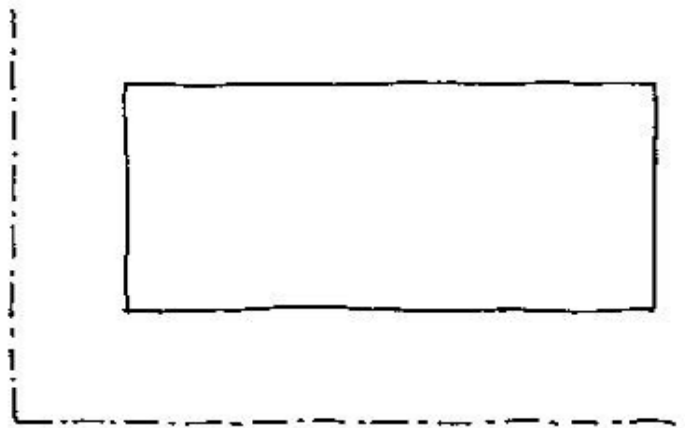
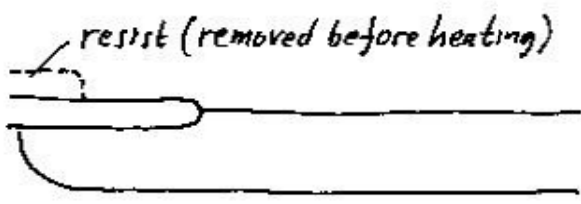


Fig. 33

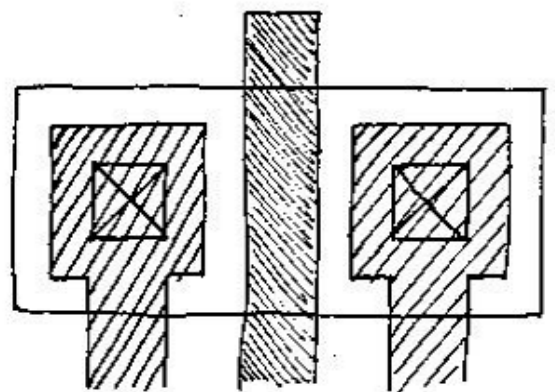
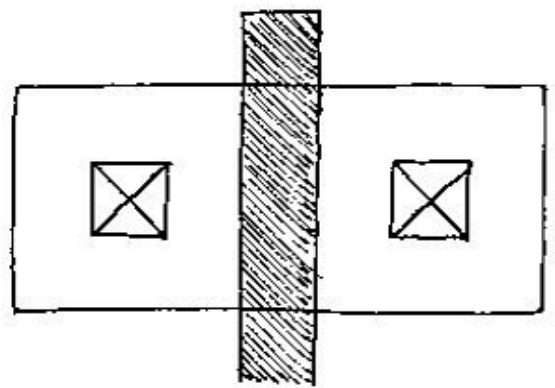
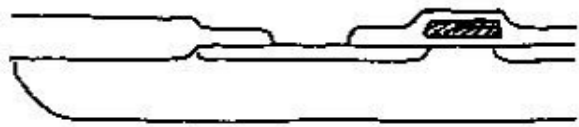
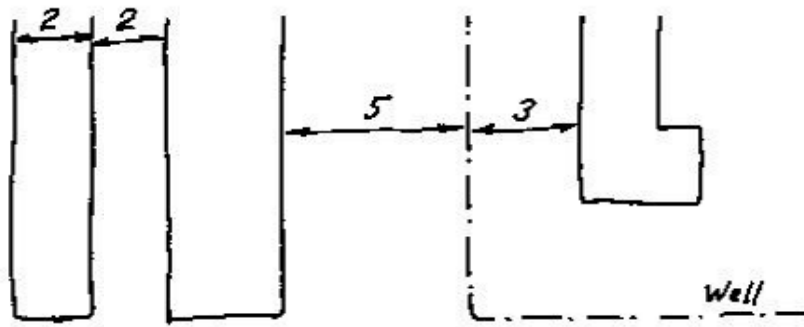
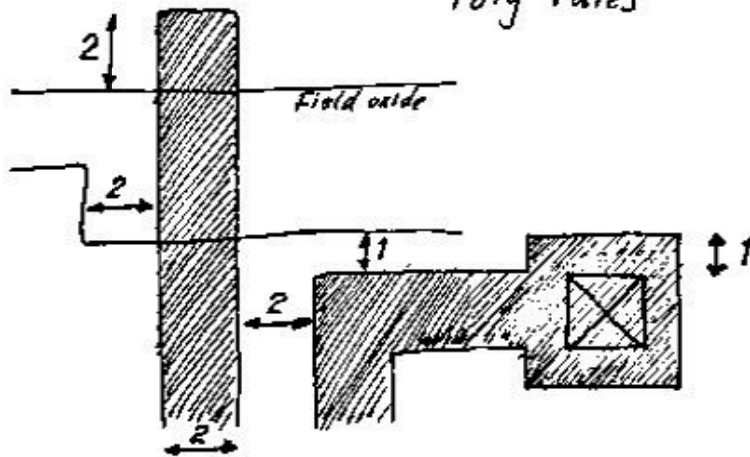


Fig 33 contd.

Field oxide rules



Poly rules



Contact cut rules

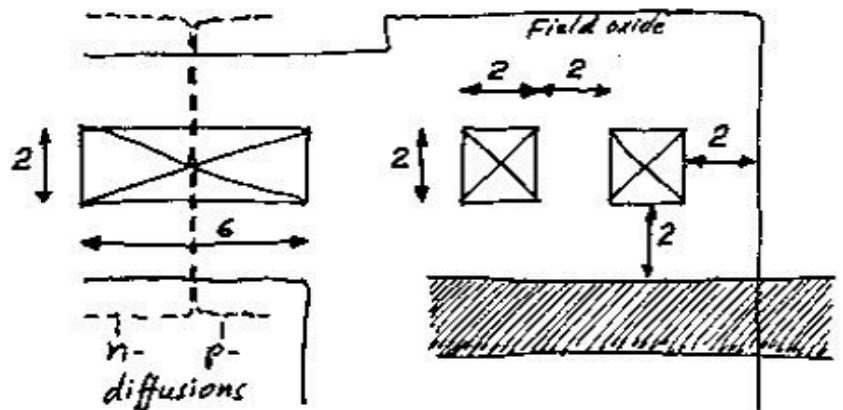


Fig. 34

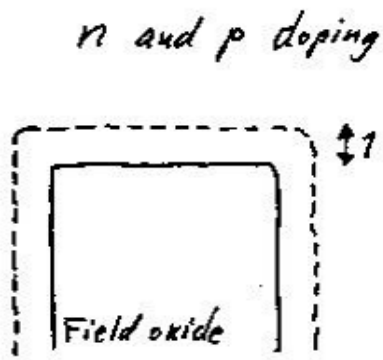
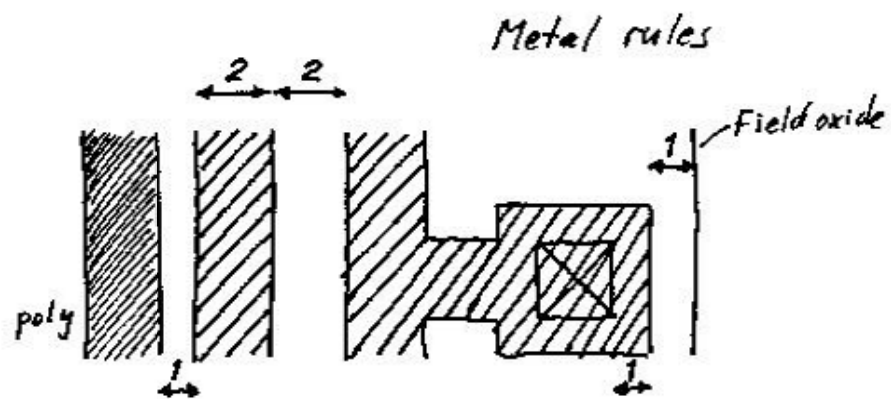
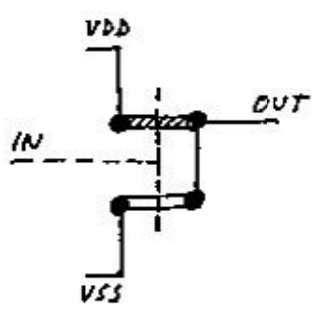
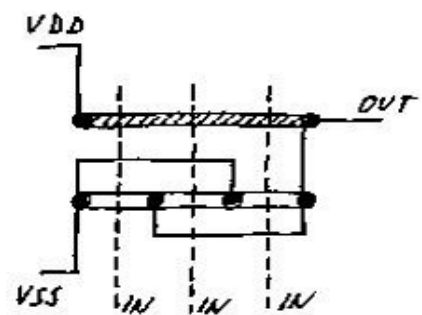


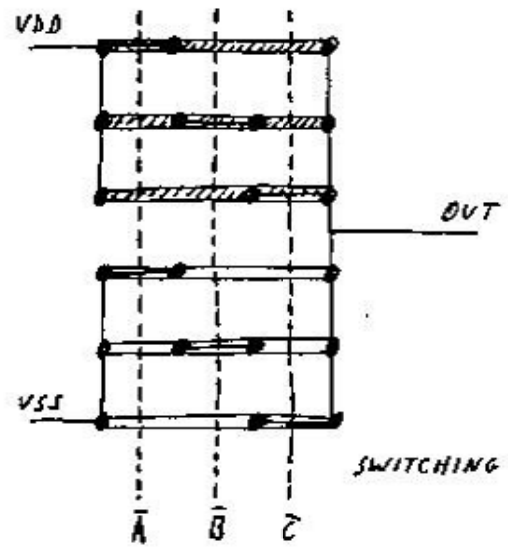
Fig 34 contd



INVERTER

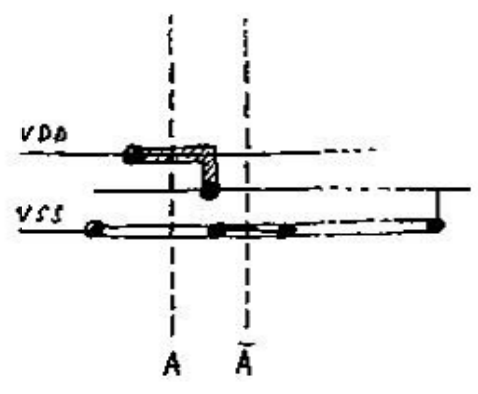


NOR-GATE



SWITCHING NETWORK

- Metal
- - - - Poly
- ▨▨▨▨ p-doping
- ▬▬▬▬ n-doping
- contact cut



PLA cells

Fig. 35