THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 57, No. 5, May–June 1978
Printed in U.S.A.

Fault Modeling and Logic Simulation of CMOS and MOS Integrated Circuits

By R. L. WADSACK

(Manuscript received October 10, 1977)

This paper addresses the simulation and detection of logic faults in CMOS integrated circuits. CMOS logic gates are intrinsically tri-state devices: output low, output high, or output open. This third, high-impedance condition introduces a new, nonclassical logic fault: the "stuck-open." The paper describes the modeling of this fault and its complement, the stuck-on, by means of gate-level networks. In addition, this paper provides a methodology for creating simulator models for tri-state and other dynamic circuit elements. The models are gate-level in structure, provide for both classical and stuck-open/stuck-on faults, and can be adopted for use on essentially any general purpose logic simulator.

I. INTRODUCTION

The challenge of testing silicon integrated circuits (ICs) is becoming more formidable with the rapidly expanding production of large-scale integrated (LSI) circuits. Increased gate-count, increased pin-count, smaller feature size, higher performance, and higher complexity all contribute to a mounting "testability" problem. Furthermore, there is considerable evidence that the economic requirements to meet that challenge will continue to grow at a rate markedly greater than that of circuit size alone.

As a further dimension to the challenge, IC tests must be specifically designed to recognize failure-mode dependence upon circuit configuration, processing parameters, and the overall technology (TTL, ECL, PMOS, CMOS, etc.). That is, a Boolean network realized in one technology can have a strikingly different implementation in another. Consequently, logic tests must be created which exercise not only the gross functional behavior of the IC but also the structure used for that function. However, for large-scale ICs, internal circuit structure and complexity are in-

creasing at a much more rapid rate than is the number of access terminals.

The rising use of MOS technology for LSI circuits has introduced a number of circuit elements whose logical behavior and faults are generally not treated by existent logic simulators.[1] These include, for example, transmission gates, tri-state inverters, and bidirectional buses. Furthermore, the failure modes of such circuits and even those of ordinary combinational logic gates can introduce nonclassic logic faults. That is, they possess a faulted behavior for which test coverage would not be verified on a conventional fault simulator.

The focus of this paper will be centered upon fault modeling and logic simulation of CMOS digital integrated circuits. The motivation for this direction is the recent emergence of CMOS as a mature technology for the design of densely-packed, low-power digital LSI circuits.[2] Secondly, CMOS is intrinsically a three-state logic technology. Consequently, it readily lends itself both to the illustration of the new, nonclassical logic faults and to a methodology for modeling the dynamic nature of MOS circuits.

II. CMOS LOGIC FAULTS

2.1 CMOS logic gates

Figure 1 shows a two-input CMOS NOR gate: the output is high if and only if $A = B = 0$. The realization of the NOR function shows the series/parallel complementary nature of CMOS logic gates: $F = \overline{A} \cdot \overline{B}$ and $\overline{F} = A + B$ where the Boolean function $\overline{A} \cdot \overline{B}$ connects the output to the "1" level and the function $A + B$ connects the output to the "0" level. Each is the complement of the other and is implemented, respectively, with p-channel FETs and n-channel FETs.

The NOR circuit is a specific example of the general CMOS characteristic of complementary pull-up/pull-down networks. The only two steady-state logic outputs are 0 and 1. The former arises when the pull-down network is conducting and the pull-up network is nonconducting. The latter, $F = 1$, occurs when the two networks reverse their conductivity states. Consequently, there is no static current path between VDD and VSS, and CMOS ICs therefore dissipate power only to charge and discharge circuit capacitance.

On the other hand, there are two common situations which can lead to a third logic state. This third condition is the "open" or high-impedance state.[3] One source of the "open" state is the presence of a logic fault which prevents one network from conducting when the other is in a nonconductive state. A second cause is the legitimate use of a high-impedance state in dynamic circuits or tri-state buffers, for example. In each instance the output retains the logic value of the previous output

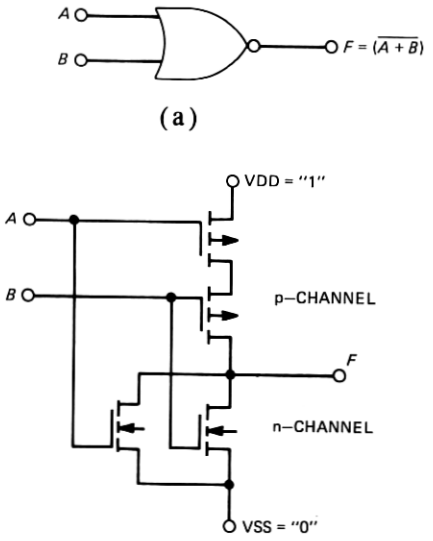


Fig. 1—The CMOS two-input NOR gate: (a) logic symbol and transfer function, and (b) FET realization.

state. This is true because the gates are loaded with capacitance only. The length of time the state is retained, however, is determined by the leakage current at the node.

Conceptually, a fourth state could exist: both networks conducting. However, this represents a logical inconsistency, i.e., the output cannot simultaneously be both high and low. Such a fault is more analog in nature because the output voltage lies somewhere between VDD and VSS. The actual value is determined by the impedance ratios of the networks and the associated fault. For the most part these failures will not be treated as logic faults.

There are two kinds of *classical* logic faults: stuck-at-one (SA1) and stuck-at-zero (SA0). These faults may occur at either an input or an output of a logic gate. On the other hand, a gate with n inputs can have only $n + 2$ distinct classical faults. Input faults must be stuck in the "nondominant" state to be distinguishable from an output stuck-at fault. For AND/NAND gates and OR/NOR gates such nondominant faults are SA1's and SA0's, respectively. These faults are sometimes called "input-open-from" faults and will be denoted as IOF faults in this paper. The two-input NOR gate of Fig. 1 has the four classical faults: SA0, SA1, IOFA, and IOFB. These are also symbolized as $F(0)$, $F(1)$, $F(A)$, and $F(B)$, respectively.

For CMOS logic gates the nonclassical "stuck-open" faults must be

Table I — CMOS two-input NOR gate: truth table

A	B	F	F (0)	F (1)	F (A)	F (B)	F (ASOP)	F (BSOP)	F (VDDSOP)
0	0	1	0	1	1	1	1	1	4
0	1	0	0	1	0	1	0	4	0
1	0	0	0	1	1	0	4	0	0
1	1	0	0	1	0	0	0	0	0
normal			classical faults				nonclassical faults		

4 = previous output state
$F(A) = F(\text{IOFA})$
$F(B) = F(\text{IOFB})$

included to represent the undesired, high-impedance state caused by a faulty pull-up or pull-down network. For the two-input NOR gate of Fig. 1, there are three such stuck-open (S-OP) faults: ASOP, BSOP, and VDDSOP. The first, ASOP, is caused by an open, or missing, n-channel A-input pull-down FET. The second, BSOP, is caused by an open, or missing, B-input pull-down FET. The third, VDDSOP, is caused by an open anywhere in the series, p-channel pull-up connection to VDD.

Table I shows the truth table for the two-input CMOS NOR gate for both the fault-free and the seven faulted conditions. For example, the fault-free gate obeys

$$F = \overline{(A + B)}$$

whereas

$$F(\text{IOFB}) = \overline{A}$$

and

$$F(\text{ASOP}) = \overline{A} \cdot \overline{B} + 4 \cdot A \cdot \overline{B},$$

where "4" denotes the previous state of F. [Using the notation of sequential circuit design, the latter equation would read $F_{n+1}(\text{ASOP}) = \overline{A} \cdot \overline{B} + F_n \cdot A \cdot \overline{B}$. The use of a "4" to symbolize F_n is a convention adopted to describe the effect of S-OP faults.]

How are the seven NOR gate logic faults related to actual physical flaws in the IC? The SA0, SA1 faults correspond to a low-impedance "short" to VSS or VDD, respectively. The IOF faults are caused by an open input to the *logic gate* as a whole. In addition to being open, the input is in a charged condition which is recognized as a logic 0. (An IOF fault in a NOR gate is an SA0, by definition; in a NAND gate the analogous fault would be an SA1, of course.) That is, both the p-channel and the n-channel FETs have a 0 applied to them. On the other hand, the nonclassical S-OP faults arise from a missing connection to the gate of *individual* FETs, for example, with the gate in a charge state such that the FET is nonconducting. Another cause of an S-OP fault is an open, or missing, connection to either the source or the drain of an FET.

Table II — CMOS two-input NOR gate: fault detection (test sequence: AB = 00,01,00,10)

	A	B	F	F (0)	F (1)	F (A)	F (B)	F (ASOP)	F (BSOP)	F (VDDSOP)
1	0	0	1	0*	1	1	1	1	1	3
2	0	1	0	0	1*	0	1*	0	1*	0
3	0	0	1	0	1	1	1	1	1	0*
4	1	0	0	0	1	1*	0	1*	0	0
	normal			classical faults				nonclassical faults		

3 = unknown output state (0 or 1).
* Vector at which simulator detects the fault.

In this context an "open" denotes an undesired high impedance at either the gate, the source, or the drain of an FET. Of course, any residual capacitive or resistive coupling must be negligibly small for a high-impedance fault to be regarded as a true "open." In addition, the actual occurrence of such faults depends on the specific topology of the logic gate.

The truth table of Table I shows that S-OP faults create sequential circuits where only a combinational circuit existed for the fault-free gate. This increases the difficulty of both testing the circuit and designing a set of input test "vectors" to achieve a high percentage of fault coverage. For example, the four input states of Table I, if applied in that order, will detect only 5 out of the 7 logic faults. ASOP and VDDSOP will be undetected. The ASOP fault is not detected because $F(10)$ = "4" = 0 which is the correct output for the "10" input vector. The VDDSOP fault will not necessarily be detected because of the chance that the gate powers up with the output in the high (but correct) state for the "00" input vector.

The primary reason that the above two faults were undetected is that the corresponding circuit paths (devices) were not tested to determine whether they fulfilled their most basic function. For example, to test the A-input n-channel *pull-down* FET, the output node must first be driven high and then that FET, and it alone, must be capable of pulling the node low. The sequence of inputs in Table I did not meet that condition.

A set of vectors which detects all 7 faults is the following: 00,01,00,10. Table II shows the response of the 8 circuits (the good circuit and the 7 faulty ones) to the above input sequence. The * symbol designates the vector at which each fault is first detected by the *simulator*. The "3" denotes an unknown state (either a 1 or a 0), caused by the VDDSOP fault as described previously. Depending upon the actual state of the circuit at power-up, the VDDSOP fault may, therefore, be detected at either vector 1 or 3.

The influence of CMOS faults on fault coverage is shown in Fig. 2. The

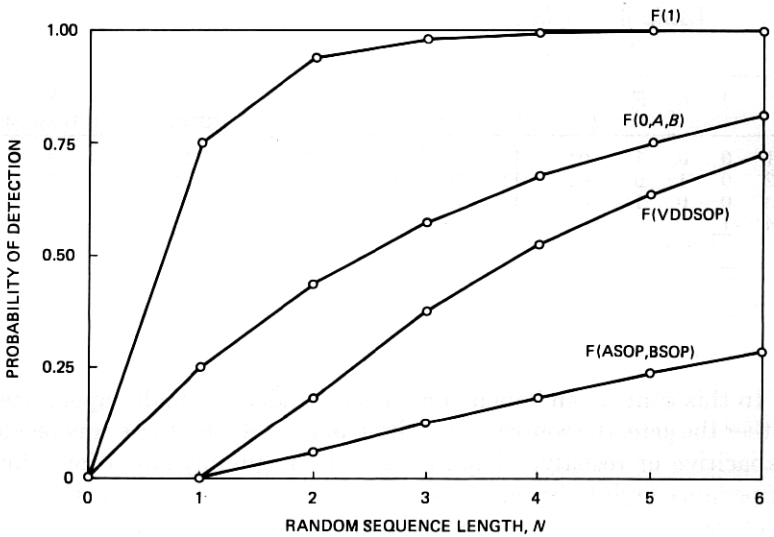


Fig. 2—The CMOS two-input NOR gate: probability of fault detection (for a simulator) versus the length N of a sequence of random input vectors.

probability of detection for random vector sequences is given for each of the 7 faults in Table I. The probability is that which would prevail for a logic simulation of a two-input NOR gate subjected to random vector sequences of length N. The output SA1 fault ($F(1)$) has a probability of detection of nearly unity for sequences of length greater than three. The two pull-down stuck-open faults have the lowest probability, reaching only 0.29 for sequences 6 vectors long (45 vectors are required to reach 0.95). Note also the similarity between the VDDSOP fault and the output SA0 fault ($F(0)$). Further, the lag for all three CMOS faults is evident.

In addition to increasing the number and complexity of CMOS logic faults, stuck-open faults are also timing-sensitive. Specifically, the above set of input vectors will detect three S-OP faults of the NOR gate only if they are applied to the gate at a rate more rapid than that associated with leakage current time-constants. A rate significantly slower than, say, 10 kHz may allow some faulty devices to charge to the correct state before the output is sampled by the test set. Truth-table testing at quasi-dc rates is inadequate. Conversely, an ill-chosen vector set, such as the binary sequence of Table I, may not detect S-OP faults no matter how fast it is applied to the device under test. Note also that the sequence of Table I is "exhaustive," but does not achieve 100 percent fault coverage.

Another aspect of stuck-open faults is that of long-term reliability. A fault caused by a missing connection to the gate of an individual FET may cause that FET to be open and yet remain undetected during production testing if the test vector set has less than 100 percent fault cov-

erage. Later, however, under actual operating conditions the FET gate may acquire charge of the opposite polarity and become conducting or "stuck-on." CMOS S-OP field failures have been observed.[4]

2.2 Modeling CMOS logic faults

The design of a set of vectors to achieve 100 percent fault coverage for small-scale integrated (SSI) circuits such as NAND and NOR gates is trivial and can be done by inspection. However, for complex high gate-count circuits such as medium- and large-scale ICs (MSI, LSI), the use of computer-based logic simulators is a necessity. To meet this need the simulation of both stuck-open faults and dynamic logic elements has been approached from the standpoint of circuit modeling. The models represent circuit elements both in their fault-free condition and in high-impedance state(s), if any. Combinational "static" gates (e.g., NAND or NOR) enter the high-impedance condition only in the presence of S-OP faults. Dynamic or tri-state logic elements, however, can intentionally be placed into a high-impedance state by auxiliary, or control, inputs.

The models presented in this paper are gate-level oriented because no other method generally exists for simulating nonclassical logic faults. On the other hand, the logical behavior of the models can be incorporated into a higher-level functional description if that capability is available on the logic simulator. On the LAMP system[1] the Function Description Language (FDL) is being modified to include such "internal faults." [5] Although the models in this paper are implemented in terms of NAND/NOR logic, they do have the advantage of being simulator independent. That is, they will correctly model fault-free and faulted logic networks regardless of the particular simulator chosen. The specific illustrations, however, are taken from the author's experience with LAMP.

Existent machine aids simulate for the most part only the classical SA0/SA1 faults and not the "stuck-open" faults. One possible solution to the problem is to use a network of conventional gates (NOT, NAND, NOR, etc.) to form a model which duplicates both the normal and the faulted behavior of a single CMOS gate. One of the basic properties of the model is that it must possess the capability of passing 0/1 data from input to output in accordance with the fault-free logic function of the gate. Second, when there is an S-OP fault, it must retain the previous state in the presence of the "provoking" input (see Table I). This suggests the use of gated latch.

The general approach to modeling either stuck-open (S-OP) faults or dynamic gates is shown in Fig. 3. The "gated latch" represents the nodal capacitance associated with the logic function. For $T = 1$, the output equals the input ($Z = D$); the $T = 0$, the output latches and stores the previous state ($Z =$ "4"). The "node faults gate" has been added to introduce the two classical SA0 and SA1 (stuck-at) faults. The gate is not

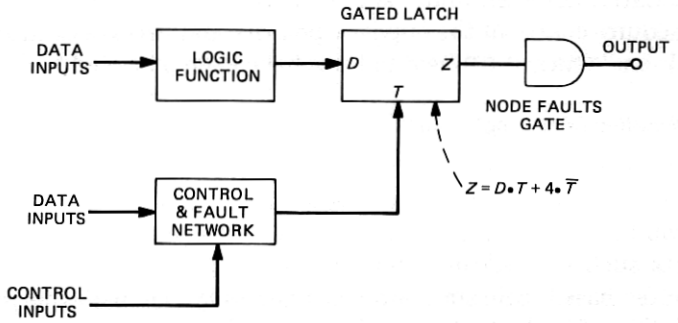


Fig. 3—Block diagram: the general approach for stuck-open/dynamic gate models.

necessary if those faults are incorporated within the "logic function" network. The latter represents the actual logical operation being modeled (NAND, NOR, etc.). The "control and fault network" establishes whether the output of the "logic function" is to be transmitted to the gates driven by the "output." In the case of static gates, there are no "control inputs" to that network. Only the presence of S-OP faults would cause $T = 0$ (i.e., no transmission). Dynamic circuit elements have "control inputs" which can cause $T = 0$ in the absence of faults. Classical faults generated in the "logic function" network at input D will, of course, propagate through the latch (and not through the S-OP fault generating network).

III. SPECIFIC MODELS

3.1 The NOR gate

The model for the two-input CMOS NOR gate is shown in Fig. 4. The NOR gate named GATE has zero delay but is faulted (i.e., the fault simulator will simulate faults for this gate) so as to introduce all the classical faults of a two-input NOR gate. Of course, GATE represents the modeled logic function. All other gates in Fig. 4 compose the fault-generating network which sets $T = 0$ in the presence of an S-OP fault *and* the "provoking" input for that fault.

The model functions as follows: If there are no faults in the circuit, then $F = \overline{(A + B)}$. If there are only classical faults (from GATE), then $T = 1$ and those faults propagate through GL. If there is an S-OP fault in the gate, than $T = 0$ for the provoking vector and $F = 4$ as required from Table I. For example, because its input is grounded the only fault that the simulator assigns to the gate ASOP is an SA1. Therefore, in the presence of the fault ASOP(1) and when $AB = 10$, then $T = 0$ and the GL latch holds the output equal to the previous value. Similarly the output F will be stuck-open in the presence of the VDDSOP(1) fault if and only if $AB = 00$.

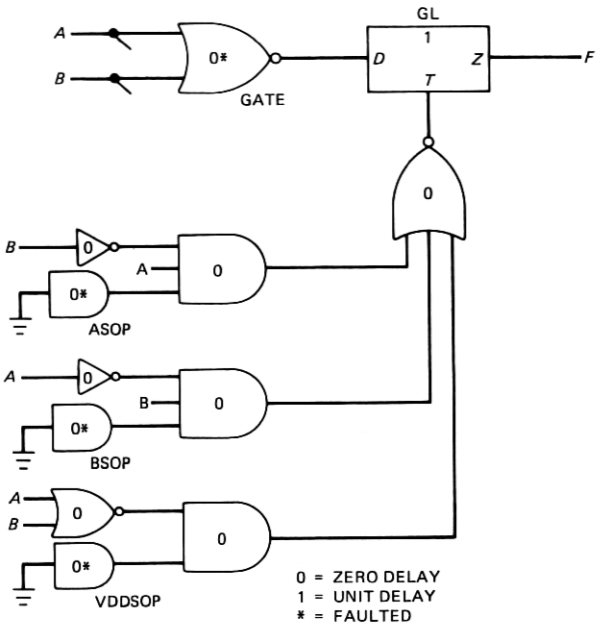


Fig. 4—The CMOS two-input NOR gate model, NR2. The block labeled GL is the gated latch circuit of Fig. 3. Only those gates marked * are faulted by the simulator. Propagation delays are denoted by either 0 or 1.

The NOR gate model of Fig. 4 has been designed to yield the correct logic behavior of the two-input CMOS NOR gate in the absence of faults. Secondly, it correctly models all 7 logic faults: SA0, SA1, IOFA, IOFB, ASOP, BSOP, and VDDSOP. Thirdly, it generates no spurious faults associated with the modeling, nor does it have a propagation delay other than the one unit that would be expected from a single NOR gate. Finally, the nature of the fault-generating network prevents the spurious propagation of faults from other gates connected to the inputs of the gate in question.

The names of the gates in the model have been chosen to provide ease of use during simulation. Specifically, if a circuit contained a two-input CMOS NOR gate with the name NOR17, then the LAMP simulator would assign the following fault list to it:

NOR17.GATE(0)
NOR17.GATE(1)
NOR17.GATE(A)
NOR17.GATE(B)
NOR17.ASOP(1)
NOR17.BSOP(1)
NOR17.VDDSOP(1)

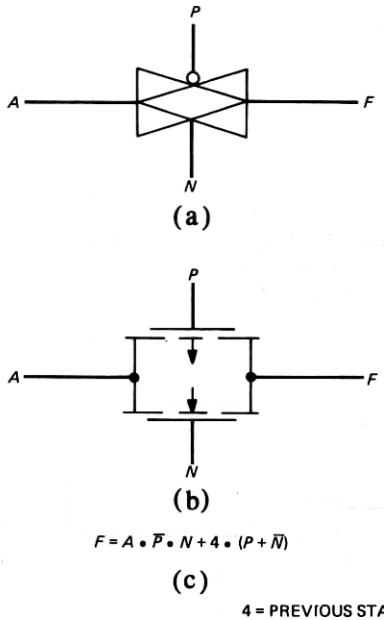


Fig. 5—The CMOS transmission gate: (a) logic symbol, (b) FET realization, and (c) the static, unilateral logic transfer function.

Consequently, the test engineer can determine at a glance the nature of a fault and where it is located. In addition, in the LAMP system all S-OP faults can be globally nonfaulted (or faulted) because they arise from gates ending in the string "SOP." This nonfaulting capability is useful in determinations of relative fault coverage.

3.2 The transmission gate

MOS technology possesses an interesting circuit element with both digital and analog capability: the transmission gate. The CMOS transmission gate is shown in Fig. 5.[6] It is a *bilateral* device with a conducting mode for PN = 01 and a nonconducting state for PN = 10. A problem occurs for the two other vectors PN = 00 and PN = 11. In each case either the p-channel or the n-channel FET will be on, but not both. Low-to-high transitions (open p-FET) or high-to-low transitions (open n-FET) will be attenuated as they pass through the transmission gate. In addition, there will be an accompanying speed degradation caused by the higher impedance of the single FET path.

Although the two input vectors 00 and 11 apply abnormal conditions to the transmission gate, the choice of whether to regard the gate as a whole as either "on" or "off" is somewhat arbitrary. For these particular

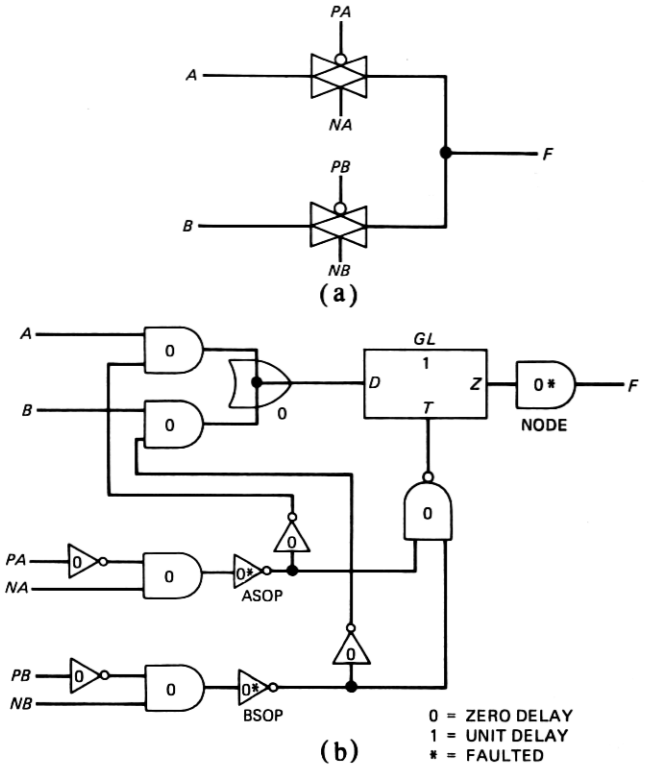


Fig. 6—The CMOS two-channel transmission gate node, XG2: (a) logic symbol and (b) a gate-level model which exhibits both the fault-free and the faulted behavior of the XG2 element. Signal flow is unilateral: from A and B to F. Normally only one channel is enabled at one time.

models the worst-case logic behavior of the gate has been taken to be

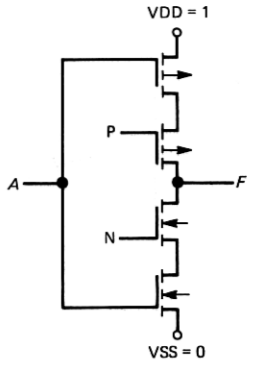
$$F = A \cdot \overline{P} \cdot N + 4 \cdot (P + \overline{N})$$

where "4" represents the previous state of the output. The above expression can be rewritten as

$$F = D \cdot T + 4 \cdot \overline{T}$$

where $T = \overline{P} \cdot N$ and $D = A$. This is just the equation for the gated latch of Fig. 3, i.e., the gated latch is equivalent to a nonfaulted transmission gate.

Functionally, transmission gates generally occur in groups of two, three, four, etc. Hence, they can be regarded as multiplexing nodes at which usually only one of the gates is conducting at a time. The model for the CMOS two-channel transmission gate node, used in the above sense, is shown in Fig. 6. The gate named SUM is a zero-delay nonfaulted



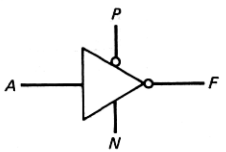
(a)

A	P	N	F
0	0	0	1
0	0	1	1
0	1	0	4
0	1	1	4
1	0	0	4
1	0	1	0
1	1	0	4
1	1	1	0

(4 = PREVIOUS STATE)

$F = \bar{A} \cdot \bar{P} + 4 \cdot (\bar{A} \cdot P + A \cdot \bar{N})$

(b)



(c)

Fig. 7—The CMOS tri-state inverter: (a) the FET realization, (b) the truth table, and (c) the logic symbol.

tied-OR node (TOR). The purpose of the gate labeled NODE is to provide the two classical faults SA0, SA1 associated with the node F. The nonclassical faults generated by this network are four in number: ASOP(0), ASOP(1), BSOP(0), and BSOP(1). As usual, ASOP(1) means that the A-channel is stuck-open and, conversely, ASOP(0) means that it is "stuck-on" (S-ON). For the latter fault, $F = A + B \cdot (\overline{PB}) \cdot (NB)$. Here, the stuck-on has been treated as a legitimate logic fault whose presence induces a "1-dominant" short, i.e., the spurious A-input is ORed with the correct B-channel response. Of course, there may be no technological reason to assign a S-ON fault as either 1-dominant or 0-dominant. In that case, the model in Fig. 6 (and others) can easily be recast to exhibit only S-OP nonclassical faults.

Stuck-open faults in a CMOS transmission gate are also timing-sensitive as in NAND and NOR gates, but in a different manner. Specifically, even with one FET of the pair open, the other can provide nearly a complete logic transition but at a higher average impedance level. Consequently, 0/1 data will propagate through the gate but at slightly slower speeds. However, the reduction in speed is much smaller than that caused by S-OP faults in NAND or NOR gates. Therefore, an S-OP fault in a transmission gate may be intrinsically undetectable even at the highest data rates that occur at a particular gate.

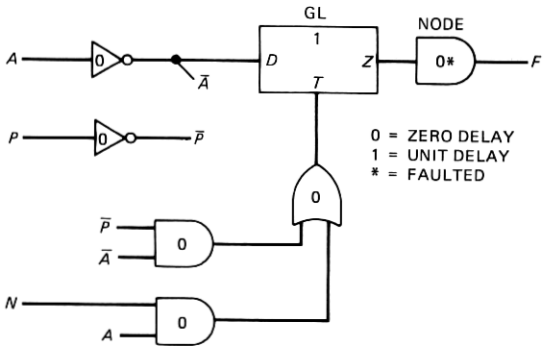


Fig. 8—The CMOS tri-state inverter model. Only the gate marked * is faulted by the simulator. Propagation delays are denoted by either 0 or 1.

3.3 The tri-state inverter

The FET implementation of the CMOS tri-state inverter is shown in Fig. 7(a). In this case P and N are control leads that determine whether the circuit inverts the input A or remains in the high-impedance ("4") state. For the latter condition, the nodal capacitance retains the value of the previous state ("0," "1," or "3"). The truth table is given in Fig. 7(b) and leads to the transfer function: $F = \overline{A}\cdot\overline{P} + 4\cdot(\overline{A}\cdot P + A\cdot\overline{N})$. The logic symbol is shown in Fig. 7(c).

The model for the tri-state inverter is given in Fig. 8. Here, only the two classical faults $F(0)$ and $F(1)$ have been assigned to the model because of the similarity of VDDSOP/VSSSOP faults to $F(0)$ and $F(1)$ faults, respectively. The S-OP and S-ON faults could be modeled, of course, but the added complexity does not warrant separate treatment.

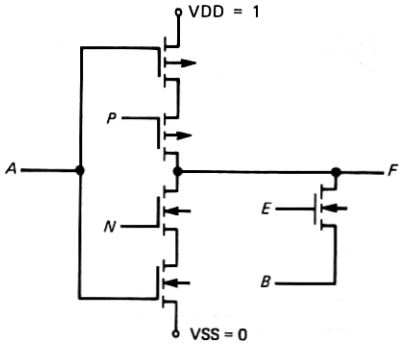
The relation of the model network to the truth table is perhaps more obvious if the transfer function is written as

$$F = 1\cdot\overline{A}\cdot\overline{P} + 0\cdot A\cdot N + 4\cdot(\overline{A}\cdot P + A\cdot\overline{N}).$$

The terms $\overline{A}\cdot\overline{P}$ and $A\cdot N$ then compose the *t*ransmit (T) function: $T = \overline{A}\cdot\overline{P} + A\cdot N$. Therefore, when $T = 0$ then $F =$ "4." That is, the output latches (or stores) the previous state.

3.4 Modified tri-state inverter

The addition of an n-channel transmission gate FET to the output of the tri-state inverter forms a "modified" inverter of Fig. 9(a). (This circuit element forms one half of a gated sense amplifier.) The resultant truth table is shown in Fig. 9(b). The "3" (unknown 0 or 1) state occurs whenever $A = B$ and both the inverter and the transmission gate are enabled. In other words, whenever each attempts to drive the node F to



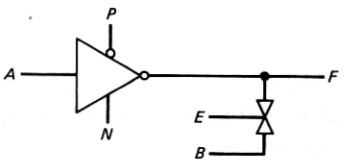
(a)

			E = 0	B = 0	B = 1
A	P	N	F	F	F
0	0	0	1	3	1
0	0	1	1	3	1
0	1	0	4	0	1
0	1	1	4	0	1
1	0	0	4	0	1
1	0	1	0	0	3
1	1	0	4	0	1
1	1	1	0	0	3

E = 1

(3 = UNKNOWN STATE)
(4 = PREVIOUS STATE)

(b)



(c)

Fig. 9—The CMOS modified tri-state inverter: (a) the FET realization, (b) the truth table where 3 indicates an unknown 0 or 1 state and 4 symbolizes the previous state of F, and (c) the logic symbol for the combination tri-state inverter and n-channel FET.

opposing logic states, the output is indeterminate ("3"). The high impedance "4" state occurs whenever both channels are disabled.

The model for the modified tri-state inverter can be developed by noting that the Boolean expression that selects the $\overline{A}$ channel is $\overline{P}\cdot\overline{A} + N\cdot A$ and the term that selects the B channel is E [see Fig. 9(a)]. Therefore, define $\text{SA} = \overline{P}\cdot\overline{A} + N\cdot A$ and $\text{SB} = E$. Then, $T = \text{SA} + \text{SB}$ or $T = \overline{\text{SA}}\cdot\overline{\text{SB}}$. Therefore,

$$F = \overline{A}\cdot\text{SA}\cdot\overline{\text{SB}} + B\cdot\overline{\text{SA}}\cdot\text{SB} + 3\cdot\text{SA}\cdot\text{SB} + 4\cdot\overline{\text{SA}}\cdot\overline{\text{SB}}$$

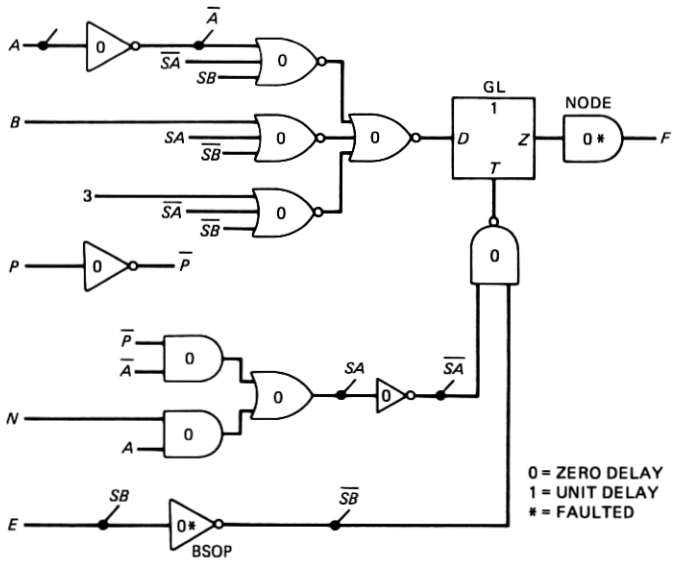


Fig. 10—The CMOS modified tri-state inverter model.

or

$$F = (\overline{A} + \overline{\text{SA}} + \text{SB}) \cdot (B + \text{SA} + \overline{\text{SB}}) \cdot (3 + \overline{\text{SA}} + \overline{\text{SB}}) \cdot (4 + \text{SA} + \text{SB}).$$

The model corresponding to the latter expression for F is shown in Fig. 10. The model has four faults assigned to it: two classical faults NODE(0,1), and two CMOS faults BSOP(0,1). For example, the fault BSOP(1) means that the B-channel transmission gate is stuck-open and nonconducting. S-OP faults in the inverter itself are ignored (see Section 3.3, above).

The model of Fig. 10 produces a "3" output whenever conflicting output conditions are generated by the simultaneous selection of both the A channel and the B channel. The "3" capability may be important during design verification of the fault-free circuit behavior. On the other hand, if it is known that mis-selection is unimportant or not possible, then the model can be reduced to that shown in Fig. 11.

It would be pointless to introduce into either model a fault whose sole effect would be the generation of a "3" output. For fault simulations "3" outputs are effectively ignored. A fault whose only result is to produce a "3" would be undetectable on a logic simulator.

3.5 Input/output port

Some integrated circuits contain pins which can serve as both input and output ports. Figure 12(a) shows the physical structure of one such

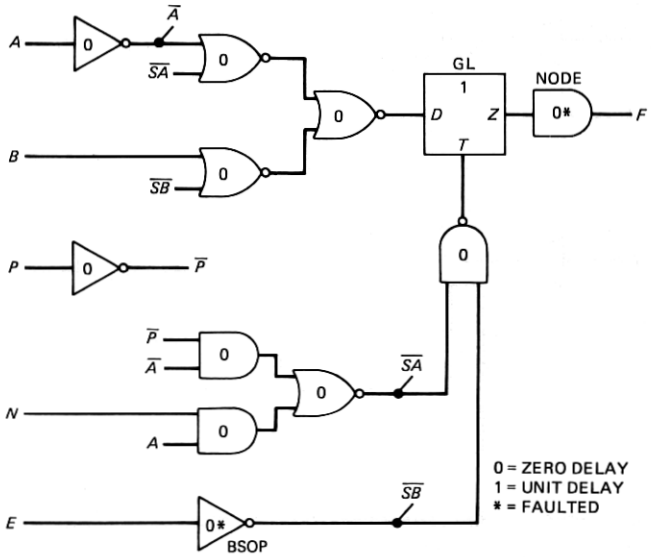


Fig. 11—The CMOS modified tri-state inverter model: reduced version. The unknown 3 condition has been ignored and replaced by a 0-dominant short assignment.

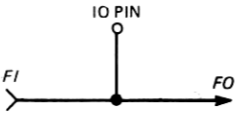
pin. IOPIN is the actual connection to the external world, FO represents all fan-out loads from the node, and FI represents all fan-in (tri-state) devices at the node. The modeled structure is shown in Fig. 12(b) where the IOPIN has been divided into separate IN and OUT functions. The D and E variables have been introduced as the control inputs which define the state of the input driver and the FI branch, respectively.

The truth table of Fig. 12(c) defines the relation between D, E, and OUT. Note that the impedance of the input driver (of the test set) is taken to be much less than that of the *FI* branch. Consequently, whenever the node is driven from an external source ($D = 1$), the logical state of the driver overrides that of FI.

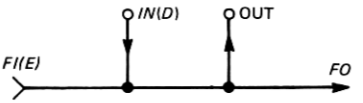
Under the above assumptions the gate-level model for the I/O port can be constructed as shown in Fig. 13. The model has been assigned four faults: the usual output SA0, SA1 faults and two S-OP/S-ON faults associated with the FI branch. Ideally each fan-in branch at the node would have two such faults. However, in the absence of a detailed knowledge of the fan-in network, just two faults have been indicated.

3.6 Tri-state bilateral buses joined by a transmission gate

The characteristics of the simple I/O port described above can be extended to more complex networks. One example is shown in Fig. 14. Two tri-state bilateral buses are connected by a bilateral transmission gate. All the data sources ("talkers") are grouped on the left and all data



(a)



(b)

D	E	OUT = FO
0	0	4
0	1	FI
1	0	IN
1	1	IN

$$OUT = IN \cdot D + FI \cdot \overline{D} \cdot E + 4 \cdot \overline{D} \cdot \overline{E}$$

(c)

Fig. 12—Input/output port: (a) physical structure, (b) modeled structure, and (c) truth table. IOPIN is the actual connection to the outside world, *FO* represents all fan-out loads, and *FI* represents all fan-in (tri-state) devices. *D* and *E* define the state of the input driver and the *FI* branch, respectively. The symbol 4 denotes the previous state of OUT.

sinks ("listeners") on the right. A device that can be both send and receive would be represented once in each group.

The models described previously in this paper have all been inde-

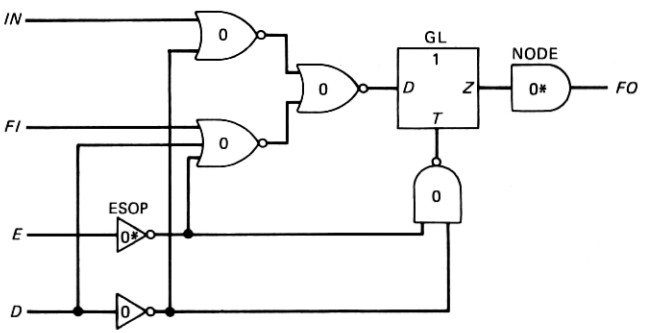


Fig. 13—Input/output port model. Only gates marked * are faulted, 0 and 1 denote delays, and *D* and *E* are enable controls for IN and FI, respectively. FI = fan-in; FO = fan-out.

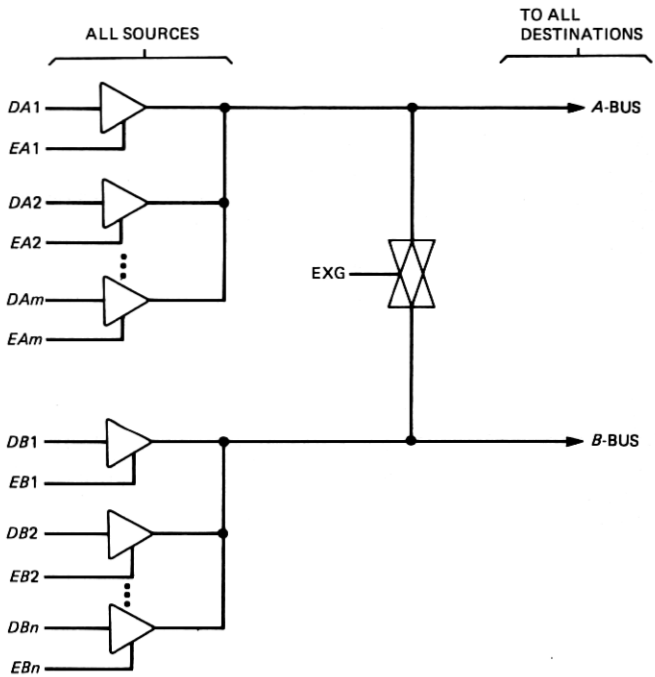


Fig. 14—Two bilateral tri-state buses connected by a transmission gate. All data sources (drivers) are grouped on the left. All data sinks (gate inputs) are to the right.

pendent of each other. Therefore, they can be organized as a library of subnetwork building blocks. Consequently, it would be tempting to model the network of Fig. 14 as an interconnection of three independent models: one for the A-bus, one for the B-bus, and one for the bilateral transmission gate. However, in this instance, that is not possible. A successful model must incorporate features from all three to correctly represent the true bilateral interactions between each bus by way of the transmission gate.

The model for the fault-free behavior of the A-bus output is shown in Fig. 15. The interbus coupling is modeled by means of the EXG·EB AND gate and the B·SB AND gate. For example, if the A-bus driving sources are in the high-impedance state (EA = 0) and the B-bus is in the low-impedance sourcing mode (EB = 1), then when the transmission gate is enabled (EXG = 1), the A-bus output takes the same value as that present on the B-bus. Conversely, information can travel from the A-bus to the B-bus. The model for the B-bus output has not been shown because it is analogous to that of the A-bus: In Fig. 15, wherever the symbol A appears substitute B and vice versa. The fault-free behavior of the bus model as compared to that of the "actual" buses is given in Table III.

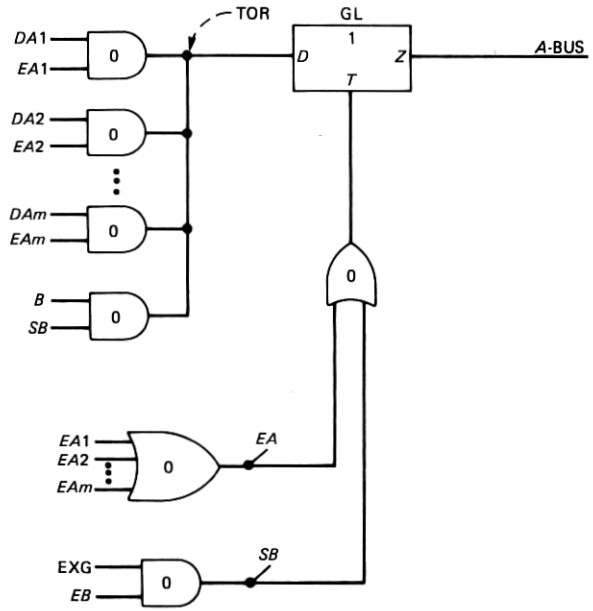


Fig. 15—The A-bus fault-free model. The unknown 3 condition is ignored here. The B-bus model is analogous.

Table III — Fault-free behavior of the coupled buses and their model

			Model		Actual Buses	
EXG	EA_i	EB_j	A	B	A	B
0	0	0	4	4	4	4
0	0	1	4	DB_j	4	DB_j
0	1	0	DA_i	4	DA_i	4
0	1	1	DA_i	DB_j	DA_i	DB_j
1	0	0	4	4	*	*
1	0	1	DB_j	DB_j	DB_j	DB_j
1	1	0	DA_i	DA_i	DA_i	DA_i
1	1	1	$DA_i + DB_j$	$DA_i + DB_j$	*	*

*

present		next	
A	B	A	B
0	0	0	0
0	1	3	3
1	0	3	3
1	1	1	1

EXG, EA_i, EB_j = enable control inputs
DA_i, DB_j = data inputs
A, B = bus outputs
3 = unknown 0 or 1

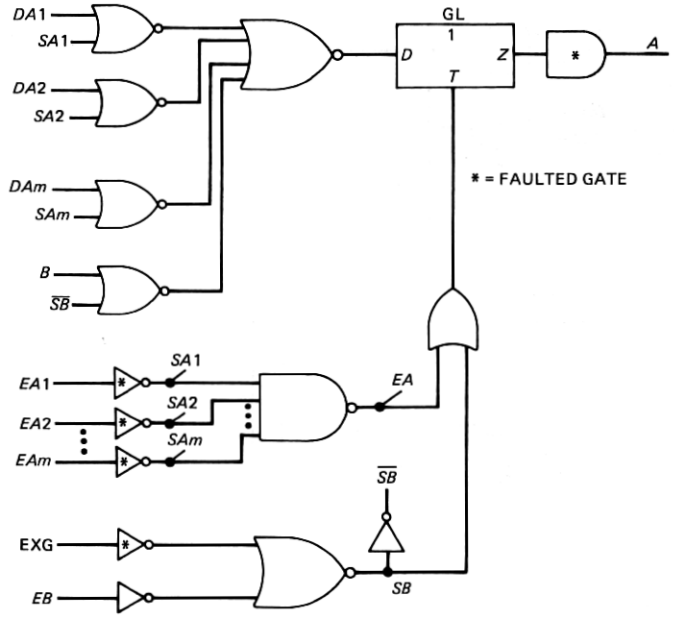


Fig. 16—The A-bus faulted model. Only gates marked * will be faulted by the simulator. Two SA0/SA1 node (pin) faults and $m + 1$ pairs of stuck-open/stuck-on faults have been assigned. The B-bus faulted model is analogous.

One possible faulted model for the A-bus is shown in Fig. 16. For multiple selections among the DAj and B the response is that of a 0-dominant short. The "3" output could be substituted, however. All elements are zero delay except the gated latch. The usual faults have been assigned to the model: two SA0/SA1 classical "pin" faults and $m + 1$ pairs of S-OP/S-ON faults. As before, the faulted model for the B-bus output is obtained by the symbolic interchange of As and Bs.

3.7 The programmable logic array (PLA)

The programmable logic array (PLA) is a simple method for implementing "random" combinational logic networks. An example of a three-variable, three output PLA is pictured in Fig. 17. The circuit is implemented in dynamic "pseudo NMOS." [7] That is, only the pull-up, or precharge, FETs are PMOS; all others are NMOS.

Clock $\Phi 1$ precharges the word and bit lines to the 1 state. Next, clock $\Phi 2$ causes the input signals x, y, z to propagate to the output terminals $W1$, $W2$, $W3$. (Although the two clock waveforms are essentially in phase, they are applied to PMOS and NMOS FETs, respectively, and the resulting conduction modes are 180° out of phase.)

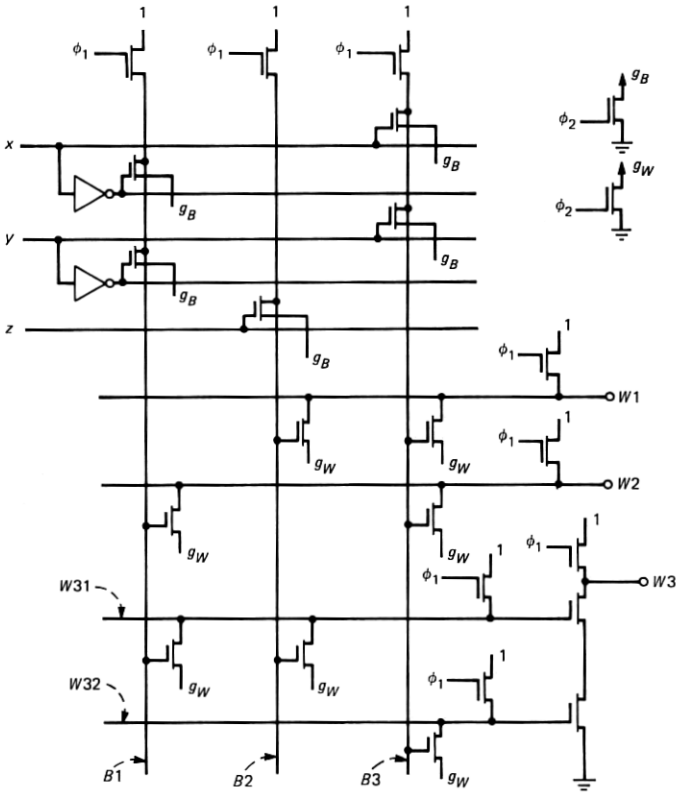


Fig. 17—Dynamic pseudo NMOS programmable logic array (PLA). There are three input variables (x, y, z) and three outputs ($W1$, $W2$, $W3$). Pull-up FETs are PMOS; all others are NMOS.[7]

The steady-state transfer functions are:

$$B1 = x \cdot y, \quad B2 = \bar{z}, \quad B3 = \bar{x} \cdot \bar{y},$$

$$\overline{W1} = B2 + B3, \quad \overline{W2} = B1 + B3, \quad \overline{W3} = W31 \cdot W32,$$

$$\overline{W31} = B1 + B2, \; \overline{W32} = B3.$$

Therefore,

$$\overline{W1} = \bar{z} + \bar{x} \cdot \bar{y},$$

$$\overline{W2} = x \cdot y + \bar{x} \cdot \bar{y},$$

$$\overline{W3} = \bar{x} \cdot y \cdot z + x \cdot \bar{y} \cdot z.$$

Models for the example PLA can be constructed by noting that the relation between inputs and bit lines and between bit lines and word lines is essentially that of a dynamic NOR gate. Figure 18 shows several

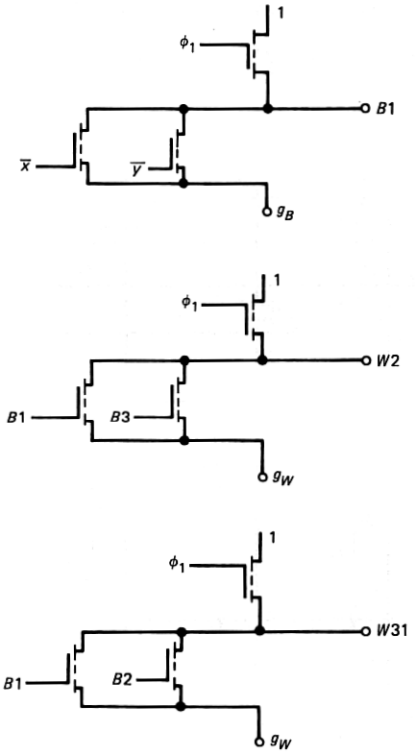


Fig. 18—Subcircuit portions of the PLA of Fig. 17.

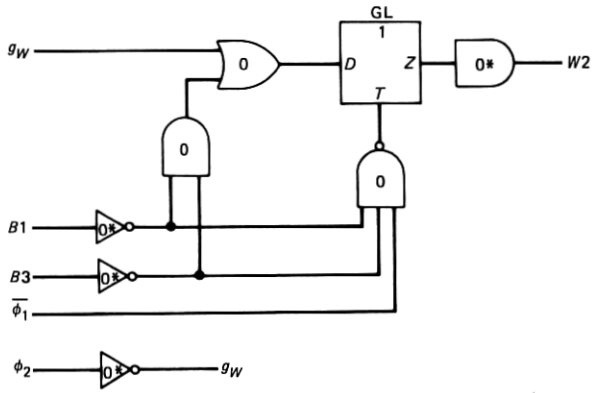
"subcircuit" portions of the PLA. Figure 19(a) gives the model for one typical subcircuit, $W2$. Figure 19(b) gives the model for the more complex $W3$ output. By this method the characteristics of any PLA, including dynamic properties, can be represented as a sum of simpler subcircuit models.

IV. THE GENERAL CASE

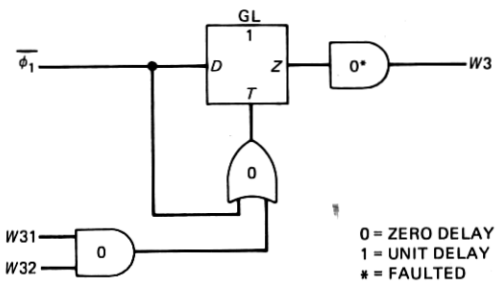
The preceding examples are particular illustrations of a more general configuration, shown in Fig. 20. The capacitance implicitly associated with the node F will be in one of four possible states: (*i*) low-impedance 0, (*ii*) low-impedance 1, (*iii*) high-impedance 0, or (*iv*) high-impedance 1. The first two conditions occur whenever one of the four input channels are enabled (conducting). The latter two cases arise when no circuit is enabled and the output is the resultant high-impedance state ($F = 4$).

In terms of the model, the logic state of the network is "transferred" to node F if and only if $T = 1$ where

$$T = \sum_i \mathrm{EX}_i \cdot \overline{\mathrm{EXSOP}}_i.$$



(a) W2 MODEL



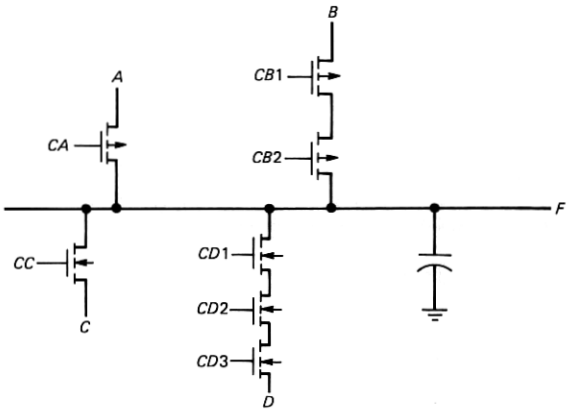
(b) W3 MODEL

Fig. 19—PLA subcircuit models. Gates marked * are faulted by the simulator, and 0 and 1 denote delays.

That is, at least one of the input circuits must be enabled (EXj = 1) and that circuit must not be stuck-open, i.e., EXSOPj = 0. Otherwise, T = 0 and F = 4.

If two channels, A and B, are simultaneously enabled ($A\&B$), then the result is said to be a 0-dominant short if $A\&B = A \cdot B$. For the 1-dominant short, $A\&B = A + B$. These two cases are shown in Figs. 21 and 22, respectively. Of course, if stuck-on faults are ignored entirely, then the network for T is represented by the summation immediately above.

Logic faults are assigned in the following manner. Simple n-input combinational gates are given the usual $n + 2$ stuck-at faults. In addition, one stuck-open fault is associated with each parallel branch of the pull-up/pull-down networks. Stuck-opens in strictly series paths to either VDD or VSS are ignored because of their similarity to SA0/SA1 faults, respectively. Unless the details of circuit technology and design dictate

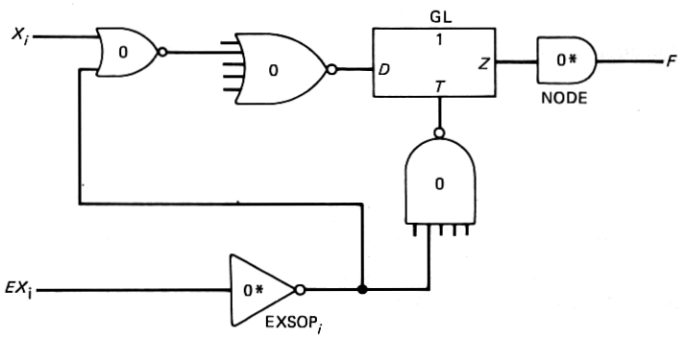


$$T = EA \cdot \overline{\text{EASOP}} + EB \cdot \overline{\text{EBSOP}} + EC \cdot \overline{\text{ECSOP}} + \cdots$$

WHERE $EA = \overline{CA}$, $EB = \overline{CB1} \cdot \overline{CB2}$
$EC = CC$, $ED = CD1 \cdot CD2 \cdot CD3$

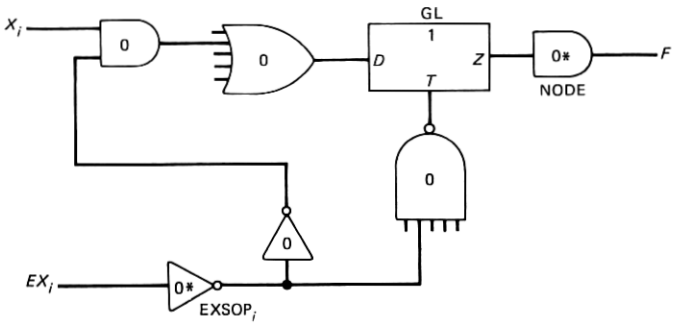
Fig. 20—The general case: multiple series/parallel combinations driving one node. FETs on the upper side are p-channel; those below are n-channel.

a clear choice, stuck-on logic faults should generally be disregarded. Nodes at which several tri-state elements connect can be assigned the two SA0/SA1 classical faults and one S-OP fault per channel. Fault conditions producing the unknown "3" state should not be modeled. On the



	T	D
NO FAULTS	$\sum_i EX_i$	$\prod_i (X_i + \overline{EX_i})$
$EXSOP_j(1)$	$\sum_{i \neq j} EX_i$	$\prod_{i \neq j} (X_i + \overline{EX_i})$
$EXSOP_j(0)$	1	$X_j \cdot \prod_{i \neq j} (X_i + \overline{EX_i})$

Fig. 21—The general case: S-OP and S-ON faults (0-dominant).



	T	D
NO FAULTS	$\sum_i EX_i$	$\sum_i X_i \cdot EX_i$
$EXSOP_j$ (1)	$\sum_{i \neq j} EX_i$	$\sum_{i \neq j} X_i \cdot EX_i$
$EXSOP_j$ (0)	1	$X_j + \sum_{i \neq j} X_i \cdot EX_i$

Fig. 22—The general case: S-OP and S-ON faults (1-dominant). This is a variant of the model of Fig. 21 except that the S-ON fault is 1-dominant.

other hand, simulation models for design verification ("true value" simulation) can include the "3" state as an indication of erroneous tri-state selection as in bus-oriented circuits.

V. SUMMARY

This paper has described a procedure for modeling faults which are a peculiarity of CMOS digital integrated circuits. Furthermore, the resultant methodology was also utilized to provide simulator models for complex MOS dynamic circuit topologies. The models are gate-level in structure and can be adapted for use on essentially any general purpose logic simulator. In addition, the models have been chosen to avoid faults which are artifacts of the model and which do not represent physically likely logic defects. The models have also been structured to preserve the distinction between classical and nonclassical faults. In addition, all models are designed to avoid races and circuit oscillations.

From the examples in this paper, it can be seen that there are a number of choices that can be taken for modeling a given logical function. In addition, a particular function may well be reduced in gate-count from the examples and the "general" realizations shown in this paper. That is, depending upon the selection of characteristics most important to the user, different models will result. In that spirit, the illustrations used above were selected to demonstrate the principles of modeling in a clear, straightforward manner.

REFERENCES

1. "LAMP: Logic Analyzer for Maintenance Planning," B.S.T.J., *53*, No. 8 (October 1974), pp. 1431–1555.
2. T. G. Athanas, "Development of COS/MOS Technology," Solid State Tech., *17*, No. 6 (June 1974), pp. 54–59.
3. H. Rombeek and P. Wilcox, "Interactive Logic Simulation and Test Pattern Development for Digital Circuitry," Electro 76, Paper 26.2, April 1976.
4. R. H. Krambeck, private communication.
5. The first successful demonstration of the FDL capability was performed for the NOR gate faults in Table I, R. E. Strebendt, private communication.
6. The logic symbol was suggested to the author by D. L. Kushler.
7. J. A. Cooper, J. A. Copeland, R. H. Krambeck, D. C. Stanzione, and L. C. Thomas, "A CMOS Microprocessor for Telecommunications Applications," ISSCC 77, Paper THPM 12.3, February 17, 1977.