NASA CONTRACTOR REPORT

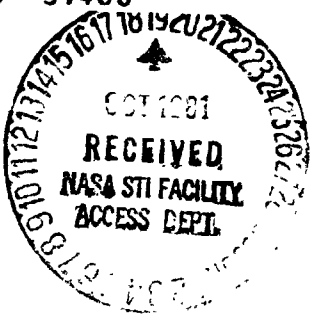
NASA CR-161295

STANDARD TRANSISTOR ARRAY (STAR)- Volume 2, Addendum 1 SIMLOG/ TESTGN USER's GUIDE

By B. D. Carroll
Electrical Engineering Department
Auburn University
Auburn, Alabama

September 14, 1979

Final Report



Prepared for

NASA - George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama 35812

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

The system of programs described herein was written to provide a cost-effective but powerful tool for simulating logic circuits and for generating test sequences for logic circuits. BASIC was chosen as the implementation language since it is commonly available on minicomputer-based and microcomputer-based computing systems. Application of the programs to circuits containing five-hundred or more gates is feasible if the host computer system provides a sufficient primary and secondary storage capacity.

The system consists of two separate programs, SIMLOG and TESTGN, that share a common data base as shown in Figure 1. Each of the programs will now be briefly described. Additional detail on the programs is presented throughout the report.

SIMLOG is a gate-level logic simulation program that is based on a three-valued logic model and on a unit-delay timing model. Both combinational and sequential circuits can be handled. Circuits containing single or multiple stuck-type faults can be simulated. Logic elements available for use include NAND gates, NOR gates, unit delay elements, and edge-triggered D flip-flops.

TESTGN is a test sequence generation program for circuits previously simulated by SIMLOG. The program can be used to generate tests for unspecified faults or for specified single stuck-type faults.

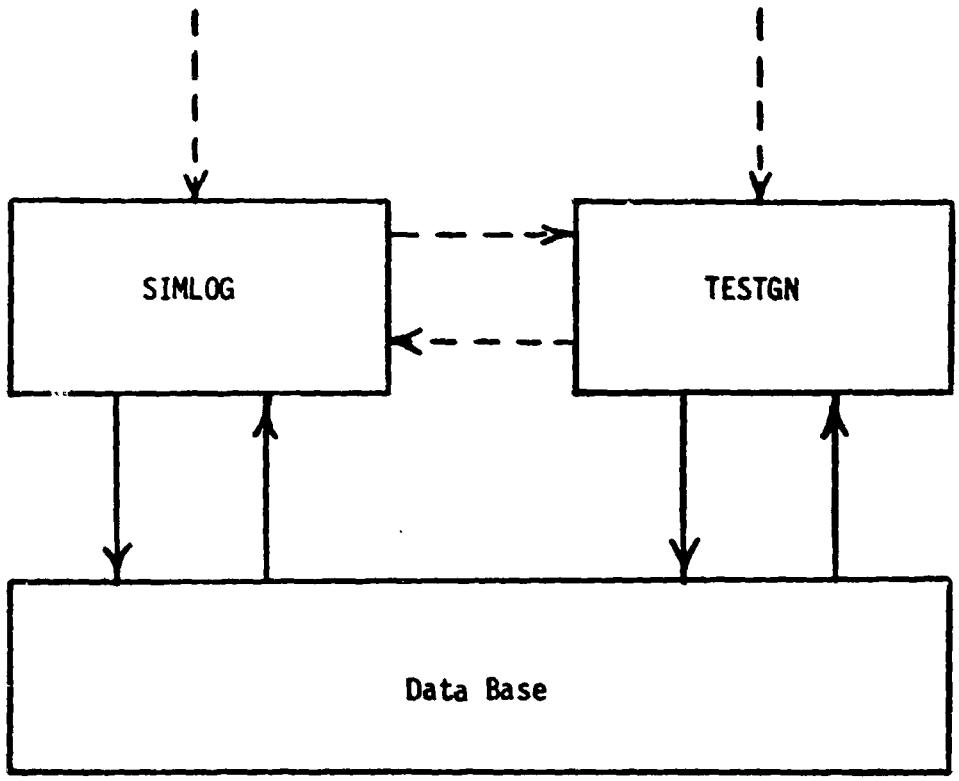


Figure 1 - SIMLOG/TESTGN System Structure.

This document is a user's guide to SIMLOG and TESTGN. The reader wanting more detail on the theoretical basis of the programs or on the programs themselves is referred to [1] and [2].

2. BACKGROUND

The three-valued logic model and the unit-delay timing model on which SIMLOG and TESTGN are based are briefly reviewed in this section. Refer to [1] for a more indepth discussion of these models.

LOGIC MODEL

SIMLOG uses a three-valued logic model with the three valves encoded as a pair of binary numbers. Logic signals can assume the values 0, 1, or unknown which are encoded as follows:

Logic 0 -- (0,1)

Logic 1 -- (1,0)

Unknown -- (0,0)

Undefined -- (1,1)

Consider a two-input NAND gate with inputs A and B and output C. Each of these signals is represented by a pair of binary variables in the model. For example, (A, A-) is used to represent input A. Variable A is called the true or one-variable, and A- is the corresponding false or zero-variable. The value of signal A at time t is represented by (At, A-t).

The truth table of the two-input NAND gate is given below for time t.

(At, A-t)	(Bt, B-t)	(Ct, C-t)
(0, 1)	(0, 1)	(1, 0)
(0, 1)	(1, 0)	(1, 0)
(1, 0)	(0, 1)	(1, 0)
(1, 0)	(1, 0)	(0, 1)

Algebraically, the NAND gate output at time t can be described by the following pair of Boolean functions.

$$Ct = A\text{-}t + B\text{-}t \quad (1)$$

$$C\text{-}t = At\ Bt \quad (2)$$

Equation (1) is referred to as the true or one-function while (2) is the false or zero-function. These concepts can be extended to more complex logic structure in a straightforward manner.

SIMLOG simulates a logic circuit by generating the function pairs for each signal in the circuit. This is done in an iterative fashion for each value of t until a stable set of equations is reached. The stable set of equations for $t = \tau$ represents the behavior of the circuit for times from t = 1 through $t = \tau$.

Timing Model

SIMLOG uses a unit-delay timing model. That is to say that each logic element is assumed to have a unit-of-time delay associated with it. Thus all elements exhibit the same amount of delay in unspecified units.

A simple example will be used to more clearly describe the timing model. Consider the simple logic circuit shown in Figure 2. Simulation of this circuit for two values of t is shown in Figure 3. Note that two time parameters (t and r) are used. Parameter t is called the input-time and is used to represent times of possible input changes. The value of t is incremented after a stable set of equations has been reached. Parameter r is called the ripple-time and represents the effect

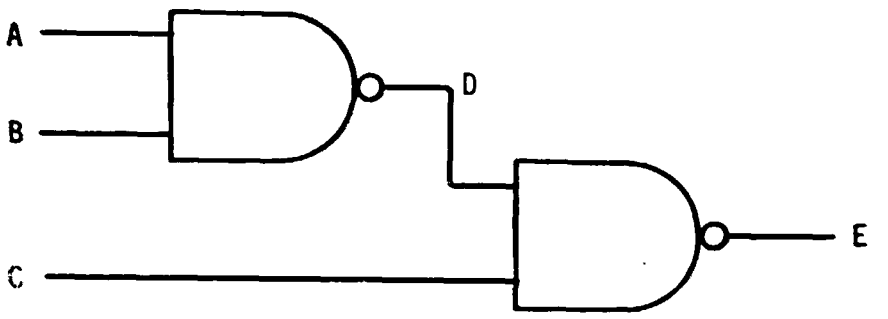


Figure 2. Circuit for Timing Model Example.

t	r	(Dt, D-t)	(Et, E-t)
0	0	(0, 0)	(0, 0)
1	1	(A-1+B-1, A1B1)	(0, 0)
	2	(A-1+B-1, A1B1)	(A1B1+C-1, A-1C1+B-1C1)
	3	(A-1+B-1, A1B1)	(A1B1+C-1, A-1C1+B-1C1)
2	3	(A-2+B-2, A2B2)	(A1B1+C-2, A-1C2+B-1C2)
	4	(A-2+B-2, A2B2)	(A2B2+C-2, A-2C2+B-2C2)
	5	(A-2+B-2, A2B2)	(A2B2+C-2, A-2C2+B-2C2)

Figure 3. SIMLOG Timing Model

of time delays in the logic elements. Note that r is not increment after a stable set of equations is observed.

The increment of ripple-time during a given input-time t represents the minimum time units needed for the effects of an input change at time t to propagate through the circuit. An input time-frame refers to the time from an input change until the next input change.

3. SYSTEM USAGE

This section is devoted to detailed descriptions of SIMLOG and TESTGN from a user's point-of-view. The following subsection outlines a typical application session. Conventions that must be followed when using the programs are then described. Finally, the input requirements and the output results of each program are discussed. All descriptions in this section are illustrated for the circuit in Figure 4.

TYPICAL APPLICATION

Use of the SIMLOG/TESTGN System generally involves the following three steps.

1. Preparation of a description of the logic circuit to be processed that is compatible with the System,
2. Execution of the SIMLOG program,
3. Execution of the TESTGN program.

These steps are outlined below in more detail for the following application. It is desired to generate test sequences for the circuit shown in Figure 4 such that any single stuck-type fault on a circuit input can be detected if possible at each circuit output. It is well known that such test sequences generally will detect many internal faults in addition to the input faults. Such test sequences are desired for both Q = 1(QBAR=0) and Q = 0(QBAR=1) initial states of the circuit.

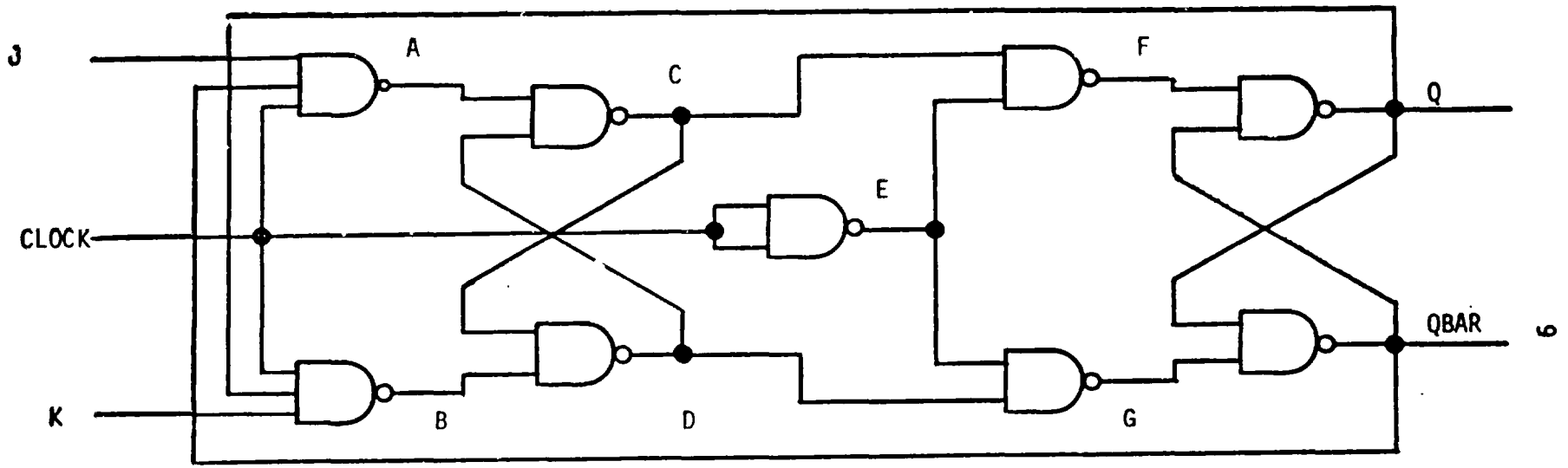


Figure 4. JK Flip-Flop Circuit

The above three steps can be expanded as follows for the given application.

1.(a) Label the circuit input nets and output nets with unique names,

(b) Label all remaining nets in the circuit with unique names.

See Figure 4 for the result of these steps.

2.(a) Load the SIMLOG program into primary memory for execution,

(b) Start execution of SIMLOG,

(c) Select NEW CIRCUIT simulation mode,

(d) Select PRINT STABLE EQUATION SETS ONLY output mode,

(e) Select RACE ANALYSIS,

(f) Enter the circuit description,

(g) Enter primary input line names,

(h) Enter primary output line names,

(i) Enter cross-coupled element pairs,

(j) Reject fault simulation,

(k) Enter a starting state compatible with Q = 1 and QBAR = 0,

(l) Enter an INPUT-TIME LIMIT,

(m) Enter a RIPPLE-TIME LIMIT,

(n) Wait for the simulation to terminate.

3.(a) Select TESTGN for execution,

(b) Select the UNSPECIFIED FAULTS test generation mode,

(c) Wait for the six test functions to be generated.

Execution and results of each of the above are shown in the appendix.

CONVENTIONS

The following conventions should be observed when using the SIMLOG/TESTGN System.

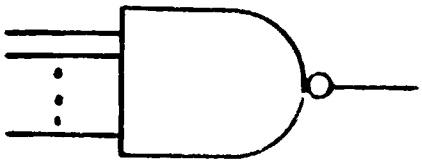
Net Names - One or more alphabetical characters. Numeric and special characters cannot be used. The user should be aware that long names will increase the run-time of the programs.

Input-Time Limit - Unrestricted.

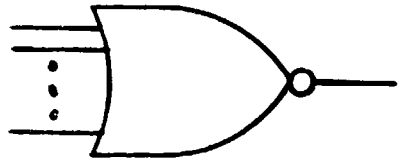
Ripple-Time Limit - Unrestricted.

Element Types - NAND gates, NOR gates, Edge-triggered D flip-flops, unit-delay elements. Fan-in of NAND and NOR gates is not restricted. The D flip-flop is a two-input, two-output device with ordered inputs. The delay element is a one-input, one-output device. See Figure 5 for more detail on the elements.

Circuit Size - The number of elements in a circuit cannot exceed two hundred with flip-flops counted as two elements. The number of cross-coupled element pairs cannot exceed fifty. Furthermore, the total fan-in of all elements combined cannot exceed four hundred. These restrictions are imposed by array dimensions and can be increased if adequate storage capacity is available. See the Programmer's Guide [2] for instructions on increasing array dimensions.

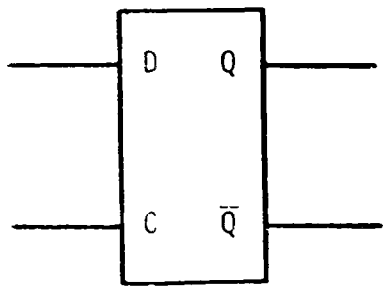


(a) NAND Gate



(b) NOR Gate

(c) Unit Delay



(d) D Flip-flop

Figure 5. SIMLOG Logic Elements

SIMLOG Usage

This subsection contains detailed descriptions of the input requirements of SIMLOG and of the output that is produced.

Input Requirements

SIMLOG inputs are entered either interactively or from a file after the program has been loaded into primary storage and program execution initiated. Each input step and the corresponding options are detailed below.

The following is the first message displayed after SIMLOG execution has begun.

```
SELECT SIMULATION MODE.  <CR>=1.
    (1) NEW CIRCUIT
    (2) NEW FAULT
    (3) NEW STARTING STATE
    (4) CIRCUIT DESCRIPTION ON FILE

?
```

One of the four simulation modes must be selected by the user before program execution continues. Mode 1 may be selected by entering a carriage return <CR> only or by entering the number 1 followed by a <CR>. The mode number followed by a <CR> is required for selecting modes 2, 3, or 4.

Selection of mode 1 is made if the user wants to begin a new terminal session and if the circuit description of interest is not currently in the NET file. A terminal session is defined to begin when a circuit description is entered by either modes 1 or 4. Mode 2 is selected when it is desired to simulate the same circuit but under a different fault condition. Likewise, mode 3 is selected when the same circuit is to

be simulated but with a different starting state.

When mode 4 is to be used, the description of the circuit must be loaded into a file named NET (Sigma 5) or NET.DAT(PDP11) prior to the beginning of the terminal session. This can be done by using an appropriate utility program such as a text or file editor.

Circuit description entered under mode 1 are placed in the NET or NET.DAT file. Hence, it is recommended that when a terminal session is completed that the contents of NET or NET.DAT be copied to a file with a name indicative of the circuit. A library of circuit description files can be easily built in this way.

Selection of an output mode is requested by the following message.

```
SELECT OUTPUT MODE.  <CR>=1.
    (1) PRINT STABLE EQUATION SETS ONLY
    (2) PRINT ALL EQUATION SETS

?
```

Output mode 1 may be selected by the entry of <CR> only or by the entry of the number one followed by <CR>. The number two followed by <CR> must be used to select output mode 2.

Selection of output mode 1 results in the output of only one equation set for each input-time step. The set contains an equation pair for each internal net of the circuit.

Output mode 2 results in the output of an equation set for each ripple-time step. In general, an input-time step contains multiple ripple-time steps. Mode 1 is recommended when only static effects are of interest. Circuit dynamic characteristics are more clearly indicated when mode 2 is chosen.

Some sequential circuits are subject to timing problems referred to as race conditions. Circuits with race conditions may function improperly under certain input sequences. SIMLOG contains a race analysis procedure which detects race conditions caused by cross-coupled NAND gates or cross-coupled NOR gates. The procedure then inhibits input sequences that would invoke the race conditions.

The race analysis procedure may be selected or deselected by the user by responding to the following query.

RACE ANALYSIS (YES OR NO). <CR>=YES.?

Race analysis is automatically deselected if no cross-coupled pairs of gates are present in the circuit. In general, race analysis should be selected.

The circuit description is now entered if simulation mode 1 was selected previously. The user is prompted for this information by the following message.

```
ENTER--ELEMENT NAME, TYPE, AND INPUT LIST.
SEPARATE ENTRIES WITH A SPACE.
ENTER <CR> TO TERMINATE ENTRY.
?
```

Element name refers to the net name that originates at the element output. Element type should be entered as NAND, NOR, DFF, or DELAY as appropriate. When entering the input list for the D flip-flop, enter the D input first followed on the right by the clock input.

Entry of the network descriptions is completed by entering the net names corresponding to primary inputs and to primary outputs and by entering the net names contained in cross-coupled element pairs. This

information is requested by the following three messages.

```
ENTER PRIMARY INPUT LINE NAMES.  TERMINATE WITH <CR>.
?

ENTER PRIMARY OUTPUT LINE NAMES.  TERMINATE WITH <CR>.
?

ENTER--CROSS-COUPLED ELEMENT PAIRS.  SEPARATE NAMES
WITH A SPACE.  ENTER <CR> TO TERMINATE.
?
```

A SIMLOG user may initiate the insertion of faults in a circuit by responding with YES to the following query.

```
FAULT SIMULATION(YES/NO).  <CR>=NO.?
```

Selection of fault simulation results in the following message being displayed.

```
SELECT FAULT CLASS OR ENTER <CR> TO TERMINATE.
    (1)  OUTPUT FAULT
    (2)  PRIMARY INPUT FAULT
    (3)  ELEMENT INPUT FAULT

FAULT CLASS?
```

Insertion of a fault on a net originating from an element output is initiated by selection of fault class 1 (output fault). All branches of the net are faulted in this class.

Fault insertion on a net originating from a primary input is initiated by selection of fault class 2 (primary input fault). All branches of the net are faulted.

Faults can also be inserted on individual element inputs without effecting other branches of the corresponding net. This in initiated by the selection of fault class 3 (element input fault).

Selection of fault class 1 produces the following response.

```
ENTER LINE NAME OR <CR> TO TERMINATE ENTRIES.
LINE NAME?
```

The type of fault, stuck-at-0 (S0) or stuck-at-1 (S1), to be inserted on the line entered previously is requested as follows.

```
ENTER FAULT CONDITIONS S0/S1?
```

Multiple faults may be entered by repeating the insertion process for each component of the multiple fault. Fault insertion is terminated by the entry of <CR> in the place of a line name or a fault class number as shown below.

```
LINE NAME?
FAULT CLASS?
```

The selection of fault class 2 from above results in a similar sequence of queries and responses as illustrated below.

```
LINE NAME?  CLOCK
ENTER FAULT CONDITIONS S0/S1? S1
LINE NAME?

FAULT CLASS?
```

A slightly different sequence of events follows the selection of fault class 3 as shown below. Note that the element output name as well as the name of the input to be faulted must be entered in order to uniquely identify the desired fault location.

```
ENTER ELEMENT INPUT NAME OR <CR> TO TERMINATE.
INPUT NAME?  E
ELEMENT OUTPUT NAME?  F
FAULT CONDITIONS S0/S1?  S0
INPUT NAME?

FAULT CLASS?
```

Starting-state (SS) mode selection is the final step of circuit description entry. Eight modes are available and are identified in the following message.

ENTER STARTING STATE MODE SELECTION. <CR> = 1.

(1) ALL UNKNOWNS
(2) DOUBLE CROSS-COUPLED GATE VARIABLES
(3) ALL VARIABLES
(4) USER SPECIFIED
(5) SINGLE CROSS-COUPLED GATE VARIABLE
(6) SPECIFIED CONSTANTS
(7) FIXED INPUT VALUES
(8) MULTIMODE

Selection of SS modes 1, 2, 3, or 5 require no further response from the user. However, SS modes 4, 6, 7, and 8 do require the user to supply additional information as is described later.

SS mode 1 initializes all nets in the circuit to the unknown (0,0) value. SS mode 2 assigns a variable for the starting state of each gate in a cross-coupled pair. For example, consider the C and D pair in Figure 4. The starting state of C will be assigned as (C0, C-0) while D is assigned (D0, D-0). Nets not in a cross-coupled pair are assigned the unknown value.

SS mode 3 assigns a variable to all nets in the circuit by the following rule. Net X is initialized as (X0, X-0). SS mode 5 assigns one variable to each cross-coupled pair of gates. Again consider the C and D pair from Figure 4. The starting state of C is assigned to be (C0, C-0) as in SS mode 2. However, D is assigned an initial value of (C-0, C0). In effect, C and D are assumed to be in complementary states.

SS mode 4 is selected when the user wants to specify an initial condition not available with the other modes. This mode is useful for example if it is desired to continue a simulation that was previously terminated. The initial conditions entered are then the final equations produced by the earlier simulation. Selection of SS mode 4 produces the following request for additional information.

```
ENTER EQUATION OR ENTER <CR> TO TERMINATE ENTRIES.
? C=XO
ENTER EQUATION OR ENTER <CR> TO TERMINATE ENTRIES.
? C-=X-O
ENTER EQUATION OR ENTER <CR> TO TERMINATE ENTRIES.
? D=X-O
ENTER EQUATION OR ENTER <CR> TO TERMINATE ENTRIES.
? D-=XO
ENTER EQUATION OR ENTER <CR> TO TERMINATE ENTRIES.
? Q=XO
ENTER EQUATION OR ENTER <CR> TO TERMINATE ENTRIES.
? Q-=X-O
ENTER EQUATION OR ENTER <CR> TO TERMINATE ENTRIES.
? QBAR=X-O
ENTER EQUATION OR ENTER <CR> TO TERMINATE ENTRIES.
? QBAR-=XO
ENTER EQUATION OR ENTER <CR> TO TERMINATE ENTRIES.
?
```

The sequence of entries above assigns (XO,X-O) as the starting state of C and Q and assigns (X-O,XO) as the starting state of D and QBAR. Nets not explicitly assigned are initialized in the unknown state.

Selection of SS mode 6 allows the user to conveniently assign constant values as the starting state of selected nets. The following sequence illustrates the assignment of (1,0) to nets A, B, C, F, G, and Q and the assignment of (0,1) to nets D, E, and QBAR.

```
ENTER--LINE NAME=0 OR 1--OR ENTER <CR> TO TERMINATE ENTRIES.
? A=1
ENTER--LINE NAME=0 OR 1--OR ENTER <CR> TO TERMINATE ENTRIES.
? B=1
ENTER--LINE NAME=0 OR 1--OR ENTER <CR> TO TERMINATE ENTRIES.
? C=1
ENTER--LINE NAME=0 OR 1--OR ENTER <CR> TO TERMINATE ENTRIES.
? D=0
ENTER--LINE NAME=0 OR 1--OR ENTER <CR> TO TERMINATE ENTRIES.
? E=0
ENTER--LINE NAME=0 OR 1--OR ENTER <CR> TO TERMINATE ENTRIES.
? F=1
ENTER--LINE NAME=0 OR 1--OR ENTER <CR> TO TERMINATE ENTRIES.
? G=1
ENTER--LINE NAME=0 OR 1--OR ENTER <CR> TO TERMINATE ENTRIES.
? Q=1
ENTER--LINE NAME=0 OR 1--OR ENTER <CR> TO TERMINATE ENTRIES.
? QBAR=0
ENTER--LINE NAME=0 OR 1--OR ENTER <CR> TO TERMINATE ENTRIES.
?
```

Inputs are normally handled as variables in SIMLOG simulations. It may be desirable to assign constant values to one or more inputs. This can be accomplished by first selecting SS mode 7 and then making the desired assignments as illustrated below. Other nets in the circuit are left in the unknown state.

```
ENTER LINE NAME=0 OR 1 OR TO STOP ENTER <CR>? J=1
ENTER LINE NAME=0 OR 1 OR TO STOP ENTER <CR>? CLOCK=1
ENTER LINE NAME=0 OR 1 OR TO STOP ENTER <CR>? K=1
ENTER LINE NAME=0 OR 1 OR TO STOP ENTER <CR>?
```

Selection of SS mode 8 allows the user to assign constants as desired to primary input nets and to also choose another SS mode as well. The following message is printed in response to SS mode 8 selection.

```
SELECT MODE (2), (3), (4), (5), OR (6).
MODE (7) IS THEN SELECTED AUTOMATICALLY.
?
```

After the circuit description entry is complete, the user is requested to enter the input-time limit and the ripple-time limit as illustrated below.

```
ENTER INPUT-TIME LIMIT? 2
ENTER RIPPLE-TIME LIMIT? 6
```

Circuit simulation begins automatically following the entry of the ripple-time limit. A description of SIMLOG output is presented in the next subsection. The message below is printed following a normal simulation termination.

```
DO YOU WISH TO ENTER A NEW TIME LIMIT--YES/NO, <CR>=NO?
```

A YES response to the above query produces the following sequence of messages which must be responded to in turn.

```
ENTER NEW INPUT--TIME LIMIT?
NEW INPUT MODE--YES/NO.  <CR>=NO.?
```

The following message is printed if a new input mode is desired.

```
SELECT DESIRED MODE.
 (1) FIXED INPUT VALUES
 (2) INPUT VARIABLES

?
```

Simulation is resumed automatically if the input mode is left unchanged.

A NO response to the earlier query concerning a new input-time limit produces the following next step.

```
GENERATE TEST SEQUENCES--YES/NO.  <CR>=NO.?
```

A YES response causes TESTGN to be loaded and execution begun. A NO response causes SIMLOG to be restarted from the first step.

Possible oscillations may be detected during the circuit simulations run. The following message is printed when this occurs.

```
POSSIBLE OSCILLATION.
ENTER NEW RIPPLE-TIME LIMIT OR <CR> TO STOP?
```

Control is returned to the first step of SIMLOG if <CR> is entered. Otherwise, simulation resumes using the new limit.

Output Interpretation

The output of SIMLOG consists of a header giving the current input-time (T) and the current ripple-time (R) followed by a list of the equation pairs for each net in the circuit being simulated. The frequency of the output is determined by which output mode was selected by the user. Output mode 1 prints only the final (stable) equation set for a given input-time, whereas output mode 2 prints the equation set corresponding to each new ripple-time.

Shown below is the last output frame produced by the fault-free simulation of the circuit shown in Figure 4 with an input-time limit of two and initial conditions A=B=C=F=G=Q=1 and D=E=QBAR=0.

```
T=2          R=11

A-2=0

A2=J-2+K-1K-2+CLOCK-2+CLOCK2K-1+CLOCK-1K-2+CLOCK1CLOCK2+CLOCK-1CLOCK2

B-2=CLOCK2K-1K2+CLOCK1CLOCK2K2+CLOCK-1CLOCK2K2

B2=K-2+CLOCK-2

C-2=CLOCK1J-2K1+CLOCK 2K-1K2+CLOCK1CLOCK2K1+CLOCK1CLOCK2K2
     +CLOCK-1CLOCK2K2+CLOCK-2CLOCK1K1

C2=K-1K-2+CLOCK-1K-2+CLOCK-2K-1+CLOCK-1CLOCK-2

D-2=K-1K-2+CLOCK-1K-2+CLOCK-2K-1+CLOCK-1CLOCK-2

D2=CLOCK1J-2K1+CLOCK2K-1K2+CLOCK1CLOCK2K1+CLOCK1CLOCL2K2
     +CLOCK-1CLOCK2K2+CLOCK-2CLOCK1K1

E-2=CLOCK2
```

E2=CLOCK-2

F-2=CLOCK-2K-1+CLOCK-1CLOCK-2

F2=CLOCK2+CLOCK1J-2K1+CLOCK-2CLOCK1K1

G-2=CLOCK-2CLOCK1K1

G2=CLOCK2+K-1K-2+CLOCK-1K-2+CLOCK-2K-1+CLOCK-1CLOCK-2

Q-2=CLOCK-2CLOCK1K1

Q2=K-1K-2+CLOCK2K-1+CLOCK-1K-2+CLOCK-2K-1+CLOCK1CLOCK2+CLOCK-1CLOCK2
+CLOCK-1CLOCK-2

QBAR-2=K-1K-2+CLOCK2K-1+CLOCK-1K-2+CLOCK-2K-1+CLOCK1CLOCK2
+CLOCK-1CLOCK2+CLOCK-1CLOCK-2

QBAR2=CLOCK-2CLOCK1K1

A detailed interpretation of selected equations from the above list will now be given. Consider the equation A-2=0. This result means that net A cannot be made 0 under the given initial conditions and input-time limit.

Now consider the equation F-2=CLOCK-2K-1+CLOCK-1CLOCK-2. This result implies that there are two input sequences that will produce a zero on net F in two input-time steps. K=0 during the first input frame followed by CLOCK=0 during the second will produce F=0 after the circuit has stabilized following the last input change. The same result can be produced by holding CLOCK=0 for two consecutive input changes.

Finally, consider the equation QBAR=CLOCK-2CLOCK1K1. This says that QBAR=1 can be produced only by applying K=CLOCK=1 during the first input frame and CLOCK=0 during the next input frame.

The output produced by fault simulations will now be discussed. Again consider the circuit of Figure 4 with the same initial conditions and input-time limit as specified above. The following is the last output frame produced by SIMLOG for the circuit with a stuck-at-0 fault inserted on net E.

```
T=2        R=11

A-2=0

A2=E*+J-2+K-1K-2+CLOCK-2+CLOCK2K-1+CLOCK-1K-2+CLOCK1CLOCK2+CLOCK-1CLOCK2

B-2=CLOCK2E*K2+CLOCK2K-1K2+CLOCK1CLOCK2K2+CLOCK-1CLOCK2K2

B2=K-2+CLOCK-2

C-2=CLOCK1E*K1+CLOCK2E*K2+CLOCK1J-2K1=CLOCK2K-1K2+CLOCK1CLOCK2K1
     +CLOCK1CLOCK2K2+CLOCK-1CLOCK2K2+CLOCK-2CLOCK1K1

C2=K-1K-2+CLOCK-1K-2+CLOCK-2K-1+CLOCK-1CLOCK-2

D-2=K-1K-2+CLOCK-1K-2+CLOCK-2K-1+CLOCK-1CLOCK-2

D2=CLOCK1E*K1+CLOCK2E*K2+CLOCK1J-2K1+CLOCK2K-1K2+CLOCK1CLOCK2K1
     +CLOCK1CLOCK2K2+CLOCK-1CLOCK2K2+CLOCK-2CLOCK1K1

E-2=CLOCK2+E*

E2=CLOCK-2E-*

F-2=CLOCK-2E-*K-1+CLOCK-1CLOCK-2E-*

F2=E*+CLOCK2+CLOCK1J-2K1+CLOCK-2CLOCK1K1

G-2=CLOCK-2CLOCK1E-*K1

G2=E*+CLOCK2+K-1K-2+CLOCK-1K-2+CLOCK-2K-1+CLOCK-1CLOCK-2

Q-2=CLOCK-2CLOCK1E-*K1

Q2=E*+K-1K-2+CLOCK2K-1+CLOCK-1K-2+CLOCK-2L-1+CLOCK1CLOCK2+CLOCK-1CLOCK2
     +CLOCK-1CLOCK-2
```

```
QBAR-2=E*+K=1K=2+CLOCK2K=1+CLOCK-1K-2+CLOCK-2K-1+CLOCK1CLOCK2
     +CLOCK-1CLOCK2+CLOCK-1CLOCK2

QBAR2=CLOCK-2CLOCK1E-*K1
```

Consider the equation F2=E*+CLOCK2+CLOCK1J-2K1+CLOCK-2CLOCK1K1. This is interpreted as saying that net F=1 after two input frames if net E is stuck-at-0, or if CLOCK=1 during input frame two, or if CLOCK=K=1 during input frame one and J=0 during input frame two, or if CLOCK=K=1 during input frame one and CLOCK=0 during input frame two.

The equation QBAR=CLOCK-2CLOCK1E-*K1 implies QBAR=1 can be produced if E is not stuck-at-0 and CLOCK=K=1 during input time one and CLOCK=0 during input time two.

The following simulation results are produced if a stuck-at-one fault is inserted on the CLOCK input line.

```
T=2        R=11

A-2=0

A2=J-2+CLOCK!+K-1K-2+CLOCK2K-1+CLOCK1CLOCK2+CLOCK-2CLOCK-!
     +CLOCK-!CLOCK-1K-2+CLOCK-!CLOCK-1CLOCK2

B-2=CLOCK!K2+CLOCK2K-1K2+CLOCK1CLOCK2K2+CLOCK-!CLOCK-1CLOCK2K2

B2=K-2+CLOCK-2CLOCK-!

C-2=CLOCK!K1+CLOCK!K2+CLOCK1J-2K1+CLOCK2K-1K2+CLOCK1CLOCK2K1
     +CLOCK1CLOCK2K2+CLOCK-!CLOCK-1CLOCK2K2
     +CLOCK-!CLOCK-2CLOCK1K1

C2=K-1K-2+CLOCK-!CLOCK-1K-2+CLOCK-!CLOCK-2K-1+CLOCK-!CLOCK-1CLOCK-2

D-2=K-1K-2+CLOCK-!CLOCK-1K-2+CLOCK-!CLOCK-2K-1+CLOCK-!CLOCK-1CLOCK-2

D2=CLOCK!K1+CLOCKK2+CLOCK1J-2K1+CLOCK2K-1K2+CLOCK1CLOCK2K1
     +CLOCK1CLOCK2K2+CLOCK-!CLOCK-1CLOCK2K2
     +CLOCK-!CLOCK-2CLOCK1K1
```

```
E-2=CLOCK!+CLOCK2

E2=CLOCK-2CLOCK-!

F-2=CLOCK-!CLOCK-2K-1+CLOCK-!CLOCK-1CLOCK-2

F2=CLOCK!+CLOCK2+CLOCK1J-2K1+CLOCK-!-2CLOCK1K1

G-2=CLOCK-!CLOCK-2CLOCK1K1

G2=CLOCK!+CLOCK2+K-1K-2+CLOCK-!CLOCK-1K-2+CLOCK-!CLOCK-2K-1
     +CLOCK-!CLOCK-1CLOCK-2

Q-2=CLOCK-!CLOCK-2CLOCK1K1

Q2=CLOCK!+K-1K-2+CLOCK2K-1+CLOCK1CLOCK2+CLOCK-!CLOCK-1K-2
     +CLOCK-!CLOCK-2K-1+CLOCK-!CLOCK-1CLOCK2
     +CLOCK-!CLOCK-1CLOCK-2

QBAR-2=CLOCK!+K-1K-2+CLOCK2K-1+CLOCK1CLOCK2+CLOCK-!CLOCK-1K-2
     +CLOCK-!CLOCK-2K-1+CLOCK-!CLOCK-1CLOCK2
     +CLOCK-!CLOCK-1CLOCK-2

QBAR2=CLOCK-!CLOCK-2CLOCK1K1
```

TESTGN Usage

This subsection contains detailed descriptions of the input requirements of TESTGN and of the output produced.

Input Requirements

TESTGN can be loaded and executed under control of SIMLOG or can be separately loaded and executed if an appropriate data base of circuit description and simulation results is on file.

The following message is the first produced after TESTGN execution has begun.

```
SELECT TEST GENERATION MODE.  <CR>=1.
 (1) UNSPECIFIED FAULTS.
 (2) SPECIFIED FAULTS.

?
```

Mode 1 should be selected if the preceeding SIMLOG simulation was for a fault-free circuit. Mode 2 should be selected otherwise.

The circuit description is printed as shown below following mode selection.

```
A NAND J CLOCK QBAR
B NAND K CLOCK Q
C NAND A D
D NAND B C
E NAND CLOCK CLOCK
F NAND C E
G NAND D E
Q NAND F QBAR
QBAR NAND Q G

J
K
CLOCK

Q
QBAR

C D
Q QBAR
```

If mode 2 was selected, the following queries are printed.

```
ENTER THE FAULTED NET NAME?

ENTER THE FAULT CONDITION?
```

This concludes the input requirements of TESTGN.

Output Description

Mode 1 Test Generation

A test function is generated and printed for each input-output pair of the circuit. The test function for a given pair represents all test sequences of a fixed length that will test for a fault on the given input line when observed from the given output line. The test function is zero when no such test sequence is possible.

Given that the circuit of Figure 4 has been simulated by SIMLOG under fault-free conditions, input-time limit=2, and initial conditions A=B=C=F=G=Q=1 and D=E=QBAR=0, the following are partial results from the execution of TESTGN in mode 1.

```
OUTPUT-LINE=Q

INPUT-LINE=J

TEST-FUNCTION=0

OUTPUT-LINE=Q

INPUT-LINE=K

TEST-FUNCTION=CLOCK-2CLOCK1K1
              CLOCK-2CLOCK1K-1
```

Each line of the test function represents a separate test sequence. In general, two test sequences should be chosen from each test function. One test sequence should include the input equal to one while the other should include the input equal to zero. Such a choice assures that the input is tested for stuck-at-0 and stuck-at-1 conditions, respectively.

Mode 2 Test Generation

A test function is generated for each output of the circuit and indicates test sequences for the specified fault as observed from the given output. The following results for the example circuit with net E stuck-at-0.

```
OUTPUT-LINE=Q
FAULT=E/0
TEST-FUNCTION=CLOCK-2CLOCK1K1
```

REFERENCES

1. B. D. Carrull, "Test Pattern Generation," Final Report-Vol. 2, NAS8-31572, Electrical Engineering Department, Auburn University, Auburn, Alabama, September 14, 1979.

2. B. D. Carroll, "SIMLOG/TESTGN Programmer's Guide," Final Report-Vol. 2-Addendum 2, NAS8-31572, Electrical Engineering Department, Auburn University, Auburn, Alabama, September 14, 1979.

APPENDIX

DETAILED SIMULATION AND TEST PATTERN GENERATION EXAMPLE

```
ENTER SIMA
SELECT SIMULATION MODE.  <CR>=1.
    (1) NEW CIRCUIT
    (2) NEW FAULT
    (3) NEW STARTING STATE
    (4) CIRCUIT DESCRIPTION ON FILE
? 1
SELECT OUTPUT MODE.  <CR>=1.
     (1) PRINT STABLE EQUATION SETS ONLY
     (2) PRINT ALL EQUATION SETS
? 2
RACE ANALYSIS(YES OR NO).  <CR>=YES.?
ENTER--ELEMENT NAME, TYPE, AND INPUT LIST.
SEPARATE ENTRIES WITH A SPACE.
ENTER <CR> TO TERMINATE ENTRY.
? A NAND J CLOCK QBAR
? B NAND K CLOCK Q
? C NAND A D
? D NAND B C
? E NAND CLOCK CLOCK
? F NAND E C
? G NAND D E
? Q NAND F QBAR
? QBAR NAND G Q
?
ENTER PRIMARY INPUT LINE NAMES.  TERMINATE WITH <CR>.
? J
? CLOCK
? K
?
ENTER PRIMARY OUTPUT LINE NAMES.  TERMINATE WITH <CR>.
? Q
? QBAR
?
ENTER--CROSS-COUPLED ELEMENT PAIRS.  SEPARATE NAMES
WITH A SPACE.  ENTER <CR> TO TERMINATE.
? C D
? Q QBAR
?
FAULT SIMULATION(YES/NO).  <CR>=NO.?
ENTER STARTING STATE MODE SELECTION.  <CR>=1.

     (1) ALL UNKNOWNS
     (2) DOUBLE CROSS-COUPLED GATE VARIABLES
     (3) ALL VARIABLES
     (4) USER SPECIFIED
     (5) SINGLE CROSS-COUPLED GATE VARIABLE
     (6) SPECIFIED CONSTANTS
     (7) FIXED INPUT VALUES
     (8) MULTIMODE
? 6
```

```
ENTER--LINE NAME=0 OR 1--OR ENTER <CR> TO TERMINATE ENTRIES.
? A=1
ENTER--LINE NAME=0 OR 1--OR ENTER <CR> TO TERMINATE ENTRIES.
? B=1
ENTER--LINE NAME=0 OR 1--OR ENTER <CR> TO TERMINATE ENTRIES.
? C=1
ENTER--LINE NAME=0 OR 1-)OR ENTER <CR> TO TERMINATE ENTRIES.
? D=0
ENTER--LINE NAME=0 OR 1--OR ENTER <CR> TO TERMINATE ENTRIES.
? E=0
ENTER--LINE NAME=0 OR 1--OR ENTER <CR> TO TERMINATE ENTRIES.
? F=1
ENTER--LINE NAME=0 OR 1--OR ENTER <CR> TO TERMINATE ENTRIES.
? G=1
ENTER--LINE NAME=0 OR 1--OR ENTER <CR> TO TERMINATA ENTRIES.
? Q=1
ENTER--LINE NAME=0 OR 1--OR ENTER <CR> TO TERMINATE ENTRIES.
? QBAR=0
ENTER--LINE NAME=0 OR 1--OR ENTER <CR> TO TERIINATE ENTRIES.
?
EXIT SIMA


ENTER SIMB
ENTER INPUT-TIME LIMIT? 2
ENTER RIPPLE-TIME LIMIT? 6
```

T= 1 R= 1

A-1=0

A1=1

B-1=CLOCK1K1

B1=K-1+CLOCK-1

C-1=0

C1=1

D-1=1

D1=0

E-1=CLOCK1

E1=CLOCK-1

F-1=0

F1=1

G-1=0

G1=1

Q-1=0

Q1=1

QBAR-1=1

QBAR1=0

T= 1 R= 2

A-1=0

A1=1

B-1=CLOCK1K1

B1=K-1+CLOCK-1

C-1=0

C1=1

D-1=K-1+CLOCK-1

D1=CLOCK1K1

E-1=CLOCK1

E1=CLOCK-1

F-1=CLOCK-1

F1=CLOCK1

G-1=0

G1=1

Q-1=0

Q1=1

QBAR-1=1

QBAR1=0

T= 1 R= 3

A-1=0

A1=1

B-1=CLOCK1K

B1=K-1+CLOCK-1

C-1=CLOCK1K1

C1=K-1+CLOCK-1

D-1=K-1+CLOCK-1

D1=CLOCK1K1

E-1=CLOCK1

E1=CLOCK-1

F-1=CLOCK-1

F1=CLOCK1

G-1=0

G1=K-1+CLOCK1+CLOCK-1

Q-1=0

Q1=1

QBAR-1=1

QBAR1=0

T= 1 R= 4

A-1=0

A1=1

B-1=CLOCK1K1

B1=K-1+CLOCK-1

C-1=CLOCK1K1

C1=K-1+CLOCK-1

D-1=K-1+CLOCK-1

D1=CLOCK1K1

E-1=CLOCK1

E1=CLOCK-1

F-1=CLOCK-1

F1=CLOCK1

G-1=0

G1=K-1+CLOCK1+CLOCK-1

Q-1=0

Q1=1

QBAR-1=K-1+CLOCK1+CLOCK-1

QBAR1=0

```
T= 1          R= 5

A-1=0

A1=J-1+K-1+CLOCK1+CLOCK-1

B-1=CLOCK1K1

B1=K-1+CLOCK-1

C-1=CLOCK1K1

C1=K-1+CLOCK-1

D-1=K-1+CLOCK-1

D1=CLOCK1K1

E-1=CLOCK1

E1=CLOCK-1

F-1=CLOCK-1

F1=CLOCK1

G-1=0

G1=K-1+CLOCK1+CLOCK-1

Q-1=0

Q1=K-1+CLOCK1+CLOCK-1

QBAR-1=K-1+CLOCK1+CLOCK-1

QBAR1=0
```

T= 2 R= 6

A-2=0

A2=J-2+K-1+CLOCK1+CLOCK-1+CLOCK-2

B-2=CLOCK2K-1K2+CLOCK1CLOCK2K2+CLOCK-1CLOCK2K2

B2=K-2+CLOCK-2

C-2=CLOCK1K1

C2=K-1+CLOCK-1

D-2=K-1+CLOCK-1

D2=CLOCK1K1

E-2=CLOCK2

E2=CLOCK-2

F-2=CLOCK-1

F2=CLOCK1

G-2=0

G2=K-1+CLOCK1+CLOCK-1

Q-2=0

Q2=K-1+CLOCK1+CLOCK-1

QBAR-2=K-1+CLOCK1+CLOCK-1

QBAR2=0

T= 2 R= 7

A-2=0

A2=J-2+K-1+CLOCK1+CLOCK-1+CLOCK-2

B-2=CLOCK2K-1K2+CLOCK1CLOCK2K2+CLOCK-1CLOCK2K2

B2=K-2+CLOCK-2

C-2=CLOCK1K1

C2=K-1+CLOCK-1

D-2=K-1K-2+CLOCK-1K-2+CLOCK-2K-1+CLOCK-1CLOCK-2

D2=CLOCK1K1+CLOCK2K-1K2+CLOCK1CLOCK2K2+CLOCK-1CLOCK2K2

E-2=CLOCK2

E2=CLOCK-2

F-2=CLOCK-2K-1+CLOCK-1CLOCK-2

F2=CLOCK2+CLOCK1K1

G-2=CLOCK-2CLOCK1K1

G2=K-1+CLOCK2+CLOCK-1

Q-2=0

Q2=K-1+CLOCK1+CLOCK-1

QBAR-2=K-1+CLOCK1+CLOCK-1

QBAR2=0

T= 2 R= 8

A-2=0

A2=J-2+K-1+CLOCK1+CLOCK-1+CLOCK-2

B-2=CLOCK2K-1K2+CLOCK1CLOCK2K2+CLOCK-1CLOCK2K2

B2=K-2+CLOCK-2

C-2=CLOCK1K1+CLOCK2K-1K2+CLOCK1CLOCK2K2+CLOCK-1CLOCK2K2

C2=K-1K-2+CLOCK-1K-2+CLOCK-2K-1+CLOCK-1CLOCK-2

D-2=K-1K-2+CLOCK-1K-2+CLOCK-2K-1+CLOCK-1CLOCK-2

D2=CLOCK1K1+CLOCK2K-1K2+CLOCK1CLOCK2K2+CLOCK-1CLOCK2K2

E-2=CLOCK2

E2=CLOCK-2

F-2=CLOCK-2K-1+CLOCK-1CLOCK-2

F2=CLOCK2+CLOCK1K1

G-2=CLOCK-2CLOCK1K1

G2=CLOCK2+K-1K-2+CLOCK-1K-2+CLOCK-2K-1+CLOCK-1CLOCK-2

Q-2=0

Q2=K-1+CLOCK1+CLOCK-1

QBAR-2=K-1+CLOCK-1+CLOCK1CLOCK2

QBAR2=CLOCK-2CLOCK1K1

```
T=-2          R= 9

A-2=0

A2=J-2+K-1+CLOCK-1+CLOCK-2+CLOCK1CLOCK2

B-2=CLOCK2K-1K2+CLOCK1CLOCK2K2+CLOCK-1CLOCK2K2

B2=K-2+CLOCK-2

C-2=CLOCK1K1+CLOCK2K-1K2+CLOCK1CLOCK2K2+CLOCK-1CLOCK2K2

C2=K-1K-2+CLOCK-1K-2+CLOCK-2K-1+CLOCK-1CLOCK-2

D-2=K-1K-2+CLOCK-1K-2+CLOCK-2K-1+CLOCK-1CLOCK-2

D2=CLOCK1K1+CLOCK2K-1K2+CLOCK1CLOCK2K2+CLOCK-1CLOCK2K2

E-2=CLOCK2

E2=CLOCK-2

F-2=CLOCK-2K-1+CLOCK-1CLOCK-2

F2=CLOCK2+CLOCK1K1

G-2=CLOCK-2CLOCK1K1

G2=CLOCK2+K-1K-2+CLOCK-1K-2+CLOCK-2K-1+CLOCK-1CLOCK-2

Q-2=CLOCK-2CLOCK1K1

Q2=K-1+CLOCK-1+CLOCK1CLOCK2

QBAR-2=K-1K-2+CLOCK2K-1+CLOCK-1K-2+CLOCK-2K-1+CLOCK1CLOCK2
      +CLOCK-1CLOCK2+CLOCK-1CLOCK-2

QBAR2=CLOCK-2CLOCK1K1
```

T= 2 R= 10

A-2=0

A2=J-2+K-1K-2+CLOCK-2+CLOCK2K-1+CLOCK-1K-2+CLOCK1CLOCK2+CLOCK-1CLOCK2

B-2=CLOCK2K-1K2+CLOCK1CLOCK2K2+CLOCK-1CLOCK2K2

B2=K-2+CLOCK-2

C-2=CLOCK1J-2K1+CLOCK2K-1K2+CLOCK1CLOCK2K1+CLOCK1CLOCK2K2
+CLOCK-1CLOCK2K2+CLOCK-2CLOCK1K1

C2=K-1K-2+CLOCK-1K-2+CLOCK-2K-1+CLOCK-1CLOCK-2

D-2=K-1K-2+CLOCK-1K-2+CLOCK-2K-1+CLOCK-1CLOCK-2

D2=CLOCK1K1+CLOCK2K-1K2+CLOCK1CLOCK2K2+CLOCK-1CLOCK2K2

E-2=CLOCK2

E2=CLOCK-2

F-2=CLOCK-2K-1+CLOCK-1CLOCK-2

F2=CLOCK2+CLOCK1K1

G-2=CLOCK-2CLOCK1K1

G2=CLOCK2+K-1K-2+CLOCK-1K-2+CLOCK-2K-1+CLOCK-1CLOCK-2

Q-2=CLOCK-2CLOCK1K1

Q2=K-1K-2+CLOCK2K-1+CLOCK-1K-2+CLOCK-2K-1+CLOCK1CLOCK2+CLOCK-1CLOCK2
+CLOCK-1CLOCK-2

QBAR-2=K-1K-2+CLOCK2K-1+CLOCK-1K-2+CLOCK-2K-1+CLOCK1CLOCK2
+CLOCK-1CLOCK2+CLOCK-1CLOCK-2

QBAR2=CLOCK-2CLOCK1K1

```
T= 2          R= 11

A-2=0

A2=J-2+K-1K-2+CLOCK-2+CLOCK2K-1+CLOCK-1K-2+CLOCK1CLOCK2+CLOCK-1CLOCK2

B-2=CLOCK2K-1K2+CLOCK1CLOCK2K2+CLOCK-1CLOCK2K2

B2=K-2+CLOCK-2

C-2=CLOCK1J-2K1+CLOCK2K-1K2+CLOCK1CLOCK2K1+CLOCK1CLOCK2K2
     +CLOCK-1CLOCK2K2+CLOCK-2CLOCK1K1

C2=K-1K-2+CLOCK-1K-2+CLOCK-2K-1+CLOCK-1CLOCK-2

D-2=K-1K-2+CLOCK-1K-2+CLOCK-2K-1+CLOCK-1CLOCK-2

D2=CLOCK1J-2K1+CLOCK2K-1K2+CLOCK1CLOCK2K1+CLOCK1CLOCK2K2
     +CLOCK-1CLOCK2K2+CLOCK-2CLOCK1K1

E-2=CLOCK2

E2=CLOCK-2

F-2=CLOCK-2K-1+CLOCK-1CLOCK-2

F2=CLOCK2+CLOCK1J-2K1+CLOCK-2CLOCK1K1

G-2=CLOCK-2CLOCK1K1

G2=CLOCK2+K-1K-2+CLOCK-1K-2+CLOCK-2K-1+CLOCK-1CLOCK-2

Q-2=CLOCK-2CLOCK1K1

Q2=K-1K-2+CLOCK2K-1+CLOCK-1K-2+CLOCK-2K-1+CLOCK1CLOCK2+CLOCK-1CLOCK2
     +CLOCK-1CLOCK-2

QBAR-2=K-1K-2+CLOCK2K-1+CLOCK-1K-2+CLOCK-2K-1+CLOCK1CLOCK2
     +CLOCK-1CLOCK2+CLOCK-1CLOCK-2

QBAR2=CLOCK-2CLOCK1K1

DO YOU WISH TO ENTER A NEW TIME LIMIT--YES/NO, <CR>=NO?
GENERATE TEST SEQUENCES--YES/NO.  <CR>=NO.?
```

```
SELECT TEST GENERATION MODE.  <CR>=1.
        (1)     UNSPECIFIED FAULTS.
        (2)     SPECIFIED FAULTS.
?
A NAND J CLOCK QBAR
B NAND K CLOCK Q
C NAND A D
D NAND B C
E NAND CLOCK CLOCK
F NAND E C
G NAND D E
Q NAND F QBAR
QBAR NAND G Q

J
CLOCK
K

Q
QBAR

C D
Q QBAR
```

```
OUTPUT-LINE= Q
INPUT-LINE= J
TIME-TAG= 1
TEST-FUNCTION = 0

OUTPUT-LINE= Q
INPUT-LINE= J
TIME-TAG= 2
TEST-FUNCTION = 0

OUTPUT-LINE= Q
INPUT-LINE= CLOCK
TIME-TAG= 1
TEST-FUNCTION = CLOCK-2CLOCK1K1

                CLOCK-1CLOCK-2K1

OUTPUT-LINE= Q
INPUT-LINE= CLOCK
TIME-TAG= 2
TEST-FUNCTION = CLOCK1CLOCK2K1

                CLOCK-2CLOCK1K1

OUTPUT-LINE= Q
INPUT-LINE= K
TIME-TAG= 1
TEST-FUNCTION = CLOCK-2CLOCK1K1

                CLOCK-2CLOCK1K-1

OUTPUT-LINE= Q
INPUT-LINE= K
TIME-TAG= 2
TEST-FUNCTION = 0

OUTPUT-LINE= QBAR
INPUT-LINE= J
TIME-TAG= 1
TEST-FUNCTION = 0

OUTPUT-LINE= QBAR
INPUT-LINE= J
TIME-TAG= 2
TEST-FUNCTION = 0
```

```
OUTPUT-LINE= QBAR
INPUT-LINE= CLOCK
TIME-TAG= 1
TEST-FUNCTION = CLOCK-2CLOCK1K1

                CLOCK-1CLOCK-2K1

OUTPUT-LINE= QBAR
INPUT-LINE= CLOCK
TIME-TAG= 2
TEST-FUNCTION = CLOCK1CLOCK2K1

                CLOCK-2CLOCK1K1

OUTPUT-LINE= QBAR
INPUT-LINE= K
TIME-TAG= 1
TEST-FUNCTION = CLOCK-2CLOCK1K1

                CLOCK-2CLOCK1K-1

OUTPUT-LINE= QBAR
INPUT-LINE= K
TIME-TAG= 2
TEST-FUNCTION = 0

ENTER SIMA
SELECT SIMULATION MODE.  <CR>=1.
     (1) NEW CIRCUIT
     (2) NEW FAULT
     (3) NEW STARTING STATE
     (4) CIRCUIT DESCRIPTION ON FILE
? ^C

Ready
```