saturn history
3 27 68

No. 68-280

SATURN HISTORY DOCUMENT
University of Alabama Research Institute
History of Science & Technology Group

Date ---------- Doc. No. --------

X1.16

TEST PROCEDURE VALIDATION BY COMPUTER SIMULATION

by

ROBERT L. JAEGLY
Convair Division of
General Dynamics Corporation
Huntsville, Alabama

AIAA Paper
No. 68-280

AIAA 2nd Flight Test, Simulation and Support Conference

LOS ANGELES, CALIFORNIA/MARCH 25-27, 1968

First publication rights reserved by American Institute of Aeronautics and Astronautics, 1290 Avenue of the Americas, New York, N. Y. 10019.
Abstracts may be published without permission if credit is given to author and to AIAA. (Price: AIAA Member $1.00. Nonmember $1.50)

-- NOTES --

TEST PROCEDURE VALIDATION BY COMPUTER SIMULATION

Robert L. Jaegly
Convair Division of General Dynamics Corporation
Huntsville Operations
Huntsville, Alabama

Abstract

Digital computer simulation of the Saturn I Instrument Unit electrical networks was accomplished using the Discrete Network Simulation programs. The schematics were analyzed and a logic model prepared which consisted of a series of Boolean equations. The test procedures, which are written in the Acceptance, Test, or Launch Language (ATOLL), consist of a sequential set of computer instructions for the RCA 110A checkout computer to control the operation of the electrical networks. The procedures also contain the predicted results for each operation. The driving functions for the simulation of the model are generated from the ATOLL test tape by the Input Generator Program. The time sequenced operation of the networks is indicated by the output from the simulation program in addition to the number of times each component in the system changes state. The results of the simulation are compared to the test procedure predictions on the ATOLL tape by the Comparator Program and any differences are listed. The Comparator Program also lists any component which did not change state at least once.

An acceptance or checkout procedure is basically an operational sequence to demonstrate that a system or subsystem operates according to its design specifications. However, the procedure itself must first be verified that it activates the hardware in the intended manner.

To accomplish verification of the checkout procedure manually is both time consuming and tedious. The Convair division has developed a fast and accurate technique that does not require the use of a test vehicle or checkout equipment. This technique has been developed for verifying the test procedures used for the checkout of the Saturn 1B space vehicle Instrument Unit at the George C. Marshall Space Flight Center of the National Aeronautics and Space Administration. The total concept utilizes a set of computer programs called, Discrete Network Simulation (DNS), developed by the Convair division of General Dynamics. This methodology is particularly applicable to systems having operations that can be defined as a series of discrete events.

The Instrument Unit, or the so-called 'brains' of Saturn V - Apollo Launch Vehicle, is a cylindrical, ι pressurized structure located forward of the upperm S-IVB stage compartment. This unit houses the vehi electronics intelligence systems and components, wi each component or group of components provided wit separately controlled environment as required. The equipment located within the Instrument Unit include the guidance and control components, a portion of th tracking equipment, and the associated networks. A antennas are mounted on the outer surface of the Inst ment Unit skin. The Apollo payload is connected to t forward end of the Instrument Unit and the landing ge of the lunar excursion module extends through this u

A block diagram of the checkout system for the Instrument Unit is shown in Figure 1. The controlling element during automated systems testing is the RC/ 110A ground computer system. This computer syste is a single address, stored program, random access general purpose digital computer containing a 32K cc memory and a 32K drum. Peripheral devices includ five magnetic tape stations, a line printer, card rea card punch, and a paper tape reader/punch. A majo advantage of this computer system is its extensive ir output capability. Six Input/Output Data Channels (I(control data buffering and permit information transfe simultaneous with program execution.

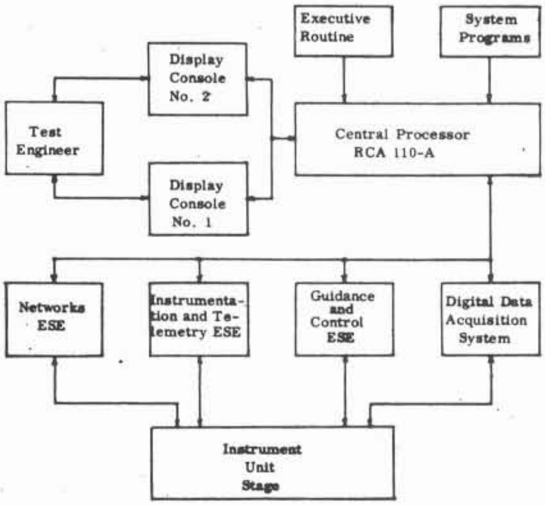


Figure 1. Block Diagram Saturn Instrument Unit Checkout System

Program control during automated testing is exercised by an Executive Checkout Monitor Program which controls the execution of test programs, allows limited on-line modifications to the programs and produces records of test actions and results. During test execution, the system provides numerous options to the test conductor. Test procedures are arranged by block, step, and sub-step. Blocks within a procedure will generally stand alone; that is, the system condition upon completion of the block is the same as it was prior to initiation of the block. Thus, within some procedures, testing may skip about from block to block, allowing a greater flexibility of testing operations. Test steps will usually perform some single operation. Within each step may be various sub-steps to initiate command, delay to allow stabilization, and scan all responses.

The test procedures which will be verified are actually part of the system programs as shown in Figure 1. These procedures are primarily used for the checkout of the Electrical Networks on the Instrument Unit. A special language developed by the Marshall Space Flight Center is used in their preparation. This is the Acceptance, Test or Launch Language (ATOLL) and the general methods for developing test procedures using ATOLL are shown in Figure 2.

The test procedure is prepared by test engineers on the ATOLL work sheet, the information is keypunched onto cards and loaded into the support computer with the ATOLL translator program. The support computer produces an ATOLL test procedure listing and the test information on magnetic tape. The information on the magnetic tape and the printed procedure is identical. It is this procedure which will be validated by using a computer simulation which is completely off line and independent of either the system hardware or the checkout computer.

Figure 3 is a typical page from the ATOLL test procedure for the Emergency Detection System on the Saturn Instrument Unit. This procedure is translated by the ATOLL translator compiler directly into a machine language that drives the checkout computer. The DISO operator is used by the procedure to turn on or off the discrete outs, or command signals from computer system, to the ESE or Instrument Unit. Thus, in Step 385, the DISO 1 instructs the computer to turn on discrete out number 163, which is the "command pad abort request".

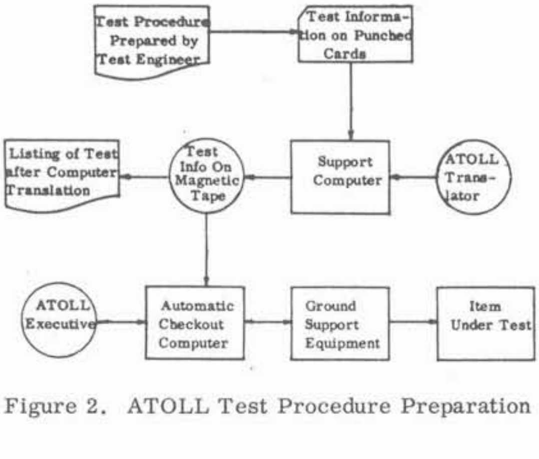

Test Procedure Prepared by Test Engineer
Test Information on Punched Cards
Listing of Test after Computer Translation
Test Info On Magnetic Tape
Support Computer
ATOLL Translator
ATOLL Executive
Automatic Checkout Computer
Ground Support Equipment
Item Under Test

Figure 2. ATOLL Test Procedure Preparation

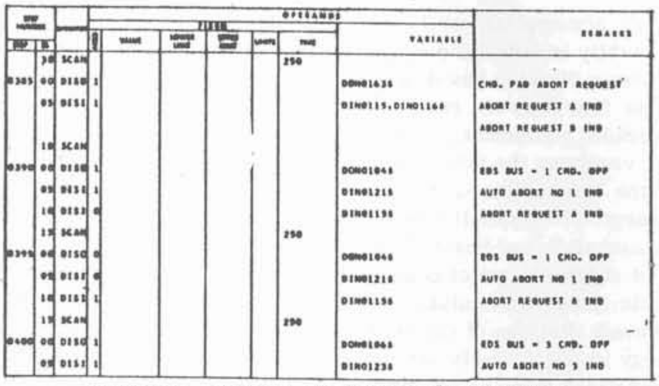
The computer utilizes two discrete value tables to continuously monitor the status of the system. In these tables each indicator in this system has a pre-defined address according to its number. The first table is the discrete input (DI) prediction table, which contains what the predicted status of each indicator should be at any point in the procedure. The other table is the actual discrete input table and contains the most recent value of each discrete indicator connected to the computer system.

The DISI command is used to update and modify the status of the DI prediction table. In Step 385, Sub-step 5, the DISI command instructs the computer to change the values in the prediction tables for number 115 and 116 from zero to one. These indicators are the Abort Request A, and Abort Request B, respectively.

The scan operator compares the values of the DI prediction table to the status of the actual indicators connected to this system. For any of the indicators in this system, a discrepancy between these two tables will cause an immediate halt in the procedure operation and revert to a semi-automatic mode of operation. Thus, while any single command to the electrical support equipment may only affect one or two indicators in this system, 1078 indicators that could be connected to the computer are checked each time the scan operator is used. Therefore, any unpredicted effect of the input will be observed.

Test procedure verification using the General Dynamics Convair division Discrete Network Simulation Program is diagrammatically shown in Figure 4. Using the schematics of the Instrument Unit and its associated electrical support equipment a logic model of the electrical networks is constructed. From the test procedure tape the commands to the stage and the electrical support equipment are abstracted and these, together with the model, make up the input to the General Dynamics Convair division Discrete Network Simulation Program.

The simulation program produces a time oriented history of the normal sequence of operation of the elec-



[illegible]		[illegible]	OPERANDS							REMARKS
			[illegible]							
[illegible]	[illegible]		[illegible]	VALUE	LOWER LIMIT	UPPER LIMIT	UNITS	TIME	VARIABLE	
	30	SCAN						250		
0385	00	DISO	1						DON01635	CMD. PAD ABORT REQUEST
	05	DISI	1						DIN0115,DIN01168	ABORT REQUEST A IND ABORT REQUEST B IND
	10	SCAN								
0390	00	DISO	1						DON01048	EDS BUS - 1 CMD. OFF
	05	DISI	1						DIN01215	AUTO ABORT NO 1 IND
	10	DISI	0						DIN01195	ABORT REQUEST A IND
	15	SCAN						250		
0395	00	DISO	0						DON01046	EDS BUS - 1 CMD. OFF
	05	DISI	0						DIN01218	AUTO ABORT NO 1 IND
	10	DISI	1						DIN01156	ABORT REQUEST A IND
	15	SCAN						250		
0400	00	DISO	1						DON01063	EDS BUS - 3 CMD. OFF
	05	DISI	1						DIN01238	AUTO ABORT NO 3 IND

Figure 3. ATOLL Test Procedure Listing

trical networks. The complete history of the normal network operation is used to validate the accuracy of the logic model. However, the checkout computer can only evaluate the results of inputs it has supplied in terms of discrete indicators -- DI's that are connected from this network back to the computer. Therefore, an edited version of the simulation history is produced which shows only the commands -- or discrete outputs -- from the computer and the results (DI's) that are transmitted back to the computer.

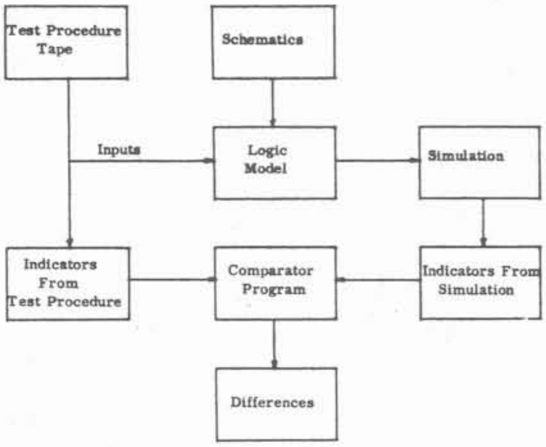
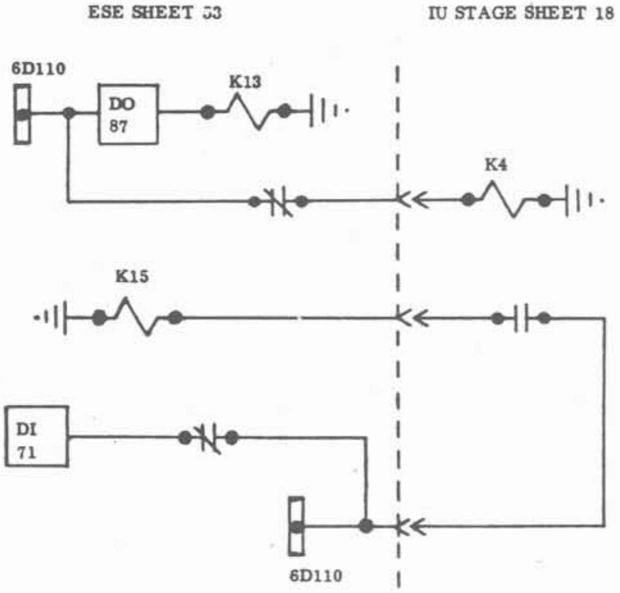


Figure 4. Test Procedure Verification by Discrete Network Simulation

The test procedure tape, in addition to the command functions, normally contains the predicted results for the procedure. These predicted results are abstracted from the test procedure tape and compared to the results of the discrete network simulation by a comparator program. Differences between predicted results in the test procedure and results of the simulation are printed out for each step and substep of the test procedure.



OFF = 0
ON = 1
* = AND
+ = OR
/ = NOT

53K13 = DO87 * 6D110.
18K4 = /53K13 * 6D110.
53K15 = 18K4 * 6D110.
DI71 = /53K15 * 6D110.

Figure 5. Logic Equations for DNS Model

Since the system simulation is for one specific purpose, certain assumptions can be made that materially reduce the size and complexity of the model. When the model is used for test procedure validation, it can be assumed that the hardware will function exactly as it was designed and documented on the schematics. Only those components that could actually change state under command of the checkout computer need be represented in the equations which describe the system. It is assumed that the contacts and coils of relays operate as a single unit, diodes do not become open circuits, and fuses do not burn out. The number of connectors in a circuit has no particular significance for this type of analysis. Since equations in the model and the resulting simulation will, in all probability, be used by relatively few people to validate accuracy of the simulation; the individual components in the model do not require a unique identity for the benefit of personnel not directly connected with the actual simulation. Therefore, a shorthand type nomenclature for system descriptions is used to reduce the model construction time.

Figure 5 shows the relationship between the electrical schematics and the equations which were constructed to represent the system. Indicators connected to the computer and command relays or DO's from the computer were identified by their computer number as indicated on the actual schematic, i.e., DI 71 and DO 87. Each component in the model is designated by the sheet number of the schematic on which they appear and the reference designator on that particular sheet, thus the relay in the ESE becomes 53K13 or 53K15 and the relay on the IU becomes 18K4.

Using logic equations, K13=1 when DO87=1 and the power is on. K4=0 if K13=1, K15=0 when K4=0 and DI71=1 when K15=0. In this manner, the complete electrical networks of the Instrument Unit and its associated ESE can be described.

Discrete Inputs (DI)	458
Discrete Outputs (DO)	283
RELAYS	900
NODES	41
FUSES	83
CABLES	48
BUSES	42
Switch Selector Channels	112
MISC.	358
TOTAL VARIABLES	2325
EQUATIONS IN MODEL	1752

Figure 6. Model of Instrument Unit Network

Figure 6 constitutes a summary of items included in the model of the Instrument Unit and its electrical support equipment to validate the electrical networks test procedure. This system was described with 2325 variables using 1752 equations.

Discrete Network Simulation

Discrete Network Simulation is a methodology which serves as an analytical tool capable of conducting a rapid and accurate analysis of a complex system. It requires a system model together with a set of computer programs to operate and activate the model. These programs are designed to provide a realistic analysis and prediction of system performance.

In this application of the methodology, the model equations along with the reaction times for each component are input to the preprocessor editor program which analyses the equations to establish the interrelationships of the model used in the simulation program.

The discrete network simulator chronologically simulates events that occur because of interactions among various elements in the model. Each 'event', or change of state, is a result of a logical cause and effect relationship among elements in the system. The system model for the simulation may be a switching circuit, a man-machine interaction, or any network whose components or subcomponents interrelations may be defined logically.

The DNS preprocessor editor program develops a series of tables that define the interdependent relationships of each active variable within a model. Self-checking diagnostic features are built into the program to ensure that every variable on the left hand side of an equation also appears on the right hand side of the equation unless it is identified as a terminal. It checks each variable to substantiate that activation times have been supplied and produces a listing of logic equations and timing information for reference.

Discrete Network Simulation Program

After the system model has been processed through the preprocessor editor program, system operation can be simulated using the DNS program. The model, as written, represents the system in a static condition. A set of drive functions (test procedures) is then required to establish the model in the initial condition for the simulation and to represent the activities to be simulated. The results of the simulation are recorded on the output tape. The printout includes:

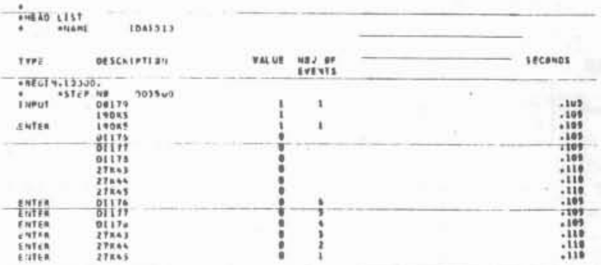
a) The order of events occurring.
b) The time of each occurrence.
c) A list showing the state of all variables at any selected time.

The program first establishes an initial condition for the state of the variables in the model. On the basis of this state, it examines all equations in the model and the logic predicts what variables will change state. The simulation represents real time; therefore, after the prediction has been made, the program looks up the activation times for the variables changing state, and imposes that time delay on the program before allowing the predicted changes to occur. This process is indicated on the printout. "Input" is the code word indicating that the state of this variable is being set by an external command in the input deck. The code word, "Enter" indicates that this variable is changing state due to the logic of the equations combined with the computer program.

This simulation represents real time. The time of the simulation is printed in the columns on the right hand side of the sheet and as indicated by the headings, can be described in days, hours, minutes, or seconds. The seconds are resolved down to the nearest millisecond. The time printed for each activity line represents the actual time for the activity to occur.

Figure 7 is a typical printout from the simulation program. The printout shows the complete operational history of the system being simulated. The procedure identification is IDA 5010. At step number 5, substep 00, when DO 179 is input to 1 it predicts that relay 190K5 will come on. Relay 190K5 comes on and predicts that DI's 176, 177, and 178; and relays 27K43, 44, and 45 will turn off. Following the predictions, the actual events take place and the absence of any predicted results show that no further activities take place as the result of DO 179 being turned on.

An edited printout of the simulation history is shown in Figure 8. At step 930, substep 0 of the Emergency Detection System test procedure DO 215 was turned off at 8.825 seconds time in the test procedure. The printout shows the results of this input in terms of which discrete inputs (DI) changed state as the result of DO 215 being turned off. In this case, there were 15 DI's that turned on as the result of one input. This edited printout, which lists only the information referenced in the test procedure, is used in the comparator program. The simulation history indicates that there are 51 elements or components in this system which changed state as a result of this one input.

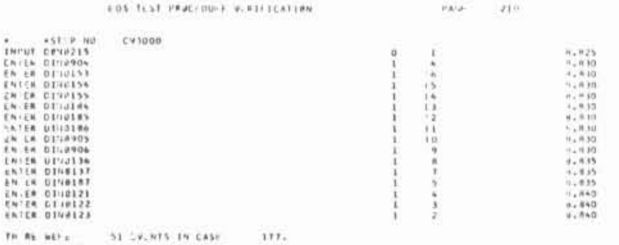


```
*
*HEAD LIST
*     *NAME      IDA1513

TYPE      DESCRIPTION          VALUE  NBJ OF                    SECONDS
                                      EVENTS
*BEGIN.13300.
*     *STEP NO    5005W0
INPUT     DO179                  1      1                          .103
          190K5                  1                                 .109
ENTER     190K5                  1      1                          .109
          DI175                  0                                 .109
          DI177                  0                                 .109
          DI178                  0                                 .109
          27K43                  0                                 .110
          27K44                  0                                 .110
          27K45                  0                                 .110
ENTER     DI176                  0      6                          .109
ENTER     DI177                  0      5                          .109
ENTER     DI178                  0      4                          .109
ENTER     27K43                  0      3                          .110
ENTER     27K44                  0      2                          .110
ENTER     27K45                  0      1                          .110
THERE WERE      8 EVENTS IN CASE    49.
```

Figure 7. Simulation History Printout

At the end of the simulation of each section of the procedure, the simulation program is commanded to printout a list showing the status (on or off) of each of the components (variables) in the system; an example is shown in Figure 9. This indicates, at the end of the procedure, which relays were on and which were off at this point in the procedure. In the simulation program, a record has been kept of the history of the activities of each of the variables in the system, as shown in the right hand column. A zero in this column indicates that these relays have not been turned on or off in this portion of the procedure. It also shows that relay 602A5K69 was turned on or off 130 times during this procedure. Each count represents a half cycle which is due to that element being turned on or off. If the figure in this counter happens to be an odd number (such as the occasion for item 163 relay A5K13) it indicates that this relay is left in the opposite state at the end of the procedure compared to what it was at the beginning of the procedure.



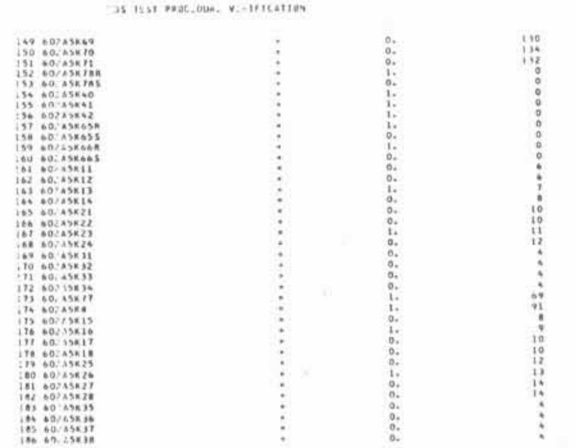
```
                 EOS TEST PROCEDURE VERIFICATION                    PAGE     210

*       *STEP NO     CV1000
INPUT  DI00215                                          0     1                  0.825
ENTER  DI00904                                          1     4                  0.830
ENTER  DI00153                                          1     6                  0.830
ENTER  DI00156                                          1    15                  0.830
ENTER  DI00155                                          1    14                  0.830
ENTER  DI00146                                          1    13                  0.835
ENTER  DI00185                                          1    12                  0.830
ENTER  DI00186                                          1    11                  0.830
ENTER  DI00905                                          1    10                  0.830
ENTER  DI00906                                          1     9                  0.830
ENTER  DI00136                                          1     8                  0.835
ENTER  DI00137                                          1     7                  0.835
ENTER  DI00187                                          1     5                  0.835
ENTER  DI00121                                          1     4                  0.840
ENTER  DI00122                                          1     3                  0.840
ENTER  DI00123                                          1     2                  0.840

THERE WERE       51 EVENTS IN CASE       177.
```

Figure 8. Edited Simulation History Printout

From these examples it can be seen that considerable information about the affect of a given test procedure upon the hardware or system under analysis can be deduced by careful evaluation of results of the system simulation -- provided a carefully prepared model is used.



```
                 DS TEST PROCEDURE VERIFICATION

149  60/A5K69                         *              0.                130
150  60/A5K70                         *              0.                134
151  60/A5K71                         *              0.                132
152  60/A5K78R                        *              1.                  0
153  60/A5K78S                        *              0.                  0
154  60/A5K40                         *              1.                  0
155  60/A5K41                         *              1.                  0
156  60/A5K42                         *              1.                  0
157  60/A5K65R                        *              1.                  0
158  60/A5K65S                        *              0.                  0
159  60/A5K66R                        *              1.                  0
160  60/A5K66S                        *              0.                  0
161  60/A5K11                         *              0.                  6
162  60/A5K12                         *              0.                  6
163  60/A5K13                         *              1.                  7
164  60/A5K14                         *              0.                  8
165  60/A5K21                         *              0.                 10
166  60/A5K22                         *              0.                 10
167  60/A5K23                         *              1.                 11
168  60/A5K24                         *              0.                 12
169  60/A5K31                         *              0.                  4
170  60/A5K32                         *              0.                  4
171  60/A5K33                         *              0.                  4
172  60/A5K34                         *              0.                  4
173  60/A5K77                         *              1.                 69
174  60/A5K8                          *              1.                 91
175  60/A5K15                         *              0.                  8
176  60/A5K16                         *              1.                  9
177  60/A5K17                         *              0.                 10
178  60/A5K18                         *              0.                 10
179  60/A5K25                         *              0.                 12
180  60/A5K26                         *              1.                 13
181  60/A5K27                         *              0.                 14
182  60/A5K28                         *              0.                 14
183  60/A5K35                         *              0.                  4
184  60/A5K36                         *              0.                  4
185  60/A5K37                         *              0.                  4
186  60/A5K38                         *              0.                  4
```

Figure 9. State List from Discrete Network Simulation

Discrete Network Simulation versus ATOLL Test Procedure Comparison

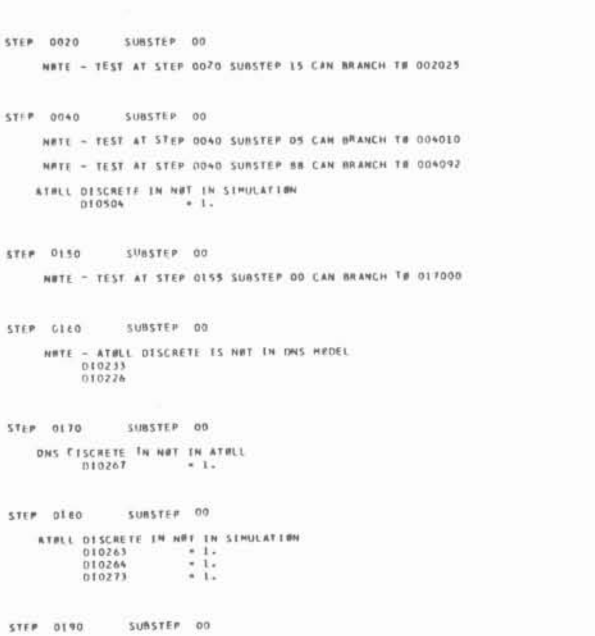
A typical printout from the ATOLL test procedure comparator program is shown in Figure 10 and presents examples of the information displayed about the procedure by this comparison. At steps 20, 40, and 150, it is indicated that the procedure contains a 'TEST' operator which gives the procedure the option of branching to a specified step and substep rather than following the normal numerical sequence (depending upon the status of a specified DI at that step in the procedure). In this application, the model contains only the elements that represent actual hardware components and the test procedure represents the driving functions for the model. When the driving functions for the model are extracted from the test procedure tape, the procedure is assumed to run in a normal sequence with no branching. In the case that branching occurs in accordance with the status of the system, then the results of this comparison would be invalidated. This eventuality is printed out in the comparator program as a warning.

At step 160 the ATOLL program is checking the status of the discretes DI 233 and 226 which were not included in the simulation model, probably because they are connected to some system outside the scope of the existing simulation.

At step 170 the simulation model indicates a change of state for DI 267 that was not predicted in the ATOLL procedure. This could be due to an oversight during the preparation of the test procedure or because the procedure writer did not consider the change of state of this indicator at this step to have any particular significance.

In step 180, the test procedure predicted that the three discretes 263, 264, and 273 would come on at this point. Since they did not come on in the simulation a discrepancy exists between the simulation and the procedure. In this case, it could be due to an error in the procedure, an error in the schematic drawings, or an error in the preparation of the logic model. Therefore, additional investigation and analysis is required to resolve the cause of the discrepancy.

As a part of the comparator program output, the state list at the end of the simulation is analyzed and all those components which have not been activated at least once in the test procedure are listed at the end of the comparator output. From this list it can be seen at a glance how much of the system has been exercised by a given procedure.



```
          DISCRETE NETWØRK SIMULATIØN VS ATØLL TEST PRØCEDURE CØMPARISØN

               IRN TEST PRØCEDURE  IDA1003


STEP  0020        SUBSTEP  00

     NØTE - TEST AT STEP 0020 SUBSTEP 15 CAN BRANCH TØ 002025


STEP  0040        SUBSTEP  00

     NØTE - TEST AT STEP 0040 SUBSTEP 05 CAN BRANCH TØ 004010

     NØTE - TEST AT STEP 0040 SUBSTEP 88 CAN BRANCH TØ 004092

    ATØLL DISCRETE IN NØT IN SIMULATIØN
          DI0504          * 1.


STEP  0150        SUBSTEP  00

     NØTE - TEST AT STEP 0155 SUBSTEP 00 CAN BRANCH TØ 017000


STEP  0160        SUBSTEP  00

     NØTE - ATØLL DISCRETE IS NØT IN DNS MØDEL
          DI0233
          DI0226


STEP  0170        SUBSTEP  00

    DNS DISCRETE IN NØT IN ATØLL
          DI0267          * 1.


STEP  0180        SUBSTEP  00

    ATØLL DISCRETE IN NØT IN SIMULATIØN
          DI0263          * 1.
          DI0264          * 1.
          DI0273          * 1.


STEP  0190        SUBSTEP  00
```

Figure 10. DNS Comparator Printout

Thus, by the use of Discrete Network Simulation a detailed analysis of the interface between the hardware system and the test procedure can be conducted. The analysis will show to what degree a test procedure will exercise a system and also to what level the system will be exercised. It can also indicate that the specified inputs will not activate the hardware system the way that the procedure writer anticipated.

In addition, the simulation will indicate any errors in the schematic drawings of the electrical networks or (if the schematic drawings are drawn correctly) that the hardware system does not respond in the manner that was intended. In the past, this type of analysis has been conducted manually by tracing through drawings to ascertain the affect of each input specified in the procedure to see what effects they have on the system using this methodology. A detailed analysis of test procedures can be performed with considerable savings in manhours and with an accuracy and depth of analysis not previously possible using manual analytical methods.

Acknowledgement

The application of the Discrete Network Simulation programs to Test Procedure Validation represents a General Dynamics Convair group effort. The author would like to express his appreciation to the following individuals for their support and contributions to this study.

J. H. Newton, NASA, MSFC
W. Finnell, NASA, MSFC

A. R. Stone
J. W. Barnes
T. C. Larson
R. D. Ritchey

The study reported in this paper was performed in support of the Marshall Space Flight Center, Quality and Reliability Assurance Laboratory under NASA Contract NAS8-21102.