NASA Contractor Report 178390

A FAULT INJECTION EXPERIMENT USING THE AIRLAB DIAGNOSTIC EMULATION FACILITY

Robert Baker, Scott Mangum, and Charlotte Scheper

Center for Digital Systems Research
Research Triangle Institute
Research Triangle Park, North Carolina 27709

Contract NAS1-17964
Task Assignment No. 5
March 1988

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665

NASA Contract Report 178390

A FAULT INJECTION EXPERIMENT USING THE AIRLAB DIAGNOSTIC EMULATION FACILITY

Robert Baker
Scott Mangum
Charlotte Scheper

Center for Digital Systems Research
Research Triangle Institute
Research Triangle Park, North Carolina 27709

Contract NAS1-17964
Task Assignment No. 5

March 1988

TABLE OF CONTENTS

LIST OF FIGURES AND TABLES

1. Introduction and Scope

Digital flight control systems for aircraft and spacecraft perform life or mission critical functions. Extremely high reliability requirements must be established and demonstrated for these systems. To meet the reliability and performance requirements, systems become complex. Complexity and demanding requirements combine to render the validation of system reliability difficult. Testing, sufficient to establish the high reliability of these systems, is not feasible. Consequently, sophisticated reliability modeling tools based on analytic models are needed to predict and validate reliability for both the design and development phases of fault tolerant systems. Reliability modeling tools depend upon the accurate determination of various model parameters such as fault coverage and fault latency. When system testing to determine such parameters is precluded, computer simulations can be used in some circumstances to *stand-in* for a system. This report describes the preparation for, conduct of, and results of a simulation-based fault injection experiment conducted using the AIRLAB Diagnostic Emulation (DE) facilities.

The primary objective of this experiment was to determine the effectiveness of the diagnostic self test sequences used to uncover latent faults in a logic network providing the key fault tolerance features for a flight control computer. Since this experiment resulted in the most extensive use of the AIRLAB Diagnostic Emulation facilities to date, a secondary, but essential, objective was to develop methods, tools, and techniques for conducting the experiment.

In this experiment, more than 1600 faults were injected into a logic gate level model of the Data Communicator/Interstage (C/I), a key component in the Charles Stark Draper Laboratories (CSDL) Fault Tolerant Processor (FTP). For each fault injected, diagnostic self test sequences consisting of over 300 test vectors were supplied to the C/I model as inputs. For each test vector within a test sequence, the outputs from the C/I model were compared to the outputs of a fault free C/I. If the outputs differed, the fault was considered detectable for the given test vector. These results were then analyzed to determine the effectiveness of various test sequences, to identify latent faults, and to identify opportunities for improving the performance of the diagnostic test sequences. Figure 1.1 diagrams the essential elements of the experiment design.

The Data Communicator/Interstage of the FTP was selected because of the following:

1. the C/I network provides for the critical fault tolerance features of input source congruency, error masking, error reporting and system reconfiguration,

2. the quadriplex C/I network used the full capacity of the AIRLAB Diagnostic Emulator,

3. documentation for the C/I design at the logic gate level was available, and

4. the results of the experiment could provide valuable information about the FTP design.

To prepare for and complete this experiment, several major issues had to be considered. Chief among these were design capture for a complex network; validation of the model derived from the design capture; analyses and decisions that reduced the required simulation time while preserving essential information; the design of test sequences for a complex state network; the applicability of fault models for LSI technology; the complexity management issues associated with modeling, validating, and simulating a large network; CAD tools and procedures to handle the complexity management problems; techniques to maintain data integrity for extended simulations; techniques for improving fault coverage and latency; and performance measures and techniques for analyzing the effectiveness of test vectors. Each of these issues can develop into significant problem areas as system complexity increases with the advancement of integrated circuit and computer architecture technology. Specifically, simulation methods, techniques, and associated tools will have to advance to meet the demands of computer technology. Moreover, validation research in general will be significantly impacted by some of these issues.

Section 2 of this report provides an overview of the C/I design and discusses the gate model used for various parts of the C/I. Section 3 describes the various activities involved in the experiment preparation process. Among the activities discussed are design capture, test sequence design, and model validation. In section 4, the conduct of the experiment is reviewed and the results are presented and analyzed. Section 5 provides a summary of experiment results, a review of lessons learned from conducting the experiment and a discussion of extensions to this experiment.

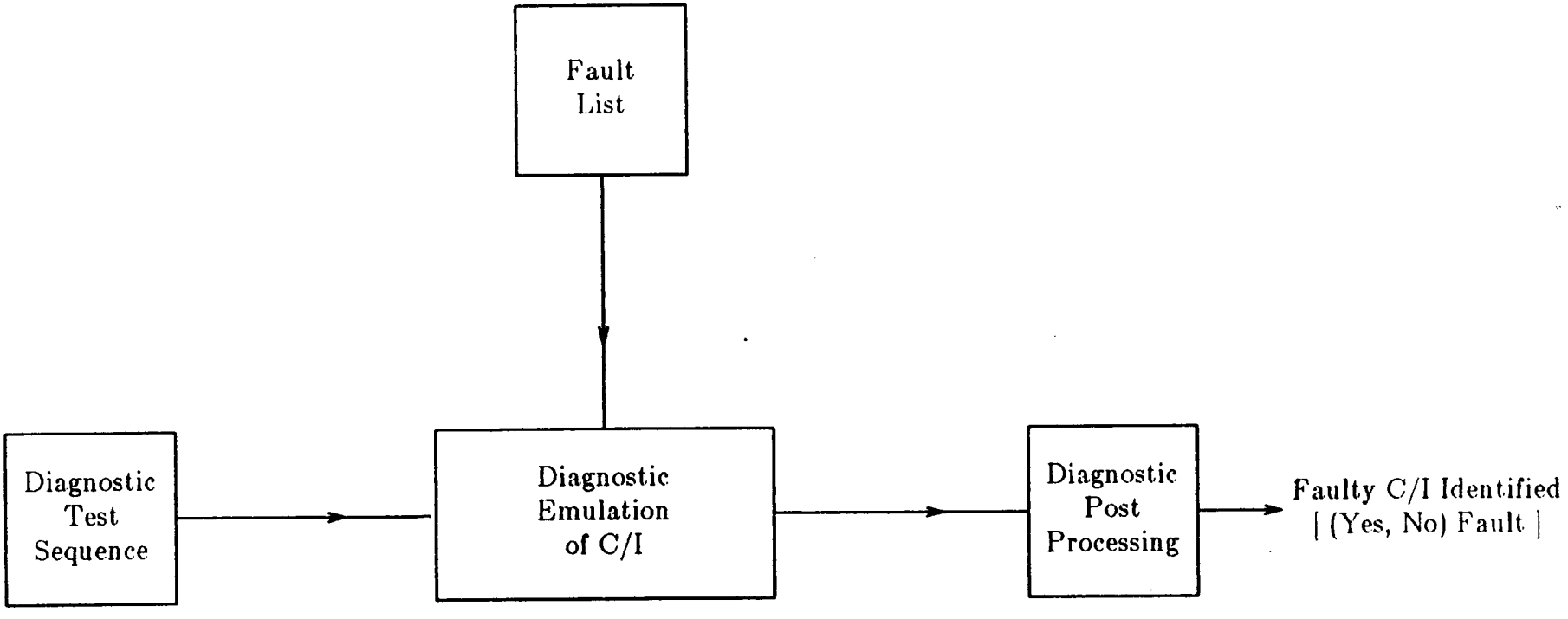


Figure 1.1. Experiment Diagram

2. Overview of the Data Communicator/Interstage and the Associated Gate Level Model

2.1. Functional Description[1] [2] [3]

The Data Communicator/Interstage (C/I), which was modeled in this experiment, provides the Fault Tolerant Processor (FTP) with certain key fault tolerance mechanisms. These mechanisms include circuitry to support the correct replication of simplex source data in all good FTP channels, to detect and mask errors in a single FTP channel, and to support FTP reconfiguration by blocking out faulty channels.

Figure 2.1 shows a quadriplex FTP.

Figure 2.2 shows the data exchange network associated with a quadriplex FTP. Each channel, A, B, C, or D, has a transmit and receive register which can be accessed by the input/output processor (IOP) or computational processor (CP) via a shared memory bus.

Simplex source data, that is, data available to one FTP channel may be distributed correctly to all good FTP channels by using a FROM *X* exchange. Referring to figure 2.2, a FROM A exchange, for example, will cause the data word in the communicator transmit register of channel A to be distributed to all other data communicators via the cross channel links. Data transfers between communicators and interstages are byte serial with two bytes comprising a data word. Data distributed to each communicator are passed to associated interstages.

Each interstage replicates the data and passes a copy to each communicator. Each communicator votes the data received from each interstage and writes the voted data into the Receive Register. The Receive Registers are then read by the associated channel processor. In this matter, simplex source data from channel A is replicated in all good processor channels. FROM B, FROM C, and FROM D exchanges occur in a similar manner except that the source channel is B, C, or D respectively.

If each channel has data which requires voting for fault masking, a FROM OWN exchange can be executed. Each channel places its copy of the data to be voted into its respective Transmit Registers. Data words are transmitted to the respective interstages which in turn replicate the data and distribute the copies to each communicator. Each communicator votes all copies of the data and returns the voted result to the Receive Register for subsequent reading by its associated computer.

Errors detected during the voting process of an exchange are recorded in the Status Registers of each communicator. The Status Registers of each communicator can be read by their associated processors. It is intended that these error reports be used by the FTP to determine and isolate faulty channels.

A write only Mask Register allows a channel to be excluded from the voting process.

The capability to execute exchanges which by-pass the interstage and voter is provided for network testing.

FAULT TOLERANT PROCESSOR
(Quadriplex Version)

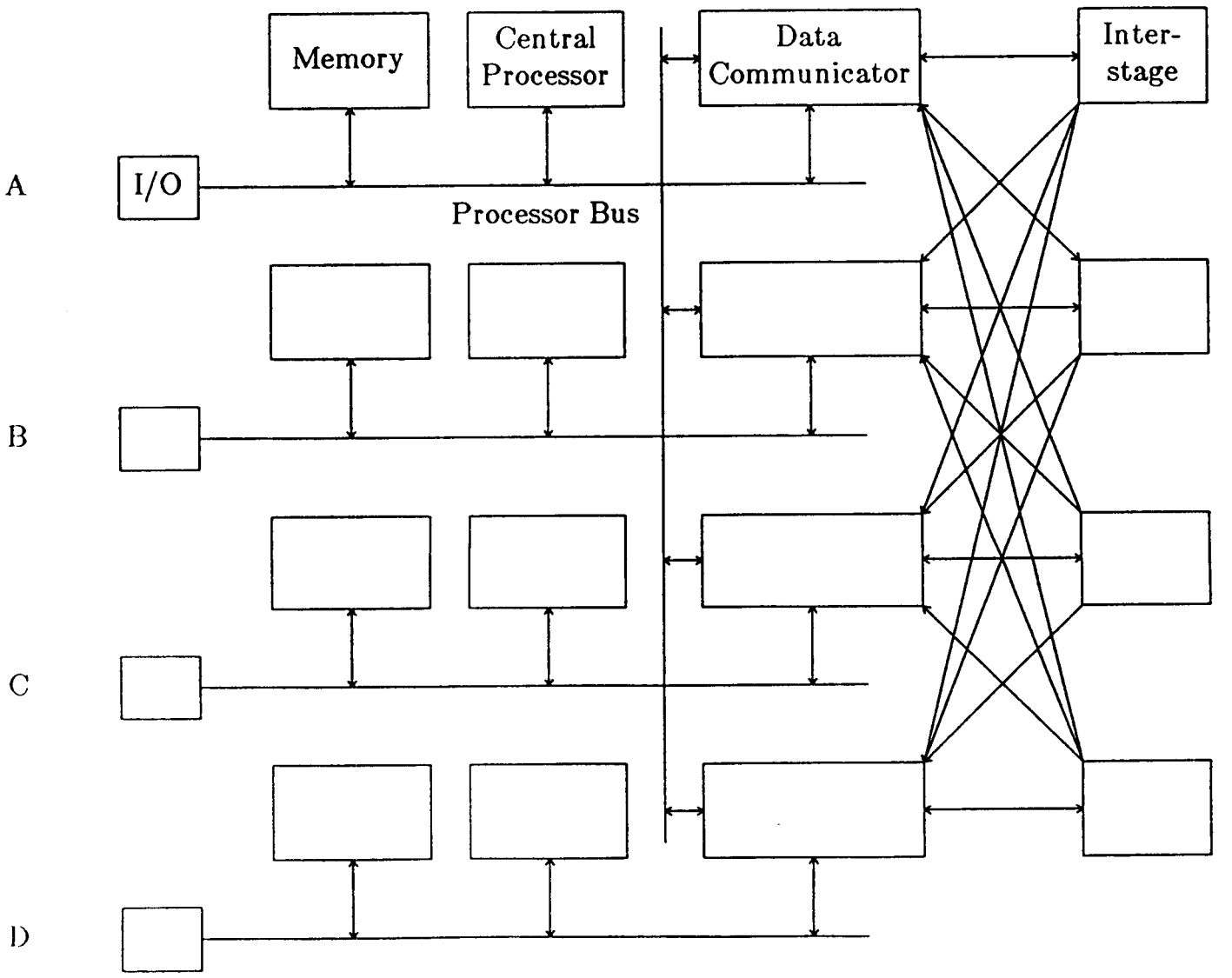


Figure 2.1

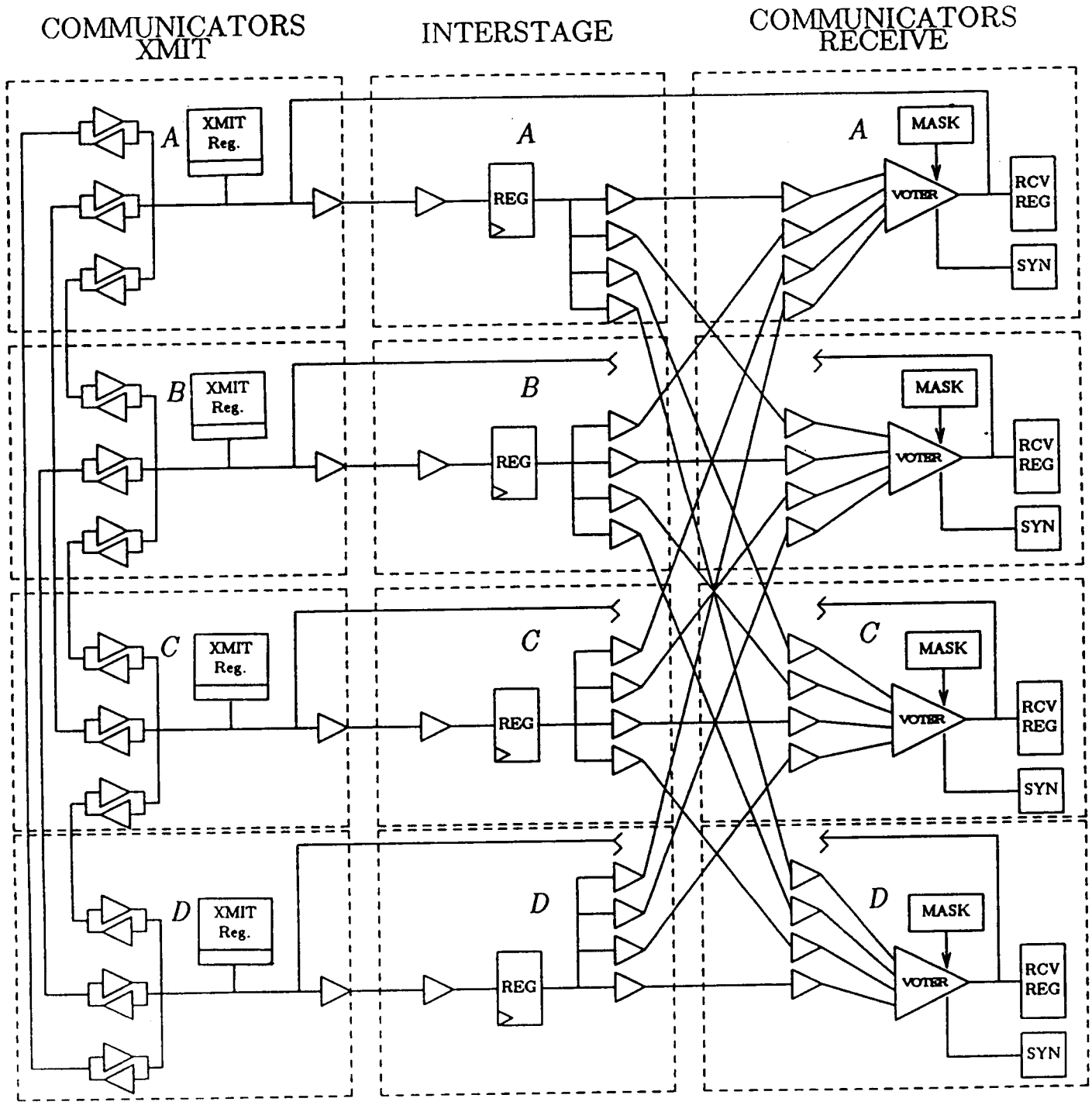

C/I EXCHANGE NETWORK
COMMUNICATORS XMIT
INTERSTAGE
COMMUNICATORS RECEIVE
A
XMIT Reg.
REG
MASK
VOTER
RCV REG
SYN
B
C
D

Figure 2.2

Table 2.1 lists the transactions on operating that can be executed by the communicator.

FROM A Exchange
FROM B Exchange
FROM C Exchange
FROM D Exchange
FROM OWN Exchange
READ Receive Register
WRITE Receive Register
WRITE Mask Register
READ FROM A Status Register
READ FROM B Status Register
READ FROM C Status Register
READ FROM D Status Register
READ FROM OWN Status Register

Table 2.1. Data Communicator Operations

2.2. Circuit Description and Modeling Issues

Figure 2.3 is a functional block diagram for the C/I. Each area and the associated gate level models will be described in the following paragraphs.

The processor interface consists of receivers for the 20 bit processor address bus; 16 bit bidirectional tri-state data bus driver/receivers; and receivers or drivers as required for processor control signals, data acknowledge, read/write, system reset, system clock and fault tolerant clock.

For purposes of simplifying the simulation model, the processor address bits associated with the base address for the memory mapped data communicator were consolidated into a single communicator select bit. This consolidation reduced the size of the input files for the simulation and speeded the simulation run time. A single *and* gate (address decoder) and the several address bit receivers were modeled as a single receiver. No essential fault coverage information was lost by this simplification.

The bidirectional processor data bus was modeled as separate input and output ports because the Diagnostic Emulator had capability for either input or output connections. No essential fault coverage information was lost due to this modeling restriction.

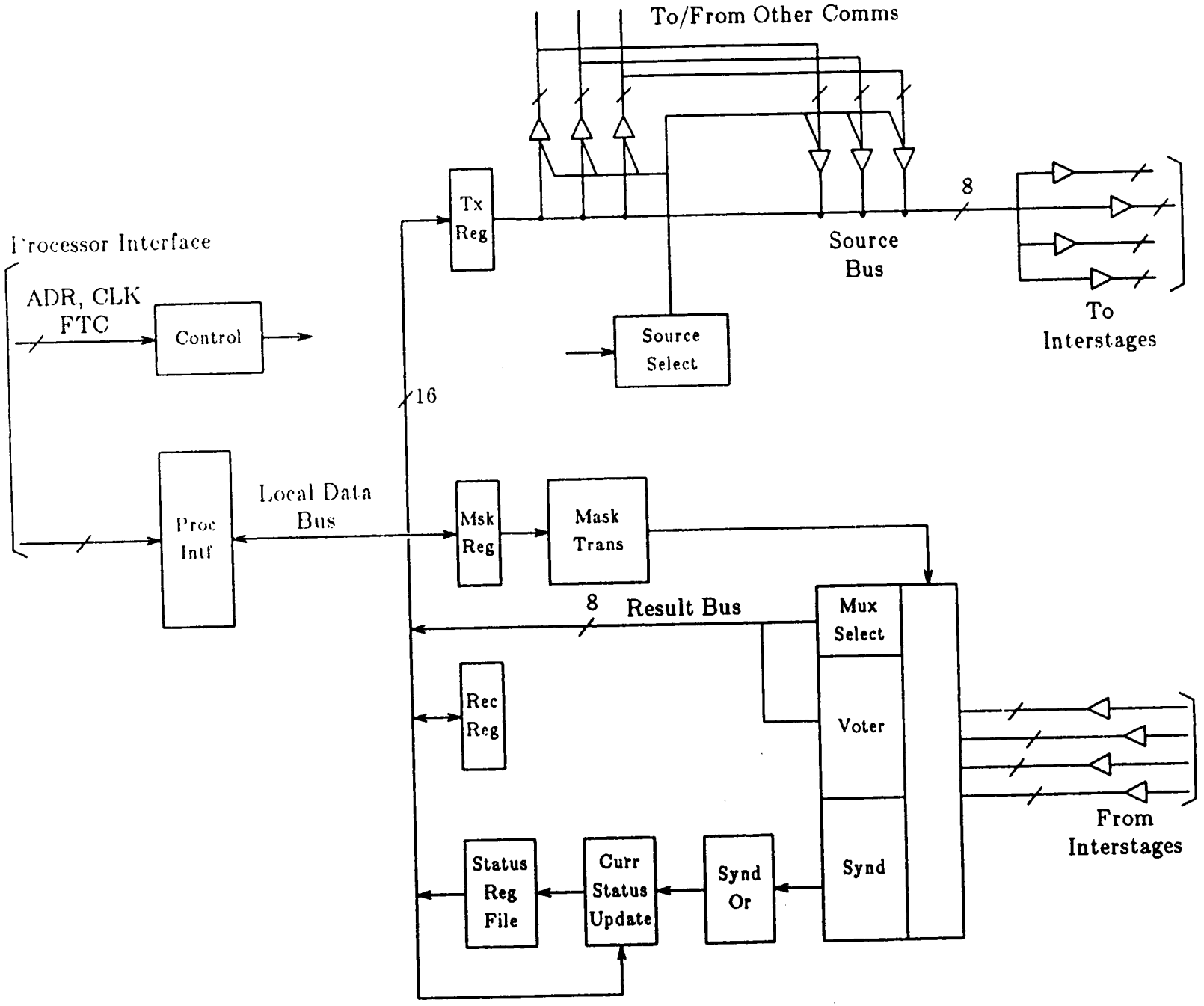


Figure 2.3. FTP Communicator

The Local Data Bus is a 16 bit tri-state bus used to communicate as follows:

From Processor Bus to Transmit Register

From Processor Bus to Mask Register (5 bits)

From Processor Bus to Receive Register

From Receive Register to Processor Bus

From Status Register (5 bits) to Processor Bus

From Status Register (5 bits) to Current Status Update Logic

From Result Bus to Receive Register Most Significant Byte

From Result Bus to Receive Register Least Significant Byte

From Transmit Register Most Significant Byte to Transmit Register Least Significant Byte

This bus was modeled by use of *and* gates at each data source, enabled by the appropriate source enable signal, and by an *or* gate to combine the various data sources to one signal line. The resulting model was functionally equivalent to a tri-state bus. The *or* function is a phantom gate in that there is no identifiable device in the network to which it can be associated. Despite its *phantom* nature, *stuck-at zero* faults were asserted on these devices during simulation to cover certain kinds of bus faults.

The Transmit Register, Mask Register and Receive Register are all implemented with 74LS374 octal D-type latches with tri-state outputs. The storage portion of these devices were modeled directly as D-type flip-flops and the tri-state output portion were modeled as *and* gates feeding an appropriate bus *or* gate as indicated in the previous paragraph.

The Source Bus communicates data bytes from the Transmit Register to cross channel links and to the interstages. Also, data bytes from cross channel links can be transferred to the interstages. This bus was modeled with *and* gates and phantom *or* gates similar to the Local Data Bus.

The cross channel links which distributes data from each communicator Source Bus to all other communicators' source busses is a bidirectional tri-state bus. The circuitry is diagrammed in figure 2.4. An adequate model for this is shown in figure 2.5. Since this model required a significant number of connections (384) and phantom gates (96) and since these particular connections had to be hand edited into the network, the simpler model (192 connections and 0 gates) shown in figure 2.6 was used. This decision was based on incomplete analysis which indicated that the model adequately

captured the network behavior. While data is handled properly by this model, the effects of faults in the Source Select logic which supplies the enable signals for the cross channel links were not accurately modeled. Consequently, certain gate faults were not found by the C/I diagnostics. As discussed in a later chapter, these faults would have been found had the more complex and accurate model been used for the cross channel links.

The States Register file is implemented with two 74LS189 4×16 bit RAM chips. Only 5 status bits, A error, B error, C error, D error and Quad error, are recorded in each of the 5 exchange types. Thus, only 5×5 of the potential 8×16 bits are used for the Status Register file. The Status Register was modeled with five, five-bit D-type flip-flop registers and with the address decoder and selection logic necessary to direct five bits of data to and from these registers.

The Mask Transform block receives the Mask Register bits (Mask A, Mask B, Mask C, Mask D, and Source Lock), the two channel identification (ID) bits, and the three exchange type bits to determine which, if any, channel to block from the voter logic and to select the source of data to be routed to the Receive Register. This function is implemented with a 2048×8-bit registered programmable read only memory. To model this device directly required that a Diagnostic Emulator overhead feature be used. This feature assigns a block of memory outside normal device simulation which can be used to store the contents of the memory being modeled. Memory address and data bit connections are associated with and determined by designated nodes within the network being simulated. To access this memory, a change on a designated control line will stop normal simulation of the network and will invoke an action which uses the state of the designated address nodes to select a particular location in the assigned memory block and to impose the state of the bits in this memory location on the designated data bit nodes.

Since this operation is generally slower than gate simulation and since validation of the operation of this PROM was judged to be more difficult than validating an equivalent gate network for this case, a decision was made to model this block with a simple gate network at least until the rest of the C/I network could be validated.

The interstages consist of drivers, receivers, and D-type flip-flops. These devices were modeled using the Diagnostic Emulator gate and flip-flop primitives.

The Source Select, Current Status Update, Syndrome Or, Voter, and Voter Syndrome blocks are implemented using programmable logic arrays (PLA). A diagram of a typical PLA is shown in figure 2.7. The logic equations programmed for each function were modeled directly. Figure 2.8 illustrates one of these functions. The adequacy of *stuck-at* gate fault models for these devices is questionable since certain bridging faults between conductors carrying the input functions and a cell with the array cannot be accurately represented by such a simple model. A more elaborate gate model which would better represent these faults was beyond the scope of this effort.

The Source Select block shown in figure 2.3 uses exchange type bits and channel ID bits to control the cross channel links. The Voter and Voter Syndrome block

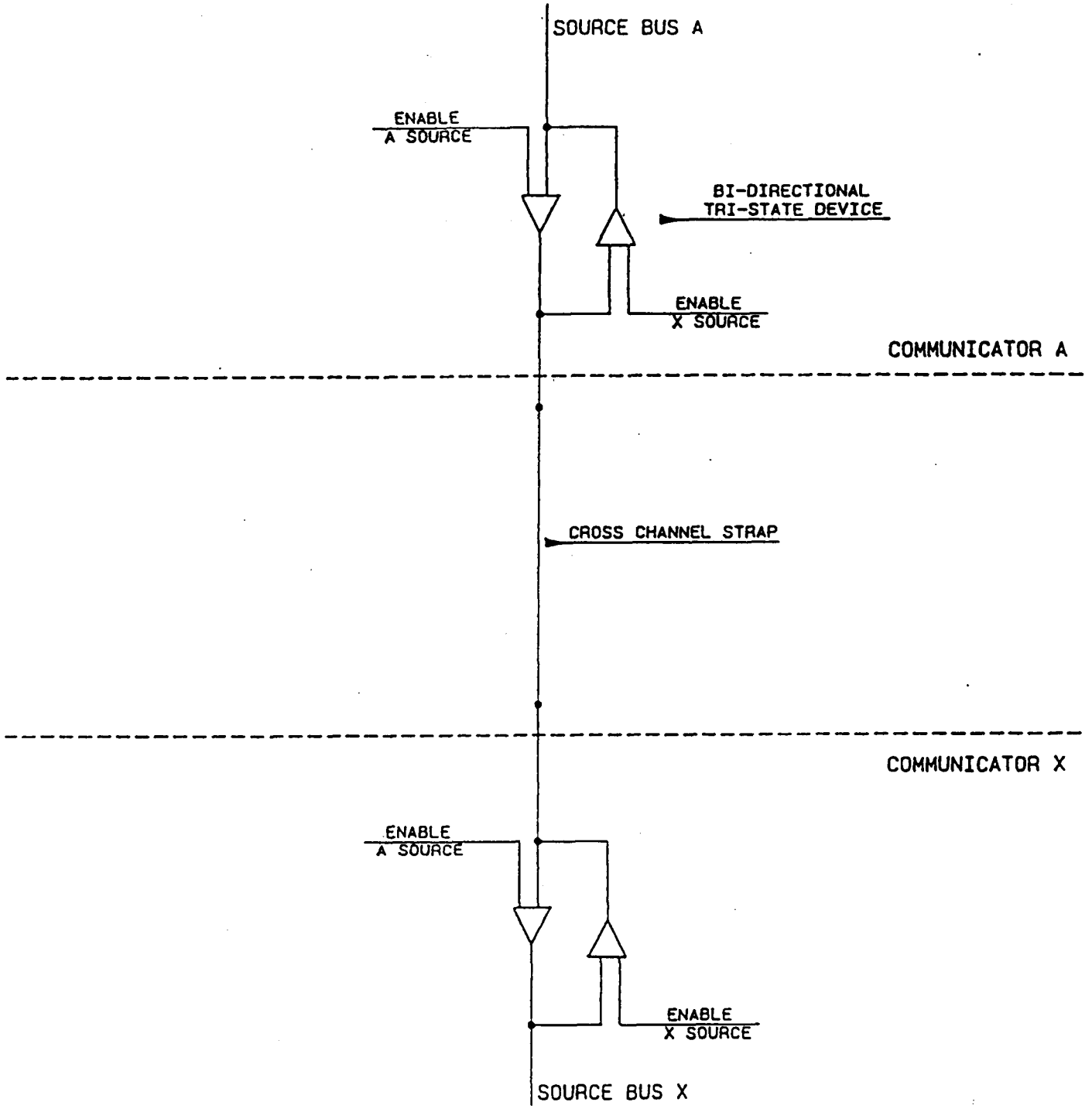


Figure 2.4. Actual Circuitry for Cross Channel Links

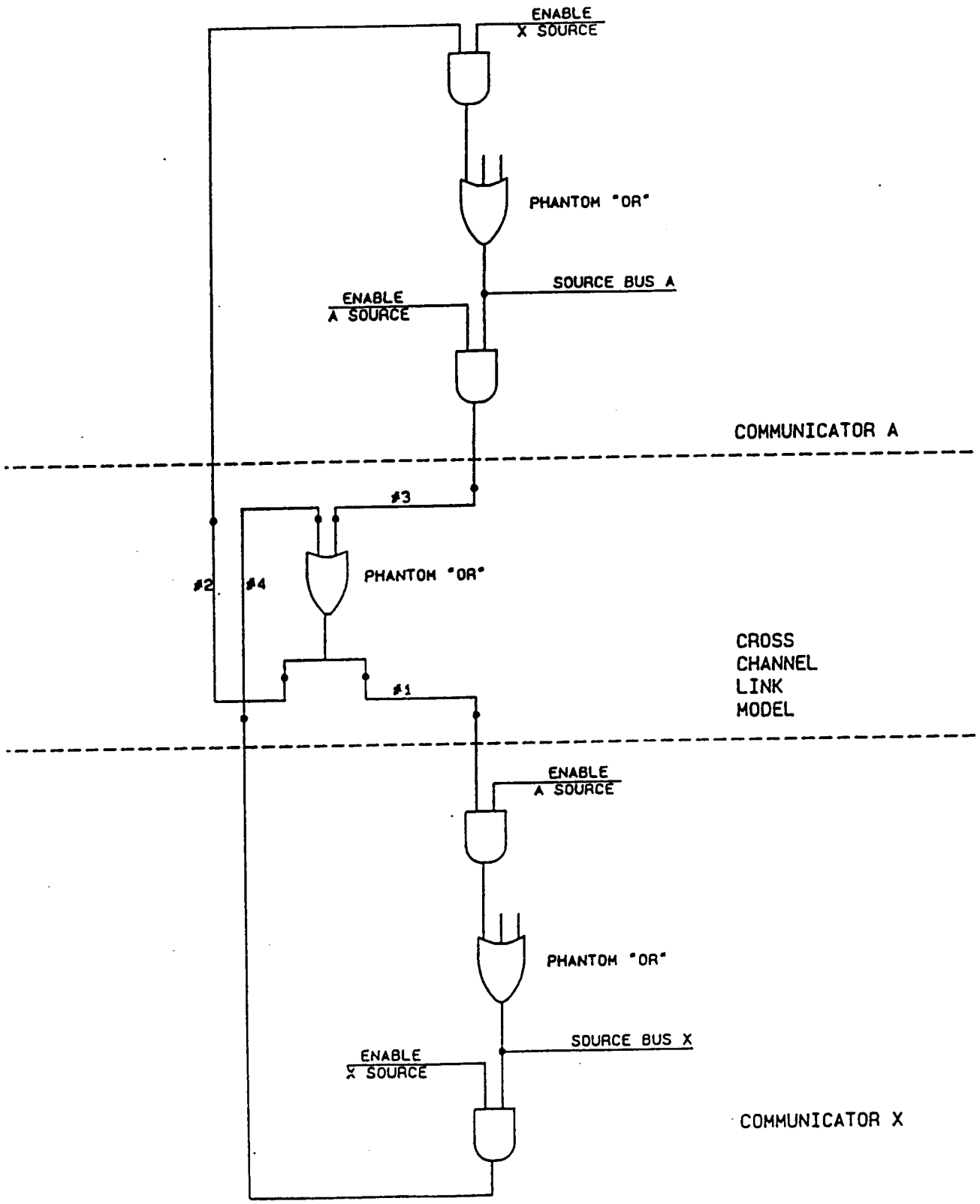


Figure 2.5. More Accurate Model Used for Cross Channel Links

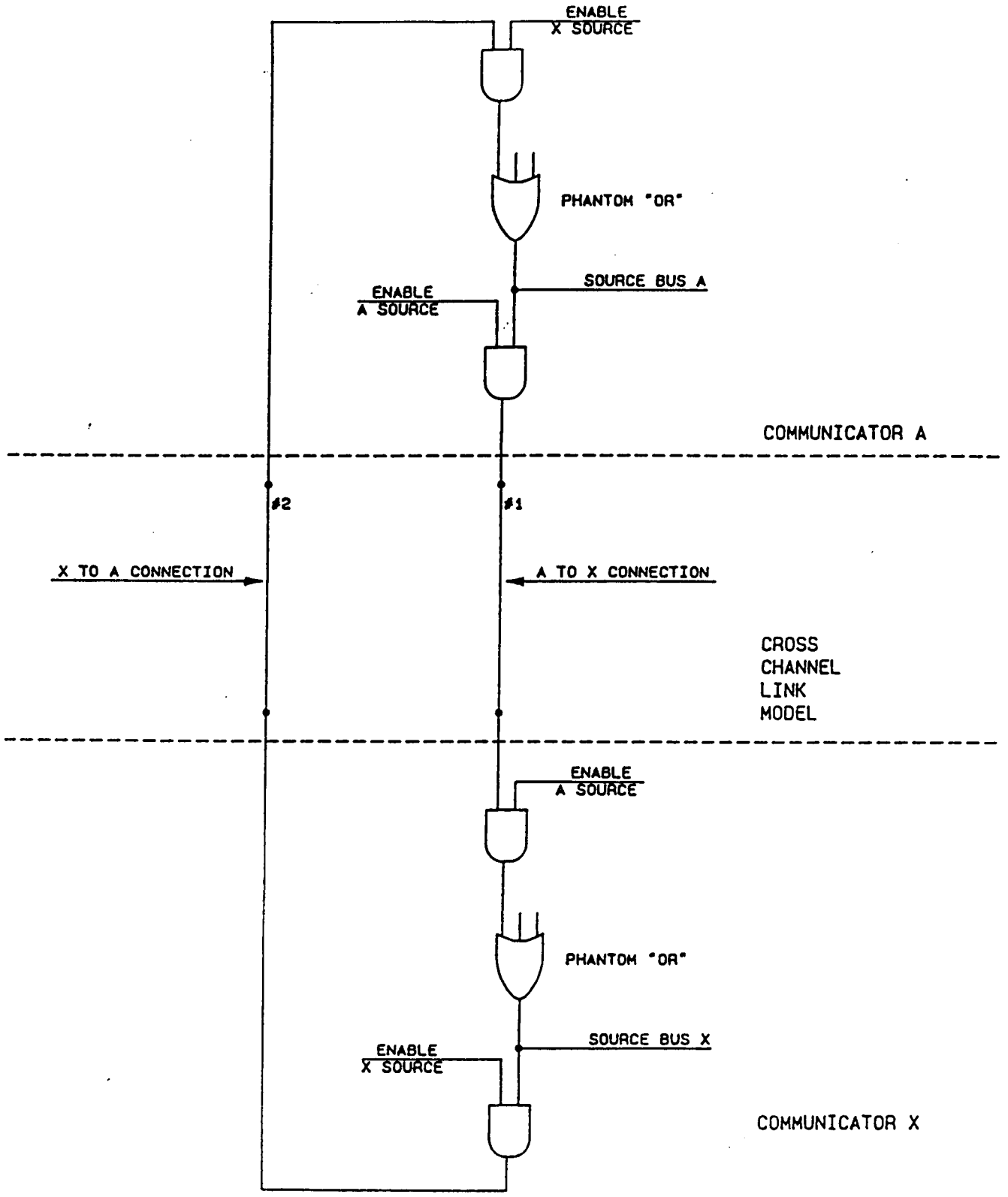


Figure 2.6. Simplified Model Used for Cross Channel Links

performs the bit-for-bit voting of non-masked data copies and detects and reports errors for non-consistent data copies. These voter related functions are implemented in four PLA's with two data bits (dibits) processed by each PLA. The Syndrome Or block consolidates errors (syndromes) reported by each Voter Syndrome dibit. The Current Status combines these consolidated errors with the contents of the Status Register.

The final major block is the Control. This block provides the timed control signals to direct the operation of the Data Communicator. It is implemented using three 2048×8-bit registered programmable read only memory devices. Since these devices are connected so that particular processor interface control signals, processor interface address bits, and certain of their own output bits are used to control their address inputs; a classical state machine with inputs, outputs, and internal states is implemented. This state machine is depicted in figure 2.9. The circuit is modeled using D flip-flops for the PROM register and the external memory feature of the Diagnostic Emulator. However, it was found that the contents of the PROM were highly redundant and that its size could be reduced to 512 words instead of 2048 words without losing any capability. Further reductions were possible but were not carried out. Figure 2.10 shows the state diagram for the the Write Xmit operation controlled by this state machine. Review of this design indicated that, as a result of the redundancy in the contents of the PROM, the state machine would require substantial additional testing to establish functionality. The redundancy likely resulted from the use of a high level design tool. This observation has implications regarding the use of high level design tools for fault tolerant systems.

Except for the Control block, four complete C/I's were modeled. Only two Control blocks were modeled. One Control block provided control signals for channels B, C, and D. The other provided control signals for channel A. The Channel A C/I was the only C/I module faulted in the experiment. In addition to the four C/I's, support logic which generated fault tolerant clock and processor clock had to be modeled. These signals were generated by a simple ring oscillator and a divide by counter. A ring counter was used to create a clock whose period in gate delays or event units is approximately equal to the actual processor clock frequency. The divide by counter used to generate the fault tolerant clock was set at 12 processor clock pulses instead of 10 as used in the FTP. This change reduced the required simulation time without losing essential information.

Every flip-flop in the C/I model has a pseudo-gate connected in line with its output so that the flip-flop output can be faulted. The addition of the pseudo-gate circumvented a Diagnostic Emulator constraint which does not allow flip-flop outputs to be directly faulted. The fault gate was automatically inserted into the netlist by the netlist translator software on each occurrence of a flip-flop. Pseudo-gates were inserted into the model for each external input to the C/I that went to a flip-flop. These inserted gates were used to satisfy a Diagnostic Emulator requirement that no external input be directly connected to a flip-flop. Finally, external output devices which had no internal connections were connected to *sink* gates. The *sink* gate

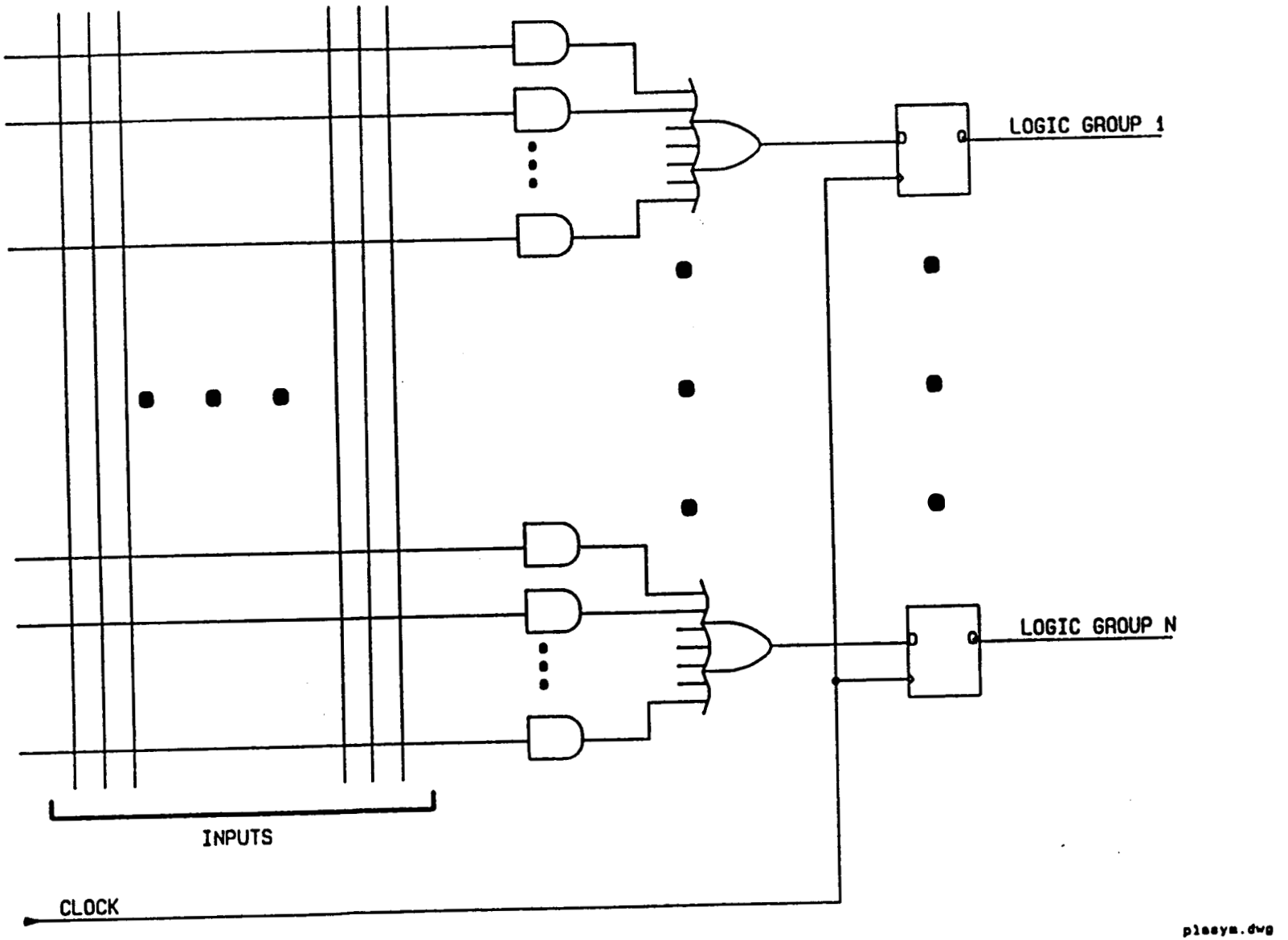


Figure 2.7. Programmable Logic Array

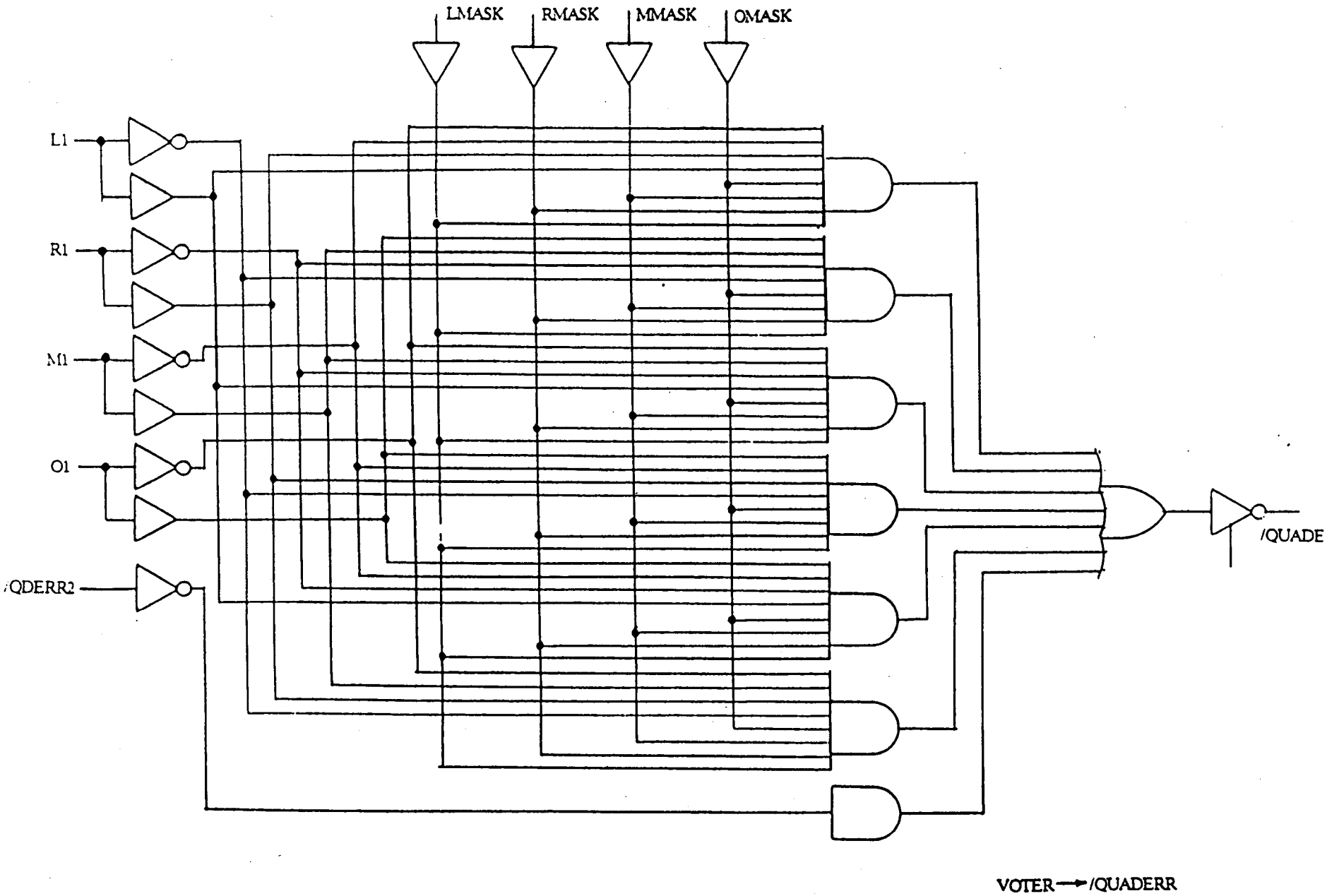


Figure 2.8. Voter Syndrome Quaderror Function

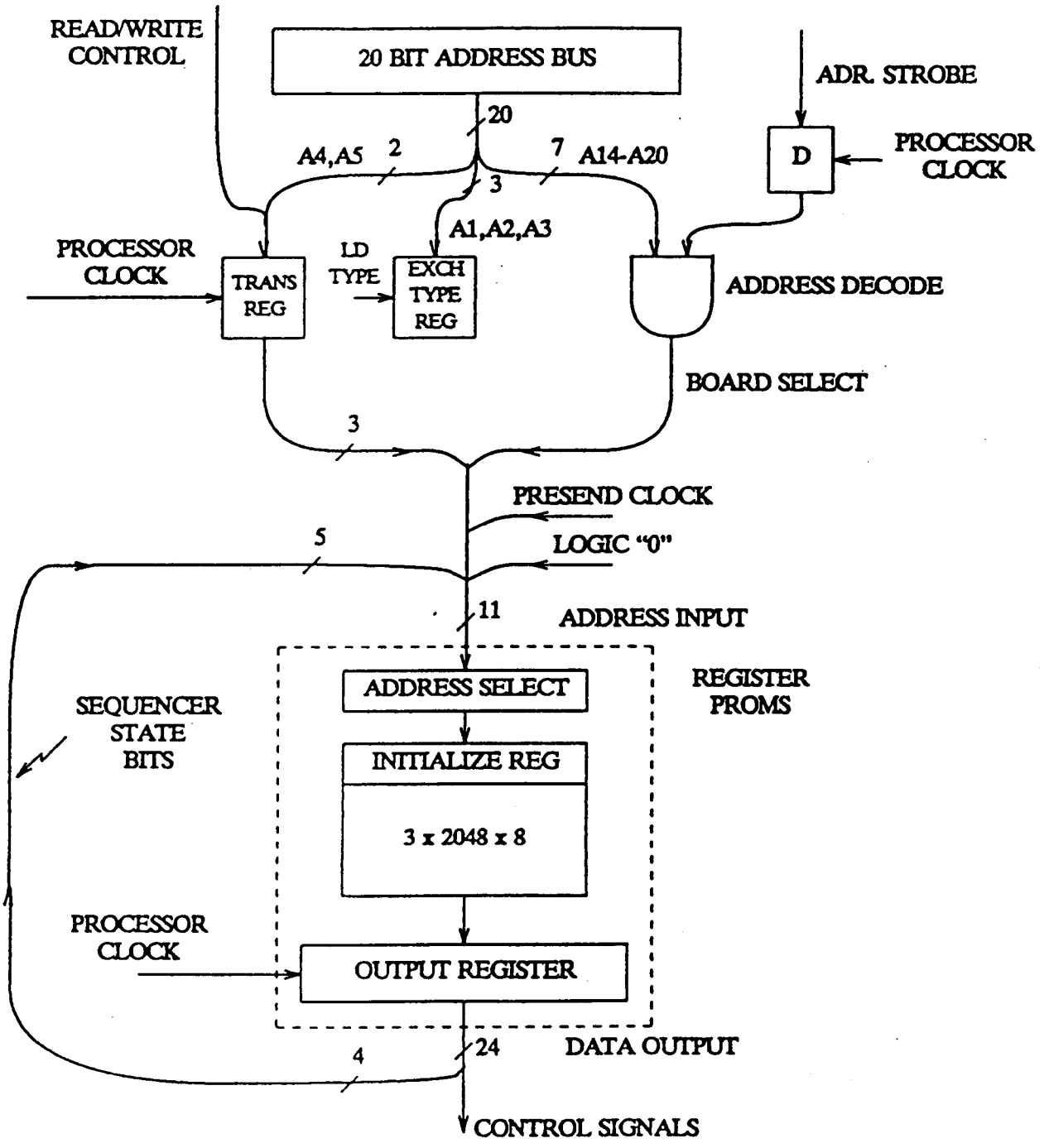


Figure 2.9. Communicator Control Sequencer

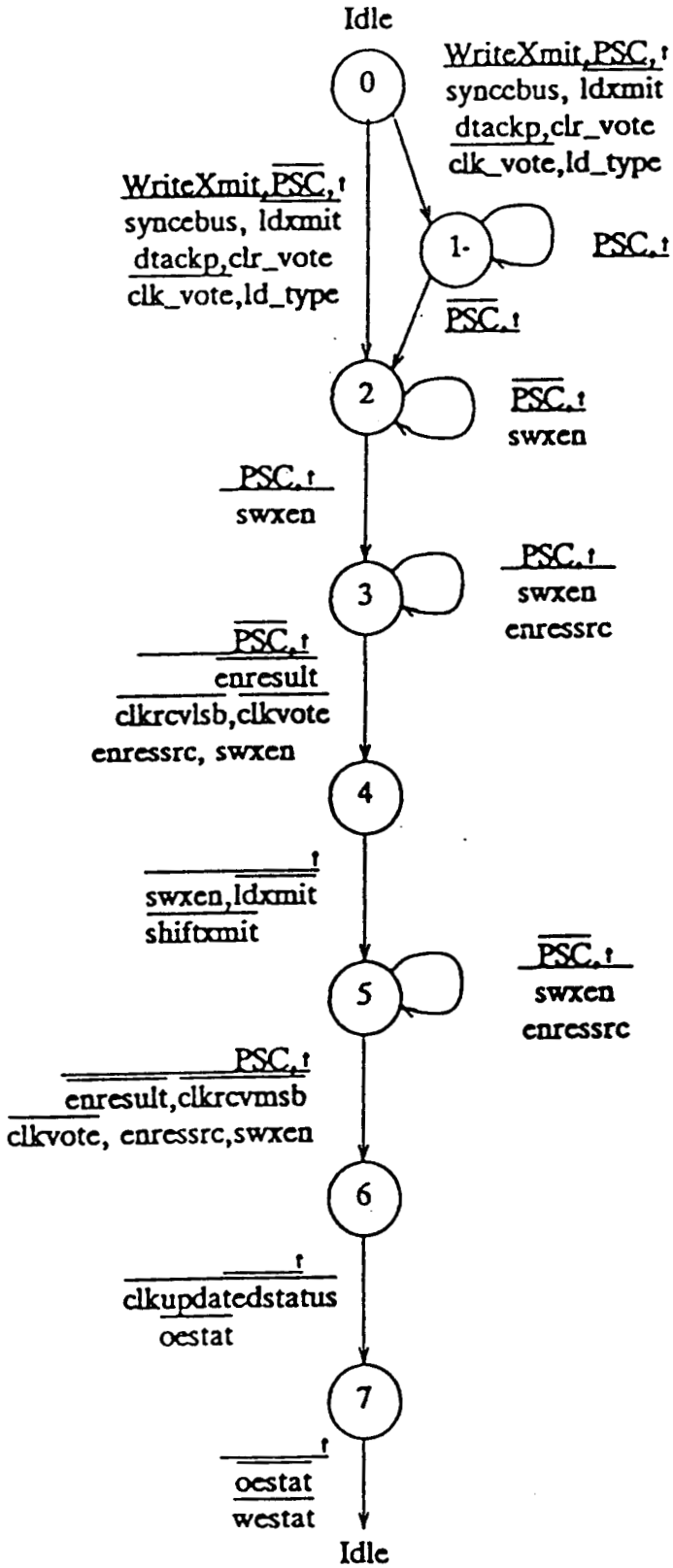


Figure 2.10. Write XMIT State Diagram

connection satisfied the restriction that all devices must have at least one internal connection. The output of each *sink* gate was in turn connected to one of its own inputs to meet this requirement.

3. Experiment Preparation

3.1. Introduction[4] [5]

Most of the effort expended in the experiment was directed toward developing and validating the Data Communicator/Interstage (C/I) gate level model and the associated experiment support software. This experiment preparation process can be divided into several general areas of activity. These areas are

1. the development and testing of experiment support software,

2. the activities associated with learning to use the diagnostic emulator,

3. the development of a gate level model of the C/I,

4. the development of C/I input sequences that would test the C/I in accordance with the C.S. Draper Laboratories C/I diagnostic software description, and

5. the validation of the C/I model.

A brief description of the process used to set up and use a Diagnostic Emulation model will be given in this paragraph. A complete description of the Diagnostic Emulation portion of this process is given in the user manual.[4] Figure 3.1 diagrams the process. A description of the C/I logic network was created by using FUTURENET, a logic schematic capture computer-aided design tool. The resulting network description file was translated to a format suitable for input to the NASA-LaRC Diagnostic Emulation network initializer. This file, labeled QM-1 network, combined with files describing initial values for network nodes and files establishing external connections to the network such as network input and memory data connections, was supplied to a network initializer program. The resulting initialized network file, combined with an emulation options file and external memory initialization files, was processed by an emulation initialization program. The resulting emulation file, along with option files, fault description files, and external input data files, was used as input to the emulation. If the FORTRAN Diagnostic Emulation program was to be used, these files were used directly. If the QM-1 emulation was to be used, these files were converted and then transferred to the NANODATA QM-1 machine for execution. The output of the emulation process was used to examine the behavior of the model during the debug and validation phase and to provide the inputs for post-processing during the experiment phase.

The process described above is shown in more detail in figures 3.2 and 3.3. Note that in figure 3.2 there is a loop through the network initializer, emulation

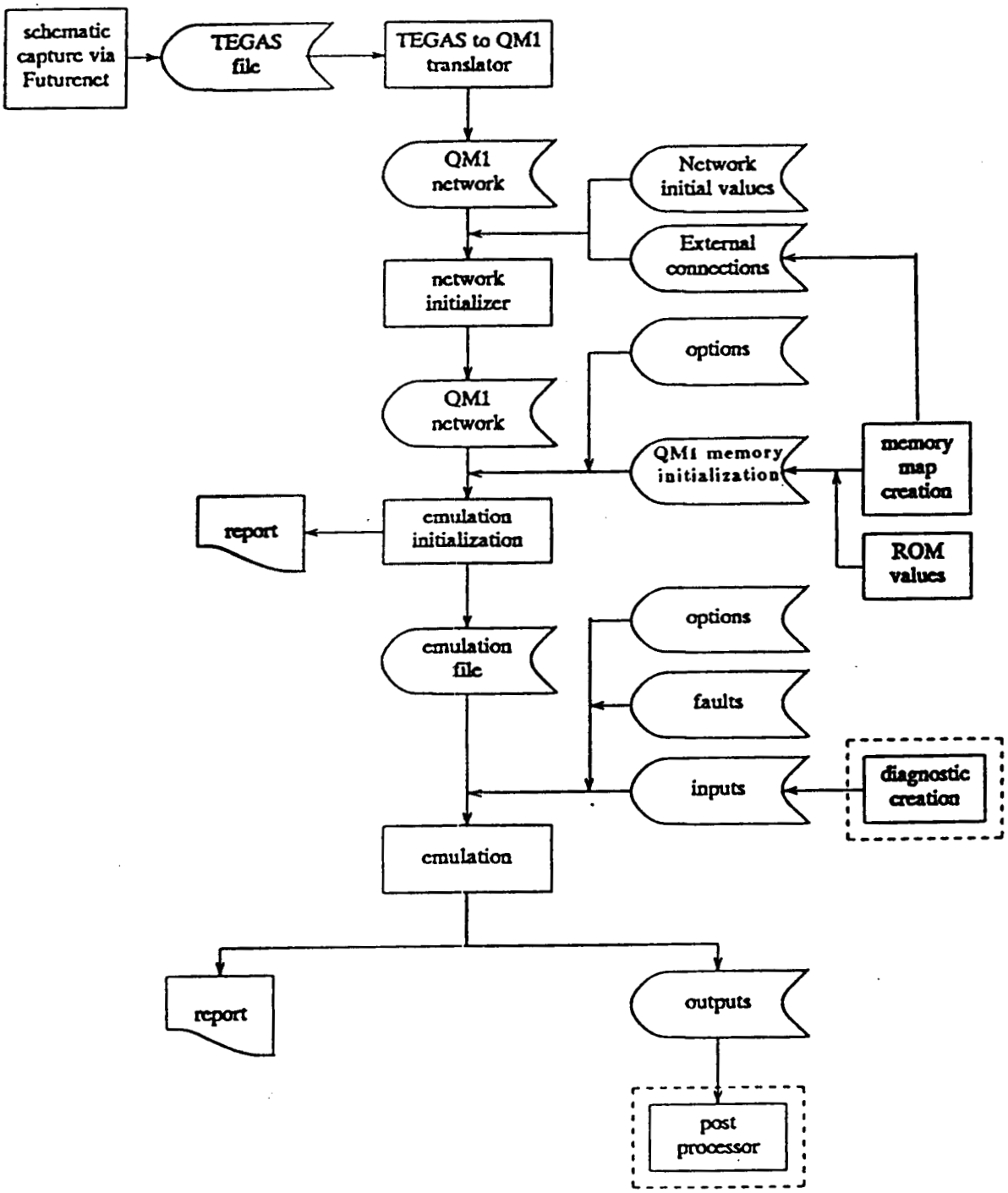


Figure 3.1. System Flow Diagram

initialization, memory map creation, external connections, and back to the network initializer. This loop is necessary to establish the address references required to define the external connections file. This part of the process is done once, provided the network does not change. Also note that in figure 3.3 the difference between a VAX FORTRAN emulation and a NANODATA QM-1 emulation is detailed.

3.2. Experiment Support Software

To use the existing Diagnostic Emulator to the extent required by the experiment, a number of new software utilities and tools had to be developed or procured. These items included the following:

1. a FUTURENET schematic capture computer-aided design tool;

2. a translator to convert FUTURENET netlists to a format suitable for the Diagnostic Emulation process;

3. an expanded set of primitive control features for the Diagnostic Emulator;

4. a set of programs to facilitate the preparation of Diagnostic Emulation options, connection, fault lists, input, memory, and initialization files;

5. a C/I mnemonic instruction translator to aid the preparation of C/I diagnostic test sequences; and

6. a program to analyze the outputs of the Diagnostic Emulation.

At the start of this project, gate level models of hardware to be emulated had to be created using a hardware description language. The large text file that resulted was then translated to a netlist suitable for Diagnostic Emulation. Creating, maintaining, changing, and verifying this large test file was a tedious, time-consuming, and error-prone process. The FUTURENET computer-aided schematic capture tools were purchased to facilitate the creation of the C/I emulation model or any other hardware models. It was felt that these tools would reduce the effort required for creating, maintaining, and verifying the C/I model. Since the schematic level is the natural level at which design engineers work with and document designs, it was felt that models created at this level would be less subject to errors.

A software program to convert FUTURENET/TEGAS netlists to the netlist format for the Diagnostic Emulator was written by RTI personnel. For the purposes of minimizing the resources dedicated to this conversion software, the devices used in the C/I schematic were restricted to those used by the Diagnostic Emulator. Adding the capability to accept more general devices, which in turn would be translated to Diagnostic Emulator devices, is not precluded by this restriction.

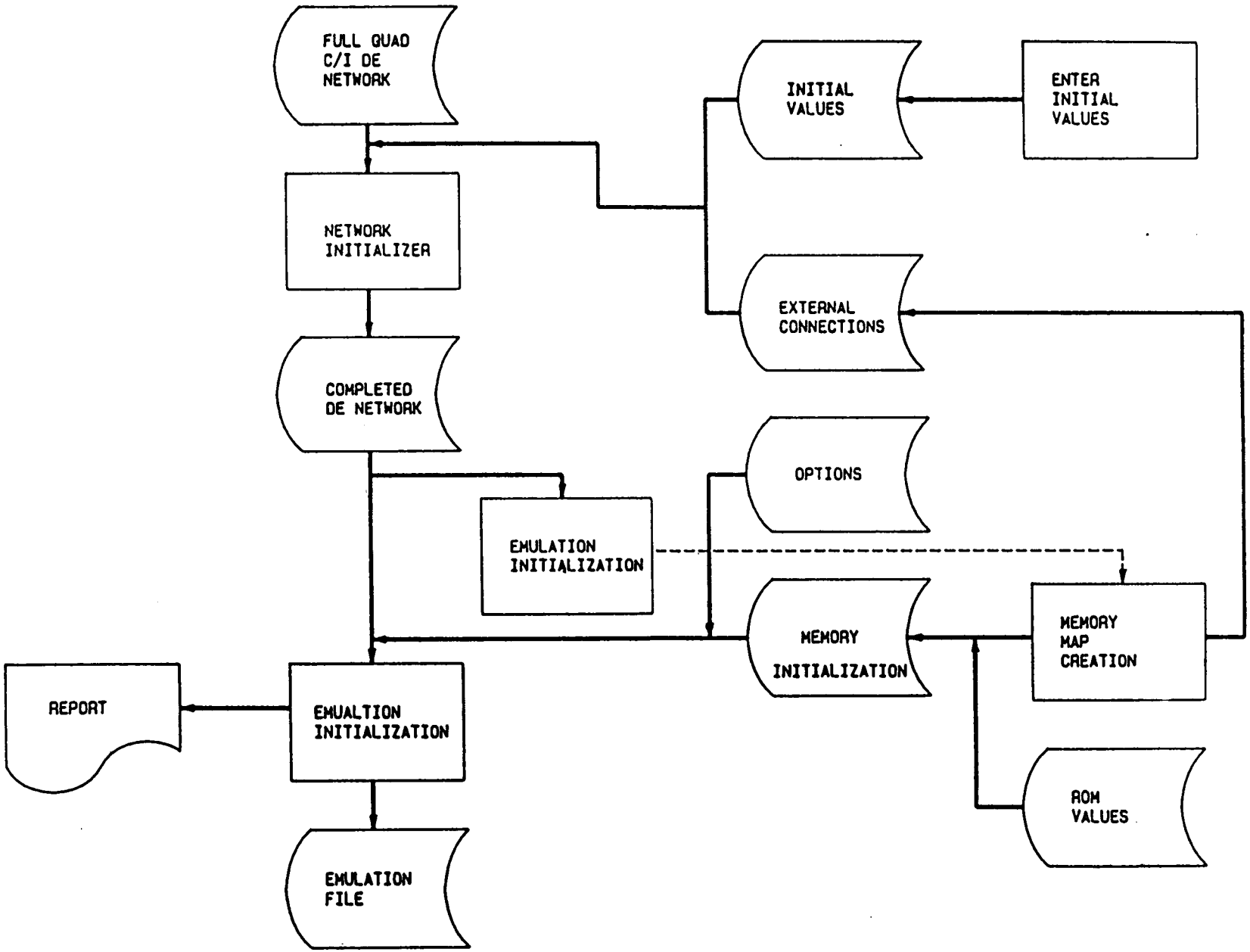


Figure 3.2. Network Initialization

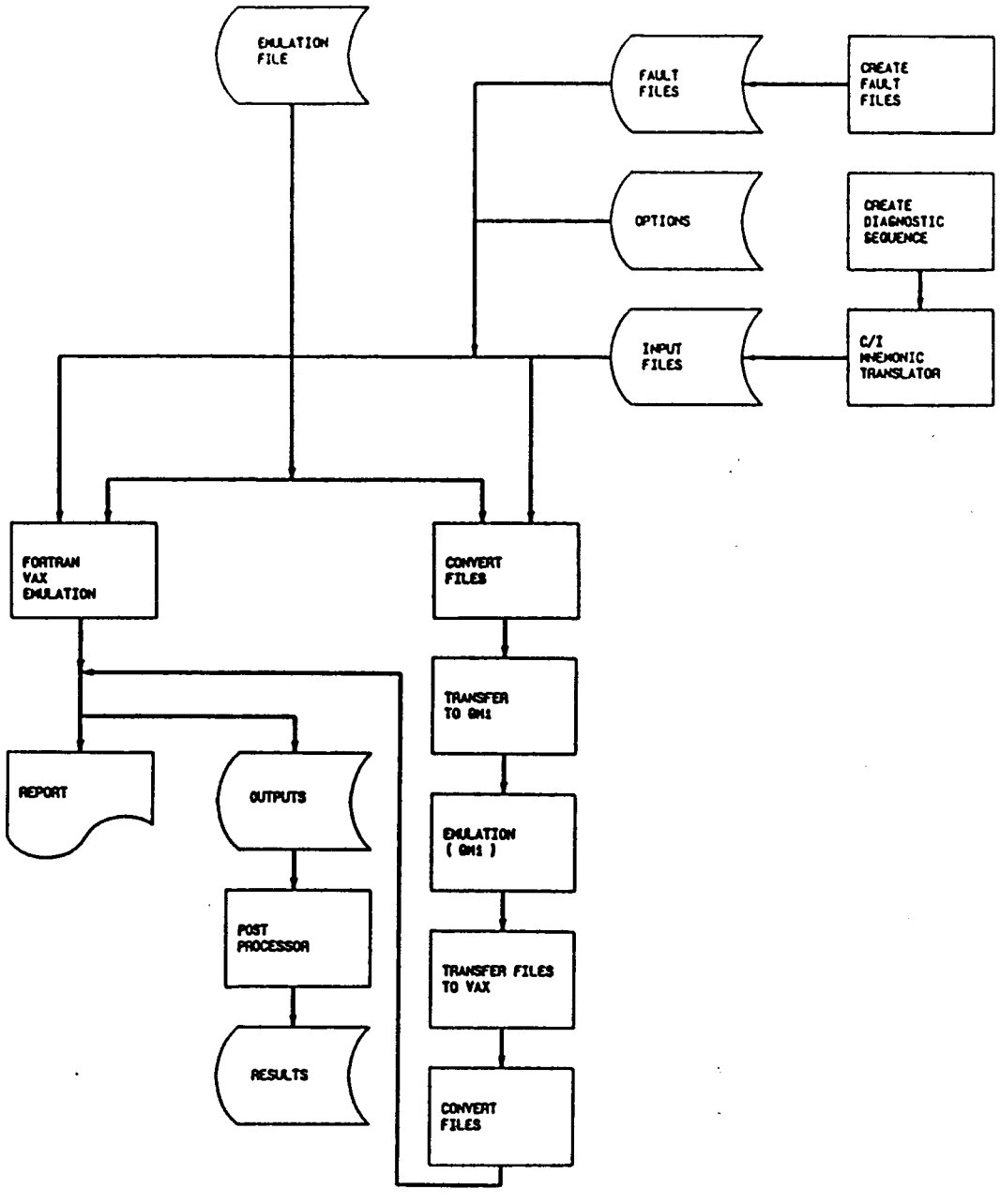


Figure 3.3. Emulation Process

To meet the needs of a fault insertion experiment, new control primitives were required for the Diagnostic Emulator. The new primitives added were as follows:

1. the capability to assert a stuck-at logic one or zero fault on any device output for a defined number of emulation steps,

2. the capability to provide external logic inputs from an input data file at designated emulation event numbers,

3. the capability to direct the output of designated logic nodes to an output file,

4. the capability to assert a fault on a bit in external memory arrays, and

5. the capability to define the length of an emulation run and to recycle the emulation using the next fault defined in a fault list.

The capability to provide each of these primitive control features had been planned but had not been implemented at the beginning of this effort. These capabilities were added to both the VAX FORTRAN emulation and the NANODATA QM-1 emulation. The implementation of these software features was done by NASA personnel.

A C/I mnemonic instruction translator was written to simplify the preparation of test sequences for the experiment. This translator provided the conversion of mnemonic representations for each C/I instruction into the proper bit pattern. In addition, *pseudo* instructions that allowed for defining iterative loops were provided. This latter capability simplified the creation of long test sequences consisting of iterative data values . Table 3.1 shows the mnemonics used.

NAME	DEFINITION
VECB	Pseudo op to define start of a new test vector
VECE	Pseudo op to define end of a test vector
LOOP	Pseudo op to set up a repetitive sequence of C/I operations. Arguments specify the level of nesting and the iteration increments and limits
END	Pseudo op to define the end of a loop
RDRR	Read receive register
WRCV	Write receive register; arguments are data for each C/I
WXMR	Write transmit register; arguments indicate source for each C/I (A, B, C, D) and data for each
WMSR	Write mask register; arguments indicate mask to be written for each C/I

Table 3.1. C/I Instruction Mnemonics

Emulation outputs from the experiment were processed to determine the faults detected by each test vector.

3.3. Learning Use of the Diagnostic Emulator and Testing of Software Items

A simple logic network was used as a vehicle to learn the use of the Diagnostic Emulator and to test the various new software items. In addition to simplicity, the network was designed to be similar to the C/I and to require the use of all Diagnostic Emulator primitives in the C/I model. These primitives include fault insertion, external inputs, external outputs, external memory arrays, and each type of logic device. Successful setup and emulation of this network was an essential first step in learning the use of the Diagnostic Emulator and for validating the new software development items.

A block diagram of this network is given in figure 3.4. The network was first analyzed on a simple IBM PC-based simulator, EZCAD. The EZCAD results were used as a basis for comparing Diagnostic Emulator results.

3.4. Development of a C/I Gate Level Model

Some of the circuit details of the C/I gate level model are discussed in section 2. The desired process for creating the model is shown in figure 3.5. In this process, a schematic of the entire quad C/I network would be captured using FUTURENET.

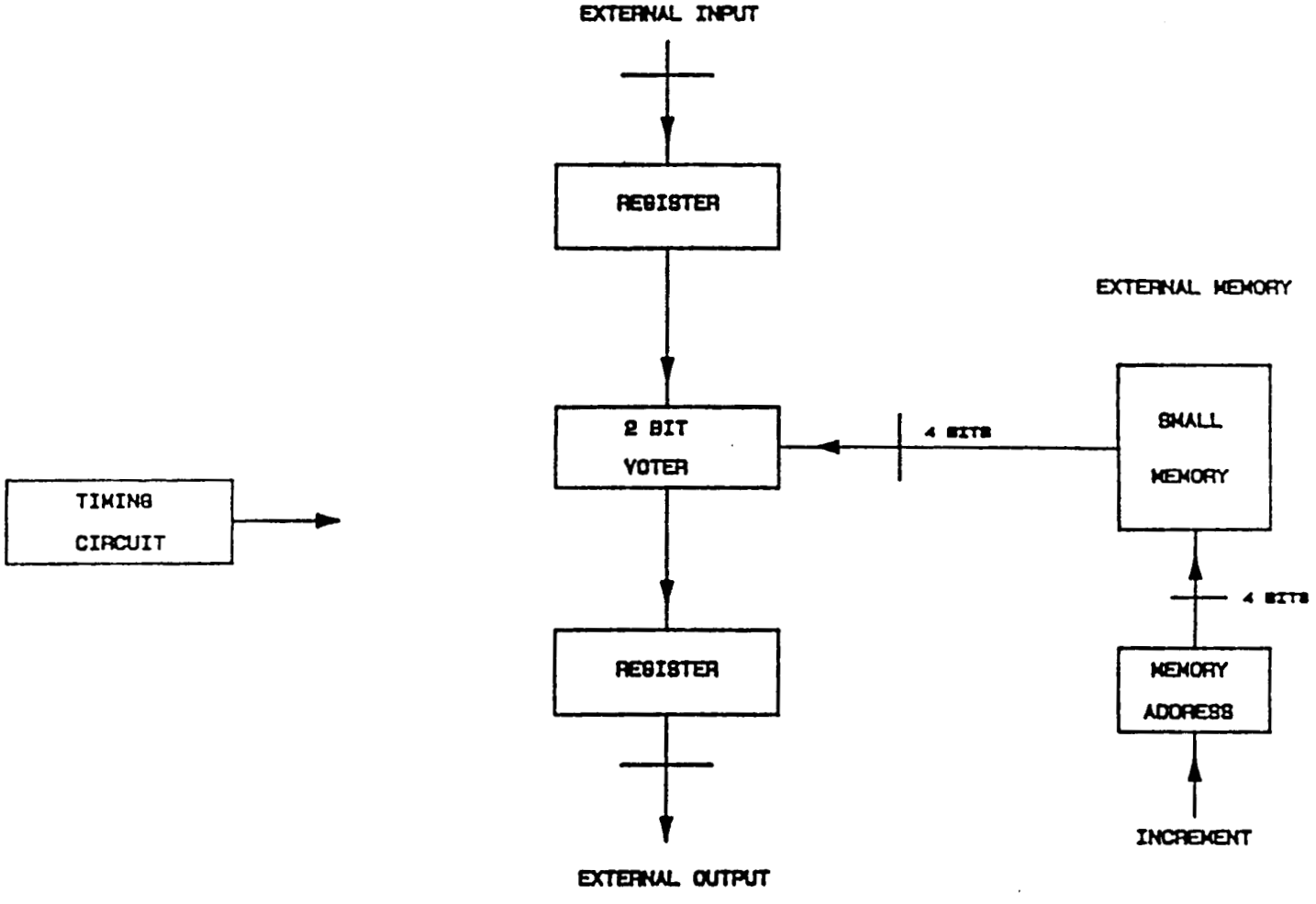


Figure 3.4. Block Diagram of Test Circuit

The resulting netlist would then be converted to a form suitable for the Diagnostic Emulator. Changes to the model would take place at the schematic entry level and would be followed by the translation process.

An attempt was made to carry out this process. However, the full quad C/I network was too large for certain of the FUTURENET support tools. Consequently, another less desirable process had to be employed. The actual process used is shown in figure 3.6. In this process, the full quad C/I was broken into smaller subnets. Each subnet was translated and these subnets were concatenated into a larger net. The interconnections between subnets were handled by editing these interconnections into the translated net file. While the number of interconnections was small relative to the total network, the editing process to produce these interconnections was prone to error. Further, this step made it difficult to return to the schematic level for model checking and for any model changes. This limitation of FUTURENET and the manner in which the model was created made it difficult to realize the full potential of computer-aided design capture. Unanticipated model network configuration control problems had to be solved, and checking of certain parts of the model was rendered much more difficult since it had to be done at the netlist level.

An indicator of the potential for difficulties is found in the network size information. The full quad C/I network consisted of more than 5000 nodes (devices) and 15,000 connections. About 20,000 lines of text, or more than 300 pages, were required to list the network.

This network is relatively small compared to most VLSI designs. The only effective way to deal with such designs is with effective CAD tools.

3.5. Diagnostic Test Sequences for the Data Communicator/Interstage[6] [7] [8] [9]

The C/I plays a critical role in FTP fault tolerance by ensuring that all good processors receive identical data values (source congruency), by masking errors, and by providing for FTP reconfiguration after faults are detected. An important part of FTP operation is the use of diagnostic test sequences for the C/I for both pre-mission and in-mission testing. These test sequences are used to uncover latent faults. This experiment was directed toward determining the effectiveness of proposed diagnostic test sequences.

A fault detection, isolation, and reconfiguration (FDIR) program is used in FTP. FDIR provides for hardware implemented fault detection and error reporting, software implemented fault isolation based on the analysis of error reports, software implemented reconfiguration by disabling failed channels, and software implemented self tests to uncover latent faults. The C/I is tested with self tests that operate within the FDIR function. FDIR tests the C/I with a frequently executed high priority limited scope test, called Presence, and a somewhat exhaustive low priority test sequence, which is executed with a much lower frequency.

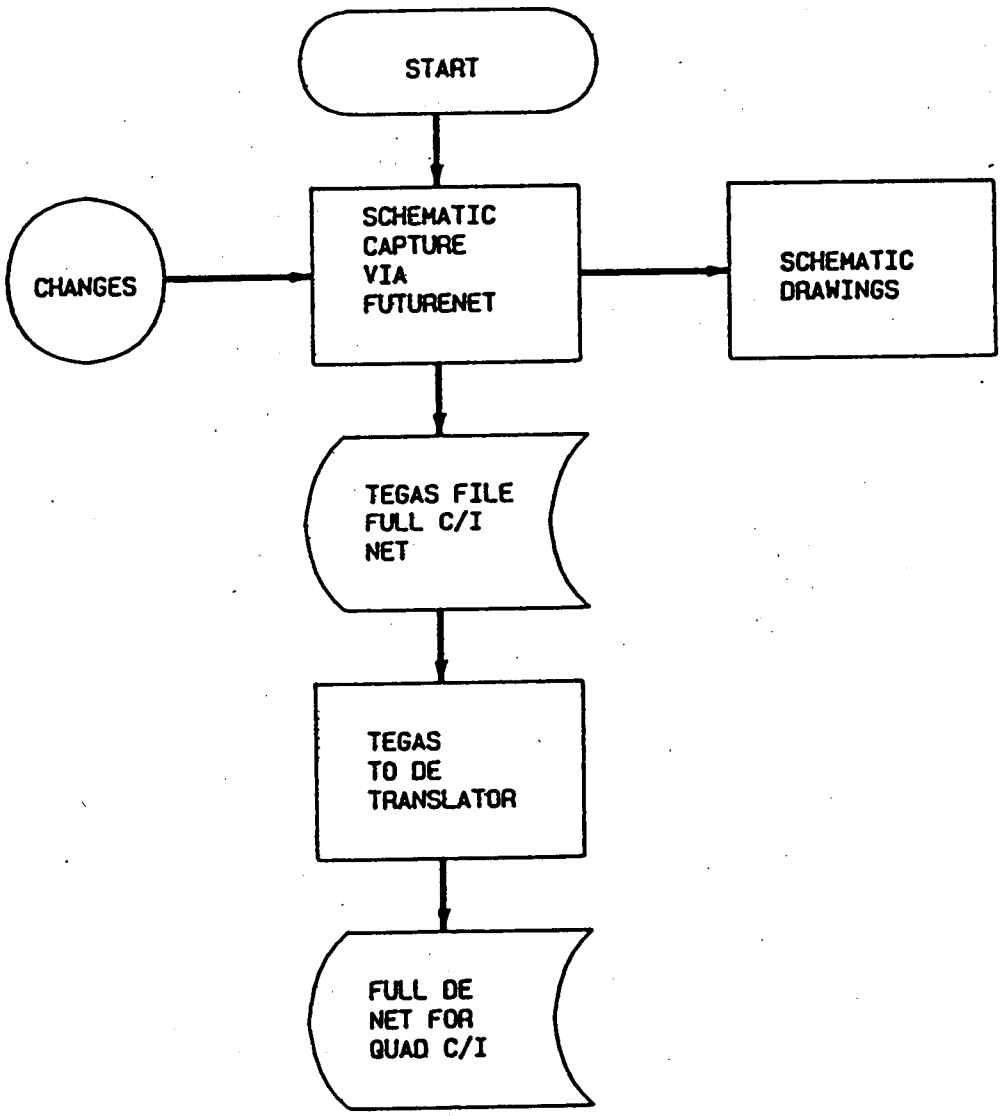


Figure 3.5. Network Model Preparation
(Desired)

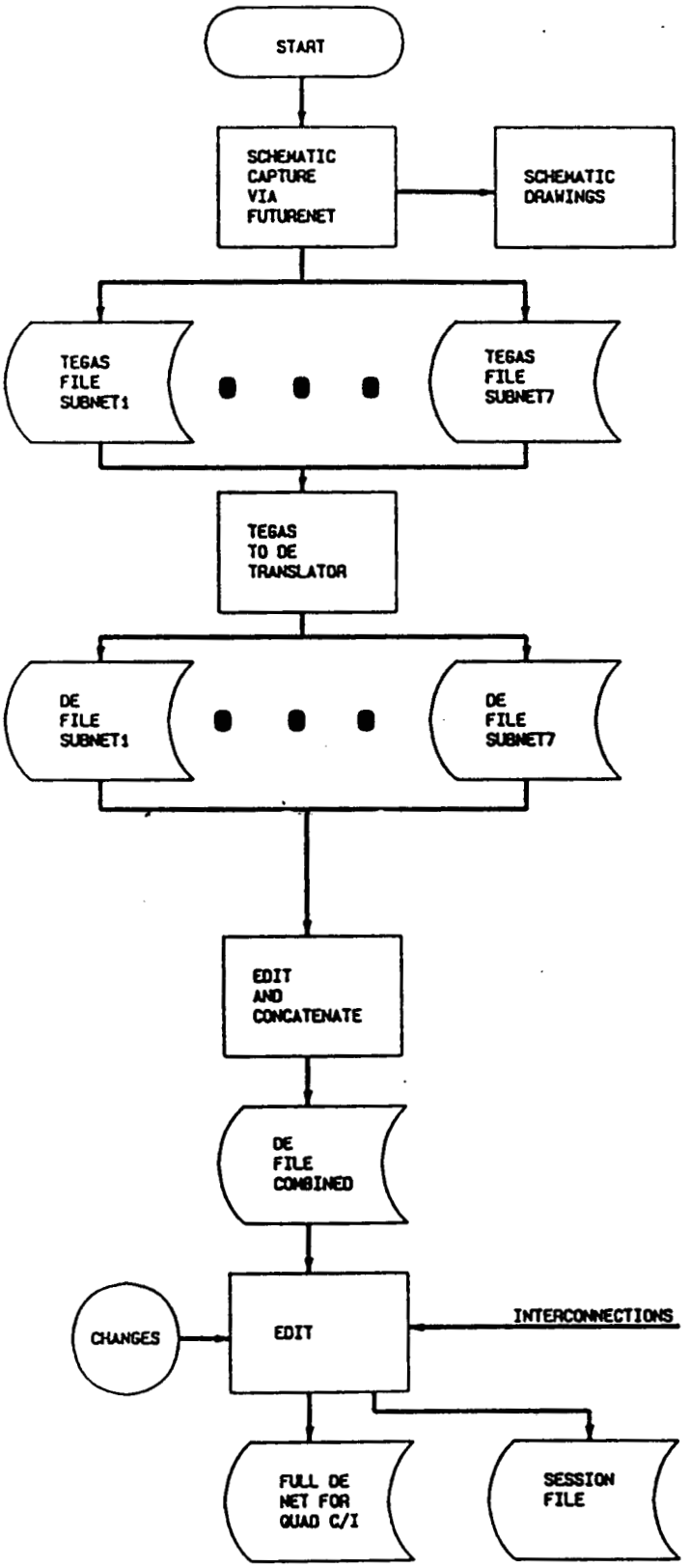


Figure 3.6. Network Model Preparation (Actual)

The Presence test is a simple test that is executed at the beginning of a processing frame to determine the *presence* of each processing channel of the FTP. During this test, unique data values from each channel are exchanged via the C/I. Correct exchange of a channel's unique values establishes presence.

The low priority C/I self tests of the FDIR assume that a certain class of C/I faults, such as hard errors in the data path, will be detected in the normal course of processing and/or by the Presence test. Those areas of the C/I logic not stimulated in the normal course of operation are the focus of the self tests. These areas include error detection, error reporting, fault masking, and reconfiguration logic.

The main components of the low priority C/I self tests are as follows:

1. the Voter PAL Test,

2. the Mask Transform Test,

3. the Voter Syndrome Or Test, and

4. the Current Status Update Test.

The C/I voter is implemented using programmable logic array chips. The voting and error detection logic is contained in the voter PAL chips. Two bits of a data byte from each FTP channel are voted within a single PAL package. Four voter PAL packages are required for each communicator. The inputs to these PAL chips include dibits from each channel and a mask bit for each channel, as shown in figure 3.7. The Voter PAL Test consists of sequencing through all combinations of inputs to each voter PAL chip for both the least and most significant byte of a data exchange. This exhaustive test is used because certain failure modes of a PAL can result in any input to the chip coupling to any PAL output. This is true even if the gates associated with a given output are not programmed to accept all input signals. The total number of test vectors required for this test are 2^{16}.

The Mask Transform Test targets the channel mask or blocking logic (see figure 3.8). This logic provides the capability to exclude a channel from the voting process. This component of the self test requires 192 test vectors.

The Voter Syndrome Or Test targets the two PAL chips associated with combining the error outputs of the 4 voter PAL chips. The test is intended to test only those gates not previously tested by the Voter PAL Test. This test was not completely defined at the time of the fault injection experiment.

The Current Status Update Test is intended to uncover faults in the current status update PAL chips (see figure 3.9). In this test, all combinations of data exchange errors are stimulated by selection of appropriate test vectors. Stimulated errors are logged into the Status Registers. These test vectors are followed by test vectors that stimulate additional data exchange errors. The Status Register is then checked to determine if it is properly updated.

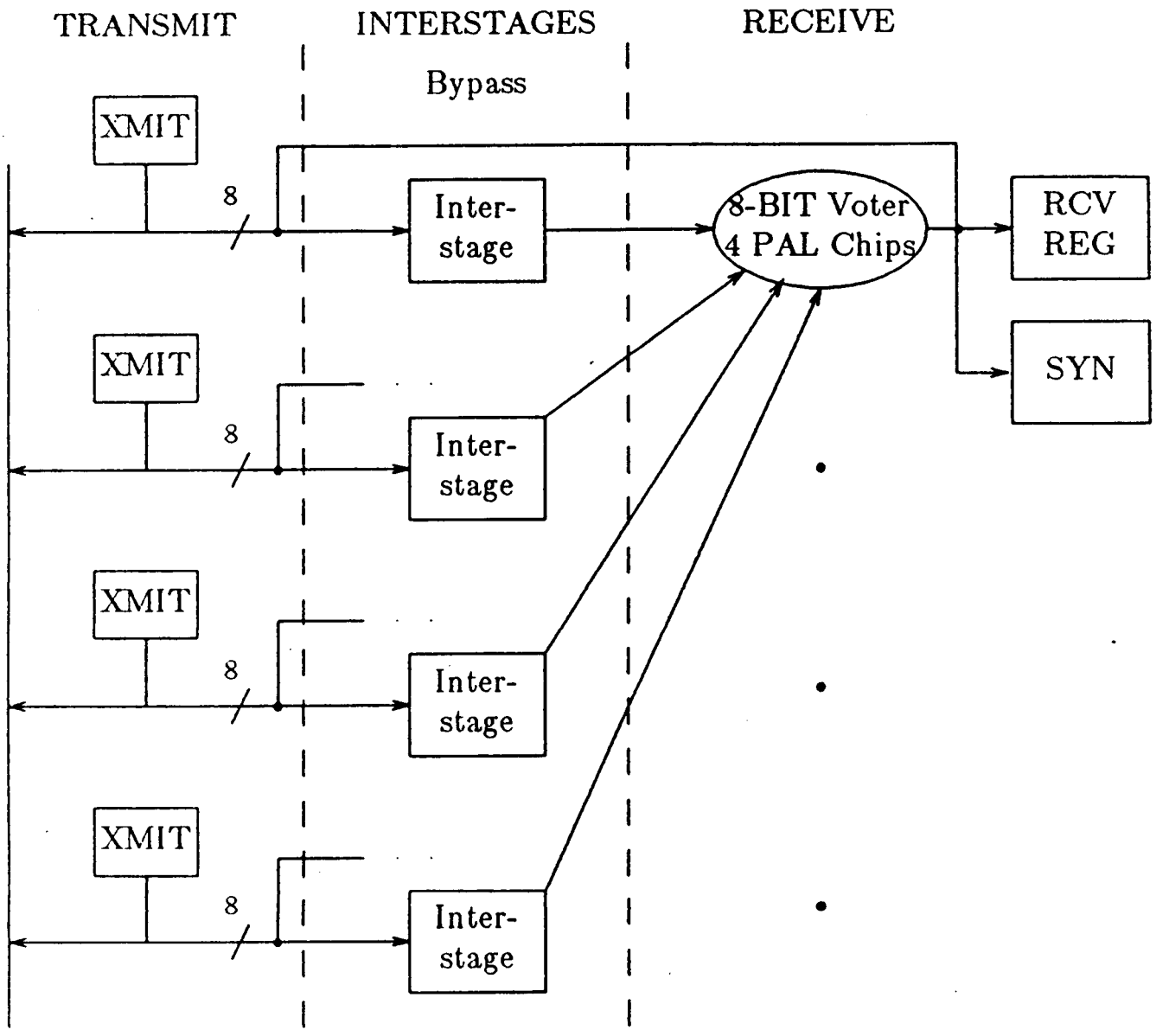
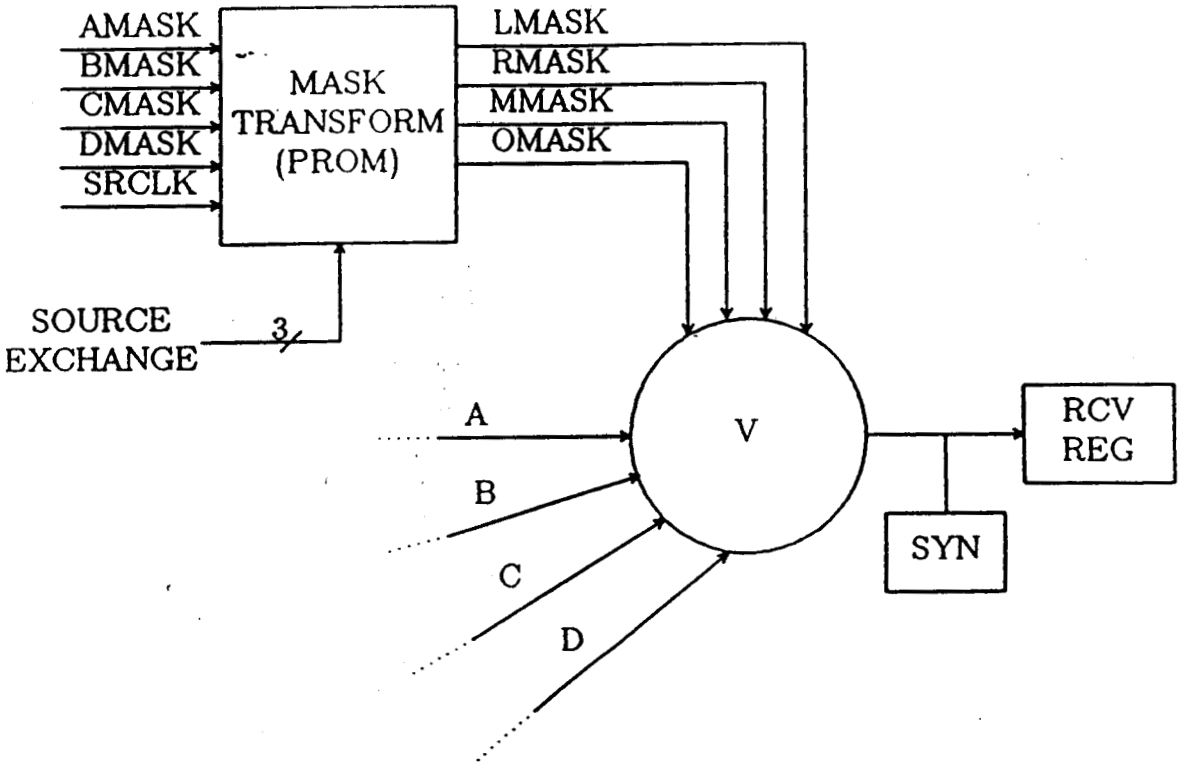


Figure 3.7. Communicator/Interstage Exchange Network



Tests For Each PROM Input:

	Test Number					
Channel	1	2	3	4	5	6
A	A	D	A	B	A	C
B	B	C	A	B	B	D
C	B	C	C	D	A	C
D	A	D	C	D	B	D

(Number of Mask Transfer PROM Test Vectors) $= 6 \cdot 32 =$ **192**

Figure 3.8. Mask Transform/Source Locking Test

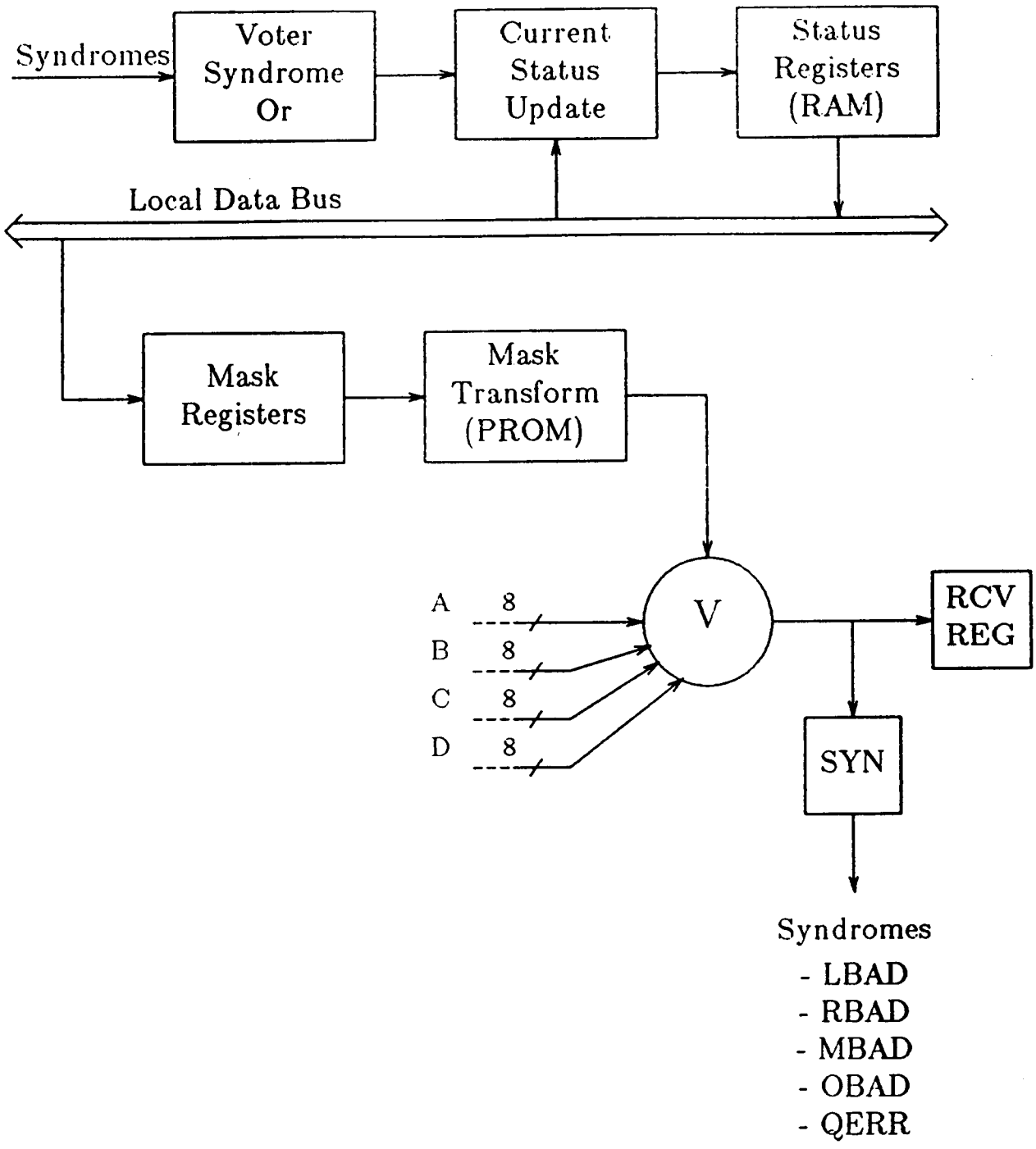


Figure 3.9. Error Logging/Masking Network
(One Network for each Processor)

Figure 3.10 illustrates a typical sequence of C/I transactions associated with a single test vector. Three transactions are required to exchange data, read the results, and retrieve the associated error report. This does not include any special setup of the Mask Registers and Status Registers. Once exchange results and error reports are obtained, it is necessary for all channels to exchange this data. This allows all good channels to arrive at a consensus on the results of a test vector. A total of nine exchange and read transactions are required for each vector. Each exchange/read transaction pair requires about 10 μsec to complete in real time.

Assuming that the C/I self test requires approximately 2^{16} test vectors, and that the self test is constrained to use only five percent of the C/I throughput capacity, the C/I self test will require approximately two minutes to execute. This time sets a bound on the fault latency time for faults detectable via the C/I self tests.

Estimates of the time required to conduct a fault injection experiment using the Diagnostic Emulator are of interest. Consider an experiment that requires the injection of 2000 faults and requires that the C/I self test sequence (about 2^{16} test vectors) be executed completely for each fault. Assume that the NANODATA QM-1 is used. Also assume that only the self test transactions must be executed, i.e., the C/I throughput is 100% dedicated to self test for the experiment. The QM-1 version of the Diagnostic Emulator will execute approximately 50,000:1 slower than real time for the C/I network. The total simulation time required would be approximately 19 years. If the VAX FORTRAN version was used at 10^6:1 slower than real time, the simulation would require approximately 3.8 centuries!

The unacceptably long run times cited in the preceding paragraph provide motivation for restricting the extent of an experiment. If the simulation time requirements are reduced by a factor of about 1000, 200 QM-1 hours would be required. Such a simulation time requirement would be well within the feasible range.

In the planning stage of this experiment, the self test for the C/I used substantially fewer test vectors. The number of test vectors used by C.S. Draper Laboratories (CSDL) at that time would have resulted in reasonable simulation time requirements. The exhaustive test implemented by the new self tests greatly expanded the simulation time.

A reduction in simulation time by a factor of 9 can be obtained by recognizing that the exchanges required to share opinions in an actual FTP would not be required in an experiment. An additional factor of about 100 was needed and was obtained by reducing the number of test vectors.

Recall that the exhaustive testing addressed certain PAL failure modes. It was determined that these failure modes would not be reflected in a direct gate level model of the PAL logic. Consequently, the large set of test vectors could be reduced and not change the outcome of an experiment based on a gate level model. Further, it was decided to test the C/I voters only for the quadriplex and triplex configurations and only for one of the voter PAL chips.

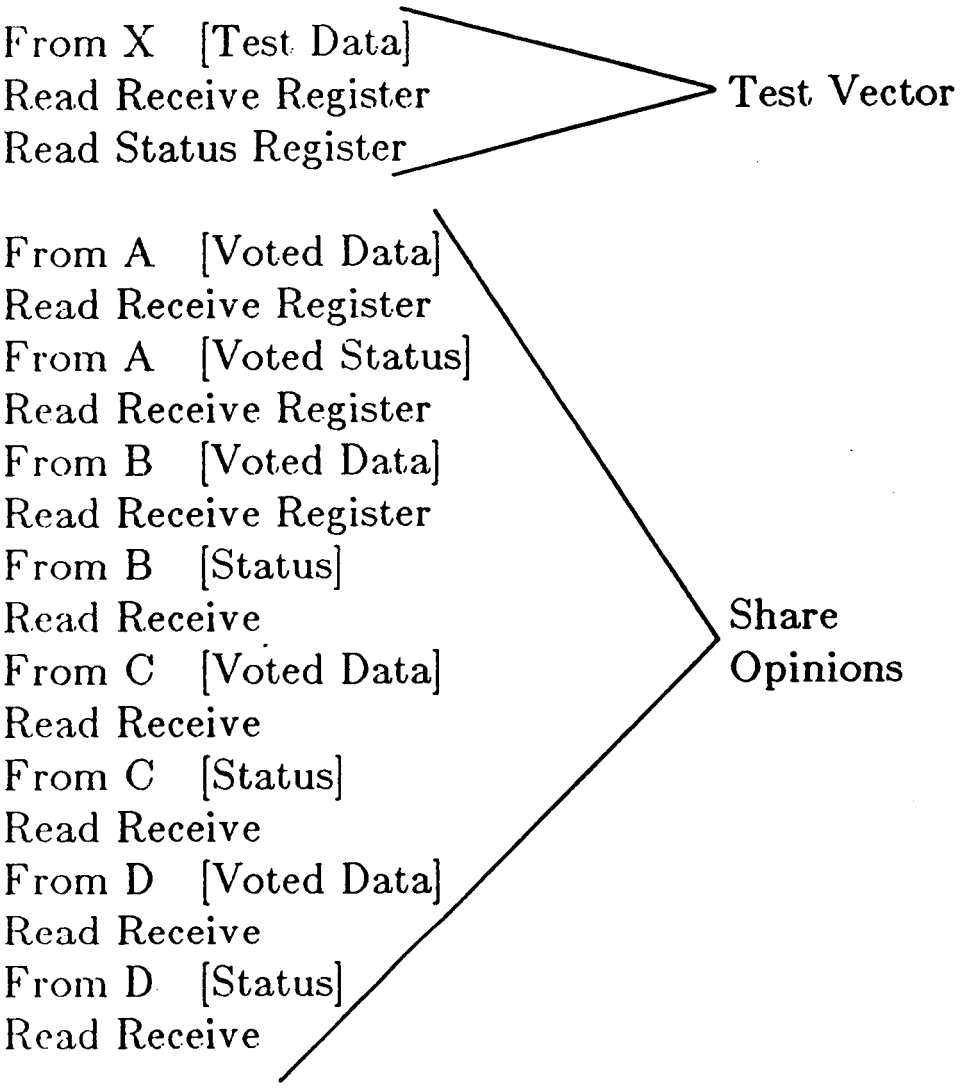


Figure 3.10. Typical C/I Self Test Sequence for a Single Test Vector

The tests prepared for the experiment are summarized in figures 3.11 and 3.12. Tests are characterized by the number of test vectors within a test, by the number of C/I transactions or instructions required for the test, and by the number of DE simulation steps required for the test. These tests include about 600 test vectors that are in the desired range. Appendix A gives detailed listings of these tests.

3.6. Validation of the C/I Emulation Model

The purpose of the validation process was to establish, to the greatest extent possible, that the C/I emulation model accurately reflected the behavior of the C/I. Most of the funding for this emulation experiment was directed toward this effort. In general, the validation of a complex model would be difficult. For this experiment, several factors further complicated the validation process.

Ideally, the emulation model for a network would be created automatically from a design data base using well-established software programs, and the design verification data base of test vectors and results would be available for use. In this experiment, design information used to create the emulation model was in the form of a schematic drawing and a design description. Furthermore, access to the C/I designer was limited. To create the emulation model, the schematic was manually entered into a computer using a CAD schematic capture tool. The devices used in the design were replaced with Diagnostic Emulation gate primitives prior to schematic capture. The translation of the netlist for the captured schematic to a form suitable for Diagnostic Emulation was accomplished using a software program developed for this experiment. As described in section 3.5, certain of the network model interconnections had to be established by editing the translated netlist text files. Additional experiment software, described in section 3.3, was developed to support the experiment. This software included modifications to the Diagnostic Emulator.

When discrepancies were found during testing/debugging of the model, errors in the design capture, errors in the new software, errors in the changes to the Diagnostic Emulator, errors in the use of the Diagnostic Emulator, and errors related to modeling issues were all candidates for the source of the problem. Engineering analysis based on limited design documentation was required to devise tests for the model, to determine the correctness of the responses, and to devise additional tests for isolating the source of detected errors.

The additional complications of the validation process for the C/I can be attributed to the following:

1. the status of the Diagnostic Emulation facilities in AIRLAB (i.e., changes, limitations, and documentation) at the start of the experiment,

2. the design capture process that was employed,

3. the development of software to facilitate model development, and

- Voter Tests
  Tested a single voter PAL chip (2 bits) in the quad and triplex configurations. Used both upper and lower bytes to give two test vectors per vote.

- Mask Transform Tests
  Tested mask transform *logic* for quad and duplex configurations

- Current Status Update Test
  Tested the current status update PAL

- Presence Test
  Equivalent to CSDL "Presence"

- Basic Test
  RTI test which tested a range of C/I functions beyond "Presence"

Figure 3.11. Tests Used for Experiment

TEST NAME	ABBREV	NBR VECTORS	NBR INSTRS	SIM STEPS
Presence	presence	10	62	8400
Current Status Update	cstupd	10	64	11900
Basic	basic	16	112	11900
Mask Transform Quad	msktrnq	12	84	9900
Mask Transform XAXB	msktrnxaxb	24	160	19200
Mask Transform XCXD	msktrnxcxd	24	160	19200
Voter Test Quad1	vb0lquad1	128	772	98000
Voter Test Quad2	vb01quad2	128	772	98000
Voter Test B01 3XA	vb013xa	64	388	48700
Voter Test B01 3XB	vb013xb	64	388	48700
Voter Test B01 3XC	vb013xc	64	388	48700
Voter Test B01 3XD	vb013xd	64	388	48700

Figure 3.12. Diagnostic Sequences

4. the absence of a design verification data base of test vectors and responses.

The effort to validate a model could have been reduced considerably by the appropriate use of well-established CAD tools that were integrated into the hardware design data base. The uncertainty as to the possible sources of model discrepancies encountered in this first AIRLAB Diagnostic Emulation experiment would be reduced for additional experiments. Integration of the design data base into the Diagnostic Emulation facilities would eliminate problems associated with design recapture. Finally, integration of the design data base would permit the use of the design verification test vectors and responses and would reduce the need for the validator/experimenter to fully understand the operation of the network modeled and to analyze the detailed responses of the network.

The validation process for the C/I model relied to some extent on the software testing afforded by the emulation of the simple network described in section 3.3. The approach to verifying the C/I model was to verify the C/I microsequencer logic and to then add additional C/I functions. The C/I microsequencer model was verified using a simple IBM PC-based simulator, EZCAD. This simulator provided easy-to-use graphical *logic analyzer*-type outputs. The microsequencer was verified against its documented state transition diagram for each operation type. A microsequencer model was prepared for the VAX FORTRAN Diagnostic Emulator and was verified for each state transition and output. This provided a source of correct control signals for subsequent testing of the C/I model.

The full C/I model that incorporated the verified microsequencer model was prepared for the VAX FORTRAN Diagnostic Emulator. This model was tested for each operation type with various data inputs. Numerous design recapture errors were found and corrected. Next, sequences of operations with various data values were emulated. Errors were found and corrected. Finally, all the diagnostic test sequences, except for the non-voter tests, were used as the emulator inputs. The emulator output for each test sequence was verified for correctness.

The QM-1 version of the Diagnostic Emulation was not available when the model verification was complete. Consequently, the experiment was started using the VAX FORTRAN version. Once the QM-1 version became available, a QM-1 C/I model was prepared. This model was verified against extensive emulation results from the VAX FORTRAN version for both good circuit and faulted circuit conditions. Non-faulted C/I outputs for each diagnostic test sequence are included in appendix B.

Errors stemming from each of the possible sources enumerated above were found during the validation process. However, most errors, including the most difficult errors to isolate, were traced to the manual editing of interconnections into the large netlist text file.

As noted earlier, the validation part of this experiment required the most effort. Without appropriate CAD tools integrated into the design data base, it would be difficult to justify the cost of experiments using other designs of similar complexity. This

experiment involved a network of relatively modest complexity (5000 gates) relative to modern VLSI (50,000 to 100,000 gates). Standards for verifying the accuracy of network models are much higher for highly reliable fault tolerant systems. The only cost-effective way to manage the complexity of modern VLSI designs for a highly fault tolerant system will be with the use of well-established validated CAD tools that are integrated into a common design data base.

4. Diagnostic Emulation Experiment Description and Results

4.1. General Plan

The general plan for this experiment is to do the following:

1. emulate at the gate level the Charles Stark Draper Laboratories (CSDL) Fault Tolerant Processor's Communicator/Interstage (C/I) hardware using the AIRLAB Diagnostic Emulator facilities,

2. insert single faults into the gate level model,

3. for each inserted fault, use the CSDL diagnostic self test sequence for the C/I as inputs for the emulation model,

4. determine if observable errors are produced in the emulation model output for each fault inserted, and

5. evaluate the results of the experiment.

Figure 4.1 diagrams the emulation setup. Test vectors from the C/I diagnostic self test sequence were used as input to a gate level model of the C/I network. Faults that model *stuck-at-one* and *stuck-at-zero* gate output failures were inserted into the gate level model. Each fault was inserted for a complete pass through the diagnostic self test sequence, only one fault was present in the network during any pass through the self test sequence, and faults were only inserted into the channel A portion of the C/I. For each fault and test vector, outputs from the C/I model and both data and error reports were compared against the outputs produced by a fault-free C/I model. If the outputs differed, the fault was designated as detectable for the test vector in question.

Figure 4.2 diagrams the output space for this experiment. Originally, 600 test vectors were planned (see section 3.5), but only 328 were used. These test vectors were grouped into tests associated with a particular purpose or functional area within the C/I. These tests are listed in figure 4.3 along with their associated parameters. Faults were grouped into sets associated with a particular functional area. Figure 4.4 lists the fault groupings, and figure 4.5 diagrams the functional areas associated with these fault sets. Note that within Voter and Voter Syndrome fault sets, devices associated with 2 of the 8 bits were faulted. Excluding these faults reduced the experiment

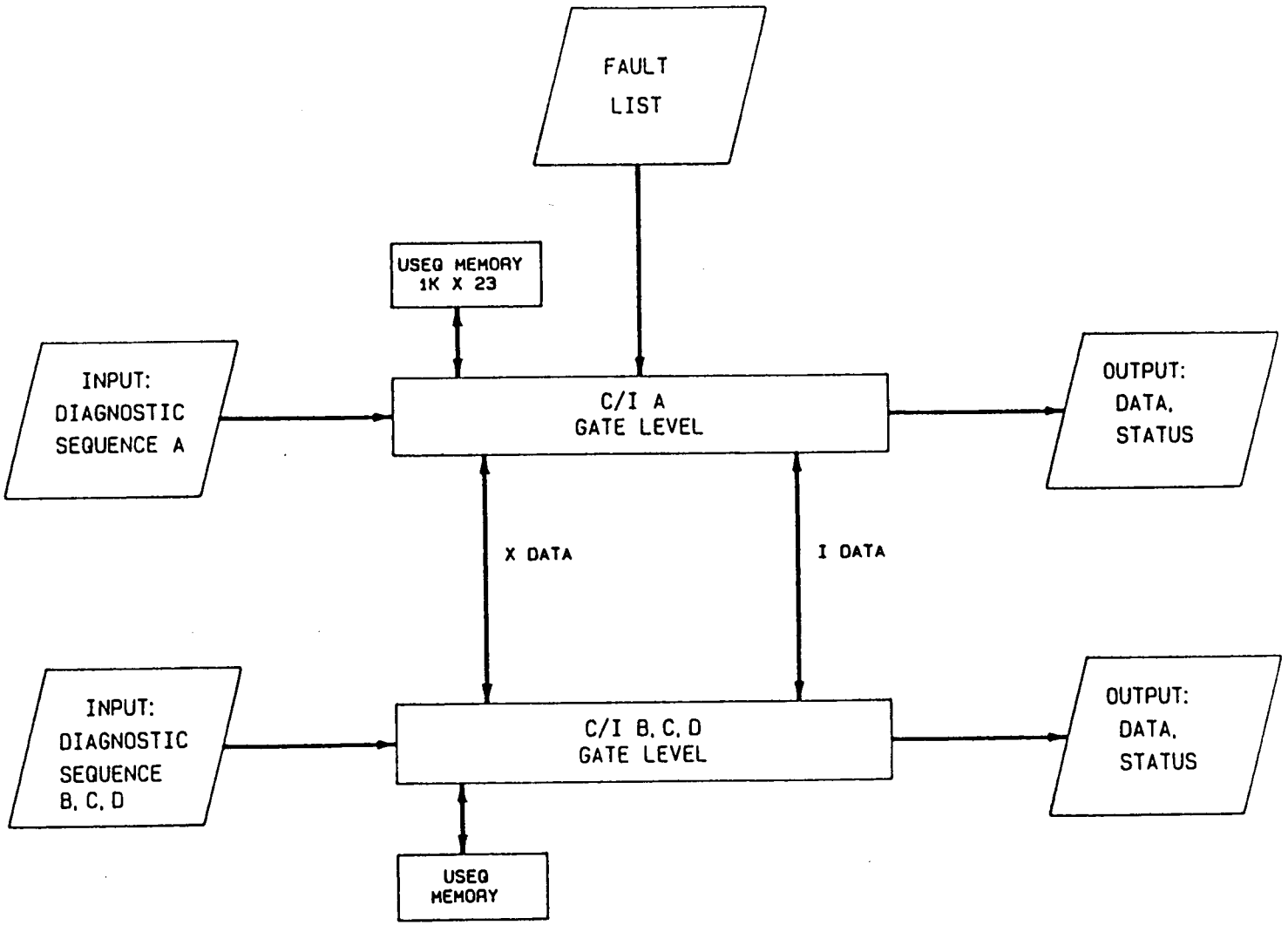


Figure 4.1. Communicator/Interstage Model

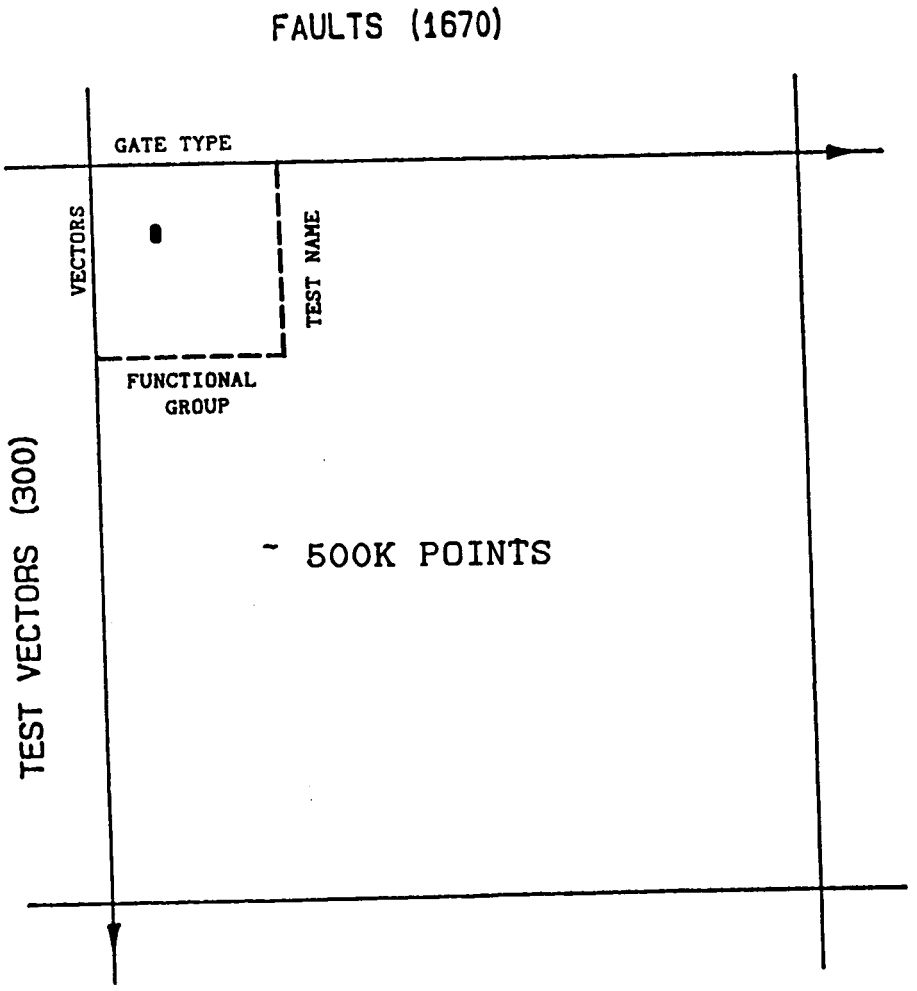


Figure 4.2. Simulation Result Space

TEST NAME	ABBREV	NBR VECTORS	NBR INSTRS	SIM STEPS
Presence	presence	10	62	8400
Current Status Update	cstupd	10	64	11900
Basic	basic	16	112	11900
Mask Transform QUAD	msktrnq	12	84	9900
Mask Transform XAXB	msktrnxaxb	24	160	19200
Mask Transform XCXD	msktrnxcxd	24	160	19200
Voter Test Quad1	vb0lquad1	128	772	98000
Voter Test Quad2	vb01quad2	128	772	98000
Voter Test B01 3XA	vb013xa	64	388	48700
Voter Test B01 3XB	vb013xb	64	388	48700
Voter Test B01 3XC	vb013xc	64	388	48700
Voter Test B01 3XD	vb013xd	64	388	48700

Figure 4.3. Diagnostic Sequences

duration. Since the network topology for the remaining 6 bits was the same as for the 2 bits where faults were injected, the results for the 2 bits could be used to determine the results for the remaining 6 bits. This grouping of faults and test vectors was found to be a useful way to interpret experiment results and to draw conclusions regarding the effectiveness of particular test vectors. Figure 4.6 shows a typical output from the emulation. The single number on a line is the simulation time step and the four numbers on a line are the outputs from the four channels of the C/I model. Figure 4.7 is a typical post-processing output. For each vector in a self test sequence and for each fault in a fault set, the faulty circuit emulation output values are compared to good circuit emulation output values. If the outputs differ, the fault is considered to be detected by that test vector. The "1" entry in figure 4.7 indicates that the fault was detected. The score (the number of faults found) for each vector is given at the bottom of the fault list, and the number of vectors in the designated test detecting a particular fault is given in the right hand column. The first run and additional runs within the fault list are for fault-free conditions. Emulator outputs from the first run were used as the reference for subsequent faulted outputs. Emulation runs required large amounts of computer execution time and complex setup procedures. The opportunity for errors was of concern. Consequently, additional fault-free runs were included to help ensure consistent results. This precaution did, in fact, result in finding several problems during the experiment.

4.2. The Experiment

Originally, the NANODATA QM-1 version of the Diagnostic Emulator was to be used for this experiment and the VAX FORTRAN version was to be used for validating the model. Since the changes to the QM-1 version were not completed when the model validation was complete, the experiment was started using the AIRLAB VAX computers. After the QM-1 version became available, it was used to complete the experiment. Figure 4.8 summarizes the use of AIRLAB computing resources during the experiment. It should also be noted that more than 300K blocks of disk storage were in use at various points during the experiment. Figure 4.9 details some interesting experiment-related numbers. One interesting observation is that the total amount of real time operation of the C/I that was simulated was only two seconds. Another interesting number is the average stack size (12 devices) for the emulation algorithm. This number indicates the average number of devices whose outputs change during a given simulation step. It is indicative of the potential savings in processing time that could be realized by an event-driven simulation versus a simulation that must compute all device outputs for each event time. The ratio of average stack size to total network size (12:5000) is substantial and represents a considerable reduction in device computations. However, it is also indicative of a design that has low device activity. A ratio of 1:20 should be more typical. From the numbers in figure 4.9, it is possible to estimate the average device computations per second required for an equivalent non-event-driven simulation of this network. Figure 4.10 summarizes these calculations. It should be noted that for fault simulation, a time resolution equivalent to 1/2 or 1 clock cycle is often sufficient. It was found that the QM-1 version operated about 60K:1 slower than real time for this network, and that the VAX 11/780 was about 1.2×10^6

Group	Number of Faults
Local Data Bus	128
Source Bus	370
Status Register	160
Current Status Update	106
Source Select	120
Syndrome Or	116
Voter/MUX/Result Bus	264
Voter Syndrome	132
Mask Transform	62
Microsequencer & Control Fan-out	216
	1674

Figure 4.4. Fault Sets by Functional Group

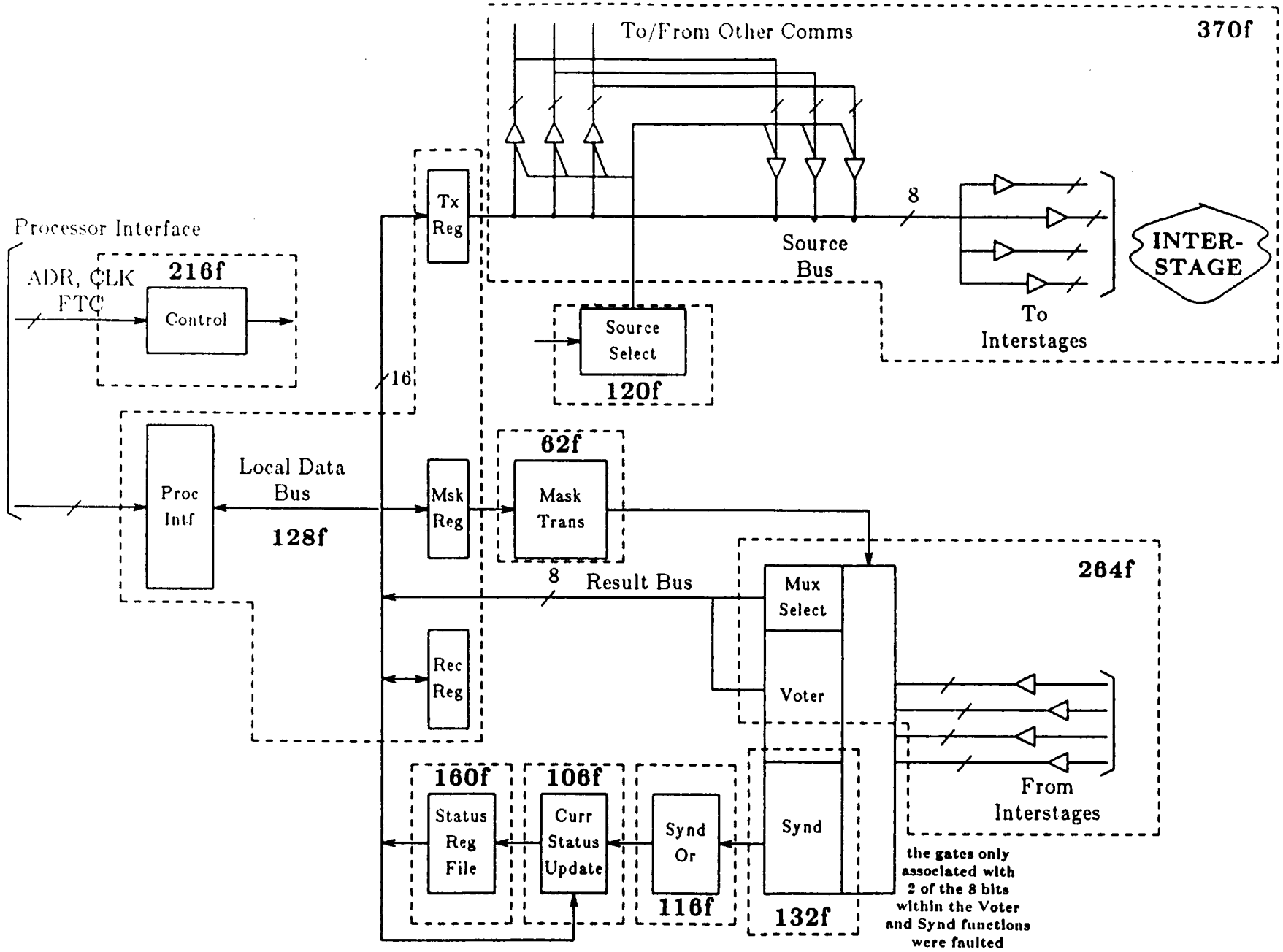


Figure 4.5. FTP Communicator Fault Set Boundaries

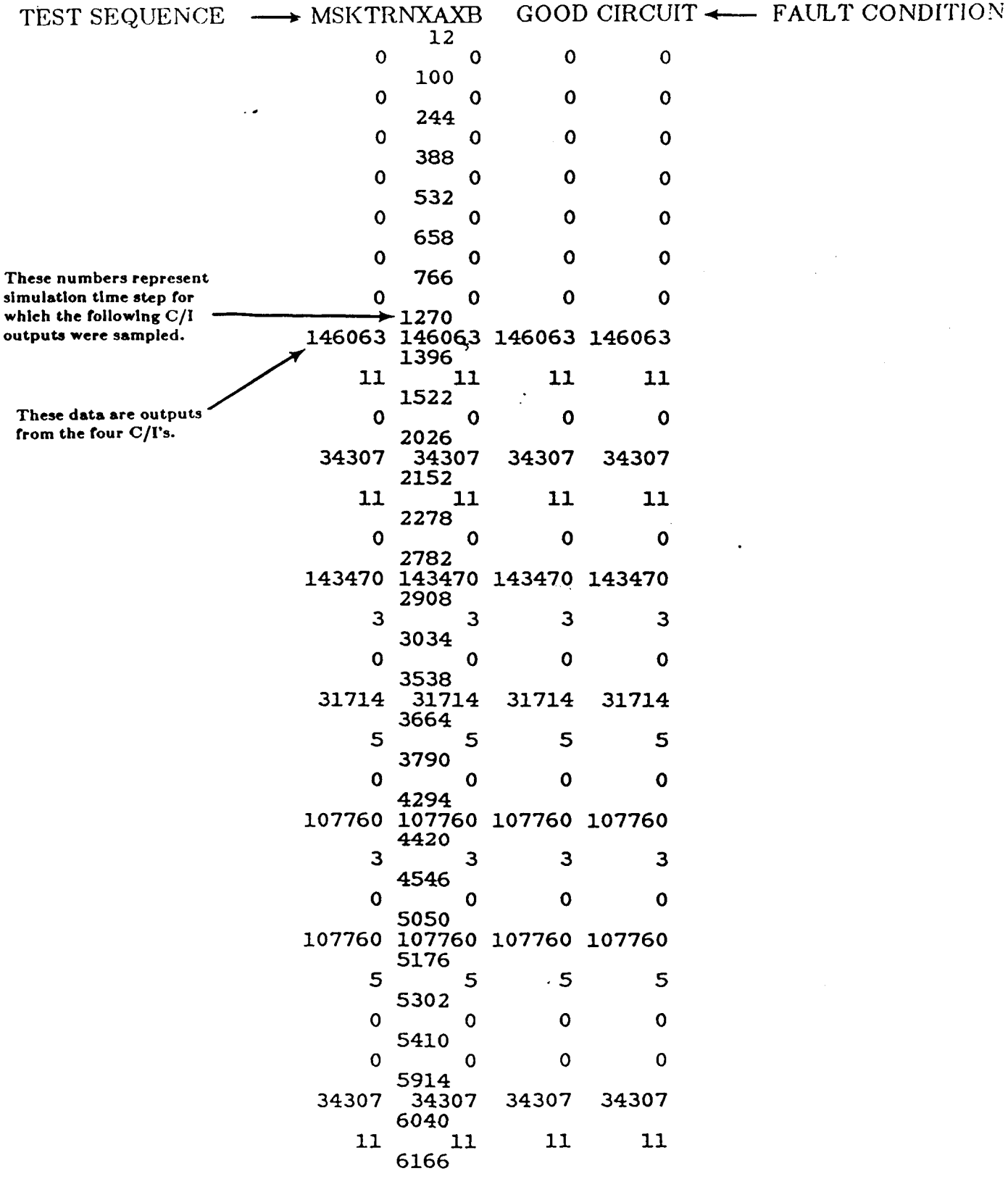


Figure 4.6. Typical Good Circuit Output

Fault file: CSTUP1FAULT.DAT
Vector file: CSTUPD.VEC
Data file: CSTUP1OUT.DAT

Fault Conditions
2 = no fault
3 = stuck at "0"
4 = stuck at "1"

Run number	Device	* Fault Condition	Vector number 1	2	3	4	5	6	7	8	9	10	
2,	APA1G01	4	1	1	1	1	1	1	1	1	1	1	10
3,	APA1G01	3	1	1	1	1	1	1	1	1	1	1	10
4,	APA1G02	4	1	1			1	1	1	1			6
5,	APA1G02	3			1	1					1	1	4
6,	APA1G03	4	1	1	1	1	1	1					6
7,	APA1G03	3							1	1	1	1	4
8,	APA1G04	4			1	1	1	1	1	1			6
9,	APA1G04	3	1	1							1	1	4
10,	APA1G05	4	1	1	1	1			1	1			6
11,	APA1G05	3					1	1			1	1	4
12,	APA1G06	4	1	1	1	1	1	1	1	1			8
13,	APA1G06	3									1	1	2
14,	APA1G07	4	1	1	1	1	1	1	1	1			8
15,	APA1G07	3									1		1
16,	APA1G08	4	1	1	1	1	1	1					6
17,	APA1G08	3							1		1		2
18,	APA1G09	4	1	1	1	1			1	1			6
19,	APA1G09	3					1				1		2
20,	APA1G10	4	1	1			1	1	1	1			6
21,	APA1G10	3			1						1		2
22,	APA1G11	4			1	1	1	1	1	1			6
23,	APA1G11	3	1								1		2
24,	APA1G12	4											0
25,	APA1G12	3	1	1	1	1	1	1	1	1	1	1	10
26,	APA1G13	4	1	1	1	1	1	1	1	1			8
27,	APA1G13	3											0
28,	APA1G14	4											0
29,	APA1G14	3	1	1	1	1	1	1	1	1	1	1	10
30,	APA1G15	4	1	1	1	1	1	1	1	1			8
31,	APA1G15	3											0
32,	APA1G16	4			1		1		1				3
33,	APA1G16	3	1	1	1	1	1	1	1	1	1	1	10
34,	APA1G17	4			1	1	1	1	1	1			6
35,	APA1G17	3	1	1							1	1	4
36,	APA1G18	4			1	1	1	1	1	1			6
37,	APA1G18	3											0
38,	APA1G19	4			1	1	1	1	1	1			6
39,	APA1G19	3											0
40,	APA1G20	4			1	1	1	1	1	1			6
41,	APA1G20	3											0
42,		2											0
43,	APA1G21	4			1	1	1	1	1	1			6
44,	APA1G21	3	1								1		2
45,	APA1G22	4			1	1	1	1	1	1			6
46,	APA1G22	3	1	1							1	1	4
47,	AFPPA1F00	4			1	1	1	1	1	1			6
48,	AFPPA1F00	3	1	1							1	1	4
49,	APA1G23	4	1	1							1	1	4
50,	APA1G23	3			1	1	1	1	1	1			6
51,		2											0
			22	20	26	24	26	24	26	24	20	14	

Figure 4.7. Typical Post-Processing Output

Test	VAX 11/750	VAX 11/780	QM-1
Presence		34 hrs.	
Basic	84 hrs.		
Current Status Update		38 hrs.	
Mask Transform XAXB		65 hrs.	
Mask Transform Quad	72 hrs.		
Voter XA			6.6 hrs.
Voter XB			6.6 hrs.
Voter XC	244 hrs.		
Voter XD		122 hrs.	
	400 hrs.	259 hrs.	13.2 hrs

Figure 4.8. Emulation CPU Time*

*Does not include post-processing

Network size	>5000 devices >15000 interconnections
Simulation resolution	1 gate delay - 5ns
Gate delays per clock	18 — 120 ns
Clocks/Fault tolerant clock	12*
Average stack size of Event-driven simulation	12 devices
Maximum stack size of Event-driven simulation	$\sim$ 500 devices
Total faults	1670
Total C/I transactions cycles/fault	2034
Transaction cycles in experiment	$\sim 3 \times 10^6$
Simulation steps	$\sim 3.6 \times 10^8$
Total real time	$\sim$ 2 seconds

*Actual fault tolerant clock has 40 processor clock periods. The simulation was scaled to 12 to reduce run time.

Figure 4.9. Some Diagnostic Emulation Experiment Numbers

Processor	Gate Events/Sec	Real Time Ratio
VAX 11/750	1×10^3	2.4×10^6 : 1 slow
VAX 11/780	2×10^3	1.2×10^6 : 1 slow
QM-1	37×10^3	60×10^3 : 1 slow

- Establishing equivalence to a non-event-driven simulation is difficult

- If equivalent time resolution is required for the non-event-driven simulation, the comparable speeds are

VAX 11/750	500k Gates/Sec
VAX 11/780	1×10^6 Gates/Sec
QM-1	20×10^6 Gates/Sec

- If 1/2 clock period time resolution is required, then the comparable speeds are

VAX 11/750	50k Gates/Sec
VAX 11/780	100k Gates/Sec
QM-1	2×10^6 Gates/Sec

Figure 4.10. Summary of Diagnostic Emulation Speed

slower than real time.

4.3. Analysis of Experiment Results

The experiment data presented in this section are not adjusted or modified to account for certain modeling problems identified after the experiment was complete. These problems will be discussed in a later section. Primarily, results are organized by diagnostic test and fault set. For each self test/fault set pair, the percentage of faults detected relative to the number of faults injected within the fault set are presented. Test vectors with the best and worst scores for each test/fault set pair are identified. The relative and combined performance results of the self tests are presented. Finally, the percentage of faults found by the best vector within a test relative to the number of faults detected by the test are examined.

Before reviewing these results, it should be noted that conclusions and observations are subject to the following:

1. There are limitations due to the adequacy of a gate level fault model for the C/I implementation technology;

2. Faults were only injected in a single emulated C/I;

3. The number of faults within a fault set varies;

4. Self tests reflect RTI's understanding of the CDSL self test description;

5. Faults were injected in devices associated with two of the eight voter bits.

The Presence (P) test results shown in figure 4.11 are of particular interest. CDSL executes all vectors of this test prior to every FTP processing frame (typically 10 msec to 40 msec) under their FDIR program. Presence is a short test (small number of test vectors) relied on to establish the general health of the C/I channels. Note that the results for the Mask fault set are not shown due to loss of these computer data. As can be observed from figure 4.11, this test is very effective for detecting data path faults (Local Data Bus and Source Fault Sets). However, the performance for other fault sets such as the Voter Syndrome set is about 50%. Overall, the performance of this test is about 62% against the injected faults. However, when adjusted for the devices not faulted in the Voter and Voter Syndrome fault sets, the performance is closer to 55%. Note also that both the single best and worst test vectors for the entire test will detect about 50% of the faults. Since this test is executed frequently, the average latency time for about half of all C/I faults is very short. Further, each vector is very effective relative to overall test performance. Since this test requires a small number of test vectors, it will require only a small portion of the FTP processing resources. These data confirm the CSDL assumption regarding the effectiveness of

Figure 4.11. Presence Test

Presence for data path faults.

Figure 4.12 gives the results for the Current Status Update (CSU) test. This short test is designed to test certain of the Status Register update logics, and of necessity must exercise parts of the error reporting logic. As can be observed and as is expected, this test performs substantially better than the Presence test for the Status Update and Syndrome Or fault sets. CSU test performance is considerably less than Presence for the data path fault sets and is about the same for the remaining fault sets. The overall performance is about 50% against the injected faults. When adjusted to include the devices that were not faulted in the Voter and Voter Syndrome fault sets, the performance is about 40%. The low CSU test performance for the Source Select fault set should also be noted.

The results for the two Mask Transform tests are given in figures 4.13 and 4.14. The Mask Transform Quad (MTQ) test has 12 test vectors and the Mask Transform XAXB (MTA/B) test has 24. Both tests target the C/I reconfiguration logic. Overall, both tests perform better than Presence. As should be expected, their performance against the mask fault set is somewhat better than the ubiquitous 50%. Also, their performance against the Voter Mux fault set is good. Their overall performances are near 70%. When adjusted for the additional voter bits, their performance is still above 65%.

The results for the last non-voter test, Basic (B), are given in figure 4.15. This test is an abbreviated version of the one of the tests designed by RTI for the validation of the C/I model. These results are included as a point of reference. Since the test was not optimized against a comprehensive fault model, it is not intended as a recommended test. It does demonstrate that simulations can be used to consider alternative tests and demonstrates the potential for improvement in test performance. Basic requires 16 test vectors. The Basic test performs well on the data path faults and has better performance than Presence against the Status Reg, Status Update, Syndrome Or, and Control fault sets. Further, its performance approaches that of MTQ and MTA/B for the Mask fault set. The overall performance of Basic is in excess of 70% against the injected faults and exceeds 65% when adjusted for the devices not faulted in the Voter and Voter Syndrome functions.

The overall performance of each non-voter is shown in figure 4.16. The combined performance is close to 90% and is not significantly affected when adjusted for the devises not faulted in the Voter and Voter Syndrome functions. Figure 4.17 shows the combined performance of all non-voter tests for each fault set. Also shown are the performance improvements provided by the voter tests. With the voter tests included, overall performance is 92%. The voter tests used were XA, XB, XC, and XD and required a total of 256 test vectors. The non-voter tests required a combined total of 72 test vectors. Figure 4.18 identifies and shows the performance of the best test for each fault set. Figure 4.19 shows the performance of the worst test for each fault set. The extremely poor performance of CSU against the Source Select fault set is not shown. Instead, the next worst test, MTA/B is indicated. Figure 4.20 shows the performance of the best single vector for each fault set and for the entire set of test

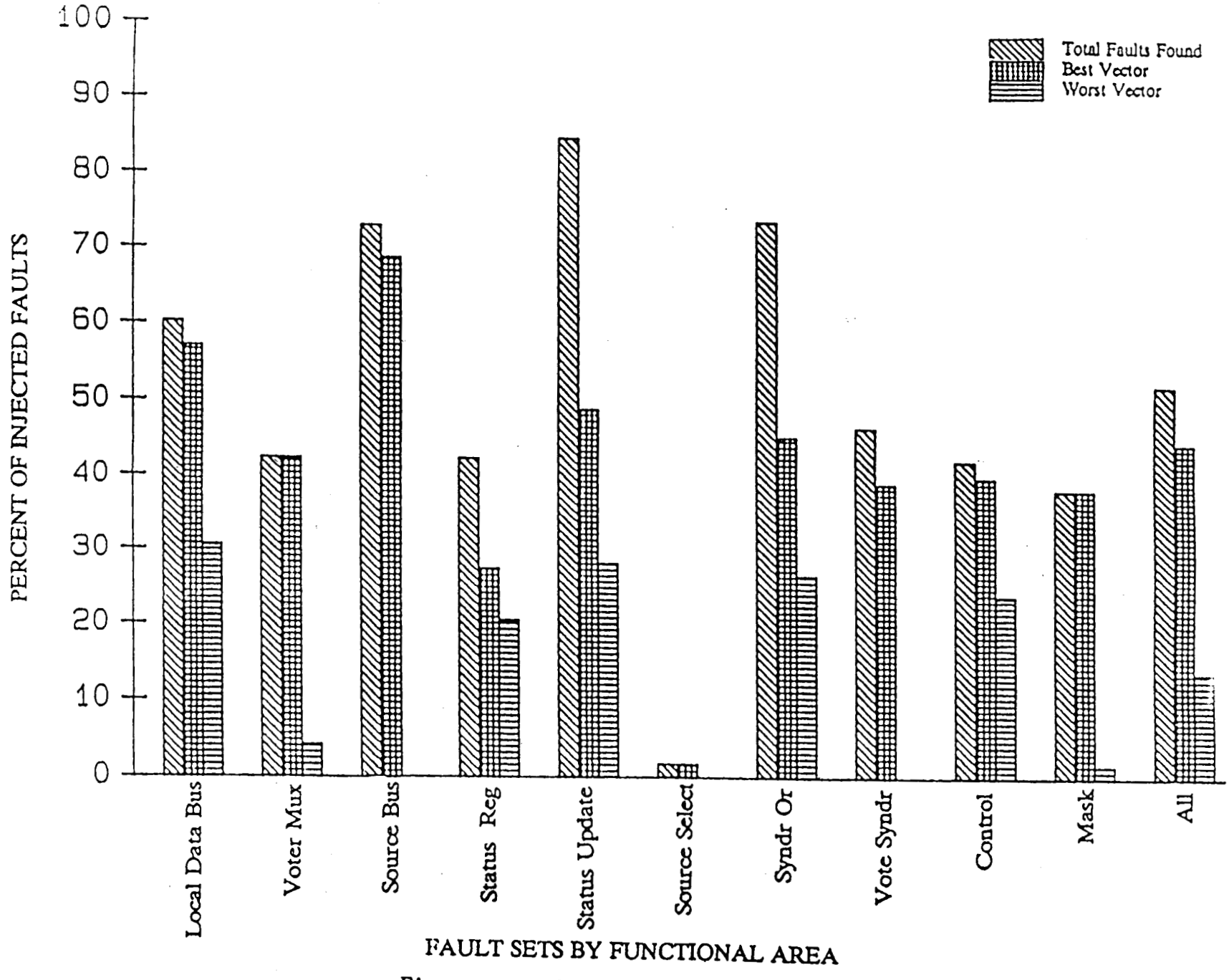


Figure 4.12. Current Status Update Test

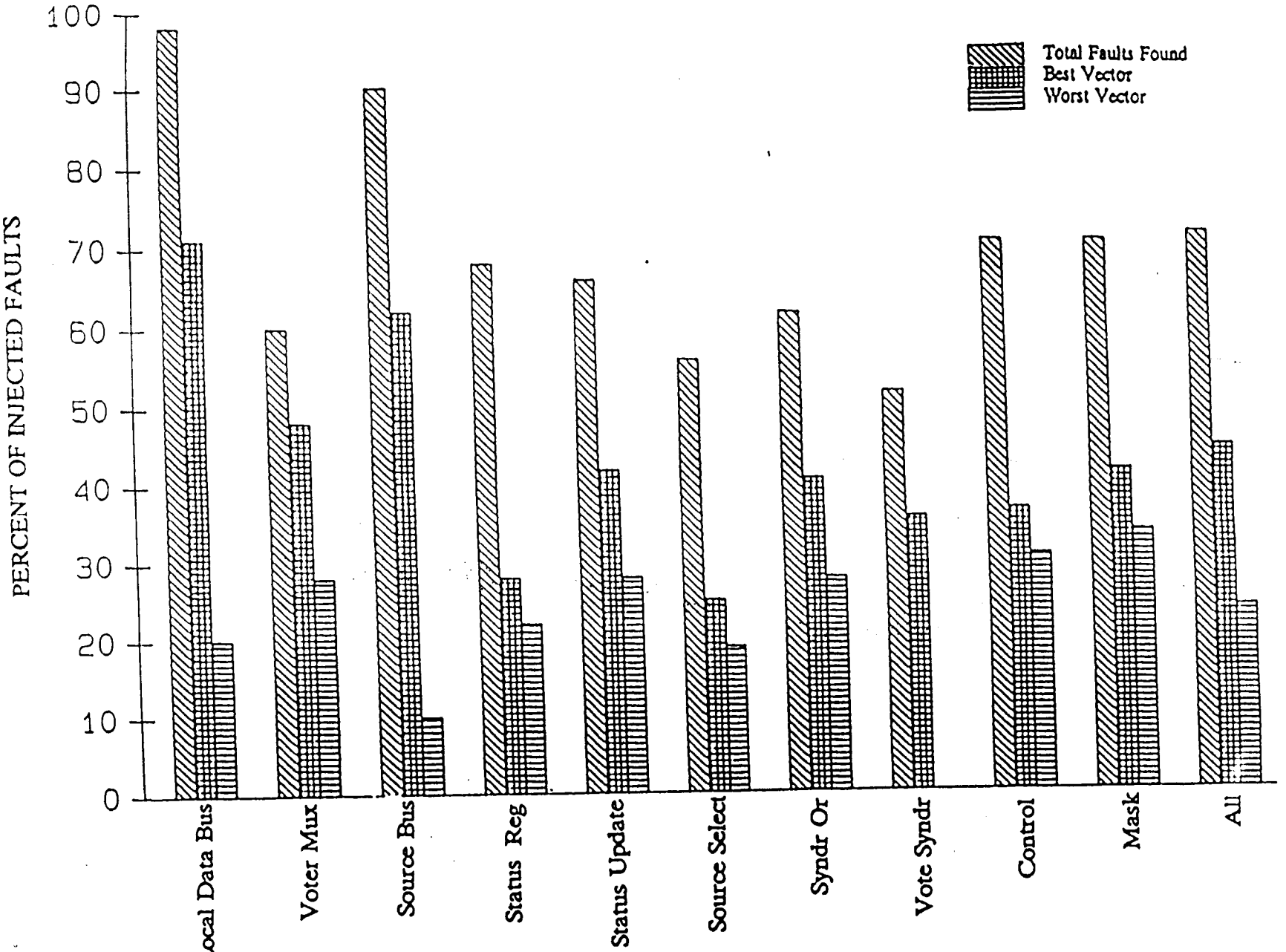


Figure 4.13. Mask Transform Quad Test

Figure 4.14. Mask Transform XAXB Test

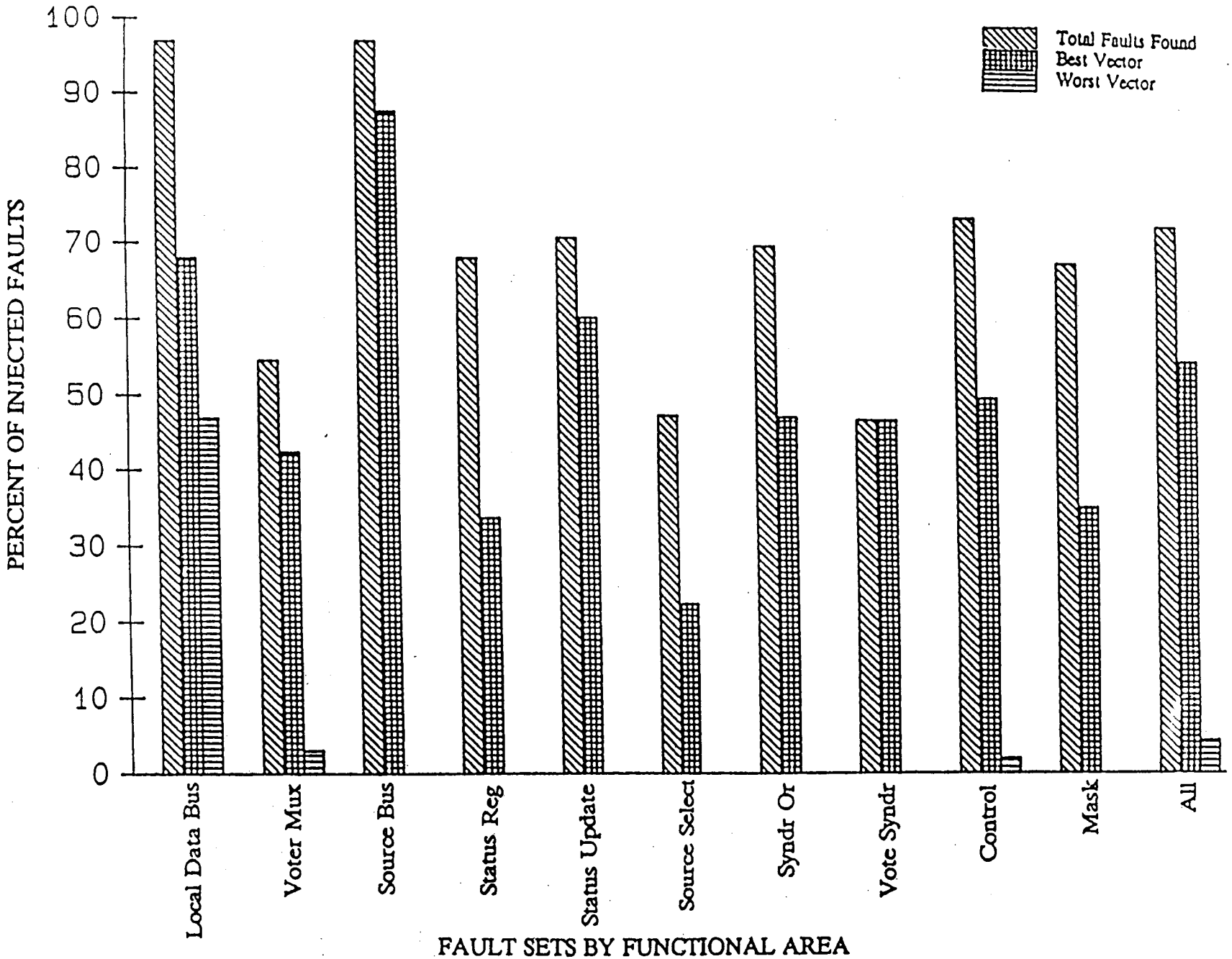


Figure 4.15. Basic Test

vectors. It is interesting to note that the performance of the best single vector is about 50%. Figures 4.21 through 4.25 show for each fault set and test the percentage of faults found by the best vector within the test relative to the total number of faults found by the test. These results tend to support an assertion that for C/I network configuration, a high percentage of the faults that can be detected by a given test will be detected by a small percentage of the test vectors. Such an assertion is not true for all network configurations and care must be exercised in the conclusions that can be drawn from these data. However, the concept of identifying those test vectors that are more effective in terms of detecting more faults has special significance for fault tolerant systems. The time required to find a fault is at most T and on average is $T2$. The average time that faults remain latent within a system can be reduced by using the more effective test vectors more frequently than the less effective vectors. The potential for reduction of fault latency time is discussed in more detail in section 4.5.

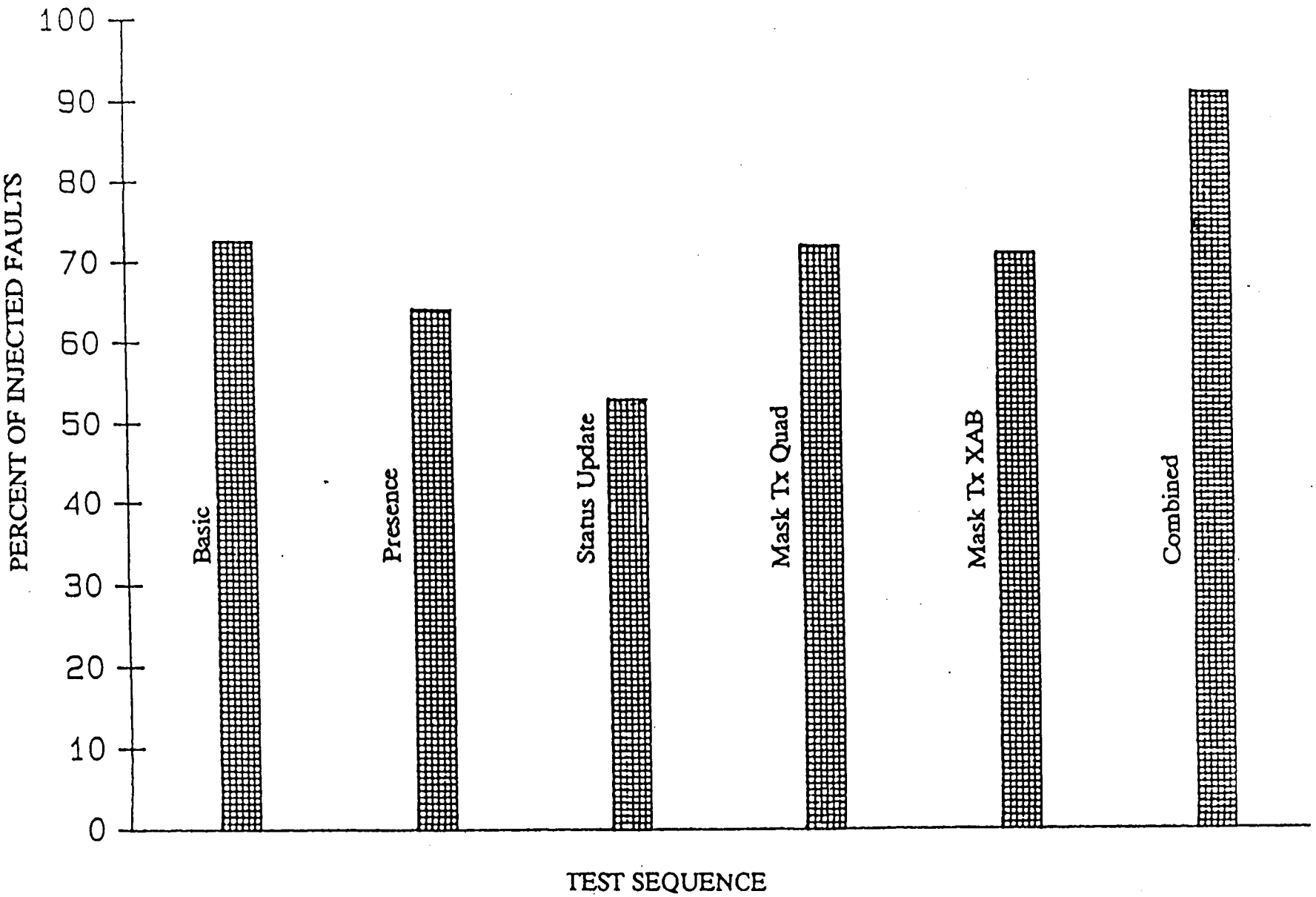


Figure 4.16. Diagnostic Test Performance

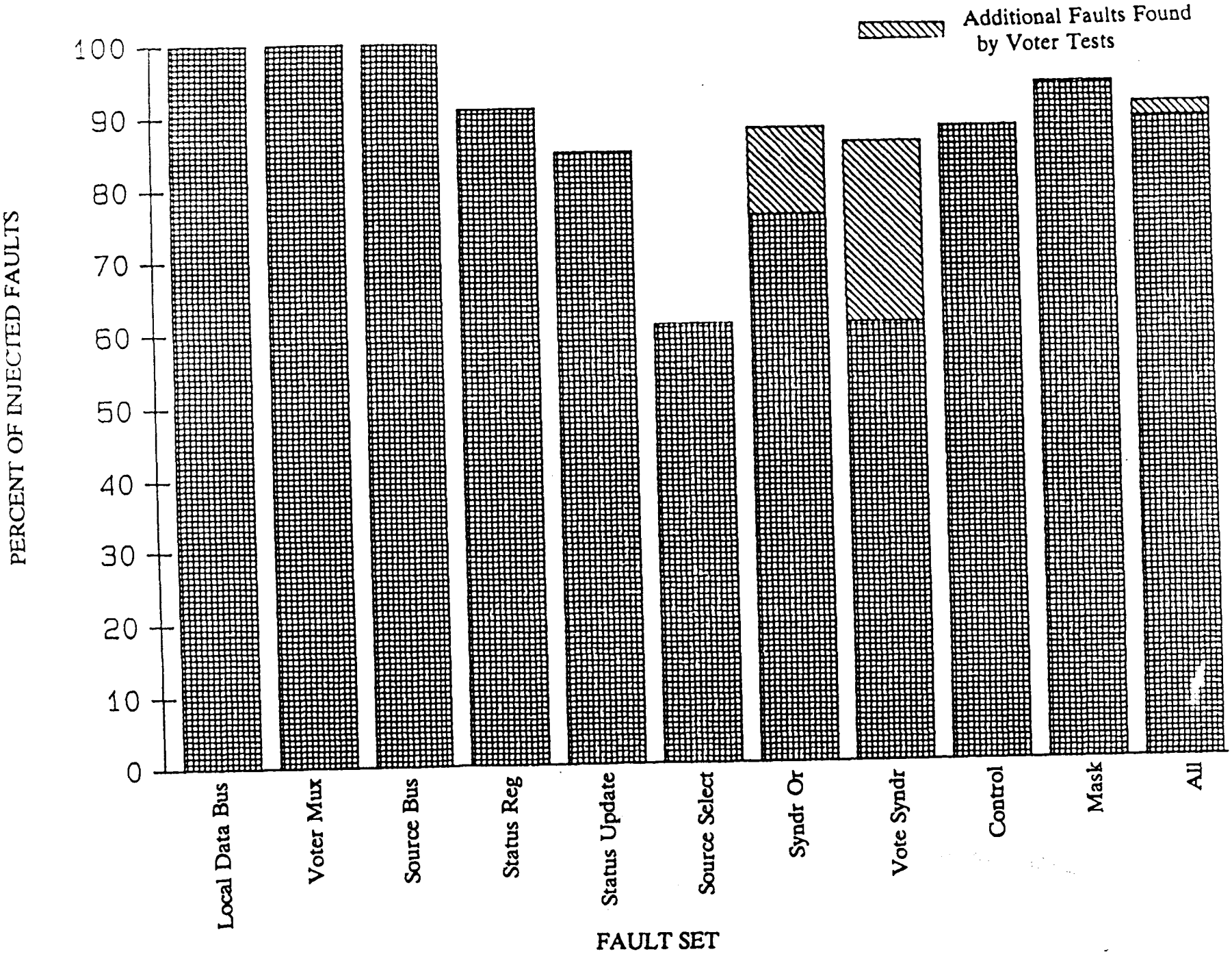


Figure 4.17. Combined Performance of all Non-voter Tests

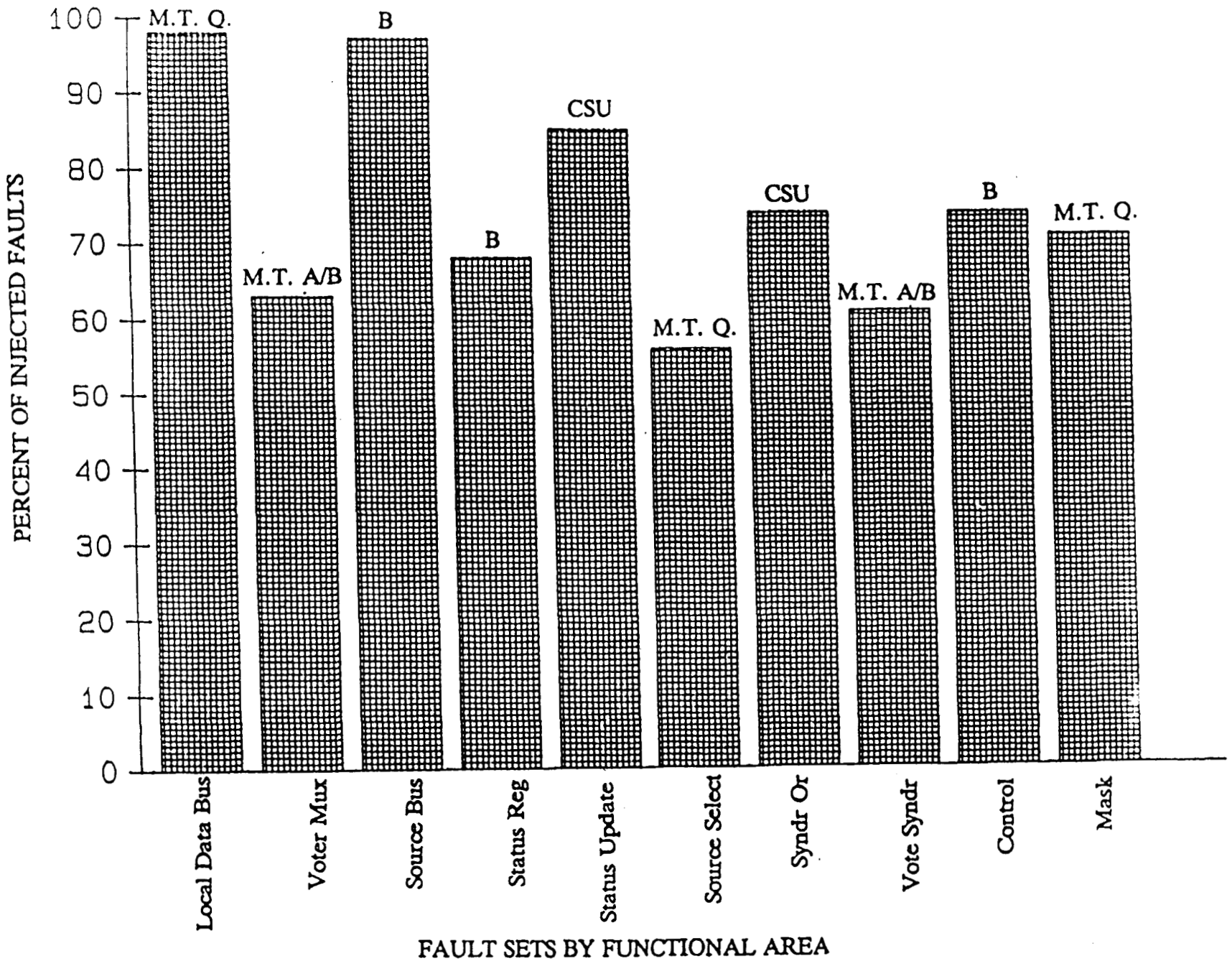


Figure 4.18 Performance of Best Test for a Given Fault Set

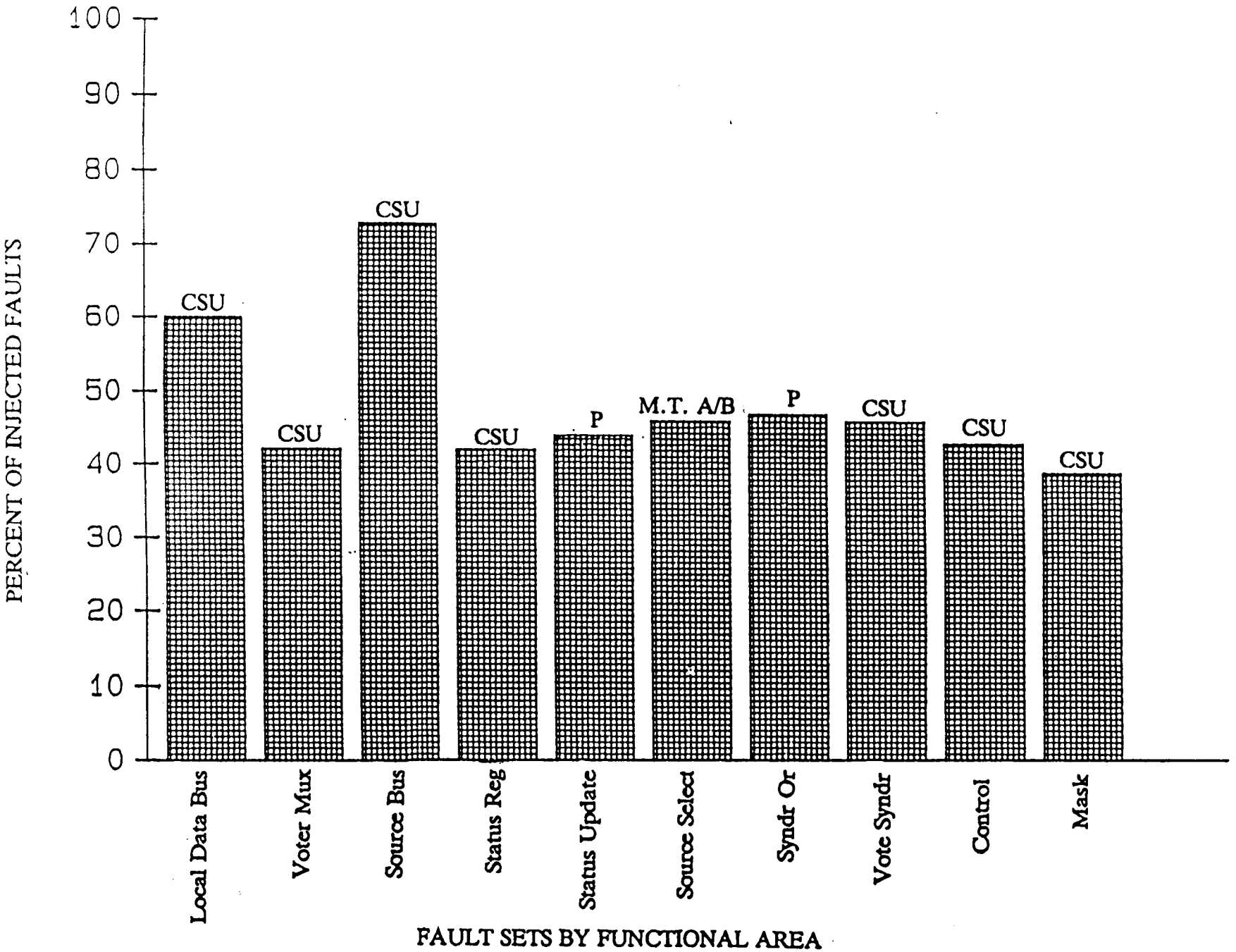


Figure 4.10. Performance of Worst Test for a Given Fault Set

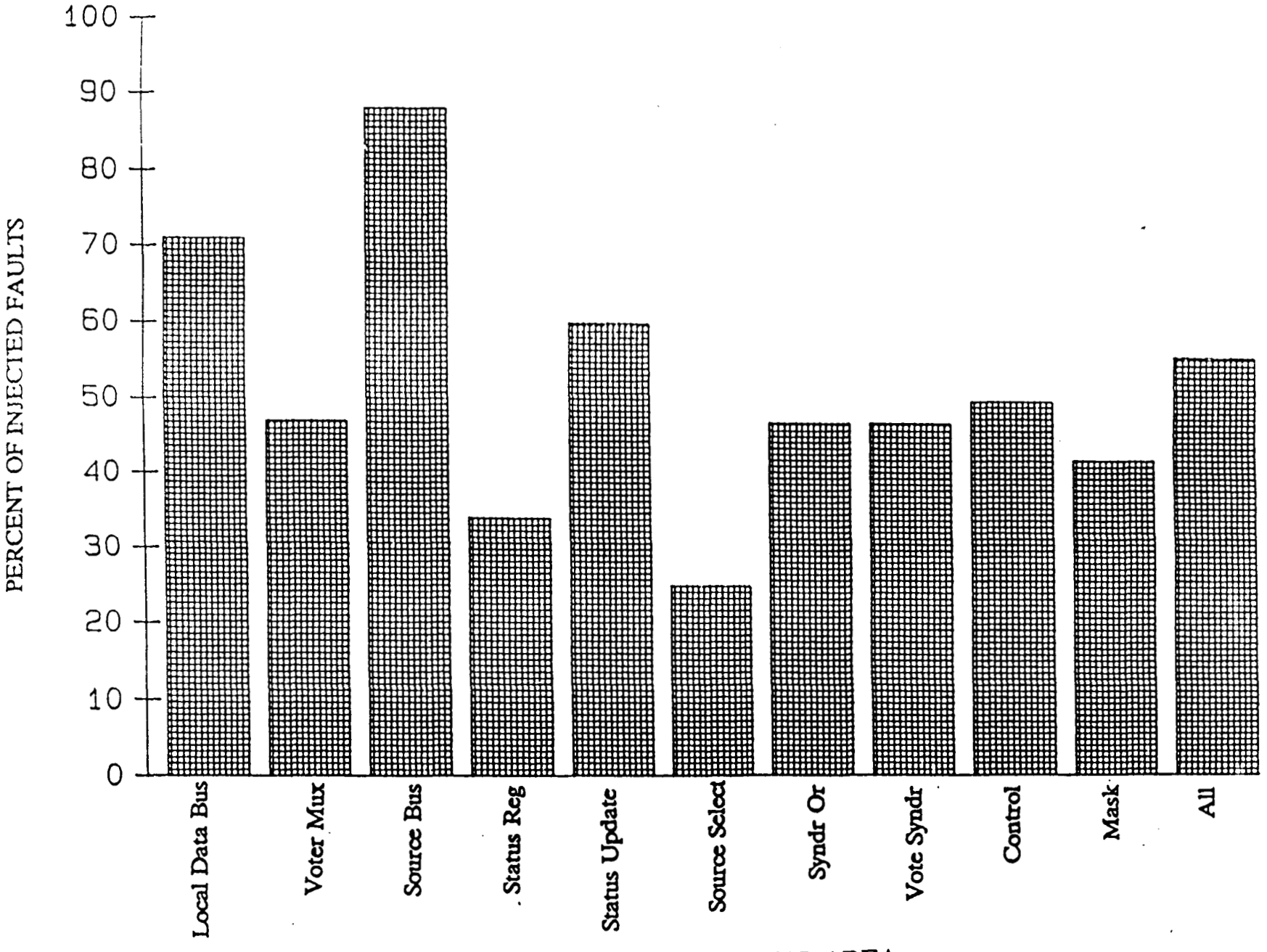


Figure 4.20. Performance of Best Vectors for Given Fault Set Across All Tests

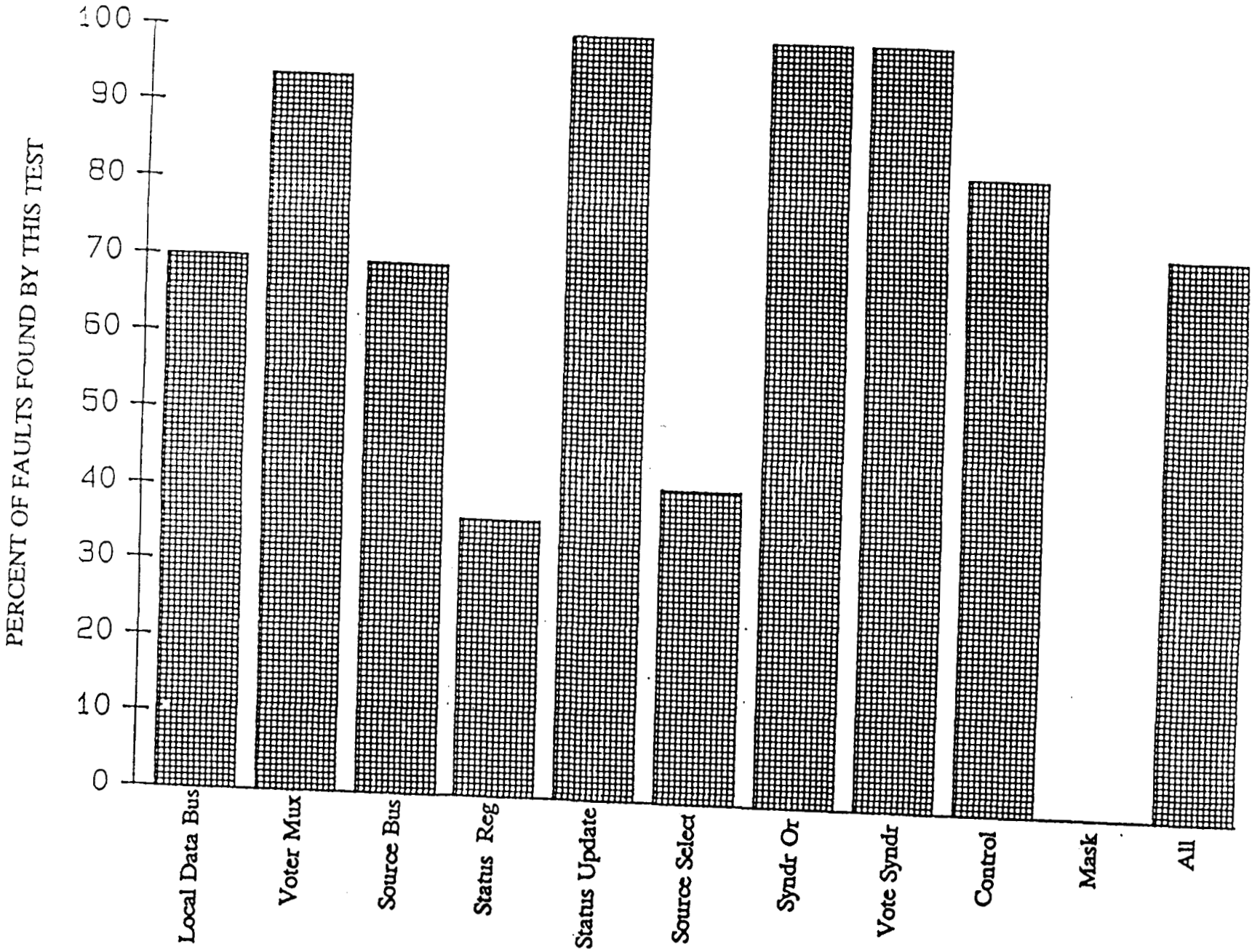


Figure 4.21. Presence Test (Performance of Best Vectors for Given Fault Set)

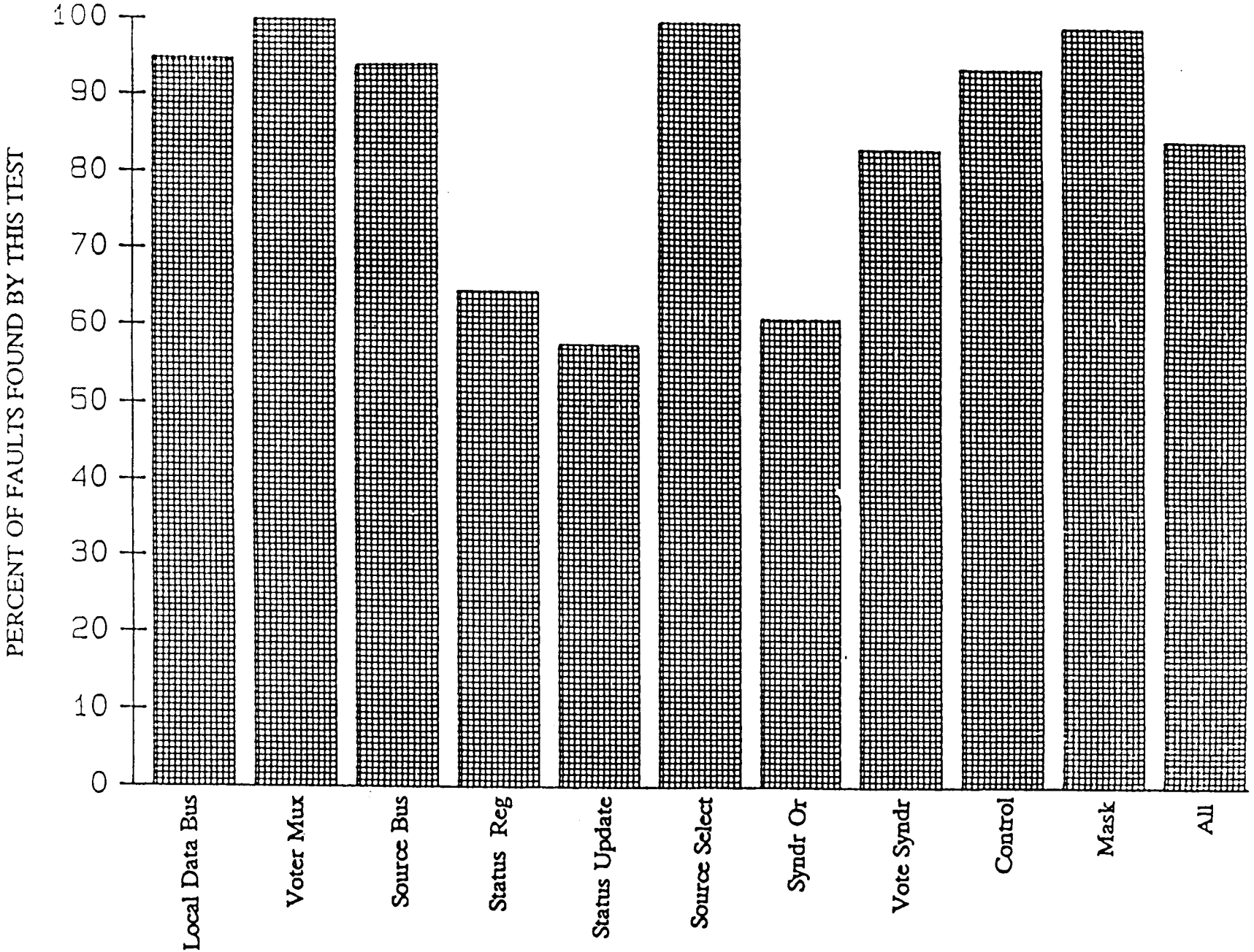


Figure 4.22. Current Status Update Test (Performance of Best Vectors for Given Fault Set)

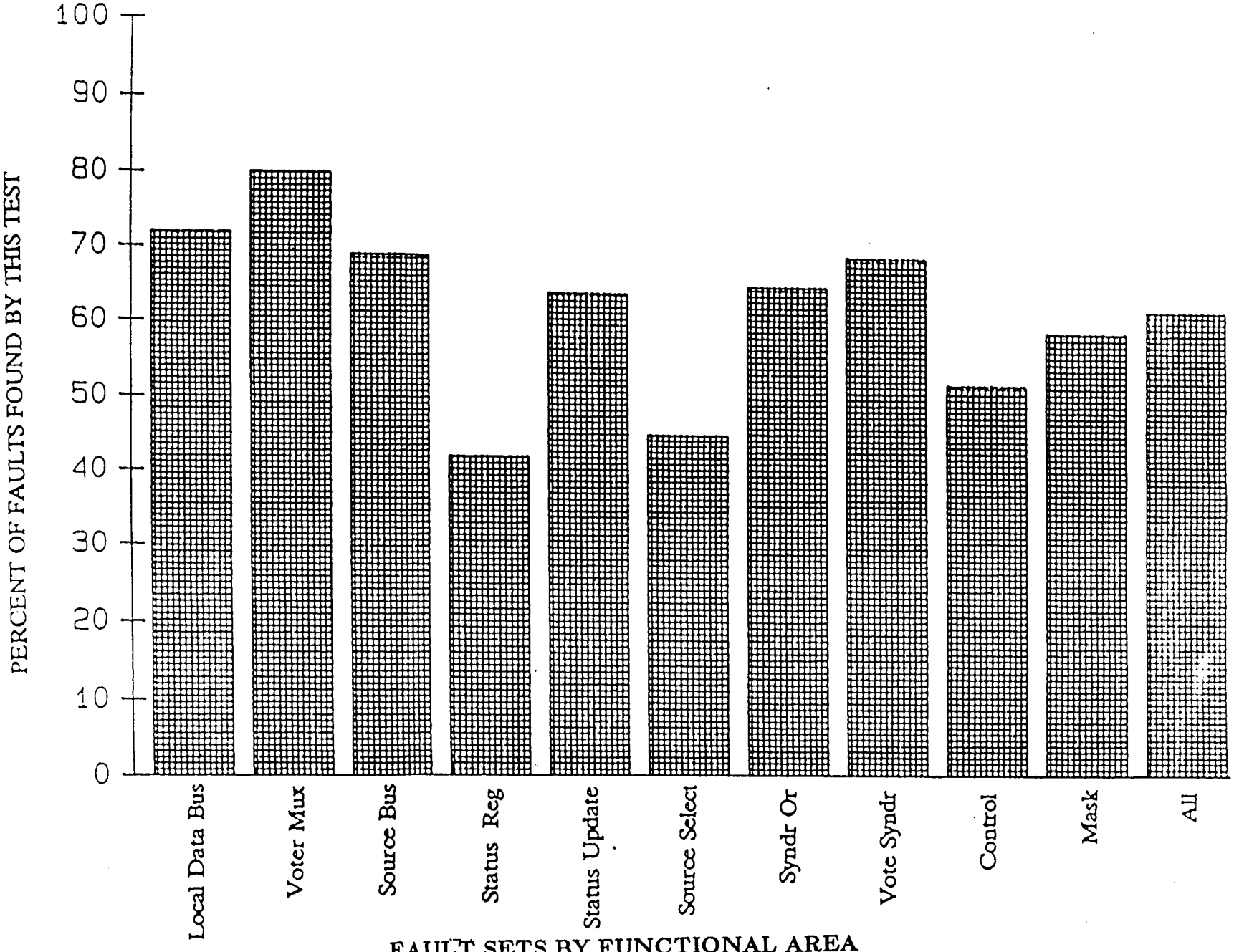


Figure 4.23. Mask Transform Quad Test (Performance of Best Vectors for Given Fault Set)

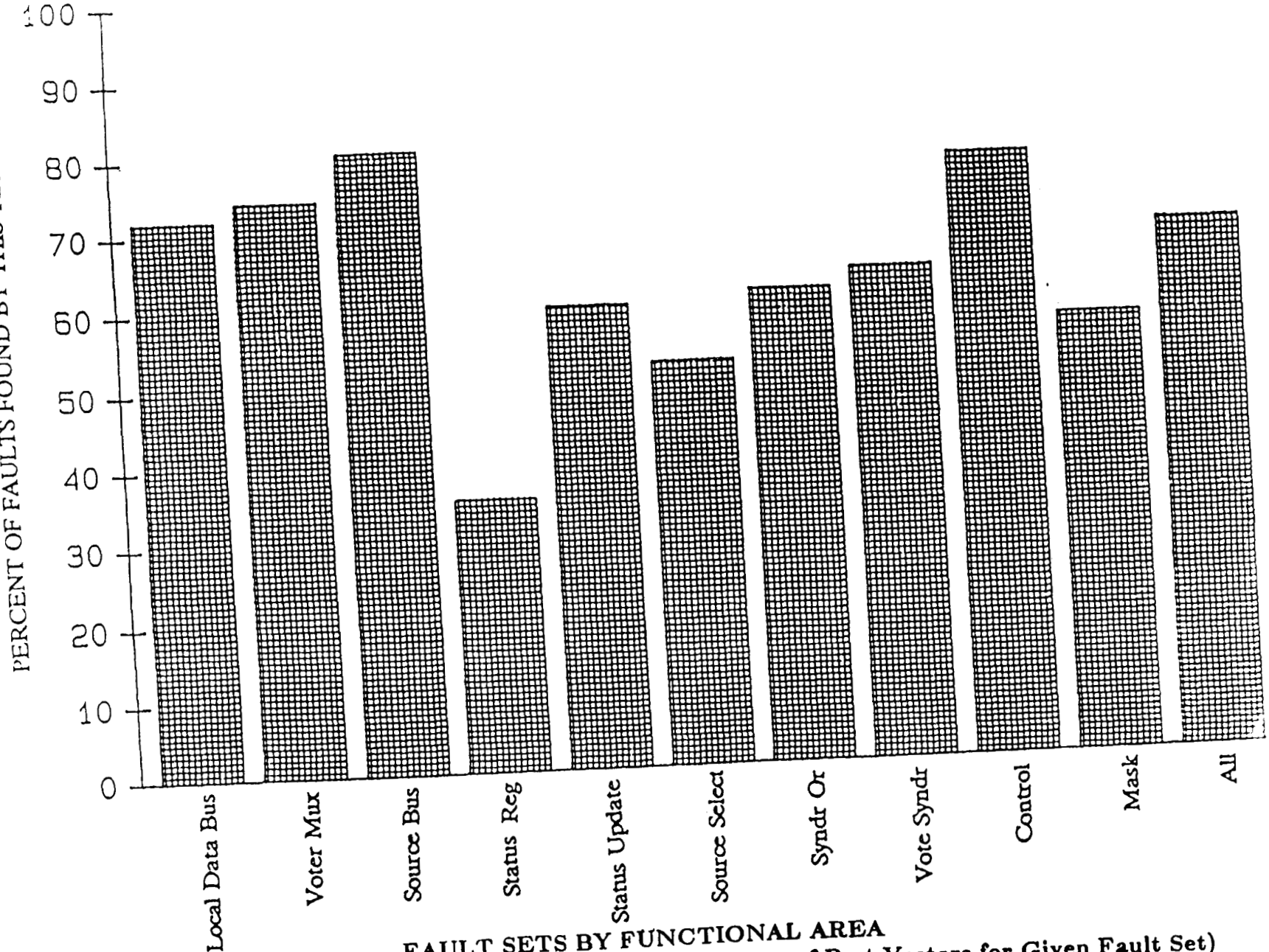


Figure 4.24. Mask Transform XAXB Test (Performance of Best Vectors for Given Fault Set)

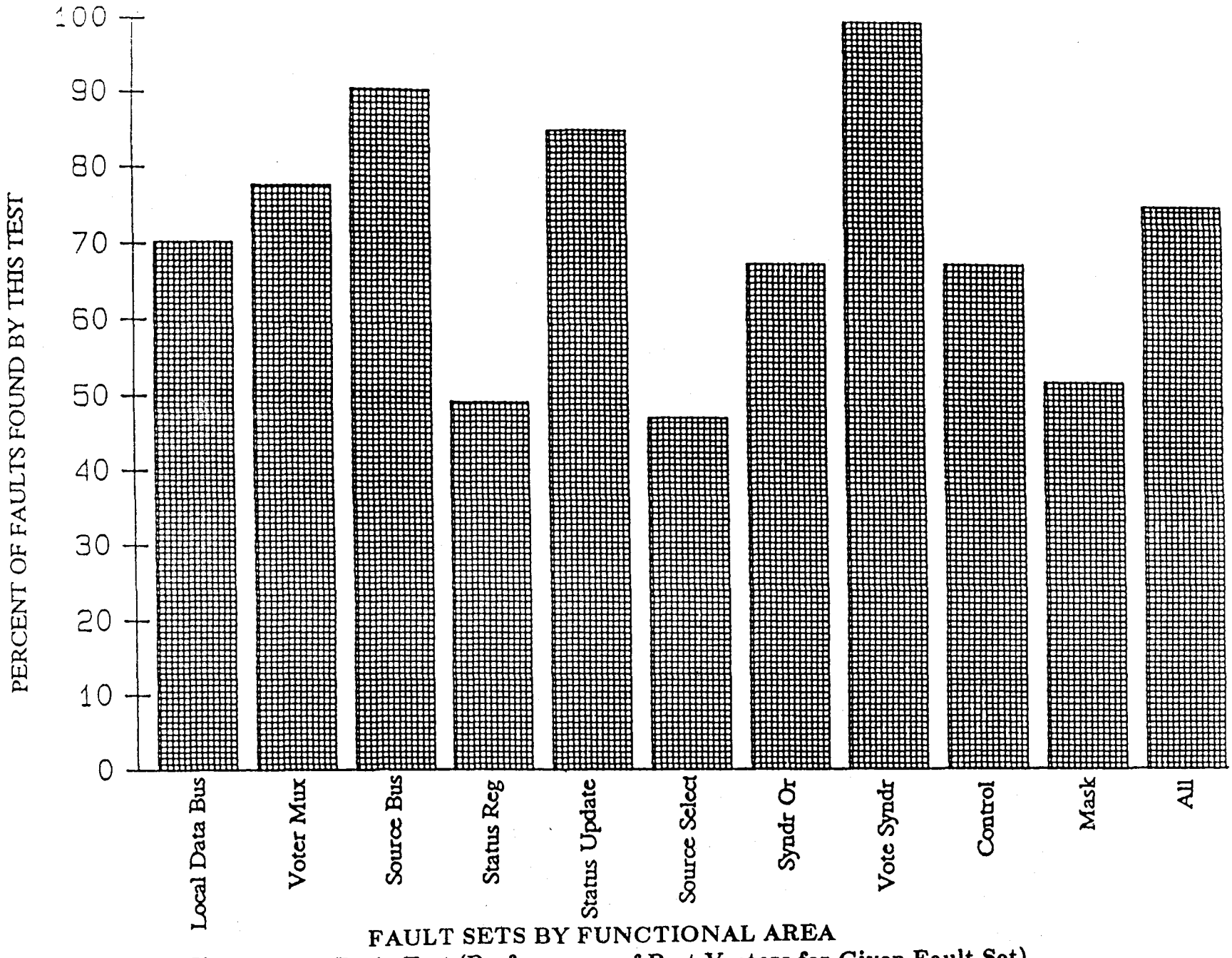


Figure 4.25. Basic Test (Performance of Best Vectors for Given Fault Set)

4.4. Review of Undetected Faults

A detailed review and analysis of the device faults that were not detected in this experiment is presented in this section. Failure to detect certain of the faults will be found to be due to bus modeling problems. Other fault detection failures will be found to be due to deficiencies in the tests used. A few fault detections failure will be found to be undetectable by any test.

The total number of faults that were not detected by any test was 141. Of these, 47 were found to be contained in the Source Select fault set. Also, figure 4.17 shows that the combined performance against Source Select faults is less than for any other fault set. A detailed examination of the devices associated with these undetected faults indicated that most, if not all, were associated with one of two problem areas. One problem area was in a group of devices whose faulted condition could not be detected due to the bus modeling problem discussed in section 2.2. An accurate model of the cross channel interconnections would likely have resulted in these faults being detected.

The remaining undetected faults in the Source Select fault set were associated with the C/I channel ID logic. The ID logic permits C/I modules to be used for any C/I channel. When a module is used in a channel A processor, hardwired inputs force the ID logic to allow the module to function as the channel A C/I. Similarly, ID logic establishes the identity of modules used in other channels. If certain of the devices associated with channel ID are faulted in such a manner that the local channel identity is unchanged, these faults are indistinguishable from fault-free devices. Since there are no provisions for stimulating different channel ID inputs under processor control, none of these faults can be detected. It is not clear whether the effects of these faults are of significance. If the devices for a given C/I fail in the manner described, the C/I module will continue to operate properly so long as the module is always used for that channel. Non-coverage of these faults could, however, affect maintenance procedures. It is not uncommon for replicate modules within a system to be interchanged with a module that is suspected of being faulty. The potential for ambiguous results exists if an apparently functioning module with one of these latent faults is substituted for a suspected faulty module.

The fault set with the next largest number of undetected faults is the Control set with 26. Examination of the associated devices indicated several problem areas. Again, certain of the undetected faults were traced to the cross channel bus modeling problem and to the ID logic problem. Several other undetected faults were traced to devices that provide certain control signals directly to the processor interface. These faults would be detected by the associated FTP channel processor. Either these devices should not have been included in the fault set, or their outputs should have been included in the C/I model's external outputs and subsequently checked for proper state in the experiment post-processing. Finally, three undetected faults were found to be associated with devices which propagate the system reset signal. Failure of these devices in a state equivalent to an inactive reset signal cannot be detected if the failure occurs after a valid reset has occurred. The significance of such latent faults is open to

speculation. Failure to be able to reset could have only transient and inconsequential effects on a system provided that the C/I microsequence controller always returns itself to an idle state. However, these faults are undetectable during normal operation and as such represent a potential problem.

The Voter Syndrome fault set had 19 undetected faults. After examination of the associated devices, it is believed that the Voter Quad test, which was not used for this experiment, should detect these faults. Although planned for the experiment, the use of the Voter Quad test could not be completed prior to the end of the experiment.

The Status Update fault set contained 16 undetected faults and the Syndrome Or fault set contained 14 undetected faults. An assessment of these faults was inconclusive, but it is believed that the Voter Quad test would detect some portion of these.

The Status Register fault set contained 15 undetected faults. An examination of these determined that the decode logic in the Status Register memory was associated with the undetected faults. They are undetected because the C/I Status Registers automatically reset after a Read Status Register operation and because the CSDL self test diagnostics, as understood by RTI, include a Read Status Register for each test vector. Unless a test causes different errors to be written in each Status Register prior to a read of any of the registers, address decode logic faults cannot be distinguished from fault-free behavior. A simple modification of the self tests should render these faults detectable.

The four undetected faults in the Mask fault set were associated with the ID logic problem described above. Fault sets, Source Bus, Voter Mux, and Local Data Bus contained no undetected faults.

Table 4.1 summarizes the undetected faults, the causes for not being detected, and the estimated relevant undetectable faults.

If the assessments of the undetected faults are correct and if the ID logic faults are considered inconsequential, the adjusted performance for the C/I self tests would exceed 98% and could approach 99.8%. If the reset faults are inconsequential, the performance approaches 100%.

Fault Set	Undetected Faults	Source	Comments	Estimated Undetectable
Source Select	47	Cross channel model & ID logic		0
Control	26	ID logic & cross channel + product interface & reset		3
Voter Syndrome	19		Run Voter Quad Test	0
Status Update	16	not known	Run Voter Quad Test	0-16
Syndrome Or	14	not known	Run Voter Quad Test	0-14
Status Reg.	15	Status Reg address decode	modify self test	0
Mask	4	ID logic		0
Local Data Bus	0			0
Voter Mux	0			0
Source Bus	0			0
Combined	141			3-33

Table 4.1. Summary of Undetected Faults

4.5. Inferences Drawn from Experiment Results

As indicated in the previous section, care must be exercised in the use of the experiment results. Performance numbers derived for the various tests are subject to the limitations of fault models, network models, and the correctness of the self tests. While such results cannot be considered to be precise, they can be used in some instances to support specific observations and conclusions. This section reviews the inferences drawn from the experiment data.

The experiment results support the CSDL assumption that the Presence test will detect a high percentage of the C/I data path faults. In addition, Presence detects approximately 60% of device output faults.

Certain faults occurring in the C/I ID logic and the system reset logic cannot be detected. The significance of these undetectable faults is open to speculation and an assessment of such is beyond the scope of this report.

Particular faults occurring in the Status Register address decode logic cannot be detected by reading the Status Register after every test vector input to the C/I. If the CSDL self tests use this procedure, these faults will be undetectable. A simple modification of a portion of the self test would remedy this problem.

Finally, the results suggest that certain of the test vectors detect a considerably larger number of faults than do other test vectors. Even though certain test vectors are less productive, they may be required to detect faults that are undetectable with the most productive test vectors. Average fault latency time could be reduced if these more productive test vectors are used more frequently than the less productive, but necessary, test vectors.

To demonstrate the potential for reduction in average latency time for faults within a network, consider a test composed of V different test vectors which detects all network faults. Assume that each vector can be imposed on the network in time T. If all faults are equally likely to occur and if each vector is used once per test repetition, the average latency time for faults is $VT/2$. Suppose that 4/5 of the faults are detected by 1/5 of the test vectors. Define a new test composed of the same test vectors. Since the prime test vectors are four times more effective than the remaining test vectors, repeat the prime vectors four times more frequently than the remaining test vectors. The time required to complete such a test would be 8/5 VT. The average latency time for a fault would be 8/25 T. Thus, the new test would give a shorter average latency time. This would directly reduce fault recovery times, which would reduce the exposure to near-coincident faults.

Consider the average latency of the C/I FDIR self tests. From section 3.5, it is known that the Presence test is repeated in every processing frame and that the remaining C/I self tests could require approximately two minutes to complete. Assume that a processing frame is typically 40 msec. Experiment results indicate that the performance of the Presence test is about 60%. The remaining 40% of the faults must be found by the slow FDIR self tests. Average latency time for C/I faults using FDIR becomes

$$\bar{L} = \frac{1}{2}\left[0.6 \text{ x } 0.04 \text{ sec} + 0.4 \text{ x } 120 \text{ sec}\right] \approx 24 \text{ sec.}$$

Experiment data indicates that the performance of the Presence test could be improved by modest increases in the number of test vectors. Both Basic and Mask Transform Quad have better performance than Presence and require fewer test vectors. Assume that this opportunity is not exploited. Experiment results also indicate that the combined performance of all non-voter tests (72 test vectors total) is on the order of 90%. When adjusted for model discrepancies and the ID logic, the performance is closer to 95%. Assume that a new FDIR test is designed. Assume that it consists of 100 test vectors that are executed with a frequency of 5 seconds and that it detects 35% more faults than does the Presence test. The slow FDIR is expected to detect the remaining 5% of the faults. The average latency for this scenario becomes:

$$\bar{L} = \frac{1}{2}\left[0.6 \text{ x } 0.04 \text{ sec} + 0.35 \text{ x } 5 \text{ sec} + 0.05 \text{ x } 120 \text{ sec}\right] \approx 4 \text{ sec.}$$

The improvement in average latency from 24 seconds to 4 seconds is substantial.

The results of this experiment have been used to identify potential problem areas and to identify areas where performance can be improved by relatively modest changes to the self test diagnostics.

5. Conclusions, Observations, and Recommendations

The objective of this section is to briefly summarize the relevant conclusions and observations discussed in detail throughout this report. In addition, several efforts that are natural extensions to this experiment are discussed.

The results of the experiment summarized in section 4.6 are as follows:

1. Presence detects a high percentage of data path faults;

2. Presence detects about 60% of all C/I gate output faults;

3. Certain faults in the C/I ID and reset devices cannot be detected by any test vector;

4. Faults in the address decode logic of the Status Register will not be detected by test vectors that include a Status Register Read after each transaction;

5. Opportunities to reduce average fault latency and to improve overall performance are suggested by the results.

Most of the effort in this experiment was directed toward recapturing the C/I design and validating the C/I model derived from the recapture process. To effectively deal with VLSI scale designs, well-established CAD tools that are integrated into a design data base containing design verification test vectors will be required.

Faster simulators will be required to support the complexity of VLSI. However, significant work can be accomplished with existing simulators. For studies where simulation results are examined in detail, faster simulators would provide results more rapidly than could be assimilated.

The applicability of the gate level fault model used in this experiment is limited for modern devices such as gate arrays [8][9]. Since bipolar technology was employed for the current C/I implementation, the experiment results are more applicable than if FET based technology had been used. The main exception to this observation is the PAL devices used in the design.

Network size relative to the tools and techniques used impacted all phases of this effort. The complexity management problems for this modest-sized network not only lead to the validation problems discussed above, but impacted areas such as experiment run times, model configuration management, and computer storage requirements. Dealing with more complex designs without better tools would not be cost-effective.

The amount of computer time required to complete simulations of this scale or greater and the standards for the integrity of results derived for highly reliable systems combine to require that a methodology for using simulations include fault tolerance and error detection mechanisms. For example, the chances of a computer error affecting the results derived from an extremely long simulation run can become significant.

The microsequencer for the C/I was designed using a high-level design tool. The state transition table generated by this tool was highly redundant due to the manner in which certain address bits were allowed to reach the microsequencer register, PROM. While the design was functionally correct and was easy for the designer to create, it was much harder to test. This relatively minor example suggests that the new high-level design tools used to support cost-effective modern VLSI design could produce designs that are less testable and more expensive to validate. These tools are optimized for design functionality and for cost-effective designs. They are not optimized for testability.

The experiment provided information on the performance of the Diagnostic Emulator. The VAX FORTRAN version is a factor of 1.2×10^6 slower than a real time C/I. The QM-1 version is about a factor of 60×10^3 slower. For the C/I with its unusually low activity gates, the QM-1 version of the Diagnostic Emulation should compare favorably with higher performance parallel non-event-driven VAX-based simulators.

Natural extensions to the experiment described in this report are as follows:

1. to execute the Voter self tests that were not completed;

2. to modify diagnostic tests to confirm that Status Register address decoder faults can be detected;

3. to modify cross channel link models and confirm that certain faults become detectable;

4. to modify the DE fault model to include faults on gate inputs and to repeat the experiment;

5. to inject faults into other C/I channels and observe test vector performance;

6. to fault the PROM bits of the C/I microsequencer to determine performance against these faults;

7. to build a gate level model of the PAL device that more accurately models the failure modes in gate arrays, to run self tests, and to compare results to the less sophisticated fault model;

8. to inject some double faults to determine the probability of a double fault leading to system failure.

These possible extensions of the original experiment would benefit from the fact that a C/I model has been created and validated.

6. References

1. J. Lala, *Advanced Information Processing System (AIPS) System Specification,* The Charles Stark Draper Laboratories, Cambridge, MA (May 29, 1984).
2. G. Hughes, *FTP Parallel Data Communicator and Interstage,* The Charles Stark Draper Laboratories, Cambridge, MA (November 26, 1984).
3. G. Hughes, *Data Communicator Schematic Number FTP023, FTP024,* The Charles Stark Draper Laboratories, Cambridge, MA (May 29, 1985).
4. B. Becker, *Diagnostic Emulator User's Manual,* NASA-Langley Research Center CR178391 (December 1987).
5. G. Migneault, *The Diagnostic Emulation Technique in the AIRLAB,* NASA TM 4027, NASA-Langley Research Center (1988).
6. G. Troxel, *Communicator Self Test Software,* The Charles Stark Draper Laboratories, Cambridge, MA (September 17, 1985).
7. G. Troxel, *Fault Detection, Isolation and Reconfiguration Software,* The Charles Stark Draper Laboratories, Cambridge, MA (September 24, 1985).
8. J. Abraham and W. Fuchs, "Fault and Error Models for VLSI," *Proceedings of the IEEE* **74, no. 5**(May 1986).
9. F. Somenzi and S. Gai, "Fault Detection in Programmable Logic Arrays," *Proceedings of the IEEE* **74, no. 5**(May 1986).

APPENDIX A
Listing of the Diagnostic Tests

Presence Test Program

```
RSTR A A A A
RSTR B B B B
RSTR C C C C
RSTR D D D D
RSTR O O O O
WMSK M31 M31 M31 M31
VECB WXMR A A A A A0 D0 D0 D0
RDRR
VECE RSTR A A A A
VECB WXMR A A A A A1 D1 D1 D1
RDRR
VECE RSTR A A A A
VECB WXMR B B B B D1 B0 D1 D1
RDRR
VECE RSTR B B B B
VECB WXMR B B B B D0 B1 D0 D0
RDRR
VECE RSTR B B B B
VECB WXMR C C C C D0 D0 C0 D0
RDRR
VECE RSTR C C C C
VECB WXMR C C C C D1 D1 C1 D1
RDRR
VECE RSTR C C C C
VECB WXMR D D D D D1 D1 D1 D2
RDRR
VECE RSTR D D D D
VECB WXMR D D D D D0 D0 D0 D3
RDRR
VECE RSTR D D D D
VECB WXMR O O O O A0 A0 A0 A0
RDRR
VECE RSTR O O O O
VECB WXMR O O O O A1 A1 A1 A1
RDRR
VECE RSTR O O O O
```

Current Status Update Test Program

```
WMSK M15 M15 M15 M15
RSTR 0 0 0 0
VECB WXMR 0 0 0 0 A0 A1 A1 A1
WXMR 0 0 0 0 A1 A1 A1 A1
VECE RSTR 0 0 0 0
VECB WXMR 0 0 0 0 A1 A0 A0 A0
WXMR 0 0 0 0 A0 A1 A1 A1
VECE RSTR 0 0 0 0
VECB WXMR 0 0 0 0 B1 B0 B1 B1
WXMR 0 0 0 0 B1 B1 B1 B1
VECE RSTR 0 0 0 0
VECB WXMR 0 0 0 0 B0 B1 B0 B0
WXMR 0 0 0 0 B1 B0 B1 B1
VECE RSTR 0 0 0 0
VECB WXMR 0 0 0 0 C1 C1 C0 C1
WXMR 0 0 0 0 C1 C1 C1 C1
VECE RSTR 0 0 0 0
VECB WXMR 0 0 0 0 C0 C0 C1 C0
WXMR 0 0 0 0 C1 C1 C0 C1
VECE RSTR 0 0 0 0
VECB WXMR 0 0 0 0 D1 D1 D1 D0
WXMR 0 0 0 0 D1 D1 D1 D1
VECE RSTR 0 0 0 0
VECB WXMR 0 0 0 0 D0 D0 D0 D1
WXMR 0 0 0 0 D1 D1 D1 D0
VECE RSTR 0 0 0 0
VECB WXMR 0 0 0 0 A1 A1 C1 C1
WXMR 0 0 0 0 D1 D1 D1 D1
VECE RSTR 0 0 0 0
VECB WXMR 0 0 0 0 B0 B0 D0 D0
WXMR 0 0 0 0 C0 C0 A0 A0
VECE RSTR 0 0 0 0
```

Basic Test Program

```
RSTR A A A A
RSTR B B B B
RSTR C C C C
RSTR D D D D
RSTR O O O O
VECB WRCV D0 D0 D0 D0
VECE RDRR
VECB WRCV D2 D2 D2 D2
VECE RDRR
VECB WRCV D3 D3 D3 D3
VECE RDRR
WMSK M0 M0 M0 M0
VECB WXMR A A A A D2 D0 D0 D0
RDRR
VECE RSTR A A A A
VECB WXMR A A A A D3 D0 D0 D0
RDRR
VECE RSTR A A A A
VECB WXMR B B B B D0 D2 D0 D0
RDRR
VECE RSTR B B B B
VECB WXMR B B B B D0 D3 D0 D0
RDRR
VECE RSTR B B B B
VECB WXMR C C C C D0 D0 D2 D0
RDRR
VECE RSTR C C C C
VECB WXMR C C C C D0 D0 D3 D0
RDRR
VECE RSTR C C C C
VECB WXMR D D D D D0 D0 D0 D2
RDRR
VECE RSTR D D D D
VECB WXMR D D D D D0 D0 D0 D3
RDRR
VECE RSTR D D D D
VECB WXMR O O O O A0 B0 C0 D8
RDRR
RSTR O O O O
VECE RSTR O O O O
WMSK M1 M1 M1 M1
VECB WXMR O O O O A1 B1 C1 D9
RDRR
VECE RSTR O O O O
WMSK M2 M2 M2 M2
VECB WXMR O O O O A1 B1 C1 D9
RDRR
VECE RSTR O O O O
WMSK M4 M4 M4 M4
VECB WXMR O O O O A1 B1 C1 D9
RDRR
VECE RSTR O O O O
WMSK M8 M8 M8 M8
VECB WXMR O O O O A1 B1 C1 D9
RDRR
VECE RSTR O O O O
```

Mask Transform Quad Test Program

```
RSTR A A A A
RSTR B B B B
RSTR C C C C
RSTR D D D D
WMSK M31 M31 M31 M31
VECB WXMR A B B A A0 B0 D0 D0
RDRR
VECE RSTR A B B A
VECB WXMR D C C D D0 D0 C0 D8
RDRR
VECE RSTR D C C D
VECB WXMR A A C C A1 D0 C1 D0
RDRR
VECE RSTR A A C C
VECB WXMR A B A B A0 B1 D0 D0
RDRR
VECE RSTR A B A B
VECB WXMR B B D D D0 B1 D0 D9
RDRR
VECE RSTR B B D D
VECB WXMR C D C D D0 D0 C1 D9
RDRR
VECE RSTR C D C D
WMSK M15 M15 M15 M15
VECB WXMR A B B A B1 C1 D0 D0
RDRR
VECE RSTR A B B A
VECB WXMR D C C D D0 D0 D9 A1
RDRR
VECE RSTR D C C D
VECB WXMR A A C C B0 D0 D8 D0
RDRR
VECE RSTR A A C C
VECB WXMR A B A B B1 C0 D0 D0
RDRR
VECE RSTR A B A B
VECB WXMR B B D D D0 C0 D0 D8
RDRR
VECE RSTR B B D D
VECB WXMR C D C D D0 D0 D8 A1
RDRR
VECE RSTR C D C D
```

Mask Transform XAXB Test Program

```
RSTR A A A A
RSTR B B B B
RSTR C C C C
RSTR D D D D
WMSK M30 M30 M30 M30
VECB WXMR A B B A D2 B2 D0 D0
RDRR
VECE RSTR A B B A
VECB WXMR D C C D D0 D0 C2 D5
RDRR
VECE RSTR D C C D
VECB WXMR A A C C D3 D0 C3 D0
RDRR
VECE RSTR A A C C
VECB WXMR A B A B D2 B3 D0 D0
RDRR
VECE RSTR A B A B
VECB WXMR B B D D D0 B3 D0 D6
RDRR
VECE RSTR B B D D
VECB WXMR C D C D D0 D0 C3 D6
RDRR
VECE RSTR C D C D
WMSK M14 M14 M14 M14
VECB WXMR A B B A B3 C2 D0 D0
RDRR
VECE RSTR A B B A
VECB WXMR D C C D D0 D0 D6 D2
RDRR
VECE RSTR D C C D
VECB WXMR A A C C B2 D0 D5 D0
RDRR
VECE RSTR A A C C
VECB WXMR A B A B B3 C2 D0 D0
RDRR
VECE RSTR A B A B
VECB WXMR B B D D D0 C2 D0 D5
RDRR
VECE RSTR B B D D
VECB WXMR C D C D D0 D0 D5 D3
RDRR
VECE RSTR C D C D
WMSK M29 M29 M29 M29
VECB WXMR A B B A C2 D5 D0 D0
RDRR
VECE RSTR A B B A
VECB WXMR D C C D D0 D0 D2 B2
RDRR
VECE RSTR D C C D
VECB WXMR A A C C C3 D0 D6 D0
RDRR
VECE RSTR A A C C
VECB WXMR A B A B C2 B3 D0 D0
RDRR
VECE RSTR A B A B
VECB WXMR B B D D D0 D3 D0 C3
RDRR
VECE RSTR B B D D
VECB WXMR C D C D D0 D0 D5 C2
RDRR
VECE RSTR C D C D
WMSK M13 M13 M13 M13
VECB WXMR A B B A D0 D1 D0 D0
RDRR
VECE RSTR A B B A
VECB WXMR D C C D D0 D0 A0 B2
RDRR
VECE RSTR D C C D
VECB WXMR A A C C C2 D0 B1 D0
RDRR
VECE RSTR A A C C
VECB WXMR A B A B A1 C3 D0 D0
RDRR
VECE RSTR A B A B
VECB WXMR B B D D D0 D5 D0 B1
RDRR
VECE RSTR B B D D
VECB WXMR C D C D D0 D0 C3 B3
RDRR
VECE RSTR C D C D
```

Mask Transform XCXD Test Program

```
RSTR A A A A
RSTR B B B B
RSTR C C C C
RSTR D D D D
WMSK M27 M27 M27 M27
VECB WXMR A B B A D2 B2 D0 D0
RDRR
VECE RSTR A B B A
VECB WXMR D C C D D0 D0 C2 D5
RDRR
VECE RSTR D C C D
VECB WXMR A A C C D3 D0 C3 D0
RDRR
VECE RSTR A A C C
VECB WXMR A B A B D2 B3 D0 D0
RDRR
VECE RSTR A B A B
VECB WXMR B B D D D0 B3 D0 D6
RDRR
VECE RSTR B B D D
VECB WXMR C D C D D0 D0 C3 D6
RDRR
VECE RSTR C D C D
WMSK M11 M11 M11 M11
VECB WXMR A B B A B3 C2 D0 D0
RDRR
VECE RSTR A B B A
VECB WXMR D C C D D0 D0 D6 D2
RDRR
VECE RSTR D C C D
VECB WXMR A A C C B2 D0 D5 D0
RDRR
VECE RSTR A A C C
VECB WXMR A B A B B3 C2 D0 D0
RDRR
VECE RSTR A B A B
VECB WXMR B B D D D0 C2 D0 D5
RDRR
VECE RSTR B B D D
VECB WXMR C D C D D0 D0 D5 D3
RDRR
VECE RSTR C D C D
WMSK M23 M23 M23 M23
VECB WXMR A B B A C2 D5 D0 D0
RDRR
VECE RSTR A B B A
VECB WXMR D C C D D0 D0 D2 B2
RDRR
VECE RSTR D C C D
VECB WXMR A A C C C3 D0 D6 D0
RDRR
VECE RSTR A A C C
VECB WXMR A B A B C2 B3 D0 D0
RDRR
VECE RSTR A B A B
VECB WXMR B B D D D0 D3 D0 C3
RDRR
VECE RSTR B B D D
VECB WXMR C D C D D0 D0 D5 C2
RDRR
VECE RSTR C D C D
WMSK M7 M7 M7 M7
VECB WXMR A B B A D0 D1 D0 D0
RDRR
VECE RSTR A B B A
VECB WXMR D C C D D0 D0 A0 B2
RDRR
VECE RSTR D C C D
VECB WXMR A A C C C2 D0 B1 D0
RDRR
VECE RSTR A A C C
VECB WXMR A B A B A1 C3 D0 D0
RDRR
VECE RSTR A B A B
VECB WXMR B B D D D0 D5 D0 B1
RDRR
VECE RSTR B B D D
VECB WXMR C D C D D0 D0 C3 B3
RDRR
VECE RSTR C D C D
```

Voter Test Quad1 Program

```
WMSK M15 M15 M15 M15
RSTR 0 0 0 0
LOOP 0 1 1 0 3 1 0 3 1 0 3 1
VECB WXMR 0 0 0 0 T4 T4 T4 T4
RDRR
VECE RSTR 0 0 0 0
END
```

Voter Test Quad2 Program

```
WMSK M15 M15 M15 M15
RSTR 0 0 0 0
LOOP 2 3 1 0 3 1 0 3 1 0 3 1
VECB WXMR 0 0 0 0 T5 T5 T5 T5
RDRR
VECE RSTR 0 0 0 0
END
```

Voter Test B01 3XA Program

```
RSTR 0 0 0 0
WMSK M14 M14 M14 M14
LOOP 0 0 0 0 1 1 0 3 1 0 3 1
VECB WXMR 0 0 0 0 D0 T4 T4 T4
RDRR
VECE RSTR 0 0 0 0
END
LOOP 0 0 0 2 3 1 0 3 1 0 3 1
VECB WXMR 0 0 0 0 D0 T5 T5 T5
RDRR
VECE RSTR 0 0 0 0
END
```

Voter Test B01 3XB Program

```
RSTR 0 0 0 0
WMSK M13 M13 M13 M13
LOOP 0 1 1 0 0 0 0 3 1 0 3 1
VECB WXMR 0 0 0 0 T4 D0 T4 T4
RDRR
VECE RSTR 0 0 0 0
END
LOOP 2 3 1 0 0 0 0 3 1 0 3 1
VECB WXMR 0 0 0 0 T5 D0 T5 T5
RDRR
VECE RSTR 0 0 0 0
END
```

Voter Test B01 3XC Program

```
RSTR 0 0 0 0
WMSK M11 M11 M11 M11
LOOP 0 1 1 0 3 1 0 0 0 0 3 1
VECB WXMR 0 0 0 0 T4 T4 D0 T4
RDRR
VECE RSTR 0 0 0 0
END
LOOP 2 3 1 0 3 1 0 0 0 0 3 1
VECB WXMR 0 0 0 0 T5 T5 D0 T5
RDRR
VECE RSTR 0 0 0 0
END
```

Voter Test B01 3XD Program

```
RSTR 0 0 0 0
WMSK M7 M7 M7 M7
LOOP 0 1 1 0 3 1 0 3 1 0 0 0
VECB WXMR 0 0 0 0 T4 T4 T4 D0
RDRR
VECE RSTR 0 0 0 0
END
LOOP 2 3 1 0 3 1 0 3 1 0 0 0
VECB WXMR 0 0 0 0 T5 T5 T5 D0
RDRR
VECE RSTR 0 0 0 0
END
```

APPENDIX B
Diagnostic Emulation
Outputs for Non-faulted C/I

Presence Test

PRESENCE GOOD CIRCUIT

12

0 0 0 0

100

0 0 0 0

244

0 0 0 0

388

0 0 0 0

532

0 0 0 0

676

0 0 0 0

802

0 0 0 0

910

0 0 0 0

1414

15744 15744 15744 15744

1540

0 0 0 0

1666

0 0 0 0

2170

162033 162033 162033 162033

2296

0 0 0 0

2422

0 0 0 0

2926

143071 143071 143071 143071

3052

0 0 0 0

3178

0 0 0 0

3682

34706 34706 34706 34706

3808

0 0 0 0

3934

0 0 0 0

4438

130516 130516 130516 130516

4564

0 0 0 0

4690

0 0 0 0

5194

47271 47271 47271 47271

5320

0 0 0 0

5446

0 0 0 0

5950

125125 125125 125125 125125

6076

0 0 0 0

6202

0 0 0 0

6706

52652 52652 52652 52652

6832

0 0 0 0

6958

0 0 0 0

7462

15744 15744 15744 15744

7588

0 0 0 0

7714

0 0 0 0

8218

162033 162033 162033 162033

8344

0 0 0 0

Current Status Update Test

CSTUPD	GOOD	CIRCUIT					
12				0	0	0	0
0	0	0	0	5968			
82				4	4	4	4
0	0	0	0	6094			
208				0	0	0	0
0	0	0	0	6598			
334				0	0	0	0
0	0	0	0	7120			
838				4	4	4	4
0	0	0	0	7246			
1360				0	0	0	0
1	1	1	1	7750			
1486				0	0	0	0
0	0	0	0	8272			
1990				10	10	10	10
0	0	0	0	8398			
2512				0	0	0	0
1	1	1	1	8902			
2638				0	0	0	0
0	0	0	0	9424			
3142				10	10	10	10
0	0	0	0	9550			
3664				0	0	0	0
2	2	2	2	10054			
3790				0	0	0	0
0	0	0	0	10576			
4294				37	37	37	37
0	0	0	0	10702			
4816				0	0	0	0
2	2	2	2	11206			
4942				0	0	0	0
0	0	0	0	11728			
5446				37	37	37	37

BASIC GOOD CIRCUIT

12
0 0 0 0
100
0 0 0 0
244
0 0 0 0
388
0 0 0 0
532
0 0 0 0
676
0 0 0 0
802
0 0 0 0
910
0 0 0 0
1018
0 0 0 0
1126
125125 125125 125125 125125
1234
0 0 0 0
1342
52652 52652 52652 52652
1450
0 0 0 0
1558
0 0 0 0
2062
125125 125125 125125 125125
2188
0 0 0 0
2314
0 0 0 0
2818
52652 52652 52652 52652
2944
0 0 0 0
3070
0 0 0 0
3574
125125 125125 125125 125125
3700
0 0 0 0
3826
0 0 0 0
4330
52652 52652 52652 52652
4456
0 0 0 0
4582
0 0 0 0
5086
125125 125125 125125 125125
5212
0 0 0 0
5338
0 0 0 0
5842
52652 52652 52652 52652
5968
0 0 0 0
6094
0 0 0 0
6598
125125 125125 125125 125125
6724
0 0 0 0
6850
0 0 0 0
7354
52652 52652 52652 52652
7480
0 0 0 0
7606
0 0 0 0
8110
15744 143071 130516 66223
8236
16 15 13 7
8380
0 0 0 0
8506
0 0 0 0
8614
0 0 0 0
9118
162033 162033 162033 162033
9244
16 16 16 16
9370
0 0 0 0
9478
0 0 0 0
9982
34706 34706 34706 34706
10108
15 15 15 15
10234
0 0 0 0
10342
0 0 0 0
10846
47271 47271 47271 47271
10972
13 13 13 13
11098
0 0 0 0
11206
0 0 0 0
11710
111554 111554 111554 111554
11836
7 7 7 7

Mask Transform Quad Test

MSKTRNQ GOOD CIRCUIT

12
0 0 0 0
100
0 0 0 0
244
0 0 0 0
388
0 0 0 0
532
0 0 0 0
658
0 0 0 0
766
0 0 0 0
1270
143071 15744 15744 143071
1396
11 6 6 11
1522
0 0 0 0
2026
130516 66223 66223 130516
2152
11 6 6 11
2278
0 0 0 0
2782
47271 47271 162033 162033
2908
3 3 14 14
3034
0 0 0 0
3538
34706 15744 34706 15744
3664
5 12 5 12
3790
0 0 0 0
4294
111554 111554 34706 34706
4420
3 3 14 14
4546
0 0 0 0
5050
111554 47271 111554 47271
5176
5 12 5 12
5302
0 0 0 0
5410
0 0 0 0
5914
77777 77777 77777 77777
6040
37 37 37 37
6166
0 0 0 0
6670
173577 173577 173577 173577
6796
37 37 37 37
6922
0 0 0 0
7426
167273 167273 167273 167273
7552
37 37 37 37
7678
0 0 0 0
8182
134716 134716 134716 134716
8308
37 37 37 37
8434
0 0 0 0
8938
176737 176737 176737 176737
9064
37 37 37 37
9190
0 0 0 0
9694
166233 166233 166233 166233
9820
37 37 37 37

Mask Transform XAXB Test

MSKTRNXAXB GOOD CIRCUIT

12
0 0 0 0
100
0 0 0 0
244
0 0 0 0
388
0 0 0 0
532
0 0 0 0
658
0 0 0 0
766
0 0 0 0
1270
146063 146063 146063 146063
1396
11 11 11 11
1522
0 0 0 0
2026
34307 34307 34307 34307
2152
11 11 11 11
2278
0 0 0 0
2782
143470 143470 143470 143470
2908
3 3 3 3
3034
0 0 0 0
3538
31714 31714 31714 31714
3664
5 5 5 5
3790
0 0 0 0
4294
107760 107760 107760 107760
4420
3 3 3 3
4546
0 0 0 0
5050
107760 107760 107760 107760
5176
5 5 5 5
5302
0 0 0 0
5410
0 0 0 0
5914
34307 34307 34307 34307
6040
11 11 11 11
6166
0 0 0 0
6670
107760 107760 107760 107760
6796
11 11 11 11
6922
0 0 0 0
7426
70017 70017 70017 70017
7552
3 3 3 3
7678
0 0 0 0
8182
34307 34307 34307 34307
8308
5 5 5 5
8434
0 0 0 0
8938
70017 70017 70017 70017
9064
3 3 3 3
9190
0 0 0 0
9694
52652 52652 52652 52652
9820
5 5 5 5
9946
0 0 0 0
10054
0 0 0 0
10558
34307 34307 34307 34307
10684
6 6 6 6
10810
0 0 0 0
11314
146063 146063 146063 146063
11440
6 6 6 6
11566
0 0 0 0
12070
107760 107760 107760 107760
12196
3 3 3 3
12322
0 0 0 0
12826
34307 34307 34307 34307
12952
12 12 12 12
13078
0 0 0 0
13582
143470 143470 143470 143470
13708

```
      3       3       3       3
  13834
      0       0       0       0
  14338
  70017   70017   70017   70017
  14464
     12      12      12      12
  14590
      0       0       0       0
  14698
      0       0       0       0
  15202
      0       0       0       0
  15328
      6       6       6       6
  15454
      0       0       0       0
  15958
 146063  146063  146063  146063
  16084
      6       6       6       6
  16210
      0       0       0       0
  16714
  34706   34706   34706   34706
  16840
      3       3       3       3
  16966
      0       0       0       0
  17470
 162033  162033  162033  162033
  17596
     12      12      12      12
  17722
      0       0       0       0
  18226
  34706   34706   34706   34706
  18352
      3       3       3       3
  18478
      0       0       0       0
  18982
 143470  143470  143470  143470
  19108
     12      12      12      12
```

C-2

Mask Transform XCXD Test

MSKTRNXCXD GOOD CIRCUIT

```
          12
     0         0         0         0
         100
     0         0         0         0
         244
     0         0         0         0
         388
     0         0         0         0
         532
     0         0         0         0
         658
     0         0         0         0
         766
     0         0         0         0
        1270
125125    125125    125125    125125
        1396
     6         6         6         6
        1522
     0         0         0         0
        2026
 70017     70017     70017     70017
        2152
     6         6         6         6
        2278
     0         0         0         0
        2782
 52652     52652     52652     52652
        2908
    14        14        14        14
        3034
     0         0         0         0
        3538
 31714     31714     31714     31714
        3664
     5         5         5         5
        3790
     0         0         0         0
        4294
 31714     31714     31714     31714
        4420
    14        14        14        14
        4546
     0         0         0         0
        5050
107760    107760    107760    107760
        5176
     5         5         5         5
        5302
     0         0         0         0
        5410
     0         0         0         0
        5914
 31714     31714     31714     31714
        6040
     6         6         6         6
        6166
     0         0         0         0
        6670
125125    125125    125125    125125
        6796
     6         6         6         6
        6922
     0         0         0         0
        7426
146063    146063    146063    146063
        7552
    14        14        14        14
        7678
     0         0         0         0
        8182
 34307     34307     34307     34307
        8308
     5         5         5         5
        8434
     0         0         0         0
        8938
 34307     34307     34307     34307
        9064
    14        14        14        14
        9190
     0         0         0         0
        9694
 52652     52652     52652     52652
        9820
     5         5         5         5
        9946
     0         0         0         0
       10054
     0         0         0         0
       10558
 70017     70017     70017     70017
       10684
    11        11        11        11
       10810
     0         0         0         0
       11314
125125    125125    125125    125125
       11440
    11        11        11        11
       11566
     0         0         0         0
       12070
143470    143470    143470    143470
       12196
    14        14        14        14
       12322
     0         0         0         0
       12826
 34307     34307     34307     34307
       12952
    12        12        12        12
       13078
     0         0         0         0
       13582
 52652     52652     52652     52652
       13708
```

```
    14      14      14      14
      13834
     0       0       0       0
      14338
 70017   70017   70017   70017
      14464
    12      12      12      12
      14590
     0       0       0       0
      14698
     0       0       0       0
      15202
177777  177777  177777  177777
      15328
    11      11      11      11
      15454
     0       0       0       0
      15958
 15744   15744   15744   15744
      16084
    11      11      11      11
      16210
     0       0       0       0
      16714
 34307   34307   34307   34307
      16840
    14      14      14      14
      16966
     0       0       0       0
      17470
```

```
162033  162033  162033  162033
      17596
    12      12      12      12
      17722
     0       0       0       0
      18226
 70017   70017   70017   70017
      18352
    14      14      14      14
      18478
     0       0       0       0
      18982
143470  143470  143470  143470
      19108
    12      12      12      12
```

Voter Test Quad1

```
VB01QUAD1        GOOD CIRCUIT
          12
      0       0       0       0
          82
      0       0       0       0
         208
      0       0       0       0
         334
      0       0       0       0
         838
 100000  100000  100000  100000
         964
      0       0       0       0
        1090
      0       0       0       0
        1594
 100000  100000  100000  100000
        1720
     10      10      10      10
        1846
      0       0       0       0
        2350
 100000  100000  100000  100000
        2476
     10      10      10      10
        2602
      0       0       0       0
        3106
 100000  100000  100000  100000
        3232
     10      10      10      10
        3358
      0       0       0       0
        3862
 100000  100000  100000  100000
        3988
      4       4       4       4
        4114
      0       0       0       0
        4618
 100001  100001  100001  100001
        4744
     23      23      23      23
        4870
      0       0       0       0
        5374
 100000  100000  100000  100000
        5500
     14      14      14      14
        5626
      0       0       0       0
        6130
 100001  100001  100001  100001
        6256
     33      33      33      33
        6382
      0       0       0       0
        6886
 100000  100000  100000  100000
        7012
      4       4       4       4
        7138
      0       0       0       0
        7642
 100000  100000  100000  100000
        7768
     14      14      14      14
        7894
      0       0       0       0
        8398
 100002  100002  100002  100002
        8524
     23      23      23      23
        8650
      0       0       0       0
        9154
 100002  100002  100002  100002
        9280
     33      33      33      33
        9406
      0       0       0       0
        9910
 100000  100000  100000  100000
       10036
      4       4       4       4
       10162
      0       0       0       0
       10666
 100001  100001  100001  100001
       10792
     27      27      27      27
       10918
      0       0       0       0
       11422
 100002  100002  100002  100002
       11548
     27      27      27      27
       11674
      0       0       0       0
       12178
 100003  100003  100003  100003
       12304
     23      23      23      23
       12430
      0       0       0       0
       12934
 100000  100000  100000  100000
       13060
      2       2       2       2
       13186
      0       0       0       0
       13690
 100001  100001  100001  100001
       13816
     25      25      25      25
       13942
      0       0       0       0
       14446
```

```
100000 100000 100000 100000
  14572
    12     12     12     12
  14698
     0      0      0      0
  15202
100001 100001 100001 100001
  15328
    35     35     35     35
  15454
     0      0      0      0
  15958
100001 100001 100001 100001
  16084
    31     31     31     31
  16210
     0      0      0      0
  16714
100001 100001 100001 100001
  16840
     1      1      1      1
  16966
     0      0      0      0
  17470
100001 100001 100001 100001
  17596
    31     31     31     31
  17722
     0      0      0      0
  18226
```

```
100001 100001 100001 100001
  18352
    11     11     11     11
  18478
     0      0      0      0
  18982
100000 100000 100000 100000
  19108
     6      6      6      6
  19234
     0      0      0      0
  19738
100001 100001 100001 100001
  19864
    25     25     25     25
  19990
     0      0      0      0
  20494
100002 100002 100002 100002
  20620
    23     23     23     23
  20746
     0      0      0      0
  21250
100003 100003 100003 100003
  21376
    27     27     27     27
  21502
     0      0      0      0
  22006
```

```
100001 100001 100001 100001
  22132
35     35     35     35
  22258
 0      0      0      0
  22762
100001 100001 100001 100001
  22888
 5      5      5      5
  23014
 0      0      0      0
  23518
100003 100003 100003 100003
  23644
33     33     33     33
  23770
 0      0      0      0
  24274
100003 100003 100003 100003
  24400
23     23     23     23
  24526
 0      0      0      0
  25030
100000 100000 100000 100000
  25156
 2      2      2      2
  25282
 0      0      0      0
  25786
100000 100000 100000 100000
  25912
12     12     12     12
  26038
 0      0      0      0
  26542
100002 100002 100002 100002
  26668
25     25     25     25
  26794
 0      0      0      0
  27298
100002 100002 100002 100002
  27424
35     35     35     35
  27550
 0      0      0      0
  28054
100000 100000 100000 100000
  28180
 6      6      6      6
  28306
 0      0      0      0
  28810
100001 100001 100001 100001
  28936
23     23     23     23
  29062
 0      0      0      0
  29566
100002 100002 100002 100002
  29692
25     25     25     25
  29818
 0      0      0      0
  30322
100003 100003 100003 100003
  30448
27     27     27     27
  30574
 0      0      0      0
  31078
100002 100002 100002 100002
  31204
31     31     31     31
  31330
 0      0      0      0
  31834
100002 100002 100002 100002
  31960
31     31     31     31
  32086
 0      0      0      0
  32590
100002 100002 100002 100002
  32716
 1      1      1      1
  32842
 0      0      0      0
  33346
100002 100002 100002 100002
  33472
11     11     11     11
  33598
 0      0      0      0
  34102
100002 100002 100002 100002
  34228
35     35     35     35
  34354
 0      0      0      0
  34858
100003 100003 100003 100003
  34984
33     33     33     33
  35110
 0      0      0      0
  35614
100002 100002 100002 100002
  35740
 5      5      5      5
  35866
 0      0      0      0
  36370
100003 100003 100003 100003
  36496
23     23     23     23
  36622
 0      0      0      0
  37126
```

```
100000 100000 100000 100000
      37252
     2       2       2       2
      37378
     0       0       0       0
      37882
100001 100001 100001 100001
      38008
    27      27      27      27
      38134
     0       0       0       0
      38638
100002 100002 100002 100002
      38764
    27      27      27      27
      38890
     0       0       0       0
      39394
100003 100003 100003 100003
      39520
    25      25      25      25
      39646
     0       0       0       0
      40150
100001 100001 100001 100001
      40276
    33      33      33      33
      40402
     0       0       0       0
      40906
100001 100001 100001 100001
      41032
     3       3       3       3
      41158
     0       0       0       0
      41662
100003 100003 100003 100003
      41788
    35      35      35      35
      41914
     0       0       0       0
      42418
100003 100003 100003 100003
      42544
    25      25      25      25
      42670
     0       0       0       0
      43174
100002 100002 100002 100002
      43300
    33      33      33      33
      43426
     0       0       0       0
      43930
100003 100003 100003 100003
      44056
    35      35      35      35
      44182
     0       0       0       0
      44686
100002 100002 100002 100002
      44812
     3       3       3       3
      44938
     0       0       0       0
      45442
100003 100003 100003 100003
      45568
    25      25      25      25
      45694
     0       0       0       0
      46198
100003 100003 100003 100003
      46324
    31      31      31      31
      46450
     0       0       0       0
      46954
100003 100003 100003 100003
      47080
    31      31      31      31
      47206
     0       0       0       0
      47710
100003 100003 100003 100003
      47836
    31      31      31      31
      47962
     0       0       0       0
      48466
100003 100003 100003 100003
      48592
     1       1       1       1
      48718
     0       0       0       0
      49222
100000 100000 100000 100000
      49348
     1       1       1       1
      49474
     0       0       0       0
      49978
100001 100001 100001 100001
      50104
    26      26      26      26
      50230
     0       0       0       0
      50734
100000 100000 100000 100000
      50860
    11      11      11      11
      50986
     0       0       0       0
      51490
100001 100001 100001 100001
      51616
    36      36      36      36
      51742
     0       0       0       0
      52246
```

100001 100001 100001 100001
52372
32 32 32 32
52498
0 0 0 0
53002
100001 100001 100001 100001
53128
2 2 2 2
53254
0 0 0 0
53758
100001 100001 100001 100001
53884
32 32 32 32
54010
0 0 0 0
54514
100001 100001 100001 100001
54640
12 12 12 12
54766
0 0 0 0
55270
100000 100000 100000 100000
55396
5 5 5 5
55522
0 0 0 0
56026
100001 100001 100001 100001
56152
26 26 26 26
56278
0 0 0 0
56782
100002 100002 100002 100002
56908
23 23 23 23
57034
0 0 0 0
57538
100003 100003 100003 100003
57664
27 27 27 27
57790
0 0 0 0
58294
100001 100001 100001 100001
58420
36 36 36 36
58546
0 0 0 0
59050
100001 100001 100001 100001
59176
6 6 6 6
59302
0 0 0 0
59806

100003 100003 100003 100003
59932
33 33 33 33
60058
0 0 0 0
60562
100003 100003 100003 100003
60688
23 23 23 23
60814
0 0 0 0
61318
100001 100001 100001 100001
61444
34 34 34 34
61570
0 0 0 0
62074
100001 100001 100001 100001
62200
4 4 4 4
62326
0 0 0 0
62830
100001 100001 100001 100001
62956
34 34 34 34
63082
0 0 0 0
63586
100001 100001 100001 100001
63712
14 14 14 14
63838
0 0 0 0
64342
100001 100001 100001 100001
64468
10 10 10 10
64594
0 0 0 0
65098
100001 100001 100001 100001
65224
0 0 0 0
65350
0 0 0 0
65854
100001 100001 100001 100001
65980
10 10 10 10
66106
0 0 0 0
66610
100001 100001 100001 100001
66736
10 10 10 10
66862
0 0 0 0
67366

```
100001 100001 100001 100001
       67492
    34     34     34     34
       67618
     0      0      0      0
       68122
100001 100001 100001 100001
       68248
     4      4      4      4
       68374
     0      0      0      0
       68878
100003 100003 100003 100003
       69004
    37     37     37     37
       69130
     0      0      0      0
       69634
100003 100003 100003 100003
       69760
    27     27     27     27
       69886
     0      0      0      0
       70390
100001 100001 100001 100001
       70516
    14     14     14     14
       70642
     0      0      0      0
       71146
100001 100001 100001 100001
       71272
     4      4      4      4
       71398
     0      0      0      0
       71902
100003 100003 100003 100003
       72028
    33     33     33     33
       72154
     0      0      0      0
       72658
100003 100003 100003 100003
       72784
    23     23     23     23
       72910
     0      0      0      0
       73414
100000 100000 100000 100000
       73540
     3      3      3      3
       73666
     0      0      0      0
       74170
100001 100001 100001 100001
       74296
    26     26     26     26
       74422
     0      0      0      0
       74926
```

```
100002 100002 100002 100002
       75052
    25     25     25     25
       75178
     0      0      0      0
       75682
100003 100003 100003 100003
       75808
    27     27     27     27
       75934
     0      0      0      0
       76438
100001 100001 100001 100001
       76564
    32     32     32     32
       76690
     0      0      0      0
       77194
100001 100001 100001 100001
       77320
     2      2      2      2
       77446
     0      0      0      0
       77950
100003 100003 100003 100003
       78076
    37     37     37     37
       78202
     0      0      0      0
       78706
100003 100003 100003 100003
       78832
    27     27     27     27
       78958
     0      0      0      0
       79462
100002 100002 100002 100002
       79588
    31     31     31     31
       79714
     0      0      0      0
       80218
100003 100003 100003 100003
       80344
    37     37     37     37
       80470
     0      0      0      0
       80974
100002 100002 100002 100002
       81100
     1      1      1      1
       81226
     0      0      0      0
       81730
100003 100003 100003 100003
       81856
    27     27     27     27
       81982
     0      0      0      0
       82486
```

100003 100003 100003 100003
82612
33 33 33 33
82738
0 0 0 0
83242
100003 100003 100003 100003
83368
33 33 33 33
83494
0 0 0 0
83998
100003 100003 100003 100003
84124
33 33 33 33
84250
0 0 0 0
84754
100003 100003 100003 100003
84880
3 3 3 3
85006
0 0 0 0
85510
100001 100001 100001 100001
85636
36 36 36 36
85762
0 0 0 0
86266
100001 100001 100001 100001
86392
6 6 6 6
86518
0 0 0 0
87022
100003 100003 100003 100003
87148
35 35 35 35
87274
0 0 0 0
87778
100003 100003 100003 100003
87904
25 25 25 25
88030
0 0 0 0
88534
100001 100001 100001 100001
88660
12 12 12 12
88786
0 0 0 0
89290
100001 100001 100001 100001
89416
2 2 2 2
89542
0 0 0 0
90046
100003 100003 100003 100003
90172
35 35 35 35
90298
0 0 0 0
90802
100003 100003 100003 100003
90928
25 25 25 25
91054
0 0 0 0
91558
100003 100003 100003 100003
91684
35 35 35 35
91810
0 0 0 0
92314
100003 100003 100003 100003
92440
35 35 35 35
92566
0 0 0 0
93070
100003 100003 100003 100003
93196
35 35 35 35
93322
0 0 0 0
93826
100003 100003 100003 100003
93952
5 5 5 5
94078
0 0 0 0
94582
100003 100003 100003 100003
94708
31 31 31 31
94834
0 0 0 0
95338
100003 100003 100003 100003
95464
31 31 31 31
95590
0 0 0 0
96094
100003 100003 100003 100003
96220
11 11 11 11
96346
0 0 0 0
96850
100003 100003 100003 100003
96976
1 1 1 1

Voter Test Quad2

VB01QUAD2 GOOD CIRCUIT

12			
0	0	0	0
82			
0	0	0	0
208			
0	0	0	0
334			
0	0	0	0
838			
25	25	25	25
964			
1	1	1	1
1090			
0	0	0	0
1594			
25	25	25	25
1720			
11	11	11	11
1346			
0	0	0	0
2350			
1025	1025	1025	1025
2476			
26	26	26	26
2602			
0	0	0	0
3106			
1025	1025	1025	1025
3232			
36	36	36	36
3358			
0	0	0	0
3862			
25	25	25	25
3988			
5	5	5	5
4114			
0	0	0	0
4618			
425	425	425	425
4744			
23	23	23	23
4870			
0	0	0	0
5374			
1025	1025	1025	1025
5500			
26	26	26	26
5626			
0	0	0	0
6130			
1425	1425	1425	1425
6256			
27	27	27	27
6382			
0	0	0	0
6886			
1025	1025	1025	1025
7012			
32	32	32	32
7138			
0	0	0	0
7642			
1025	1025	1025	1025
7768			
32	32	32	32
7894			
0	0	0	0
8398			
1025	1025	1025	1025
8524			
2	2	2	2
8650			
0	0	0	0
9154			
1025	1025	1025	1025
9280			
12	12	12	12
9406			
0	0	0	0
9910			
1025	1025	1025	1025
10036			
36	36	36	36
10162			
0	0	0	0
10666			
1425	1425	1425	1425
10792			
33	33	33	33
10918			
0	0	0	0
11422			
1025	1025	1025	1025
11548			
6	6	6	6
11674			
0	0	0	0
12178			
1425	1425	1425	1425
12304			
23	23	23	23
12430			
0	0	0	0
12934			
25	25	25	25
13060			
3	3	3	3
13186			
0	0	0	0
13690			
425	425	425	425
13816			
25	25	25	25
13942			
0	0	0	0
14446			

1025 1025 1025 1025
14572
26 26 26 26
14698
0 0 0 0
15202
1425 1425 1425 1425
15328
27 27 27 27
15454
0 0 0 0
15958
425 425 425 425
16084
31 31 31 31
16210
0 0 0 0
16714
425 425 425 425
16840
1 1 1 1
16966
0 0 0 0
17470
1425 1425 1425 1425
17596
37 37 37 37
17722
0 0 0 0
18226

1425 1425 1425 1425
18352
27 27 27 27
18478
0 0 0 0
18982
1025 1025 1025 1025
19108
32 32 32 32
19234
0 0 0 0
19738
1425 1425 1425 1425
19864
37 37 37 37
19990
0 0 0 0
20494
1025 1025 1025 1025
20620
2 2 2 2
20746
0 0 0 0
21250
1425 1425 1425 1425
21376
27 27 27 27
21502
0 0 0 0
22006

```
1425    1425    1425    1425
    22132
  33      33      33      33
    22258
   0       0       0       0
    22762
1425    1425    1425    1425
    22888
  33      33      33      33
    23014
   0       0       0       0
    23518
1425    1425    1425    1425
    23644
  33      33      33      33
    23770
   0       0       0       0
    24274
1425    1425    1425    1425
    24400
   3       3       3       3
    24526
   0       0       0       0
    25030
1025    1025    1025    1025
    25156
  34      34      34      34
    25282
   0       0       0       0
    25786
1025    1025    1025    1025
    25912
  34      34      34      34
    26038
   0       0       0       0
    26542
1025    1025    1025    1025
    26668
   4       4       4       4
    26794
   0       0       0       0
    27298
1025    1025    1025    1025
    27424
  14      14      14      14
    27550
   0       0       0       0
    28054
1025    1025    1025    1025
    28180
  34      34      34      34
    28306
   0       0       0       0
    28810
1425    1425    1425    1425
    28936
  37      37      37      37
    29062
   0       0       0       0
    29566
1025    1025    1025    1025
    29692
   4       4       4       4
    29818
   0       0       0       0
    30322
1425    1425    1425    1425
    30448
  27      27      27      27
    30574
   0       0       0       0
    31078
1025    1025    1025    1025
    31204
  10      10      10      10
    31330
   0       0       0       0
    31834
1025    1025    1025    1025
    31960
  10      10      10      10
    32086
   0       0       0       0
    32590
1025    1025    1025    1025
    32716
   0       0       0       0
    32842
   0       0       0       0
    33346
1025    1025    1025    1025
    33472
  10      10      10      10
    33598
   0       0       0       0
    34102
1025    1025    1025    1025
    34228
  14      14      14      14
    34354
   0       0       0       0
    34858
1425    1425    1425    1425
    34984
  33      33      33      33
    35110
   0       0       0       0
    35614
1025    1025    1025    1025
    35740
   4       4       4       4
    35866
   0       0       0       0
    36370
1425    1425    1425    1425
    36496
  23      23      23      23
    36622
   0       0       0       0
    37126
```

1025 1025 1025 1025
37252
36 36 36 36
37378
0 0 0 0
37882
1425 1425 1425 1425
38008
35 35 35 35
38134
0 0 0 0
38638
1025 1025 1025 1025
38764
6 6 6 6
38890
0 0 0 0
39394
1425 1425 1425 1425
39520
25 25 25 25
39646
0 0 0 0
40150
1425 1425 1425 1425
40276
35 35 35 35
40402
0 0 0 0
40906
1425 1425 1425 1425
41032
35 35 35 35
41158
0 0 0 0
41662
1425 1425 1425 1425
41788
35 35 35 35
41914
0 0 0 0
42418
1425 1425 1425 1425
42544
5 5 5 5
42670
0 0 0 0
43174
1025 1025 1025 1025
43300
12 12 12 12
43426
0 0 0 0
43930
1425 1425 1425 1425
44056
35 35 35 35
44182
0 0 0 0
44686
1025 1025 1025 1025
44812
2 2 2 2
44938
0 0 0 0
45442
1425 1425 1425 1425
45568
25 25 25 25
45694
0 0 0 0
46198
1425 1425 1425 1425
46324
31 31 31 31
46450
0 0 0 0
46954
1425 1425 1425 1425
47080
11 11 11 11
47206
0 0 0 0
47710
1425 1425 1425 1425
47836
31 31 31 31
47962
0 0 0 0
48466
1425 1425 1425 1425
48592
1 1 1 1
48718
0 0 0 0
49222
25 25 25 25
49348
1 1 1 1
49474
0 0 0 0
49978
425 425 425 425
50104
27 27 27 27
50230
0 0 0 0
50734
1025 1025 1025 1025
50860
27 27 27 27
50986
0 0 0 0
51490
1425 1425 1425 1425
51616
26 26 26 26
51742
0 0 0 0
52246

425 425 425 425
52372
33 33 33 33
52498
0 0 0 0
53002
425 425 425 425
53128
3 3 3 3
53254
0 0 0 0
53758
1425 1425 1425 1425
53884
36 36 36 36
54010
0 0 0 0
54514
1425 1425 1425 1425
54640
26 26 26 26
54766
0 0 0 0
55270
1025 1025 1025 1025
55396
33 33 33 33
55522
0 0 0 0
56026
1425 1425 1425 1425
56152
36 36 36 36
56278
0 0 0 0
56782
1025 1025 1025 1025
56908
3 3 3 3
57034
0 0 0 0
57538
1425 1425 1425 1425
57664
26 26 26 26
57790
0 0 0 0
58294
1425 1425 1425 1425
58420
32 32 32 32
58546
0 0 0 0
59050
1425 1425 1425 1425
59176
32 32 32 32
59302
0 0 0 0
59806
1425 1425 1425 1425
59932
32 32 32 32
60058
0 0 0 0
60562
1425 1425 1425 1425
60688
2 2 2 2
60814
0 0 0 0
61318
425 425 425 425
61444
35 35 35 35
61570
0 0 0 0
62074
425 425 425 425
62200
5 5 5 5
62326
0 0 0 0
62830
1425 1425 1425 1425
62956
36 36 36 36
63082
0 0 0 0
63586
1425 1425 1425 1425
63712
26 26 26 26
63838
0 0 0 0
64342
425 425 425 425
64468
11 11 11 11
64594
0 0 0 0
65098
425 425 425 425
65224
1 1 1 1
65350
0 0 0 0
65854
1425 1425 1425 1425
65980
36 36 36 36
66106
0 0 0 0
66610
1425 1425 1425 1425
66736
26 26 26 26
66862
0 0 0 0
67366

1425 1425 1425 1425
67492
36 36 36 36
67618
0 0 0 0
68122
1425 1425 1425 1425
68248
36 36 36 36
68374
0 0 0 0
68878
1425 1425 1425 1425
69004
36 36 36 36
69130
0 0 0 0
69634
1425 1425 1425 1425
69760
6 6 6 6
69886
0 0 0 0
70390
1425 1425 1425 1425
70516
32 32 32 32
70642
0 0 0 0
71146
1425 1425 1425 1425
71272
32 32 32 32
71398
0 0 0 0
71902
1425 1425 1425 1425
72028
12 12 12 12
72154
0 0 0 0
72658
1425 1425 1425 1425
72784
2 2 2 2
72910
0 0 0 0
73414
1025 1025 1025 1025
73540
35 35 35 35
73666
0 0 0 0
74170
1425 1425 1425 1425
74296
36 36 36 36
74422
0 0 0 0
74926
1025 1025 1025 1025
75052
5 5 5 5
75178
0 0 0 0
75682
1425 1425 1425 1425
75808
26 26 26 26
75934
0 0 0 0
76438
1425 1425 1425 1425
76564
36 36 36 36
76690
0 0 0 0
77194
1425 1425 1425 1425
77320
36 36 36 36
77446
0 0 0 0
77950
1425 1425 1425 1425
78076
36 36 36 36
78202
0 0 0 0
78706
1425 1425 1425 1425
78832
6 6 6 6
78958
0 0 0 0
79462
1025 1025 1025 1025
79588
11 11 11 11
79714
0 0 0 0
80218
1425 1425 1425 1425
80344
36 36 36 36
80470
0 0 0 0
80974
1025 1025 1025 1025
81100
1 1 1 1
81226
0 0 0 0
81730
1425 1425 1425 1425
81856
26 26 26 26
81982
0 0 0 0
82486

1425 1425 1425 1425
82612
32 32 32 32
82738
0 0 0 0
83242
1425 1425 1425 1425
83368
12 12 12 12
83494
0 0 0 0
83998
1425 1425 1425 1425
84124
32 32 32 32
84250
0 0 0 0
84754
1425 1425 1425 1425
84880
2 2 2 2
85006
0 0 0 0
85510
1425 1425 1425 1425
85636
34 34 34 34
85762
0 0 0 0
86266
1425 1425 1425 1425
86392
34 34 34 34
86518
0 0 0 0
87022
1425 1425 1425 1425
87148
34 34 34 34
87274
0 0 0 0
87778
1425 1425 1425 1425
87904
4 4 4 4
88030
0 0 0 0
88534
1425 1425 1425 1425
88660
34 34 34 34
88786
0 0 0 0
89290
1425 1425 1425 1425
89416
34 34 34 34
89542
0 0 0 0
90046
1425 1425 1425 1425
90172
14 14 14 14
90298
0 0 0 0
90802
1425 1425 1425 1425
90928
4 4 4 4
91054
0 0 0 0
91558
1425 1425 1425 1425
91684
34 34 34 34
91810
0 0 0 0
92314
1425 1425 1425 1425
92440
14 14 14 14
92566
0 0 0 0
93070
1425 1425 1425 1425
93196
34 34 34 34
93322
0 0 0 0
93826
1425 1425 1425 1425
93952
4 4 4 4
94078
0 0 0 0
94582
1425 1425 1425 1425
94708
10 10 10 10
94834
0 0 0 0
95338
1425 1425 1425 1425
95464
10 10 10 10
95590
0 0 0 0
96094
1425 1425 1425 1425
96220
10 10 10 10
96346
0 0 0 0
96850
1425 1425 1425 1425
96976
0 0 0 0

Voter Test B01 3XA

```
VB013XA      GOOD CIRCUIT
          12
      0        0       0       0
         100
      0        0       0       0
         226
      0        0       0       0
         334
      0        0       0       0
         838
 100000   100000  100000  100000
         964
      1        1       1       1
        1090
      0        0       0       0
        1594
 100000   100000  100000  100000
        1720
     11       11      11      11
        1846
      0        0       0       0
        2350
 100000   100000  100000  100000
        2476
     11       11      11      11
        2602
      0        0       0       0
        3106
 100000   100000  100000  100000
        3232
     11       11      11      11
        3358
      0        0       0       0
        3862
 100000   100000  100000  100000
        3988
      5        5       5       5
        4114
      0        0       0       0
        4618
 100001   100001  100001  100001
        4744
      3        3       3       3
        4870
      0        0       0       0
        5374
 100000   100000  100000  100000
        5500
     15       15      15      15
        5626
      0        0       0       0
        6130
 100001   100001  100001  100001
        6256
     13       13      13      13
        6382
      0        0       0       0
        6886
 100000   100000  100000  100000
        7012
      5        5       5       5
        7138
      0        0       0       0
        7642
 100000   100000  100000  100000
        7768
     15       15      15      15
        7894
      0        0       0       0
        8398
 100002   100002  100002  100002
        8524
      3        3       3       3
        8650
      0        0       0       0
        9154
 100002   100002  100002  100002
        9280
     13       13      13      13
        9406
      0        0       0       0
        9910
 100000   100000  100000  100000
       10036
      5        5       5       5
       10162
      0        0       0       0
       10666
 100001   100001  100001  100001
       10792
      7        7       7       7
       10918
      0        0       0       0
       11422
 100002   100002  100002  100002
       11548
      7        7       7       7
       11674
      0        0       0       0
       12178
 100003   100003  100003  100003
       12304
      3        3       3       3
       12430
      0        0       0       0
       12934
 100000   100000  100000  100000
       13060
      3        3       3       3
       13186
      0        0       0       0
       13690
 100001   100001  100001  100001
       13816
      5        5       5       5
       13942
      0        0       0       0
       14446
```

```
100000 100000 100000 100000
      14572
    13     13     13     13
      14698
     0      0      0      0
      15202
100001 100001 100001 100001
      15328
    15     15     15     15
      15454
     0      0      0      0
      15958
100001 100001 100001 100001
      16084
    11     11     11     11
      16210
     0      0      0      0
      16714
100001 100001 100001 100001
      16840
     1      1      1      1
      16966
     0      0      0      0
      17470
100001 100001 100001 100001
      17596
    11     11     11     11
      17722
     0      0      0      0
      18226
```

```
100001 100001 100001 100001
      18352
    11     11     11     11
      18478
     0      0      0      0
      18982
100000 100000 100000 100000
      19108
     7      7      7      7
      19234
     0      0      0      0
      19738
100001 100001 100001 100001
      19864
     5      5      5      5
      19990
     0      0      0      0
      20494
100002 100002 100002 100002
      20620
     3      3      3      3
      20746
     0      0      0      0
      21250
100003 100003 100003 100003
      21376
     7      7      7      7
      21502
     0      0      0      0
      22006
```

```
100001 100001 100001 100001
     22132
    15     15     15     15
     22258
     0      0      0      0
     22762
100001 100001 100001 100001
     22888
     5      5      5      5
     23014
     0      0      0      0
     23518
100003 100003 100003 100003
     23644
    13     13     13     13
     23770
     0      0      0      0
     24274
100003 100003 100003 100003
     24400
     3      3      3      3
     24526
     0      0      0      0
     25030
    25     25     25     25
     25156
     3      3      3      3
     25282
     0      0      0      0
     25786
    25     25     25     25
     25912
    13     13     13     13
     26038
     0      0      0      0
     26542
  1025   1025   1025   1025
     26668
     5      5      5      5
     26794
     0      0      0      0
     27298
  1025   1025   1025   1025
     27424
    15     15     15     15
     27550
     0      0      0      0
     28054
    25     25     25     25
     28180
     7      7      7      7
     28306
     0      0      0      0
     28810
   425    425    425    425
     28936
     3      3      3      3
     29062
     0      0      0      0
     29566
```

```
1025   1025   1025   1025
     29692
     5      5      5      5
     29818
     0      0      0      0
     30322
1425   1425   1425   1425
     30448
     7      7      7      7
     30574
     0      0      0      0
     31078
1025   1025   1025   1025
     31204
    11     11     11     11
     31330
     0      0      0      0
     31834
1025   1025   1025   1025
     31960
    11     11     11     11
     32086
     0      0      0      0
     32590
1025   1025   1025   1025
     32716
     1      1      1      1
     32842
     0      0      0      0
     33346
1025   1025   1025   1025
     33472
    11     11     11     11
     33598
     0      0      0      0
     34102
1025   1025   1025   1025
     34228
    15     15     15     15
     34354
     0      0      0      0
     34858
1425   1425   1425   1425
     34984
    13     13     13     13
     35110
     0      0      0      0
     35614
1025   1025   1025   1025
     35740
     5      5      5      5
     35866
     0      0      0      0
     36370
1425   1425   1425   1425
     36496
     3      3      3      3
     36622
     0      0      0      0
     37126
```

25 25 25 25
37252
3 3 3 3
37378
0 0 0 0
37882
425 425 425 425
38008
7 7 7 7
38134
0 0 0 0
38638
1025 1025 1025 1025
38764
7 7 7 7
38890
0 0 0 0
39394
1425 1425 1425 1425
39520
5 5 5 5
39646
0 0 0 0
40150
425 425 425 425
40276
13 13 13 13
40402
0 0 0 0
40906
425 425 425 425
41032
3 3 3 3
41158
0 0 0 0
41662
1425 1425 1425 1425
41788
15 15 15 15
41914
0 0 0 0
42418
1425 1425 1425 1425
42544
5 5 5 5
42670
0 0 0 0
43174
1025 1025 1025 1025
43300
13 13 13 13
43426
0 0 0 0
43930
1425 1425 1425 1425
44056
15 15 15 15
44182
0 0 0 0
44686
1025 1025 1025 1025
44812
3 3 3 3
44938
0 0 0 0
45442
1425 1425 1425 1425
45568
5 5 5 5
45694
0 0 0 0
46198
1425 1425 1425 1425
46324
11 11 11 11
46450
0 0 0 0
46954
1425 1425 1425 1425
47080
11 11 11 11
47206
0 0 0 0
47710
1425 1425 1425 1425
47836
11 11 11 11
47962
0 0 0 0
48466
1425 1425 1425 1425
48592
1 1 1 1

Voter Test B01 3XB

```
VB013XB      GOOD CIRCUIT
          12
       0        0        0        0
         100
       0        0        0        0
         226
       0        0        0        0
         334
       0        0        0        0
         838
  100000   100000   100000   100000
         964
       2        2        2        2
        1090
       0        0        0        0
        1594
  100000   100000   100000   100000
        1720
      12       12       12       12
        1846
       0        0        0        0
        2350
  100000   100000   100000   100000
        2476
      12       12       12       12
        2602
       0        0        0        0
        3106
  100000   100000   100000   100000
        3232
      12       12       12       12
        3358
       0        0        0        0
        3862
  100000   100000   100000   100000
        3988
       6        6        6        6
        4114
       0        0        0        0
        4618
  100001   100001   100001   100001
        4744
       3        3        3        3
        4870
       0        0        0        0
        5374
  100000   100000   100000   100000
        5500
      16       16       16       16
        5626
       0        0        0        0
        6130
  100001   100001   100001   100001
        6256
      13       13       13       13
        6382
       0        0        0        0
        6886
  100000   100000   100000   100000
        7012
       6        6        6        6
        7138
       0        0        0        0
        7642
  100000   100000   100000   100000
        7768
      16       16       16       16
        7894
       0        0        0        0
        8398
  100002   100002   100002   100002
        8524
       3        3        3        3
        8650
       0        0        0        0
        9154
  100002   100002   100002   100002
        9280
      13       13       13       13
        9406
       0        0        0        0
        9910
  100000   100000   100000   100000
       10036
       6        6        6        6
       10162
       0        0        0        0
       10666
  100001   100001   100001   100001
       10792
       7        7        7        7
       10918
       0        0        0        0
       11422
  100002   100002   100002   100002
       11548
       7        7        7        7
       11674
       0        0        0        0
       12178
  100003   100003   100003   100003
       12304
       3        3        3        3
       12430
       0        0        0        0
       12934
  100000   100000   100000   100000
       13060
       3        3        3        3
       13186
       0        0        0        0
       13690
  100001   100001   100001   100001
       13816
       6        6        6        6
       13942
       0        0        0        0
       14446
```

```
100000 100000 100000 100000
    14572
    13     13     13     13
    14698
     0      0      0      0
    15202
100001 100001 100001 100001
    15328
    16     16     16     16
    15454
     0      0      0      0
    15958
100001 100001 100001 100001
    16084
    12     12     12     12
    16210
     0      0      0      0
    16714
100001 100001 100001 100001
    16840
     2      2      2      2
    16966
     0      0      0      0
    17470
100001 100001 100001 100001
    17596
    12     12     12     12
    17722
     0      0      0      0
    18226
100001 100001 100001 100001
    18352
    12     12     12     12
    18478
     0      0      0      0
    18982
100000 100000 100000 100000
    19108
     7      7      7      7
    19234
     0      0      0      0
    19738
100001 100001 100001 100001
    19864
     6      6      6      6
    19990
     0      0      0      0
    20494
100002 100002 100002 100002
    20620
     3      3      3      3
    20746
     0      0      0      0
    21250
100003 100003 100003 100003
    21376
     7      7      7      7
    21502
     0      0      0      0
    22006
```

```
100001 100001 100001 100001
    22132
    16     16     16     16
    22258
     0      0      0      0
    22762
100001 100001 100001 100001
    22888
     6      6      6      6
    23014
     0      0      0      0
    23518
100003 100003 100003 100003
    23644
    13     13     13     13
    23770
     0      0      0      0
    24274
100003 100003 100003 100003
    24400
     3      3      3      3
    24526
     0      0      0      0
    25030
    25     25     25     25
    25156
     3      3      3      3
    25282
     0      0      0      0
    25786
    25     25     25     25
    25912
    13     13     13     13
    26038
     0      0      0      0
    26542
  1025   1025   1025   1025
    26668
     6      6      6      6
    26794
     0      0      0      0
    27298
  1025   1025   1025   1025
    27424
    16     16     16     16
    27550
     0      0      0      0
    28054
    25     25     25     25
    28180
     7      7      7      7
    28306
     0      0      0      0
    28810
   425    425    425    425
    28936
     3      3      3      3
    29062
     0      0      0      0
    29566
```

1025 1025 1025 1025
29692
6 6 6 6
29818
0 0 0 0
30322
1425 1425 1425 1425
30448
7 7 7 7
30574
0 0 0 0
31078
1025 1025 1025 1025
31204
12 12 12 12
31330
0 0 0 0
31834
1025 1025 1025 1025
31960
12 12 12 12
32086
0 0 0 0
32590
1025 1025 1025 1025
32716
2 2 2 2
32842
0 0 0 0
33346
1025 1025 1025 1025
33472
12 12 12 12
33598
0 0 0 0
34102
1025 1025 1025 1025
34228
16 16 16 16
34354
0 0 0 0
34858
1425 1425 1425 1425
34984
13 13 13 13
35110
0 0 0 0
35614
1025 1025 1025 1025
35740
6 6 6 6
35866
0 0 0 0
36370
1425 1425 1425 1425
36496
3 3 3 3
36622
0 0 0 0
37126
25 25 25 25
37252
3 3 3 3
37378
0 0 0 0
37882
425 425 425 425
38008
7 7 7 7
38134
0 0 0 0
38638
1025 1025 1025 1025
38764
7 7 7 7
38890
0 0 0 0
39394
1425 1425 1425 1425
39520
6 6 6 6
39646
0 0 0 0
40150
425 425 425 425
40276
13 13 13 13
40402
0 0 0 0
40906
425 425 425 425
41032
3 3 3 3
41158
0 0 0 0
41662
1425 1425 1425 1425
41788
16 16 16 16
41914
0 0 0 0
42418
1425 1425 1425 1425
42544
6 6 6 6
42670
0 0 0 0
43174
1025 1025 1025 1025
43300
13 13 13 13
43426
0 0 0 0
43930
1425 1425 1425 1425
44056
16 16 16 16
44182
0 0 0 0
44686

```
1025   1025   1025   1025
    44812
3      3      3      3
    44938
0      0      0      0
    45442
1425   1425   1425   1425
    45568
6      6      6      6
    45694
0      0      0      0
    46198
1425   1425   1425   1425
    46324
12     12     12     12
    46450
0      0      0      0
    46954
1425   1425   1425   1425
    47080
12     12     12     12
    47206
0      0      0      0
    47710
1425   1425   1425   1425
    47836
12     12     12     12
    47962
0      0      0      0
    48466
1425   1425   1425   1425
    48592
2      2      2      2
100000 100000 100000 100000
     7012
6      6      6      6
     7138
0      0      0      0
     7642
100000 100000 100000 100000
     7768
16     16     16     16
     7894
0      0      0      0
     8398
100002 100002 100002 100002
     8524
```

```
5      5      5      5
     8650
0      0      0      0
     9154
100002 100002 100002 100002
     9280
15     15     15     15
     9406
0      0      0      0
     9910
100000 100000 100000 100000
    10036
6      6      6      6
    10162
0      0      0      0
    10666
100001 100001 100001 100001
    10792
7      7      7      7
    10918
0      0      0      0
    11422
100002 100002 100002 100002
    11548
7      7      7      7
    11674
0      0      0      0
    12178
100003 100003 100003 100003
    12304
5      5      5      5
    12430
0      0      0      0
    12934
100000 100000 100000 100000
    13060
5      5      5      5
    13186
0      0      0      0
    13690
100001 100001 100001 100001
    13816
6      6      6      6
    13942
0      0      0      0
    14446
```

Voter Test B01 3XC

```
VB013XC      GOOD CIRCUIT
          12
      0       0       0       0
         100
      0       0       0       0
         226
      0       0       0       0
         334
      0       0       0       0
         838
 100000  100000  100000  100000
         964
      4       4       4       4
        1090
      0       0       0       0
        1594
 100000  100000  100000  100000
        1720
     14      14      14      14
        1846
      0       0       0       0
        2350
 100000  100000  100000  100000
        2476
     14      14      14      14
        2602
      0       0       0       0
        3106
 100000  100000  100000  100000
        3232
     14      14      14      14
        3358
      0       0       0       0
        3862
 100000  100000  100000  100000
        3988
      6       6       6       6
        4114
      0       0       0       0
        4618
 100001  100001  100001  100001
        4744
      5       5       5       5
        4870
      0       0       0       0
        5374
 100000  100000  100000  100000
        5500
     16      16      16      16
        5626
      0       0       0       0
        6130
 100001  100001  100001  100001
        6256
     15      15      15      15
        6382
      0       0       0       0
        6886
 100000  100000  100000  100000
       14572
     15      15      15      15
       14698
      0       0       0       0
       15202
 100001  100001  100001  100001
       15328
     16      16      16      16
       15454
      0       0       0       0
       15958
 100001  100001  100001  100001
       16084
     14      14      14      14
       16210
      0       0       0       0
       16714
 100001  100001  100001  100001
       16840
      4       4       4       4
       16966
      0       0       0       0
       17470
 100001  100001  100001  100001
       17596
     14      14      14      14
       17722
      0       0       0       0
       18226
 100001  100001  100001  100001
       18352
     14      14      14      14
       18478
      0       0       0       0
       18982
 100000  100000  100000  100000
       19108
      7       7       7       7
       19234
      0       0       0       0
       19738
 100001  100001  100001  100001
       19864
      6       6       6       6
       19990
      0       0       0       0
       20494
 100002  100002  100002  100002
       20620
      5       5       5       5
       20746
      0       0       0       0
       21250
 100003  100003  100003  100003
       21376
      7       7       7       7
       21502
      0       0       0       0
       22006
```

100001 100001 100001 100001
22132
16 16 16 16
22258
0 0 0 0
22762
100001 100001 100001 100001
22888
6 6 6 6
23014
0 0 0 0
23518
100003 100003 100003 100003
23644
15 15 15 15
23770
0 0 0 0
24274
100003 100003 100003 100003
24400
5 5 5 5
24526
0 0 0 0
25030
25 25 25 25
25156
5 5 5 5
25282
0 0 0 0
25786
25 25 25 25
25912
15 15 15 15
26038
0 0 0 0
26542
1025 1025 1025 1025
26668
6 6 6 6
26794
0 0 0 0
27298
1025 1025 1025 1025
27424
16 16 16 16
27550
0 0 0 0
28054
25 25 25 25
28180
7 7 7 7
28306
0 0 0 0
28810
425 425 425 425
28936
5 5 5 5
29062
0 0 0 0
29566
1025 1025 1025 1025
29692
6 6 6 6
29818
0 0 0 0
30322
1425 1425 1425 1425
30448
7 7 7 7
30574
0 0 0 0
31078
1025 1025 1025 1025
31204
14 14 14 14
31330
0 0 0 0
31834
1025 1025 1025 1025
31960
14 14 14 14
32086
0 0 0 0
32590
1025 1025 1025 1025
32716
4 4 4 4
32842
0 0 0 0
33346
1025 1025 1025 1025
33472
14 14 14 14
33598
0 0 0 0
34102
1025 1025 1025 1025
34228
16 16 16 16
34354
0 0 0 0
34858
1425 1425 1425 1425
34984
15 15 15 15
35110
0 0 0 0
35614
1025 1025 1025 1025
35740
6 6 6 6
35866
0 0 0 0
36370
1425 1425 1425 1425
36496
5 5 5 5
36622
0 0 0 0
37126

25 25 25 25
37252
5 5 5 5
37378
0 0 0 0
37882
425 425 425 425
38008
7 7 7 7
38134
0 0 0 0
38638
1025 1025 1025 1025
38764
7 7 7 7
38890
0 0 0 0
39394
1425 1425 1425 1425
39520
6 6 6 6
39646
0 0 0 0
40150
425 425 425 425
40276
15 15 15 15
40402
0 0 0 0
40906
425 425 425 425
41032
5 5 5 5
41158
0 0 0 0
41662
1425 1425 1425 1425
41788
16 16 16 16
41914
0 0 0 0
42418
1425 1425 1425 1425
42544
6 6 6 6
42670
0 0 0 0
43174
1025 1025 1025 1025
43300
15 15 15 15
43426
0 0 0 0
43930
1425 1425 1425 1425
44056
16 16 16 16
44182
0 0 0 0
44686
1025 1025 1025 1025
44812
5 5 5 5
44938
0 0 0 0
45442
1425 1425 1425 1425
45568
6 6 6 6
45694
0 0 0 0
46198
1425 1425 1425 1425
46324
14 14 14 14
46450
0 0 0 0
46954
1425 1425 1425 1425
47080
14 14 14 14
47206
0 0 0 0
47710
1425 1425 1425 1425
47836
14 14 14 14
47962
0 0 0 0
48466
1425 1425 1425 1425
48592
4 4 4 4

Voter Test B01 3XD

```
VB013XD      GOOD CIRCUIT
          12
       0        0        0        0
         100
       0        0        0        0
         226
       0        0        0        0
         334
       0        0        0        0
         838
  100000   100000   100000   100000
         964
      10       10       10       10
        1090
       0        0        0        0
        1594
  100000   100000   100000   100000
        1720
      14       14       14       14
        1846
       0        0        0        0
        2350
  100000   100000   100000   100000
        2476
      14       14       14       14
        2602
       0        0        0        0
        3106
  100000   100000   100000   100000
        3232
      14       14       14       14
        3358
       0        0        0        0
        3862
  100000   100000   100000   100000
        3988
      12       12       12       12
        4114
       0        0        0        0
        4618
  100001   100001   100001   100001
        4744
      11       11       11       11
        4870
       0        0        0        0
        5374
  100000   100000   100000   100000
        5500
      16       16       16       16
        5626
       0        0        0        0
        6130
  100001   100001   100001   100001
        6256
      15       15       15       15
        6382
       0        0        0        0
        6886
  100000   100000   100000   100000
        7012
      12       12       12       12
        7138
       0        0        0        0
        7642
  100000   100000   100000   100000
        7768
      16       16       16       16
        7894
       0        0        0        0
        8398
  100002   100002   100002   100002
        8524
      11       11       11       11
        8650
       0        0        0        0
        9154
  100002   100002   100002   100002
        9280
      15       15       15       15
        9406
       0        0        0        0
        9910
  100000   100000   100000   100000
       10036
      12       12       12       12
       10162
       0        0        0        0
       10666
  100001   100001   100001   100001
       10792
      13       13       13       13
       10918
       0        0        0        0
       11422
  100002   100002   100002   100002
       11548
      13       13       13       13
       11674
       0        0        0        0
       12178
  100003   100003   100003   100003
       12304
      11       11       11       11
       12430
       0        0        0        0
       12934
  100000   100000   100000   100000
       13060
      11       11       11       11
       13186
       0        0        0        0
       13690
  100001   100001   100001   100001
       13816
      12       12       12       12
       13942
       0        0        0        0
       14446
```

```
100000 100000 100000 100000
      14572
    15     15     15     15
      14698
     0      0      0      0
      15202
100001 100001 100001 100001
      15328
    16     16     16     16
      15454
     0      0      0      0
      15958
100001 100001 100001 100001
      16084
    14     14     14     14
      16210
     0      0      0      0
      16714
100001 100001 100001 100001
      16840
    10     10     10     10
      16966
     0      0      0      0
      17470
100001 100001 100001 100001
      17596
    14     14     14     14
      17722
     0      0      0      0
      18226
100001 100001 100001 100001
      18352
    14     14     14     14
      18478
     0      0      0      0
      18982
100000 100000 100000 100000
      19108
    13     13     13     13
      19234
     0      0      0      0
      19738
100001 100001 100001 100001
      19864
    12     12     12     12
      19990
     0      0      0      0
      20494
100002 100002 100002 100002
      20620
    11     11     11     11
      20746
     0      0      0      0
      21250
100003 100003 100003 100003
      21376
    13     13     13     13
      21502
     0      0      0      0
      22006
100001 100001 100001 100001
      22132
    16     16     16     16
      22258
     0      0      0      0
      22762
100001 100001 100001 100001
      22888
    12     12     12     12
      23014
     0      0      0      0
      23518
100003 100003 100003 100003
      23644
    15     15     15     15
      23770
     0      0      0      0
      24274
100003 100003 100003 100003
      24400
    11     11     11     11
      24526
     0      0      0      0
      25030
    25     25     25     25
      25156
    11     11     11     11
      25282
     0      0      0      0
      25786
    25     25     25     25
      25912
    15     15     15     15
      26038
     0      0      0      0
      26542
  1025   1025   1025   1025
      26668
    12     12     12     12
      26794
     0      0      0      0
      27298
  1025   1025   1025   1025
      27424
    16     16     16     16
      27550
     0      0      0      0
      28054
    25     25     25     25
      28180
    13     13     13     13
      28306
     0      0      0      0
      28810
   425    425    425    425
      28936
    11     11     11     11
      29062
     0      0      0      0
      29566
```

1025 1025 1025 1025
29692
12 12 12 12
29818
0 0 0 0
30322
1425 1425 1425 1425
30448
13 13 13 13
30574
0 0 0 0
31078
1025 1025 1025 1025
31204
14 14 14 14
31330
0 0 0 0
31834
1025 1025 1025 1025
31960
14 14 14 14
32086
0 0 0 0
32590
1025 1025 1025 1025
32716
10 10 10 10
32842
0 0 0 0
33346
1025 1025 1025 1025
33472
14 14 14 14
33598
0 0 0 0
34102
1025 1025 1025 1025
34228
16 16 16 16
34354
0 0 0 0
34858
1425 1425 1425 1425
34984
15 15 15 15
35110
0 0 0 0
35614
1025 1025 1025 1025
35740
12 12 12 12
35866
0 0 0 0
36370
1425 1425 1425 1425
36496
11 11 11 11
36622
0 0 0 0
37126

25 25 25 25
37252
11 11 11 11
37378
0 0 0 0
37882
425 425 425 425
38008
13 13 13 13
38134
0 0 0 0
38638
1025 1025 1025 1025
38764
13 13 13 13
38890
0 0 0 0
39394
1425 1425 1425 1425
39520
12 12 12 12
39646
0 0 0 0
40150
425 425 425 425
40276
15 15 15 15
40402
0 0 0 0
40906
425 425 425 425
41032
11 11 11 11
41158
0 0 0 0
41662
1425 1425 1425 1425
41788
16 16 16 16
41914
0 0 0 0
42418
1425 1425 1425 1425
42544
12 12 12 12
42670
0 0 0 0
43174
1025 1025 1025 1025
43300
15 15 15 15
43426
0 0 0 0
43930
1425 1425 1425 1425
44056
16 16 16 16
44182
0 0 0 0
44686

1025	1025	1025	1025
44812			
11	11	11	11
44938			
0	0	0	0
45442			
1425	1425	1425	1425
45568			
12	12	12	12
45694			
0	0	0	0
46198			
1425	1425	1425	1425
46324			
14	14	14	14
46450			
0	0	0	0
46954			
1425	1425	1425	1425
47080			
14	14	14	14
47206			
0	0	0	0
47710			
1425	1425	1425	1425
47836			
14	14	14	14
47962			
0	0	0	0
48466			
1425	1425	1425	1425
48592			
10	10	10	10

NASA
National Aeronautics and Space Administration

Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA CR-178390		

4. Title and Subtitle	5. Report Date
A Fault Injection Experiment Using the AIRLAB Diagnostic Emulation Facility	March 1988
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Robert Baker, Scott Mangum, and Charlotte Scheper	
	10. Work Unit No.
	505-66-21-01

9. Performing Organization Name and Address	11. Contract or Grant No.
Center for Digital Systems Research Research Triangle Institute Research Triangle Park, NC 27709	NAS1-17964, Task 5
	13. Type of Report and Period Covered
12. Sponsoring Agency Name and Address	Contractor Report
National Aeronautics and Space Administration Langley Research Center Hampton, VA 23665-5225	14. Sponsoring Agency Code

15. Supplementary Notes

Technical Monitor: Charles W. Meissner, Jr.
Langley Research Center

16. Abstract This report describes the preparation for, conduct of, and results of a simulation based fault injection experiment conducted using the AIRLAB Diagnostic Emulation facilities. The primary objective of this experiment was to determine the effectiveness of the diagnostic self-test sequences used to uncover latent faults in a logic network providing the key fault tolerance features for a flight control computer. Since this experiment resulted in the most extensive use of the AIRLAB Diagnostic Emulation facilities to date, a secondary, but essential, objective was to develop methods, tools, and techniques for conducting the experiment.

In this experiment, more than 1600 faults were injected into a logic gate level model of the Data Communicator/Interstage (C/I), a key component in the C.S. Draper Laboratories Fault Tolerant Processor. For each fault injected, diagnostic self-test sequences consisting of over 300 test vectors were supplied to the C/I model as inputs. For each test vector within a test sequence, the outputs from the C/I model were compared to the outputs of a fault free C/I. If the outputs differed, the fault was considered detectable for the given test vector. These results were then analyzed to determine the effectiveness of various test sequences.

The experiment results established the overall coverage of various self-test diagnostics, identified certain areas in the C/I logic where the diagnostic tests did not locate faults, and suggest opportunities for reducing average fault latency and improving overall diagnostic test performance.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Logic Simulation Fault Effects Self-Test Design Fault Coverage Fault Latency	Unclassified-Unlimited Subject Category 62

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of pages	22. Price
Unclassified	Unclassified	129	A07