

Spring 5-31-1982

Pattern generation and fault detection in digital circuits using a microprocessor

Syed Riffat Ali
New Jersey Institute of Technology

Follow this and additional works at: <https://digitalcommons.njit.edu/theses>



Part of the [Electrical and Electronics Commons](#)

Recommended Citation

Ali, Syed Riffat, "Pattern generation and fault detection in digital circuits using a microprocessor" (1982).
Theses. 1467.
<https://digitalcommons.njit.edu/theses/1467>

This Thesis is brought to you for free and open access by the Electronic Theses and Dissertations at Digital Commons @ NJIT. It has been accepted for inclusion in Theses by an authorized administrator of Digital Commons @ NJIT. For more information, please contact digitalcommons@njit.edu.

Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be “used for any purpose other than private study, scholarship, or research.” If a user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of “fair use” that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select “Pages from: first page # to: last page #” on the print dialog screen

The Van Houten library has removed some of the personal information and all signatures from the approval page and biographical sketches of theses and dissertations in order to protect the identity of NJIT graduates and faculty.

PATTERN GENERATION AND FAULT
DETECTION IN DIGITAL CIRCUITS
USING A MICROPROCESSOR

by
Syed Riffat Ali

Dissertation submitted to the Faculty of the Graduate School
of the New Jersey Institute of Technology in partial fulfillment
of the requirements for the professional
degree of Electrical Engineer
1982

APPROVAL SHEET

Title of Thesis: Pattern Generation and Fault Detection in Digital Circuits
Using a Microprocessor

Name of Candidate: Syed Riffat Ali
Professional Degree of Electrical Engineer, 1982

Thesis and Abstract Approval:

_____	_____
Dr. J. Frank Assistant Professor Electrical Engineering	Date
_____	_____
_____	Date
_____	_____
_____	Date
_____	_____
_____	Date

170345

VITA

Name: Syed Riffat Ali.

Degree and date to be conferred: Deg. Elect. Engg., 1982.

Secondary education: Notre Dame College, 1963.

Collegiate institutions attended	Dates	Degree	Date of Degree
East Pakistan University of Engineering Technology	1963-1967	B.S.E.E.	October 1967
Tuskegee Institute	1968-1971	M.S.E.E.	September 1971
New Jersey Institute of Technology	1972-1982	Deg.E.E.	May 1982

Major: Electrical Engineering.

- Publications: 1. A thesis on Single Side Band Exciter (B.S.E.E. Thesis).
2. The Study of the Performance of a System Having Multiple Nonlinearities Using Describing Function Technique (M.S.E.E. Thesis).

Positions held: R & D Engineer

ABSTRACT

Title of Thesis: Pattern Generation and Fault Detection in Digital Circuits
Using a Microprocessor

Syed Riffat Ali, Professional Degree of Electrical Engineer, 1982

Thesis directed by: Assistant Professor J. Frank *JF 5/3/82*

The main object of this Professional Project was to establish a microcomputer based system which could detect faults in digital circuits. Hardware and software for the Motorola 6800 microcomputer system was fully developed. Node level diagnostic programs were then used to pump known "Bit Patterns" into digital circuits and responses recorded via Pseudorandom binary sequence generator using CRC code. This method also known as "Signature Analysis" was then applied to detect faults successfully in Universal Asynchronous Receiver Transmitter or UART Circuits.

TO: MY JAAN

ACKNOWLEDGEMENT

The author wishes to acknowledge the help rendered by his advisor Dr. J. Frank, without whose constant direction, dedication and understanding this project would have been an impossibility in the true sense of the word.

The author also wishes to thank Dr. E. M. Rips and Dr. W. H. Ball for their patient attention and help.

The greatest sacrifice, however, was rendered by my dear wife Ishrat, who lived through my lack of attention, indifference to house chores, etc., during the last two years of this project. Above all I am grateful for her encouragement which finally pushed me into finishing this project.

Last, but not least, I would like to thank Ms. Judi Engelhardt for her excellent typing and proofreading of my manuscript.

TABLE OF CONTENTS

	Page
INTRODUCTION	1
LIST OF FIGURES	-v-
ACKNOWLEDGMENT	-ii-
 Chapter	
I. Faults in Digital Circuits	2
Brief Survey of Techniques which can be Applied for Detection of Faults in Digital Circuits	
Limitations of Conventional Techniques	
II. Signature Analysis	11
Signature Analysis	
Signature Analysis - Formal Definition	
Pseudorandom Binary Sequence (PRBS)	
PRBS Generator	
Error Detection by PRBS	
Top Selection of PRBS Generator	
Probability of Error Detection in Signature Analysis	
III. Description of a Signature Analysis System	19
Proposed System	
Description of the Basic System	
Description of Additional Hardware and Software Required for the Signature Analysis System	
IV. Diagnostic Bus Interface	23
Peripheral Interface Adapter	
Internal Organization of PIA	
PIA Operating Modes	
PIA Internal Registers	
Parallel Interface Circuit	
I/O Decoder Circuit Operational Note	
Interfacing the Diagnostic Interface to the Signature Analysis System	

	Page
V. Signature Analysis Module	32
System Structure	
Definition of Key Signals - Stimulus, Monitored Signals, Clock Signal, Start Signal and Stop Signal	
Procedure for Obtaining a Signature	
VI. Circuit Under Test	37
Special Requirements for Circuit Under Test	
Special Schematic for the Circuit Under Test	
CONCLUSION	41
APPENDIX A	42
SS50 - S100 Bus Conversion Design	
APPENDIX B	46
Software for the Signature Analysis System	
LIST OF REFERENCES	55

LIST OF FIGURES

Figure	Page
1. A Simplified Block Diagram of the Diagnostic System	1
2. OR, and Function Tables	4
3. D-Cube Fault	5
4. Diagnostic Tree	6
5. Reduced Tree	7
6. Flow Table	7
7. Detection of s-a-1 Fault	8
8. Success or Tree Fault	9
9. PRB Sequence	12
10. Selection of TAP for $T(x) = x^{16} + x^{12} + x^9 + x^7 + 1$	14
11. 20-Bit Sequence Generating a Signature	15
12. Probability of Detection of Errors for Signature Analysis for $K = 16$ for PRBS Generator	18
13. The Signature Analysis System	20
14. MC6820 PIA I/O Diagram	23
15. I/O PORTS	25
16. Internal Registers of PIA	26
17. Control Register Interpretation	28
18. Diagnostic Interface Block Diagram	31
19. Signature Analysis Module	32
20. Signature Collection System	34
21. Signature Schematic of the Circuit Under Test	40

INTRODUCTION

The main object of the project will be to establish a Microcomputer based system which will be able to diagnose a faulty digital system by pumping a known "bit pattern" into the digital system, recording the system response at various "strategic points" in a "good" system and then comparing the same response in a "faulty" system, and then diagnosing the faulty component.

The proposed "Diagnostic System" must have the following capabilities:

1. It should be capable of generating test patterns.
2. It should be able to apply the patterns to a "client" system and collect responses and store them.
3. It should be able to compare the patterns of a "good" system to a "faulty" one, diagnose the faulty system and detect the faulty element.
4. Finally, the system should be able to diagnose itself, before proceeding to diagnose a faulty system.

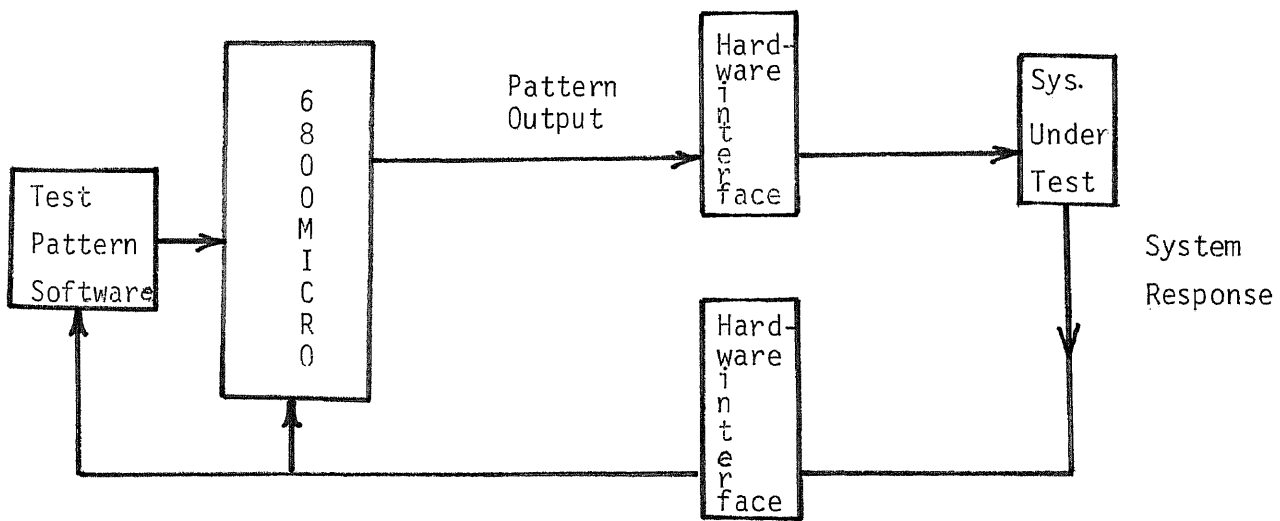


FIG. 1

A Simplified Block Diagram of the Diagnostic System

CHAPTER 1

FAULTS IN DIGITAL CIRCUITS

The prime requirement of a digital system is the ability to operate correctly over a sufficiently long period of time. This requirement becomes stringent in communication systems and computers that operate in real time. To meet this requirement the system has to be tested from time to time to determine if it is functioning properly. The fault, if any, should be detected and the faulty unit should be repaired or replaced.

The set of tests used should be complete and should be capable of detecting any fault that is likely to occur.

The set should be made as small as possible in order to minimize the time required for this maintenance function.

The number of tests required will obviously depend on the type of circuit involved. Redundant circuits where faults are masked should also be considered in the tests.

Digital circuits are normally divided into two basic classes:

1. Combinational Circuits
2. Sequential Circuits

A Combinational Circuit has no feedback loops. A Combinational function Z_i of a set of variable (y_1, y_2, \dots, y_n) depends only on the present value of inputs, i.e., $Z_i = f_i(y_1, y_2, \dots, y_n)$. A Combinational Circuit can be represented by Truth table, Karnaugh map, etc.

In a Sequential Circuit the output not only depends on present values of input but also on the past values of the input. The past values of the inputs

are represented by the internal status of the Sequential Circuit.

Two different models of sequential machines have been presented by Mealy and Moore.

In the Mealy machine the output at any time is dependent on the input and internal state at that time, and in the Moore machine the output is a function of only the internal state.

Brief Survey of Different Methods Used for Test Derivation for Combinational Circuits

The faults will be of the following types:

1. Stuck at zero (s-a-0)
2. Stuck at one (s-a-1)

Truth Table Method

This method compares the truth tables of normal and faulty circuits. Suppose that the inputs to a combinational circuit are x_1, x_2, \dots, x_n and outputs are Z_1, Z_2, \dots, Z_m where $Z_i = f_i(x_1, x_2, \dots, x_n)$, $i = 1, 2, \dots, m$. For a set of faults F and any fault $d \in F$, let $Z_i^d = f_i^d(x_1, x_2, \dots, x_n)$ be the value of the i th output when the fault d is present. Then an input vector $x^j = (x_1^j, x_2^j, \dots, x_n^j)$ is a test for detecting the fault d if and only if

$f_i(x^j) \oplus f_i^d(x^j) = 1$ for some i , $1 \leq i \leq m$ where \oplus represents the Exclusive - or Operator.

Method of Boolean Differences

The Boolean difference of a Function $F(x_1, x_2, \dots, x_n)$ with respect to one of its inputs x_i (denoted as $dF(x)/dx_i$) is defined as:

$$\frac{dF(x)}{dx_i} = F(x_1, x_2, \dots, 0, \dots, x_n) \oplus F(x_1, x_2, \dots, 1, \dots, x_n); \text{ for the } i^{\text{th}} \text{ component.}$$

The Boolean difference is not a derivative but a notation.

$dF(x)/dx_i \equiv 0$, means that F is independent of x_i .

and $dF(x)/dx_i \equiv 1$, means that input affects output.

The set of tests for a fault on x_i can be represented by the following:

$$x_i \frac{dF(x)}{dx_i} \text{ for } x_i \text{ s-a-0}$$

and $\bar{x}_i \frac{dF(x)}{dx_i} \text{ for } x_i \text{ s-a-1}$

For example, consider the circuit shown in Fig. 2.

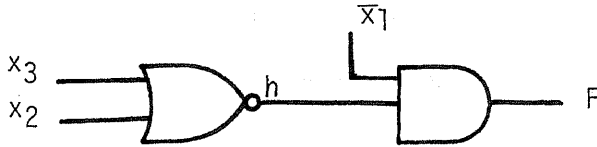


Fig. 2

The output F for the above circuit is given by:

$$F = (x_3 + x_2) \bar{x}_1 \equiv \bar{x}_1 h; \quad \frac{dF}{dh} = \bar{x}_1$$

Now the test for h s-a-0 would be:

$$h \frac{dF}{dh} = (x_3 + x_2) \bar{x}_1 = \bar{x}_1 x_3 + \bar{x}_1 x_2$$

And test for h s-a-1 would be:

$$\bar{h} \frac{dF}{dh} = (\bar{x}_3 + \bar{x}_2) \bar{x}_1 = \bar{x}_1 \bar{x}_3 \bar{x}_2$$

Path Sensitizing

Derivation of the test using Path Sensitizing is established by assigning a value opposite to the fault condition. A value of "1" is assigned to a terminal with s-a-0 fault and a "0" for a s-a-1 fault. A path is chosen from the fault to one of the output terminals. The inputs to the gates along this path are assigned values so as to propagate any change on the faulty terminal along the chosen path to the output terminal.

The path is now said to be sensitized. One or more tests for detecting the fault are obtained by determining network inputs which will produce the desired values on the gate inputs along the sensitized path. If a unique set of inputs is not obtained, the process is repeated and a different path is chosen for sensitizing.

D-Algorithm

The symbol D is used to represent a signal that is 1 in the normal circuit and 0 in the faulty circuit. The symbol \bar{D} is used to represent the signal which is normally 0 and 1 when faulty. And in using D-Algorithm, the OR and AND functions can be represented by tables shown in Fig. 2a and Fig. 2b.

+	0	1	D	\bar{D}
0	0	1	D	\bar{D}
1	1	1	1	1
D	D	1	D	1
\bar{D}	\bar{D}	1	1	\bar{D}

OR - Function

Fig. 2a

Input functions are read on the row and column sides of the tables within the bold lines. The tables shown are only good for two input gates. The mid part of the table defines the output in D-Algorithm. For example input of "0" and "D" will cause the output to be "D" for the OR-function and "0" for the AND-function, etc.

	0	1	D	\bar{D}
0	0	0	0	0
1	0	1	D	\bar{D}
D	0	D	D	0
\bar{D}	0	\bar{D}	0	\bar{D}

AND - Function

Fig. 2b

Using the D-Algorithm one should identify two special types of inputs to a logic block.

The first type consists of those inputs which cause the output of the block to be different (assuming single output blocks) from its normal value and these inputs are called PRIMITIVE D-Cubes of the fault.

And in case the fault propagates to the output, i.e., assuming that the output depends on more than one input, the inputs are called PROPAGATION D-Cubes of a block.

For example, if the output lead of a NOR gate is s-a-1, then inputs of 0 to both inputs will cause a 1 output instead of 0. This can be represented by the following D-Cube of the fault:



Fig. 3---D-Cube Fault

Diagnosing Tree

The diagnosing tree is a direct graph whose nodes are tests. The outgoing branches from a node represent the different outcomes of the

particular test. For a single output circuit, there will be only two branches:

- (a) Success of the Test
- (b) Failure of the Test

And the branches will be labelled so. The diagnosing tree is continued until there is at most one fault associated with each branch.

Example:

Let three tests T_1, T_2 and T_3 be used for distinguishing between four possible faults in a circuit.

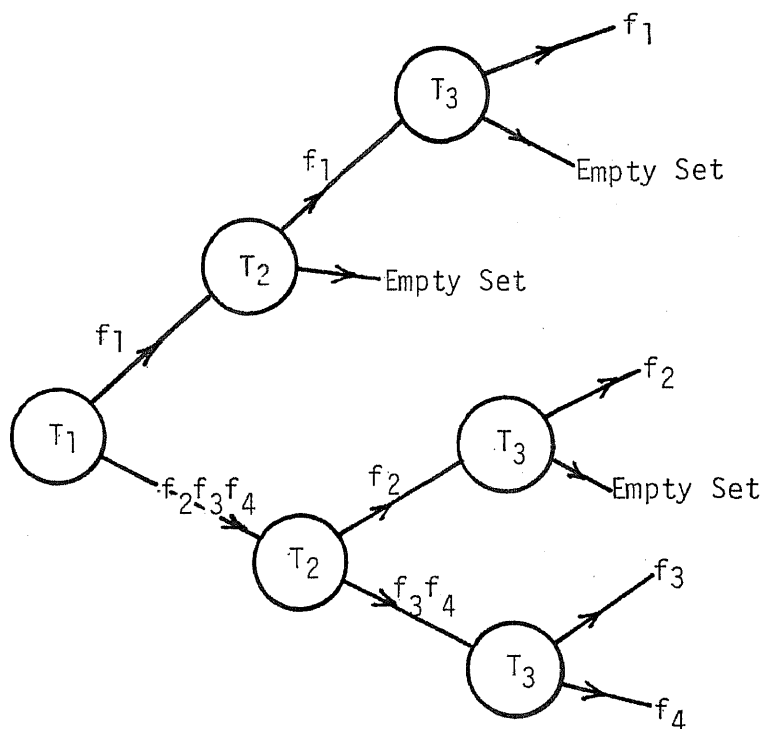


Fig. 4---Diagnosing Tree

In this example f_1, f_2, f_3 and f_4 are four faults picked up with test T_1, T_2 and T_3 . Note that MINIMIZATION can be affected in this method. Test T_1 and T_2 both detect fault f_1 , therefore T_1 test is redundant. And, therefore, the

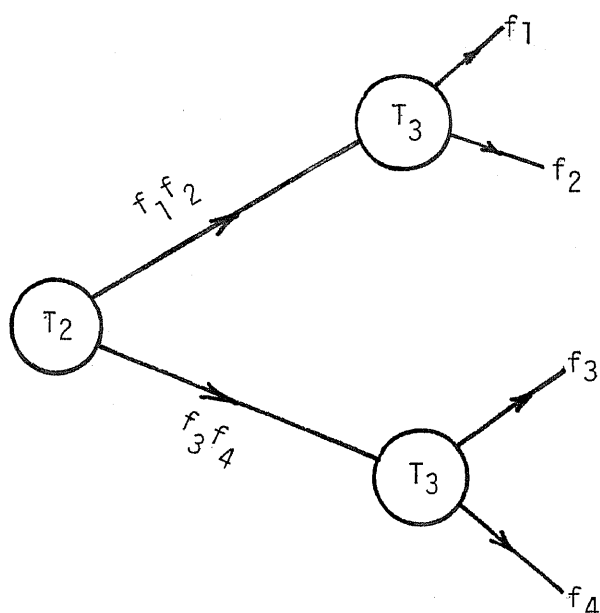


Fig. 5---Reduced Tree

Brief Survey of Different Methods Used for Test Derivation for Sequential Circuits

In a sequential circuit the outputs are dependent not only on its inputs but also on its INTERNAL STATE. The internal state is represented by the combination of signals on feedback leads. The derivation of fault-detection and diagnostic tests for sequential circuits is complicated by the fact that the state signals are usually neither observable nor directly controllable. Two approaches are normally useful:

1. Circuit---Testing Approach

The normal and faulty machines are assumed to operate in a synchronous manner, i.e., there are no races or hazards

Example:

Let M_0 represent the Flow Table for a normal circuit and M_1 of the faulty circuit where one of the leads a is stuck at 1. Now let a test sequence T be applied to M_0 . We get four different output sequences depending on the initial state. And the same test when applied to M_1 will lead to different outputs.

If T starts with 0, the first output will be 0 for all sequences under consideration except M_0 initial state D. Thus this sequence is distinguished from all the M_1 sequences. If the second input is a 1, the output will be 0 for all sequences except M_0 , initial state C. It remains to distinguish M_0 initial state A or B from the four M_1 sequences. If the next two inputs are 10, only the sequence generated by M_0 with initial state B produces an 11 output. Finally, if the fifth input is 1, the sequence from M_0 initial state A produces a 1 output while all of the M_1 sequences produce an 0 output. Thus the test sequence $T = 01101$ detects a s-a-1.

The eight responses to T are shown below:

M_{CHINE}	Initial State	Response to $T=01101$
M_0	A	00001
	B	00110
	C	01000
	D	10000

(a)

M_{CHINE}	Initial State	Response to $T=01101$
M_1	A	00000
	B	00100
	C	00100
	D	00000

(b)

Fig. 7---Detection of s-a-1 Fault

Successor Tree

A minimal length sequence can be found by exhaustively examining all sequences of length 1,2,3 until we find a sequence which detects all faults.

This can be done by constructing a Successor Tree as shown on the right-hand side.

Machines M_0 and M_1 are both started in state A.

X indicates that the output of the faulty machine is no longer of interest since it was detected prior to this state. From the Successor Tree it is seen that only input sequence of length four can distinguish M_1 from M_0 when they start at state A with either sequence 1110 and 1101, being suitable.

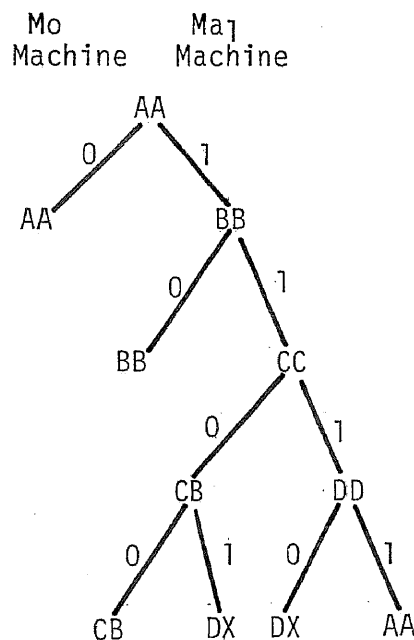


Fig. 8

Successor Tree Fault

Limitation of Conventional Techniques

All the methods described so far in this chapter are classical methods and there are some large machines and test facilities which do use these techniques. However, to use these techniques for a very complex L.S.I. (Large Scale Integration) is not practical. As packaging density increases fewer test points become available and the data streams at the available test points become complex and such techniques will not lead to a quick resolution of the problem. All the classical methods have at least one shortcoming. Some do not test a realistic set of input conditions, while others perform well at detecting logical errors and stuck nodes but fail to detect timing related problems. Therefore, for a complex data stream, "compression" of data becomes essential.

One method for compression of data for a "multiple - bit burst" into a form that can be handled easily by a microprocessor-based tester, is based

on "transition counting." The other method borrowed from the telecommunication field is the CRC (cyclic redundancy check code) a sort of checksum produced by a pseudorandom binary sequence (PRBS) generator.

The CRC method has the main advantage that the tester does not have to know how the particular digital circuit works; only the designer of the test itself need know the details. In classical systems the troubleshooter has to know the circuit details before he can attempt to fix the fault.

CHAPTER 2

SIGNATURE ANALYSIS

Using a known input signal, a Signature Analysis System generates a unique coded presentation at each point in a digital circuit. These responses are noted for a circuit which has no defects. If and when the circuit becomes defective, these points are again checked and defective nodes isolated. The resolution of this technique is excellent and can lead to a chip-level fault detection.

The difference between signature analysis and other techniques, such as transition counting, lies in the "coding" technique. The signature of a data stream is a 4-digit display of what is left in a 16 bit serial-in, parallel-out shift register (one with a specific feedback path) after the data stream has been clocked through the register. This signature is unique to the data stream, regardless of its length. Any change in the stream, so long as it continues past one clock edge, changes the signature. Therefore, in short, this is the essence of signature analysis.

The remainder of this chapter will be dedicated to the theoretical aspects of signature analysis.

Signature Analysis Formal Definition

In short signature analysis is a technique that uses cyclic redundancy check (CRC) code, a cyclic hashsum, produced by pseudorandom binary sequence (PRBS) generator.

PRBS Sequence

A pseudorandom binary sequence is a pattern of binary ones and zeros

that appear to be random, but has a very long period. After some sequential length the pattern repeats itself. The random-like sequence is statistical in nature but is predictable.

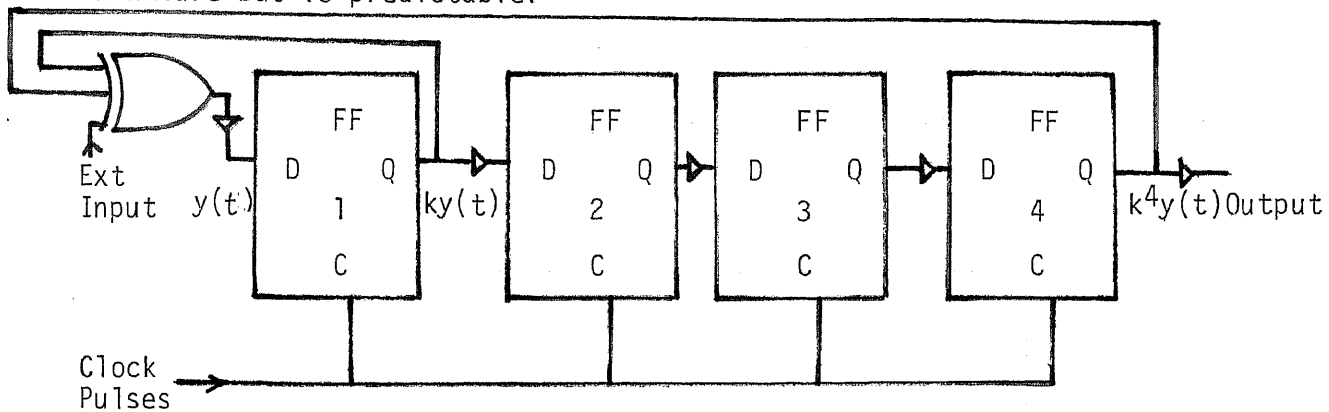


Fig. 9---PRBS Generator

A PRBS generator can be constructed by an exclusive - OR gate or a Modulo-2 adder, and Memory Elements such as D-type flip-flops. The flip-flops are connected in cascade to form a shift register. The output of these flip-flops is then fed back through the exclusive - OR gate to the Shift Register. This arrangement will then produce a pseudorandom binary sequence. By properly choosing feedback taps, maximum length linear sequences can be produced.

A shift register may be described using a transform K operator, multiplying by which is equivalent to delaying data by one unit of time. Redefining the delayed data by:

$$k^n y(t) = x^n$$

One has the feedback equation of Fig. 9 as:

$$k^4 y(t) \oplus ky(t) \oplus y(t) = 0$$

which may be written as:

$$x^4 \oplus x \oplus 1 = 0$$

Since the feedback taps were implemented at delays of four clock cycles (F/F 4) and one clock cycle (F/F 1).

Now when the feedback shift register shown in Fig. 9 is provided with an external input, data can be overlaid on the PRBS generated by the circuit. Feeding data into a PRBS generator is the same as dividing the data by the characteristic polynomial of the generator.

Error Detection by PRBS Generator

Now different input sequences fed to the same PRBS generator will produce very different output sequences even though the input sequences may differ only by one bit. If the generators are now stopped at any time and the patterns remaining in the flip-flops are compared, they will also show different states. These remainder patterns are called "Signatures." This shows the effect of an error sequence when added to a data stream even when the error occurs only once in a long measurement window.

Therefore we have now found a simple "data compression algorithm" which could be used to detect errors in a synchronous digital circuit.

Tap Selection of PRBS Generator

The Signature for this project consists of a four-character display for a 16-bit register. The tap selection for the feedback can be accomplished $2^{16}/2 \cdot 2^4$ or 2048 different ways. We will be using a feedback equation of:

$$T(X) = X^{16} + X^{12} + X^9 + X^7 + 1$$

The polynomial scatters missed errors as much as possible. This arrangement also avoids evenly spaced taps at four or eight bits apart.

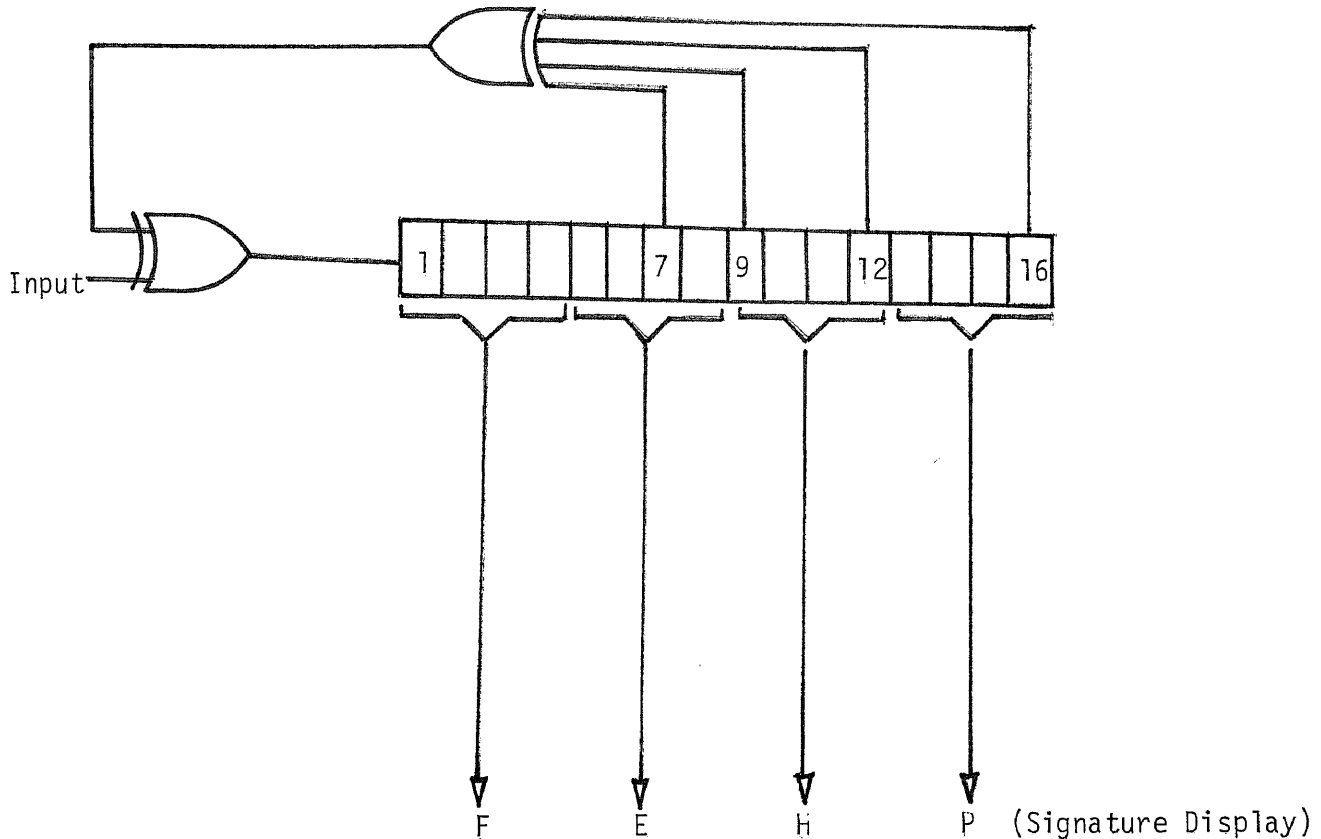


Fig. 10

Selection of TAP for $V(X) = X^{16} + X^{12} + X^9 + X^7 + 1$

Pseudorandom Binary Sequence (PRBS) Generator

As discussed before the PRBS generator for this project is nothing more than a 16-bit Shift Register with feedback taps at positions 7, 9, 12 and 16 of the Register Bits. An exclusive - OR gate is inserted between the input Bit Stream and these taps. At the end of the Bit Stream the Shift Register is frozen and the Signature read.

The following figure illustrates the Shift Register arrangement along with an example of the Signature obtained for a known Bit Steam.

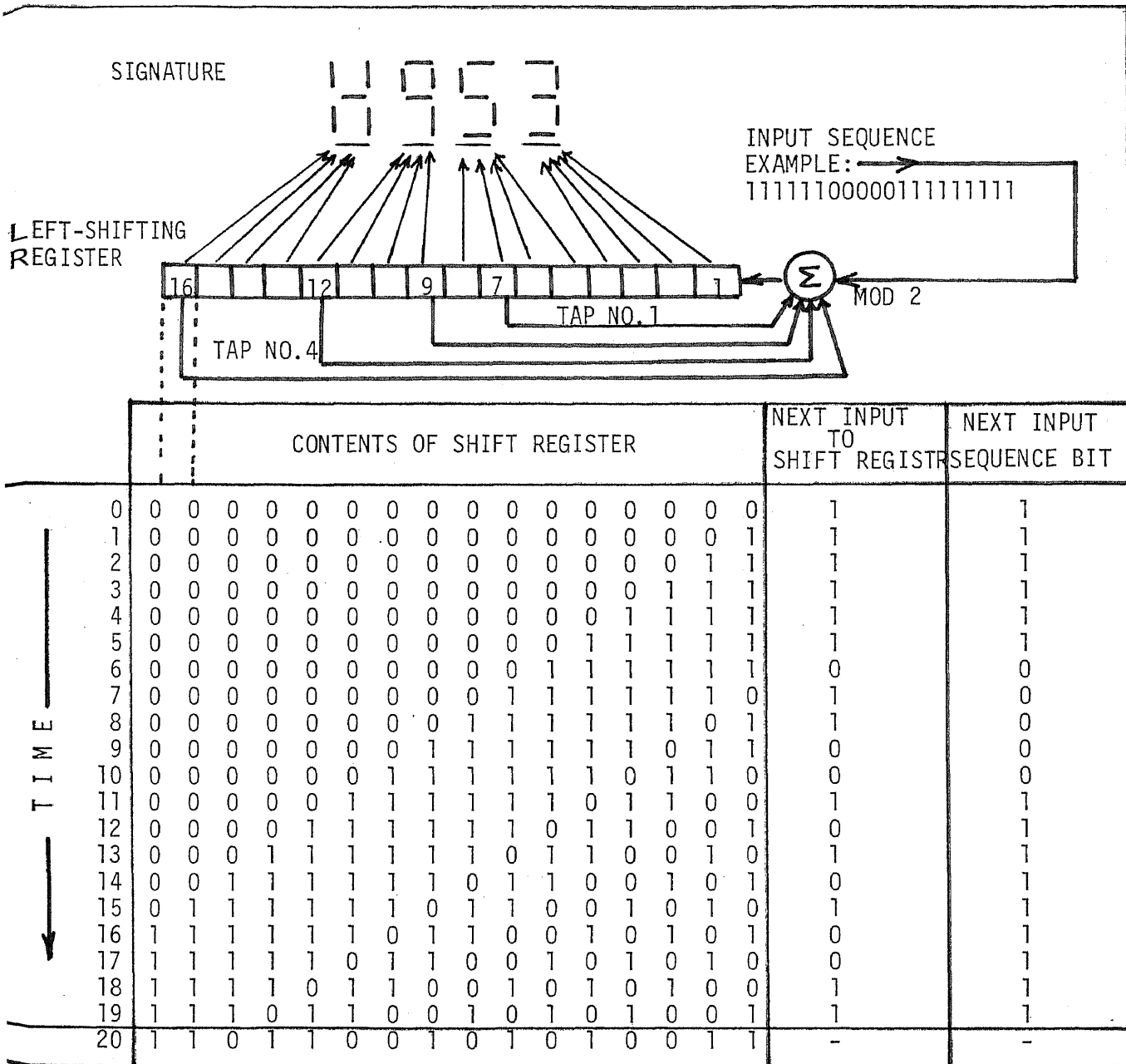


Fig. 11

20-Bit Sequence Generating a Signature

Figure 11 shows how the Register generates a Signature from a 20-bit sequence 11111100000111111111. Initially (time 0 through 7) the Register acts merely as a Shift Register. At time 7, the first 1 of the input sequence has reached the first feedback tap (tap 1). It is fed back and mixed with the input 0, with the result that a "1," not a "0," is next clocked into the Register (time 8). This behavior continues until the end of the measurement. At time 20 a residue of 16 bits 1101100101010011 is all that is left from the 20-bit sequence. This residue is the Signature and will be displayed as H953 in a hexadecimal format using (0123456789ACFHPU) arrangement since these characters are easy to display on 7-segment displays rather than normal hexadecimal characters which will require alphanumeric display.

Probability of Error Detection in Signature Analysis

Assume Y is a data stream of n bits. Let S be a K -bit, PRBS generator and S^{-1} be its inverse (i.e., $SS^{-1} = 1$). Let Q be a quotient and R the remainder.

$$\text{Then } S(Y) = Q(Y) 2^K + R(Y) \quad (1)$$

Now assume another n -bit sequence Z that is not the same as Y and therefore differs by a n -bit Error Sequence E .

$$\text{Then } Z = Y + E$$

$$\text{Also } S(Z) = Q(Z) \cdot 2^K + R(Z)$$

$$\text{So } S(Y + E) = Q(Y + E) \cdot 2^n + R(Y + E)$$

Since operators are linear, we can write

$$S(Y) + S(E) = Q(Y) \cdot 2^n + Q(E) \cdot 2^n + R(Y) + R(E) \quad (2)$$

Subtracting (2) from (1) one obtains

$$S(E) = Q(E) \cdot 2^K + R(E)$$

For undetectable errors $R(E) = 0$

$$\text{or } S(E) = Q(E) \cdot 2^K$$

$$\text{or } E = S^{-1} Q(E) \cdot 2^K \quad (3)$$

Since Y , Z and E are all n -bit sequences, then $Q \cdot 2^K$ must be an m -bit sequence containing K final zeros. Q therefore contains $(m-K)$ bits. Hence there are 2^{m-K} sequences that map into the same residue as the correct sequence and there are $2^{m-K} - 1$ error sequences that are undetectable because they leave the same residue as the correct sequence. 2^K sequences can be generated using m -bits and only one of these is correct, therefore the probability of failing to detect an error by PRBS is

$$P(\text{PRBS}) = \frac{\text{Undetectable Errors}}{\text{Total Errors}}$$

or

$$P(\text{PRBS}) = \frac{2^{m-K} - 1}{2^m - 1}$$

for $m \longrightarrow$ Large

$$P(\text{PRBS}) \approx 1/2^K$$

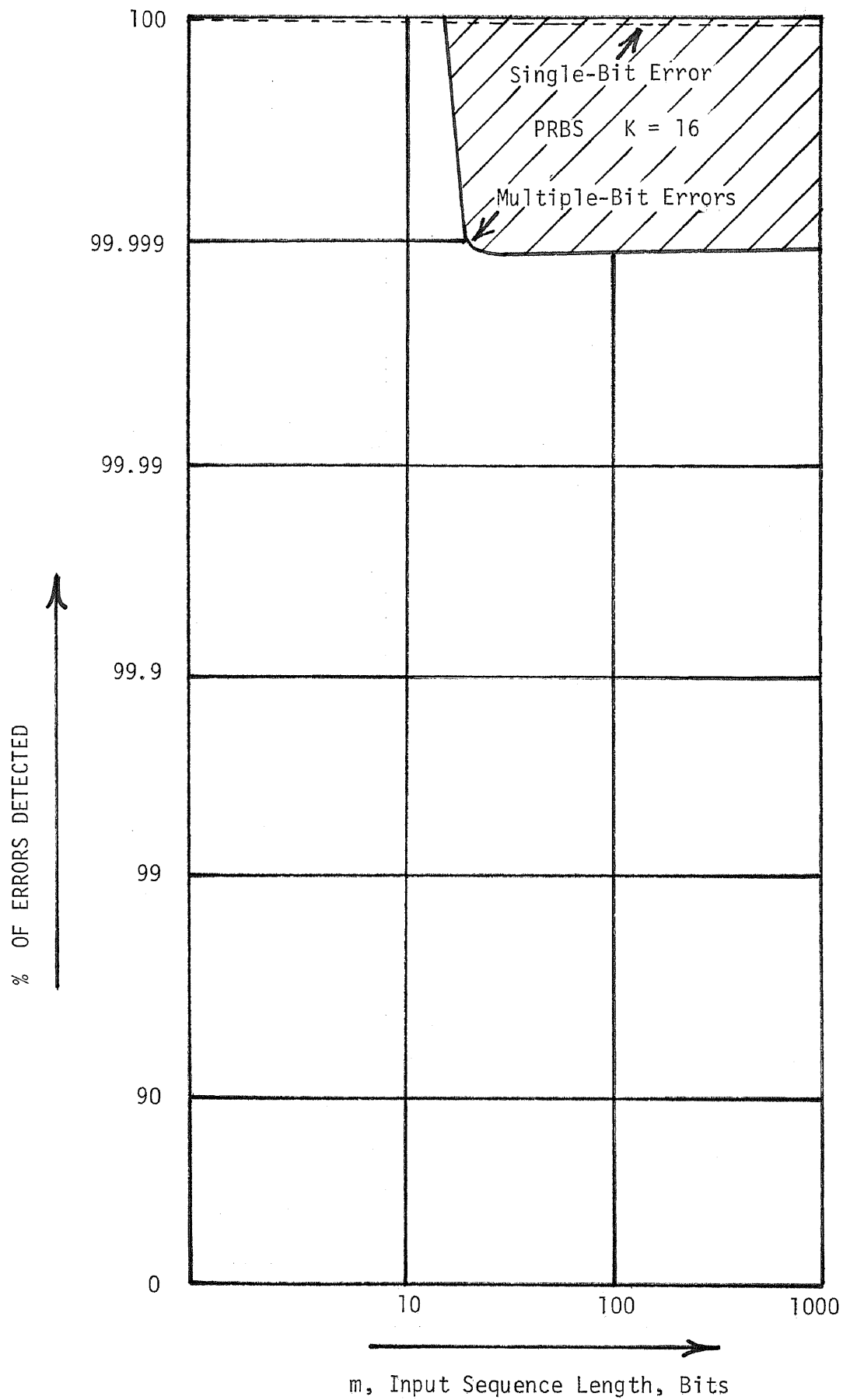


Fig. 12

Probability of Detection of Errors for Signature Analysis for K = 16

CHAPTER 3

DESCRIPTION OF A SIGNATURE ANALYSIS SYSTEM

The key object of this project is to design a Signature Analysis System which could collect unique Signatures at different nodes of a circuit under test. The circuit under test is first subjected to a stimulus. The response of the circuit is then collected as signatures.

This chapter will deal with the development of the system and discuss special hardware and software requirements for the Signature Analysis System.

The Signature Analysis System

The Signature Analysis System will be based on a Motorola 6800 Microprocessor System. The Microprocessor has been selected over other popular Microprocessors because of the ease by which this Microprocessor can be I/O (Input-Output) interfaced. The memory of the 6800 system is I/O mapped. This Microprocessor also has an excellent mini-computer type instruction set and many addressing modes.

Description of the Proposed System

The basic system diagram for the Diagnostic System is shown in Fig. 13. The pattern is generated in (a) and is fed to the system under test (d) via the microprocessor (b) and hardware interface (c). The system response is shifted through the PRBS Generator (f) via the hardware interface (e). The system "Signature" is then stored in the memory. The "good" Signature is then compared with the Signature of a faulty system. The Operating System Program controls the operation described above and other system I/O functions. The Signature information is "backed up" in the Floppy Disk

storage device (h). Teletype and KCRT (g) are used to control the system and communicate with the microprocessor.

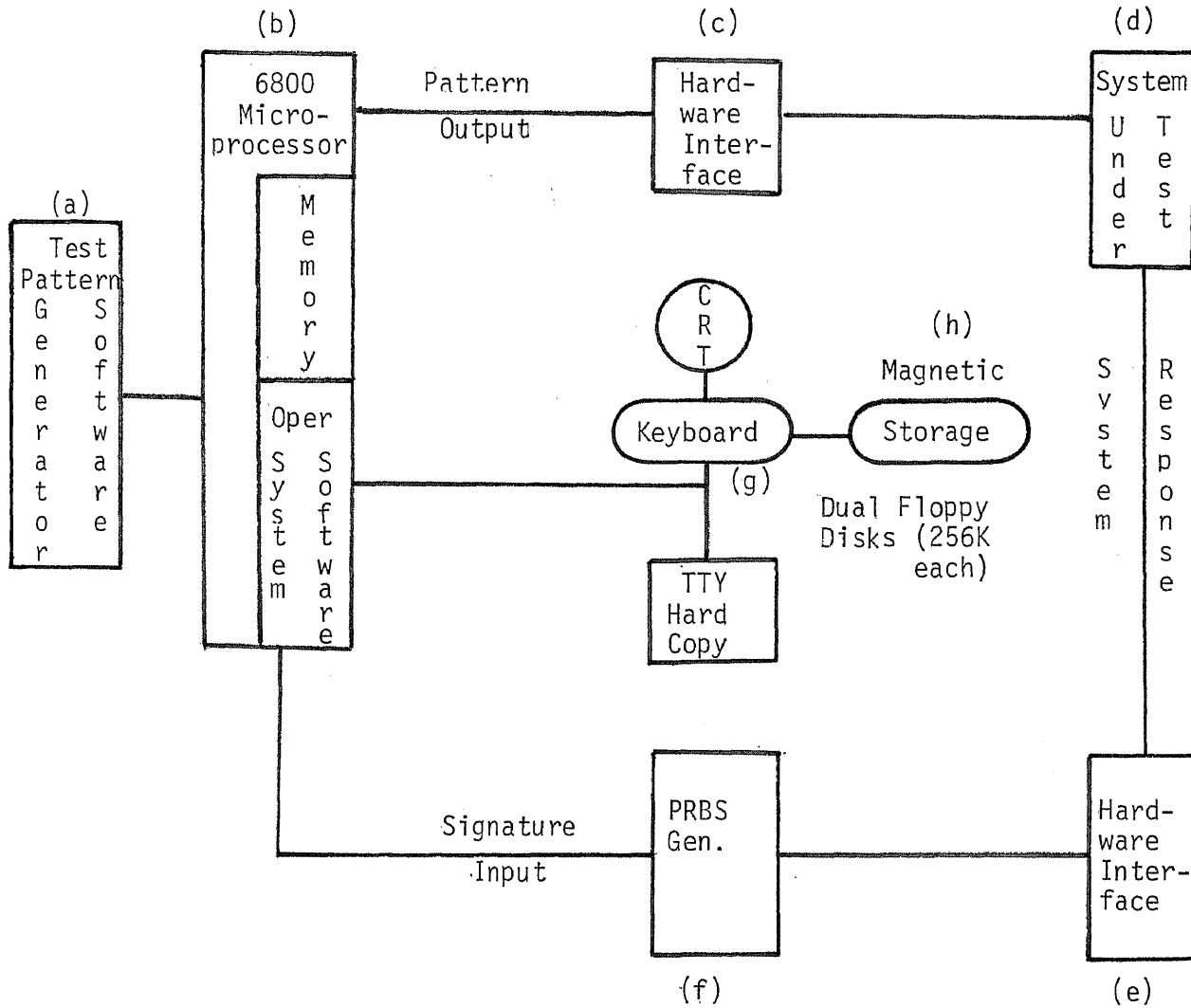


Fig. 13---The Signature Analysis System

The microcomputer system which will be used for this project has the following basic hardware and software available:

Hardware

1. 6800 Microprocessor
2. 48K Static Memory

Hardware (cont'd)

3. One Serial Input/Output port equipped with:
 - (a) 1200 Baud CRT with keyboard
 - (b) 110 Baud Teletype unit
 - (c) Dual 8" Floppy Disk Storage
4. One parallel input/output port to be used for the pattern generator input and system response output.
5. Expansion possible to 60K of memory and I/O ports can be expanded to a total of eight ports.

Software

1. Disk Operating System (FLEX 2.0)
2. Disk Editor
3. Disk Assembler

It should be noted that the hardware and software mentioned above is for the "Development System" which was used to develop final hardware and software for the Signature Analysis System. The special hardware and software is discussed next.

Description of Additional Hardware and Software Required for the Signature Analysis System

As shown in Fig. 13 the Signature Analysis System required the development of the following hardware and software:

Hardware

1. A Diagnostic Bus System which will connect microcomputer to the circuit under test. This Bus System will also be used to send stimulus to the circuit under test.
2. A Shift Register with proper taps to generate maximum length PRBS sequence and combine the input from a circuit under test forming various signatures.

Software

1. A Pattern or Stimulus Output Program which sends known pattern to the circuit under test.
2. Signature Input Program which corrects Signature for various nodes of the circuit under test.
3. Signature Comparison Program which compares bad Signatures with good Signatures and detects faulty nodes.

Thus far we have only discussed very briefly the requirements for the Signature Analysis System. Now we will proceed with details of hardware and software which was developed for this project.

CHAPTER 4

DIAGNOSTIC BUS INTERFACE

This part of the project is related to the software and hardware of the interface circuitry. The main component of the interface circuitry is called the Peripheral Interface Adapter and in the Motorola nomenclature it is designated MC6820.

Peripheral Interface Adapter

The MC6820 Peripheral Interface Adapter (PIA) provides a flexible method of connecting Byte-oriented peripherals to the MPU.

An input/output diagram of the MC6820 is shown below:

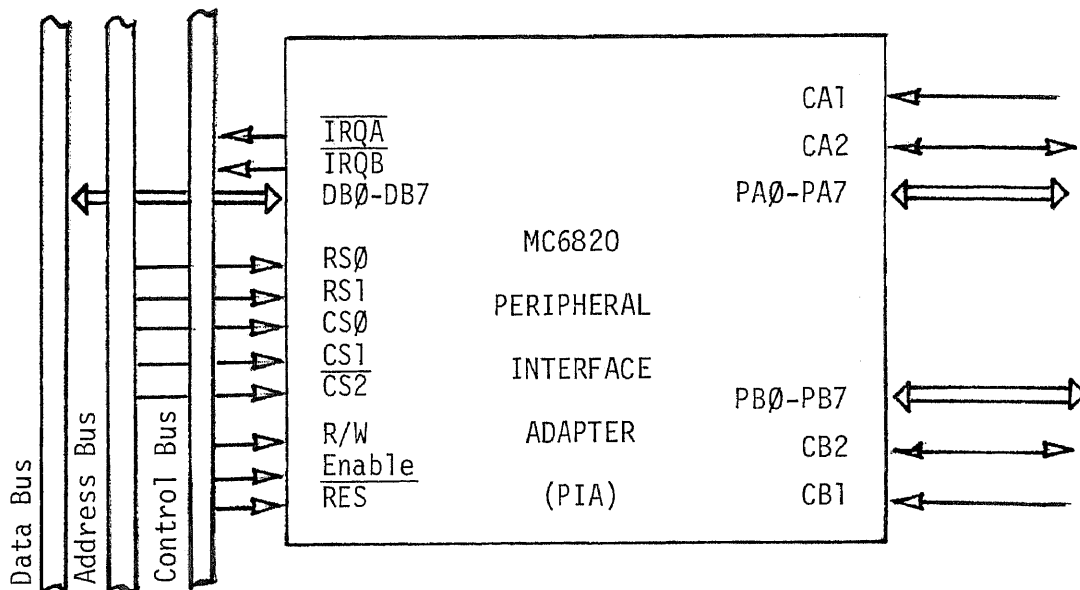


Fig. 14

MC6820 PIA I/O Diagram

The data flows between the MPU and the PIA on the System Data Base via eight bidirectional data lines, D0 through D7. The direction of data

flow is controlled by the MPU via read/write input to the PIA. The MPU side of the PIA also includes three chip select lines, CS0, CS1 and CS2 for selecting a particular PIA. Two addressing inputs, RS0 and RS1, are used in conjunction with a control bit within the PIA for selecting specific registers in the PIA. The MPU can read or write into the PIA's internal registers by addressing the PIA via the System Address Bus. Using these five input lines and the R/W signal from the MPU's point of view, each PIA is simply four Memory locations that are treated in the same manner as any other read/write Memory.

The MPU also provides a timing signal to the PIA via the Enable input. The Enable (E) pulse is used to condition the PIA's internal Interrupt control circuitry and for the timing of Peripheral control signals. Since all data transfers take place during the ϕ_2 portion of the clock cycle, the Enable pulse is normally ϕ_2 .

The peripheral side of the PIA includes two 8-bit bidirectional data buses (PA0-PA7 and PB0-PB7) and four interrupt/control lines; CA1, CA2, CB1 and CB2. All of the lines on the peripheral side of the PIA are compatible with standard TTL logic. In addition, all lines serving as outputs on the B-side of each PIA (PB0-PB7, CB1, CB2) will supply up to one milliamp of drive current at 15 volts.

Internal Organization

The MC6820 Peripheral Interface Adapter (PIA) provides 16 I/O pins which may be grouped into I/O ports as shown on the following page:

Internal Registers

The internal registers of the PIA is shown below:

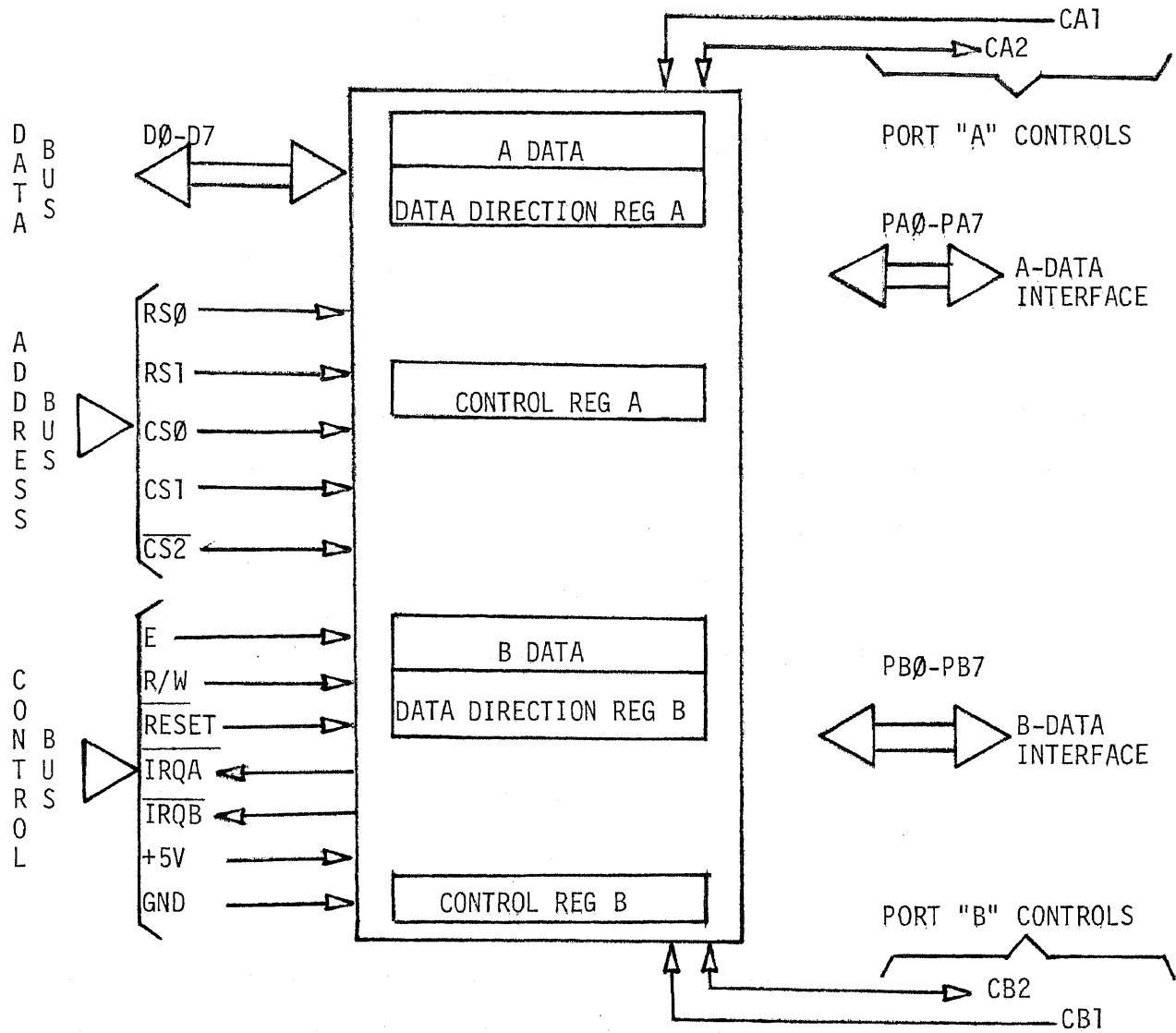


Fig.16---Internal Registers of PIA

The PIA has six 8-bit registers. These are organized in two nearly identical groups, designated the "A" side and the "B" side.

Let us consider the "A" side of the PIA.

The three registers are designated:

DDRA - Data Direction Register, A-Side

ORA - Output Register, A-Side

CRA - Control Register, A-Side

Similarly the B-Side Registers are designated DDRB, ORB and CRB.

The A-Side outputs are TTL compatible while the B-Side outputs are Tri-State Buffered and Can Sink 1 Ma at 1.5 volts.

Data Direction Register

The Data Direction Register assigns the direction of each of the bidirectional lines of the 8-bit bus to the outside world. If the given line is to be used for input, the corresponding bit must be set to a "0." And if the line is to be used for output the bit is set to a "1." This setup is likely to be done only once at the beginning of a program.

Output Register

The Output Register has 8-bit positions which correspond to the eight peripheral data lines. The program can place a byte in the Output Register. When it does, those peripheral data lines which are conditioned for output by the DDR will be set to levels corresponding to the bit pattern in the Output Register. Note that the DDR must have a "1" in the proper bit position to condition a peripheral data line for output. A bit value of "1" in the Output Register will then cause that line to be high, while a "0" will cause it to be low. Note further that a "0" bit in the DDR conditions a line to be input and the bit in the Output Register has no effect.

When the DDR has a line conditioned for input, a high level on the line is interpreted as a "1." This "1" is sent to the Buffer for use by the MPU. In this system we refer to both input and output as the Output Register and it is accessed by the same address.

Control Register

The Control Register controls two Peripheral Control lines and some internal functions of the PIA. Each of the eight bits in the register is assigned a special meaning and these are generally set to fixed values at the beginning of the program. During execution of a program some of these bits may be changed or examined. It should be noted that the changing of a single bit in this register can only be accomplished by sending a full byte to it. Even so, certain bits of this Control Register cannot be changed by writing into this register.

Let us now examine how the individual bits of each Control Register functions:

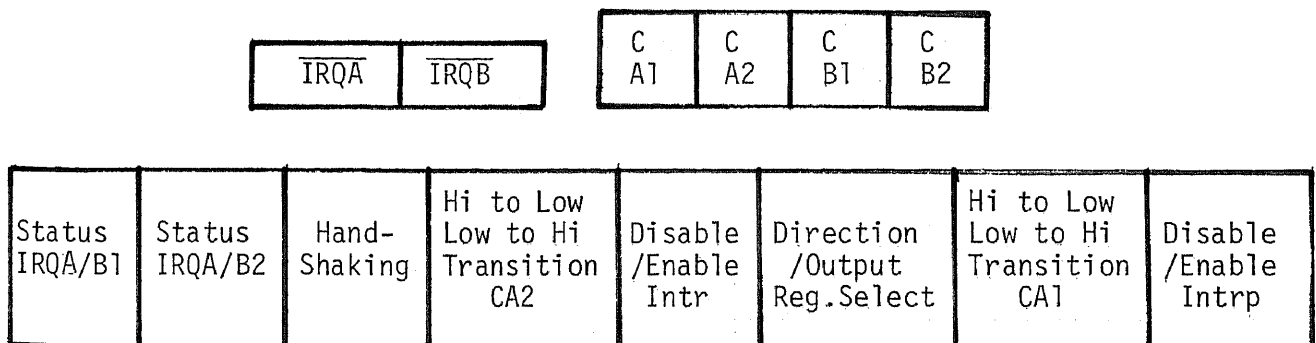


Fig. 17---Control Register Interpretation

Interrupt Trigger

Control Register bit "0" enables or disables \overline{IRQA} and \overline{IRQB} based on signal CA1 and CB1 transition respectively.

Control Register bit "3" enables or disables IRQA and IRQB based on CA2 and CB2 transition respectively. And thus interpretation is only true when Bit 5 = 0.

Transition Effect

The active transitions of Control Signals may be high to low or low to high. For CA1 and CB1 the active transition is selected by the Control Register bit 1. For CA2 and CB2 the active transition is selected by Control Register Bit 4, but only if Control Bit 5 = 0.

Status of Interrupts

Irrespective of whether interrupt request signals \overline{IRQA} and \overline{IRQB} have been enabled or disabled Control Register Bits 6 and 7 will report the interrupt request as a Status. Control Bits 6 and 7 can only be reset to 0 when a Read Operation is performed at the Control Register address.

Handshaking

If Control Register Bit 5 = 1, then the Control Register Bits 4 and 3 take on a second interpretation. If Control Register Bit 5 and 4 are both 1, then Control Signal CA2 and CB2 will output at all times with the level of "Control Register Bit 3."

Automatic Handshaking

If Control Register Bits 5 and 4 are 1 and 0, then Control Register Bit 3 specifies an automatic handshaking signal sequence.

Interfacing the Diagnostic Interface to the Signature Analysis System

The test pattern is generated via the software program and is sent to the system under test. The path to the system under test is selected through the I/O Decoder circuit (Fig. 18). The Parallel Interface under software control then directs the output pattern to the system under test using Port B.

On receiving the test pattern the system under test now generates an output. The Port A is now activated via software and after some appropriate delay the test result is channeled to the microcomputer.

In this way a complete closed loop diagnostic bus is created and used for fault detection. If more inputs and outputs are desired up to seven more such interfaces can be added to the present system, adding 56 more input points and 56 output points. For even larger circuits the decoder circuit will require expansion.

Timing Requirements

There is no direct timing relationship between the microcomputer clock and the circuit under test, since the clock for the signature analyzer is provided by the circuit under test alone. The microcomputer software need only assure a setup time for 35ns for the signature analyzer hardware to generate a stable signature.

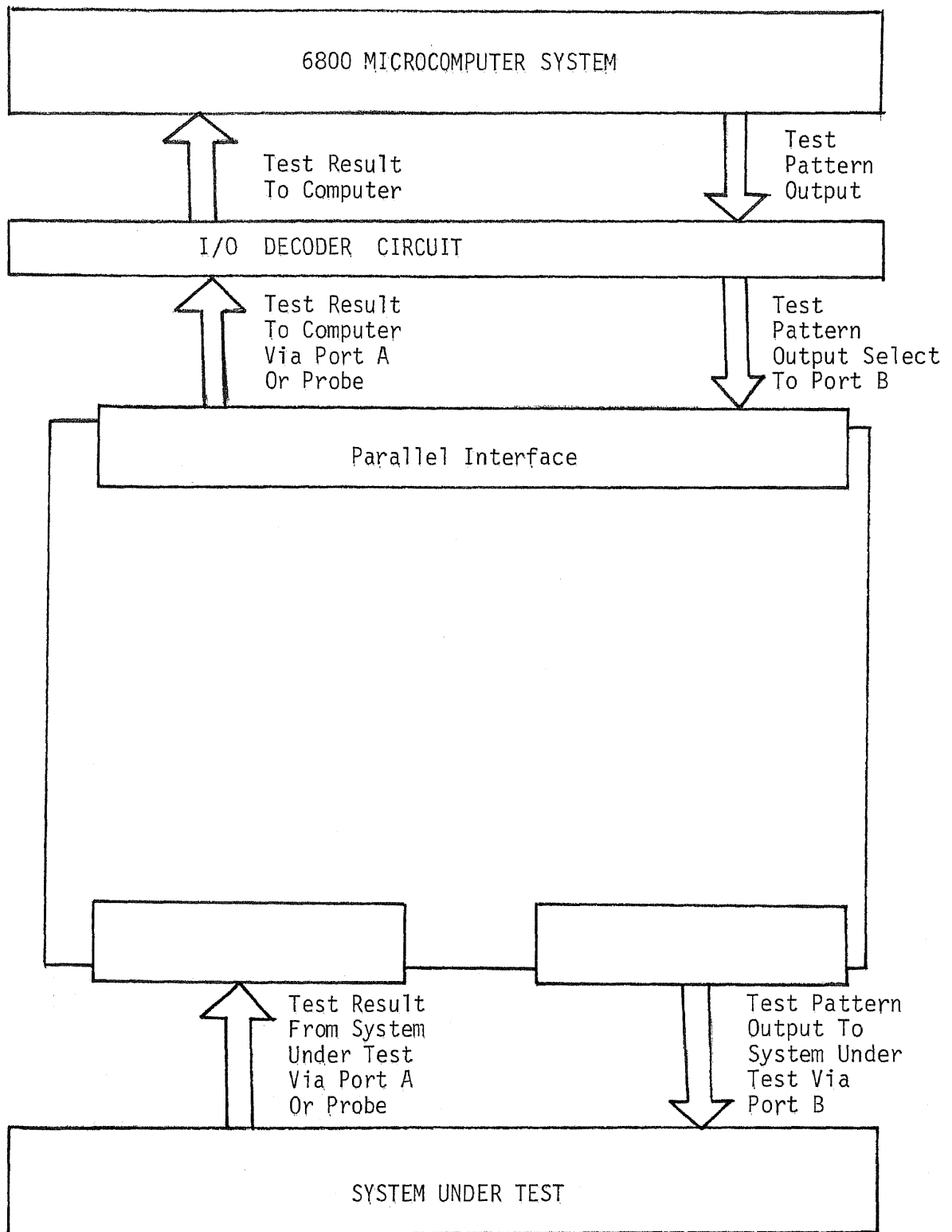


Fig. 18

Diagnostic Interface Block Diagram

CHAPTER 5

SIGNATURE ANALYSIS MODULE

The Signature Analysis Module selected for this project was manufactured by Pheonix Digital. The organization of this module is shown below:

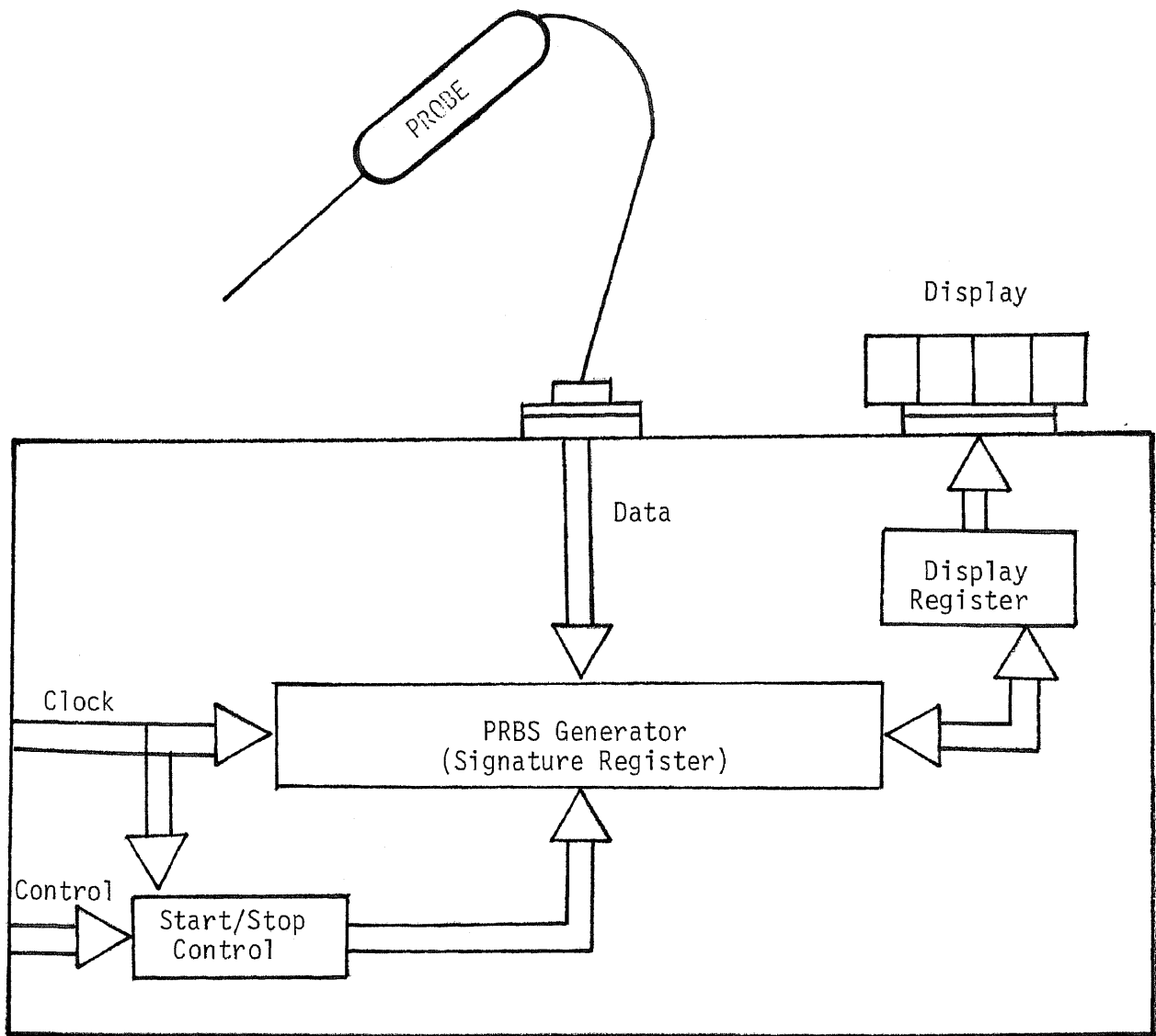


Fig. 19---Signature Analysis Module

This module was supplied for use on a S-100 microcomputer bus system. Since we are using a Motorola 6800 Microcomputer based on SS-50 bus of Southwest Technical, a bus conversion interface card had to be designed which would convert SS-50 signals to S-100 signals.

As is evident from Fig. 19, the Signature Analysis Module requires the following signals:

1. Start Control
2. Stop Control
3. Clock Signal

This is the basic drawback of the Signature Analysis System. A circuit under test must be able to supply these three signals. If a circuit is already designed with this in mind there is no problem, it adds a little to the cost of the circuit but is well worth it. However, if a circuit does not have these signals, Signature Analysis cannot be applied to it without major modification.

System Structure

A Block diagram of the Signature Analysis System was shown in Fig. 13. A part of that figure is repeated here to show an expanded portion of the Signature Collection System. Fig. 20 shows that the Serial Bit Stream is picked up by the probe. The bit stream is shifted through the PRBS Generator whenever the circuit under test executes a start control. Now when the circuit under test executes a stop control the PRBS

Generator is frozen and the 4-Digit Signature displayed. This information is also saved in the Memory under software control for future use and comparison.

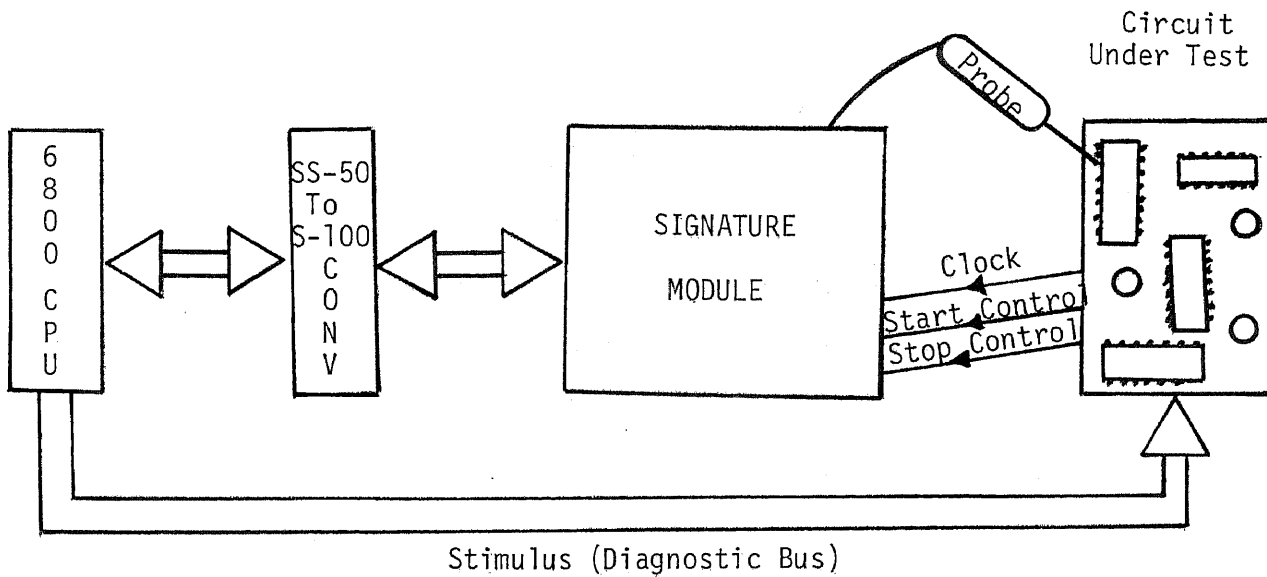


Fig. 20---Signature Collection System

Definition of Key Signals

Definition of Key Signals required for the Signature Analysis System is stated below:

Stimulus: The Signature Module does not supply the stimulus or exercising bit pattern to the circuit under test. This pattern is to be generated by the circuit under test or supplied by a host microprocessor.

Monitored Signals: These are the signals which should be monitored for Stable Signatures after the Stimulus is applied to the circuit under test. Examples of these signals are: data, address and control lines of the microcomputer bus, chip enabler for various IC's, clocks, interrupt lines, reset lines and returning data lines.

Clock: The Clock Pulse has to be generated by the circuit under test also. The Clock Signal is used to shift data through the PRBS generator and eventually causes the Signature to be generated when the Stop Signal freezes the Shift Register.

Start-Stop: As mentioned before this signal is also generated by the circuit under test. The Start Signal along with the Clock Signal starts shifting the Serial Bit Stream in the Shift Register. The Stop Signal along with the Clock Signal freezes the Shift Register and hence generates the Signature via the residue in the Shift Register.

Procedure for Obtaining a Signature

A certain protocol of events is necessary in obtaining Signatures from a circuit under test. These important steps are listed below:

1. Initialization of the Signature Analysis Module.
2. Enabling the Signature Analysis Module and waiting for the data.
3. Starting the Pattern Generator which would pump Stimulus into the circuit under test.

4. Shutting off the Signature Analysis module after a certain delay so that the Signatures would become stable.
5. Collecting, storing and displaying the Signatures for each important node of the circuit under test.

Once the above procedure is followed, each node can be identified in the circuit under test with specific Signatures. This kind of "good" Signature generation must obviously be done when the circuit under test is operating normally with no faults. If and when the circuit under test develops a fault, a new set of Signatures will be obtained for the defective nodes with the same stimulus. This "bad" signature will now lead to the defective components or conditions on the circuit under test.

Documentation of Signatures on each node for a known pattern is very important. Chapter 6 explains how this should be done. Appendix B lists the software for obtaining and controlling Signatures for a circuit under test. Appendix A contains the design of SS-50 to S-100 Bus Conversion Circuitry.

CHAPTER 6

CIRCUIT UNDER TEST

A Parallel to Serial and Serial to Parallel Bit Converter Circuit was selected for this project to serve as a circuit under test. The Universal Asynchronous Receiver Transmitter or UART chip AY-5-5013A was selected as the main element, ancilliary circuits like Baud Rate Generator, strobe and clocking circuits were then designed around the UART chip. This kind of circuit is used extensively in computer interfacing with printers, modems and other I/O devices. Basically a Parallel 8-Bit Information is strobed into the UART chip. The UART then converts the Parallel Bits of information to a Serial Bit Stream with various start and stop bits inserted before and after the bit stream. This circuit is also capable of converting Serial Bit Streams into 8-Bit Parallel outputs. The conversion speed is governed by the Baud Rate Generator.

Special Requirements for Circuit Under Test

As mentioned earlier the circuit under test must be able to supply the following three signals for it to be Signature analyzable:

1. Clock
2. Start Signal
3. Stop Signal

Clock: The output of the Baud Rate Generator of the circuit under test was used as the clock signal. This served very well since the same clock also controlled the shifting rate in the UART chip.

Start Signal: This signal though not directly provided by the circuit under test was flagable to the Signature Analysis Module. The microcomputer was programmed to send one in Bit 8 alone as a start bit and this flagged the Signature Analysis Module to start forming a Signature. This type of arrangement did not cause a problem since Bit 8 is normally used as a Parity Bit and not an Information Bit.

Stop Signal: This signal was provided directly by the circuit under test. An 8-Bit Nand gate was provided to detect all "1"s condition on the pattern generator output. Once this condition was detected, having been sent deliberately as an end of pattern code, the stop signal was then generated and the Signature frozen for display.

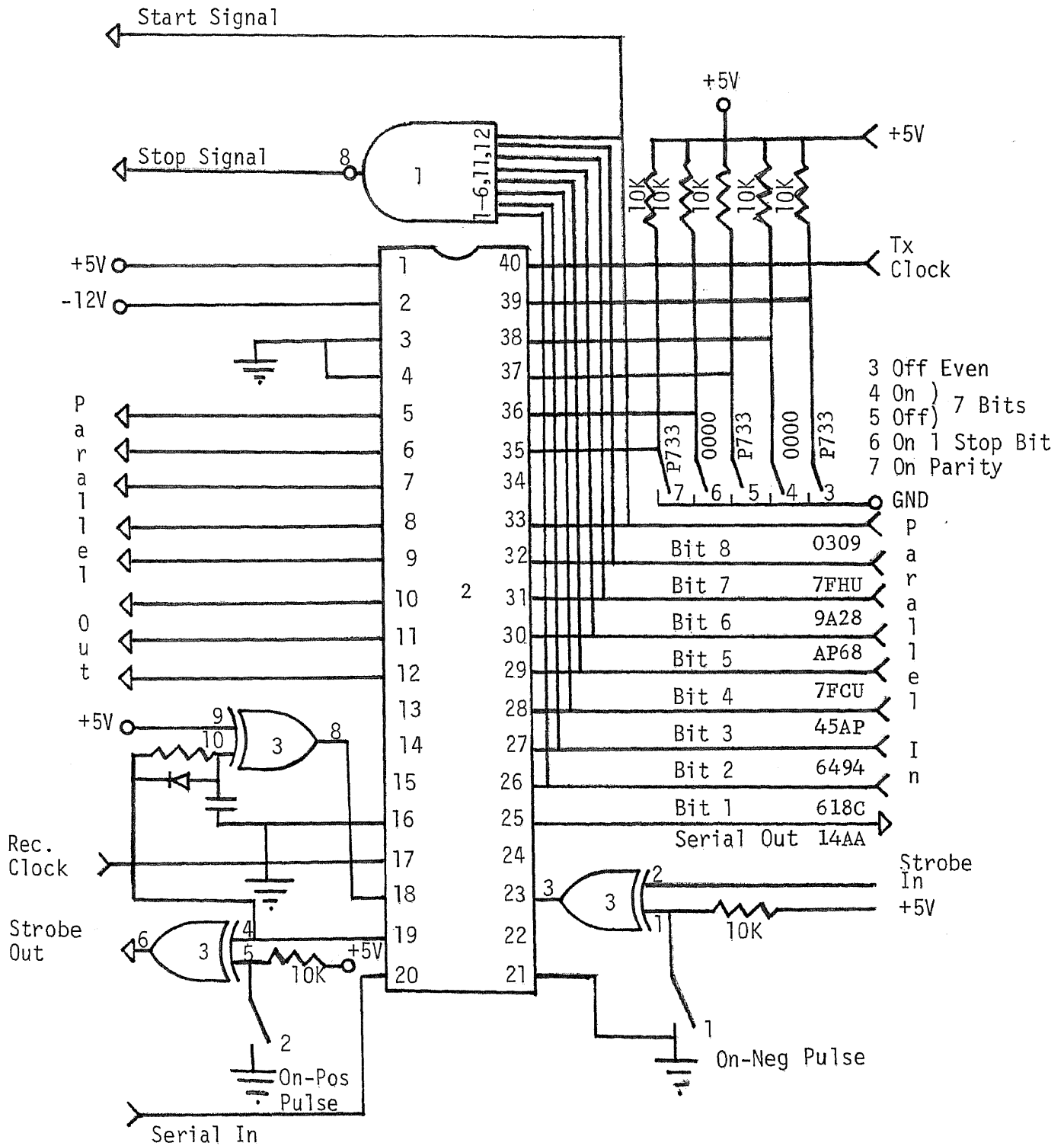
Special Schematic for the Circuit Under Test

This is the main object of this project, to show that different nodes on a Schematic can be represented by Signatures not unlike wave-form diagrams on Analog circuits. In digital circuits only uni-level square waves are visible on the Scopes. In order to troubleshoot such a digital circuit a good working knowledge of the circuit becomes necessary. With Signature Analysis, however, such a knowledge is not essential and the troubleshooter can follow Signature failures to the faulty component.

A Special Schematic diagram overlaid with Signatures is shown in Fig. 21. Each important node is identified with a 4-digit Signature. Because of race and various other conditions some nodes are not capable

of producing stable Signatures. These nodes are marked with an "*" which means they are unstable Signatures.

Such a Schematic diagram is the key to Signature Analysis. The troubleshooter follows the Signature at various points until he finds a failing Signature. He then decides which component is at fault. This procedure could easily be automated for a quicker resolution of problems.



IC1: 74LS30, IC2: AY5-1013, IC3: 74C86
 IC1 and IC3: VCC = Pin 14, GND = Pin 7.

Fig. 21
 Signature Schematic of the Circuit Under Test

CONCLUSION

The main object of this project was to show that a Signature Analysis System could be conceived using a microprocessor. This project showed all the steps necessary in developing such a system and pointed out the salient features of the system.

The developed system was capable of generating and comparing signals and finding faults in the circuit under test. The circuit under test was designed in such a way that Signature analysis could be applied to it. This has been the main drawback of the Signature Analysis System. It cannot be applied to a product which does not have circuitry to support Signature Analysis. This means the circuit under test has to supply 1) Clock, 2) Start Signal and 3) Stop Signal, which leads to another disadvantage, which is that Signature Analysis cannot be applied to circuits which are not clock driven.

On the pro side, once the circuit under test is capable of providing these three signals, Signature Analysis provides a very powerful means of troubleshooting it. The troubleshooter in this case does not require a detailed information of the circuit operation and yet is able to find defective components at chip levels.

APPENDIX A

SS-50 - S-100 BUS CONVERSION

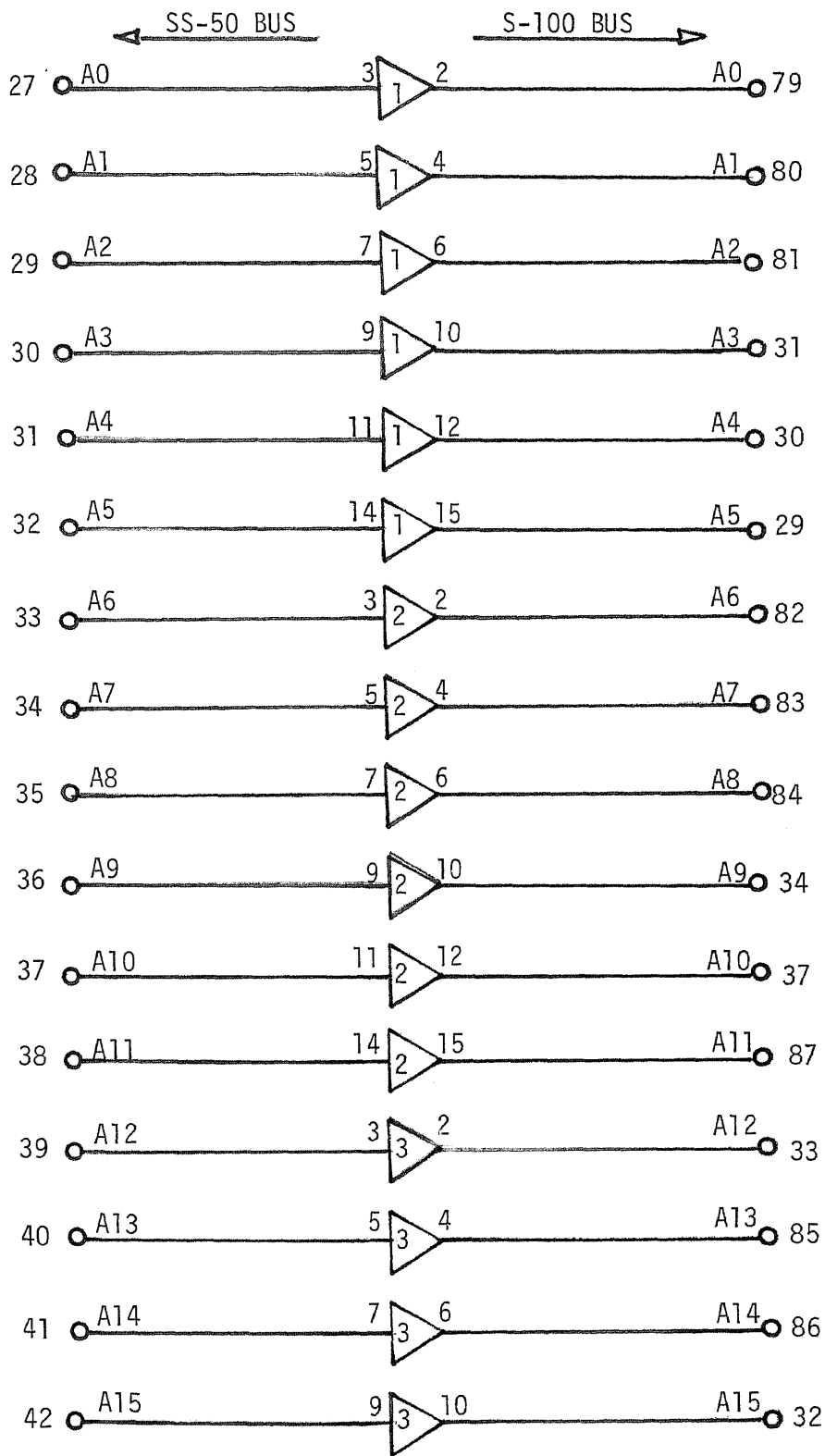
The design enclosed is for changing a 50 pin SS-50 Bus to a 100 pin S-100 Microcomputer Bus.

The Address bus lines are unaffected since they are similar in nature on both of the buses. A buffer chip was inserted between them to avoid overloading of the address bus lines. This arrangement is shown in Fig. A1.

The Data Bus lines are different. In the SS-50 bus they are bidirectional while in the S-100 bus they are unidirectional. This required a gating arrangement which would switch input/output status of Tri-State Quad bus receiver chip 8T26's via read/write control. This arrangement is shown in Fig. A2.

The other part of the conversion required development of special leads which do not exist on the SS-50 bus. The development of these leads is shown in Fig. A3. These special control leads are defined below and are essential for the S-100 bus:

SEMR	=	Status for Memory Access
Ø2	=	Master Bus Timing Signal
PWR	=	Pulsed Write Strobe (Negative True)
M-WRITE	=	Pulsed Write Strobe (Positive True)
PD BIN	=	Data-In Bus Enable
SINP	=	Status for IN instruction
SOUT	=	Status for OUT instruction



IC1, IC2, IC3: 4050

VCC = PIN 1, GND = PIN 8

Fig. A1

Address Bus Interface

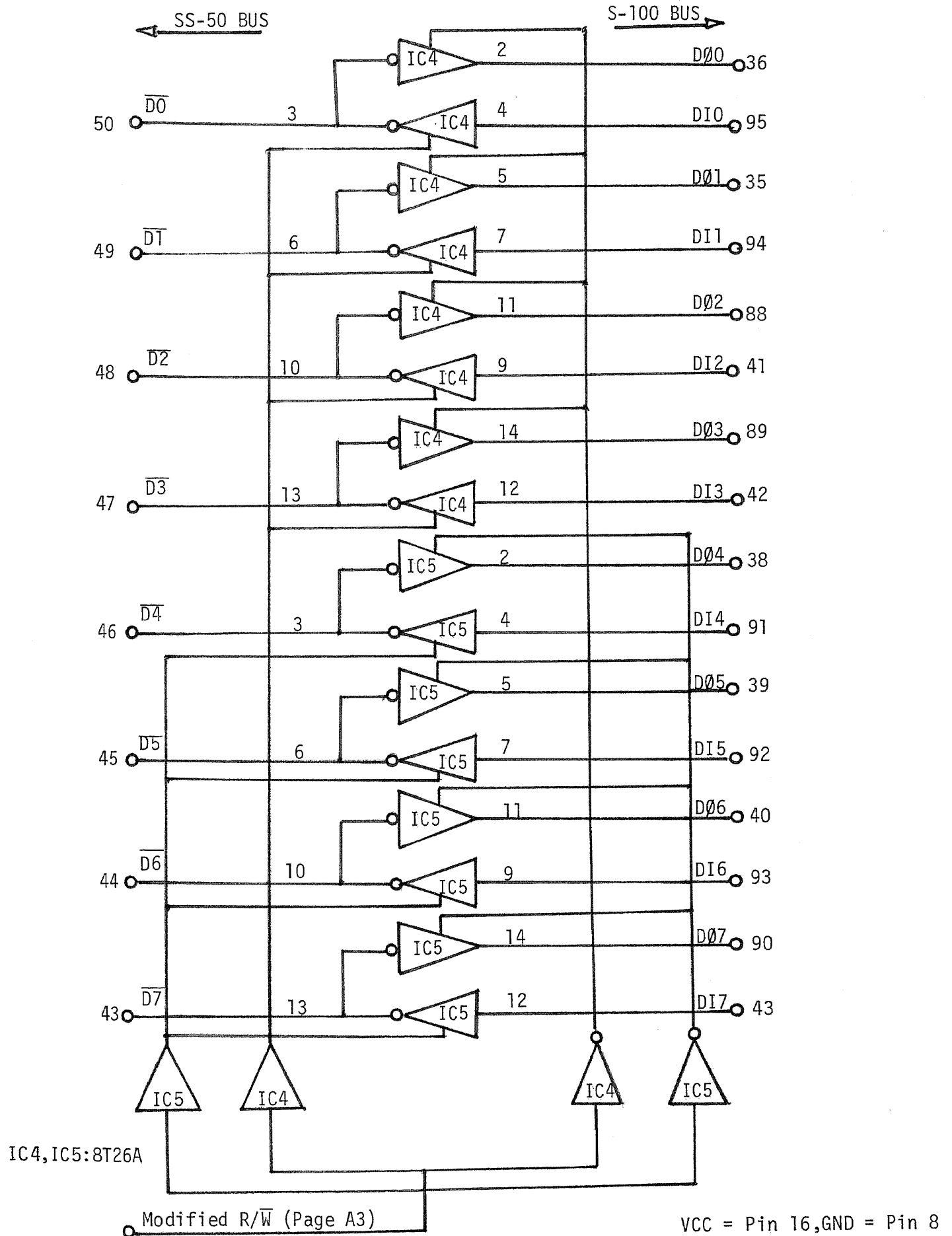
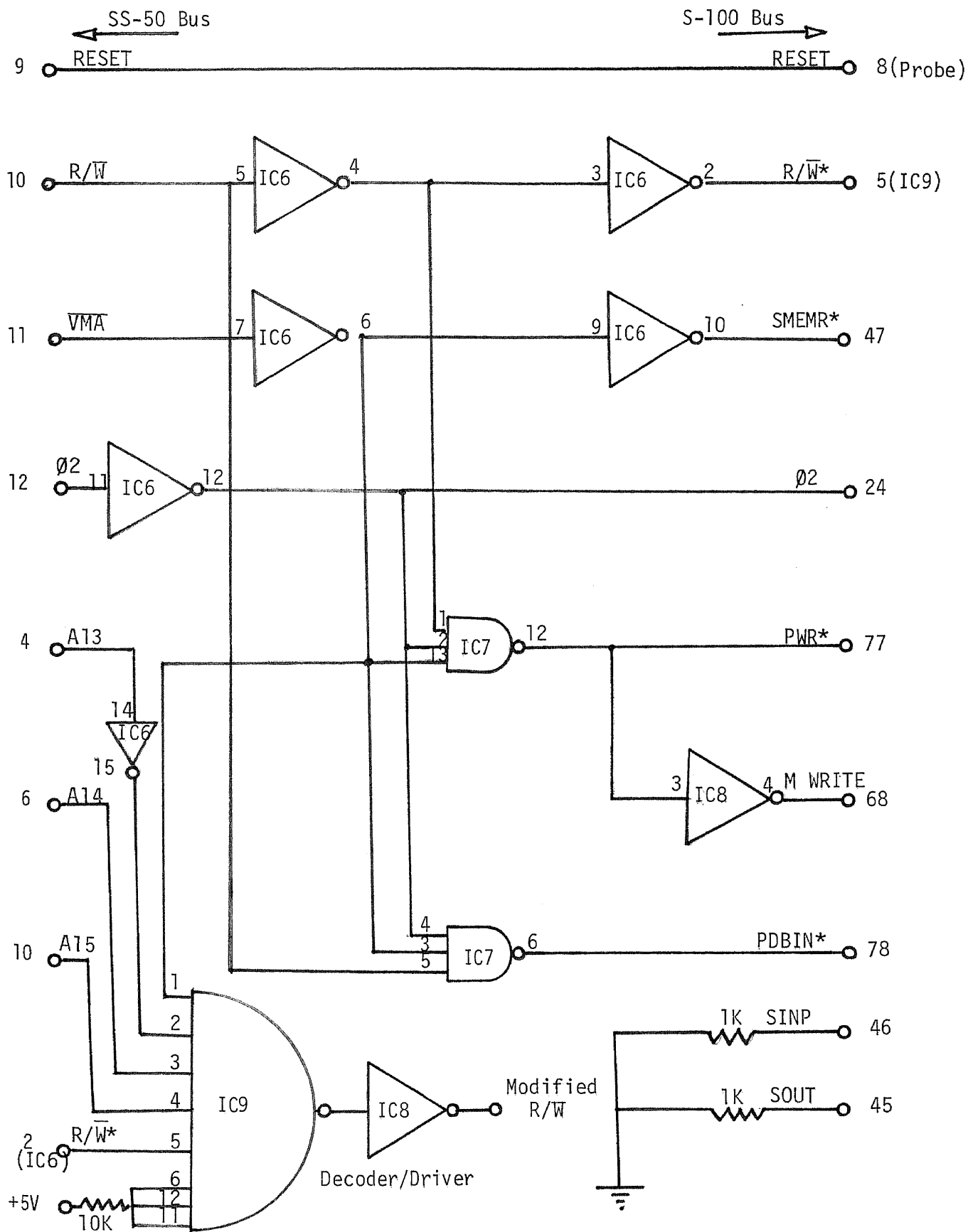


Fig. A2---Data Bus Interface



IC6: 4049; VCC = Pin 1, GND = Pin 8. IC7: 74LS10. IC8: 7404
 IC9: 74LS30. All VCC = Pin 14, GND = Pin 7.

Fig. A3---Control Bus Interface

APPENDIX B

SOFTWARE FOR SIGNATURE ANALYSIS

The attached software was developed for use with the Signature Analysis System.

The software was developed on a Motorola 6800 Microcomputer System. Assembly Language Programming was used throughout the development and a T.S.C. Assembler was used for generating the object code. All the editing, etc., was done via the T.S.C. Editor. All the information was stored and retrieved using FLEX 2.0 Disk Operating System via Dual Eight Inch Shugart Floppy Disks.

The programs which were developed for the project were divided into three functional parts and they are listed below:

1. Initialization Program: This program initializes all the functions of Signature Analysis Subsystem.
2. Pattern Generator Program: This program generates special start and stop codes to the circuit under test as well as the patterns which are used as stimulus to the circuit under test.
3. Control Program: This program displays the Signatures and controls the Signature analysis flow of logic to determine the faults.

NAM PATTERN GENERATOR PROGRAM

*VERSION 3.00 DATED 01-05-80
 *FOR USE WITH SWTC 6800 SYSTEM
 *USES 6820 PIA "B" SIDE FOR OUTPUT
 *PROGRAMMER: SYED R. ALI

```

      OPT     SYM
801A     PIA2 EQU $801A     3-SIDE OR3:OUTPUT REG.,DDR3.
801B     PIA3 EQU $801B     3-SIDE CR3:CONTROL REGISTER
A550     SAVEA EQU $A550
A551     SAVEB EQU SAVEA+1
A552     SAVEX EQU SAVEB+1
A500     ORG     $A500
  
```

SAVE REGISTERS

```

A500 37 A5 50     STA A     SAVEA
A503 F7 A5 51     STA B     SAVEB
A506 FF A5 52     STX      SAVEX
  
```

*PIA INITIALIZATION

```

A509 0F          START SEI           SET INTERRUPT MASK
A50A 7F 80 1B   CLR      PIA3       CLEAR CR3,BIT 2=0 GIVES ACCESS
A50D 86 FF     LDA A     #$FF       OUTPUT CONDITION BITS FOR DDR
A50F B7 80 1A   STA A     PIA2       STORE ABOVE IN DDR
A512 C6 3E     LDA B     #$3E       SETS BIT 2=1 GIVES CTRL TO OR3
A514 F7 80 1B   STA B     PIA3       ALSO STORES CR3,CB1,CB2,IR3.
A517 0E          CLI           CLEAR INTERRUPT MASK
  
```

*PATTERN GENERATOR PROGRAM

```

A518 86 80     LDA A     #$80       START BIT 8 FOR SIG. MODULE
A51A B7 80 1A   STA A     PIA2       OUTPUT START BIT TO SIG. MODUE
A51D 86 00     LDA A     #00       INITIALIZE A-REGISTER
A51F 01          NOP
A520 01          NOP
A521 4C          OUT      INC A           INCREMENT A-REGISTER
A522 C6 00     LDA B     #$00
A524 B7 80 1A   STA A     PIA2       OUTPUT PATTERN
A527 11          CBA
A528 26 F7     SNE      OUT
  
```

RESTORE REGISTERS

```

A52A B6 A5 50     LDA A     SAVEA
A52D F6 A5 51     LDA B     SAVEB
A530 FE A5 52     LDX      SAVEX
A533 39          RTS
      END
  
```

NO ERROR(S) DETECTED

SYMBOL TABLE:

OUT	A521	PIA2	801A	PIA3	801B	SAVEA	A550
SAVEB	A551	SAVEX	A552	START	A509		

CONTROL PROGRAM
 *THIS PROGRAM DISPLAYS THE SIGNATURE AND
 *ALSO CONTROLS THE SIGNATURE ANALYSIS MODULE.
 *THE BASIC PRINT PROGRAM INTERFACES WITH THIS
 *PROGRAM AT SYMBOLIC "USTART".

1	A2E2		INITLZ	EQU	\$A2E2	
2	A279		WATSTX	EQU	\$A279	
3	A25D		CONDX	EQU	\$A25D	
4	A2A7		WWDW	EQU	\$A2A7	
5	A500		PAT	EQU	\$A500	
6	A24E		HALTX	EQU	\$A24E	
7	A2CA		WHALT	EQU	\$A2CA	
8	A15D		REJSG	EQU	\$A15D	
9	A400		CRTSGX	EQU	\$A400	
10	A5F0		YTEMP	EQU	\$A5F0	
11	A186		DISPLY	EQU	\$A186	
12	A5FD		TEMPZ	EQU	\$A5FD	
13			OPT		SYM	
14	0028		ORG		\$0028	
15	0028	A6 00	FDB		USTART	
16	A600		ORG		\$A600	
17	A600	7E A6 0E	USTART	JMP	START1	
18	A603	FE C0 00	BEGIN	LDX	\$C000	SIG.MODULE INTERNAL ADDRESS
19	A606	BD A2 E2		JSR	INITLZ	INITIALIZE CHECK FOR PROPER
20	A609	C6 01		LDA	#501	INITIALIZATION
21	A60B	11		JBA		OTHERWISE TRY AGAIN
22	A60C	26 F5		BNE	BEGIN	
23	A60E	BD A2 79	START1	JSR	WATSTX	HOLD & START SIG.MODULE
24	A611	BD A2 5D		JSR	CONDX	ENABLE SIG.MODULE
25	A614	BD A2 A7		JSR	WWDW	WAIT FOR GATE
26	A617	BD A5 00	PATOUT	JSR	PAT	START PATTERN GENERATOR
27	A61A	BD A6 39		JSR	DELAY1	DELAY FOR SYSTEM TO SETTLEDOWN
28	A61D	BD A2 4E		JSR	HALTX	TURN SIG.MODULE OFF
29	A620	BD A2 CA		JSR	WHALT	WAIT FOR IT TO STOP
30	A623	BD A1 5D	READ	JSR	REJSG	READ SIGNATURE
31	A626	FE A4 00		LDX	CRTSGX	LOAD SIGNATURE
32	A629	FF A5 F0		STX	YTEMP	AT LOCATION YTEMP
33	A62C	FE A4 01		LDX	CRTSGX+1	AND STORE NEXT BYTE
34	A62F	FF A5 F2		STX	YTEMP+2	AT LOCATION YTEMP+2
35	A632	CE A5 F0	DISPLAY	LDX	#YTEMP	X-REG POINTS TO SIGNATURE
36	A635	BD A1 86		JSR	DISPLY	FOR THE SIGNATURE DISPLAY PR.
37	A638	39		RTS		
38	A639	FF A5 FD	DELAY1	STX	TEMPZ	SAVE X-REGISTER
39	A63C	CE 00 FF		LDX	#500FF	LOAD X-REG WITH DELAY
40	A63F	09	TIME	DEX		DECREMENT COUNT
41	A640	26 FD		BNE	TIME	TILL ZERO
42	A642	FE A5 FD		LDX	TEMPZ	RESTORE X-REGISTER
43	A645	39		RTS		RETURN
44			END			
45						
46						
47						
48						
49						
50						
51						
52						
53						
54						
55						
56						
57						
58						
59						
60						
61						
62						
63						
64						
65						
66						
67						
68						
69						
70						
71						
72						
73						
74						
75						
76						
77						
78						
79						
80						
81						
82						
83						
84						
85						
86						
87						
88						
89						
90						
91						
92						
93						
94						
95						
96						
97						
98						
99						
100						

NO ERROR(S) DETECTED

SYMBOL TABLE:

BEGIN	A603	CONDX	A25D	CRTSGX	A400	DELAY1	A639
DISPLA	A632	DISPLY	A186	HALTX	A24E	INITLZ	A2E2
PAT	A500	PATOUT	A617	READ	A623	REJSG	A15D
START1	A60E	TEMPZ	A5FD	TIME	A63F	USTART	A600
WATSTX	A279	WHALT	A2CA	WWDW	A2A7	YTEMP	A5F0

*INITIALIZATION PROGRAM

*
 *THIS PROGRAM INITIALIZES ALL FUNCTIONS OF
 *THE SIGNATURE ANALYSIS MODULE. THIS PROGRAM
 *ALSO CONTAINS JUMP VECTOR TABLE TO VARIOUS
 *SUBROUTINES.
 *

C000	LEFTSG EQU	\$C000
C002	RGHTSG EQU	LEFTSG+2
C004	AQUCSS EQU	RGHTSG+2
C006	AJUSSG EQU	AQUCSS+2
C008	LEFTDS EQU	AJUSSG+2
C00A	RIGHTS EQU	LEFTDS+2
C00C	STATUS EQU	RIGHTS+2

*
 *
 *CURRENT VALUE HOLDING AREA
 *

A400	CRTSGX EQU	\$A400
A402	CRTSTX EQU	CRTSGX+2
A403	CRTCXX EQU	CRTSTX+1
A404	CRTSXX EQU	CRTCXX+1
A405	LEFTCR EQU	CRTSXX+1
A406	RGHTCR EQU	LEFTCR+1
A407	ZTEMP EQU	RGHTCR+1
A409	HREG EQU	ZTEMP+2
A40A	LREG EQU	HREG+1
A433	STAC1 EQU	LREG+41

*
 *SPECIAL CONTROL BITS
 *

0080	WAITS EQU	\$80
0002	GOXXX EQU	\$02
0010	BLANK EQU	\$10
0080	TSC EQU	\$80
007F	TSCM EQU	\$7F
0040	DEC EQU	\$40
00BF	DECM EQU	\$BF
0004	HALT EQU	\$04
0003	NES EQU	\$03
0001	MMSET EQU	\$01
0080	WNDW EQU	\$80
0040	WAIT EQU	\$40

* VECTOR JUMP TABLE

A0FF		ORG	\$A0FF
A0FF	7E A2 E2	JMP	INITLZ
A102	7E A1 47	JMP	WRTSG
A105	7E A1 5D	JMP	REDSG
A108	7E A1 6E	JMP	READSM
A10B	7E A1 7D	JMP	REDST
A10E	7E A1 36	JMP	DISPLY
A111	7E A1 F9	JMP	BLANKX
A114	7E A2 07	JMP	WRDC1
A117	7E A2 15	JMP	DWRDC1
A11A	7E A2 23	JMP	EDGEX
A11D	7E A2 31	JMP	DEDEGX
A120	7E A2 3C	JMP	HOLDNO
A123	7E A2 3A	JMP	WWAIT
A126	7E A2 3F	JMP	GOXXXX
A129	7E A2 4E	JMP	HALTX
A12C	7E A2 5D	JMP	CONDX

A12F	7E	A2	D4	JMP	DECTR
A132	7E	A2	6B	JMP	WAITX
A135	7E	A2	79	JMP	WATSTX
A138	7E	A2	30	JMP	MRESET
A133	7E	A2	8F	JMP	EDGES
A13E	7E	A2	A7	JMP	WWNDW
A141	7E	A2	31	JMP	SGCLR
A144	7E	A2	CA	JMP	WHALT

*
*CONTROL SUB-PROGRAMS

*
*SIGNATURE WRITE

A147	E6	00		WRTSG	LDA B 0,X
A149	FF	A4	07	STX	ZTEMP
A14C	FE	C0	00	LDX	LEFTSG
A14F	E7	00		STA B 0,X	
A151	FE	A4	07	LDX	ZTEMP
A154	08			INX	
A155	E6	00		LDA B 0,X	
A157	FE	C0	02	LDX	RGHTSG
A15A	E7	00		STA B 0,X	
A15C	39			RTS	

*SIGNATURE READ

A15D	FE	C0	00	REDSG	LDX LEFTSG
A160	E6	00		LDA B 0,X	
A162	F7	A4	00	STA B CRTSGX	
A165	FE	C0	02	LDX	RGHTSG
A168	E6	00		LDA B 0,X	
A16A	F7	A4	01	STA B CRTSGX+1	
A16D	39			RTS	

*READ STATUS OF SIGNATURE

*MODULE THEN MASK

A16E	F7	A4	09	READSM	STA B HREG
A171	FE	C0	0C	LDX	STATUS
A174	E6	00		LDA B 0,X	
A176	F7	A4	02	STA B CRTSTX	
A179	F4	A4	09	AND B HREG	
A17C	39			RTS	

*READ STATUS OF SIGNATURE MODULE

A17D	FE	C0	0C	REDST	LDX STATUS
A180	E6	00		LDA B 0,X	
A182	F7	A4	02	STA B CRTSTX	
A185	39			RTS	

*DISPLAY CONTROL

A186	F6	A4	03	DISPLY	LDA B CRTCXX
A189	C4	E0		AND B #5E0	
A18B	F7	A4	03	STA B CRTCXX	
A18E	7F	A4	09	CLR	HREG
A191	FF	A4	07	STX	ZTEMP
A194	BD	A1	B9	JSR	DPLYA
A197	FE	C0	08	LDX	LEFTDS
A19A	E7	00		STA B 0,X	
A19C	F7	A4	05	STA B LEFTCR	
A19F	BD	A1	B9	JSR	DPLYA
A1A2	FE	C0	0A	LDX	RIGHTS
A1A5	E7	00		STA B 0,X	
A1A7	F7	A4	06	STA B RGHTCR	
A1AA	F6	A4	03	LDA B CRTCXX	
A1AD	FE	C0	04	LDX	AQUOSG
A1B0	FA	A4	09	ORA B HREG	
A1B3	E7	00		STA B 0,X	
A1B5	F7	A4	03	STA B CRTCXX	
A1B8	39			RTS	
A1B9	F6	A4	09	DPLYA	LDA B HREG
A1BC	59			ROL B	
A1BD	59			ROL B	

6	A13E	F7	A4	09		STA	B	HREG
7	A131	FE	A4	07		LJX		ZTEMP
8	A134	E6	00			LDA	B	0,X
9	A136	C4	0A			AND	B	#10
10	A138	27	08			BEQ		DPLYB
11	A13A	F6	A4	09		LDA	B	HREG
12	A13D	CA	02			ORA	B	#02
13	A13F	F7	A4	09		STA	B	HREG
14	A1D2	E6	00		DPLYB	LDA	B	0,X
15	A1D4	C4	0F			AND	B	#30F
16	A1D6	59				ROL	B	
17	A1D7	59				ROL	B	
18	A1D8	59				ROL	B	
19	A1D9	59				ROL	B	
20	A1DA	F7	A4	0A		STA	B	LREG
21	A1DD	08				INX		
22	A1DE	E6	00			LDA	B	0,X
23	A1E0	C4	0A			AND	B	#10
24	A1E2	27	08			BEQ		DPLYC
25	A1E4	F6	A4	09		LDA	B	HREG
26	A1E7	CA	01			ORA	B	#01
27	A1E9	F7	A4	09		STA	B	HREG
28	A1EC	E6	00		DPLYC	LDA	B	0,X
29	A1EE	C4	0F			AND	B	#30F
30	A1F0	FA	A4	0A		ORA	B	LREG
31	A1F3	08				INX		
32	A1F4	FF	A4	07		STX		ZTEMP
33	A1F7	08				INX		
34	A1F8	39				RTS		
35								* CLEAR DISPLAY
36	A1F9	FE	C0	04	BLANKX	LDX		AQUCSG
37	A1FC	F6	A4	03		LDA	B	CRTCXX
38	A1FF	CA	10			ORA	B	#BLANK
39	A201	E7	00			STA	B	0,X
40	A203	F7	A4	03		STA	B	CRTCXX
41	A206	39				RTS		
42								*CONTROL WORD "ON"
43	A207	FE	C0	04	WRDC1	LDX		AQUCSG
44	A20A	F6	A4	03		LDA	B	CRTCXX
45	A20D	CA	80			ORA	B	#TSC
46	A20F	E7	00			STA	B	0,X
47	A211	F7	A4	03		STA	B	CRTCXX
48	A214	39				RTS		
49								*CONTROL WORD "OFF"
50	A215	FE	C0	04	DWRDC1	LDX		AQUCSG
51	A218	F6	A4	03		LDA	B	CRTCXX
52	A21B	C4	7F			AND	B	#TSCM
53	A21D	E7	00			STA	B	0,X
54	A21F	F7	A4	03		STA	B	CRTCXX
55	A222	39				RTS		
56								*EDGE CONTROL SELECT
57	A223	FE	C0	04	EDGEX	LDX		AQUCSG
58	A226	F6	A4	03		LDA	B	CRTCXX
59	A229	CA	40			ORA	B	#DEC
60	A22B	E7	00			STA	B	0,X
61	A22D	F7	A4	03		STA	B	CRTCXX
62	A230	39				RTS		
63								*EDGE CONTROL "OFF"
64	A231	FE	C0	04	DEDEX	LDX		AQUCSG
65	A234	F6	A4	03		LDA	B	CRTCXX
66	A237	C4	3F			AND	B	#DECM
67	A239	E7	00			STA	B	0,X
68	A23B	F7	A4	03		STA	B	CRTCXX
69	A23E	39				RTS		
70								*ACTUATE SIGNATURE MODULE
71	A23F	FE	C0	06	GOXXXX	LDX		AJUS3G


```

A242 F6 A4 04 LDA B CRTSXX
A245 C8 02 EOR B #30XXX
A247 E7 00 STA B 0,X
A249 CA 02 ORA B #30XXX
A24B E7 00 STA B 0,X
A24D 39 RTS

```

*STOP SIGNATURE MODULE

```

A24E FE 00 06 HALTX LDX AQUSSG
A251 F6 A4 04 LDA B CRTSXX
A254 C8 04 EOR B #HALT
A256 E7 00 STA B 0,X
A258 CA 04 ORA B #HALT
A25A E7 00 STA B 0,X
A25C 39 RTS

```

*ENABLE SIGNATURE MODULE

```

A25D FE 00 06 JONDX LDX AQUSSG
A260 F6 A4 04 LDA B CRTSXX
A263 CA 08 ORA B #NES
A265 F7 A4 04 STA B CRTSXX
A268 E7 00 STA B 0,X
A26A 39 RTS

```

*PUT SIGNATURE MODULE IN WAIT STATE

```

A26B FE 00 06 WAITX LDX AQUSSG
A26E F6 A4 04 LDA B CRTSXX
A271 CA 80 ORA B #WAITS
A273 F7 A4 04 STA B CRTSXX
A276 E7 00 STA B 0,X
A278 39 RTS

```

*WAIT AND START SIGNATURE MODULE

```

A279 BD A2 6B WATSTX JSR WAITX
A27C BD A2 3F JSR GOXXXX
A27F 39 RTS

```

*

SIGNATURE MODULE RESET

```

A280 FE 00 06 MRESET LDX AQUSSG
A283 F6 A4 04 LDA B CRTSXX
A286 CA 01 ORA B #MMSET
A288 E7 00 STA B 0,X
A28A D8 01 EOR B MMSET
A28C E7 00 STA B 0,X
A28E 39 RTS

```

*SELECT EDGE TYPE

```

A28F FE 00 06 EDGES LDX AQUSSG
A292 C4 07 AND B #507
A294 59 ROL B
A295 59 ROL B
A296 59 ROL B
A297 59 ROL B
A298 F7 A4 09 STA B HREG
A29B F5 A4 04 LDA B CRTSXX
A29E FA A4 09 ORA B HREG
A2A1 F7 A4 04 STA B CRTSXX
A2A4 37 00 STA B 0,X
A2A6 39 RTS

```

*DELAY FOR WINDOW

```

A2A7 FE 00 03 WWNDW LDX STATUS
A2AA E6 00 LDA B 0,X
A2AC C4 80 AND B #WNDW
A2AE 27 F7 BEQ WWNDW
A2B0 39 RTS

```

*

ZERO THE SIGNATURE

```

A2B1 FE A2 38 SGCLR LDX CLRS1
A2B4 BD A1 47 JSR WRTSG
A2B7 39 RTS
A2B8 00 CLRS1 FCB 0

```

```

A2B9 00          FC3  0
A2BA 00          FC3  0
A233 00          FC3  0

```

*ZERO THE DELAY

```

A23C FE C0 06  HOLDNO LDX  AQUSSG
A23F F6 A4 04          LDA  B  CRTSXX
A2C2 C8 80          EOR  B  #WAITS
A2C4 E7 00          STA  B  0,X
A2C6 F7 A4 04          STA  B  CRTSXX
A2C9 39          RTS

```

*DELAY FOR ACCEPTED HOLD CONDITION

```

A2CA          WHALT  EQU  *
A2CA FE C0 0C  WAIT   LDX  STATUS
A2CD E6 00          LDA  B  0,X
A2CF C4 40          AND  B  #WAIT
A2D1 27 F7          BEQ   WAIT
A2D3 39          RTS

```

*DECONTROL SIGNATURE MODULE

```

A2D4 FE C0 06  DECTR  LDX  AQUSSG
A2D7 F6 A4 04          LDA  B  CRTSXX
A2DA C8 08          EOR  B  #NES
A2DC F7 A4 04          STA  B  CRTSXX
A2DF E7 00          STA  B  0,X
A2E1 39          RTS

```

*INITIALIZE SIGNATURE MODULE

```

A2E2 FF C0 00  INITLZ STX  LEFTSG
A2E5 08          INX
A2E6 FF C0 02          STX  RGHTSG
A2E9 08          INX
A2EA FF C0 04          STX  AQUCSG
A2ED 08          INX
A2EE FF C0 06          STX  AQUSSG
A2F1 08          INX
A2F2 FF C0 08          STX  LEFTDS
A2F5 08          INX
A2F6 FF C0 0A          STX  RIGHTS
A2F9 08          INX
A2FA 08          INX
A2FB FF C0 0C          STX  STATUS
A2FE C6 06          LDA  B  #506
A300 F7 A4 04          STA  B  CRTSXX
A303 C6 00          LDA  B  #500
A305 F7 A4 03          STA  B  CRTSXX
A308 C6 01          LDA  B  #501
A30A 39          RTS

```

END

NO ERROR(S) DETECTED

SYMBOL TABLE:

AQUCSG	C004	AQUSSG	C006	BLANK	0010	BLANKX	A1F9
CLRS1	A2B8	CONDX	A25D	CRTSXX	A403	CRTSGX	A400
CRTSTX	A402	CRTSXX	A404	DEC	0040	DECM	00BF
DECTR	A234	DEDEX	A231	DISPLY	A186	DPLYA	A1B9
DPLYB	A1D2	DPLYC	A1EC	DWRDC1	A215	EDGES	A28F
EDGEX	A223	GOXXX	0002	GOXXXX	A23F	HALT	0004
HALTX	A24E	HOLDNO	A23C	HREG	A409	INITLZ	A2E2
LEFTCR	A405	LEFTDS	C008	LEFTSG	C000	LREG	A40A
MMSET	0001	MRESET	A280	NES	0008	READSM	A16E
REDSG	A15D	REDST	A17D	RGHTCR	A406	RGHTSG	C002
RIGHTS	C00A	SGCLR	A2B1	STAC1	A433	STATUS	C00C
TSC	0080	TSCM	007F	WAIT	0040	WAITS	0080
WAITX	A26B	WATSTX	A279	WHALT	A2CA	WWDW	0080
WRDC1	A207	WRTSG	A147	WWAIT	A2CA	WWDW	A2A7
ZTEMP	A407						

LIST

```

*
0001 REM THIS IS A BASIC PROGRAM WHICH IS USED FOR PRINT ROUTINES
0002 REM AND SIGNATURE COMPARISONS. THIS PROGRAM INTERFACES WITH
0003 REM MACHINE LANGUAGE PROGRAM VIA "USER(O)" FUNCTION AT
0004 REM REM LOCATION UPSTART IN THE CONTROL PROGRAM.
0005 DIM N1(64),R5(64),J(64),B5(64)
0006 POKE( 0040,166)
0007 POKE( 0041,14)
0010 PRINT "SIGNATURE ANALYSIS SYSTEM ON LINE"
0020 PRINT "WHICH MODE DO YOU WANT? D=DIAGNOSTIC,L=LEARNING"
0030 LET D$="D"
0040 LET L$="L"
0050 INPUT M$
0060 IF M$=D$ THEN 300
0070 IF M$=L$ THEN 190
0190 PRINT "YOU ARE IN LEARNING MODE"
0200 PRINT "HOW MANY NODES ARE YOU GOING TO TEST?"
0210 INPUT X
0212 FOR I=1 TO X
0225 PRINT "TYPE S WHEN YOU HAVE YOUR PROBE ON A NODE FOR START"
0230 INPUT K$
0235 IF K$="S" THEN 240
0237 GOTO 225
0240 LET A=USER(O)
0241 PRINT "DO YOU WANT TO RETEST THIS NODE?"
0242 INPUT Q$
0243 IF Q$="YES" THEN 225
0250 PRINT "TYPE NODE NUMBER FOLLOWED BY ITS GOOD SIGNATURE"
0270 INPUT N1(I),R5(I)
0290 NEXT I
0294 PRINT "DO YOU WANT DIAGNOSTIC MODE?"
0295 INPUT U$
0296 IF U$="NO" THEN 475
0300 PRINT "YOU ARE IN DIAGNOSTIC MODE"
0305 PRINT "TYPE D WHEN YOU HAVE YOUR PROBE ON A NODE FOR DIAGNOSTIC"
0306 INPUT T$
0307 IF T$="D" THEN 309
0308 GOTO 190
0309 LET A=USER(O)
0310 GOSUB 430
0320 INPUT J,B5
0330 IF J>X THEN 400
0340 IF B5=R5(J) THEN 380
0350 PRINT "THIS NODE IS DEFECTIVE"
0360 PRINT "NODE #      GOOD SIG.      BAD SIG."
0370 PRINT J,R5(J),B5
0375 GOTO 390
0380 PRINT "TESTS O.K NODE #";J
0390 GOTO 305
0400 PRINT "NODE # OUT OF RANGE"
0410 GOTO 300
0430 PRINT "DO YOU WANT TO RE-DIAGNOSE THIS NODE?"
0440 INPUT R$
0450 IF R$="YES" THEN 305
0460 PRINT "TYPE NODE NUMBER YOU HAVE JUST TESTED AND ITS SIGNATURE"
0470 RETURN
0475 PRINT "P R O G R A M   S T O P P E D"
0480 END

```

BASIC

#

LIST OF REFERENCES

1. Akers, Sheldon B. A Logic System for Fault Test Generation. IEEE Transaction on Computers, Vol. C-25, No. 6, June 1976.
2. Daniel, H., Lindbloom, E. and Carpenter, R. G. The Weighted Random Test Pattern Generator. IEEE Transactions on Computers. Vol. C-24, No. 7, July 1975.
3. Friedman, Arthur D. and Menon, P. R. Fault Detection in Digital Circuits. New Jersey: Prentice-Hall, Inc., 1971.
4. Fromwerk, Robert A. Signature Analysis: A New Digital Field Service Method. Hewlett Packard Journal, May 1977, Volume 28, Number 9.
5. Motorola. M6800 Microprocessor Applications Manual. Motorola, Inc., 1975.
6. Motorola. M6800 Microprocessor Programming Manual. Motorola, Inc., 1976.
7. Muth, Peter. A Nine-Valued Model for Test Generation. IEEE Transactions on Computers. Vol. C-25, No. 6, June 1976.
8. Soucek, B. R. and K. O. Microprocessors and Minicomputers. New York: John Wiley and Sons, 1976.