To the Graduate Council:

I am submitting herewith a thesis written by Mark Steven Sitter entitled "Hardware Design of TMS9995 Interface Pod." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Engineering, with a major in Electrical Engineering.

Donald W. Bouldin

Donald W. Bouldin, Major Professor

We have read this thesis
and recommend its acceptance:

Robert E Bodenheimer

Marshall Pace

Accepted for the Council:

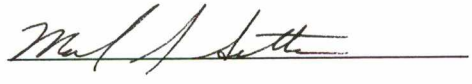
Lew Minkel

Vice Provost
and Dean of The Graduate School

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a Master's degree at The University of Tennessee, Knoxville, I agree that the Library shall make it available to borrowers under rules of the Library. Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of the source is made.

Permission for extensive quotation from or reproduction of this thesis may be granted by my major professor, or in his absence, by the Head of Interlibrary Services when, in the opinion of either, the proposed use of the material is for scholarly purposes. Any copying or use of the material in this thesis for financial gain shall not be allowed without my written permission.

Signature 

Date 12/1/88

HARDWARE DESIGN OF TMS9995 INTERFACE POD

A Thesis

Presented for the

Master of Engineering

Degree

The University of Tennessee, Knoxville

Mark Steven Sitter

December 1988

ABSTRACT

The 9010A troubleshoot and repair stations installed in the Industrial Systems Division (ISD) were not able to troubleshoot new product due to the inability to purchase interface pods. The use of adapter pods to expand the capability of the stations proved ineffective.

To construct an interface pod, the capabilities of the existing 9010A system was examined along with the software required to communicate with the 9010A. The data books of existing parts used by ISD were used to select the parts of the design.

The TMS9995 interface pod met the power, board space, and timing requirements necessary to work with the 9010A and effectively mesh into the buss structure of designs using the TMS9995 microprocessor.

TABLE OF CONTENTS

CHAPTER		PAGE
INTRODUCTION		1
I.	RATIONALE FOR DEVELOPMENT OF TROUBLESHOOT AND REPAIR CENTERS USING 9010A	3
II.	OVERVIEW OF THE 9010A MICRO-SYSTEM TROUBLESHOOTER	9
III.	RATIONALE FOR THE DEVELOPMENT OF TMS9995 INTERFACE POD	25
IV.	HARDWARE DESIGN OF TMS9995 INTERFACE POD	29
V.	SOFTWARE DESIGN GUIDELINES	59
VI.	CONCLUSION	69
LIST OF REFERENCES		72
APPENDIXES		74
APPENDIX A.	SCHEMATIC OF TMS9995 INTERFACE POD	75
APPENDIX B.	PAL EQUATIONS AND SIMULATION FILES	85
APPENDIX C.	SAMPLE PROGRAM FOR 9010A MICRO-SYSTEM TROUBLESHOOTER	110
APPENDIX D.	LIST OF MATERIALS	112
APPENDIX E.	BOARD SPACE REQUIREMENTS	114
APPENDIX F.	INSTRUCTION EXECUTION TIMES	117
APPENDIX G.	SET-UP SEQUENCE	120
VITA		122

LIST OF TABLES

TABLE		PAGE
2-1.	Relationship of Probe to Synchronization . . .	16
2-2.	Pulse Output With "HIGH" and "LOW" Keys . . .	16
2-3.	Set-Up Messages	21
2-4.	Set-Up Messages Related To Aux I/F Command . .	22
3-1.	Summary of Micro-System Troubleshooter Based Equipment Installed In The Industrial Systems Division of Texas Instruments	26
4-1.	Summary of Power Consumption	31
4-2.	9901 Pin Definitions and Locations in Interface Pod "CRU" Memory Map	41
4-3.	Troubleshooting Control Lines	52
4-4.	Memory Location of Error Registers	56
5-1.	"CRU" Read/Write Operations at "CRU" Address 0000 Effect of Quantity Bits	61
5-2.	Example of "Quick" Test Routine "RAM Test"	62
5-3.	Status Information For "Quick" Routine "RAM Test"	64
5-4.	9010A Command List and Address Modifiers . .	65
5-5.	Example of "READ" Command	66
5-6.	Summary of Error Codes	67
5-7.	Error String Returned to 9010A	68

LIST OF FIGURES

FIGURE		PAGE
2-1.	Communication Between 9010A and UUT	10
2-2.	9010A Keyboard	12
2-3.	Probe Information As Returned In Register Zero	14
2-4.	Counting Events On A Falling Edge	15
4-1.	"CRU" Input and Output Cycles	33
4-2.	TMS9995 Memory Map	34
4-3.	Memory Breakdown of TMS9995 Interface Pod . .	36
4-4.	"CRU" Address Map	38
4-5.	Handshaking Process	43
4-6.	9901 Switching Characteristics	44
4-7.	TMS9995 "HOLD" State	50
4-8.	Timing Characteristics	53
4-9.	TMS9995 Memory Interface Timing	54

INTRODUCTION

In today's competitive programmable controller market, survival requires a streamline production flow which eliminates unnecessary material and labor waste. The benefits of a streamlined process are a reduction of work in process, reduction in material storage, and a leaner labor force. One area that has proven beneficial to Texas Instruments in streamlining its process and in improving quality in its products has been the installation of 9010A based troubleshoot and repair centers.

These centers serve as focal points to isolate breakdowns in the manufacturing process. They also serve as the hub which interconnects all Engineering Departments. The data collected by the centers aides not only Procurement Engineering in the identification of vendor process problems, but it also provides valuable indications of process problems within the scope of Texas Instruments' manufacturing operations.

Chapter I of this thesis will discuss the rationale used by Texas Instruments' Industrial Systems Division (ISD) to develop the 9010A-based troubleshoot

and repair centers. An overview of the functions of the 9010A Micro-System troubleshooter will be provided in Chapter II. The rationale for developing the TMS9995 interface pod and the hardware design of the pod will be discussed in Chapters III and IV. Chapter V will cover software techniques used to program the interface pod. This will be followed by the Conclusion in Chapter VI.

CHAPTER I

RATIONALE FOR DEVELOPMENT OF TROUBLESHOOT AND REPAIR CENTERS USING 9010A

During the early 1980's Texas Instruments' manufacturing line was structured in a way that all testing and troubleshooting was done on the same test station. The test station is a 990 based machine which uses precision voltage sources, current meters, and digital multimeters through a matrix box to take parametric measurements of the unit under test (UUT). As production began to increase through the introduction of new products, Texas Instruments began to examine the production lines and field service functions to locate areas which were causing bottlenecks in production, as well as response time to customer needs.

Five areas were identified as bottlenecks in the production line and emphasized areas where excessively long turnaround times existed for repairing defective product. The areas are summarized in the following five sections.

Time Allocation

There was little or no time allocated to the technician for troubleshoot and repair since this

interrupted testing of new product. A product with no defects could be completely tested in less than 10 minutes, whereas a defective product could take up to one hour for the technician to determine the cause of the failure and repair the product. Initially, there was no identified repair station which was dedicated to repair defective product and record the cause of the failure. When a manufacturing supervisor was faced with meeting production schedules, defective product was the last material to be worked. On many lines, the dollar value of the product in troubleshoot and repair (TS&R) was equivalent to or greater than a month's production.

Communication Problems

The test station communicated with the UUT over a communication buss. Consequently, if the unit being tested failed to communicate with the test station, no indication of the cause of the failure is returned. If no problem was found with the communication buss and the product was found to be defective, the test station returned error codes. Because the error codes returned by the test station were vague or non-existent, this lengthened the time that a knowledgeable technician took to repair the board. Another problem was the inability of the test station to loop on errors. This also resulted in lengthy troubleshoot time.

The standard procedure under these circumstances was to change groups of chips which perpetuated the

existence of the problem due to the technicians inability to pinpoint specific problem chips or vendors. Additionally, this procedure masked potential design tolerance problems. If this shotgun approach failed to repair the product, the technician would scrap the product which generated a need for more material. This proved to be a nonproductive waste of material and labor and an unnecessary drain on profitability.

International Repair

International repair sites complained that they did not have 990-based test stations that were being used at the Industrial Systems Division site; therefore, they were unable to repair defective products. The test documentation provided by ISD was centered around the use of 990 based test stations. Without these stations, the documentation provided only enough information to determine some test limits.

The primary task of the international site was to repair defective product. Consequently, there were insufficient resources available to develop test programs and stations. However, an attempt was made by the international sites to provide some testing capability. The tests that were developed often allowed defective product to escape detection. This caused customer discontent, as well as a degradation to the quality image

of Texas Instruments products. Subsequently, the international repair sites were required to maintain a large inventory of known good products to meet customer delivery schedules and quality expectations.

Lack of Training

Training programs did not exist to teach new technicians or engineers the theory of operation of the products. The technician had to learn the product through experimentation and with the help of fellow technicians who may have previously worked on the product.

Lack of training, along with the error reporting methods of the test stations, aggravated existing TS&R problems. The inexperienced technician who did not understand the function of the board found the vague error codes to be useless. The lack of a troubleshoot and repair training program was also a complaint of the international repair centers. The amount of training their engineers and technicians received was not sufficient to repair the product for which they were responsible. An additional problem was that training was extremely expensive since representatives from the international sites were required to train at the ISD site in Johnson City, TN.

New Product Testing

Design engineering was not able to fully test their new hardware designs because test engineering was

busy building test fixtures and writing test software off the design specifications at the same time the design engineers were working on the hardware. Consequently, costly design errors went undetected until they were discovered during first production of the product. These errors usually resulted in rework of product. For example, blue wire fixes would often stay with product for up to two years due to the inability of design to detect errors early in the design cycle. These errors degraded the reliability of the product, as well as adding an unnecessary drain on labor.

Starting in 1984, Texas Instruments purchased 9010A troubleshooters and interface pods from the John Fluke Company. A team of engineers and technicians was established to assist manufacturing and the international repair centers in troubleshooting their product. Within a year of the 9010A's installation, the Industrial Systems Division had achieved the following goals.

1. Establishment of a stand alone troubleshoot and repair center. This eliminated the necessity of utilizing a 990 based test station as a troubleshoot and repair tool.

2. The ability to test engineering prototypes prior to the completion of the test program designed for the 990 test station. This provided identification of design errors early in the design cycle thereby minimizing the cost of redesign.

3. Texas Instruments expansion into the contract manufacturing business in 1985 resulted in additional demands for stand-alone troubleshoot and repair centers. These demands were met by the introduction of the 9010A troubleshoot and repair system.

4. The international repair centers now had a cost effective means to repair defective product. After initial training on the 9010A system was completed, no additional training was required to cover any new product introduction. The repair center in the Industrial Systems Division would forward new programs to the international sites along with documentation on the hardware that was being tested. This resulted in decreased cycle times and reduction in product inventories.

Achievement of the above goals allowed Texas Instruments to separate the troubleshoot and repair stations from the test stations. No longer limited, manufacturing lines were capable of repairing defective product which provided an avenue for reporting and identifying quality problems. These troubleshoot and repair stations also provided a tool to verify hardware design integrity prior to production. This verification capability resulted in an improvement to the cost of quality due to early detection of product errors. These stations provide a low cost, user friendly troubleshoot and repair center.

CHAPTER II

OVERVIEW OF THE 9010A MICRO-SYSTEM TROUBLESHOOTER

The Fluke 9010A Micro-System Troubleshooter interacts with the UUT through the microprocessor socket on the board. Unlike other testing equipment, the 9010A does not gather information by taking samples of the data directly from the unit under test. The 9010A takes control of the UUT via the microprocessor inside the interface pod.

All 9010A operations are based on its ability to manipulate data at any address. Figure 2-1 illustrates the connection between the 9010A troubleshooting system and the UUT.

9010A System Functions

The 9010A troubleshooting system is designed to troubleshoot buss-oriented architectures. The system uses interface pods which mimic microprocessors with up to 32 address lines and 32 data lines. The 9010A is connected to a UUT through an interface pod which uses the same microprocessor as the unit currently being tested. This is done to match the timing structure as closely as possible.

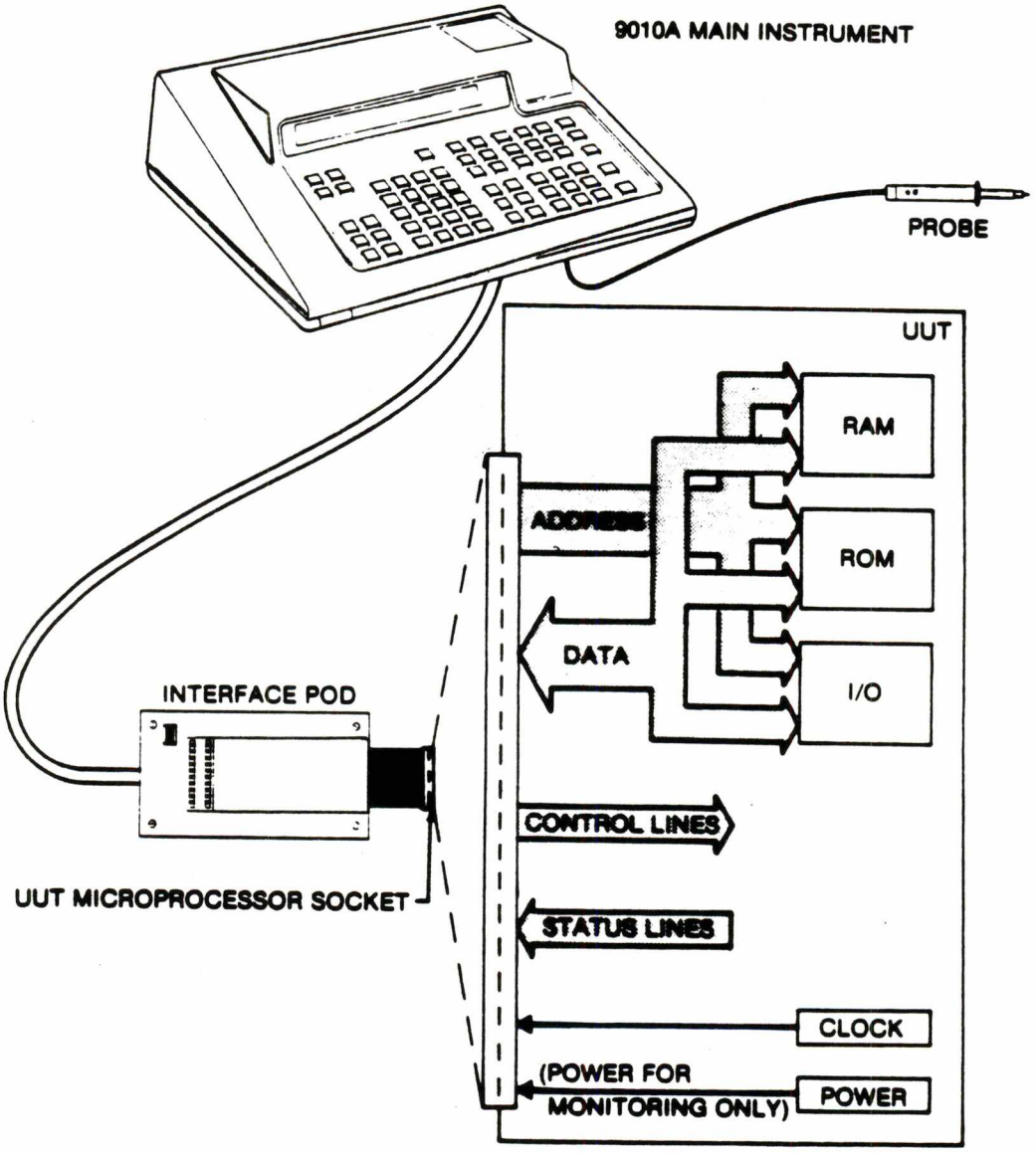


FIGURE 2-1

COMMUNICATION BETWEEN 9010A AND UUT

Source: 9010A Micro-System Troubleshooter Operator Manual, John Fluke Manufacturing Company, Incorporated, June 1981, p. 3A-3.

The timing of the signals to the unit under test will not change dramatically, thereby allowing the user to access the hardware without introducing major timing problems. In fact, the extra loading by the interface pod on the buss structure of the UUT tends to aggravate any existing timing problems. This will highlight to Design Engineering potential timing problems.

The 9010A issues commands to the interface pod to perform requested troubleshoot operations. Each interface pod contains the information required to perform the task. The 9010A serves strictly as a reporting and command center. All intelligence is contained on the interface pod with the exception of reading and writing to magnetic tapes, keyboard inputs, data display and troubleshoot probe functions.

Figure 2-2 illustrates the keyboard of a 9010A system. A brief discussion of the various functions of the keyboard will follow.

Magnetic Tape

The user is able to save or retrieve 10K of programs to and from a magnetic tape drive referred to as Tape on Figure 2-2. When the user presses the "READ TAPE" key, all existing programs residing in the 9010A will be written over by the programs on the tape drive. When the user presses the "WRITE" key, all programs residing in the 9010A memory will be loaded to magnetic tape.

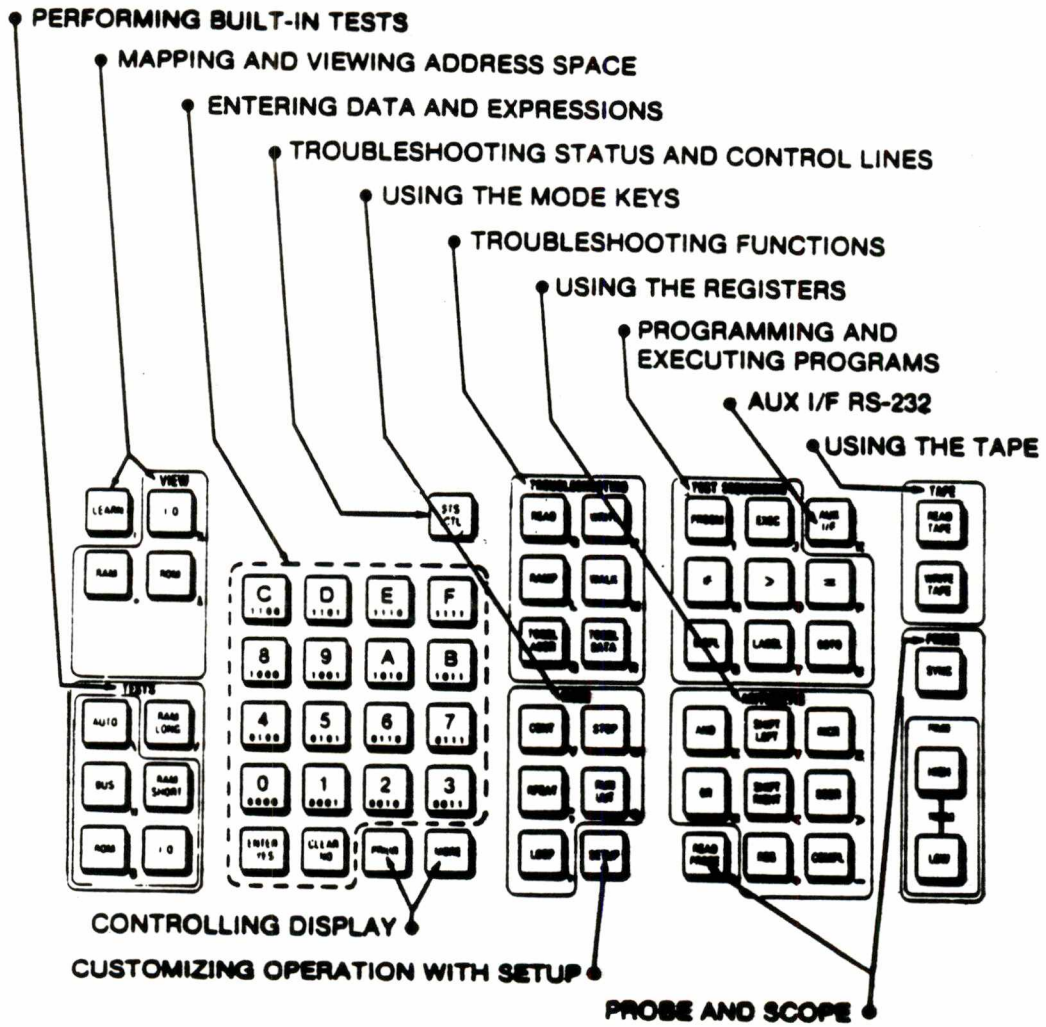


FIGURE 2-2

9010A KEYBOARD

Source: 9010A Micro-System Troubleshooter Operator Manual, John Fluke Manufacturing Company, Incorporated, June 1981, p. 3C-1.

Probe

The "READ PROBE" key serves as the starting and ending point for gathering a signature, counting events, and determining logic level history. The user provides a stimulus between two actuations of the "READ PROBE" key to generate activity on the buss. The data will be returned in register zero in the form shown in Figure 2-3.

The logic level history provides the user with the voltage level the signal obtained during the sampling period. If a signal was above 2.4 volts, it was considered a "HIGH," if a signal was between 2.4 and 0.8 volts, it was considered "TRI-STATE," and below 0.8 volts was considered a "LOW."

The computer signature provides a unique hexadecimal number between 0000 and FFFF. The probe must be synchronized to the interface pod to gather signatures which is accomplished by pressing the "SYNC" key on the 9010A keyboard.

Figure 2-4 represents an event count cycle. The event count returns a decimal number between 0 and 127 during the sampling period. An event is determined when the signal falls below 2.4 volts. If more than 127 events occur, the counter simply resets and begins counting from zero.

Additionally, the probe can serve as a pulse generator. By pushing the "HIGH" key, the probe will pulse

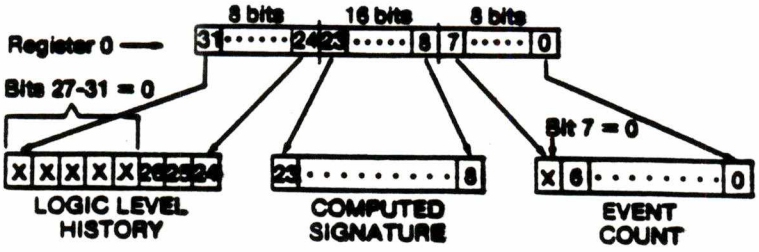
DISPLAY MESSAGE FORMAT: *PROBE-LVL abc COUNT ddd SIG nnnn*

a = *L* if logic low detected, blank otherwise
b = *X* if invalid logic level detected (this level is not detected in a synchronization mode), blank otherwise
c = *H* if logic high detected, blank otherwise
ddd = decimal number between 0 and 127
nnnn = hexadecimal signature between 0000 and FFFF

Example: *PROBE-LVL L H COUNT 060 SIG 264D*

This message indicates that the probe detected the low and high logic levels, 60 events were counted, and a signature of 264D was computed.

The probe response data is stored in bits 0-31 of Register 0 as follows:



Bit 24 = 1 if logic high detected

Bit 25 = 1 if logic invalid detected

Bit 26 = 1 if logic low detected

If register 0 is examined with the REG key, the contents are displayed as a seven-digit hexadecimal quantity. (Note that the eighth hexadecimal digit is always zero, and is not displayed). For the example above, the contents of Register 0 are displayed as the following:

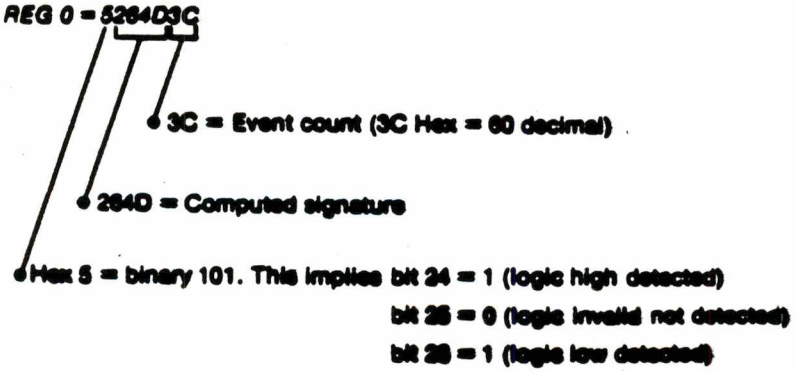


Figure 2-3

PROBE INFORMATION AS RETURNED IN REGISTER ZERO

Source: 9010A Micro-System Troubleshooter Operator Manual, John Fluke Manufacturing Company, Incorporated, June 1981, p. 4K-7.

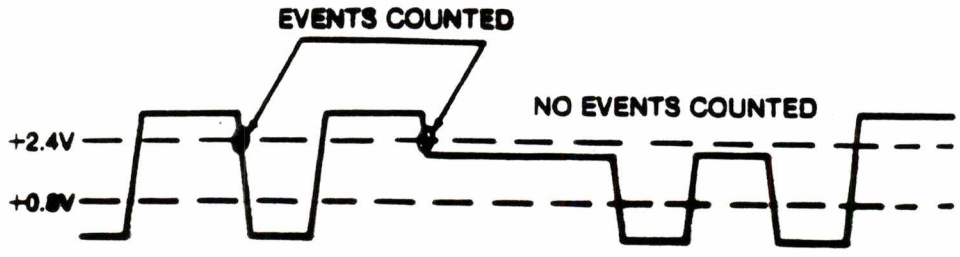


FIGURE 2-4

COUNTING EVENTS ON A FALLING EDGE

Source: 9010A Micro-System Troubleshooter Operator Manual. John Fluke Manufacturing Company, Incorporated, June 1981, p. 4K-6.

from "TRI-STATE" to "HIGH." By pushing the "LOW" key, the probe will pulse from "TRI-STATE" to "LOW." If both keys are pressed, the probe will pulse from "LOW" to "HIGH." See Tables 2-1 and 2-2 for illustrations of the pulsing functions.

Built-In Tests

The 9010A system has five canned routines -- the buss test, ROM test, I/O test, RAM short, and RAM long test. A brief discussion of each of the tests will follow.

Buss test. This test will attempt to drive all lines from the microprocessor socket to the first group of buffers on the UUT. It determines "stuck at faults" and lines tied together on the buss local to the microprocessor. Consequently, any buss lines with problems

TABLE 2-1

RELATIONSHIP OF PROBE TO SYNCHRONIZATION

PROBE USE	ADDRESS OR DATA SYNC	FREE-RUN
PROBE STIMULUS (determined by HIGH and LOW keys)	One pulse per address or data valid period. Pulse equals period in length.	2 μs pulses at 1 kHz.
PROBE RESPONSE (start and stop points determined by Read Probe operation)		
Event Count*	Counts all events (unaffected by sync).	Counts all events.
Signature Gathering	Counts only during address or data valid period.	No signatures gathered in free-run.
Logic Level History	Records only during address or data valid period (only high or low levels recorded, not invalid).	Records any levels at any time.
INDICATOR LIGHTS	Indicates probe tip activity only during address or data valid period.	Indicates probe tip activity during all times.

*An event occurs when a signal falls below the high threshold (+2.4V)

TABLE 2-2

PULSE OUTPUT WITH "HIGH" AND "LOW" KEYS

HIGH KEY	LOW KEY	TYPE OF STIMULUS GENERATED
In	Out	High pulses.
Out	In	Low pulses.
In	In	Toggle between high and low pulses.
Out	Out	No stimulus generated.

Source: 9010A Micro-System Troubleshooter Operator Manual, John Fluke Manufacturing Company, Incorporated, June 1981, p. 4K-4.

past the buffers will have to be detected with another routine.

"ROM" test. This test will calculate the signature of the "ROMs" in a specific memory block and compare it to the signature previously calculated or specified by the user.

"RAM" short test. This test verifies that each data bit can be drive and that the address buss is decoding properly in the memory block specified.

"RAM" long test. This test performs the same operation specified in the "RAM" short test, as well as a more extensive test of the address buss, and a check for soft "RAM" errors.

I/O test. This test performs a read/write test on the specified I/O registers.

An additional "AUTO" test function is available with the Built-In Tests. The user may elect to use the "AUTO" key to perform the following sequence of canned routines: Buss test, "RAM" short test, "ROM" test, and I/O test. The "RAM" long test is not included in the auto test function due to the length of time it requires for completion.

The view block directly above the test block on the 9010A keyboard allows the user to specify blocks of memory

to be tested by the canned routines. The "RAM", I/O and "ROM" keys allow the user to view the existing area specified or to specify new areas in the UUT memory map for testing. When the user specifies a new "ROM" area in the memory map, the 9010A will request a signature. If the user does not enter a signature, one will be calculated for that block by the 9010A.

If the user does not know the memory map of the UUT, he may use the "LEARN" key. This key will instruct the 9010A to sample the memory map at 4K intervals to determine "RAM," "ROM" and I/O areas.

Keyboard Tests

The next area of discussion will be the troubleshooting routines available via the keyboard of the 9010A. These routines are "READ," "WRITE," "WALK," "TOGGLE ADDRESS," and "TOGGLE DATA."

"READ" command. The "READ" command reads data from an address specified by the user.

"WRITE" command. The "WRITE" command allows the user to write data to a specified address on the UUT.

"RAMP" command. The "RAMP" command writes a series of hexadecimal patterns to an address specified by the user on the UUT. The hexadecimal patterns will be from 0000 to FFFF, i.e., 0000, 0001, 0002...FFFE, FFFE, FFFF.

"WALK" command. The "WALK" command will write a specified data pattern into a UUT address determined by the user. The 9010A will then shift the data pattern to the right one position taking the least significant bit and placing it in the most significant position. This shifting operation will continue until the original data pattern is returned.

"TOGGLE ADDRESS." This routine will allow the user to toggle a specified address bit at a user determined address on the UUT.

"TOGGLE DATA." This routine will allow the user to toggle a specified data bit at an address identified by the user on the UUT.

Status/Control "STS/CTL" Lines

Included in the troubleshooting routines is the ability to write and read the status and control lines. Status lines are defined as lines which are inputs to the microprocessor, and control lines are defined as lines which are outputs from the microprocessor. The 9010A is designed to handle eight conrol lines and 16 status lines. The user can read the status lines by pressing the "READ" key, followed by the "STS/CTL" key on the 9010A keyboard. The user can write the to the control lines by pressing the "WRITE" key followed by the "STS/CTL" key. The user can

also toggle a specified control line by pressing the pressing the "TOGGLE DATA" key followed by the "STS/CTL" key. Writing to control lines allows the user to set the status of all control lines.

Command Center

Tables 2-3 and 2-4 contain lists of set-up messages and default values. By pressing the set-up key, the user is given the ability to enable microprocessor lines or specify the recording of microprocessor errors.

Mode Keys

The next section concerns the mode key which are "STOP," "CONTINUE," "REPEAT," "LOOP" and "RUN UUT." The "STOP" key terminates the command the 9010A is currently executing. The "CONTINUE" key instructs the 9010A to contine executing the command after an error has been reported. The "REPEAT" key instructs the 9010A to repeat the previous command given and the "LOOP" key instructs the 9010A to continuously execute the last command given.

Additionally, if an error is detected, the 9010A asks the user if a looping function is required. If "yes" is entered, the 9010A will continue to execute the operation where the error occurred. This provides the user with a continuous report of the status of the error as attempts are being made to repair it.

TABLE 2-3

SET-UP MESSAGES

DISPLAY MESSAGE	DESCRIPTION
SET - TRAP BAD PWR SUPPLY? YES *SET - TRAP ILLEGAL ADDRESS? YES* *SET - TRAP ACTIVE INTERRUPT? NO* *SET - TRAP ACTIVE FORCE LINE? YES* *SET - TRAP CTL ERR? YES* *SET - TRAP ADDR ERR? YES* *SET - TRAP DATA ERR? YES*	The first seven messages correspond to the seven UUT system errors. If YES is selected, the UUT system error is reported to the operator is it occurs. If NO is selected, the UUT system error is not reported to the operator if it occurs. UUT system errors are described in Table 4F-1.
SET - ENABLE xxxx? NO	This message appears for each μp line that may be enabled or disabled. The letters *xxxx* correspond to the name of the line, such as ready or wait. The actual names and number of lines are documented in the appropriate interface pod manual.
SET - BUS TEST @ aaaa - CHANGE?	When the Bus Test is performed, testing of data lines occurs at the address listed.
SET - RUN UUT @ aaaa - CHANGE?	When the address for the Run UUT operation is allowed to default, this address is used.
SET - TIMEOUT 200 - CHANGE?	The parameter following the word TIMEOUT represents a count of how long the 9010A waits before timing out on an interface pod operation. The parameter may be any decimal number between 0 and 60,000.
SET - EXERCISE ERRORS? YES	If YES is selected, the 9010A displays detected error messages and prompts the operator to loop on the errors. If NO is selected, the errors are not reported to the operator, but error messages are transmitted to the RS-232 if it is connected (without the -LOOP? portion of the message).
SET - BEEP ON ERR TRANSITION? YES	The YES enables the audible beep which sounds whenever an error is detected and reported. The beep also sounds whenever the error is removed.

Source: 9010A Micro-System Troubleshooter Operator Manual, John Fluke Manufacturing Company, Incorporated, June 1981, p. 4M-2.

TABLE 2-4

SET-UP MESSAGES RELATED TO AUX I/F COMMAND

DISPLAY MESSAGE	DESCRIPTION
The following Setup parameters relate to the operation of the optional AUX I/F. If the 9010A detects at power-on that the optional AUX I/F is not installed, the following Setup parameters do not appear in the Setup menu. For a complete description of the following Setup parameters and the AUX I/F, refer to Section 5 of this manual.	
SET - STALL 13-CHANGE?	Any hexadecimal value from 0 to FF may be entered. The corresponding ASCII character is the stall character.
SET - UNSTALL 11-CHANGE?	Any hexadecimal value from 0 to FF may be entered. The corresponding ASCII character is the unstall character.
SET - NEWLINE 00000D0A - CHANGE?	This is the terminator sequence and timing delay between lines.
SET - LINESIZE 79 — CHANGE?	This is the maximum line length for data transmission from the 9010A. The length may be any decimal value from 10 to 255.

NOTE: The microprocessor Enable lines, the Buss Test address, and the "RUN UUT" default address are pod-dependent, and are supplied to the 9010A by the interface pod that is connected.

Source: 9010A Micro-System Troubleshooter Operator Manual, John Fluke Manufacturing Company, Incorporated, June 1981, p. 4M-3.

The "RUNUUT" command allows the user to specify an address to execute an assembly language program on the UUT. At this point, the interface pod will be disconnected from the 9010A. Only a request from the keyboard redirects the attention of the interface pod to the 9010A system.

The previous discussion concerned the 9010A operating in the immediate mode. Simple programs may be written using the test sequence section and arithmetic section of the keyboard. These programs use internal registers in the 9010A and the functions previously discussed to form a guided fault test. Appendix C contains a sample of the 9010A programming language which is a cross between Basic and assembly language.

Conclusion

In summary, the 9010A is simply the command center issuing commands to the interface pod to perform specified tasks. The interface pod contains all the intelligence for the troubleshoot station as well as provides the operating power to the interface pod at 5 volts, 12 volts and -5 volts. Since the 9010A provides power, it protects the interface pod from being damaged from a defective power supply on the UUT.

Starting in 1985, many of the interface pods had additional capabilities which sped up the troubleshoot time. A summary of these capabilities is included in

Chapter IV during the design discussion of the TMS9995 interface pod. A more in-depth discussion is also provided regarding how the troubleshoot commands are executed at the interface pod level as they pertain to the TMS9995 interface pod.

CHAPTER III

RATIONALE FOR THE DEVELOPMENT OF TMS9995 TROUBLESHOOTER

In 1987, the John Fluke Company introduced the 9100 Series personal computer-based troubleshooter. This product was targeted as an engineering aid and testing tool. The cost of one 9100 series unit was as much as 35% of the total investment to date of the troubleshoot equipment installed at Texas Instruments.

Table 3-1 summarizes the troubleshoot equipment currently installed at Texas Instruments' Industrial Systems Division.

This equipment initially provided the Industrial Systems Division with the capability to cover the majority of products manufactured. However, with the introduction of new products, several inadequacies were uncovered in existing troubleshoot and repair centers.

The primary cause of all the inadequacies identified was the inability to purchase interface pods from the John Fluke Company for microprocessors not used extensively in the industry. This required ISD

TABLE 3-1

SUMMARY OF MICRO-SYSTEM TROUBLESHOOTER BASED EQUIPMENT INSTALLED IN THE INDUSTRIAL SYSTEMS DIVISION OF TEXAS INSTRUMENTS

Interface Pods		Adapter Pods		Test Systems	
Type	Quantity	Type	Quantity	Type	Quantity
68000	5	9995 (6802)	3	9010A	8
6802	5	NCS80 (8085)	2	9005A	2
9900	3	80286 (8088)	1		
8088	2				
8085	2				
8051	1				
6809	1				

to construct adapter pods to make an existing interface pod resemble another processor. This technique was only partially successful since the adapter pods developed were not able to run assembly language programs and often could not interface with large scale integrated (LSI) circuits. The problem with interfacing with LSI circuits and running assembly language programs was due to the inability of the adapter pod to mimic the exact timing structure of the microprocessor the user was trying to model. This reintroduced the problem of verifying a new hardware design since in recent years the Industrial Systems Division has incorporated more LSI chips to reduce the chip count and improve the reliability of the designs.

With the introduction of the 9100 Series, the John Fluke Company stated that any new interface pods constructed would not be downward compatible with the 9010A system. This announcement, along with the inability to persuade Fluke to design pods which did not possess high sales potential, required ISD to develop the capability to design interface pods for the troubleshoot stations installed on the manufacturing floor and at international sites. The microprocessor chosen by the troubleshoot and repair teams at the Industrial Systems Division for future interface pods was the TMS9995. This pod was chosen since several new products released in the second quarter of 1988 used the TMS9995 microprocessor.

The remainder of this paper will describe the rationale and process used to design the hardware for the TMS9995 interface pod.

CHAPTER IV

HARDWARE DESIGN OF TMS9995 INTERFACE POD

As previously discussed in Chapter I, it is important that the interface pod look as much like the microprocessor as possible. This is a requirement if the pod is to effectively mesh into the design of the UUT. The hardware was designed to implement the following criteria:

1. must be able to perform buss operations on the UUT with the least amount of distortion to the buss timing;
2. must be able to latch the levels of address, data, control, and status signals;
3. must be able to run assembly language programs from the pod;
4. must be able to run from the clock on the board;
5. must be able to protect against shorted conditions on the buss;
6. must be able to run self diagnostics;
7. must maintain power consumption from the 9010A below 10 watts;

8. must keep the design inside the existing printed wiring board dimensionos of the previous interface pods (10.5" x 5.5").

To maintain the buss operations with a minimum of distortion, the interface pod was designed using the TMS9995 microprocessor. By using the TMS9995, the timing of the control signals, status signals, address buss, and data buss remained the same as the timing structure the UUT was designed to use. Some distortion is present due to the address buffers, data buffers, protection network, and latching circuits. This distortion was kept to a minimum through the introduction of high speed PALs. Unfortunately, the use of these PALs required a large amount of the available power. To compensate for this use of power, Low-Power Schottky was used in areas where the speed of operation was not as critical to the pod performance. This technique proved to be successful in meeting the power consumption goal and board space requirements. Table 4-1 contains a summary of the power used by the interface pod and Appendix E contains the percent of usable board space required to implement the design on the PWB.

The TMS9901 Programmable Systems Interface chip was chosen to serve as the main control chip for the interface pod because of the multiple tasks the chip was capable of performing. A summary of these tasks follow:

TABLE 4-1

SUMMARY OF POWER CONSUMPTION

DEVICE	QUANTITY	POWER CONSUMPTION	
TMS9995	1	0.792	5.50v/180.0ma
TMS9901	1	0.630	5.25v/150.0ma
16L8	4	3.024	5.25v/180.0ma
74LS00	1	0.023	5.25v/ 4.4ma
74LS138	1	0.042	5.25v/ 10.0ma
74LS245	3	1.197	5.25v/ 95.0ma
74LS374	6	1.008	5.25v/ 40.0ma
6264	1	0.462	5.25v/110.0ma
2764	1	0.525	5.25v/125.0ma
OP400	1	0.015	5.25v/ 2.9ma
REF02	1	0.017	12.00v/ 1.4ma
Self-Test Circuitry	-	1.529	5.25v/364.0ma
TOTAL		9.264	

NOTE: The power consumption was calculated by using the maximum voltage times the current developed multiplied by 80%. The assumption to use the figure 80% was based on the principle that most parts would not be operating near their maximum power dissipation.

1. the ability to interface directly with the TMS9995 via the "CRU" buss. This eliminated the requirement of glue logic which saved power and board space;
2. the TMS9901 can function as a programmable I/O port and monitor interrupts simultaneously. This reduced the amount of logic required to interface between the 9010A and the interface pod and reduced the amount of support logic required to run the different troubleshoot functions of the pod.

The price paid for using the TMS9901 was all data transfers made over the "CRU" data buss were bit transfers. Consequently, to process a byte of data takes eight times as long. The "CRU" input and output cycles are shown in Figure 4-1.

Since the interface pod is using the TMS9995 microprocessor, the layout of the memory map is similar to the UUT. Figure 4-2 shows the memory map of the TMS9995. The TMS9995 interface pod breaks this memory map into areas residing on the UUT and on the interface pod. The upper and lower 4K block of memory are able to be mapped either onto the UUT or on the interface pod. The memory between 1000 and EFFF is always mapped onto the UUT. The operating system for the interface pod will reside in both the upper and lower 4K block of memory. This 4K block of

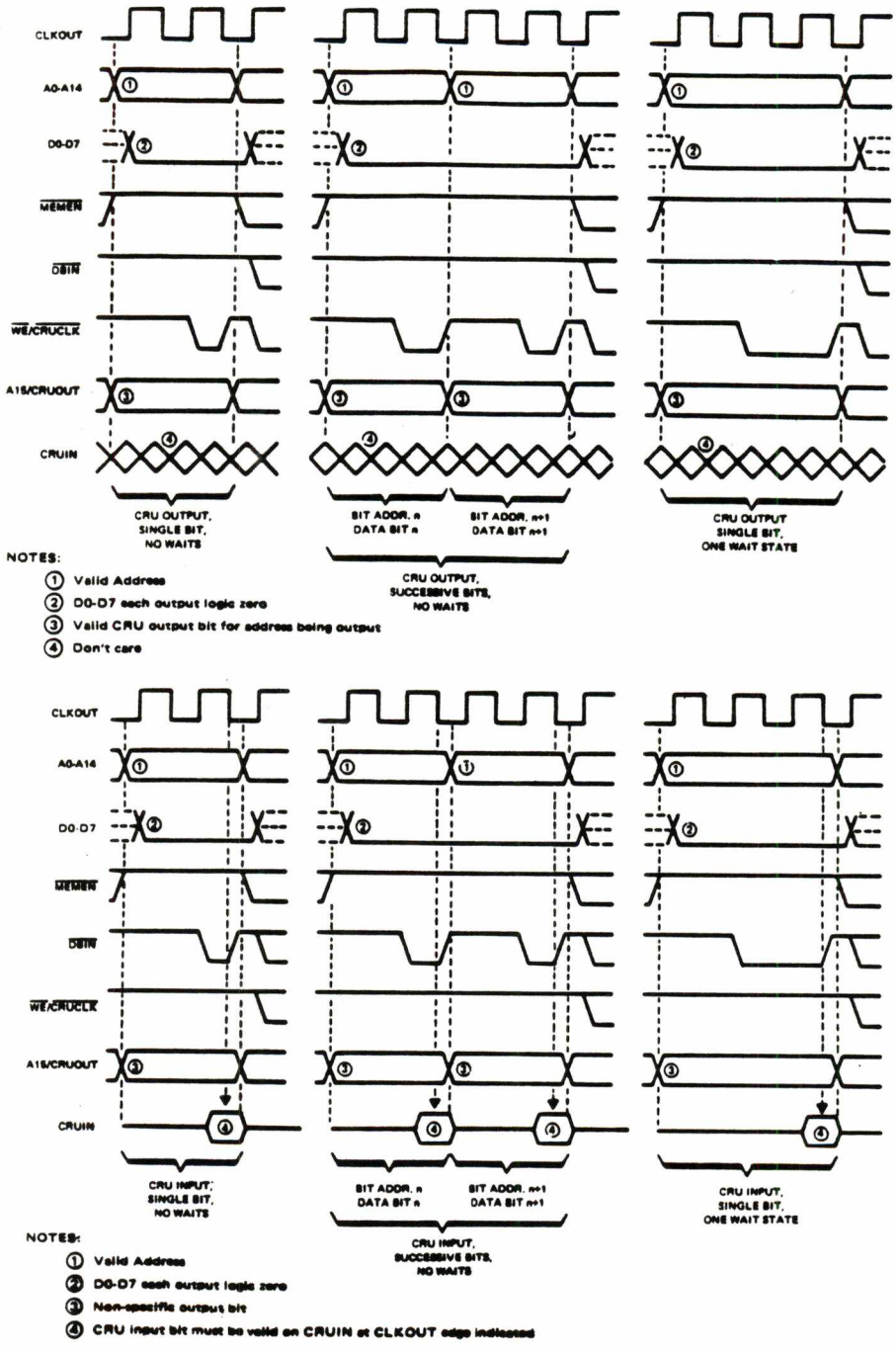
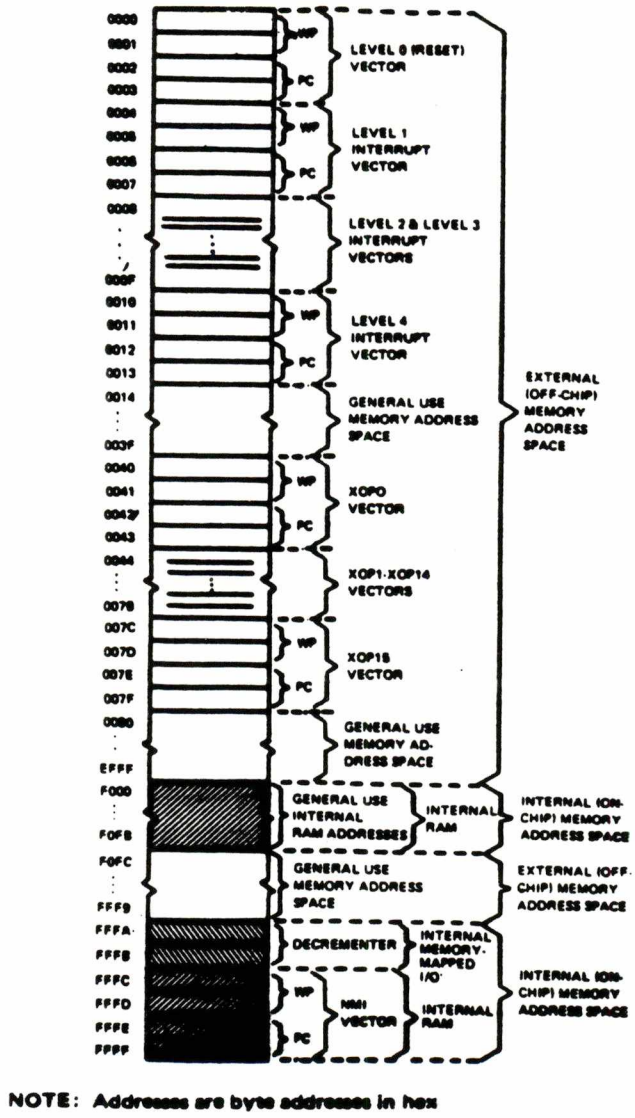


FIGURE 4-1

"CRU" INPUT AND OUTPUT CYCLES

Source: "TMS9995 16-Bit Microcomputer," Texas Instruments Incorporated, August 1982, pp. 22-23.



NOTE: Addresses are byte addresses in hex

FIGURE 4-2

TMS9995 MEMORY MAP

Source: "TMS9995 16-Bit Microcomputer," Texas Instruments Incorporated, August 1982, p. 2.

memory shown in Figure 4-3 is broken down into areas available for programming, interrupt vectors, and prohibited zones.

Interface Pod Memory Zones

The areas marked with an "I" are contained inside the TMS9995 microprocessor. The areas marked with a "*" are not available for programming. If the block of memory mapped between 0000 - 0FFF is mapped onto the UUT, any program placed in the "*" areas will not be accessible to the interface pod. The memory block F200/0200 - FFF9/0FF9 is the area provided for the pod operating system.

The memory map is divided into "EPROM" and "RAM" areas. The "EPROM" will be selected whenever memory blocks F200-F7FF, 0000-00FF, or 0200-07FF are accessed. The RAM will be accessed whenever either memory block F800-FFFF or 0800-0FFF is accessed. Whenever an internal RAM location is accessed in the TMS9995 microprocessor, the data buss will be placed in a high impedance state when the TMS9995 is in a "READ" cycle. However, when operating in internal "RAM" during a "WRITE" cycle, the TMS9995 will drive the data buss.

Since the microprocessor will drive the data buss on an internal "RAM" "WRITE" cycle, the "ERPOM" will not be accessible from the F000-F0FF, to avoid a buss conflict

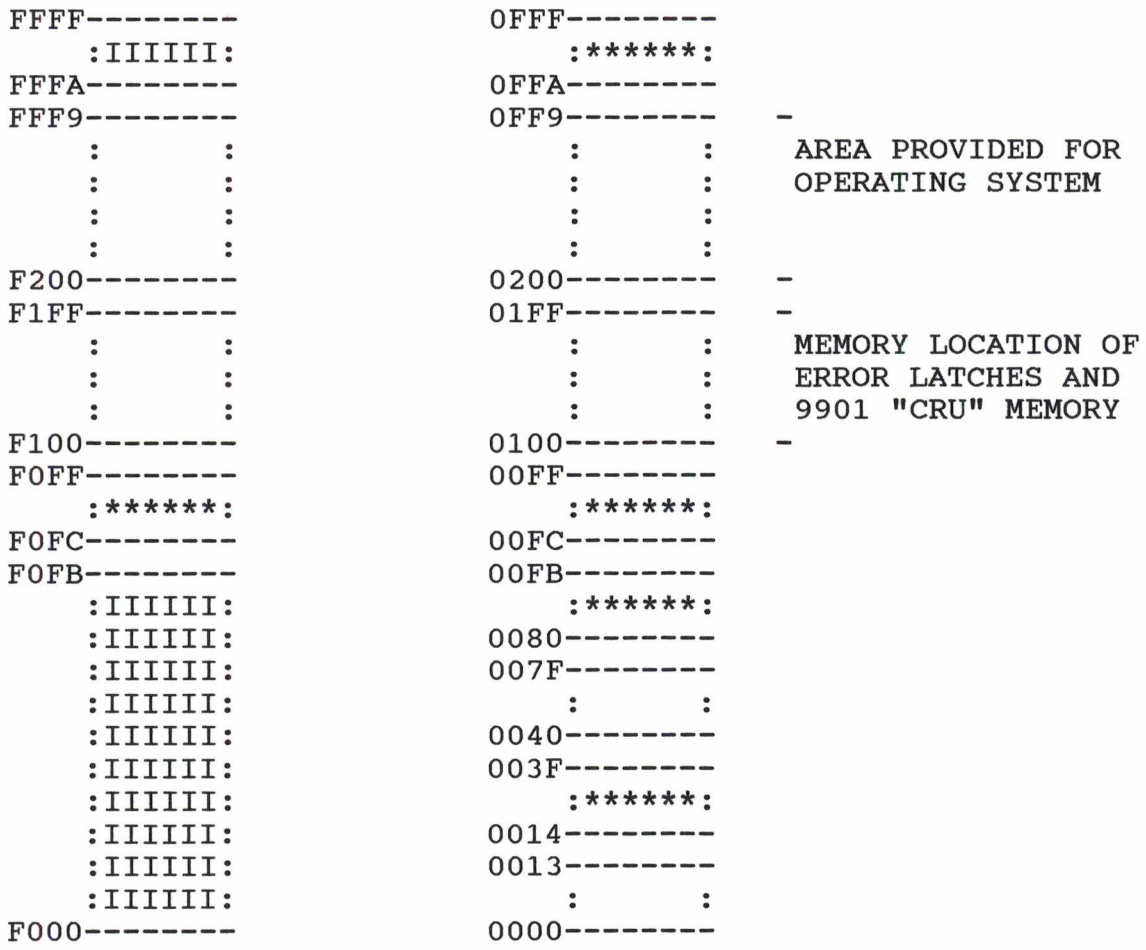


FIGURE 4-3

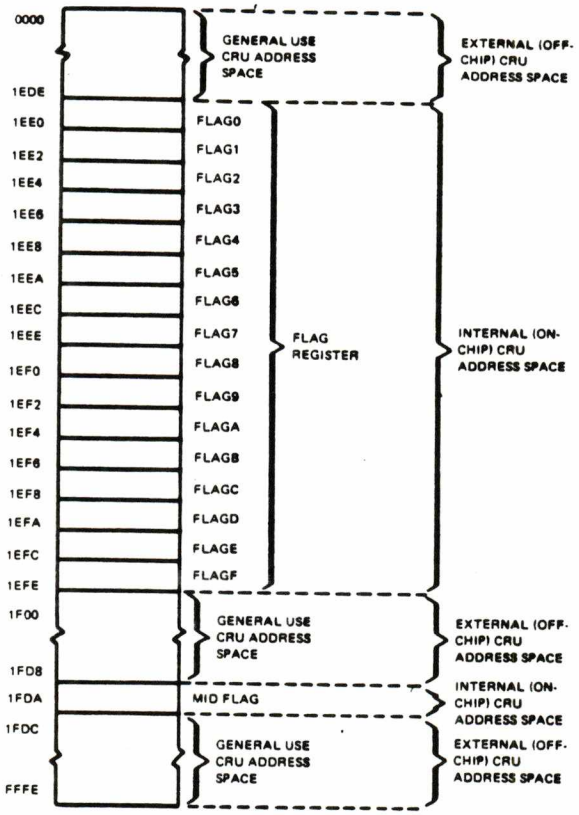
MEMORY BREAKDOWN OF TMS9995 INTERFACE POD

when the pod is operating in the upper 4K block of memory. The error latch registers reside in the following memory locations F100-F10A and 0100-010A. See Page 51 for a description of the latching process and identity of the latched data.

The "CRU" address map of the TMS9995 is shown in Figure 4-4. The only "CRU" device mapped onto the interface pod is the TMS9901 which is mapped from F100-F13F and 0100-013F. This area does not overlap any of the internal "CRU" addresses mapped into the TMS9995.

The "NMI" interrupt is the interrupt which must be used to handle all communications between the pod and the 9010A. This interrupt is chosen for the following reasons:

1. highest priority external interrupt with the exception of reset;
2. interrupt cannot be masked off;
3. the workspace pointer and program counter of the "NMI" interrupt is contained in internal RAM of the TMS9995. Consequently, this position is not affected by the pod relocating memory from the interface pod to the UUT memory map;
4. the remaining interrupts will not be accessible if the interface pod has shifted the lower 4K block of memory onto the UUT.



NOTE: These hex addresses are the software base addresses and are obtained by placing the 15-bit Address Bus CRU bit address in a 16-bit field, left-justifying the 15 bits in the field, and setting the LSB of the field to zero.

FIGURE 4-4

"CRU" ADDRESS MAP

Source: "TMS9995 16-Bit Microcomputer," Texas Instruments Incorporated, August 1982, p. 21.

The events which occur when the 9010A issues a "RESET" to the interface pod is covered in the section entitled "Software/Hardware Reset."

The internal RAM from F000-F0FC should be used to store a program to begin executing code on the UUT in the "RUNUUT" mode. The program must be written in this area because the entire memory map of the interface pod is mapped onto the UUT with the exception of the internal memory of the TMS9995. If the "NMI" interrupt has been set to vector into the internal RAM area, the software engineer should set the "NMI" interrupt to vector into an external memory space before beginning to run assembly code on the UUT. This eliminates any possibility of accidentally running code remaining in the internal RAM. The software programmer must build his program between X200H to XFF9. This area will occupy the same space in both the upper and lower memory blocks.

The user may request an access of the UUT address space in either the upper or lower 4K block. Consequently, the software must be able to operate in either area. The code should be written in a manner to allow a context shift to the upper byte from the lower byte or from the lower byte to the upper byte.

U3 is tasked with decoding the memory and "CRU" operations of the pod. The two upper bytes of the address buss (A0-A7), two status flags ("FLOW" and "FHIGH") from

the TMS9901, and "MEMEN" determine whether the pod is addressing memory or "CRU" on the interface pod. PAL equations and simulation runs can be found in Appendix B.

Programmable Systems Interface

Since the TMS9901 is able to monitor interrupts and has sixteen programmable I/O points it was chosen to serve as the main control chip in the interface pod. The functions of the chip are broken into the five areas of control:

1. interface between the interface pod and the 9010A;
2. sets the two flags ("FLOW" and "FHIGH") governing the position of the upper and lower 4K memory blocks;
3. sets the three flags ("DC1," "DC2" and "DC3") which determine which control lines will be driven to locate "stuck at faults" on the control lines;
4. sets the two flags which will enable the status lines "READY" and "HOLD;"
5. provides a monitor of the logic states of the status lines.

As previously discussed, the TMS9901 is located in "CRU" memory from F100-F13F and 0100-013F. The "CRU" address of the 32 "CRU" bits and their function is shown in Table 4-2.

TABLE 4-2

9901 PIN DEFINITIONS AND LOCATIONS IN INTERFACE POD
"CRU" MEMORY MAP

ADDRESS	DESCRIPTION
F13E/013E	"DC1"
F13C/013C	"DC2"
F13A/013A	"DC3"
F138/0138	"ENHOLD"
F136/0136	"ENRDY"
F134/0134	"FLOW"
F132/0132	"FHIGH"
F130/0130	"PODSTAT"
F12E/012E	D7
F12C/012C	D6
F12A/012A	D5
F128/0128	D4
F126/0126	D3
F124/0124	D2
F122/0122	D1
F120/0120	D0
F11E/011E	SOFTWARE RESET
F11C/011C	MASK OFF
F11A/011A	MASK OFF
F118/0118	MASK OFF
F116/0116	MASK OFF
F114/0114	MASK OFF
F112/0112	MASK OFF
F110/0110	MASK OFF
F10E/010E	MASK OFF
F10C/010C	"HOLD" (INT6)
F10A/010A	"INT4" (INT5)
F108/0108	"INT1" (INT4)
F106/0106	"NMI" (INT3)
F104/0104	"RESET" (INT2)
F102/0102	"MAINSTAT" (INT1)
F100/0100	CONTROL BIT

Communication Buss Between 9010A and Interface Pod

The TMS9901 communicates with the 9010A over an eight-bit data buss with "MAINSTAT" and "PODSTAT" serving as the handshaking lines. Figure 4-5 shows the timing diagram of the process. In a write operation from the 9010A to the interface pod, "MAINSTAT" will be pulled low by the 9010A stating that either data or a command has been placed on the eight-bit data buss. "MAINSTAT" will activate "INT1" on the TMS9901. Whereupon, the TMS9901 will output the identity of the interrupt on the lines "IC0-IC3" as well as issue "INTREQ." At this point, the TMS9995 will receive a "NMI" interrupt from U20 which was generated by the "INTREQ" signal sent by the TMS9901. The interface pod has two ways of determining the identity of the interrupt which will be discussed in the following paragraph.

Figure 4-6 shows the switching characteristics of the TMS9901. The "NMI" service routine may be set up in either of the following two ways:

1. the TMS9995 may read "INT1-INT6" on the TMS9901 which correspond to "CRU" locations F100-F10C or 0100-010C. This method of determining the state of the interrupt is only effective if the interrupt remains active long enough for the TMS9995 to read it;

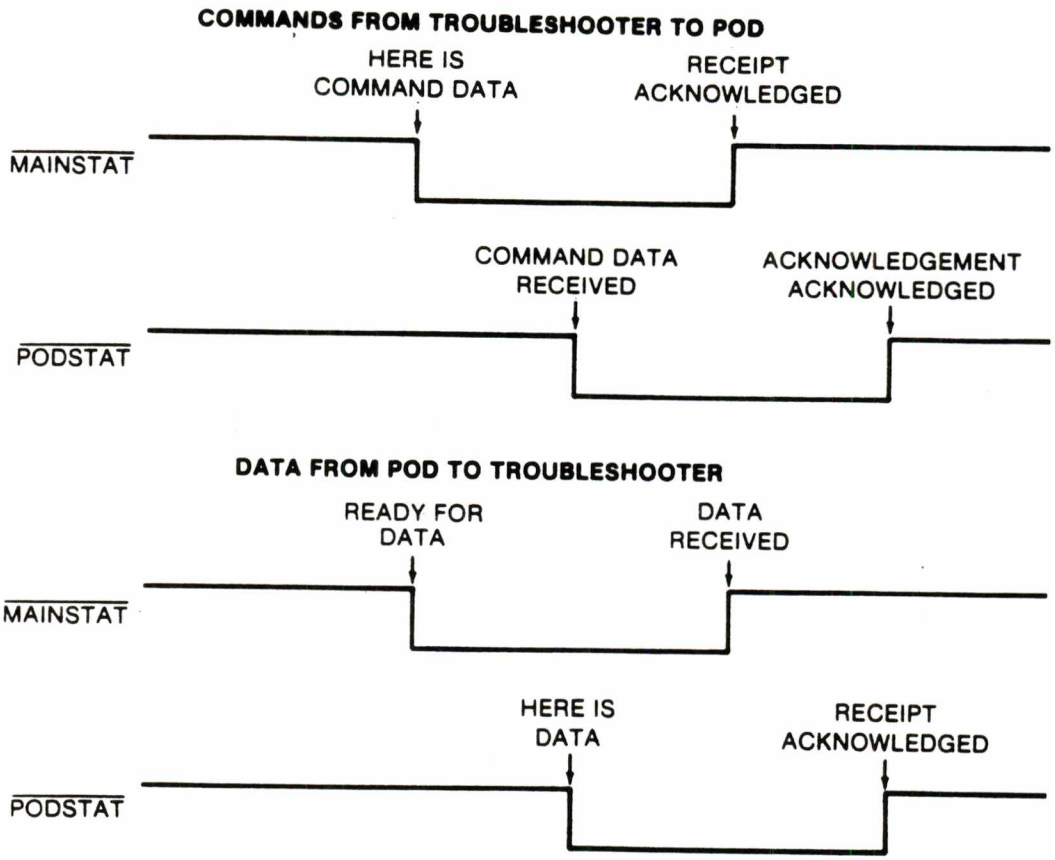
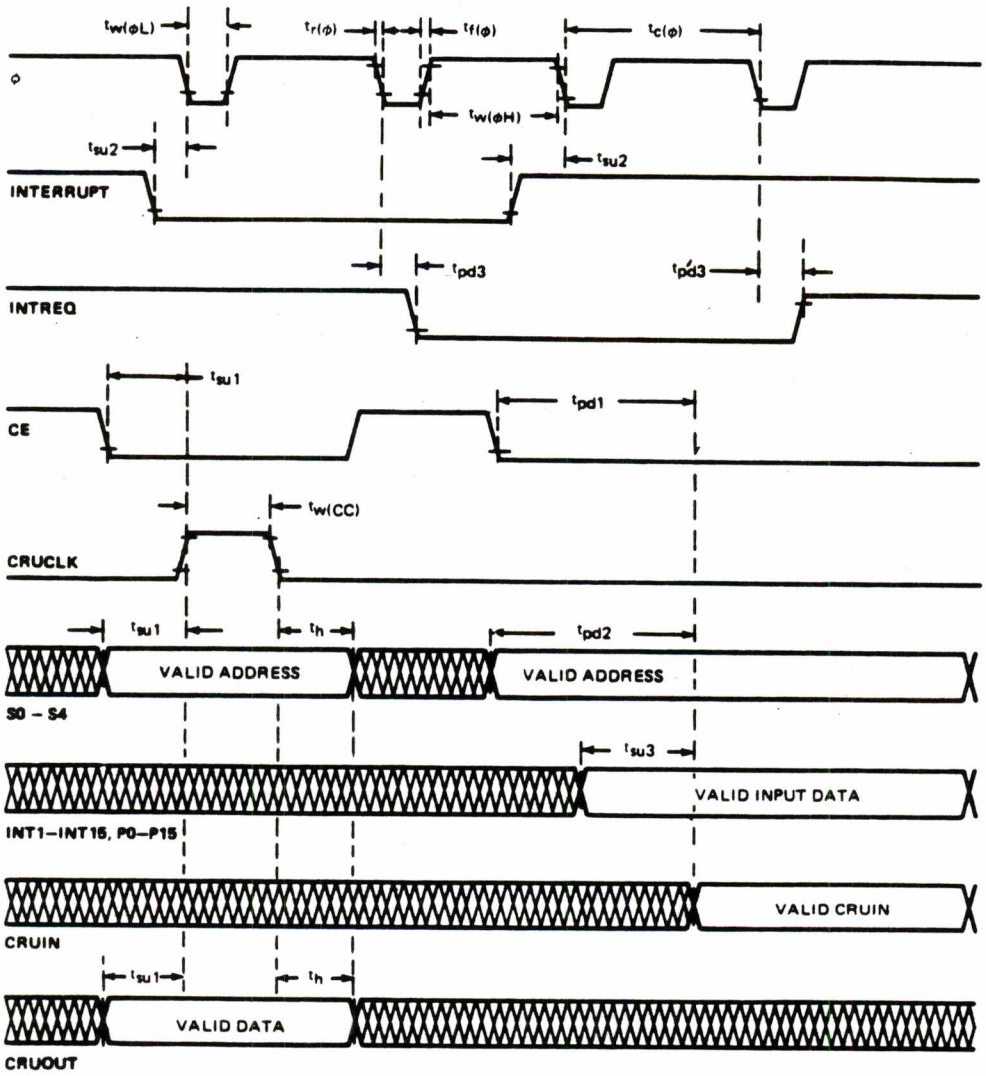


FIGURE 4-5

HANDSHAKING PROCESS

Source: "9000A-6802 Interface Pod Instruction Manual," John Fluke Manufacturing Company, May 1982, p. 4-8.

	PARAMETER	TEST CONDITION	MIN	TYP	MAX	UNIT
t_{pd1}	Propagation delay, $\overline{CE}$ to valid CRUIN	C_L = 100 pF			300	ns
t_{pd2}	Propagation delay, S0-S4 to valid CRUIN	C_L = 100 pF			320	ns
t_{pd3}	Propagation delay, $\overline{\phi}$ low to valid $\overline{INTREQ}$, IC0-IC3	C_L = 100 pF			110	ns
t_{pd}	Propagation delay, $\overline{CRUCLK}$ to valid data out (P0-P15)	C_L = 100 pF			300	ns



NOTE 1: ALL TIMING MEASUREMENTS ARE FROM 10% and 90% POINTS.

FIGURE 4-6

9901 SWITCHING CHARACTERISTICS

Source: "TMS9901 Programmable Systems Interface Manual," Texas Instruments Incorporated, July 1978, p. 20.

2. the TMS9995 may read the identity of the interrupt which is latched in U13 by reading memory location F106 or 0106. "INTREQ" was delayed by approximately 30ns to allow adequate setup time for the 74LS374.

After the TMS9995 has determined that "MAINSTAT" has been pulled low by the 9010A, the TMS9995 will read the data from the TMS9901 "CRU" locations F120-F12E or 0120-012E. Next, the TMS9995 will write a zero to location F130 or 0130 to pull "PODSTAT" low. When the 9010A sees "PODSTAT" low, the 9010A will remove the data from the data buss as well as pull "MAINSTAT" high. When the TMS9995 sees "MAINSTAT" go high by reading the state of the TMS9901, the TMS9995 will write a high to the "PODSTAT" "CRU" bit. When "PODSTAT" goes high the current "WRITE" cycle has been completed.

The handshaking procedure between the interface pod and the 9010A during a "READ" cycle is similar. The cycle begins with the TMS9995 writing a zero into the "PODSTAT" "CRU" bit and the corresponding data byte in bits "D0-D7" on the 9010A data buss. The 9010A will see the "PODSTAT" line pulled low and read the data from the buss. After the 9010A has read the data, the 9010A will pull "MAINSTAT" low to flag the interface pod that it has received the data.

The TMS9995 will read the status of the "MAINSTAT" "CRU" bit. When this bit goes low, the TMS9995 will begin

the reset process of the TMS9901 by writing a one to the control bit in location F100 or 0100 which puts the TMS9901 into the clock mode. Next, the TMS9995 will write a zero to F11E or 011E which will force a software reset on the TMS9901. Once the TMS9901 has been reset, the software engineer must write a zero into "CRU" control bit returning the TMS9901 to the interrupt mode of operation.

Software/Hardware Reset

The software reset will set all I/O points to the input mode. This procedure must be installed in the software to avoid damage to the the interface pod or to the 9010A. The TMS9901 I/O ports remain in the output mode when instructed to output data. To reconfigure an I/O point to the input mode, the TMS9901 must receive a hardware or software reset. The software engineer must reconfigure the TMS9901 after it has been reset. The following procedures must be performed after a reset:

1. If the "HOLD" or "READY" had been enabled the software engineer must write a zero to the appropriate "CRU" bits. After a reset, the enable lines for these signals are pulled high to prevent "HOLD" and "RESET" from reaching the TMS9995;
2. After a hardware reset, the TMS9901 will come up with all interrupts masked off. The software

engineer must enable "INT1" and the remaining interrupts if the user had requested them to be monitored.

Memory Location Flags

The memory location of the upper and lower 4K block of memory is determined by U13 which uses the "FLOW" and "FHIGH" flags generated by the TMS9901. The effects these flags have on the memory map has been previously covered in section concerning the memory map of the interface pod. However, two important points must be brought out concerning the reaction of these flags during a hardware and software reset. During either reset the flags will be pulled high by 1K pullup resistors to assure the state of the flags. The default state is to use the onboard memory of the pod.

When the user requests a "RUNUUT" operation, the software must set both flags low. By setting both flags low, the entire memory map will be located onto the UUT with exception of the internal RAM of the TMS9995. The PAL in location U13 will enable the output buffers for offboard operation. As previously mentioned, the software should have planted a small program to vector to the location the user had requested for the interface pod to execute code. With both flags low, the PAL in location U20 will allow all interrupts through to the TMS9995. When the user requests

another operation after a "RUNUUT" command, the 9010A will issue a hardware reset to the interface pod which will reset the TMS9901 and the TMS9995.

After reset has been issued by the 9010A, the "FLOW" and "FHIGH" flags will be pulled high by 1K pullup resistors. The TMS9995 will go to location 0000 and 0002 and retrieve the program counter and workspace pointer for reset. The memory on the interface pod should still be intact with the exception of the internal RAM area which was exposed to the UUT operating system while the interface pod was operating on the UUT as the microprocessor. Consequently, the software must restore the "NMI" vectors and the TMS9901 as previously discussed in the section entitled "Software/Hardware Reset."

After the 9010A has reset the interface pod, any further communication during that command will be handled using the handshake operation previously described in the section entitled "Communication Buss Between 9010A and Interface Pod."

"READY" and "HOLD" Operations

The user is given the option to enable "READY" and "HOLD" to the TMS9995 on the interface pod. The signal "READY" posed no problem to the design since it only requested additional clock cycles. The "HOLD" operation required the interface pod to respond like a processor

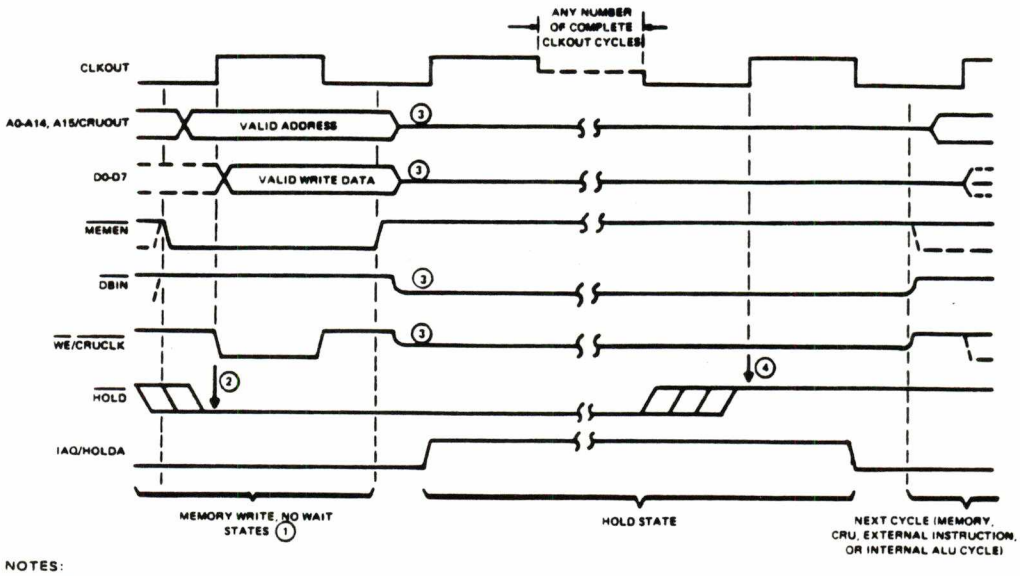
placed on "HOLD." The "HOLD" state timing diagram is shown in Figure 4-7. When the "HOLD" function is not enabled, "IAQ/HOLDA" is always held in the low state. This will keep all "DMA" devices off the data, address, and control busses. When the "HOLD" function is enabled, U20 will pass the "HOLD" request to the TMS9995. The TMS9995 will complete the instruction currently being executed and will "TRI-STATE" its address buss, data buss, "WE," and "DBIN." After these lines are "TRI-STATED," the TMS9995 will issue a "HOLD ACKNOWLEDGE" signal. This signal will be used by U22 to "TRI-STATE" the output drivers.

Under normal conditions, the buss drivers are "TRI-STATED" since the pod operating system should only access the "UUT" when the its executing a command from the 9010A. However, if the pod is running in the "RUNUUT" mode of operation, the output drivers would always be enabled. Consequently, the drivers had to respond to a "HOLD" cycle in the same manner as a TMS9995 data and address buss driver.

The worst case timing delay from "IAQ/HOLDA" to the buffers going "TRI-STATE" is 38ns. After "HOLD" has been released the buffers will be reactivated no later than 50ns from the removal of "IAQ/HOLDA."

Troubleshooting Control Lines

The control lines can be driven by the pod as an aid in troubleshooting. The three lines "DC1-DC3"



NOTES:

(1) Cycle before the hold state could have been memory (with any number of wait states), CRU, external instruction, or internal ALU

(2) $\overline{HOLD}$ must be valid at last low-to-high CLKOUT transition of a cycle for next low-to-high CLKOUT transition to begin a hold state

(3) In high-impedance mode (output drivers)

(4) Next cycle will begin after first low-to-high CLKOUT transition at which $\overline{HOLD}$ is high

FIGURE 4-7

TMS9995 "HOLD" STATE

Source: "TMS9995 16-Bit Microcomputer," Texas Instruments Incorporated, August 1982, p. 11

generated from the TMS9901 instruct the PALs U22 and U21 to drive certain control lines as show in Table 4-3. The function on the 9010A which would request this operation is the control line section referenced in Chapter II.

Latch Circuitry

The error latch circuitry latches data on the rising edge of "ERRLCH" originating from U3. To produce the clock for the SN74LS374's, the "CLKOUT" signal is "ANDed" with external chip select ("EXTCS"). "EXTCS" is generated from the addresses mapped on the UUT and the flags "FLOW" and "FHIGH." This allows the pod to latch the levels of the address buss, data buss, control and status lines. To minimize the amount of propagation time in this clock, a TIBPAL16L-10C was used to limit the propagation delay to 10ns. A diagram of the timing characteristics and latching process is shown in Figures 4-8 and 4-9.

As long as the address is in the UUT memory space the data will be latched. To minimize the amount of chips required to latch the state of the the address and data buss during a "READ" and "WRITE" operation, the software will be constructed to execute the "READ/WRITE" commands from the 9010A in byte format. Therefore, two byte operations must be performed to implement a word instruction. However, the interfacing between the 9010A

TABLE 4-3

TROUBLESHOOTING CONTROL LINES

DC1	DC2	DC3	LINES DRIVEN
1	1	1	No Lines Driven
1	1	0	"MEMEN" Line Driven
1	0	1	"DBIN" Line Driven
1	0	0	"WE/CRUCLK" Line Driven
0	1	1	"CLKOUT" Line Driven
0	1	0	"CRUIN" Line Driven
0	0	1	"IAQ/HOLDA" Line Driven
0	0	0	All Control Lines Driven

TIMING REQUIREMENTS OVER RECOMMENDED OPERATING CONDITIONS

	PARAMETER	MIN	NOM	MAX	UNITS
t_{su1}	Setup time, READY prior to ↓ CLKOUT (memory cycles)	100			ns
t_{h1}	Hold time, READY after ↓ CLKOUT (memory and CRU cycles)	0			ns
t_{su2}	Setup time, Data prior to ↓ CLKOUT	65			ns
t_{h2}	Hold time, Data after ↓ CLKOUT	0			ns
t_{su3}	Setup time, CRUIN prior to ↓ CLKOUT	100			ns
t_{h3}	Hold time, CRUIN prior to ↓ CLKOUT	0			ns
t_{su4}	Setup time, READY prior to ↓ CLKOUT (CRU cycles)	100			ns
t_{su5}	Setup time, $\overline{HOLD}$ prior to ↑ CLKOUT	125			ns
t_{su6}	Setup time, $\overline{RESET}$ and $\overline{NMI}$ prior to ↓ CLKOUT	140			ns
t_{WL4}	Pulse width, Interrupt inputs	$\frac{1}{2}t_{c2}$			ns
t_{f3}	Fall time, $\overline{INT1}$, $\overline{INT4/EC}$ inputs			15	ns
t_{WH3}	Pulse width, $\overline{EC}$ input high	160			ns
t_{WL5}	Pulse width, $\overline{EC}$ input low	160			ns
t_{c3}	Cycle time, $\overline{EC}$ input	$3t_{c2}$			ns

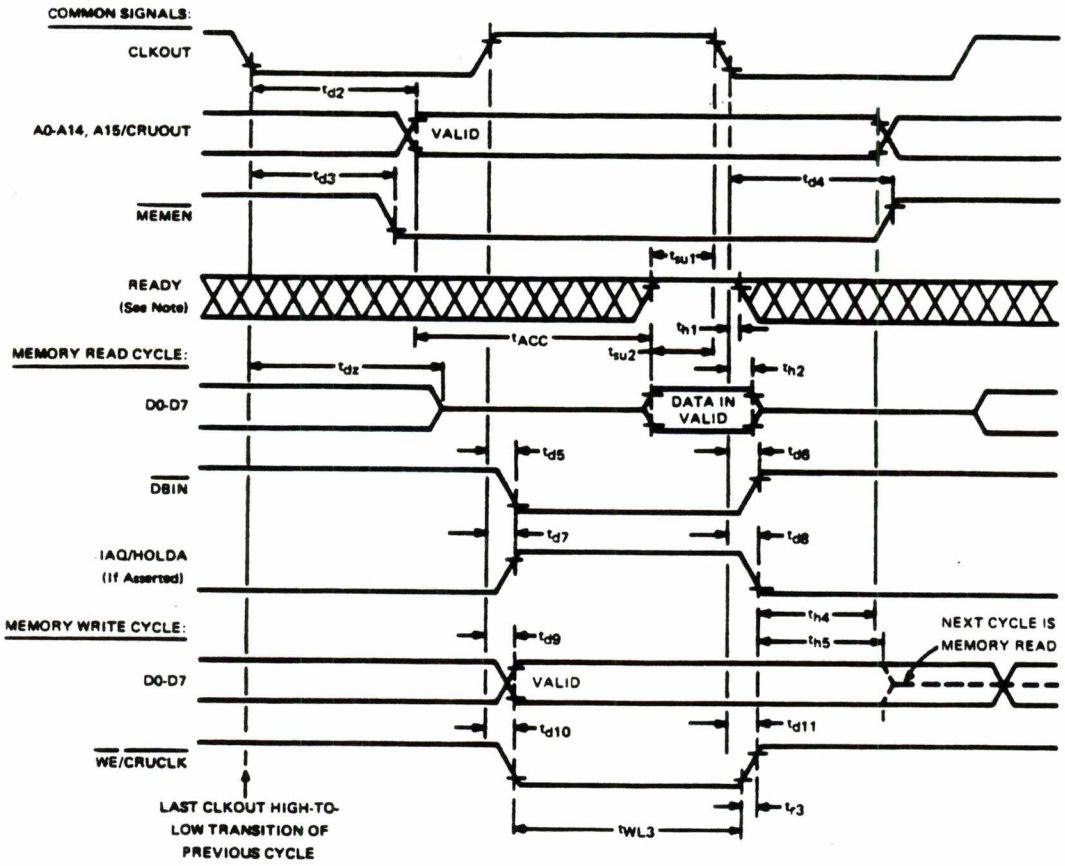
SWITCHING CHARACTERISTICS OVER RECOMMENDED OPERATING CONDITIONS

	PARAMETER	TEST CONDITIONS	MIN	TYP	MAX	UNITS
t_{d1}	Delay time, CLKIN ↓ to CLKOUT ↓	R_1 = 2.4 kΩ R_2 = 24 kΩ C_L = 100 pF	5	150		ns
t_{c2}	CLKOUT Cycle time — External clock source			$4t_{c1}$		ns
	CLKOUT Cycle time — Internal oscillator XTAL freq = f_{xx}			$4/f_x$		ns
t_{r2}	Rise time, CLKOUT output			20	30	ns
t_{f2}	Fall time, CLKOUT output			10	20	ns
t_{WH2}	Pulse width high, CLKOUT output			$\frac{1}{2}t_{c2}-t_{r2}$		ns
t_{WL2}	Pulse width low, CLKOUT output			$\frac{1}{2}t_{c2}-t_{f2}$		ns
t_{d2}	Delay time, ↓ CLKOUT to address valid		$\frac{1}{4}t_{c2}$		$\frac{1}{4}t_{c2}+45$	ns
t_{d3}	Delay time, ↓ CLKOUT to $\overline{MEMEN}$ low		$\frac{1}{4}t_{c2}$		$\frac{1}{4}t_{c2}+40$	ns
t_{d4}	Delay time, ↓ CLKOUT to $\overline{MEMEN}$ high		$\frac{1}{4}t_{c2}$		$\frac{1}{4}t_{c2}+50$	ns
t_{d5}	Delay time, ↑ CLKOUT to $\overline{DBIN}$ low		0		40	ns
t_{d6}	Delay time, ↓ CLKOUT to $\overline{DBIN}$ high		0		50	ns
t_{d7}	Delay time, ↑ CLKOUT to IAQ/HOLDA high		0		40	ns
t_{d8}	Delay time, ↓ CLKOUT to IAQ/HOLDA low		0		50	ns
t_{d9}	Delay time, ↑ CLKOUT to data output valid		0		40	ns
t_{d10}	Delay time, ↑ CLKOUT to $\overline{WE/CRUCLK}$ low		0		40	ns
t_{d11}	Delay time, ↑ CLKOUT to $\overline{WE/CRUCLK}$ high		0		50	ns
t_{r3}	Rise time, $\overline{WE/CRUCLK}$ outputs			20	50	ns
t_{ACC}	Access time, memory read cycles			$\frac{3}{4}t_{c2}-135$		ns
t_{h4}	Hold time, address and CRUOUT outputs		$\frac{1}{4}t_{c2}-40$			ns
t_{h5}	Hold time, data output		$\frac{1}{4}t_{c2}-40$			ns
t_{WL3}	Pulse width low, $\overline{WE/CRUCLK}$ output		$\frac{1}{2}t_{c2}-40$			ns
t_{dz}	Output disable time				$\frac{1}{4}t_{c2}+60$	ns

FIGURE 4-8

TIMING CHARACTERISTICS

Source: "TMS9995 16-Bit Microcomputer," Texas Instruments Incorporated, August 1982, p. 56.



NOTE: Cycle shown is for no wait states (with wait states, CLKOUT cycles are added, but the switching parameters do not change).

FIGURE 4-9

TMS9995 MEMORY INTERFACE TIMING

Source: "TMS9995 16-Bit Microcomputer," Texas Instruments Incorporated, August 1982, p. 57.

and the interface pod is already slow, consequently the delay should not be noticed. Appendix F contains a list of instruction execution times.

The level of the address buss, data buss, status and control lines can be read at the memory addresses listed in Table 4-4. It should be noted that U3 decodes the addresses in the error registers and provides the enable signal to U4 ("ERREG").

Since the interface pod is not set up to service any of the devices on the UUT which issue interrupts, the PAL in location U20 blocks all interrupts from the board. The interrupts from the board are enabled only in the "RUNUUT" mode of operation. Two ways exist for monitoring the state of the interrupts:

1. "READ" the status latch at the memory location referenced in Table 4-4;
2. "READ" the levels of "INT2-INT6" on the TMS9901 (reference Table 4-2, Page 41 for "CRU" Memory Locations).

Protection Networks

Protection against shorted buss lines to potentially damaging voltages is provided by 1N751A Zener diodes (5.1v +/- 5%) in line with 250 OHM 1/4 watt resistors. This will not allow the voltage to go above 5.36v or below -1.0v. This will protect the

TABLE 4-4

MEMORY LOCATION OF ERROR REGISTERS

Memory Address	Latch Data
F100/0100	UPPER ADDRESS BYTE
F102/0102	LOWER ADDRESS BYTE
F106/0106	DATA BYTE
F108/0108	CONTROL SIGNALS
F10A/010A	STATUS SIGNALS

"TRI-STATE" outputs of the 74LS245's and the PALs in locations U20, U21, and U22 whose outputs cannot go above 5.5v without risking damage to the chip. The 74LS374's inputs can withstand up to 7v on their inputs.

The maximum positive voltage the design can withstand is 12v. The maximum negative voltage the design can withstand is -7v. The 250 ohm resistor provides a current limit to the protection circuit. This resistor was chosen to minimize the voltage drop due to the protection circuitry of the board. The drop is 0.25v per milliamp of current. The OP400's in the power failure detection circuit are protected with a 1N750A Zener diode.

The clock inputs to the TMS9995 microprocesser are protected by two 1N751A zener diodes. This was done to protect the clock inputs from a shorted condition. The TMS9995 is designed to be driven by an external clock or by a crystal. The external clock will drive into the "XTAL2" input with "XTAL1" not connected. If a crystal is used, it will be connected between "XTAL1" and "XTAL2".

Power Failure Detection Circuit

The power failure detection circuit on Page 5 of the schematic in Appendix A, uses a "REF02" precision voltage source and 1% resistors to check the integrity of the power buss on the UUT. The voltage is divided in half to provide a 2.5v signal. Two OP400's are setup as an

upper and lower voltage comparator. If the voltage on the board is above 5.25v or below 4.75v the "BAD/PS" signal will be latched into the status latch at U7. However, if power falls below 3.0v the 9010A will be informed directly on the "POWFAIL" line.

Self-Test Socket

The self-test socket provides a load for the interface pod to toggle the data buss, address buss, status, and control lines. This assures the user that the interface pod is functioning properly. The load resistors are 220/330 OHM termination resistors. The pod will determine whether it is performing a self test by reading the status register at U15. If data bit 7 is high, the pod is flagged to perform a self-test.

CHAPTER V

SOFTWARE DESIGN GUIDELINES

The 9010A is set up to handle a maximum of 32 address lines and 32 data lines. Since the TMS9995 interface pod is set-up as a 16 address bit machine, the upper 16 address bits are available as software flags to the interface pod. Since these upper 16 address bits are not used by the TMS9995, they will be used to identify "CRU" operations and quick routines. Quick routines are software programs written by the software engineer which avoid using the handshaking procedure described in the section of Chapter IV entitled "Communication Buss Between 9010A and Interface Pod." The software engineer will have to write a routine to differentiate between a memory access and a "CRU" memory access.

"CRU" Memory Access

The operating system of the interface pod should be set up to recognize the address block from 1X0000-1XFFFF as the "CRU" memory block. This block of memory will represent the same memory block as shown in Figure 4-3 on

page 36. The operating system will break this address into the following format. The leading "one" will identify the address as a "CRU" operation. The next four bits will determine the number of "CRU" bits to be read or written. Table 5-1 shows the relationship of these four bits to the quantity of "CRU" data bits the user is requesting to be read or written.

The software should prohibit a "CRU" "WRITE" command into "CRU" bit address 1EE2. If this "CRU" bit is set the Decrementer on the TMS9995 will be enabled and can cause a Level 3 interrupt. The interface pod may not be set up to service an interrupt 3. Furthermore, the interface pod may have transferred the lower 4K block of memory onto the UUT making any vectors on the interface pod unaccessible.

"Quick" Routines

The software engineer can use the remaining upper address bits as flags to execute assembly language test programs. These assembly language programs will increase the test speed of the interface pod dramatically. This is due to the elimination of the handshaking procedure outlined in the section concerning Communication Between the 9010A and the Interface Pod in Chapter IV on page 42. The software engineer can construct the request for a "Quick" test in a manner similar to the method shown in Table 5-2.

TABLE 5-1

"CRU" READ/WRITE OPERATIONS AT "CRU" ADDRESS 0000
EFFECT OF QUANTITY BITS

ADDRESS	CRU LOCATIONS ACCESSED	RANGE ACCESSED
100000	16	0000-001E
110000	1	0000-0000
120000	2	0000-0002
130000	3	0000-0004
140000	4	0000-0006
150000	5	0000-0008
160000	6	0000-000A
170000	7	0000-000C
180000	8	0000-000E
190000	9	0000-0010
1A0000	10	0000-0012
1B0000	11	0000-0014
1C0000	12	0000-0016
1D0000	13	0000-0018
1E0000	14	0000-001A
1F0000	15	0000-001C

TABLE 5-2

EXAMPLE OF "QUICK" TEST ROUTINES
"RAM TEST"

REQUEST	ADDRESS	DESCRIPTION OF INFORMATION
WRITE	20XXX0	UPPER TEST ADDRESS
WRITE	20XXX2	LOWER TEST ADDRESS
READ	20XXX4	STATUS OF TEST
READ	20XXX6	LOCATION OF FAILURE
READ	20XXX8	DATA EXPECTED
READ	20XXXA	DATA RECEIVED

An example of the status information returned from the "Quick" routine is shown in Table 5-3. This format shows the status of a "Quick" routine during execution. It shows the location of the test with 4K resolution.

The techniques outlined in Tables 5-2 and 5-3 can be applied to a "ROM" checksum calculation routine and "CRU" memory test routine. The software must restrict the users access to prohibited memory to avoid writing into areas used by the interface pod. While executing the "Quick" routines the interface pod is still running out of the upper or lower 4K block of memory mapped on the interface pod.

Table 5-4 contains the list of commands the 9010A will transmit to the interface pod. All functions outlined in Chapter II are implemented using these commands. An example of the codes exchanged during a "READ" command is shown in Table 5-5.

The failure codes returned from the interface pod to the 9010A area summarized in Table 5-6. After a failure code is received by the 9010A from the interface pod, the 9010A will send the command "OC" which is a request for more data. The 9010A will expect the error data to be returned in the format shown in Table 5-7. Appendix G contains the set-up sequence which must be transmitted by the software engineer after a pod reset.

TABLE 5-3

STATUS INFORMATION FOR "QUICK" ROUTINES
"RAM TEST"

DATA RETURNED	INFORMATION PROVIDED
B0ZZ	Currently performing a "RAM" decode test on the 4K block ZZ
B1ZZ	Currently performing a "RAM" memory bit test on the 4K block ZZ
C0ZZ	"RAM" test passed
F0ZZ	"RAM" decode failure detected at the 4K block ZZ
F1ZZ	"RAM" memory failure detected at the 4K block ZZ

TABLE 5-4

9010A COMMAND LIST AND ADDRESS MODIFIERS

COMMAND	FUNCTION
X1	"RUNUUT" at a specified address
02	"SYNC" probe
X3	"WRITE" at a specified address
04	Not assigned a function
X5	"READ" at a specified address
06	Enable inputs ("READY" or "HOLD")
X7	Change data bits during a "Buss Test"
08	Read Status
X9	Change address bits during a "Buss Test"
0A	"WRITE" to a control line*
AB	Request identification information from interface pod
0C	Send more information about failure
XD	I/O test at an address or addresses
0E	Not assigned a function
0F	Not assigned a function
X=0**	No address byte changed
X=1	Address byte 0-7 changed
X=2	Address byte 8-15 changed
X=3	Address byte 16-23 changed
X=4	Address byte 24-31 changed

*See the discussion on Troubleshooting Control Lines on pages 49-50 of Chapter IV.

**The "X" is an indicator to which address byte has changed.

TABLE 5-5

EXAMPLE OF "READ" COMMAND

9010A TRANSMITS	POD SENDS	
F5*	--	
00	--	
00	--	
04	--	
00	--	
	xx**	data bits 08-15
	xx	data bits 00-07
	ec	error code

*The "F" means four bytes of address will follow. The "5" means the pod must execute a "READ" at the address sent.

**The pod reads address 4000 and returns the data in the this format.

TABLE 5-6

SUMMARY OF ERROR CODES

FAILURE CODE	DESCRIPTION OF FAILURE
00	No Error Detected
01	Data Error
02	Address Error
04	Control Error
08	Active Force Line
10	Active Interrupt
20	Illegal Address
40	Pod Failure
80	Power Failure

TABLE 5-7

ERROR STRING RETURNED TO 9010A

BYTE OF DATA	DESCRIPTION OF INFORMATION
AA	Error Type*
BB	Error in Address Bits (24-31)**
CC	Error in Address Bits (16-23)
DD	Error in Address Bits (08-15)
EE	Error in Address Bits (00-07)
FF	Error in Control Bits (08-15)
GG	Error in Control Bits (00-07)
HH	Error in Status Bits (08-15)
II	Error in Status Bits (00-07)
JJ	Error in Data Bits (08-15)
KK	Error in Data Bits (00-07)
LL	Error Type***

*Indicates error type: status line, control line, address and data codes are placed here.

**A one in a bit identifies the location of the error.

***Indicates error type: illegal address, force line, active interrupt and power failure.

CHAPTER VI

CONCLUSION

In Chapters I and III the implications of not having the 9010A based test stations and the reduction of their capabilities due to the inability to purchase additional interface pods was examined. The following paragraph summarizes the main points covered in these chapters.

The lack of an interface pod for the TMS9995 had caused a reduction in the effectiveness of the existing troubleshoot and repair stations. This drop in effectiveness caused an increase in the amount of defective product and left the international repair sites with no means of testing the TMS9995 based designs. Additionally, the design group released new designs which could not be debugged until the test hardware was released. The time lost in debugging the hardware required costly rework.

The capabilities of the troubleshoot station is covered in Chapter II. The software required inside the interface pod was discussed in Chapters IV and V. The

software was required to be relocatable in memory due to the ability of the user to request an operation in either the upper or lower 4K block of the TMS9995 memory map. The use of the upper 16 address bits as software flags was covered in Chapter V. These flags are used to identify "CRU" memory operations, as well as directing the interface pod to perform "Quick" routines. These routines enabled the interface pod to bypass the time consuming process of handshaking with the 9010A.

Due to the problems examined in Chapter I and III, the interface pod was constructed to meet the following requirements.

1. to perform buss operations with the least amount of distortion to the buss;
2. to provide a means of latching the state of the address buss, data buss, control and status signals;
3. to execute assembly language programs from the pod;
4. to run from the clock on the UUT;
5. to protect against shorted conditions on the buss;
6. to run self-diagnostics;
7. to keep the power consumption below 10 watts;
8. to maintain the same board dimensions.

In Chapter IV, the characteristics of the TMS9995 and the TMS9901 were discussed. These chips were used to assure the timing structure was not distorted and to reduce the amount of support chips required on the design. This reduction in support chips helped maintain the power budget since high speed PALs had to be used in the latching circuits to reduce the distortion of the control and status signals. The hardware required to protect the interface pod, to provide the clock, and to provide a load for the self-test was outlined in this chapter.

In conclusion, the TMS9995 interface pod has provided the Industrial Systems Division of Texas Instruments with the ability to troubleshoot defective boards and provide design support on TMS9995 based designs. Additionally, the interface pod provides ISD with the ability to use this design as a model for future interface pod designs.

LIST OF REFERENCES

LIST OF REFERENCES

CUPL 2.0, "The University Compiler for Programmable Logic, Users Manual." Assisted Technology. San Jose, CA, 1984.

FUTURENET, "A Data I/O Company Schematic Designer, Users Manual." Data I/O Corporation. February 1988.

"Master Selection Guide." Motorola Literature Distribution. Phoenix, Arizona.

"MOS Memory Data Book." Texas Instruments Incorporated. Dallas, Texas, 1984.

"Precision Analog Integrated Circuits." PMI Data Book, 1988.

"Programmable Logic Data Book." Texas Instruments Incorporated. Dallas, Texas, 1988.

"The TTL Data Book." Texas Instruments Incorporated. Vol. 2. Dallas, Texas, 1985.

"Toyocom Outline of Crystal Products." Toyo Communication Equipment Company, Ltd. Tokyo, Japan.

"TMS9901 Programmable Systems Interface Manual." Texas Instruments Incorporated, Dallas, Texas July 1978.

"TMS9995 16-Bit Microcomputer." Texas Instruments Incorporated, Dallas, Texas, August 1982.

"9000A-6802 Interface Pod Instruction Manual." John Fluke Manufacturing Company, Everett, Washington, May 1982.

"9010A Micro-System Troubleshooter, Operator Manual." John Fluke Manufacturing Corporation, Everett, Washington, June 1981.

"9010A Micro-System Troubleshooter, Programming Manual." John Fluke Manufacturing Corporation, Everett, Washington, June 1981.

"9010A Micro-System Troubleshooter, Service Manual." John Fluke Manufacturing Corporation, Everett, Washington, April 1982.

APPENDIXES

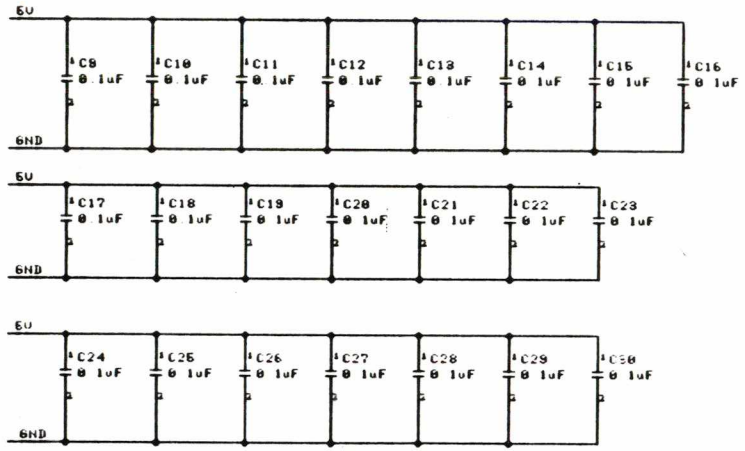
APPENDIX A

SCHEMATIC OF TMS9995 INTERFACE POD

NOTES. UNLESS OTHERWISE SPECIFIED.
1 ALL RESISTOR VALUES ARE IN OHMS. 1/4W. TOLERANCE 5%
2 ALL CAPACITOR VALUES ARE IN MICROFARADS.

REVISIONS			
REV	DESCRIPTION	DATE	APPROVED

FILTER CAPACITORS ACCROSS IC POWER/GROUND



TI INTERNAL DATA

-1 QTY	ITEM NO	PART OR IDENTIFYING NUMBER	NOMENCLATURE OR DESCRIPTION	REMARKS
			PARTS LIST	

DRAWN	MARK S SITTER
CHK	
APVD	
MFG	
COMP	
RLSE	

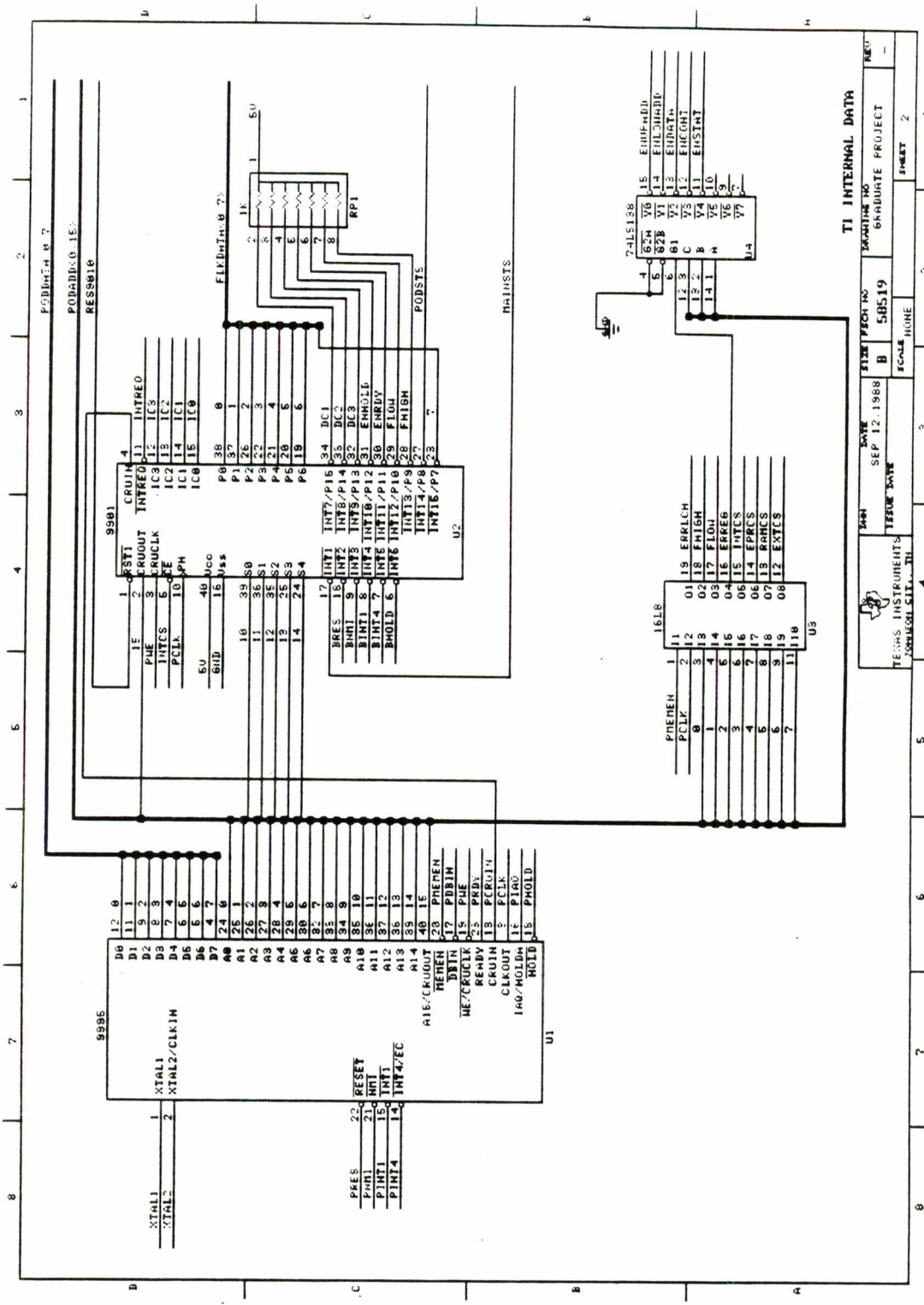
TEXAS INSTRUMENTS
JOHNSON CITY, TN

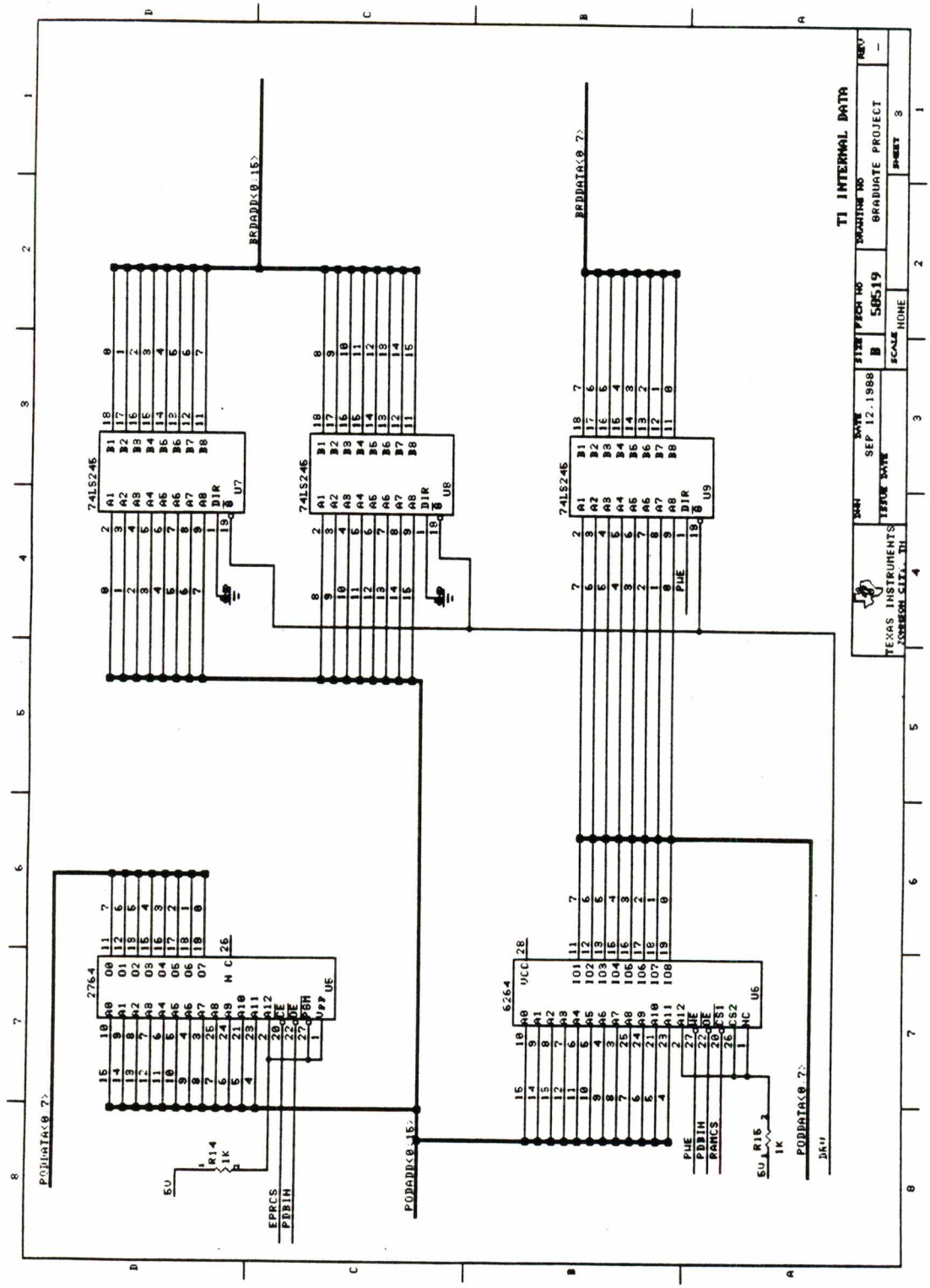
SCHEMATIC TMS9995 POD
GRADUATE PROJECT

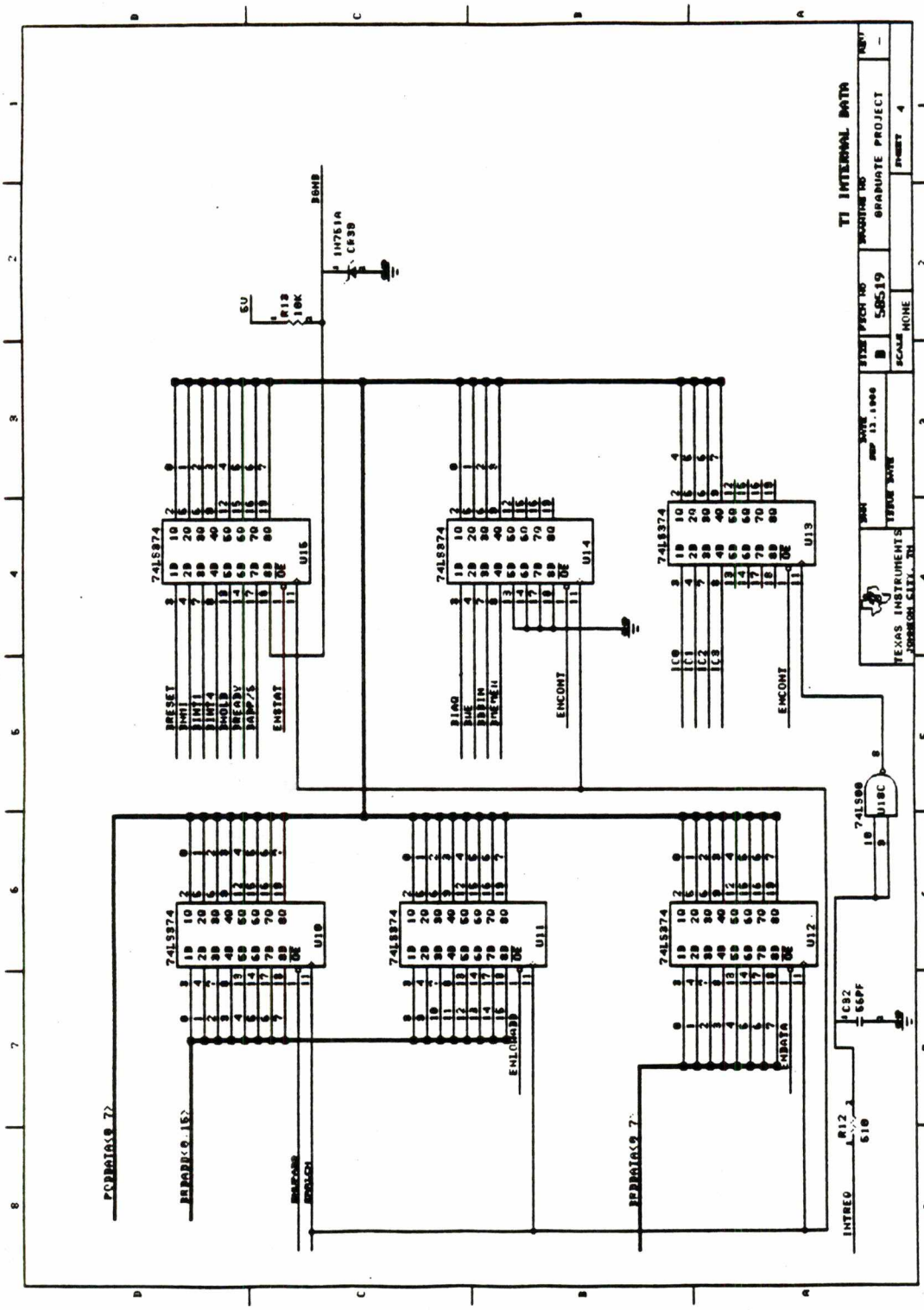
SIZE	FSCH NO	DRAWING NO	REV
B	58519		-

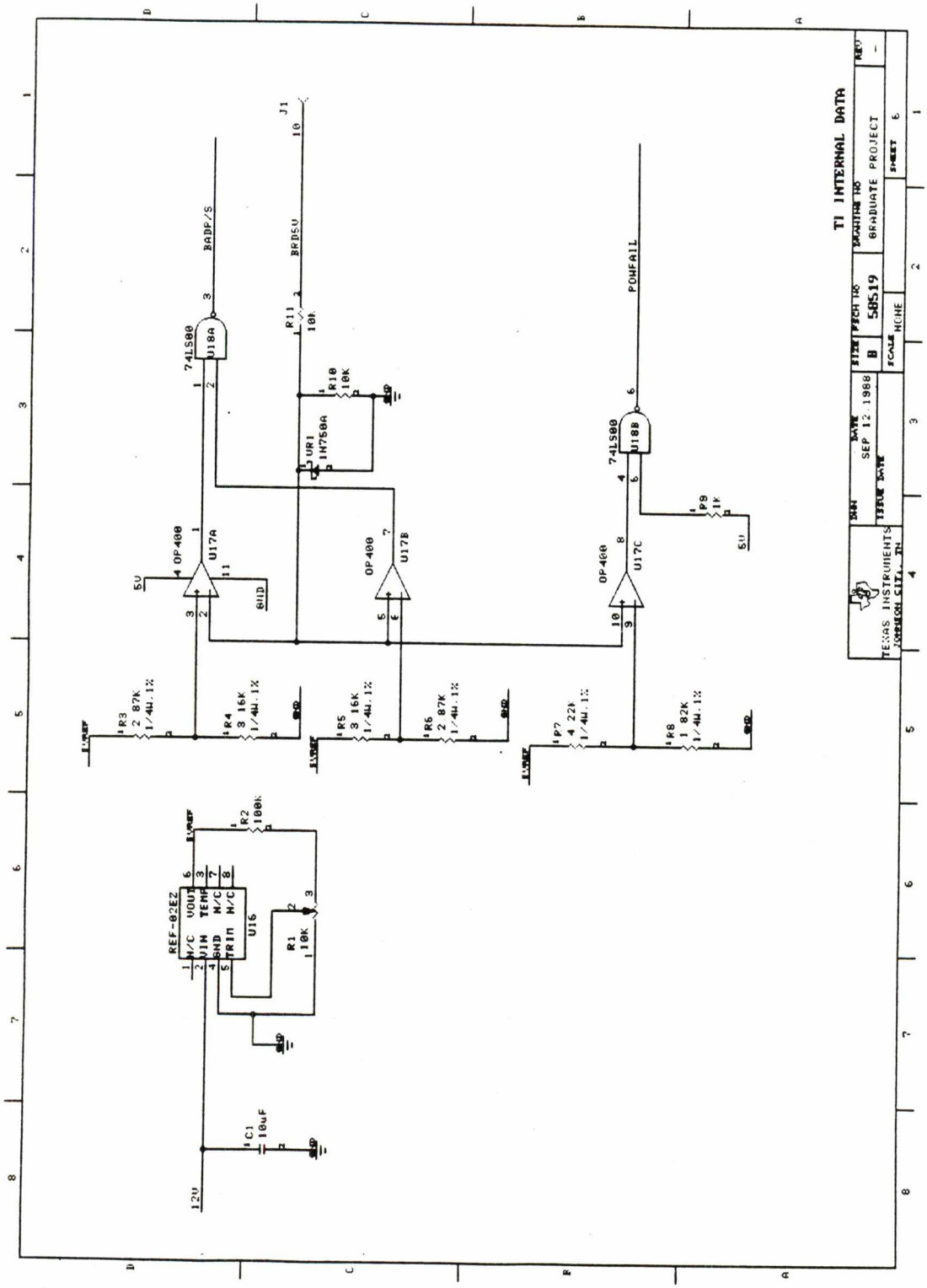
SCALE NONE SHEET 1 OF 9

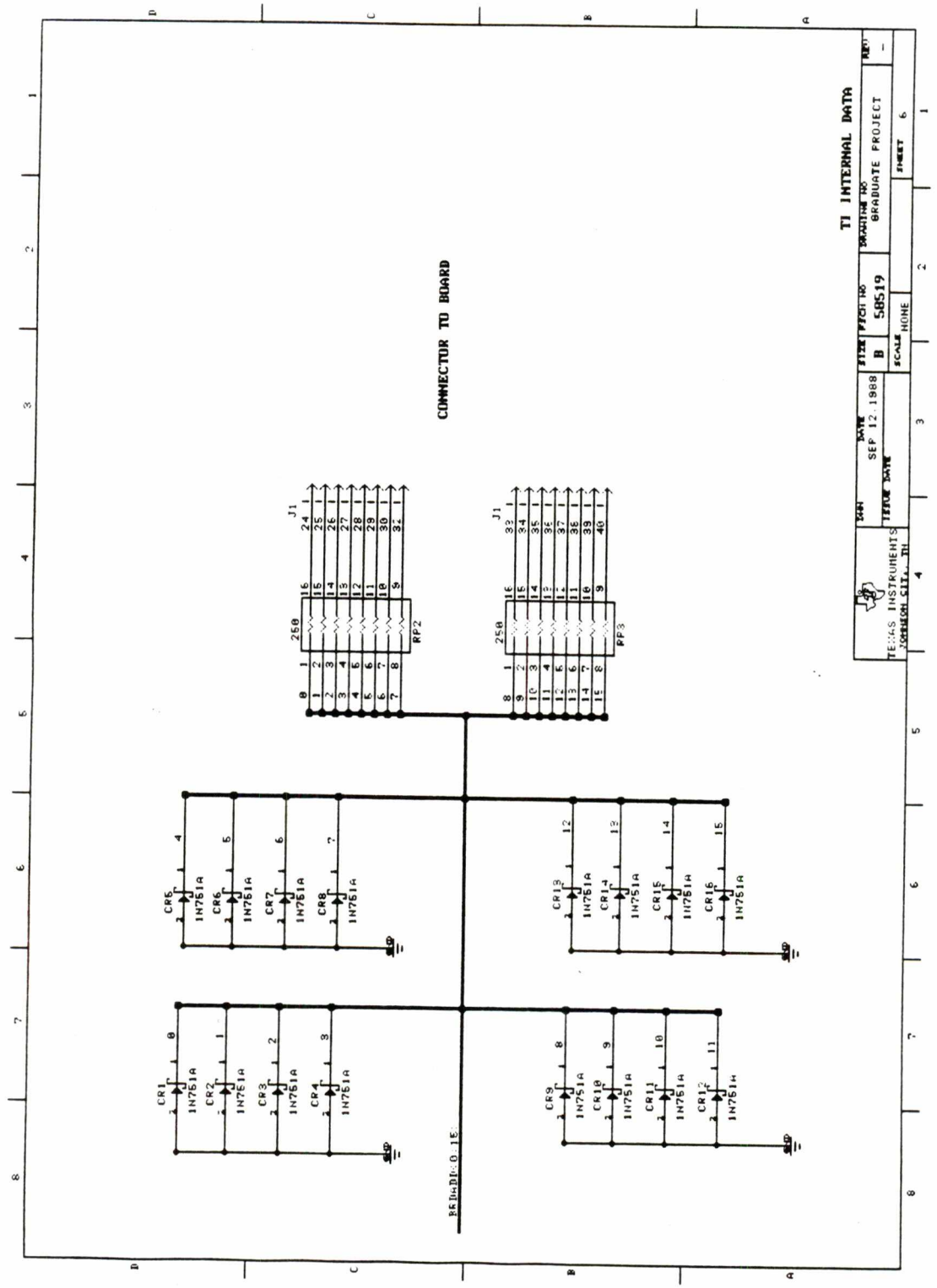
NEXT ASSY USED ON
APPLICATION

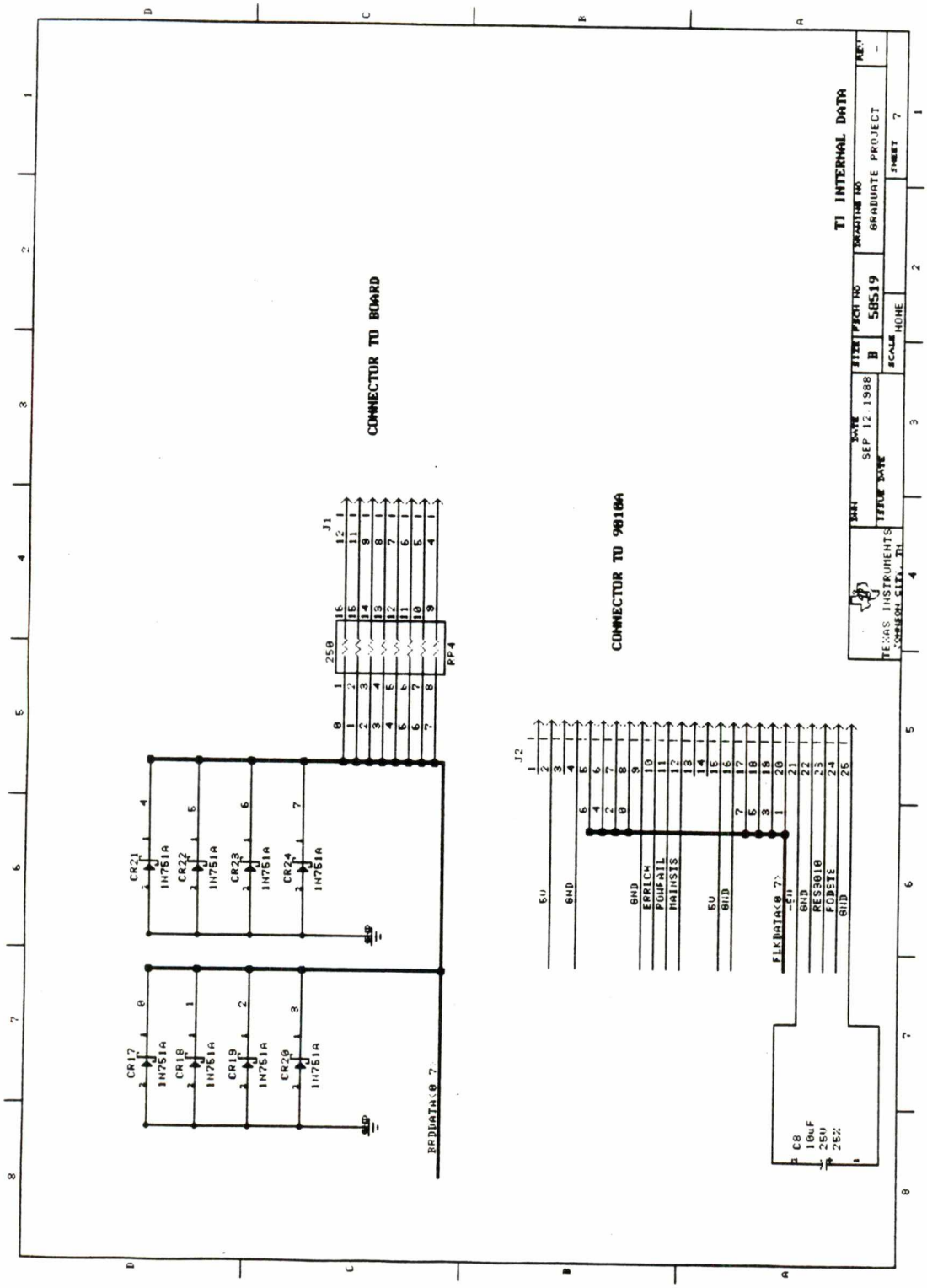

TI INTERNAL DATA
SEP 12 1988
B
58519
GRADUATE PROJECT
TEXAS INSTRUMENTS
9995
9901
74LS138
1618
RP1
U1
U2
U3
U4
RESET
INT1
INT4/EC
PRES
PWR1
PINT1
PINT4
XTAL1
XTAL2
PODBUS 0-15
RES9818
PODATA 0-7
FLXBUS 0-7
DATASTS
PODSTS
1K
5V
PMEMEN
PCLK
ERRLCH
FHIGH
FLOW
ERRES
INTCS
EPRCS
RAMCS
EXTCS
ENHOLD
ENRDY
ENHALT
ENDATA
ENCONT
ENSTAT

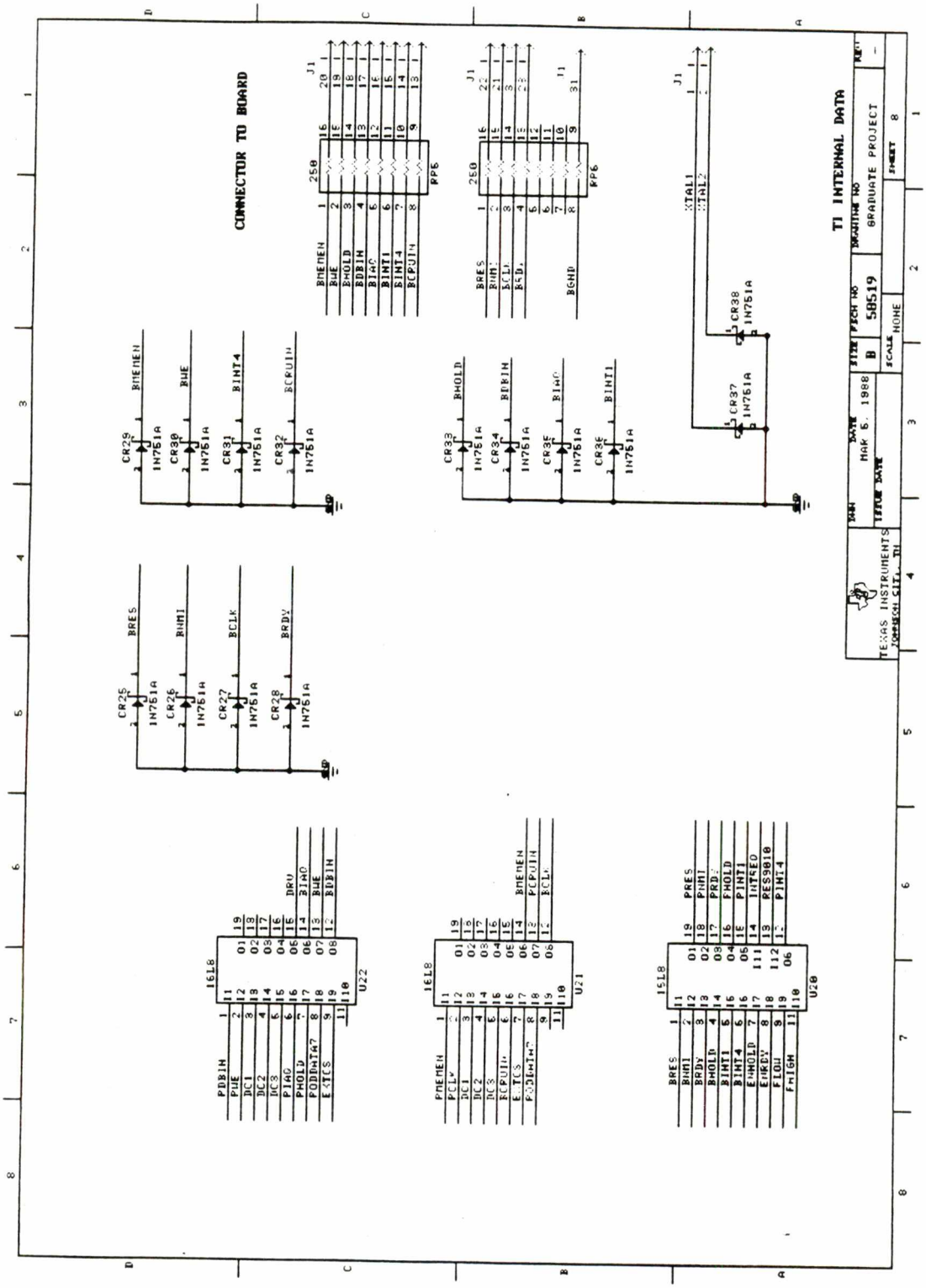

TI INTERNAL DATA
TEXAS INSTRUMENTS
DRAWN
SEP 12 1988
SIZE
B
SCALE
HOME
DRAWING NO
GRADUATE PROJECT
SHEET 3
REV
PRDATA<0 7>
PRDADR<0 15>
PODATA<0 7>
PODADR<0 15>
74LS245
U7
U8
U9
2764
U5
6264
U6
R14
1K
R15
1K
5V
EPRCS
PDBIN
PWE
PDBIN
RAMCS
DIR

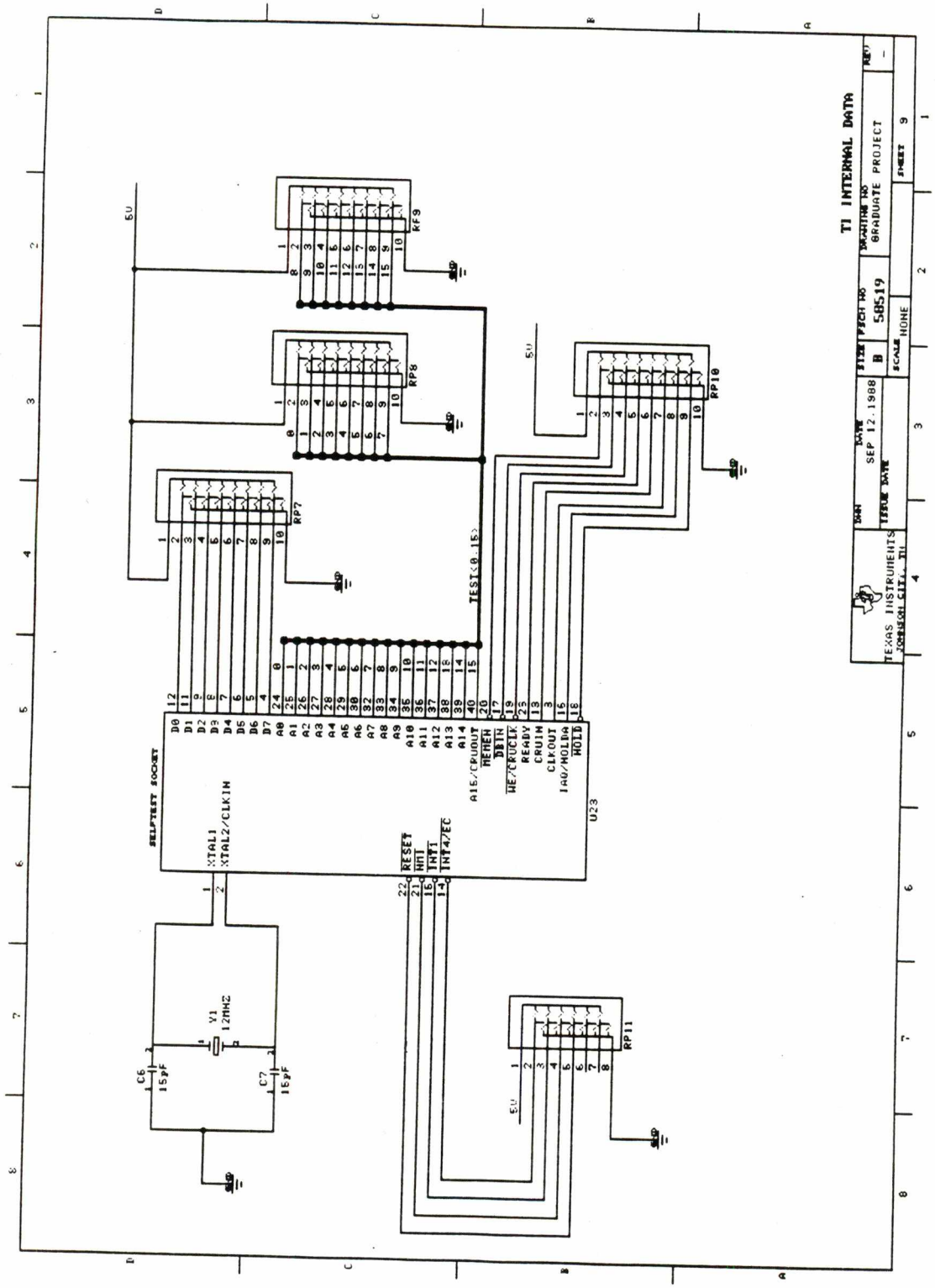

TI INTERNAL DATA
GRADUATE PROJECT
TEXAS INSTRUMENTS
5V
R13
10K
DBHB
1N751A
C839
74LS374
U15
U14
U13
U10
U11
U12
74LS00
U16C
RESET
INT1
INT4
HOLD
READY
RAMCS
ENSTAT
BIAO
BUS
BRIN
PRECHN
ENCONT
IC0
IC1
IC2
IC3
C82
66PF
R12
610
INTREQ
ENLOADD
ENDATA
PDDATA(0..7)
PADDR(0..15)
RDSTRB
RDSTCH


TI INTERNAL DATA
TEXAS INSTRUMENTS
SEP 12 1988
B
58519
GRADUATE PROJECT
SCALE NONE
SHEET 6
J1
BADF/S
BRDSV
POWFAIL
741S00
U18A
U18B
OP400
U17A
U17B
U17C
R10 10K
R11 10K
R1 10K
R2 100K
R3 2.87K 1/4W 1%
R4 3.16K 1/4W 1%
R5 3.16K 1/4W 1%
R6 2.87K 1/4W 1%
R7 4.22K 1/4W 1%
R8 1.82K 1/4W 1%
R9 1K
C1 10uF
REF-02EZ
U16
URI
1N758A
5V
12V
GND


CONNECTOR TO BOARD
TI INTERNAL DATA
TEXAS INSTRUMENTS
DATE
SEP 12 1988
SIZE
B
58519
SCALE NONE
GRADUATE PROJECT
SHEET 6
J1
RP2
RP3
CR1
CR2
CR3
CR4
CR5
CR6
CR7
CR8
CR9
CR10
CR11
CR12
CR13
CR14
CR15
CR16
1N761A


TI INTERNAL DATA
GRADUATE PROJECT
SEP 12, 1988
B
58519
SCALE NONE
SHEET 7
TEXAS INSTRUMENTS
CONNECTOR TO BOARD
CONNECTOR TO 9818A
J1
J2
PP4
CR17
CR18
CR19
CR20
CR21
CR22
CR23
CR24
1N751A
C8
10uF
25V
GND
5V
ERRICH
POWFAIL
MAINSTS
FLKDATA<0 7>
RESET0
FOUSTE
EFDATA<0 7>


CONNECTOR TO BOARD
TI INTERNAL DATA
TEXAS INSTRUMENTS
DATE
MAR 6, 1988
SIZE
B
DRAWING NO
SCALE
NONE
SHEET
8
GRADUATE PROJECT
CR25
CR26
CR27
CR28
CR29
CR30
CR31
CR32
CR33
CR34
CR35
CR36
CR37
CR38
IN761A
BRES
BHMI
BCLK
BRDY
BMEMEN
BWE
BINT4
BCPUIN
BHOLD
BDBIN
BIAQ
BINT1
XTAL1
XTAL2
U20
U21
U22
1618
J1
P96


TI INTERNAL DATA
TEXAS INSTRUMENTS
SEP 12 1988
B
585519
GRADUATE PROJECT
NONE
SELFTEST SOCKET
XTAL1
XTAL2/CLKIN
RESET
HOLD
U23
TEST(0..15)
RP7
RP8
RP9
RP10
RP11
Y1
12MHZ
C6
15pF
C7
15pF
5V

APPENDIX B

PAL EQUATIONS AND SIMULATION FILES

```
PARTNO            PAL16L8;
NAME        MEMORY DECODE;
DATE              08/20/88;
REV                   01;
DESIGNER      MARK SITTER;
COMPANY               TI;
ASSEMBLY      TMS9995 POD;
LOCATION              U3;

/*************************************************************/
/**/
/* THIS DEVICE PROVIDES THE CHIP SELECTS FOR THE 9901        */
/* INTERRUPT CONTROLLER, EPROM, RAM, ERROR REGISTER          */
/* SELECTOR, AND EXTERNAL BUSS DRIVERS.  THE PAL ALSO        */
/* PROVIDES THE CLOCK FOR THE EXTERNAL BUSS LATCHES.         */
/**/
/*************************************************************/
/**  ALLOWABLE TARGET DEVICE TYPES:  PAL16L8                 */
/*************************************************************/
/**  INPUTS                                                  **/

 PIN 1       = MEMEN      ;    /*  MEMORY ENABLE             */
 PIN [2..9]  = [A0..A7]   ;    /*  ADDRESS BUSS HIGH BYTE    */
 PIN 11      = CLKOUT     ;    /*  SYSTEM CLOCK              */
 PIN 17      = FLOW?      ;    /*  (0000-0FFF) ON EXT BUSS   */
 PIN 18      = FHIGH      ;    /*  (F000-FFFF) ON EXT BUSS   */

/**  OUTPUTS                                                 **/

 PIN 19       = EXTLCH     ;    /*  ERROR LATCH CLOCK        */
 PIN 16       = ERREG      ;    /*  ERROR LATCH SELECT       */
 PIN 15       = INTCS      ;    /*  9901 CHIP SELECT         */
 PIN 14       = EPRCS      ;    /*  EPROM CHIP SELECT        */
 PIN 13       = RAMCS      ;    /*  RAM CHIP SELECT          */
 PIN 12       = EXTCS      ;    /*  EXTERNAL DRIVER ENABLE   */


/** DECLARATIONS AND INTERMEDIATE VARIABLE DEFINITIONS  **/


 FIELD ADDRESS = [A0..A7] ;
 FIELD MEMADD  = [A0..A4] ;
 ADDF = ADDRESS:8F  ;    /* 9901 UPPER ADDRESS BYTE        */
 ADD0 = ADDRESS:80  ;    /* 9901 UPPER ADDRESS BYTE        */
 ERRF = ADDRESS:8F  ;    /* ERROR LATCH SELECT ADD         */
 ERR0 = ADDRESS:80  ;    /* ERROR LATCH SELECT ADD         */
 EPRF1= MEMADD:0F   ;    /* EPROM UPPER ADD BYTE           */
 EPR01= MEMADD:00   ;    /* EPROM UPPER ADD BYTE           */
 EPRF2= ADDRESS:0F  ;    /* INTERNAL RAM AREA              */
 RAMF = MEMADD:1F   ;    /* RAM UPPER ADDRESS BYTE         */
 RAM0 = MEMADD:10   ;    /* RAM UPPER ADDRESS BYTE         */
```

```
/** LOGIC EQUATIONS                                                **/

!INTCS  =(ADD0 & FLOW # ADDF & FHIGH) & MEMEN;

!ERREG =(ERR0 & FLOW # ERRF & FHIGH) & !MEMEN;

!EXTLCH =!CLKOUT & (EPRCS #(EPRF1 & FHIGH & !MEMEN))
 & INTCS & ERREG & RAMCS;

!EPRCS = ((EPRF1&FHIGH&!EPRF2) # EPR01&FLOW)
 &!MEMEN & ERREG;

!RAMCS = (RAMF & FHIGH # RAM0 & FLOW) & !MEMEN;

!EXTCS = RAMCS & (EPRCS #(EPRF1 & FHIGH & !MEMEN))
 & ERREG & INTCS;
```

```
PARTNO            PAL16L8;
NAME    CONTROL SIGNAL EN;
DATE             08/20/88;
REV                    01;
DESIGNER      MARK SITTER;
COMPANY                TI;
ASSEMBLY      TMS9995 POD;
LOCATION               U20;

/*************************************************************/
/*                                                           */
/* THIS DEVICE KEEPS THE INTERRUPTS, READY, AND HOLD         */
/* FROM INTERFERING WITH THE 9995 PROCESSOR IN THE           */
/* POD.  THE USER CAN ENABLE THE READY LINE AND THE          */
/* HOLD LINE.  WHEN THE POD GOES INTO THE RUNUUT MODE        */
/* ALL INTERRUPTS, THE READY LINE, AND THE HOLD LINE         */
/* WILL BE ALLOWED TO PASS THROUGH TO THE 9995.  THE         */
/* RESET FROM THE 9010A WILL BE ALLOWED TO PASS TO           */
/* THE 9995.  THE INTERRUPT REQUEST FROM THE 9901 WILL       */
/* BE SENT AS AN NMI INTERRUPT TO THE 9995.                  */
/*                                                           */
/*************************************************************/
/**  ALLOWABLE TARGET DEVICE TYPES:  PAL16L8                 */
/*************************************************************/
/**  INPUTS                                                 **/

  PIN 1       = RESET     ;    /*  RESET FROM BOARD          */
  PIN 2       = NMI       ;    /*  NMI FROM BOARD            */
  PIN 3       = READY     ;    /*  READY FROM BOARD          */
  PIN 4       = HOLD      ;    /*  HOLD FROM BOARD           */
  PIN 5       = INT1      ;    /*  INT1 FROM BOARD           */
  PIN 6       = INT4      ;    /*  INT4 FROM BOARD           */
  PIN 7       = ENHOLD    ;    /*  ENABLE HOLD SIGNAL        */
  PIN 8       = ENREADY   ;    /*  ENABLE READY SIGNAL       */
  PIN 9       = FLOW      ;    /*  LOW BYTE ON BOARD         */
  PIN 11      = FHIGH     ;    /*  HIGH BYTE ON BOARD        */
  PIN 13      = RES9010   ;    /*  RESET FROM 9010A          */
  PIN 14      = INTREQ    ;    /*  INTREQ FROM 9901          */


/**  OUTPUTS                                                **/

  PIN 19      = RESPOD    ;    /*  RESET TO 9995             */
  PIN 18      = NMIPOD    ;    /*  NMI TO 9995               */
  PIN 17      = RDYPOD    ;    /*  READY TO 9995             */
  PIN 16      = HOLDPOD   ;    /*  HOLD TO 9995              */
  PIN 15      = INT1POD   ;    /*  INT1 TO 9995              */
  PIN 12      = INT4POD   ;    /*  INT4 TO 9995              */


/** DECLARATIONS AND INTERMEDIATE VARIABLE DEFINITIONS     **/
```

```
/** LOGIC EQUATIONS                                                **/

 !RESPOD =(!RES9010#!RESET)&(!RES9010#!FHIGH)&(!RES9010
  #!FLOW);

 !NMIPOD =(!INTREQ#!NMI)&(!INTREQ#!FLOW)&(!INTREQ#!FHIGH);

 !RDYPOD =!READY&(!ENREADY#!FLOW)&(!ENREADY#!FHIGH);

 !HOLDPOD=!HOLD&(!ENHOLD#!FLOW)&(!ENHOLD#!FHIGH);

 !INT1POD=!INT1&!FHIGH&!FLOW;

 !INT4POD=!INT4&!FHIGH&!FLOW;
```

```
PARTNO            PAL16L8;
NAME    EXT DRIVE CONTROL;
DATE              08/20/88;
REV                    01;
DESIGNER      MARK SITTER;
COMPANY                TI;
ASSEMBLY      TMS9995 POD;
LOCATION               U21;

/*************************************************************/
/*                                                           */
/* THIS DEVICE GIVES THE USER THE ABILITY TO TOGGLE          */
/* SELECTED CONTROL SIGNALS.  THREE LINES FROM THE           */
/* 9901 WILL BE USED TO DETERMINE THE CONTROL SIGNAL         */
/* THE USER WISHES TO TOGGLE.  DATA LINE 7 WILL              */
/* DETERMINE THE STATE.                                      */
/*                                                           */
/*      DC1   DC2   DC3         CONDITION                    */
/*       1     1     1     LINES ENABLED TO BOARD           */
/*       1     1     0     MEMEN LINE DRIVEN                */
/*       0     1     1     CLKOUT DRIVEN                    */
/*       0     0     1     CRUIN LINE DRIVEN                */
/*       0     0     0     ALL CONTROL LINES DRIVEN         */
/*                                                           */
/*************************************************************/
/**  ALLOWABLE TARGET DEVICE TYPES:  PAL16L8                **/
/*************************************************************/
/**  INPUTS                                                 **/

 PIN 1      = MEMEN    ;    /*   MEMEN FROM 9995           */
 PIN 2      = CLKOUT   ;    /*  CLKOUT FROM 9995           */
 PIN 3      = DC1      ;    /*  DRIVE CONTROL 1            */
 PIN 4      = DC2      ;    /*  DRIVE CONTROL 2            */
 PIN 5      = DC3      ;    /*  DRIVE CONTROL 3            */
 PIN 6      = BCRUIN   ;    /*  CRUIN FROM BOARD           */
 PIN 7      = EXTMEM   ;    /*  EXTERNAL MEMORY ACCESS     */
 PIN 8      = D7       ;    /*  DATA BIT 7                 */

/**  OUTPUTS                                                **/

 PIN 12      = BMEMEN    ;    /*  MEMEN TO BOARD           */
 PIN 13      = CRUIN     ;    /*  CRUIN TO 9995            */
 PIN 14      = BCLKOUT   ;    /*  CLKOUT TO BOARD          */

/** DECLARATIONS AND INTERMEDIATE VARIABLE DEFINITIONS     **/

/** LOGIC EQUATIONS                                        **/

 !BMEMEN =!MEMEN & !D7 # !MEMEN & DC3 # !DC1 & DC2 & !MEMEN
  # !MEMEN & DC1 & !DC2 # !D7 & !DC1 & !DC2 & !DC3
  # !D7 & DC1 & DC2 & !DC3;
```

```
!BCLKOUT=!CLKOUT & DC1 # !CLKOUT & !D7  # !CLKOUT & !DC2 &
 DC3 #  !CLKOUT & DC2  & !DC3 #  !DC1 & DC2  & DC3 & !D7 #
 !DC1 & !DC2 & !DC3 & !D7;

!CRUIN =!BCRUIN &  DC1 # !BCRUIN &  !D7 # DC2 &  !BCRUIN #
!DC1 & !DC2 & !D7;

CRUIN.OE=!EXTMEM;
```

```
PARTNO            PAL16L8;
NAME    EXT DRIVE CONTROL;
DATE              08/20/88;
REV                    01;
DESIGNER      MARK SITTER;
COMPANY                TI;
ASSEMBLY      TMS9995 POD;
LOCATION               U22;


/*************************************************************/
/*                                                           */
/* THIS DEVICE GIVES THE USER THE ABILITY TO TOGGLE          */
/* SELECTED CONTROL SIGNALS.  THREE LINES FROM THE           */
/* 9901 WILL BE USED TO DETERMINE THE CONTROL SIGNAL         */
/* THE USER WISHES TO TOGGLE.  DATA LINE 7 WILL              */
/* DETERMINE THE STATE.                                      */
/*                                                           */
/*      DC1   DC2   DC3         CONDITION                    */
/*       1     1     1     LINES ENABLED TO BOARD           */
/*       1     0     1     DBIN LINE DRIVEN                 */
/*       1     0     0     WE/CRUCLK LINE DRIVEN            */
/*       0     1     0     IAQ/HOLDA DRIVEN                 */
/*       0     0     0     ALL CONTROL LINES DRIVEN         */
/*                                                           */
/*************************************************************/
/**  ALLOWABLE TARGET DEVICE TYPES:  PAL16L8                **/
/*************************************************************/


/**  INPUTS                                                 **/

 PIN 1      = DBIN      ;   /*  DBIN FROM 9995              */
 PIN 2      = WE        ;   /*  WE/CRUCLK FROM 9995         */
 PIN 3      = DC1       ;   /*  DRIVE CONTROL 1             */
 PIN 4      = DC2       ;   /*  DRIVE CONTROL 2             */
 PIN 5      = DC3       ;   /*  DRIVE CONTROL 3             */
 PIN 6      = IAQ       ;   /*  IAQ/HOLDA FROM 9995         */
 PIN 7      = HOLDPOD   ;   /*  HOLDLINE TO 9995            */
 PIN 8      = D7        ;   /*  DATA BIT 7                  */
 PIN 9      = EXTCS     ;   /*  REQUEST TO ENABLE DRIVERS   */

/**  OUTPUTS                                                **/

 PIN 12      = BDBIN    ;   /*  DBIN TO BOARD               */
 PIN 13      = BWE      ;   /*  WE/CRUCLK TO BOARD          */
 PIN 14      = BIAQ     ;   /*  IAQ/HOLDA TO BOARD          */
 PIN 15      = DRV      ;   /*  ENABLE TO DRIVER CHIPS      */
 PIN 16      = ENBUSS   ;   /*  ENABLE DBIN,WE,AND DRIVERS  */

 /** DECLARATIONS AND INTERMEDIATE VARIABLE DEFINITIONS **/
```

```
/**  LOGIC EQUATIONS                                                **/

!BDBIN    =!DBIN &D7 # DC2  &  !DBIN # !DC1  & DC3 & !DBIN
           # DC1 & !DBIN & !DC3 # !DC1 & !DC2 & !DC3 & !D7
           # DC1 & !DC2 & DC3 & !D7;

!ENBUSS =HOLDPOD # !BIAQ

BDBIN.OE=!ENBUSS;

!BWE =!WE & !D7 # !WE & DC2 # !WE & DC3 # !D7 & !DC2
      & !DC3;

 BWE.OE =!ENBUSS;

!BIAQ =!IAQ & !HOLDPOD # HOLDPOD & DC1 # HOLDPOD & DC3
       # HOLDPOD & !DC3 & !D7;

 DRV.OE =!ENBUSS;

!DRV =!EXTCS;
```

```
PARTNO            PAL16L8;
NAME    EXT DRIVE CONTROL;
DATE              08/20/88;
REV                    01;
DESIGNER      MARK SITTER;
COMPANY                TI;
ASSEMBLY      TMS9995 POD;
LOCATION               U22;

/*************************************************************/
/*                                                           */
/* THIS DEVICE GIVES THE USER THE ABILITY TO TOGGLE          */
/* SELECTED CONTROL SIGNALS.  THREE LINES FROM THE           */
/* 9901 WILL BE USED TO DETERMINE THE CONTROL SIGNAL         */
/* THE USER WISHES TO TOGGLE.  DATA LINE 7 WILL              */
/* DETERMINE THE STATE.                                      */
/*                                                           */
/*      DC1   DC2   DC3         CONDITION                    */
/*       1     1     1     LINES ENABLED TO BOARD           */
/*       1     0     1     DBIN LINE DRIVEN                 */
/*       1     0     0     WE/CRUCLK LINE DRIVEN            */
/*       0     1     0     IAQ/HOLDA DRIVEN                 */
/*       0     0     0     ALL CONTROL LINES DRIVEN         */
/*                                                           */
/*************************************************************/
/**  ALLOWABLE TARGET DEVICE TYPES:  PAL16L8                */
/*************************************************************/

 ORDER:
  DBIN, WE, DC1, DC2, DC3, IAQ, D7, HOLDPOD, EXTCS,
  %4, DRV, BDBIN, BWE, BIAQ;


 VECTORS:
  $MSG "                                          ";
  $MSG "       DWDDDIDHE    DBBB                  ";
  $MSG "       BECCCA7OX    RDWI                  ";
  $MSG "       I 123Q LT    VBEA                  ";
  $MSG "       N      DC     I Q                  ";
  $MSG "              PS     N                    ";
  $MSG "              O                           ";
  $MSG "              D                           ";
  $MSG "                                          ";
  $MSG "                                          ";


       11111XX11  HHHL
       01111XX10  LLHL
       10111XX11  HHLL
       00110X110  LLLL
       11110X011  HHHL
```

```
X0101X110  LHLL
X1101X011  HLHL
0X100X110  LLHL
1X100X011  HHLL
00011X110  LLLL
11011X011  HHHL
00010X110  LLLH
11010X011  HHHL
00001X110  LLLL
11001X011  HHHL
XX000X110  LHHH
XX000X011  HLLL
001111X00  ZZZH
111111X01  ZZZH
```

```
PARTNO            PAL16L8;
NAME    EXT DRIVE CONTROL;
DATE              08/20/88;
REV                    01;
DESIGNER      MARK SITTER;
COMPANY                TI;
ASSEMBLY      TMS9995 POD;
LOCATION               U21;

/*************************************************************/
/*                                                           */
/* THIS DEVICE GIVES THE USER THE ABILITY TO TOGGLE          */
/* SELECTED CONTROL SIGNALS.  THREE LINES FROM THE           */
/* 9901 WILL BE USED TO DETERMINE THE CONTROL SIGNAL         */
/* THE USER WISHES TO TOGGLE.  DATA LINE 7 WILL              */
/* DETERMINE THE STATE.                                      */
/*                                                           */
/*      DC1   DC2   DC3         CONDITION                    */
/*       1     1     1     LINES ENABLED TO BOARD           */
/*       1     1     0     MEMEN LINE DRIVEN                */
/*       0     1     1     CLKOUT DRIVEN                    */
/*       0     0     1     CRUIN LINE DRIVEN                */
/*       0     0     0     ALL CONTROL LINES DRIVEN         */
/*                                                           */
/*************************************************************/
/**  ALLOWABLE TARGET DEVICE TYPES:  PAL16L8                */
/*************************************************************/

     ORDER:
      MEMEN, CLKOUT, DC1, DC2, DC3,BCRUIN, EXTMEM, D7,
      %5, BMEMEN, BCLKOUT, CRUIN;


     VECTORS:
      $MSG "                                    ";
      $MSG "       MCDDDBED      BBC            ";
      $MSG "       ELCCCCX7      MCR            ";
      $MSG "       MK123RT       ELU            ";
      $MSG "       EO   UM       MKI            ";
      $MSG "       NU   IE       EON            ";
      $MSG "        T   NM       NU             ";
      $MSG "                      T             ";
      $MSG "                                    ";
      $MSG "                                    ";


                   11111X1X  HHZ
                   0011100X  LLL
                   1111110X  HHH
                   X0110001  HLL
                   X1110100  LHH
```

```
00101001  LLL
11101100  HHH
00100001  LLL
11100100  HHH
0X011001  LHL
1X011100  HLH
00010001  LLL
11010100  HHH
00001X01  LLH
11001X00  HHL
XX000X01  HHH
XX000X00  LLL
01111X1X  LHZ
10111X1X  HLZ
0111110X  LHH
1011100X  HLL
```

```
PARTNO            PAL16L8;
NAME    CONTROL SIGNAL EN;
DATE             08/20/88;
REV                   01;
DESIGNER      MARK SITTER;
COMPANY                TI;
ASSEMBLY      TMS9995 POD;
LOCATION              U20;

/*************************************************************/
/*                                                           */
/* THIS DEVICE KEEPS THE INTERRUPTS, READY, AND HOLD         */
/* FROM INTERFERING WITH THE 9995 PROCESSOR IN THE           */
/* POD.  THE USER CAN ENABLE THE READY LINE AND THE          */
/* HOLD LINE.  WHEN THE POD GOES INTO THE RUNUUT MODE        */
/* ALL INTERRUPTS, THE READY LINE, AND THE HOLD LINE         */
/* WILL BE ALLOWED TO PASS THROUGH TO THE 9995.  THE         */
/* RESET FROM THE 9010A WILL BE ALLOWED TO PASS TO           */
/* THE 9995.  THE INTERRUPT REQUEST FROM THE 9901 WILL       */
/* BE SENT AS AN NMI INTERRUPT TO THE 9995.                  */
/*                                                           */
/*                                                           */
/*************************************************************/
/**  ALLOWABLE TARGET DEVICE TYPES:  PAL16L8                */
/*************************************************************/
     ORDER:
      RESET, NMI, READY, HOLD, INT1, INT4, ENHOLD,
      ENREADY, FLOW, FHIGH, RES9010, INTREQ, %5, RESPOD,
      NMIPOD, RDYPOD, HOLDPOD, INT1POD, INT4POD;


     VECTORS:
      $MSG "      RNRHIIEEFFRI     RNRHII     ";
      $MSG "      EMEONNNNLHEN     EMDONN     ";
      $MSG "      SIALTTHROIST     SIYLTT     ";
      $MSG "      E DD14OEWG9R     PPPD14     ";
      $MSG "      T Y   LA H0E     OOOPPP     ";
      $MSG "            DD  1Q     DDDOOO     ";
      $MSG "             Y  0        DDD     ";
      $MSG "                                  ";
      $MSG "                                  ";

                  01111111X111  HHHHHH
                  10111111X111  HHHHHH
                  11011111X111  HHHHHH
                  11101111X111  HHHHHH
                  11110111X111  HHHHHH
                  11111011X111  HHHHHH
                  011111111X11  HHHHHH
                  101111111X11  HHHHHH
                  110111111X11  HHHHHH
```

```
111011111X11  HHHHHH
111101111X11  HHHHHH
111110111X11  HHHHHH

011111110011  LHHHHH
101111110011  HLHHHH
110111110011  HHLHHH
111011110011  HHHLHH
111101110011  HHHHLH
111110110011  HHHHHL
11101101X111  HHHLHH
111011011X11  HHHLHH
11011110X111  HHLHHH
110111101X11  HHLHHH
11111111XX01  LHHHHH
11111111X110  HLHHHH
111111111X10  HLHHHH
```

```
PARTNO            PAL16L8;
NAME        MEMORY DECODE;
DATE              08/20/88;
REV                   01;
DESIGNER      MARK SITTER;
COMPANY                TI;
ASSEMBLY      TMS9995 POD;
LOCATION               U3;

/*************************************************************/
/*                                                           */
/* THIS DEVICE PROVIDES THE CHIP SELECTS FOR THE 9901,       */
/* EPROM, RAM, ERROR LATCH ENABLE, AND EXTERNAL BUSS         */
/* DRIVERS.  THIS DEVICE ALSO PROVIDES A CLOCK FOR THE       */
/* EXTERNAL LATCHES.                                         */
/*                                                           */
/*************************************************************/

   ORDER:
    A0, A1, A2, A3, A4, A5, A6, A7, CLKOUT, MEMEN,
    FLOW, FHIGH, %3, INTCS, ERREG, EXTLCH, EPRCS,
    RAMCS, EXTCS;

   VECTORS:
    $MSG "                                                  ";
    $MSG "       AAAAAAAACMFF     E                         ";
    $MSG "       01234567LELH    IEXERE                     ";
    $MSG "              KMOI     NRTPAX                     ";
    $MSG "              OEWG     TRLRMT                     ";
    $MSG "              UN H     CECCCC                     ";
    $MSG "              T        SGHSSS                     ";
    $MSG "                                                  ";

              000000011011    HLHHHH
              000000010011    HLHHHH
              000000011111    LHHHHH
              000000010111    LHHHHH
              111100011011    HLHHHH
              111100010011    HLHHHH
              111100011111    LHHHHH
              111100010111    LHHHHH
              0001XXXX1XXX    HHHHHL
              0001XXXX0XXX    HHLHHL
              0010XXXX1XXX    HHHHHL
              0011XXXX0XXX    HHLHHL
              0100XXXX1XXX    HHHHHL
              0101XXXX0XXX    HHLHHL
              0110XXXX1XXX    HHHHHL
              0111XXXX0XXX    HHLHHL
              1000XXXX1XXX    HHHHHL
              1001XXXX0XXX    HHLHHL
```

```
1010XXXX1XXX   HHHHHL
1100XXXX0XXX   HHLHHL
1101XXXX1XXX   HHHHHL
1110XXXX0XXX   HHLHHL
111110001011   HHHHLH
111100000011   HHLHHL
000010001011   HHHHLH
000000000011   HHHLHH
1111XXXX1XX0   HHHHHL
0000XXXX0X0X   HHLHHL
```

```
PARTNO            PAL16L8;
NAME        MEMORY DECODE;
DATE              08/20/88;
REV                    01;
DESIGNER      MARK SITTER;
COMPANY                TI;
ASSEMBLY      TMS9995 POD;
LOCATION               U3;

/*************************************************************/
/*                                                           */
/* THIS DEVICE PROVIDES THE CHIP SELECTS FOR THE 9901,       */
/* EPROM, RAM, ERROR LATCH ENABLE, AND EXTERNAL BUSS         */
/* DRIVERS.  THIS DEVICE ALSO PROVIDES A CLOCK FOR THE       */
/* EXTERNAL LATCHES.                                         */
/*                                                           */
/*************************************************************/

   ORDER:
    A0, A1, A2, A3, A4, A5, A6, A7, CLKOUT, MEMEN,
    FLOW, FHIGH, %3, INTCS, ERREG, EXTLCH, EPRCS,
    RAMCS, EXTCS;


============================================================
                      Simulation Results
============================================================

   AAAAAAAACMFF      E
   01234567LELH    IEXERE
           KMOI    NRTPAX
           OEWG    TRLRMT
           UN H    CECCCC
            T       SGHSSS

   0001: 000000011011    HLHHHH
   0002: 000000010011    HLHHHH
   0003: 000000011111    LHHHHH
   0004: 000000010111    LHHHHH
   0005: 111100011011    HLHHHH
   0006: 111100010011    HLHHHH
   0007: 111100011111    LHHHHH
   0008: 111100010111    LHHHHH
   0009: 0001XXXX1XXX    HHHHHL
   0010: 0001XXXX0XXX    HHLHHL
   0011: 0010XXXX1XXX    HHHHHL
   0012: 0011XXXX0XXX    HHLHHL
   0013: 0100XXXX1XXX    HHHHHL
   0014: 0101XXXX0XXX    HHLHHL
   0015: 0110XXXX1XXX    HHHHHL
   0016: 0111XXXX0XXX    HHLHHL
```

```
0017: 1000XXXX1XXX   HHHHHL
0018: 1001XXXX0XXX   HHLHHL
0019: 1010XXXX1XXX   HHHHHL
0020: 1100XXXX0XXX   HHLHHL
0021: 1101XXXX1XXX   HHHHHL
0022: 1110XXXX0XXX   HHLHHL
0023: 111110001011   HHHHLH
0024: 111100000011   HHLHHL
0025: 000010001011   HHHHLH
0026: 000000000011   HHHLHH
0027: 1111XXXX1XX0   HHHHHL
0028: 0000XXXX0X0X   HHLHHL
```

```
PARTNO            PAL16L8;
NAME   CONTROL SIGNAL EN;
DATE             08/20/88;
REV                   01;
DESIGNER     MARK SITTER;
COMPANY               TI;
ASSEMBLY     TMS9995 POD;
LOCATION             U20;


/*************************************************************/
/*                                                           */
/* THIS DEVICE KEEPS THE INTERRUPTS, READY, AND HOLD         */
/* FROM INTERFERING WITH THE 9995 PROCESSOR IN THE           */
/* POD.  THE USER CAN ENABLE THE READY LINE AND THE          */
/* HOLD LINE.  WHEN THE POD GOES INTO THE RUNUUT MODE        */
/* ALL INTERRUPTS, THE READY LINE, AND THE HOLD LINE         */
/* WILL BE ALLOWED TO PASS THROUGH TO THE 9995.  THE         */
/* RESET FROM THE 9010A WILL BE ALLOWED TO PASS TO           */
/* THE 9995.  THE INTERRUPT REQUEST FROM THE 9901 WILL       */
/* BE SENT AS AN NMI INTERRUPT TO THE 9995.                  */
/*                                                           */
/*                                                           */
/*************************************************************/
/**  ALLOWABLE TARGET DEVICE TYPES:  PAL16L8                **/
/*************************************************************/
     ORDER:
      RESET, NMI, READY, HOLD, INT1, INT4, ENHOLD,
      ENREADY, FLOW, FHIGH, RES9010, INTREQ, %5, RESPOD,
      NMIPOD, RDYPOD, HOLDPOD, INT1POD, INT4POD;


============================================================
                      Simulation Results
============================================================

     RNRHIIEEFFRI      RNRHII
     EMEONNNNLHEN      EMDONN
     SIALTTHROIST      SIYLTT
     E DD14OEWG9R      PPPD14
     T Y   LA H0E      OOOPPP
     DD  1Q      DDDOOO
     Y  0          DDD


     0001: 01111111X111      HHHHHH
     0002: 10111111X111      HHHHHH
     0003: 11011111X111      HHHHHH
     0004: 11101111X111      HHHHHH
     0005: 11110111X111      HHHHHH
```

```
0006: 11111011X111      HHHHHH
0007: 011111111X11      HHHHHH
0008: 101111111X11      HHHHHH
0009: 110111111X11      HHHHHH
0010: 111011111X11      HHHHHH
0011: 111101111X11      HHHHHH
0012: 111110111X11      HHHHHH
0013: 011111110011      LHHHHH
0014: 101111110011      HLHHHH
0015: 110111110011      HHLHHH
0016: 111011110011      HHHLHH
0017: 111101110011      HHHHLH
0018: 111110110011      HHHHHL
0019: 11101101X111      HHHLHH
0020: 111011011X11      HHHLHH
0021: 11011110X111      HHLHHH
0022: 110111101X11      HHLHHH
0023: 11111111XX01      LHHHHH
0024: 11111111X110      HLHHHH
0025: 111111111X10      HLHHHH
```

```
PARTNO            PAL16L8;
NAME    EXT DRIVE CONTROL;
DATE              08/20/88;
REV                    01;
DESIGNER      MARK SITTER;
COMPANY                TI;
ASSEMBLY      TMS9995 POD;
LOCATION               U21;

/*************************************************************/
/*                                                           */
/* THIS DEVICE GIVES THE USER THE ABILITY TO TOGGLE          */
/* SELECTED CONTROL SIGNALS.  THREE LINES FROM THE           */
/* 9901 WILL BE USED TO DETERMINE THE CONTROL SIGNAL         */
/* THE USER WISHES TO TOGGLE.  DATA LINE 7 WILL              */
/* DETERMINE THE STATE.                                      */
/*                                                           */
/*      DC1   DC2   DC3        CONDITION                     */
/*       1     1     1     LINES ENABLED TO BOARD           */
/*       1     1     0     MEMEN LINE DRIVEN                */
/*       0     1     1     CLKOUT DRIVEN                    */
/*       0     0     1     CRUIN LINE DRIVEN                */
/*       0     0     0     ALL CONTROL LINES DRIVEN         */
/*                                                           */
/*************************************************************/
/**  ALLOWABLE TARGET DEVICE TYPES:  PAL16L8                */
/*************************************************************/

     ORDER:
      MEMEN, CLKOUT, DC1, DC2, DC3,BCRUIN, EXTMEM, D7,
      %5, BMEMEN, BCLKOUT, CRUIN;


=============================================================
                      Simulation Results
=============================================================

     MCDDDBED      BBC
     ELCCCCX7      MCR
     MK123RT       ELU
     EO   UM       MKI
     NU   IE       EON
     T    NM       NU
          T


     0001: 11111X1X     HHZ
     0002: 0011100X     LLL
     0003: 1111110X     HHH
     0004: X0110001     HLL
```

```
0005: X1110100      LHH
0006: 00101001      LLL
0007: 11101100      HHH
0008: 00100001      LLL
0009: 11100100      HHH
0010: 0X011001      LHL
0011: 1X011100      HLH
0012: 00010001      LLL
0013: 11010100      HHH
0014: 00001X01      LLH
0015: 11001X00      HHL
0016: XX000X01      HHH
0017: XX000X00      LLL
0018: 01111X1X      LHZ
0019: 10111X1X      HLZ
0020: 0111110X      LHH
0021: 1011100X      HLL
```

```
PARTNO            PAL16L8;
NAME    EXT DRIVE CONTROL;
DATE              08/20/88;
REV                    01;
DESIGNER      MARK SITTER;
COMPANY                TI;
ASSEMBLY      TMS9995 POD;
LOCATION               U22;

/*************************************************************/
/*                                                           */
/* THIS DEVICE GIVES THE USER THE ABILITY TO TOGGLE          */
/* SELECTED CONTROL SIGNALS.  THREE LINES FROM THE           */
/* 9901 WILL BE USED TO DETERMINE THE CONTROL SIGNAL         */
/* THE USER WISHES TO TOGGLE.  DATA LINE 7 WILL              */
/* DETERMINE THE STATE.                                      */
/*                                                           */
/*     DC1   DC2   DC3        CONDITION                      */
/*      1     1     1     LINES ENABLED TO BOARD            */
/*      1     0     1     DBIN LINE DRIVEN                  */
/*      1     0     0     WE/CRUCLK LINE DRIVEN             */
/*      0     1     0     IAQ/HOLDA DRIVEN                  */
/*      0     0     0     ALL CONTROL LINES DRIVE           */
/*                                                           */
/*************************************************************/
/**  ALLOWABLE TARGET DEVICE TYPES:  PAL16L8                */
/*************************************************************/

     ORDER:
      DBIN, WE, DC1, DC2, DC3, IAQ, D7, HOLDPOD, EXTCS,
      %4, DRV, BDBIN, BWE, BIAQ;


=============================================================
                      Simulation Results
=============================================================

     DWDDDIDHE     DBBB
     BECCCA7OX     RDWI
     I 123Q LT     VBEA
     N      DC      I Q
     PS     N
     O
     D


     0001: 11111XX11     HHHL
     0002: 01111XX10     LLHL
     0003: 10111XX11     HHLL
     0004: 00110X110     LLLL
     0005: 11110X011     HHHL
```

```
0006: X0101X110    LHLL
0007: X1101X011    HLHL
0008: 0X100X110    LLHL
0009: 1X100X011    HHLL
0010: 00011X110    LLLL
0011: 11011X011    HHHL
0012: 00010X110    LLLH
0013: 11010X011    HHHL
0014: 00001X110    LLLL
0015: 11001X011    HHHL
0016: XX000X110    LHHH
0017: XX000X011    HLLL
0018: 001111X00    ZZZH
0019: 111111X01    ZZZH
```

APPENDIX C

SAMPLE PROGRAM FOR 9010A MICROSYSTEM TROUBLESHOOTER

```
1:   LABEL   1
     REG0  = 0                  Up/down counter set up to
     DPY-ADDRESS /1             Prompt for starting address

2:   LABEL   2
     REG2  = 40                 No key has this value
     DPY-+%2                    Enable ASYNC input for REG 2

3:   LABEL   3
     If REG2 = 40 GOTO 6        Branch if no key pressed
     If REG2 = 1C GOTO 1        Branch if ENTER pressed
     If REG2 = 1B GOTO 4        Branch if MORE pressed
     If REG2 = 1A GOTO 5        Branch if PRIOR pressed
     DPY-+#                     Beep if other key pressed
     GOTO 2                     Branch to restart

4:   LABEL   4
     REG0  = 0                  Up/down counter set to up
     GOTO  2                    Branch to restart

5:   LABEL   5
     REG0  = 1                  Up/down counter set to down
     GOTO 2                     Branch to restart

6:   LABEL   6
     IF REG0 = 1 GOTO 7         Branch if counter set to down
     INC REG1                   Increment REG with address
     GOTO 8

7:   LABEL   7
     DEC REG1                   Decrement REG with address

8:   LABEL   8
     READ @ REG1                Read address in REG 1
     DPY-ADDRESS $1 DATA $E     Display address and data
     REG6  = 10                 Delay loop count

9:   LABEL   9
     DEC REG6
     If REG6 > 0 GOTO 9         Delay loop branch
     GOTO 3
```

APPENDIX D

LIST OF MATERIALS

LIST OF MATERIALS 11/29/88

ASSEMBLY -0001

ITEM	QTY	PART NUMBER	DESCRIPTION	REF DESIGNATOR
1	1	XXXXXXX-XXXX	PRINTED WIRING BOARD (4 LAYER)	
2	1	2458308-0001	IC, TMS9995,MICROCOMPUTER,16-BIT,CER	U1
3	6	2458149-0001	IC, 74LS374,F/F,EDGE-TRIG,D-TYPE	U10-U15
4	1	2587387-0001	IC, REF-02EZ,VOLT REF/TEMP XDUCER,5V	U16
5	1	2600446-0001	IC, OP400,OP AMP,QUAD,HIGH ACCURACY	U17
6	1	2457901-0001	IC, 74LS00,NAND,QUAD 2-INPUT	U18
7	1	2458302-0001	IC, TMS9901,PROG SYSTEM INTERFACE	U2
8	1	XXXXXXX-XXXX	SOCKET, SELF TEST	U23
9	4	2702227-0010	IC, 16L8,PAL,10nS,180mA,PLASTIC	U3,U20-U22
10	1	2458007-0001	IC, 74LS138,DECODER,DEMUX,3-8,2-ST	U4
11	1	2492362-0008	IC, EPROM,2764A,200nS,8K x 8,5%,CER	U5
12	1	2702198-0004	IC, 6264,SRAM,8K x 8,CMOS,150nS	U6
13	3	2458095-0001	IC, 74LS245,OCTAL BUS XCEIVER	U7-U9
14	2	2459197-0302	CAP, 10 uF,25V,25%,RDL,TANT	C1,C8
15	1	2460202-0019	CAP, 56 pF,200V,10%,RDL,CER	C32
16	2	2458991-0100	CAP, 15 pF,100V,10%,AXL,CER	C6,C7
17	22	2499415-0036	CAP, 0.1 uF,50V,10%,RDL,CER	C9-C30
18	39	2458381-0018	ZENER, 1N751A,5.1V;400mW	CR1-CR39
19	1	2460503-0020	CONN, HEADER,STRT DUAL ROW,40-POS	J1
20	1	2498081-0025	CONN, MALE,0.1 CTR,25-POS	J2
21	1	2459190-0048	RES, VARIABLE,10K OHM,10%,25-TURN	R1
22	3	2458320-0093	RES, 10K OHM,1/4W,5%,CARB FILM	R10,R11,R13
23	1	2458320-0062	RES, 510 OHM,1/4W,5%,CARB FILM	R12
24	1	2458320-0117	RES, 100K OHM,1/4W,5%,CARB FILM	R2
25	2	2458340-0237	RES, 2.87K OHM,1/4W,1%,MET FILM	R3,R6
26	2	2458340-0241	RES, 3.16K OHM,1/4W,1%,MET FILM	R4,R5
27	1	2458340-0253	RES, 4.22K OHM,1/4W,1%,MET FILM	R7
28	1	2458340-0218	RES, 1.82K OHM,1/4W,1%,MET FILM	R8
29	3	2458320-0069	RES, 1K OHM,1/4W,5%,CARB FILM	R9,R14,R15
30	1	2702116-1023	RES, SIP,1K OHM,1W,5%,8-PIN,7-ELE	RP1
31	5	2458332-0017	RES, DIP,100 OHM,1/4W,2%,16-PIN,8-EL	RP2-RP6
32	5	2703894-0103	RES, SIP,220/330 OHM,2%,8-PIN,12-ELE	RP7-RP11
33	1	2458381-0016	ZENER, 1N750A,4.7V,400mW	VR1
34	1	2458687-0035	XTAL, 12 MHz,QUARTZ,0.01%	Y1

APPENDIX E

BOARD SPACE REQUIREMENTS

PCB Density Report 11/29/88 21:34 PAGE 1
DWG # REV - DWG TITLE:

ITEM	QTY	PART NUMBER	LENGTH (IN.)	WIDTH (IN.)	DIP	UNIT AREA (SQ. IN.)	TOTAL AREA (SQ. IN.)	PINS
1	1	100K	0.500	0.089	N	0.083	0.083	2
2	3	10K	0.500	0.089	N	0.083	0.248	6
3	3	1K	0.500	0.089	N	0.083	0.248	6
4	1	1N750A	0.300	0.107	N	0.078	0.078	2
5	39	1N751A	0.300	0.107	N	0.078	3.046	78
6	1	2458302-0001	2.090	0.600	Y	2.151	2.151	40
7	1	2458308-0001	2.090	0.600	Y	2.151	2.151	40
8	1	2458308-0002	2.090	0.600	Y	2.151	2.151	40
9	5	2458332-0017	0.840	0.300	Y	0.684	3.420	80
10	1	2458340-0218	0.282	0.098	N	0.070	0.070	2
11	2	2458340-0237	0.282	0.098	N	0.070	0.140	4
12	2	2458340-0241	0.282	0.098	N	0.070	0.140	4
13	1	2458340-0253	0.282	0.098	N	0.070	0.070	2
14	1	2458687-0035	0.400	0.142	N	0.118	0.118	2
15	2	2458991-0100	0.280	0.150	N	0.106	0.213	4
16	1	2459190-0048	0.375	0.195	N	0.157	0.157	3
17	2	2459197-0302	0.310	0.230	N	0.170	0.340	4
18	1	2460202-0019	0.200	0.125	N	0.079	0.079	2
19	1	2460503-0020	2.000	0.200	N	0.486	0.486	40
20	1	2492362-0008	1.465	0.600	Y	1.588	1.588	28
21	1	2498081-0025	2.484	0.092	N	0.268	0.268	25
22	22	2499415-0036	0.200	0.125	N	0.079	1.732	44
23	1	2702116-1023	0.800	0.100	N	0.123	0.123	8
24	1	2702198-0004	1.465	0.600	Y	1.588	1.588	28
25	4	2702227-0010	1.070	0.300	Y	0.822	3.288	80
26	5	2703894-0103	0.700	0.100	N	0.113	0.565	40
27	1	510	0.500	0.089	N	0.083	0.083	2
28	1	74LS00	0.770	0.300	Y	0.642	0.642	14
29	1	74LS138	0.840	0.300	Y	0.684	0.684	16
30	3	74LS245	1.070	0.300	Y	0.822	2.466	60
31	6	74LS374	1.070	0.300	Y	0.822	4.932	120
32	1	OP400	0.840	0.300	Y	0.684	0.684	14
33	1	REF-02EZ	0.400	0.300	Y	0.420	0.420	8

PCB Density Report 11/29/88 21:34 PAGE 2
DWG # REV - DWG TITLE:

ITEM	QTY	PART NUMBER	LENGTH (IN.)	WIDTH (IN.)	DIP	UNIT AREA (SQ. IN.)	TOTAL AREA (SQ. IN.)	PINS
					TOTALS		34.5	848

Useable Board Area: 40.0

PC Board Density: 86.1%

Pin Density: 21.2 PINS/SQ IN

Assumed number of board layers for area calculations: 4

NOTES:
Explanation of error symbols:
- L = Part has invalid length in part data base
- W = Part has invalid width in part data base
- P = Part has invalid number of pins in part data base

APPENDIX F

INSTRUCTION EXECUTION TIMES

INSTRUCTION	Opcodes & All Operands On Chip		Opcodes & Immediate Operands Off Chip; All Other Operands On Chip		Opcodes & Immediate Operands Off Chip; Source Operand Off Chip; Destination Operand On Chip ⑤		Opcodes & All Operands Off Chip		Operand Address Derivation ①	
	C1	XM1	C1	XM1	C1	XM1	C1	XM1	Source	Dest
A	4	0	5	2	6	4	8	8	A	A
AB	4	0	5	2	5	3	5	5	A	A
ABS	3	0	4	2	6	6	6	6	A	–
AI	4	0	6	4	6	4	8	8	–	–
ANDI	4	0	6	4	6	4	8	8	–	–
B	3	0	4	2	4	2	4	2	A	–
BL	5	0	6	2	7	4	7	4	A	–
BLWP	11	0	12	2	14⑤	6⑤	17	12	A	–
C	4	0	5	2	6	4	7	6	A	A
CB	4	0	5	2	5	3	5	4	A	A
CI	4	0	6	4	6	4	7	6	–	–
CKOF	7	0	8	2	8	2	8	2	–	–
CKON	7	0	8	2	8	2	8	2	–	–
CLR	3	0	4	2	5	4	5	4	A	–
COC	4	0	5	2	6	4	7	6	A	–
CZC	4	0	5	2	6	4	7	6	A	–
DEC	3	0	4	2	6	6	6	6	A	–
DECT	3	0	4	2	6	6	6	6	A	–
DIV (ST4 is set)	6	0	7	2	8	4	10	8	A	–
DIV (ST4 is reset)②	28	0	29	2	30	4	34	12	A	–
DIVS (ST4 is set)	10	0	11	2	12	4	36	8	A	–
DIVS (ST4 is reset)②	33	0	34	2	35	4	39	12	A	–
IDLE③	7+2I	0	8+2I	2	8+2I	2	8+2I	2	–	–
INC	3	0	4	2	6	6	6	6	A	–
INCT	3	0	4	2	6	6	6	6	A	–
INV	3	0	4	2	6	6	6	6	A	–
JUMP (All Jump Instructions)	3	0	4	2	4	2	4	2	–	–
LDCR (C=0)	41	0	42	2	43	4	44	6	A	–
LDCR (1≤C≤15)	9+2C	0	10+2C	2	11+2C	4	12+2C	6	A	–
LI	3	0	5	4	5	4	6	6	–	–
LIMI	5	0	7	4	7	4	7	4	–	–
LREX	7	0	8	2	8	2	8	2	–	–
LST	5	0	6	2	6	2	7	4	–	–
LWP	4	0	5	2	6	2	6	4	–	–
LWPI	4	0	6	4	6	4	6	4	–	–
MOV	3	0	4	2	5	4	6	6	A	A
MOVB	3	0	4	2	4	3	4	4	A	A
MPY†	23	0	24	2	25	4	28	10	A	–
MPY‡	23	0	25	4	26	6	29	12	A	–
MPYS	25	0	26	2	27	4	30	10	A	–
NEG	3	0	4	2	6	6	6	6	A	–
ORI	4	0	6	4	6	4	8	8	–	–
RSET	7	0	8	2	8	2	8	2	–	–
RTWP	6	0	7	2	7⑦	2⑦	10	8	–	–
S	4	0	5	2	6	4	8	8	A	A

† These values will apply to future parts following Revision B. (Revision B parts are identifiable by a "B" in the date code of the symbolization on the part. Future versions will use D, E, etc. instead of B.)

‡ These values apply up to and including Revision B.

INSTRUCTION	Opcodes & All Operands On Chip		Opcodes & Immediate Operands Off Chip; All Other Operands On Chip		Opcodes & Immediate Operands Off Chip; Source Operand Off Chip; Destination Operand On Chip ⑥		Opcodes & All Operands Off Chip		Operand Address Derivation ①	
	C1	XM1	C1	XM1	C1	XM1	C1	XM1	Source	Dest
SB	4	0	5	2	5	3	5	5	A	A
SBO	8	0	9	2	9	2	10	4	–	–
SBZ	8	0	9	2	9	2	10	4	–	–
SETO	3	0	4	2	5	4	5	4	–	–
SHIFT (C≠0)	5+C	0	6+C	2	6+C	2	8+C	6	–	–
SHIFT (C=0, Bits 12-15 of WRO=0)	23	0	24	2	24	2	27	8	–	–
SHIFT (C=0, Bits 12-15 of WRO=N≠0)	7+N	0	8+N	2	8+N	2	11+N	8	–	–
SOC	4	0	5	2	6	4	8	8	A	A
SOCB	4	0	5	2	5	3	5	5	A	A
STCR (C=0)	43	0	44	2	46	6	47	8	A	–
STCR (1≤C≤8)	19+C	0	20+C	2	22+C	6	23+C	8	A	–
STCR (9≤C≤15)	27+C	0	28+C	2	30+C	6	31+C	8	A	–
STST	3	0	4	2	4	2	5	4	–	–
STWP	3	0	4	2	4	2	5	4	–	–
SWPB	13	0	14	2	16	6	16	6	A	–
SZC	4	0	5	2	6	4	8	8	A	A
SZCB	4	0	5	2	5	3	5	5	A	A
TB	8	0	9	2	9	2	10	4	–	–
X④	2	0	3	2	4	4	4	4	A	–
XOP	15	0	16	2	18⑤	6⑤	22	14	A	–
XOR	4	0	5	2	6	4	8	8	A	–
Interrupt Context Switch (For any interrupt, including Reset, NMI, MID, and overflow)	14⑧	0⑧	17⑤	6⑤	17⑤	6⑤	20⑨	12⑨	–	–

NOTES:

① Additional cycles to be added, if appropriate, are listed in Table "A" (Table 11).

② Execution time is dependent upon the partial quotient after each clock cycle during execution. Clock cycles shown are for worst-case operands.

③ Will remain in idle state until an unmasked interrupt request occurs (I = number of CLKOUT cycles until request occurs).

④ Execution time shown does not include execution time of instruction at source address.

⑤ Trap vector off chip; New workspace on chip.

⑥ Registers for register-only instructions are on chip (Shift instructions, STST, LST, STWP, LWP) and registers for instructions where an additional register is required are on-chip (AI, ANDI, BL, CI, LDCR, LI, ORI, SBO, SBZ, STCR, TB, Shift instructions).

⑦ Workspace on chip

⑧ Trap vector on chip; New workspace on chip (NMI only)

⑨ Trap vector and New workspace on chip

APPENDIX G

SET-UP SEQUENCE

BYTES TRANSMITTED		DESCRIPTION OF INFORMATION
39		9
39		9
39		9
35		5
20		SPACE
20		SPACE
50		THIS IS A 16 BIT POD
02		TWO ADDRESS RANGES
00	<=======	MEMORY STARTING AT 0000
00	:	
00	:	
00	<=======	
00	<=======	MEMORY ENDING AT FFFF
00	:	
FF	:	
FF	<=======	
00	<=======	CRU MEMORY STARTING
10	:	AT 10000
00	:	
00	<=======	
00	<=======	CRU MEMORY ENDING
1F	:	AT 1FFFFF
FF	:	
FF	<=======	
00	<=======	BUSS TEST ADDRESS
00	:	AT F080
F0	:	
80	<=======	
00	<=======	RUNUUT ADDRESS
00	:	AT 0000
00	:	
00	<=======	
07		7 STATUS LINES
07		ENABLE BITS FOR READY, HOLD, AND INTERRUPTS
52		R
45		E
41		A
44		D
59		Y
20		SPACE
48		H
4F		O
4C		L
44		D
20		SPACE
20		SPACE
49		I
4E		N
54		T
52		R
20		SPACE
20		SPACE
C1		CHECKSUM 2'S COMPLEMENT

VITA

Mark Steven Sitter was born in Erie, Pennsylvania December 8, 1959. He graduated with a BS in Electrical Engineering in June 1982 from Virginia Polytechnic Institute and State University.

After graduation, Mark went to work with Texas Instruments in Lubbock, Texas where he worked as a Quality Control Engineer. In 1983, Mark transferred to the Industrial Systems Division of Texas Instruments, Johnson City, Tennessee as a Production Engineer.

In 1984, he began working on his Master of Engineering Degree with a major in Electrical Engineering from the University of Tennessee and graduated in December 1988.