TEXAS
INSTRUMENTS

Dear TI-99/4A Users Group:

As part of our continuing support of the TI-99/4A, we are providing certain programs to established Users Groups for the purpose of testing different components of their home computer systems.

The enclosed diskettes contain the same programs except that one is for use with Extended BASIC software, and the other is designed for use with Mini Memory software. We have also included the schematic drawing for a small cable which will be necessary in order to perform the RS232 peripheral test.

This material is provided at no charge by Texas Instruments and is not intended for resale. It is given without representation or warranty of any kind. TI has no further obligation to support or supplement this software and documentation. Therefore, we assume no responsibility and shall have no liability, consequential or otherwise, arising from its use. This material was developed and is considered the property of Texas Instruments. We reserve the right to use, publish, reproduce or sell this material in any manner desired without compensation of any kind.

We at TI truly appreciate the support your organization has given our company and products. If we may be of assistance to you in the future, please call 806/741-2663 or write to the letterhead address.

Cordially,

Pat Hicks

Consumer Relations

/bjc
Enclosure

FE-09-CR-282/06

TI 99/4A

RS232 CARD

TEST INTERPRETATION

The assembly language test for the RS232 card is designed to exercise all of the functions available on the device. If during the course of the test cycle, an error is detected, the test will halt and one of several error messages will be printed to the monitor screen. This document is intended to aid the repair technician in determining the cause of the failure. The error messages are presented here in the order in which they are encountered in the test.

1. ILLEGALLY DRIVEN BUS - The software turns off all peripheral CRU enable bits and reads at 4000; if the the result is not zero, this error is displayed. The fault may be located by using EASYBUG debugger to read the bus.

2. BAD ROM - A CRC check is performed on the data in the ROM. Failure to obtain the correct checksum results in this error. Do not replace the ROM until it is verified that it is being accessed properly and there are no databus faults.

3. CRU BIT BAD - Performs a walking data read/write on the CRU registers at 1302 through 130E. The failed bit can be determined from the last digit of the error address (1-7). These bits can be manipulated with EASYBUG for troubleshooting.

4. PARALLEL PORT EXPECTED/ACTUAL - Test first turns on the output control of the parallel port output buffer, then performs a walking data read/write test. After the data test, the output control is truned off and the parallel port is read. Since the test fixture has a pull down resistor for each data line, any bits read as a one are reported as OPEN BUS LINES.

5. SERIAL PORT DTR ERRORS - Since the Data Terminal Ready inputs must work in order to transmit data, the clear to send outputs (flag 1 & 2) have been connected to DTR0 and DTR1 in the test fixture. The flags are set high and low while the CTS/DTR inputs are sampled at both UARTS. An error is reported as which UART was read, along with the input error condition. It should be remembered that flags 1 & 2 were tested during the CRU test. However, the I/O circuitry and level shifters were not tested. This circuitry may be verified by setting the flags in EASYBUG (flag 1 @ 130A, 2 @ 130C). This is the first test to require a response from the 9902s. The inputs are read as CRU bits at 1376 & 1378 for DTR0 and 13B6 & 13B8 for DTR1.

6. TOO MUCH TIME TRANSMIT BUFFER X - Between the DTR input test and this error, several things have occured: the UARTs have been initialized to 8 data bits, 1 stop bit, 300 BAUD, no parity. Once this is done, the selected UART (0 first) is turned on by writing a one to the RTS bit (UART 0 @ 1360, UART 1 @ 13A0). This should cause the Carrier Detect LED for the selected UART to switch. The program now reads the Transmitt Buffer Register Empty status bit (CRU @ 136C & 13AC). If this bit is not a one by the specified time, "TOO MUCH TIME TRANSMIT BUFFER X" IS PRINTED. Once the transmit buffer is empty, the transmit buffer is loaded with a data word. As soon as it is loaded, transmission should begin.

7. TOO MUCH TIME RECEIVE BUFFER X - Now the program begins testing the Receive Buffer Loaded bit in the receiving UART. This bit is set when a valid input word and stop bit is detected. If this does not occur within a specified time limit, the error "TOO MUCH TIME RECIEIVE BUFFER X" is printed.

8. SERIAL DATA ERROR - Once the 9902 detects a valid input word made up of a start bit, 8 data bits, and a stop bit, and the RBL bit is set, the received data word is compared to the word that was sent to the transmitting 9902. if the words are not equal, the error "SERIAL DATA ERROR X TO X " is printed, depending on the direction of transmission. After good data is detected, RTS is turned off, causing the Carrier Detect LED to switch to its reset condition.

Serial port testing begins with UART 0 transmitting and UART 1 receiving. When a succesful T/R cycle is complete, the same data is transmitted in the opposite direction. The data word is shifted left one bit, and the entire process is repeated eight times.

Occasionally an RS232 card will not allow the system to complete power up initialization, resulting in a "blue screen lock up". This condition is usually tracable to a problem on the ILA output to External Interrupt.

Attached to this document is a memory map to aid in using the EASYBUG deugger to troubleshoot the card. Happy hunting!

RS232 CARD MEMORY MAP

4000 - 4FFE	DSR ROM
5000 - 5FFE	PARALLEL I/O

RS232 CARD CRU OUTPUT BIT DEFINITION

ADDR	BIT	DEFINITION
1300	0	DSR ROM Page Enable, 1 = enabled
1302	1	Parallel Port mode set, 1 = input mode
1304	2	Parallel Port Strobe bit
1306	3	Spare Parallel Port bit
1308	4	Flag 0
130A	5	Clear to Send, RS232 Port 0, 0 = active
130C	6	Clear to Send, RS232 Port 1, 0 = active
130E	7	Indicator LED control, 1 = LED ON

RS232 CARD CRU INPUT BIT DEFINITION

ADDR	BIT	DEFINITON
1300	0	Spare
1302	1	Parallel Port configuration sense
1304	2	Parallel Port Acknowledge sense bit
1306	3	Spare Parallel Port sense bit
1308	4	Flag 0
130A	5	Clear to Send, RS232 Port 0 sense
130C	6	Clear to Send, RS232 Port 1 sense
130E	7	LED state sense

9902 UART BASE ADDRESSES

UART 0 = 1340
UART 1 = 1380

wb3/rsinst

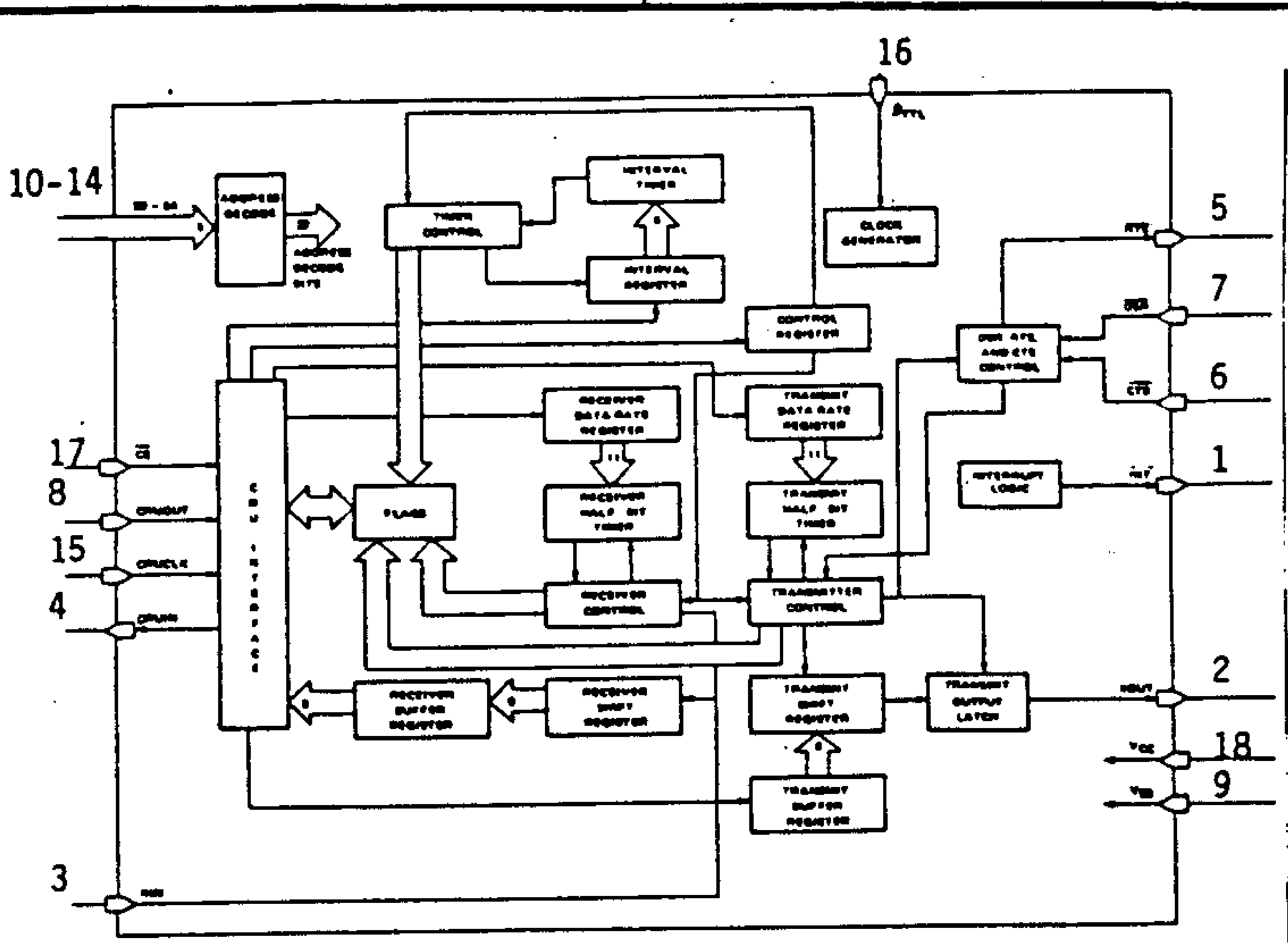


FIGURE 2. TMS 9902 ASYNCHRONOUS COMMUNICATIONS CONTROLLER (ACC) BLOCK DIAGRAM

SYSTEM CRU ADDRESS = UART BASE ADDRESS + BIT ADDRESS X 2

UART 0 BASE = 1340
UART 1 BASE = 1380

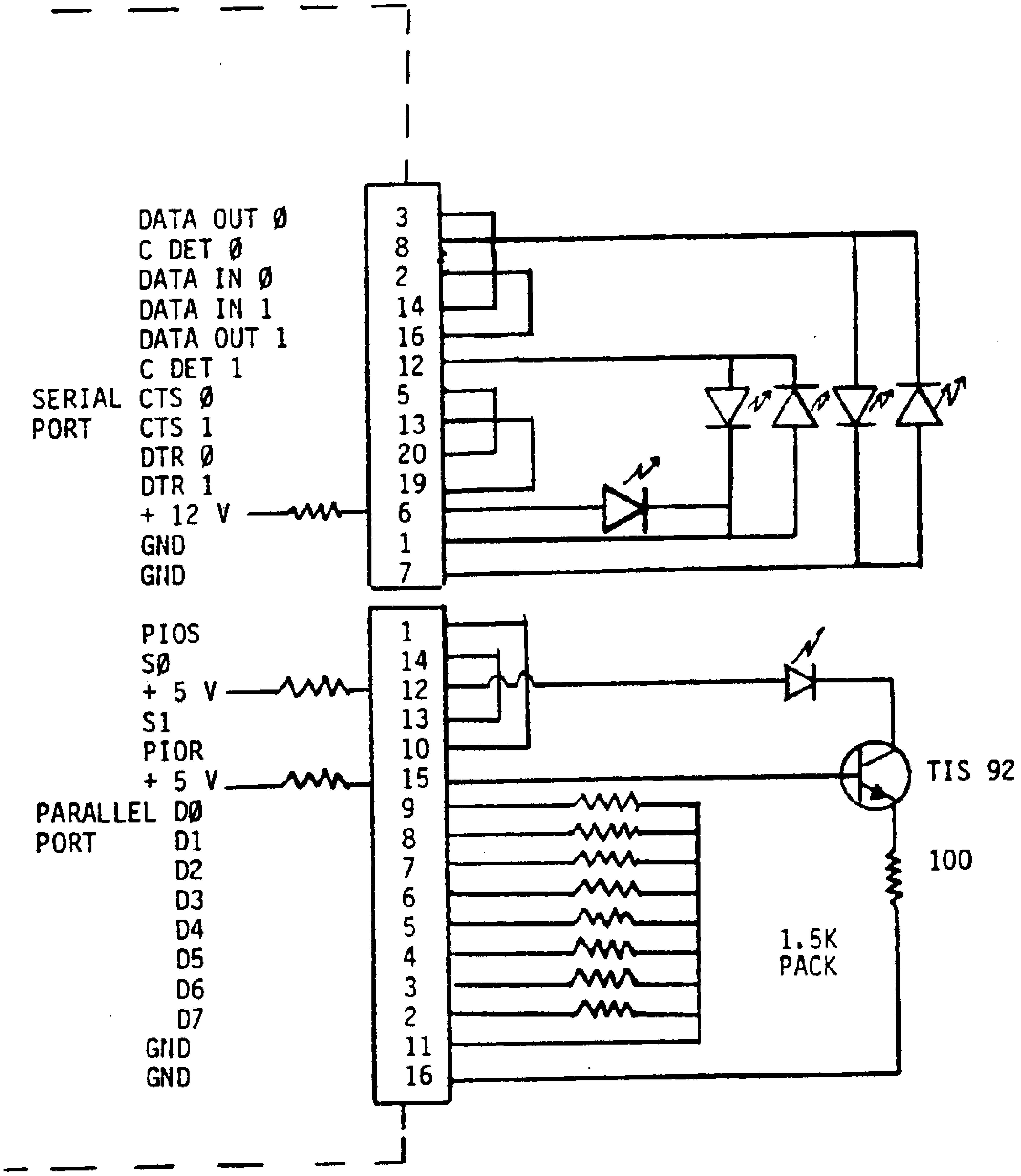
TMS 9902 ACC OUTPUT BIT ADDRESS ASSIGNMENTS

ADDRESS$_2$					ADDRESS$_{10}$	NAME	DESCRIPTION
S0	S1	S2	S3	S4			
1	1	1	1	1	31	RESET	Reset device.
					30-22		Not used.
1	0	1	0	1	21	DSCENB	Data Set Status Change Interrupt Enable.
1	0	1	0	0	20	TIMENB	Timer Interrupt Enable
1	0	0	1	1	19	XBIENB	Transmitter Interrupt Enable
1	0	0	1	0	18	RIENB	Receiver Interrupt Enable
1	0	0	0	1	17	BRKON	Break On
1	0	0	0	0	16	RTSON	Request to Send On
0	1	1	1	1	15	TSTMD	Test Mode
0	1	1	1	0	14	LDCTRL	Load Control Register
0	1	1	0	1	13	LDIR	Load Interval Register
0	1	1	0	0	12	LRDR	Load Receiver Data Rate Register
0	1	0	1	1	11	LXDR	Load Transmit Data Rate Register
					10-0		Control, Interval, Receive Data Rate, Transmit Data Rate, and Transmit Buffer Registers

TMS 9902 ACC INPUT BIT ADDRESS ASSIGNMENTS

ADDRESS$_2$					ADDRESS$_{10}$	NAME	DESCRIPTION
S0	S1	S2	S3	S4			
1	1	1	1	1	31	INT	Interrupt
1	1	1	1	0	30	FLAG	Register Load Control Flag Set
1	1	1	0	1	29	DSCH	Data Set Status Change
1	1	1	0	0	28	CTS	Clear to Send
1	1	0	1	1	27	DSR	Data Set Ready
1	1	0	1	0	26	RTS	Request to Send
1	1	0	0	1	25	TIMELP	Timer Elapsed
1	1	0	0	0	24	TIMERR	Timer Error
1	0	1	1	1	23	XSRE	Transmit Shift Register Empty
1	0	1	1	0	22	XBRE	Transmit Buffer Register Empty
1	0	1	0	1	21	RBRL	Receive Buffer Register Loaded
1	0	1	0	0	20	DSCINT	Data Set Status Change Interrupt (DSCH · DSCENB)
1	0	0	1	1	19	TIMINT	Timer Interrupt (TIMELP · TIMENB)
1	0	0	1	0	18	–	Not Used (always = 0)
1	0	0	0	1	17	XBINT	Transmitter Interrupt (XBRE · XBIENB)
1	0	0	0	0	16	RBINT	Receiver Interrupt (RBRL · RIENB)
0	1	1	1	1	15	RIN	Receive Input
0	1	1	1	0	14	RSBD	Receive Start Bit Detect
0	1	1	0	1	13	RFBD	Receive Full Bit Detect
0	1	1	0	0	12	RFER	Receive Framing Error
0	1	0	1	1	11	ROVER	Receive Overrun Error
0	1	0	1	0	10	RPER	Receive Parity Error
0	1	0	0	1	9	RCVERR	Receive Error
0	1	0	0	0	8	–	Not Used (always = 0)
					7-0	RBR7 - RBR0	Receive Buffer Register (Received Data)

SIZE A		DRAWING NO. RS232 CARD UART MEMORY MAP
SCALE	REV	SHEET 4



RS232 TEST ADAPTER

TEXAS INSTRUMENTS
INCORPORATED
DIGITAL SYSTEMS DIVISION
HOUSTON, TEXAS

A	3/6/84	REV
	SHEET 1 of 1	

* TI 99/4A SOFTWARE TEST SYSTEM *

*SOFTWARE INCLUDES:

ONE DISKETTE WHICH CONTAINS EIGHTEEN TEST PROGRAMS WRITTEN TO TEST THE TI 99/4A CONSOLE & PERIPHERALS.

*SYSTEM REQUIREMENTS:

TI 99/4A CONSOLE,MINI MEMORY COMMAND MODULE,DISK CONTROLLER AND DISK DRIVE. FOUR OF THE PROGRAMS ALSO REQUIRE EXTENDED BASIC COMMAND MODULE AND MEMORY EXPANSION.

* LOADING INSTRUCTION FOR MINI MEMORY *

1.INSERT MINI MEMORY COMMAND MODULE

2.TURN UNIT ON (TITLE SCREEN IS DISPLAYED)

3.PRESS ANY KEY (SELECTION LIST IS DISPLAYED)

4.SELECT #3 FOR MINI MEMORY (SELECTION LIST #2 IS DISPLAYED)

5.SELECT #3 FOR RE-INITIALIZE (PROMPT APPEARS AT THE TOP OF THE SCREEN)

6.PRESS RROCEED <FCTN 6} (SELECTION LIST #2 IS DISPLAYED)

7.SELECT #1 FOR LOAD AND RUN ("FILE NAME" IS DISPLAYED)

8.TYPE "DSK1.FILE NAME" PRESS ENTER

9.AFTER LOADING IS COMPLETE "FILE NAME" IS DISPLAYED AGAIN

10.PRESS ENTER ("PROGRAM NAME" IS DISPLAYED)

11.TYPE "RUN" PRESS ENTER

THE PROGRAM SHOULD BE RUNNING,DISPLAYING A SELECTION LIST FROM ONE OF THE TEST PROGRAMS PROVIDED AND WAITING FOR A SELECTION TO BE MADE.

WHEN TESTING IS COMPLETE TURN UNIT OFF BEFORE REMOVING COMMAND MODULE. THIS WILL ALLOW THE USE OF THIS PROGRAM IN OTHER UNITS OR AT ANOTHER TIME BY FOLLOWING THE STEPS BELOW:

1.INSERT MINI MEMORY

2.TURN UNIT ON (TITLE SCREEN APPEARS)

3.PRESS ANY KEY (SELLECTION LIST APPEARS)

4.SELECT #3 FOR MIMI MEMORY (SELECTION LIST #2 APPEARS)

5.SELECT #2 FOR RUN ("PROGRAM NAME" APPEARS)

6.PRESS ENTER (PROGRAM BEGINS)

TABLE OF CONTENTS

FILE NAME	TEST PROGRAMS
*DIAGNOSTIC	1.DIAGNOSTIC
	2.KEYBOARD TEST
	3.SPEECH TEST
	4.JOYSTICK TEST
*P/CARDS	1.PASCAL CARD TEST
	2.MEMORY EXPANSION TEST
	3.RS232 INTERFACE TEST
	4.CASSETTE TEST
	5.BIT-MAP MODE TEST

THE FOLLOWING TEST PROGRAMS REQUIRE EXTENDED BASIC COMMAND MODULE AND MEMORY EXPANSION FOR EXECUTION.

1.IMPACT SERIAL PRINTER TEST

2.SPEECH TEST

3.THERMAL PRINTER TEST

4.IMPACT PARALLEL PRINTER TEST

*BELOW IS EXECUTING INSTRUCTION FOR THE FOUR PRECEDING PROGRAMS:

1.INSERT EXTENDED BASIC COMMAND MODULE

*TITLE SCREEN IS DIAPLAYED

2.PRESS ANY KEY

*SELECTION LIST IS DISPLAYED

3.SELECT #2 FOR EXTENDED BASIC

*THE LOADING PROCESS BEGINS AFTER A SLIGHT DELAY AND THE SELECTION LIST IS DISPLAYED.

* DIAGNOSTIC *

FILE NAME: DIAGNOSTIC   PROGRAM NAME: RUN

THE DIAGNOSTIC TEST CONSISTS OF SEVEN INDIVIDUAL PROGRAMS LINKED TOGETHER TO FORM A COMPLETE CONSOLE TEST EXCLUDING KEYBOARD AND INPUT-OUTPUT PORT.

***** 1. COLOR TEST:

THIS TEST PLACES ALL 16 COLORS ON THE SCREEN FOR VISUAL INSPECTION AND WAITS FOR KEY ENTRY FROM THE OPERATOR BEFORE PROCEEDING TO THE NEXT TEST.

***** 2. SOUND TEST:

THIS TEST REQUIRES A MID-RANGE VOLUME SETTING.

IT CONSISTS OF 5 DISTINCT SOUNDS WHICH EXERCISE ALL 3 SOUND GENERATORS AND NOISE GENERATOR.

***** 3. SPRITE-COINCIDENCE TEST:

THIS TEST PLACES 11 SPRITES ON THE SCREEN AND MOVES 2 OF THEM DOWN AND ACROSS THE SCREEN FOR VISUAL INSPECTION WHILE CHECKING FOR BOTH FIFTH SPRITE AND COINCIDENCE.

***** 4. CHARACTER & TEXT MODE:

THIS TEST DISPLAYS IN THE TEXT MODE, 4 ROWS OF 100 CHARACTERS FOR APPROXIMATELY THREE SECONDS FOR VISUAL INSPECTION.

***** 5. VDP TEST (DYNAMIC RAMS)

***** 6. ROM TEST

***** 7. GROM TEST

***** 8. 6810s (STATIC RAMS)

THE PREVIOUS LISTED MEMORY DEVICES ALL FOLLOW THE SAME TESTING FORMAT. IF THE DEVICE TESTS GOOD, "GOOD __?__" IS DISPLAYED AND THE PROGRAM PROCEEDS TO THE NEXT TEST. IF A DEVICE IS DEFECTIVE, "BAD __?__" IS DISPLAYED AND THE PROGRAM STOPS AT THAT POINT. AFTER THE DIAGNOSTIC TEST IS COMPLETE, PRESS ENTER TO RETURN TO THE SELECTION LIST.

THE FOLLOWING INFORMATION CONCERNING THE KEYBOARD, SPEECH, AND JOYSTICK TEST COMPLETES THE SELECTION LIST ACCOMPANYING THE DIAGNOSTIC TEST.

* FILE NAME: DIAGNOSTIC *PROGRAM NAME: RUN

2. KEYBOARD TEST:

A. FOLLOW PROMPTS (PRESS SPACE BAR OR !, 1, @, A, x, ETC.) DISPLAYED. WHEN SMALL CHARACTER "x" IS INDICATED, THE ALPHA LOCK MUST BE RELEASED. ENTER "x" AND RELOCK ALPHA LOCK. THE TEST IS THEN CONTINUED.

B. IF A MULTIPLE ENTRY IS DETECTED, "MULTIPLE ENTRY TEST "_" KEY" IS DISPLAYED. THE ASCII CHARACTER IS PLACED IN THE QUOTATIONS AND A ROW IS RESERVED BELOW PROMPT FOR TESTING. AFTER THE KEY IS THOROUGHLY TESTED, PRESS ENTER AND THE TEST RESUMES AT THE POINT OF INTERRUPTION.

C. IF A KEY DOESN'T ENTER OR IF A WRONG ENTRY IS DETECTED, THE FOLLOWING PROMPT IS DISPLAYED "WRONG ENTRY TEST "_" KEY" WHERE THE CORRECT ENTRY IS DISPLAYED IN QUOTATIONS AND A ROW IS RESERVED BELOW FOR TESTING. PRESS ENTER TO RESUME THE TEST.

3. SPEECH TEST:

THIS TEST REQUIRES A MID-RANGE VOLUME SETTING. IT SAYS "READY TO START. DID YOU UNDERSTAND ME?" AS THE WORDS ARE DISPLAYED. THIS TEST WAS WRITTEN TO TEST INPUT-OUTPUT CIRCUITRY OF THE CONSOLE.

4. JOYSTICK TEST:

THIS TEST DISPLAYS THE NUMBERS 1 & 2 INDICATING THE JOYSTICK CONTROL NUMBER. AS THE JOYSTICK IS MOVED, THE EIGHT CONTACT POINTS ARE INDICATED BY AN ASTERISK DISPLAYED IN THE RELATIVE POSITION AROUND THE NUMBER. THE FIRE BUTTON OF JOYSTICK #1 CLEARS THE ASTERISKS AND THE FIRE BUTTON OF JOYSTICK #2 RETURNS TO THE SELECTION LIST.

* PERIPHERAL TEST SYSTEM *

*FILE NAME: P/CARDS *PROGRAM NAME: RUN

*P-CODE CARD TEST:

THIS PROGRAM WAS WRITTEN TO INDIVIDUALLY TEST THE TWO ROMS, EIGHT GROMS AND ASSOCIATED CIRCUITRY. THE PROGRAM ALSO INDICATES WHICH DEVICE IS BEING TESTED AND THE DEFECTIVE DEVICE.

*MEMORY EXPANSION TEST:

THIS PROGRAM TESTS THE 32K BYTES OF MEMORY EXPANSION. IT INDICATES THE AREA OF MEMORY BEING TESTED AND STOPS IF AN ERROR IS DETECTED AND INDICATES THE AREA IN WHICH THE ERROR WAS DETECTED.

*RS232 INTERFACE TEST:

THIS PROGRAM TESTS THE ROM AND BOTH THE SERIAL AND PARALLEL PORTS. THIS TEST REQUIRES A SPECIAL SERIAL TO PARALLEL CONNECTOR. IF AN ERROR IS DETECTED, THE PROGRAM STOPS AND A DIAGNOSTIC ERROR MESSAGE IS DISPLAYED.

*THE THREE PRECEDING PROGRAMS WERE WRITTEN IN A CONSTANT LOOP. THIS MEANS
*
*THAT THE TESTING CONTINUES UNTIL AN ERROR IS DETECTED, THE UNIT IS TURNED
*
*OFF, OR A QUIT IS PERFORMED.

*CASSETTE TEST:

THIS PROGRAM TESTS THE CONSOLE CASSETTE CIRCUITRY BY WRITING 2K BYTES FROM THE CONSOLE TO THE CASSETTE AND THEN BACK. A BYTE FOR BYTE COMPARISON IS PERFORMED AND THE RESULT IS INDICATED. TO EXECUTE, SELECT #4 AND FOLLOW PROMPTS.

*BIT-MAP MODE TEST:

THIS PROGRAM SWITCHES FROM GRAPHIC TO BITMAP MODE AND BACK AT APPROXIMATELY 2 SECOND INTERVALS. "DEFECTIVE BIT MAP MODE" WILL BE DISPLAYED AT THE TOP AND BOTTOM OF THE SCREEN IF DEFECTIVE. THIS MODE IS USED FOR MORE DESCRIPTIVE GRAPHICS SUCH AS PARSEC.

THE FOLLOWING TEST PROGRAMS REQUIRE EXTENDED BASIC COMMAND MODULE FOR EXECUTION.(SEE PAGE 2 FOR MORE INFORMATION.)

*SERIAL IMPACT PRINTER TEST:

THIS PROGRAM PRINTS (USING THE SERIAL PORT) THE CHARACTERS, SIZING,AND TYPE DENSITIES AVAILABLE THROUGH SOFTWARE IN GRAPHIC AND TEXT MODES.THE BUZZER AND CARRIAGE RETURN ARE ALSO TESTED.THE PRINTER MUST BE SET TO 300 BPS FOR TESTING.

*SPEECH TEST:

THIS PROGRAM RECITES THE ENTIRE RESIDENT VOCABULARY OF THE SPEECH SYNTHESIZER,AND IS USEFUL IN DETECTING INTERMITTENT OR THERMAL RELATED SPEECH PROBLEMS.

*THERMAL PRINTER TEST:

THIS PROGRAM IS WRITTEN TO ACTIVATE ALL THE HEATING ELEMENTS USED BY THE THERMAL PRINTER. THIS IS DONE BY PRINTING ROWS OF CHARACTERS AND DOTS FOR VISUAL INSPECTION.

*IMPACT PARALLEL PRINTER TEST:

THIS PROGRAM PRINTS (USING THE PARALLEL PORT) THE CHARACTERS, SIZING,AND TYPE DENSITIES AVAILABLE THROUGH SOFTWARE IN BOTH GRAPHIC AND TEXT MODES.THE BUZZER AND CARRIAGE RETURN ARE ALSO TESTED.

TROUBLE-SHOOTING TIPS

TEST PROGRAM	COMPLAINT	DEFECTIVE DEVICE
*DIAGNOSTIC	DISTORTED VIDEO	*CONSOLE/MONITOR/MODULATOR
"	NO COLOR	" " "
"	NO SOUND	" " "
"	DISTORTED SPRITE MOTION	"
"	ERRATIC PRINT	"
"	WON’T RUN KNOWN GOOD PROG.	"
"	WON’T SAVE PROGRAM	"
*KEYBOARD TEST	WON’T ENTER	*KEYBOARD
"	INTERMITTENT ENTRY	"
"	MULTIPLE ENTRY	"
"	WON’T PRINT LARGE CHARACTERS	"
*SPEECH TEST	NO SPEECH	*SYNTHESIZER/CONSOLE
"	GARBLED SPEECH	" "
"	STATIC IN SPEECH	" "
*JOYSTICK TEST	NO RESPONSE	*JOYSTICKS/CONSOLE
"	WON’T MOVE UP	" " (RELEASE ALPHA LOCK)
"	BOTH JOYSTICKS MOVE SIMULTANEOUSLY	*CONSOLE
"	WON’T MOVE DIAGONALLY	*JOYSTICKS

TROUBLE SHOOTING TIPS

TEST PROGRAMS	COMPLAINT	DEFECTIVE DEVICE
*PASCAL CARD TEST	LIGHT COMES ON AND COMPUTER LOCKS UP.	PASCAL CARD
"	WON'T RUN KNOWN GOOD PASCAL PROGRAM.	" "
"	NO LIGHT COMPUTER POWERS UP AND DISPLAYS TITLE SCREEN.	" "
*MEMORY EXP. TEST	COMPUTER DISPLAYS "NO MEMORY EXPANSION"	MEMORY EXP.CARD
"	WON'T EXECUTE PROGRAM FROM MEMORY EXPANSION.	" " "
"	MEMORY EXP. LIGHT COMES ON & COMPUTER LOCKS UP.	" " "
*RS232 INTERFACE	LIGHT COMES ON AND COMPUTER LOCKS UP.	RS232 INTERFACE
" "	WON'T PRINT	" "
*CASSETTE TEST	WON'T RECORD	CASSETTE OR CONSOLE
" "	WON'T READ CASSETTE	" "
" "	READS OK BUT WON'T RUN PROG.	" "

*PASCAL CARD REQUIRES MEMORY EXPANSION FOR NORMAL OPERATION.

* THE LIGHT MENTIONED IN THE COMPLAINTS ABOVE IS REFERRING TO THE CARD INDICATOR LIGHT.

TROUBLE SHOOTING TIPS

TEST PROGRAMS	COMPLAINT	DEFECTIVE DEVICE
*PASCAL CARD TEST	LIGHT COMES ON AND COMPUTER LOCKS UP.	PASCAL CARD
"	WON'T RUN KNOWN GOOD PASCAL PROGRAM.	" "
"	NO LIGHT COMPUTER POWERS UP AND DISPLAYS TITLE SCREEN.	" "

*PASCAL CARD REQUIRES MEMORY EXPANSION FOR NORMAL OPERATION.

*MEMORY EXP. TEST	COMPUTER DISPLAYS "NO MEMORY EXPANSION"	MEMORY EXP.CARD
"	WON'T EXECUTE PROGRAM FROM MEMORY EXPANSION.	" " "
"	MEMORY EXP. LIGHT COMES ON & COMPUTER LOCKS UP.	" " "
*RS232 INTERFACE	LIGHT COMES ON AND COMPUTER LOCKS UP.	RS232 INTERFACE
" "	WON'T PRINT	" "
*CASSETTE TEST	WON'T RECORD	CASSETTE OR CONSOLE
" "	WON'T READ CASSETTE	" "
" "	READS OK BUT WON'T RUN PROG.	" "

* THE LIGHT MENTIONED IN THE COMPLAINTS ABOVE IS REFERRING TO THE CARD INDICATOR LIGHT.

SOFTWARE CONTROLLED TROUBLESHOOTING TECHNIQUES

One of the difficulties in troubleshcoting microprocessor based systems like the 99/4A, in which system control is handled by dedicated ROM, lies in the fact that the flow of the program (and therefore address, data, and control lines) cannot be modified by the technician. Once power has been applied, the system is under the control of firmware masked in ROM. Practically speaking, this means that if a circuit node for an address line is probed with a scope, the result will be a very long complex pulse train as the microprocessor responds to the instructions of firmware. The technician can see the high/low signals from the scope but cannot determine if the entire address is correct or if the timing is proper. While the processor is under control of this firmware, it is not possible to stop the system to probe multiple points in order to determine proper address, data, or control.

The Mini Memory module has provided a way of escape from this delima. The EASYBUG DEBUGGER contained in the Mini Memory allows the user to:

1) Address any device in the 9900 address field.
2) Inspect and, optionally, modify the contents of RAM.
3) Display the contents of GROM and ROM.
4) Execute assembly language programs from EASYBUG.
5) Directly access devices which are controlled by the TMS 9900 microprocessor's Serial I/O Port, the communications register unit (CRU).

Although these functions are designed to allow debugging of assembly level software, EASYBUG, as a side effect, allows the user to manipulate the hardware in ways not previously possible without a great deal of trouble.

Quite a lot of manipulation is possible by using the EASYBUG subroutines as they are available by simple menu selection. More detailed checking of data, address, and control is possible with the use of very short assembly language programs designed to give predictable and regularly repeating results at certain circuit nodes.

This technique requires that the user understand the operation of the 99/4A computer. It will always be necessary for the technician to understand what signal should be at a circuit node, when it should be there, and why. Obviously these techniques are useless on units that are dead, locked up or otherwise incapable of executing a program.

In summation, the EASYBUG can do several things to aid the technician:

1) Provide a means to control the CRU bus, and therefore CRU devices such as the TMS 9901.
2) Allow activation of signals that are normally inactive to verify operation.
3) Provide a means to escape control of system firmware, allowing the technician to test functions in detail.

The approach to describing use of this technique will be in the form of a series of "EASYBUG NOTES." These will be labled as to use and procedure.

EASYBUG NOTE #1: CONTROLLING THE 9901

Required reference: CRU map for 9901

Use EASYBUG command for CRU singlebit I/O. Referring to the 9901 CRU map, enter C and the address for the function to be exercised. A one or zero may then be entered to enable or disable the function.

EXAMPLE: Audio Gate

DISPLAY	ENTRIES	RESULT
?	C0030 (ENTER)	
C0030=00	1 (ENTER)	9901 PIN 27=LL1
C0031=01	(MINUS)	
C0030=01	0 (ENTER)	9901 PIN 27=LL0
C0031=00		

9901 INPUT/OUTPUT MAP

ADDRESS	CRU BIT	PORT DESIGNATION	PIN	FUNCTION
0000	0	CONTROL		CONTROL
0002	1	INTERRUPT 1	17	EXTERNAL
0004	2	INTERRUPT 2	18	VDP VERTICAL SYNC.
0006	3	INTERRUPT 3	9	KEYBOARD: :/.,MN= JOYSTICK: FIRE
0008	4	INTERRUPT 4	8	KEYBOARD: ;LKJH SPACE JOYSTICK: LEFT
000A	5	INTERRUPT 5	7	KEYBOARD: POIUY ENTER JOYSTICK: RIGHT
000C	6	INTERRUPT 6	6	KEYBOARD: 09876 JOYSTICK: DOWN
000E	7	INTERRUPT 7 (P15)	34	KEYBOARD: 12345 JOYSTICK: UP
0010	8	INTERRUPT 8 (P14)	33	KEYBOARD: ASDFG SHIFT
0012	9	INTERRUPT 9 (P13)	32	KEYBOARD: QWERT
0014	10	INTERRUPT 10 (P12)	31	KEYBOARD: ZXCVB
0016	11	INTERRUPT 11 (P11)	30	NOT USED
0018	12	INTERRUPT 12 (P12)	29	RESERVED
001A - 1E	13 - 15	INTERRUPT 13 - 15	28,27 & 23	NOT USED
0020	16	PROGRAMMABLE 0	38	NOT USED
0022	17	PROGRAMMABLE 1	37	NOT USED
0024	18	PROGRAMMABLE 2	26	BIT 2 OF KEYBD SELECT(LSB)
0026	19	PROGRAMMABLE 3	22	BIT 1 OF KEYBD SELECT
0028	20	PROGRAMMABLE 4	21	BIT 0 OF KEYBD SELECT(MSB)
002A	21	PROGRAMMABLE 5	20	ALPHA LOCK KEY
002C	22	PROGRAMMABLE 6	19	CASSETTE MOTOR CONTROL 1
002E	23	PROG. 7 / INT. 15	23	CASSETTE MOTOR CONTROL 2
0030	24	PROG. 8 / INT. 14	27	AUDIO GATE
0032	25	PROG. 10 / INT. 12	28	MAG TAPE DATA OUT
0036	27	PROG. 11 / INT. 11	30	MAG TAPE DATA INPUT
0038 - 3E	28 - 32	PROG. 12 - PROG.15	31-34	NOT USED

EASYBUG NOTE #2: CONTROLLING MEMORY SELECTION LOGIC

Required reference: System Memory Map

Use EASYBUG command to modify CPU memory. Referring to the system memory map, ENTER M and an address that is in the block decoded by memory enable in question. When ENTER is pressed, the enable line should go active(LOW) for approximately 2us. (A storage scope or a sharp eye is required here.)

EXAMPLE: ROM gate

DISPLAY	ENTRIES	RESULT
?	M7000 (ENTER)	U504 PIN 12= LLO for 2us
M7001	(ENTER)	SAME

SYSTEM MEMORY MAP

HEX ADDRESS

0 - 1FFF Console ROM Space
2000 - 3FFF Memory Expansion
4000 - 5FFF Peripheral Expansion (predecoded to I/O Connector)
6000 - 7FFF Game Cartridge ROM/RAM (predecoded to GROM Connector)
8000 - 9FFF Microprocessor RAM, VDP, GROM, SOUND, and SPEECH select.
A000 - BFFF Memory Expansion
C000 - DFFF Memory Expansion
E000 - FFFF Memory Expansion.

EASYBUG NOTE #3 PROGRAM CONTROL OF DATA, ADDRESS AND CONTROL

Required reference: System Memory Map

Use EASYBUG command for modify CPU memory. Starting at an unused location in mini memory ROM, enter the following program:

```
7000   02
7001   00
7002   XX
7003   XX
7004   02
7005   01
7006   XX
7007   XX
7008   C4
7009   01
700A   10
700B   FE
```

Program explanation:

```
0200   Load immediate register 0.
XXXX   Data to be loaded in register 0.  (Address)
0201   Load immediate register 1.
XXXX   Data to be loaded in register 1.  (Data)
C401    Mov R1, *R0 (Move the contents of register 1 to the address
       specified by the contents of register 0)
10FE   Jump -2 (do previous instruction again)
```

Start program by using easybug command: EXXXX (where is the address of the first word of the program in RAM.

Application: This program will cause the computer to execute a move instruction in a two-instruction loop.

This creates a situation which has several advantages:

1) Address lines (when valid) should have a known state determined by the data that was loaded into register 0.

2) Data lines (when valid) should have a known state determined by the data that was loaded into register 1.

3) Control line signals should become regular & predictable due to the repetitious nature of the program.

Some applications for this program follow:

1) Memory Selection Logic Example

Test Memory Block Enable.
Enter address word in program as some value from 4000-5FFF inclusive. Data word does not matter. Upon program execution, MBE* should go active. This is applicable to any signal in memory selection logic.

EASYBUG NOTE #3 CONT'D

2) Data and Address Example

Program running as before, address @ 4000. (Do <u>not</u> attempt to write to read only memory, damage to components may result).
Use both channels on scope.
Channel 1 - WE* (9900 Pin 61).
Channel 2 - DATA or address line in question.

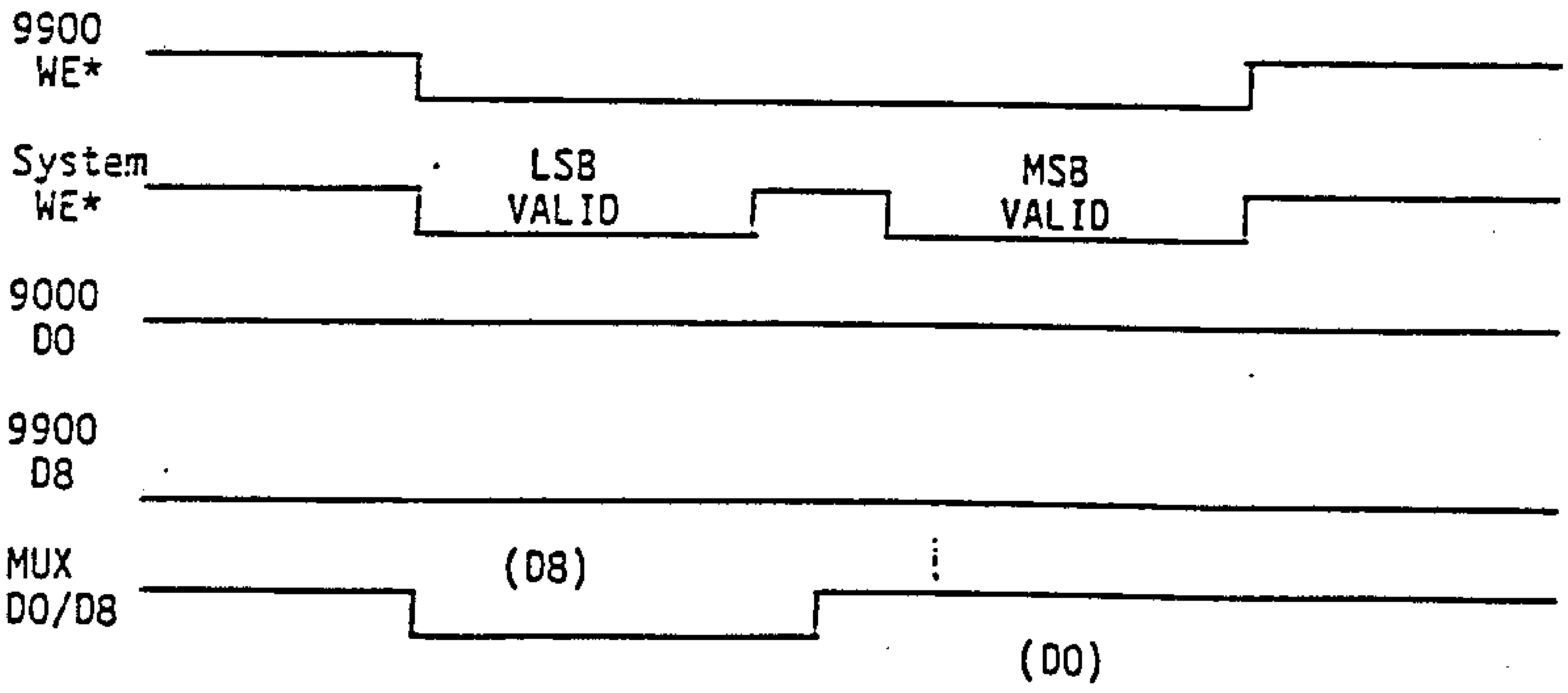
Data and address are valid when WE* is active (LOW).

Address lines and data lines should match data words for data and address in the program when WE* is true.

3) Multiplex Data Write

Run program with address @ 4000. Data may be varied. Connect scope channel 1 to U606-3 (System WE*).
Data word loaded in program can be traced through the multiplexers using channel two. System write enable* (chan 1) should display two negative going pulses. During the first of these LSB data is valid, and during the second, MSB data is valid.

EXAMPLE DATA = FF00



```
99/4 ASSEMBLER
VERSION 1.2                                                  PAGE 0001
 0001               ** TOGGLE CRU OUTPUT BIT **
 0002
 0003 7118                 AORG >7118
 0004                      DEF  RUN
 0005
 0006 7118 020C     RUN    LI   R12,>1100    LOAD CRU BIT ADDRESS
      711A 1100
 0007 711C 1D00     LOOP   SBO  0            TOGGLE HIGH
 0008 711E 1E00            SBZ  0            TOGGLE LOW
 0009 7120 10FD            JMP  LOOP
 0010               ** POKE DATA FROM LINES 5 THROUGH 8 IN ANY FREE **
 0011               ** MINI MEMORY ADDRESS SPACE IN EASYBUG MODE.   **
 0012               ** RUN FROM EASYBUG EXECUTE MODE.
 0013                      END
 0000 ERRORS
```

```
 99/4 ASSEMBLER
VERSION 1.2                                                  PAGE 0001
 0001               ** TEST CRU INPUT BIT **
 0002
 0003 7118                 AORG >7118
 0004                      DEF  RUN
 0005
 0006 7118 020C     RUN    LI   R12,>1100    LOAD CRU BIT ADDRESS
      711A 1100
 0007 711C 1F00     LOOP   TB   0            INPUT BIT
 0008 711E 10FE            JMP  LOOP
 0009               ** POKE DATA FROM LINES 5 THROUGH 7 IN ANY FREE **
 0010               ** MINI MEMORY ADDRESS SPACE IN EASYBUG MODE.   **
 0011               ** RUN FROM EASYBUG EXECUTE MODE.
 0012                      END
 0000 ERRORS
```

F