TI-99/4A Diagnostic Software

Converted by Chris Schneider (SHIFT838)

Last Updated: April 13, 2025

Table of Contents

Users Group Letter

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,

Consumer Relations

/bjc
Enclosure

FE-09-CR-282/06

TEXAS INSTRUMENTS INCORPORATED · POST OFFICE BOX 53 · LUBBOCK, TEXAS 79408

Summary

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.

Disk Software

The current disk images for the software can be obtained at the below download links:

- https://www.shift838.com/downloads/CRUTST.dsk
- https://www.shift838.com/downloads/TIDIAGMM.dsk
- https://www.shift838.com/downloads/TIDIAGXB.dsk

Error Messages

ILLEGALLY DRIVEN BUS

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

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.

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.

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 turned 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*.

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 DTRO 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 DTRO and 13B6 & 13B8 for DTR1.

TOO MUCH TIME TRANSMITE BUFFER X

Between the DTR input test and this error, several things have occurred; 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 Transmit Buffer Register Empty status bit (CRU @ 136C & 13AC). If this bit is not a one by the specified time, "**TOO MUCH TIME TRANSMIT BRUFFER 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. **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.

SERIAL DATA ERROR

Once the 9902 detects a valid input word made up of a start bit, 8 data bits, 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.

Testing Summary

Serial Port testing begins with UART 0 transmitting and UART 1 receiving. When a successful 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 traceable to a problem on the ILA output to External Interrupt.

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

RS232 Card Memory Map Table

RS232 CARD MEMORY MAP	
Address	**Definition**
4000 - 4FFE	DSR ROM
5000 - 5FFE	PARALLEL I/0

RS232 Card CRU Output Bit Definition

RS232 CARD CRU OUTPUT BIT DEFINITION			
Bit	**R12 address**	**Meaning when read**	**Effect when written**
0	>1300	Always 0	1: Turn card ROMs on
1	>1302	PIO direction 0=output 1=input	0: Set PIO as output 1: Set PIO as input
2	>1304	Status of HANSHAKEIN pin (PIO)	Set HANSHAKEOUT pin (PIO)
3	>1306	Status of SPAREIN pin (PIO)	Set SPAREOUT pin (PIO)
4	>1308	Read itself	Writes to itself
5	>130A	Status of CTS1 pin	Set CTS1 pin
6	>130C	Status of CTS2 pin	Set CTS2 pin
7	>130E	LED status	1: LED on
0-7	>1340-134E	Content of Receive buffer (8 bits)	Value to load in selected register (11 bits)
8	>1350	-	
9	>1352	1: Reception error (bit 10/11/12 =1)	
10	>1354	1: Parity error	
11	>1356	1: Overflow (bit arrived when buffer full)	1: Load Emission-Rate register
12	>1358	1: Frame error (0 received as stop bit)	1: Load Reception-Rate register (reset upon loading)
13	>135A	1: First bit has arrived	1: Load Interval register (reset upon loading)
14	>135C	1: Receiving byte (for test purposes)	1: Load Control register (reset upon loading)
15	>135E	1: Status of RIN pin	1: Test mode (RTS->CTS, XOUT->RIN, timer*32)
16	>1360	1: Reception interrupt occurred (reset by writing to bit 18)	Set RTS pin (RTS=1 only if input bits 22+23=0)
17	>1362	1: Emission interrupt occurred (reset by loading emission register)	1: Abort transmission (XOUT=0 if bits 22+23=0)
18	>1364	-	1: Enable reception interrupts

19	>1366	1: Timer interrupt occurred (reset by writing to bit 20)	1: Enable emission interrupts
20	>1368	1: CTS or RTS interrupt occurred (reset by writing to bit 21)	1: Enable timer interrupts
21	>136A	1: Receive register full (reset by writing to bit 18)	1: Enable interrupts when CTS or DSR change
22	>136C	1: Emission register empty (reset by loading emission register)	-
23	>136E	1: No data currently sent (shift register is empty)	-
24	>1370	1: Timer error (time elapsed twice) (reset by writing to bit 20)	-
25	>1372	1: Time elapsed (reset by writing to bit 20)	-
26	>1374	Status of RTS pin (inverted)	-
27	>1376	Status of DSR pin (inverted)	-
28	>1378	Status of CTS pin (inverted)	-
29	>137A	1: Change of DSR or CTS detected (reset by writing to bit 21)	-
30	>137C	1: Register being loaded	-
31	>137E	1: An interrupt occurred	1: Reset. Output bits 11-14=1, bits 17-20=0 Input bits 22,23=1, bits 13,21,25=0
0-31	>1380-13BE	Same at >1340-137E for 2nd chip	Ditto for second chip (RS232/2)

9902 UART Base Addresses

9902 UART BASE ADDRESSES	
UART	**Address**
0	1340
1	1380

TMS9902 Asynchronous Communications Controller Block Diagram

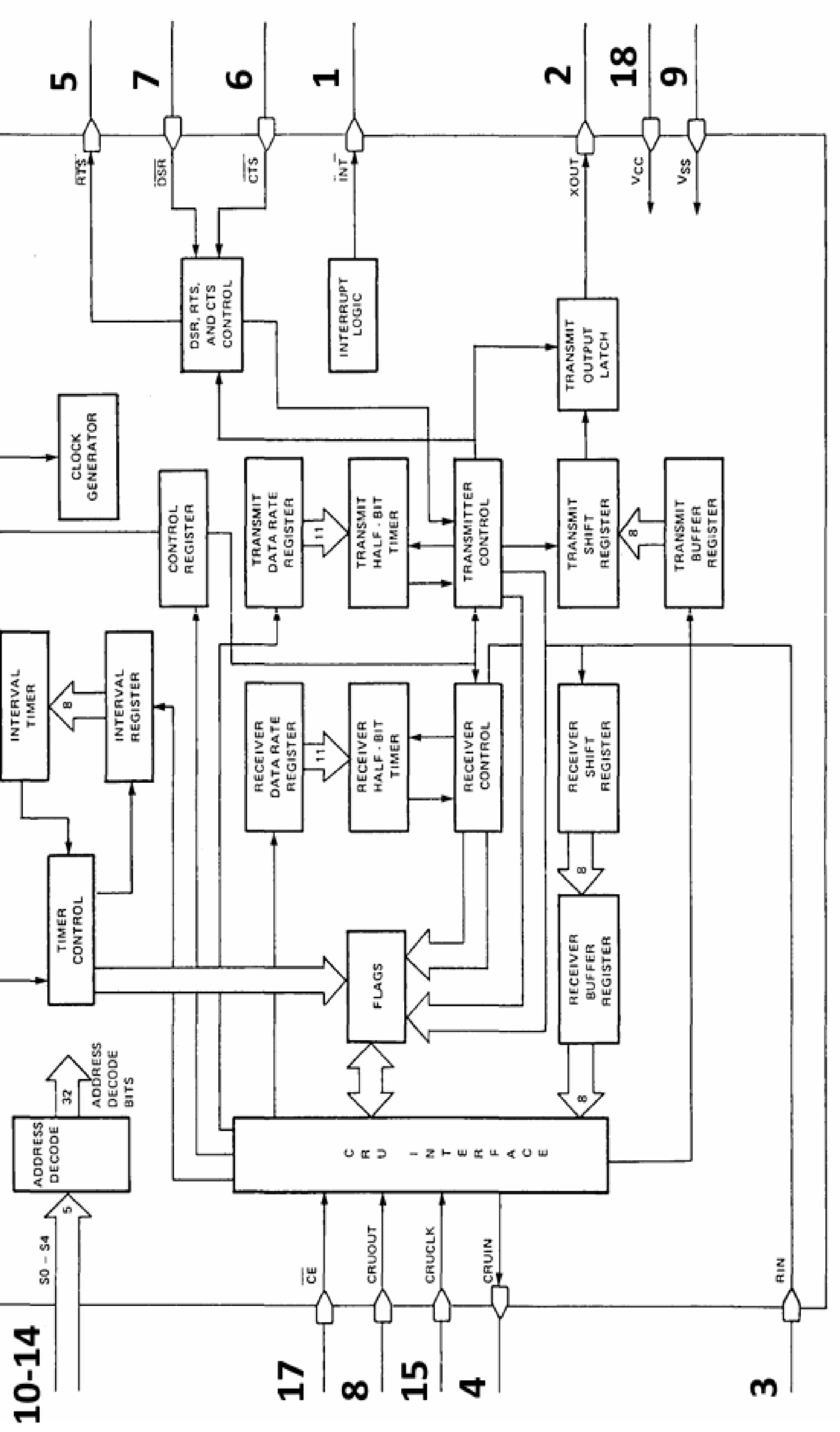


FIGURE 2. TMS 9902 ASYNCHRONOUS COMMUNICATIONS CONTROLLER (ACC) BLOCK DIAGRAM

TMS 9902 ACC Output Bit Address Assignments

TABLE 1
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

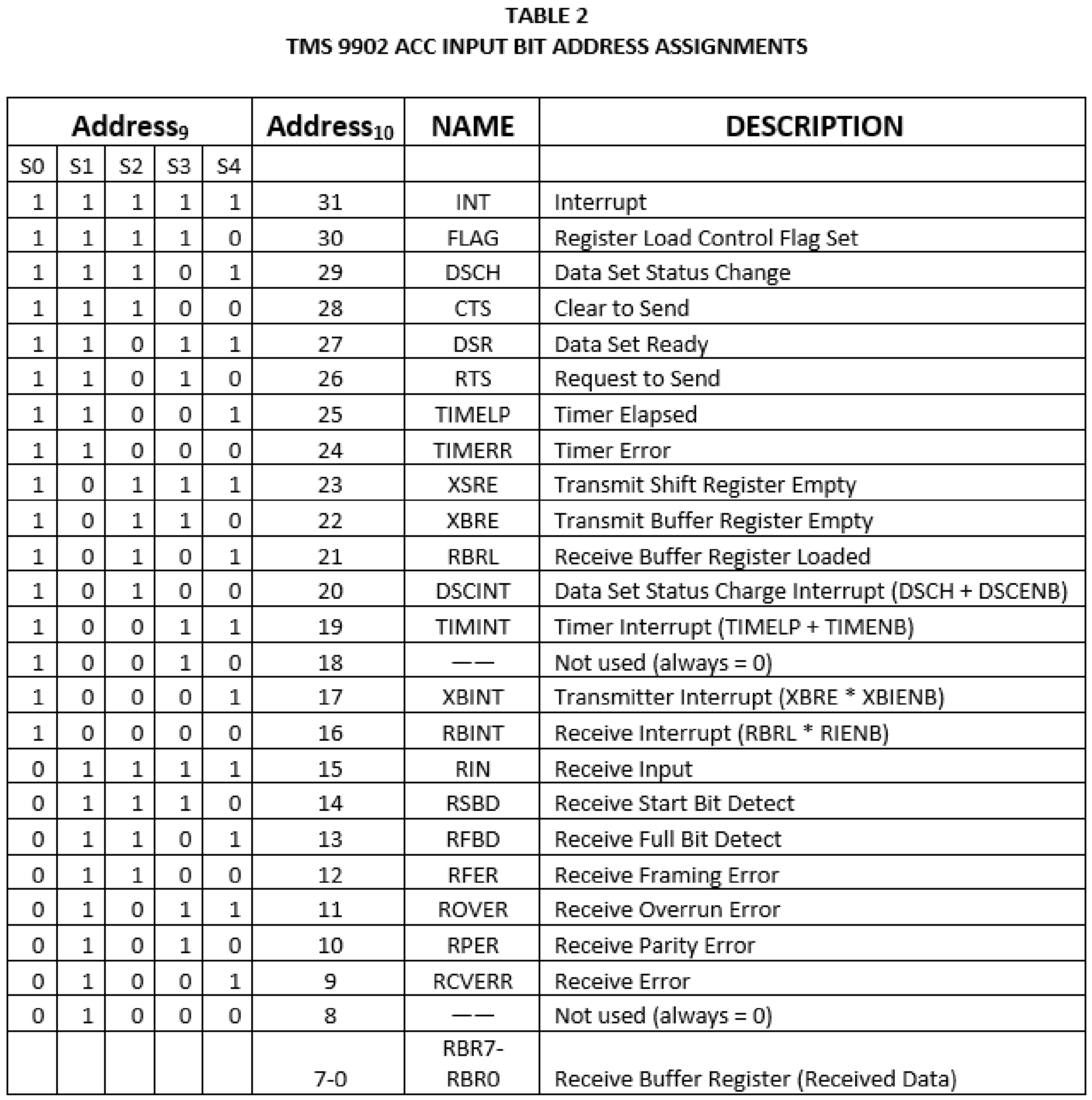
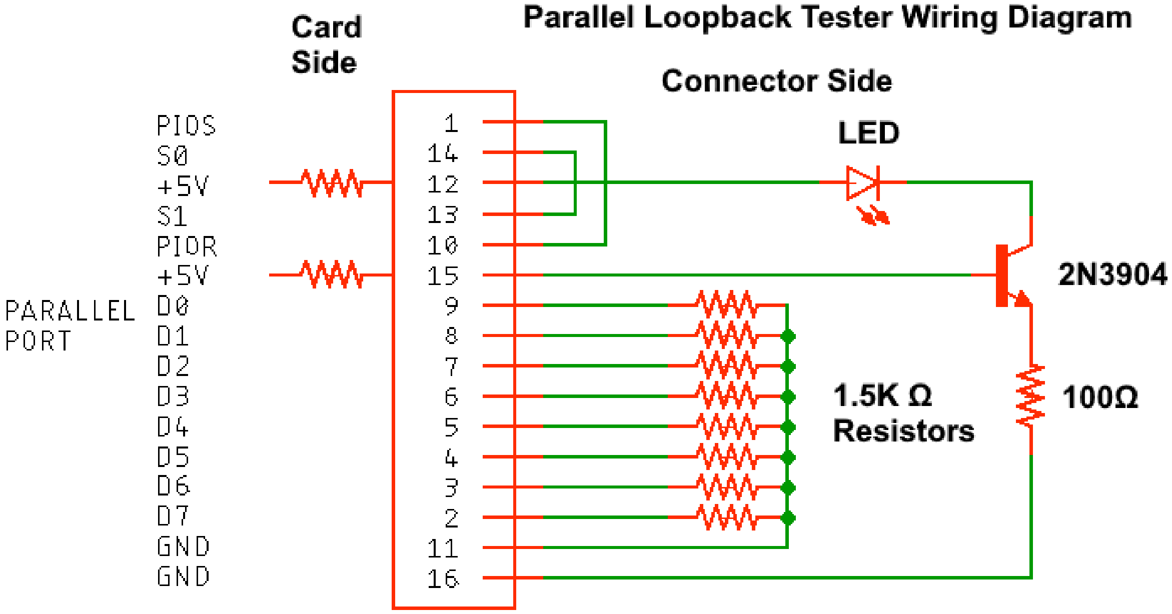


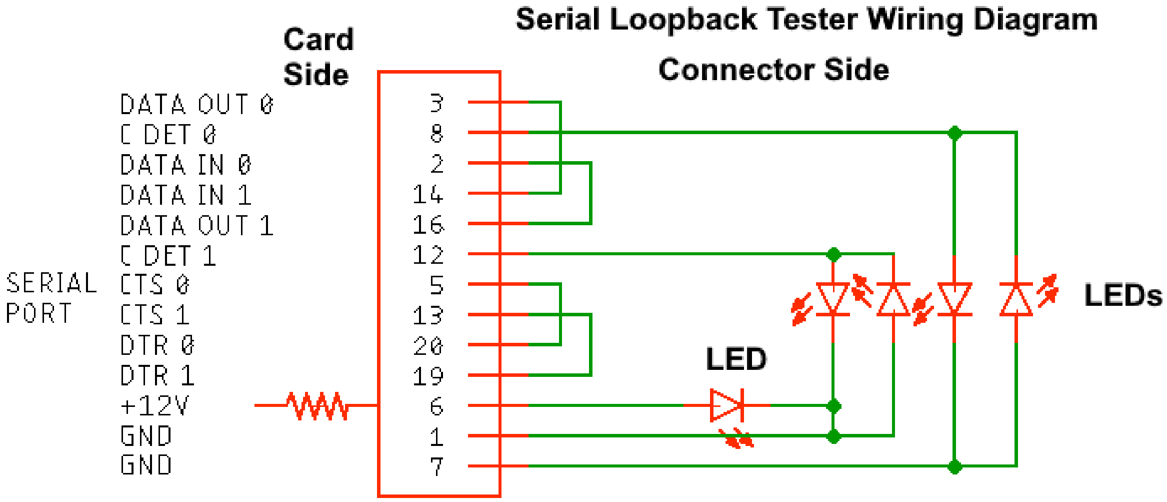
TABLE 2

TMS 9902 ACC INPUT BIT ADDRESS ASSIGNMENTS

Address$_9$					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 Charge 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	Receive 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)

Loopback Tester Schematics





TI-99/4x Software Test System

Software Includes:

- One Diskette which contains eighteen test programs written to test the TI-99/4A console and 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 & Execution

MINI MEMORY Command Module

1. Insert the 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 PROCEED {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" and press ENTER

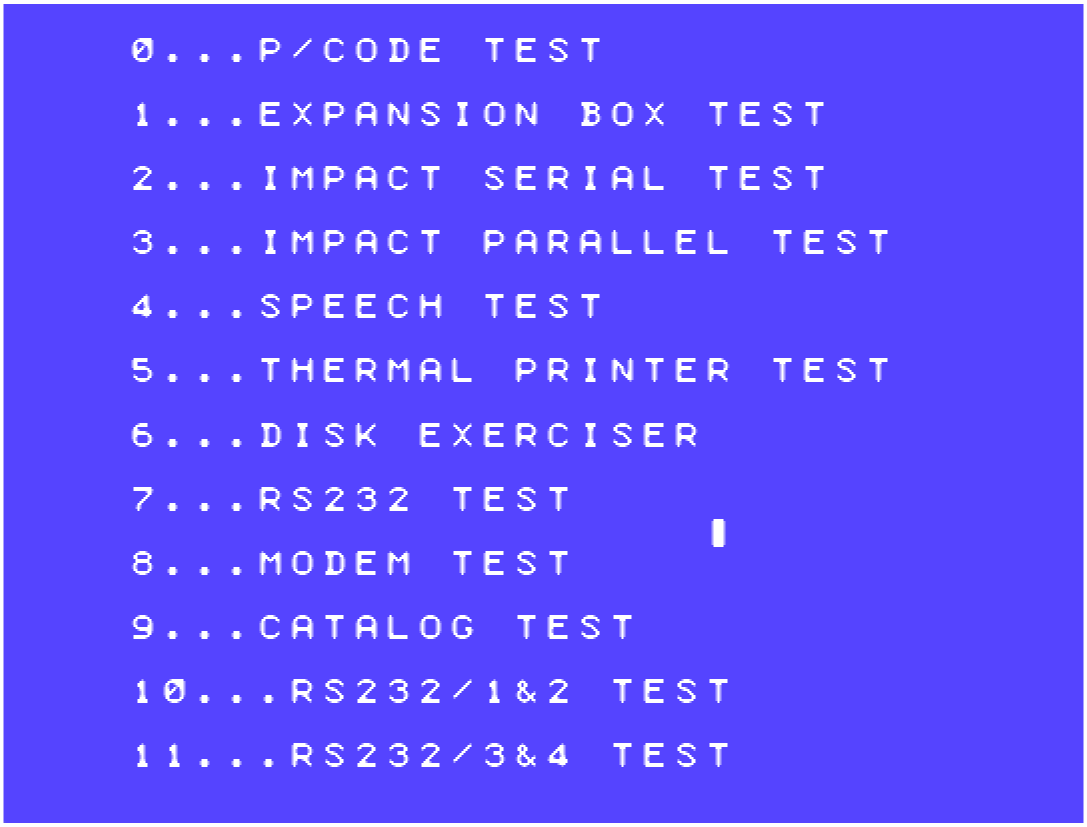
The program should be running, displaying a selection list from one of the test programs provided and waiting for a selection to be made.

When testing is complete turn unit off before removing the command module. This will allow the use of this program in other units or at another time by following the steps below:

1. Insert the MINI MEMORY command module
2. Turn unit on (Title screen is displayed)
3. Insert Floppy Disk/Disk Image
4. Press any key (Selection list is displayed)
5. Select #3 for MINI MEMORY (Selection list #2 is displayed)
6. Select #2 for RUN ("PROGRAM NAME" appears)
7. Press ENTER (Program begins)

TI EXTENDED BASIC Command Module

1. Insert the TI EXTENDED BASIC command module
2. Turn unit on (Title screen is displayed)
3. Insert Floppy Disk/Disk Image
4. Press any key (Selection list is displayed)
5. Press #2 for TI EXTENDED BASIC
    a. Disk drive will activate and load the “LOAD” program, after a short delay a selection list is displayed.



```
0...P/CODE TEST
1...EXPANSION BOX TEST
2...IMPACT SERIAL TEST
3...IMPACT PARALLEL TEST
4...SPEECH TEST
5...THERMAL PRINTER TEST
6...DISK EXERCISER
7...RS232 TEST
8...MODEM TEST
9...CATALOG TEST
10...RS232/1&2 TEST
11...RS232/3&4 TEST
```

Note: The load program is setup to load subprograms from DSK1. If you are using another drive number, you will need to either load the subprogram manually or modify the LOAD program to reference the correct drive number.

Note: ***Expansion Box Test*** program requires 32K expanded memory.

File Names

MINI MEMORY Diskette

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

Extended Basic Diskette

File Name	Test Programs
CATALOG	1. TI's catalog program from the disk controller manual
PC	2. P-Code Test
EXPBOX	1. Expansion Box Test
PRINTER#1	1. Impact Serial Printer Test
PRINTER#2	1. Impact Parallel Printer Test
SPEECH	1. Speech Synthesizer Test
TP	1. Thermal Printer Test
DE	1. Disk Exerciser
RS232	1. RS232 Test
MODEM	1. Modem Test
CATALOG	1. Disk Catalog Program
RO/1	1. RS232/1, RS232/2 & PIO Test Program
RO/3	1. RS232/3, RS232/4 & PIO/2 Test Program

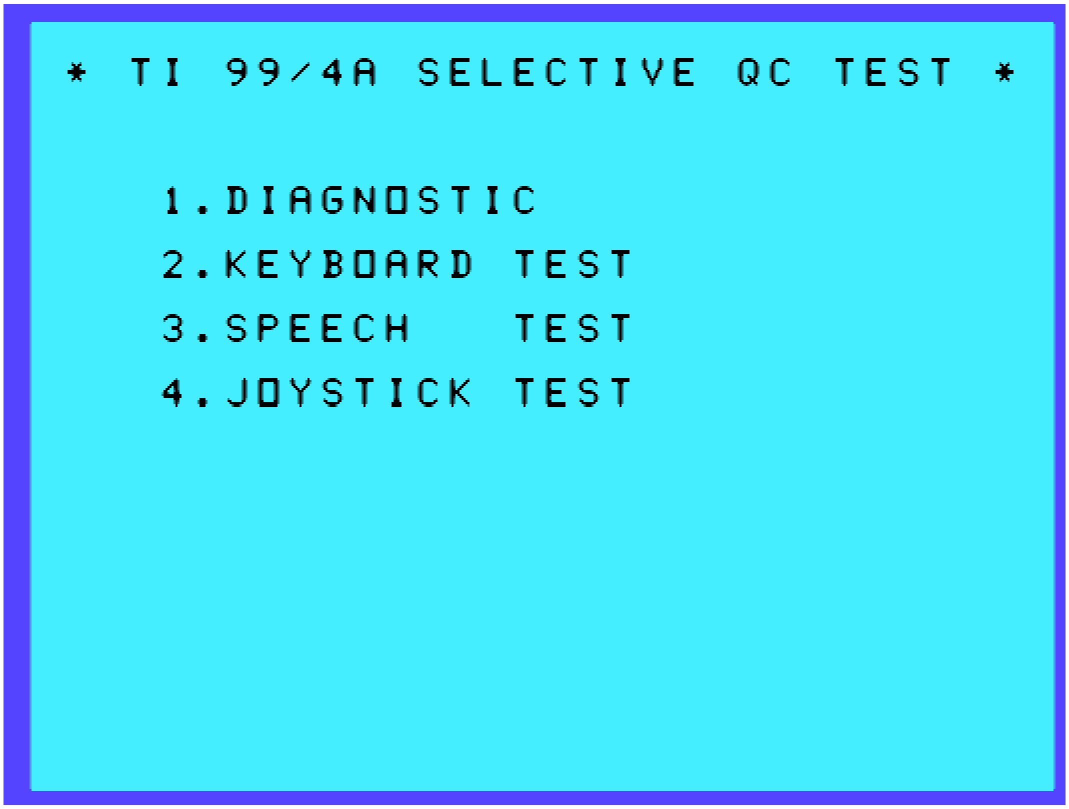
Note: *RO/1 and RO/2 will run a continuous loop. If there is an issue and error will be displayed otherwise only the main screen will show it is currently testing.*

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

DIAGNOSTIC

Filename: **DIAGNOSTIC** PROGRAM NAME: **RUN**



```
* TI 99/4A SELECTIVE QC TEST *

    1.DIAGNOSTIC
    2.KEYBOARD TEST
    3.SPEECH   TEST
    4.JOYSTICK TEST
```

The diagnostic test consists of seven individual programs linked together to form a complete console test excluding keyboard and input-output port.

Color Test

This test places all 16 colors on the screen for visual inspection and waits for a key entry from the operator before proceeding to the next test.

Sound Test

This test requires a mid-range volume setting. It consists of 5 distinct sound which exercise all 3 sound generators and noise generator.

Sprint-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 coincidence.

Character & Text Mode

This test displays in the text mode, 4 rows of 100 characters for approximately three seconds for visual inspection.

VDP Test (DYNAMIC RAMS)

ROM Test

GROM Test

6810s (Static RAMS)

The previous listed memory devices (VDP, ROM, GROM & 6810s) 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.

Keyboard Test

A. Follow prompts (PRESS SPACE BAR or !, 1, @ A, x, ETC.) displayed. When a 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 laced 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.

Speech Test

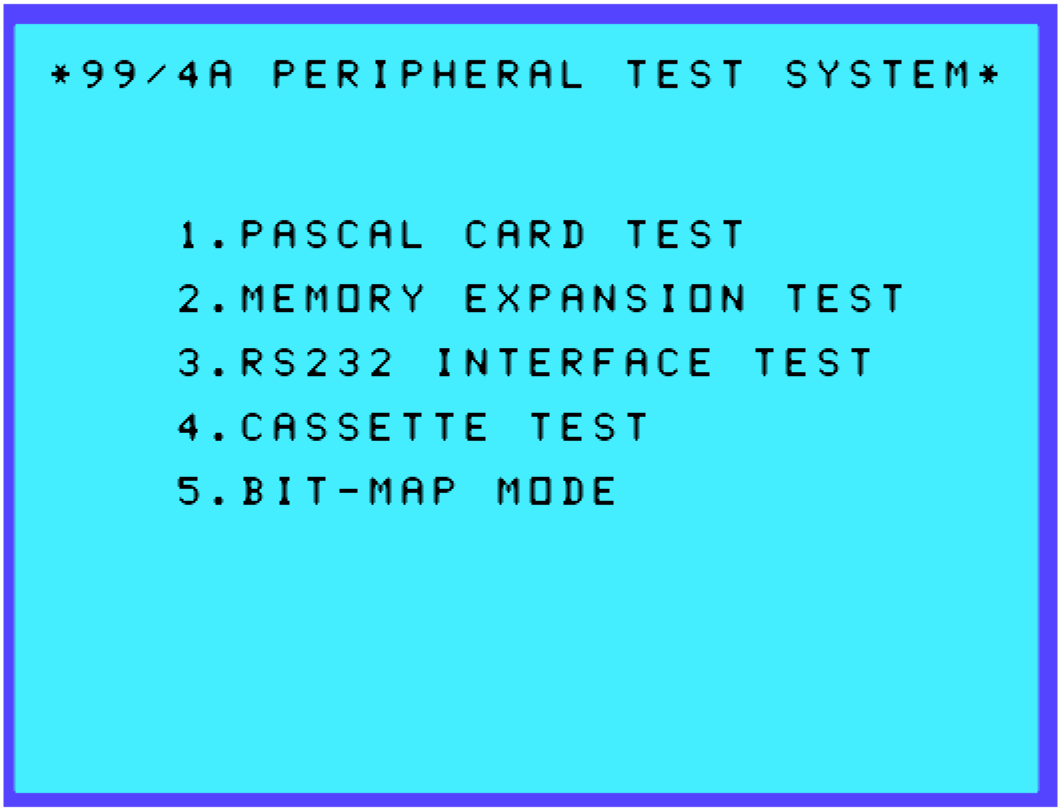
This test requires a mid-range volume setting. It says “READY TO START. DID YOU UNDERSTAND ME?” as the words are displayed. This test was written to test input-output circuitry of the console.

Joystick Test

This test displays the numbers 1 & 2 indicating 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 the 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**



```
*99/4A PERIPHERAL TEST SYSTEM*


      1.PASCAL CARD TEST
      2.MEMORY EXPANSION TEST
      3.RS232 INTERFACE TEST
      4.CASSETTE TEST
      5.BIT-MAP MODE
```

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.

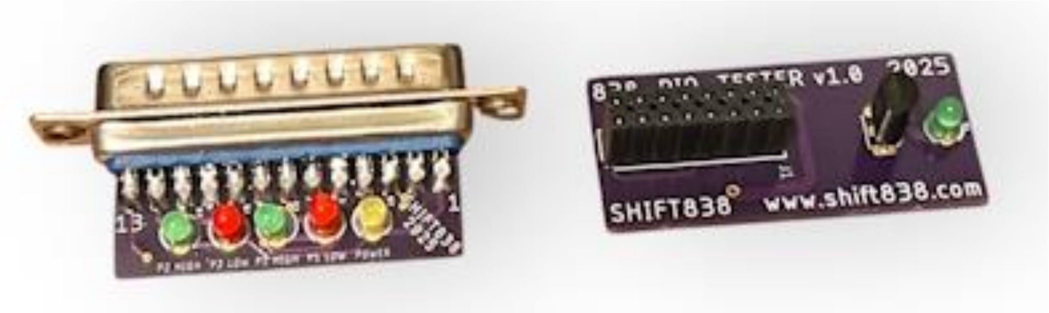
Memory Expansion Test

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.

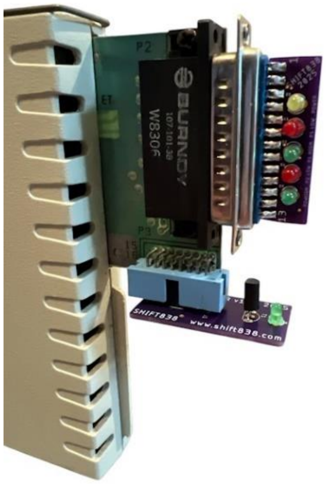
RS232 Interface Test

This requires a special serial and parallel loopback tester adapter/connector. Both adapters need to be plugged in at the same time for this test to work correctly.

Current testers are sold by SHIFT838 at https://www.shift838.com/store



Testers connected correctly:



This program tests the ROM and both the serial and parallel ports. If an error is detected, the program stops and a diagnostic error message is displayed.

Note: 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 graphics 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 13 for more information)

Serial Impact Printer Test

This program prints (using the serial port) the characters, sizing and type densities available through the software in the 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 the 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	Console / Monitor / Modulator
	No sound	Console
	Distorted Sprite Motion	Console
	Erratic Print	Console
	Won't RUN known good program	Console
	Won't SAVE program	Console
Keyboard Test	Won't Enter	Keyboard
	Intermittent Entry	Keyboard
	Multiple Entry	Keyboard
	Won't print large characters	Keyboard
Speech Test	No Speech	Speech Synthesizer / Console
	Garbled Speech	Speech Synthesizer / Console
	Static in speech	Speech Synthesizer / Console
Joystick Test	No Response	Joysticks / Console
	Won't Move Up	Joysticks / Console (***release ALPHA LOCK***)
	Both joysticks move simultaneously	Console
	No diagonal movement	Console

Pascal Card Test (***requires memory expansion***)	Card light comes on and computer locks up	Pascal Card
	Won’t run known good Pascal program	Pascal Card
	No light on card, computer powers up and displays title screen	Pascal Card
Memory Expansion	Computer displays “NO MEMORY EXPANSION”	Memory expansion card
	Won’t execute program from memory expansion	Memory expansion card
	Memory expansion card light comes on but computer locks up.	Memory expansion card
RS232 Interface	Card Light comes on and computer locks up	RS232 Interface
	Won’t print	RS232 Interface
Cassette Test	Won’t record	Cassette or Console
	Won’t read cassette	Cassette or Console
	Reads OK but won’t run program	Cassette or Console

Software Controlled Troubleshooting Techniques

One of the difficulties in troubleshooting microprocessor bases systems like the TI-99/4A, in which system control is handled by a 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 result will be a very long complex pulse train as the microprocessor responds to the instructions of firmware. The technician can see the 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 labeled as to use and procedure.

EASYBUG Note #1: Controlling the 9901

Required reference: CRU map of 9901

Use **EASYBUG** command for CRU single bit I/O. Referring to the 9901 CRU map, enter C and the address for the function to be exercises. 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 keyboard select (LSB)
0026	19	Programmable 3	22	BIT 1 of keyboard select
0028	20	Programmable 4	21	BIT 0 of keyboard select (MSB)
002A	21	Programmable 5	20	ALPHA LOCK Key
002C	22	Programmable 6	19	Cassette Motor 1
002E	23	PROG. 7 / INT 15	23	Cassette Motor 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=LL0 for 2us
M7001	(ENTER)	U504 Pin 12=LL0 for 2us

System Memory Map	
Hexadecimal Address	**Description**
0 – 1FFF	Console ROM Space
2000 – 3FFF	Memory Expansion
4000 – 5FFF	Peripheral Expansion (pre-decoded to I/O Connector)
6000 – 7FFF	Game Cartridge ROM/RAM (pre-decoded 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 modifying 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 XXXX 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-instructions 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 and predictable due to the repetitious nature of the program.

Some applications for this program follow:

1) Memory Selection Logic Example:

Test Memory Bock 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.

2) Data and Address Example

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

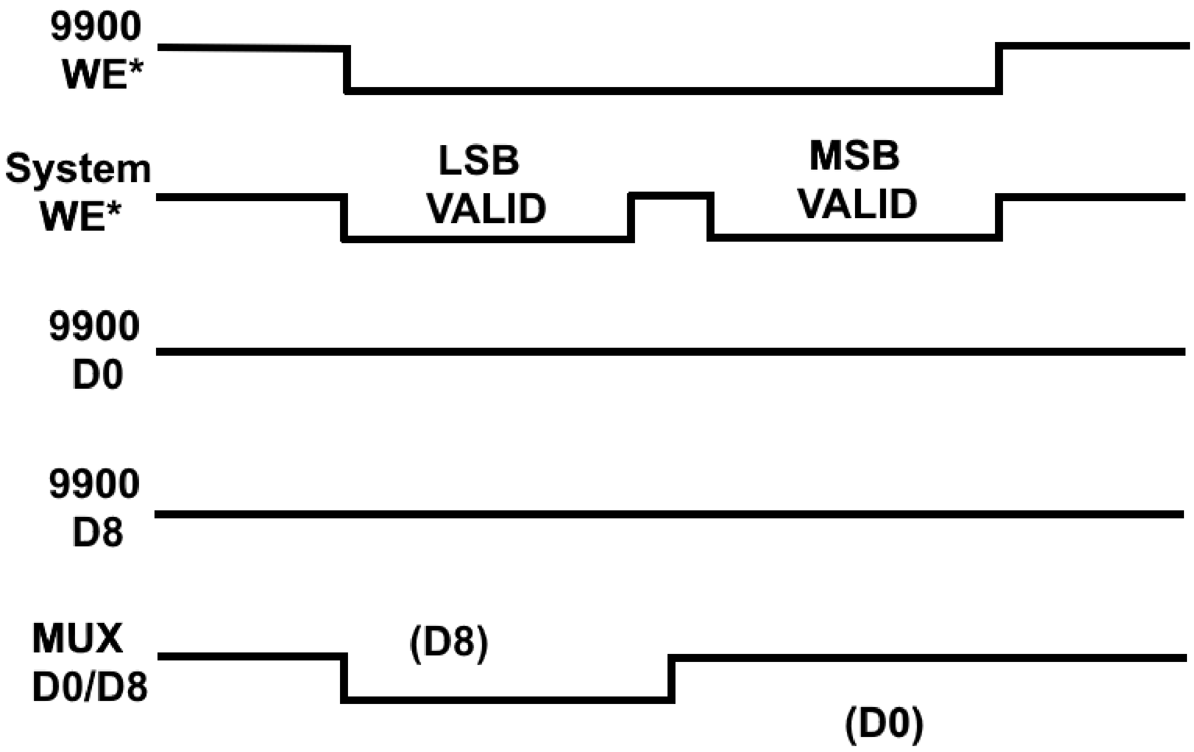
Data and address are valid when WE* is active (LOW).

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* (channel 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



```
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 7120 10FD          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
```

Links

- SHIFT838 Web Site: https://www.shift838.com
- SHIFT838 Web Store: https://www.shift838.com/store

Support

If you need support feel free to drop an email to support@shift838.com

Telnet to heatwave.ddns.net port 9640

Revision History

Date	Author	Notes
February 13, 2025	Chris Schneider	Original Release
February 14, 2025	Chris Schneider	Replaced RS232 CRU Table
April 13, 2025	Chris Schneider	Added notes and screenshots