CALIFORNIA COMPUTER SYSTEMS®

model XVI™

16K STATIC RAM MODULE

OWNER'S MANUAL

TABLE OF CONTENTS

FEATURES LIST

FULLY STATIC

USES POPULAR 2114 STATIC RAMs

+5 VOLT OPERATION ONLY

BANK SELECT AVAILABLE BY BANK PORT AND BANK BYTE

PHANTOM LINE CAPABILITY

ADDRESSABLE IN 4K BLOCKS

4K BLOCKS CAN BE ADDRESSED ANYWHERE WITHIN 64K in 4K INCREMENTS

CAN BE BUILT UP 4K AT A TIME

FULLY BUFFERED

MEETS IEEE PROPOSED S-100 SIGNAL STANDARDS

LED INDICATORS FOR BOARD SELECTION AND BANK SELECTION

FR-4 EPOXY PC BOARDS

SOLDER MASKED ON BOTH SIDES

SILK SCREEN OF PART NUMBER AND PART DESIGNATOR

AVAILABLE IN KIT, ASSEMBLED AND TESTED, OR BARE BOARD

THEORY OF OPERATION

GENERAL

The M16 is a 16k x 8 bit memory board for use on S100 bus systems. The memory is broken into four groups which are independently addressable. Bank select capability is available which is compatible with AM100 and Cromemco systems. The bank select is flexible and may be used in systems other than these two. Note, this board is not compatible with address select systems such as IMSAI. If bank select is not desired it may be disabled. Inserting a jumper allows the M16 to be used in systems using a phantom signal. The M16 is fast enough for present S100 systems. If on future systems the processor were faster than the memory, a wait state can be enabled.

MEMORY

The memory chips are 2114 type static RAMs. To get an 8 bit word structure the 1k x 4 RAMs are arranged in columns of two. Four columns (or pairs) of RAMs form a memory group. The memory board has four memory groups, A thru D. The data in (DI) and data out (DO) system bus lines are buffered into a common internal data bus by U57 thru U59.

MEMORY ADDRESSING

Address lines A0 thru A16 enter the board and are inverted by buffers U54 thru U56 and become -IA0 thru -IA16. The RAM chip address lines are all paralled and are determined by lines -IA0 thru -IA9. Lines -IA12 thru -IA15 are paralleled into 4 sets of open collector exclusive OR gates, U2 thru U5. The other input of these two input gates is connected to the memory address switches. For example, suppose we set memory group A's switches all closed. This put a 0 on one input of each of the gates. If address lines A9 thru A15 are 0's, then the inverters will give 1's at the other gate inputs. The gate outputs are 1's telling the priority encoder U7 that group A has been addressed. Since the open collector gate outputs for each group are tied together, any wrong address would not leave the output line high. Suppose group A and B switches were both set for the same address. Since group A has the highest priority, only group A would be selected for that address. Buffered address bits -IA10 and -IA11 and the priority encoder outputs are decoded by U8 and U9 to enable the memory column chip selects. This completely determines each memory data location.

MEMORY READ/WRITE

To write into memory, the -WRITE signal to the memory chips must be low and the -IN ENABLE signal to the data buffers must be low. If either MWRITE (memory write) is high or $\overline{\text{PWR}}$ (not processor write) is low then -IN ENABLE goes low activating the data buffers to let bus data onto the board. The -WRITE signal is a little more involved. Line I/O, which indicates an I/O signal, is SOUT OR SINP. $\overline{\text{SWO}}$ (not write in I/O or memory) and I/O must be low. If SMEMR (status memory write) is high or both I/O and $\overline{\text{SWO}}$ are low then pin 3 of U60 is high and its output is low (if bank activate is also high). This signal enables the priority encoder. It is also ORed with -IN ENABLE to form the -WRITE line to the memory chips. Reading the memory also uses -WRITE to set the memory chips and -OUT ENABLE for the data buffers. +OUT ENABLE is just -IN ENABLE AND MEMORY BOARD ACTIVATE AND PDBIN (data expected on bus). When these 3 are all high the data buffer lets the processor read the memory.

PHANTOM - RAM READ DISABLE

Users with a PHANTOM line should put in jumper W3. The PHANTOM is a RAM read disable signal. When the PHANTOM line is high the board operates normally. When this line is low the MEMORY BOARD ACTIVATE line goes low preventing memory output from reaching the system. In this manner RAM memory is disabled so a ROM can be used for startup. If the jumper is not used the line is held high by a pull up resistor and the board operates normally.

LEDS

When the memory address has selected one of the four memory groups, the BOARD SELECT LED is lit. Under this condition MEMORY BOARD ACTIVATE is high setting the output of the U11 driver gate low allowing current to flow thru the LED. When the bank address for this board has been selected the BANK SELECT LED is lit. With the BANK ACTIVATE signal high the U11 driver gate goes low allowing current to flow thru the LED.

BANK SELECTION

The bank selection of the M16 is very flexible. The bank byte, bank address, and other switches allow operation with Cromemco, AM100, and other port bank select systems. Note, this board is not compatible with address select systems such as IMSAI. The bank port address gates U15 and U16 compare address lines -IA0 thru -IA9 with the switch settings of S7. If the address is correct, the common collector outputs remain high. If I/O signal and IPWR (write into memory or I/O) are also high then the bank byte registers U12 and U13 are clocked. This clocks in the data bus D0 thru D7. The register outputs go thru switches S5 into gate U10. If a low register output goes thru a closed switch to gate U10 the output (bank activate) goes high. Bank activate high will make LED driver U11 go low lighting the BANK SELECT LED. It also enables the priority encoder thru gate U60. Because the bank byte register is composed of flipflops, the outputs remain the same until the register is clocked again or cleared. $\overline{\text{POC}}$ (not power on clear) always resets the registers. If desired either $\overline{\text{EXT CLR}}$ or $\overline{\text{PRESET}}$ may be jumpered in as a clear signal. Bank byte bit 0 may be used in two ways. If switch BD is open and the register is cleared, the bit 0 register output is inverted by U11 and sets bank activate if the S5-0 switch is closed. This allows a bank to be selected just by the $\overline{\text{POE}}$ or other reset pulse. For clocked data the 0 bit acts like the others regardless of switch position. Bit 0 also drives the DI0 line if bank address, I/O, and IPDBIN signals are high. This is used as an acknowledge signal that an active memory bank has been enabled.

WAIT STATE

If the memory should be slower than the processor, a wait state may be added. If the wait state switch S6-WA is open the system operates normally with no wait state delay. If the switch is closed, the PRDY line is now controlled by the PSYNC signal introducing a delay of one major clock cycle.

ADDRESSING INSTRUCTIONS

MEMORY ADDRESSING

The memory is addressed in 4 separate groups of 4096 bytes each. The switches for each group are:

GROUP A = S4
GROUP B = S3
GROUP C = S2
GROUP D = S1

Starting Addr. Hex	Octal	A15	A14	A13	A12	Binary Number
0000	000:000	closed	closed	closed	closed	0000
1000	020:000	closed	closed	closed	open	0001
2000	040:000	closed	closed	open	closed	0010
3000	060:000	closed	closed	open	open	0011
4000	100:000	closed	open	closed	closed	0100
5000	120:000	closed	open	closed	open	0101
6000	140:000	closed	open	open	closed	0110
7000	160:000	closed	open	open	open	0111
8000	200:000	open	closed	closed	closed	1000
9000	220:000	open	closed	closed	open	1001
A000	240:000	open	closed	open	closed	1010
B000	260:000	open	closed	open	open	1011
C000	300:000	open	open	closed	closed	1100
D000	320:000	open	open	closed	open	1101
E000	340:000	open	open	open	closed	1110
F000	360:000	open	open	open	open	1111

closed = 0
open = 1

BANK PORT SELECT

The bank port address is determined by switch S7. A few combinations are shown, including hex 40 used in Cromemco systems.

Start Adr. Hex	Octal	A7	A6	A5	A4	A3	A2	A1	A0
00	000	0	0	0	0	0	0	0	0
01	001	0	0	0	0	0	0	0	1
02	002	0	0	0	0	0	0	1	0
40	100	0	1	0	0	0	0	0	0
FF	377	1	1	1	1	1	1	1	1

closed = 0
open = 1

BANK BYTE SELECT

The bank byte select is determined by switch S5. The bank activate signal is enabled if any high data bit (D0 thru D7) has its corresponding switch closed (S5-0 thru S5-7). A few combinations are shown.

Data Bit	A7	A6	A5	A4	A3	A2	A1	A0
0 only	0	0	0	0	0	0	0	1
1 only	0	0	0	0	0	0	1	0
2 or 3	0	0	0	0	1	1	0	0
4, 5, or 7	1	0	1	1	0	0	0	0

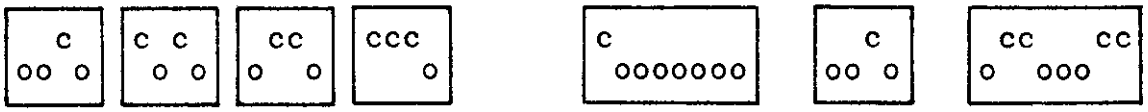
closed = 1
open = 0

BANK SELECT DISABLE

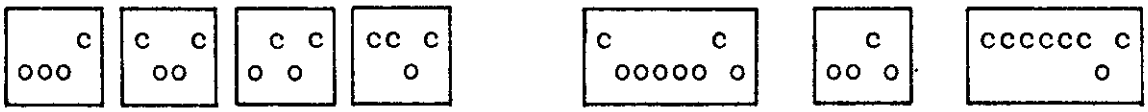
The bank select feature may be disabled by closing switch S6-BD.

EXAMPLES OF TYPICAL ADDRESSING

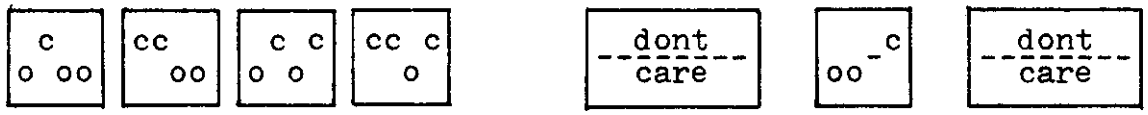
The diagram below shows switch settings for a typical AM100 system. Assume the port address is hex 39 and the port comes up active on startup. Let the memory location be 32k to 48k. Note o=open and c=closed on diagram below.



The next diagram (below) shows switch settings for a typical Cromemco system. The port address must be hex 40. Let byte bits 0 or 6 activate the bank and bit 0 be active on startup. Let the memory location be 16k to 32k.



In the next example, assume bank select is not needed and the memory address is split between 16k to 24k and 48k to 56k.

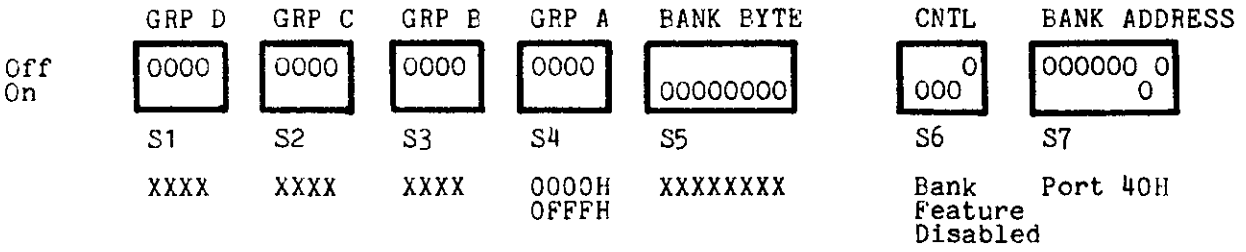


Again note that in these diagrams o=open and c=closed.

FRONT PANEL QUICK CHECKOUT

If the computer does not have a front panel, skip this subsection.

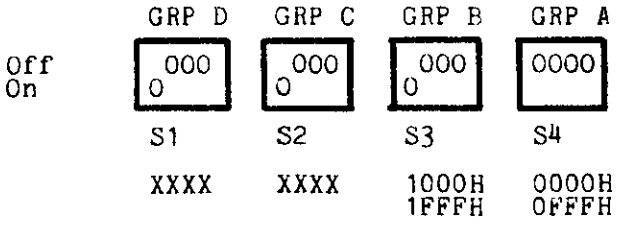
Before powering on the computer, the memory board switches should be set up as follows:



The priority feature will allow only Group A to be selected.

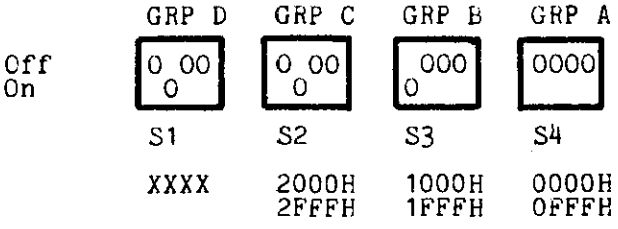
Set the Front Panel Address Switches A0 thru A15 to the off position (0000H). Examine that address. Set the Data Switches D1 thru D7 to the off position, and D0 to the on position (01H). Deposit (write) into memory and compare the Data readout with the switch setting. Set D1 on and D0, D2 thru D7 off. Deposit (write) into memory again and compare the result. Continue the pattern of one data bit on (high) and the rest off (low) until all data bits have been checked. If the data does not compare, isolate the malfunction with a logic probe or voltmeter before continuing.

After Group A has been checked, power down the computer and set the switches as follows to check Group B:



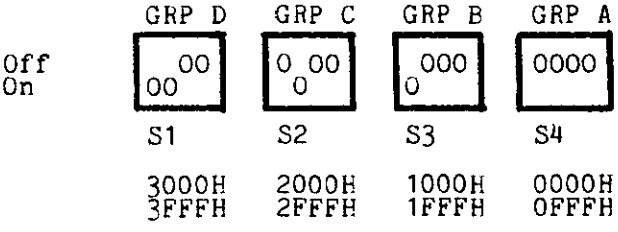
Examine 1000H (A12 on, the rest off), and deposit the same pattern as was done with Group A. Isolate and correct any malfunctions as they become appearent.

To check Group C, power down the computer and set the switches as follows:



Examine 2000H (A13 on, the rest off), and test as done previously.

To check Group D, power down the computer and set the switches as follows:



Examine 3000H (A12 and A13 on the rest off), and test as done previously. Power down the computer and reinstall any boards required to run as a system.

DIAGNOSTIC TEST OVERVIEW:

These memory diagnostics run an 8080 or Z80 systems and provide practical test of a memory board. Two diagnostics are provided: a walking bit test and a burn-in test. The routines have been written so that they do not require RAM other than the system stack and the RAM under test. Likewise, the routines may be executed from either RAM or ROM.

Diagnostics in general can be divided into three classes: fault detection, fault isolation, and fault correction. These routines preform the fault detection, and provide sufficient data to preform the fault isolation. After a fault is isolated, correction becomes practical matter.

Detected errors are displayed on the console device when they are detected. Two formats are used - the first, used by the burn-in test and the first stage of the walking bit test, show errors in the form:

xx yyyy zz

where: xx = the bad data
yyyy = the address where the bad data occured
zz = what the data should have been

Each character is a hexadecimal digit.

The last part of the walking bit test logs its errors in the form:

wwww xx yyyy zz

where: wwww = the address where the error was found
xx = the bad data
yyyy = the address where data was last written
zz = the last written data

Each character is a hexidecimal digit.

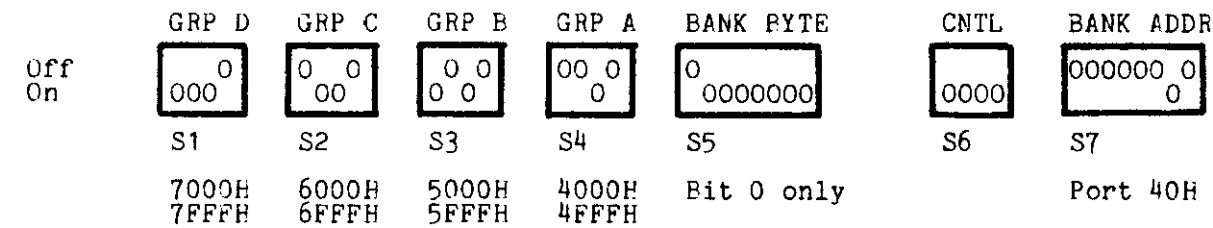
This print out allows a logical deductive process to isolate the problem.

LOADING THE DIAGNOSTIC:

No special precautions are necessary. The user should use their standard method to load the routines.

SETTING UP FOR THE TEST:

Before powering on the computer, the memory board switches should be set up as follows:



Note: The setting of S6 provides a partial test of the bank byte.

The bank reset strap is immaterial, as is the phantom strap. It is recommended strapping the bank reset in the RST (W2) position.

INSTALLATION:

The memory test routines are capable of standing alone, except for the system unique input/output drivers. The drivers must be provided by the user. Three routines are needed:

CONIN: Console input. Reads one ASCII character from the console keyboard and sets the parity bit (bit 7) equal to 0. The character is returned in the accumulator (A register).

CONOUT: Console output. Writes one ASCII character to the console display device. The character to be output is passed to CONOUT in the C register. If the console output device is sensitive to bit 7, then the user must set/reset bit 7 to what is needed in the CONOUT routine.

CONST: Console status: This routine reads the console input status. If data is not available, then the accumulator is set to 0 and the status flags must match. If data is pending, then a -1 (0FFH) should be returned in the accumulator (A Register). The status flags must show at least a non-zero condition on return.

After these routines have been prepared, they must be loaded into memory. To allow the diagnostics to find them, three jump instructions are located at the front of the diagnostic (locations 0103H for CONIN, 0106H for CONOUT, and 0109H for CONST). The user should put the addresses of his I/O routines into these locations. See lines 0122, 0124, and 0126 in the assembly listings.

Examples of the routines which worked on the development system are shown at lines 9000 to 9040 of the listing. These used a MITS PIO board with status port address of 04H and data port address of 05H for input from an Electrol ASCII keyboard. For output, a SSM VB1A was used with a custom driver located in ROM at location 0FE00. This driver expected data in the B register. Notice that none of these routines alter any register except for the A register.

At this point, the board is ready to put into the computer. Insure no other memory will respond to addresses in the range 4000H - 0BFFFH.

RUNNING THE DIAGNOSTIC:

Load the diagnostic into your system at location 0100H. The diagnostic is small enough to fit into less than 1K of memory, starting at 0000H. It was assembled assuming a 16K block of memory starting at 0000H; the only change needed if less memory is available is to change the STACK location. It is currently initialized to 3F76H; a good alternate location would be 0100H.

Transfer control of the computer to location 0100H. The computer will type out:

DIAGNOSTIC:

You can now select which diagnostic you want. Current options are "C" for continuous burn-in or "W" for walking bit test. Any other selection will Cause ???? to print out and DIAGNOSTIC: will again appear. For initial test, push W. The computer will respond:

DIAGNOSTIC: WALKING BIT TEST
BLOCK SIZE:

Select a small block size initially. This way, the read/write circuitry can be checked out without a flood of error or printouts. A block size of 2 is suggested. To terminate entry, type in a space, a comma, or carriage return. If the wrong number is typed in, continue typing in until the last four digits are correct.

The computer will now ask for:

BASE ADDRESS:

Type in the desired base address.

NOTE: The base address must be a multiple of 1024 (400H). For the board setup suggested, a base address of 4000H is indicated.

At this time, the diagnostic will do its test. On completion, it will type out:

```
TEST DONE
DIAGNOSTIC:
```

It is now ready for the next test. If errors were logged, see the trouble shooting subsection and correct the malfunction. Rerun the diagnostic until an error free run is achievied.

Rerun the Walking Bit Test with a block size of 1K (400H) and a base address of 16,384D (4000H). This tests U30 and U47.

Increase the base address by 1K (400H) increments leaving the block size at 1K (400H). This tests, in the order specified:

Base Address	Devices under test		
4000H	U30	U47	Group A
4400H	U31	U48	"
4800H	U32	U49	"
4C00H	U33	U50	"
5000H	U26	U43	Group B
5400H	U27	U44	"
5800H	U28	U45	"
5C00H	U29	U46	"
6000H	U22	U39	Group C
6400H	U23	U40	"
6800H	U24	U41	"
6C00H	U25	U42	"
7000H	U18	U35	Group D
7400H	U19	U36	"
7800H	U20	U37	"
7C00H	U21	U38	"

If errors are logged, replace the appropriate chip. The above table narrows the problem to 2 chips. The exact chip is indicated by the bad data: If the data is bad in the upper half of the byte, replace the lower numbered chip (physically higher on the board). If the bad data is in the lower half byte, replace the higher numbered chip.

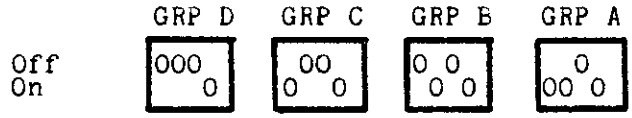
EXAMPLE PRINTOUT:

```
5C02          84          5C02 04

Group B Chips U29,46      Chip U29 bad
```

After a good run for all 16, 1K increments run the walking bit with a block size of 16K (4000H).

At this time, invert the settings of S1, S2, S3, and S4 and run a 16K block starting at 8000H. This tests the group select circuitry completely.

The inverted switch settings are as follows:



S1	S2	S3	S4
8000H	9000H	0A000H	0B000H
8FFFH	9FFFH	0AFFFH	0BFFFH

Primary chips tested here are U2, U3, U4, U5, and U6.

After passing all walking bit tests, type in C for CONTINUOUS BURNIN test.

Specify a block size of 4000H and the appropriate base address (8000H if the above procedure is followed). Leave it run for 1 to 2 hours to shake out the weak links (infant mortality). To terminate this test, type in Control C. Errors, if any, will be printed out as they occur, as well as a sum total on test completion.

TROUBLESHOOTING GUIDE:

Several problems that tend to be most prevalent:

1. Solder bridges - Inspect solder work carefully. If a suspicious pair or joints is noticed, test them with an ohmmeter.
NOTE: Some pins of support chips are purposefully connected together. If you find a short, check the logic print to see if it is correct.

2. Cold or rosinous solder joints - These cause unpredictable problems. They are characterized by a grainy, non-shiny appearance. They may have dark specks embedded in the solder. Reheat any suspicious joints.

3. Curled IC pins - This problem occurs from misalignment of the pins when inserting the ICs into the sockets. The diagnostic detects this as an open condition. To correct it, remove the chip from the socket and carefully straighten the pin(s).

All of these problems should be detectable by careful inspection. After the board is inspected, have someone else inspect it. Correct any problems when found.

Another, less common problem - unsoldered joints. There are over 1,100 solder joints on a 16K memory board. It is easy to skip 1 or 2 joints. Watch carefully for them.

ERROR PRINTOUT INTERPRETATION:

NOTE: Errors may show up with many symptoms. The following guidelines show typical symptoms and probable cause. The best method of isolating a problem (and fixing it at the same time) is to pull out a suspect component and replace it with a known good part. Then re-run the diagnostic and see if the problem is still present.

If a problem persists after replacing all suspect parts, set up a controlled test condition and trouble shoot the problem with a logic probe or a voltmeter, using the logic print to identify test points.

	ERROR CONDITION	PROBABLE CAUSE	SUSPECT PARTS
1.)	Bad data = OFFH all groups.	a) Bank select b) Board select	U10, U11 U59, U7, U6, U60 U57, U58, U14
2.)	Random data or all 0 data, all groups	a) Bad write control	U61, U60, U56, U57
3.)	OFFH data one group only	a) Group select A " B " C " D	U5, U6, U7, U9 U4, U6, U7, U9 U3, U6, U7, U8 U2, U6, U7, U8
4.)	One address line hung Printout: good data bad address	a) Address buffers	A0,1,4,5 (U53) A2,3,6,7,12,15 (U54) A8 - 11,13,14 (U55)
5.)	One data line hung a.) Hung "0" (good address, bad data = 0	a) Grounded data line	U56, U57, maybe U58
	b.) Hung "1" (non-zero data, all addresses)	a) Open data line b) Data line shorted to +5V	U56, U57, U58 Same plus memory chips

6.)	Soft errors (random addresses and data, non-repeatable)	c) Memory chip access time b) Heat sensitive parts	Try closing S6 (WA) and return tests Treat as a hard error and replace suspect parts
7.)	Hard memory errors	Bad memory chip	See earlier table

SAMPLE MEMORY DIAGNOSTIC RUN

```
DIAGNOSTIC: WALKING BIT TEST              Typed in "W"
BLOCK SIZE:  30                           Note: Base address must be
BASE ADDRESS: 300                         multiple of 1K (400 HEX)
BAD BASE ADDRESS:                         Block size may be any size
BASE ADDRESS: 500
BAD BASE ADDRESS:
BASE ADDRESS: 600
BAD BASE ADDRESS:
BASE ADDRESS: 700
BAD BASE ADDRESS:
BASE ADDRESS: 400
TEST DONE
DIAGNOSTIC: WALKING BIT TEST              New Test
BLOCK SIZE: 400                           Equal block size, base address
TEST DONE
DIAGNOSTIC: WALKING BIT TEST              Larger block size test
BLOCK SIZE: 1000
BASE ADDRESS: 400
TEST DONE
DIAGNOSTIC: WALKING BIT TEST
BLOCK SIZE: 1800
BASE ADDRESS: 400
TEST DONE
DIAGNOSTIC: ????                          Typed in "1"
DIAGNOSTIC: WALKING BIT TEST              Odd block size test
BLOCK SIZE: 579
BASE ADDRESS: 400
TEST DONE
DIAGNOSTIC: CONTINUOUS BURNIN             Different Test
BLOCK SIZE: 3765                          No parameter restrictions
BASE ADDRESS: 3D3                         Up to 0FFH (255D) errors shown
00 ERRORS
TEST DONE
DIAGNOSTIC: ????
DIAGNOSTIC: CONTINUOUS BURNIN
BLOCK SIZE: 3ABC
BASE ADDRESS: 3EF
00 ERRORS
TEST DONE
DIAGNOSTIC:
```

```
 1 0000                          TITLE   '2114 MEMORY DIAGNOSTIC VER 1.1'
 2 0000              ;
 3 0000              ;
 4 0000              ; Console input/output support routines
 5 0000              ;
 6 0000              ; These routines are a highly matured, well thought
 7 0000              ; out set based on Intel's monitor.  They provide a
 8 0000              ; significant capabillity to converse with an 8080,
 9 0000              ; 8085, or Z-80 based microprocessor system.  The
10 0000              ; only registers altered are the accumulator and
11 0000              ; the pass register carrying active parameters upon
12 0000              ; entry to routine.  The stack is used extensively;
13 0000              ; sufficient stack space must be provided by the
14 0000              ; calling programs.  The stack pointer is returned
15 0000              ; pointing to same place on exit unless an error
16 0000              ; was detected (SP=?) or parameters are returned on
17 0000              ; the stack.  In the last case, the stack is offset
18 0000              ; by 2 times the requested number of parameters and
19 0000              ; will be set right after popping these parameters
20 0000              ; off the stack.
21 0000              ;
22 0000              ; Register usage conforms to ICOM and CP/M defined
23 0000              ; conventions:  Output data is passed in the C
24 0000              ; register and input data is expected in the A
25 0000              ; register.  These routines require CP/M compatable
26 0000              ; CONIN and CONOUT routines as contained in the
27 0000              ; user`s BIOS program; or CI and CO as in the ICOM
28 0000              ; Resident PROM.
29 0000              ;
30 0000   000A       LF      EQU     0AH     ; ASCII line feed
31 0000   000D       CR      EQU     0DH     ; ASCII carriage return
32 0000   0040       CNTL    EQU     40H     ; ASCII Cntl offset
33 0000   0040       STACK   EQU     40H
34 0000              ;
35 0000              ;
36 0000              ;
37 0000   0040               ORG     40H
38 0040              ;
39 0040 C38F03               JMP     INIT
40 0043              ;
41 0043   0100               ORG     0100H
42 0100              ;
43 0100              ; SYSTEM LINKAGES
44 0100              ;
45 0100   C003       CONIN   EQU     0C003H
46 0100   C006       CONOUT  EQU     0C006H
47 0100   C373       CONST   EQU     0C373H
48 0100   C000       USER    EQU     0C000H
49 0100              ;
50 0100 C38F03               JMP     INIT
51 0103 C303C0       CONI:   JMP     CONIN
52 0106 C306C0       CONO:   JMP     CONOUT
53 0109 C373C3       CST:    JMP     CONST
54 010C C300C0       ERR:    JMP     USER
55 010F              ;
```

```
                        ; Routine BLK prints one blank on the current
                        ; console device.
                        ;
                        ; Entry parameters:     None
                        ; Return parameters:    None
                        ; Stack usage:          4 bytes
                        ;
010F C5         BLK:    PUSH    B       ; Save (BC)
0110 0E20               MVI     C,' '   ; Get an ASCII space
0112 C34901             JMP     ECH2    ; Go output it
0115                    ;
                        ; Routine CONV converts a 4 bit binary number to
                        ; its ASCII equivelant.  The high order 4
                        ; accumulator bits are lost.
                        ;
                        ; Entry parameter:      4 bit binary number in
                        ;                       lower half of accumulator
                        ; Exit parameter:       ASCII character in (A)
                        ; Stack usage:          0 bytes
                        ;
0115 E60F       CONV:   ANI     0FH     ; Clear high bits
0117 C690               ADI     90H     ; Insert partial ASCII
0119 27                 DAA             ; Zone
011A CE40               ACI     40H     ; Insert rest of ASCII
011C 27                 DAA             ; Zone
011D C9                 RET
011E                    ;
                        ; Routine CRLF prints an ASCII carriage return and
                        ; line feed (in that order) on the console.  It
                        ; follows these with 4 blanks to create a lefthand
                        ; margin.
                        ;
                        ; Entry parameter:      None
                        ; Exit parameter:       None
                        ; Stack Usage:          8 bytes
                        ;
011E E5         CRLF:   PUSH    H       ; Save (H,L)
011F 212701             LXI     H,CRMSG ; Get message address
0122 CDAE01             CALL    PRTWA   ; Print message
0125 E1                 POP     H       ; Restore (HL)
0126 C9                 RET
0127                    ;
0127 0D0A20A0   CRMSG:  DB      CR,LF,' ',' '+80H
012B                    ;
                        ; Routine DEPRT prints the contents of the (DE)
                        ; register pair as a 4 digit hexadecimal number on
                        ; the console.
                        ;
                        ; Entry parameter:      (DE) = 4 digit hex number
                        ;                       to be printed on console.
                        ; Exit parameter:       None
                        ; Stack usage:          10 bytes
                        ;
012B CD1E01     DEPRT:  CALL    CRLF    ; Print a CR, LF
012E                    ; Alternate entry point if no CR, LF wanted
```

```
 111 012E 7A        DEPRA:  MOV     A,D     ; Get high order byte
 112 012F CD3301            CALL    HEX2    ; Print 2 numbers
 113 0132 7B                MOV     A,E     ; Get low order byte
 114 0133           ; Alternate entry point to print (A) as two hex
 115 0133           ; digits
 116 0133 F5        HEX2:   PUSH    PSW     ; Save low order byte
 117 0134 0F                RRC             ; Move high order nibble
 118 0135 0F                RRC             ; to lower half of (A)
 119 0136 0F                RRC
 120 0137 0F                RRC
 121 0138 CD3C01            CALL    HEX1    ; Print the nibble
 122 013B F1                POP     PSW     ; Get low nibble back
 123 013C           ; Alternate entry point to print low order nibble
 124 013C           ; on console
 125 013C CD1501    HEX1:   CALL    CONV    ; Convert to ASCII
 126 013F C34501            JMP     ECH1    ; Go print it
 127 0142           ;
 128 0142           ; Routine ECHO reads one character from the calling
 129 0142           ; program and then echos it back. It is assumed the
 130 0142           ; console is in a full duplex mode.
 131 0142           ;
 132 0142           ; Entry parameter:      None
 133 0142           ; Exit parameter:       (A) = Character read from
 134 0142           ;                       the console keyboard
 135 0142           ; Stack usage:          4 bytes
 136 0142           ;
 137 0142 CD0301    ECHO:   CALL    CONI    ; Read a character
 138 0145           ; Alternate entry point to print (A)
 139 0145 C5        ECH1:   PUSH    B       ; Save (BC)
 140 0146 E67F              ANI     7FH     ; Strip off parity bit
 141 0148 4F                MOV     C,A     ; Put character into (C)
 142 0149           ; Alternate entry point for BLK routine
 143 0149 CD0601    ECH2:   CALL    CONO    ; Output it
 144 014C C1                POP     B       ; Restore (BC)
 145 014D C9                RET
 146 014E           ;
 147 014E           ; Routine HLPRT prints the contents of the (HL)
 148 014E           ; register as 4 hexadecimal digits on the console.
 149 014E           ;
 150 014E           ; Entry parameter:      (HL) = 4 hex digit number
 151 014E           ;                       to be printed
 152 014E           ; Exit parameter:       None
 153 014E           ; Stack usage:          10 bytes
 154 014E           ;
 155 014E CD1E01    HLPRT:  CALL    CRLF    ; Print a (CR,LF)
 156 0151           ; Alternate entry point if no CR,LF wanted
 157 0151 EB        HLPRA:  XCHG            ; Swap (HL), (DE)
 158 0152 CD2E01            CALL    DEPRA   ; Go print (DE)
 159 0155 EB                XCHG            ; Unswap (HL), (DE)
 160 0156 C9                RET
 161 0157           ;
 162 0157           ; Routine PCHK reads a character from the console
 163 0157           ; and checks if it is a valid delimiter (' ', ',',
 164 0157           ; or carriage return.)  If valid, it returns a zero
 165 0157           ; indication in the status flags.   Further, if it
```

```
 166 0157                ; is a carriage return, then the carry bit is set.
 167 0157                ; If it is not a delimiter; a non-zero, no carry
 168 0157                ; indication is required.
 169 0157                ;
 170 0157                ; Entry parameters:     None
 171 0157                ; Exit Parameters:      See description above.
 172 0157                ; Stack usage:          6 bytes
 173 0157                ;
 174 0157 CD4201         PCHK:   CALL    ECHO    ;Read a character
 175 015A                ; Alternate entry point if CHAR already in (A)
 176 015A FE20           PCH2:   CPI     ' '     ; Check for a blank
 177 015C C8                     RZ              ; Return if (SO)
 178 015D FE2C                   CPI     ','     ; Check for a comma
 179 015F C8                     RZ              ; Return if (SO)
 180 0160 FE0D                   CPI     'M'-CNTL
 181 0162                                        ; Check for a CAR RET
 182 0162 37                     STC             ; Set the carry flag
 183 0163 C8                     RZ              ; Return if CAR RET
 184 0164 3F                     CMC             ; Reset the carry flag
 185 0165 C9                     RET
 186 0166                ;
 187 0166                ; Routine PRM reads characters from the console and
 188 0166                ; pushes them onto the stack.  Multiple parameters
 189 0166                ; may be read:  values are delimited by a ' ' or a
 190 0166                ; ','.  If a carriage return is entered, it stops
 191 0166                ; reading values and returns to the caller.  Only
 192 0166                ; the last 4 characters of a string are saved:  An
 193 0166                ; error may be corrected by typing more characters
 194 0166                ; until the last 4 are correct.  The caller may
 195 0166                ; retrieve the values by popping them off the stack
 196 0166                ; in reverse order to which they were entered.
 197 0166                ;
 198 0166                ; Entry parameter:      (C) = number of expected
 199 0166                ;                       parameters
 200 0166                ; Exit parameters:      (C) Parameters on stack:
 201 0166                ;                       If a bad value was entered,
 202 0166                ;                       '????' is printed and
 203 0166                ;                       control transferred to a
 204 0166                ;                       user provided error handler.
 205 0166                ;                       The stack pointer value is
 206 0166                ;                       indeterminant and needs
 207 0166                ;                       to be reset
 208 0166                ; Stack usage:          4 + 2 = (C) bytes
 209 0166                ;
 210 0166                ; Alternate entry point if only one parameter is
 211 0166                ; desired.
 212 0166 0E01           PARM1:  MVI     C,1
 213 0168                ; Normal entry point
 214 0168 210000         PRM:    LXI     H,0     ; Set (HL) = 0
 215 016B CD4201         PRA:    CALL    ECHO    ; Get a character
 216 016E 47             PRB:    MOV     B,A     ; Save input character
 217 016F CD9901                 CALL    NIBBL   ; Check it and CVB
 218 0172 DA7E01                 JC      PRC     ; Not hex, see if delim
 219 0175 29                     DAD     H       ; Multiply (HL) by 16
 220 0176 29                     DAD     H
```

```
 221 0177 29                    DAD     H
 222 0178 29                    DAD     H
 223 0179 B5                    ORA     L       ; Add on new 4 bits
 224 017A 6F                    MOV     L,A
 225 017B C36B01                JMP     PRA     ; Go get next character
 226 017E               ;
 227 017E E3            PRC:    XTHL            ; Swap value and RET ADDR
 228 017F E5                    PUSH    H       ; Resave return address
 229 0180 78                    MOV     A,B     ; Get last input char
 230 0181 CD5A01                CALL    PCH2    ; See if delimiter
 231 0184 D28901                JNC     PRD     ; Not a carriage return
 232 0187 0D                    DCR     C       ; CR, see if all values in
 233 0188 C8                    RZ              ; Yes, done
 234 0189 C2C401        PRD:    JNZ     QPRT    ; Take error exit if not 0
 235 018C 0D                    DCR     C       ; All in?
 236 018D C26801                JNZ     PRM     ; No, go get another
 237 0190 C9                    RET
 238 0191               ;
 239 0191               ; Alternate entry point if only one parameter
 240 0191               ; wanted and first character already in (A).
 241 0191 0E01          PRF:    MVI     C,1
 242 0193 210000                LXI     H,0     ; Set up (HL)
 243 0196 C36E01                JMP     PRB     ; Go get rest of parameter
 244 0199               ;
 245 0199               ; Routine NIBBL strips the ASCII zone off a
 246 0199               ; character in the (A) reguster and verifies it is
 247 0199               ; a valid hex digit.  If so, the binary value is
 248 0199               ; returned in the lower half of the A register: the
 249 0199               ; upper half is set to zero.  If not, the carry
 250 0199               ; flag is set and return control to the caller.
 251 0199               ;
 252 0199               ; Entry Parameter:      (A) = ASCII CHAR
 253 0199               ; Exit parameters:      See description above
 254 0199               ; Stack usage:          None
 255 0199               ;
 256 0199 D630          NIBBL:  SUI     '0'     ; Strip off 0-9 Zone
 257 019B D8                    RC              ; Invalid value RET
 258 019C C6E9                  ADI     '0'-'G' ; Strip off (AF) zone
 259 019E D8                    RC              ; Invalid value RET
 260 019F C606                  ADI     6       ; Sort out in between values
 261 01A1 F2A701                JP      NIO     ; Jump if (AF)
 262 01A4 C607                  ADI     7       ; Insure it is 0-9
 263 01A6 D8                    RC              ; wasn't: Return
 264 01A7 C60A          NIO:    ADI     10      ; Adjust binary value
 265 01A9 B7                    ORA     A       ; Reset carry bit
 266 01AA C9                    RET
 267 01AB               ;
 268 01AB               ; Routine PRTWD prints a character string on the
 269 01AB               ; console.  Depending on the entry point, a CR, LF
 270 01AB               ; may be printed first.  Three forms of message end
 271 01AB               ; delimiters are accepted: Bit 7 = 1 in last
 272 01AB               ; character to be output: ASCII ETX (CONTROL C)
 273 01AB               ; following the last character: or a user specified
 274 01AB               ; delimiter following the last character.  If the
 275 01AB               ; last option is used: (B) must have the delimiter
```

```
 276 01AB             ; on entry to PRTA.
 277 01AB             ;
 278 01AB             ; Entry Parameters:     (HL) = Message start address
 279 01AB             ;                       (B) = ETX delimiter (See
 280 01AB             ;                       description above.)
 281 01AB             ; Exit Parameters:      None - (HL) is altered
 282 01AB             ; Stack usage:          12 bytes MAX
 283 01AB             ;
 284 01AB             ; Entry point for CR,LF (will not work with user
 285 01AB             ; defined ETX delimiter).
 286 01AB CD1E01      PRTWD:  CALL    CRLF
 287 01AE             ; Entry point for No. CR,LF and a bit 7 or ASCII
 288 01AE             ; ETX Delimiter.
 289 01AE C5          PRTWA:  PUSH    B       ; Save (BC)
 290 01AF 0603                MVI     B,3     ; Get an ASCII ETX
 291 01B1 CDB601              CALL    PRTA    ; Print message
 292 01B4 C5                  PUSH    B       ; Restore (BC)
 293 01B5 C9                  RET
 294 01B6             ;
 295 01B6             ; Entry point for user defined ETX delimiter
 296 01B6 78          PRTA:   MOV     A,B     ; Put ETX in A
 297 01B7 4E                  MOV     C,M     ; Get next character
 298 01B8 B9                  CMP     C       ; EOM?
 299 01B9 C8                  RZ              ; Yes, done
 300 01BA CD0601              CALL    CONO    ; No, output it
 301 01BD 79                  MOV     A,C     ; Retrieve CHAR
 302 01BE 23                  INX     H       ; Point to next CHAR
 303 01BF B7                  ORA     A       ; See if bit 7 is set
 304 01C0 F2B601              JP      PRTA    ; No, continue
 305 01C3 C9                  RET
 306 01C4             ;
 307 01C4             ; Routine QPRT prints '????', and transfers control
 308 01C4             ; to the user's error recovery routine.  (SP) is
 309 01C4             ; indeterminant on exit.
 310 01C4             ;
 311 01C4 21CD01      QPRT:   LXI     H,QMSG  ; Message address
 312 01C7 CDAE01              CALL    PRTWA   ; Print it
 313 01CA C30C01              JMP     ERR     ; Go to error recovery
 314 01CD             ;
 315 01CD 3F3F3FBF    QMSG:   DB      '???','?'+80H
 316 01D1             ;
 317 01D1             ; Hardware diagnostics can be divided into 3
 318 01D1             ; categories:
 319 01D1             ;       1) Fault detection
 320 01D1             ;       2) Faul isolation
 321 01D1             ;       3) Fault correction
 322 01D1             ; These automate the first category only.  See the
 323 01D1             ; user's manual for guidelines to the second
 324 01D1             ; category.  After the second step is done the
 325 01D1             ; fault correction should be obvious.
 326 01D1             ;
 327 01D1             ;
 328 01D1             ; SUBROUTINES FOR THE MEMORY DIAGNOSTICS
 329 01D1             ;
 330 01D1             ; When a bad memory cell is detected, this routine
```

```
 331 01D1                ; is called to print the bad address, bad data,
 332 01D1                ; test address, and test data (in that order).
 333 01D1                ; From this error log, the fault isolation process
 334 01D1                ; can be conducted.
 335 01D1                ;
 336 01D1 CD2B01     ADPRT:  CALL    DEPRT   ; Print bad address
 337 01D4 CD0F01             CALL    BLK     ; Print a blank
 338 01D7 78                 MOV     A,B     ; Get a bad data
 339 01D8 C3E001             JMP     ADPRB
 340 01DB                ;
 341 01DB                ; Alternate entry point when bad address is
 342 01DB                ; meaningless
 343 01DB F5         ADPRA:  PUSH    PSW
 344 01DC CD1E01             CALL    CRLF    ; Do a (CR,LF)
 345 01DF F1                 POP     PSW
 346 01E0 CD3301     ADPRB:  CALL    HEX2    ; Print bad data
 347 01E3 CD0F01             CALL    BLK
 348 01E6 CD0F01             CALL    BLK
 349 01E9 CD5101             CALL    HLPRA   ; Print test address
 350 01EC CD0F01             CALL    BLK
 351 01EF 79                 MOV     A,C     ; Get test data
 352 01F0 C33301             JMP     HEX2    ; Print it
 353 01F3                ;
 354 01F3                ; Routine BREAK tests the console status to see if
 355 01F3                ; a character has been typed in.  If so, it checks
 356 01F3                ; to see if it is an ASCII ETX (Cntl C). If both
 357 01F3                ; tests are met, it types an 'ABORT' message and
 358 01F3                ; returns control to the calling routine.
 359 01F3                ;
 360 01F3 CD0901     BREAK:  CALL    CST     ; Character waiting?
 361 01F6 C8                 RZ              ; No, return
 362 01F7 CD0301             CALL    CONI    ; Yes, get it
 363 01FA FE03               CPI     'C'-CNTL
 364 01FC                                    ; See if Cntl C
 365 01FC C0                 RNZ             ; No, return
 366 01FD 210702             LXI     H,ABMSG ; Print out the
 367 0200 CDAB01             CALL    PRTWD   ; 'ABORT' message
 368 0203 313E00             LXI     SP,STACK-2
 369 0206                                    ; Reset the stack
 370 0206 C9                 RET             ; Return to exec
 371 0207                ;
 372 0207 41424F52   ABMSG:  DB      'ABOR','T'+80H
     020B D4
 373 020C                ;
 374 020C                ; Routine PARM reads in the desired test block size
 375 020C                ; and block base address.  Both parameters are
 376 020C                ; pushed on the stack.
 377 020C                ;
 378 020C CDAE01     PARM:   CALL    PRTWA   ; Print caller's name
 379 020F 212402             LXI     H,BZMSG ; Print BLOCK SIZE message
 380 0212 CDAB01             CALL    PRTWD
 381 0215 CD6601             CALL    PARM1   ; Get block size
 382 0218 E1                 POP     H       ; Retrieve it
 383 0219 E3                 XTHL
 384 021A E5                 PUSH    H       ; Save return address
```

```
 385 021B 213002    PARMA:  LXI     H,BAMSG ; Print BASE ADDRESS
 386 021E CDAB01            CALL    PRTWD   ; message
 387 0221 C36601            JMP     PARM1   ; Get it and return
 388 0224           ;
 389 0224 424C4F43  BZMSG:  DB      'BLOCK SIZE:',' '+80H
     0228 4B205349
     022C 5A453AA0
 390 0230 42415345  BAMSG:  DB      'BASE'
 391 0234 20414444  ADMSG:  DB      ' ADDRESS:',' '+80H
     0238 52455353
     023C 3AA0
 392 023E           ;
 393 023E           ; Routine MADT performs a 'Walking Bit' test on
 394 023E           ; both the data and address lines of a 2114 pair at
 395 023E           ; the same time. First, it zeros all cells in the
 396 023E           ; specified block, then insures they are all zero.
 397 023E           ; It tests each 1K section separately.  Detected
 398 023E           ; errors are logged on the console as they occur.
 399 023E           ;
 400 023E           ; The base address, when asked for, must be on 1K
 401 023E           ; boundaries or it will be rejected and another
 402 023E           ; address asked for.
 403 023E           ;
 404 023E           ; The operator can abort the test at any time by
 405 023E           ; typing a ETX (Cntl C) should too many errors be
 406 023E           ; detected.  Allowing the test to complete will
 407 023E           ; insure adequate data to perform a thorough fault
 408 023E           ; isolation.
 409 023E           ;
 410 023E           ; Without errors, this diagnostic tests a 1K cell
 411 023E           ; in approximately 2 seconds.
 412 023E           ;
 413 023E 217F02    MADT:   LXI     H,WBMSG ; Sign on
 414 0241 CD0C02            CALL    PARM    ; Get parameters
 415 0244 E1        MADTA:  POP     H       ; Retreive BASE ADDRESS
 416 0245 D1                POP     D       ; Retreive BLOCK SIZE
 417 0246 7C                MOV     A,H     ; Test for 1K boundary
 418 0247 E603              ANI     3
 419 0249 B5                ORA     L
 420 024A CA6002            JZ      MADTB   ; OK, jump
 421 024D D5                PUSH    D       ; Save block size
 422 024E 217B02            LXI     H,BEMSG ; Reject base address
 423 0251 CDAB01            CALL    PRTWD
 424 0254 213002            LXI     H,BAMSG
 425 0257 CDAE01            CALL    PRTWA
 426 025A CD1B02            CALL    PARMA   ; Ask for another
 427 025D C34402            JMP     MADTA   ; Test it again
 428 0260           ;
 429 0260 CD9902    MADTB:  CALL    ZTBK    ; Zero the block
 430 0263 D5        MADTC:  PUSH    D       ; Save block size
 431 0264 3E04              MVI     A,4     ; Set 1K sections
 432 0266 BA                CMP     D       ; See if < 1K
 433 0267 F26B02            JP      MADTD   ; Yes, test it
 434 026A 57                MOV     D,A     ; No, set to 1K
 435 026B CDBB02    MADTD:  CALL    WLKAD   ; Test it
```

2114 MEMORY DIAGNOSTIC VER 1.1

```
 436 026E E1                POP     H       ; Get remaing size
 437 026F 7D                MOV     A,L     ; Subtract tested size
 438 0270 93                SUB     E
 439 0271 6F                MOV     L,A
 440 0272 7C                MOV     A,H
 441 0273 9A                SBB     D
 442 0274 67                MOV     H,A
 443 0275 C8                RZ              ; Return if done
 444 0276 EB                XCHG            ; (DE) = untested
 445 0277                                   ; (HL) = previous increment
 446 0277 09                DAD     B       ; Set new base address
 447 0278 C36302            JMP     MADTC   ; Do it again
 448 027B             ;
 449 027B 424144A0  BEMSG:  DB      'BAD',' '+80H
 450 027F 57414C4B  WBMSG:  DB      'WALKING BIT TEST',' '+80H
     0283 494E4720
     0287 42495420
     028B 54455354
     028F A0
 451 0290 54455354  TDMSG:  DB      'TEST DON','E'+80H
     0294 20444F4E
     0298 C5
 452 0299             ;
 453 0299             ; Routine ZTBK zeros and tests for a contiguous
 454 0299             ; block of memory.  On entry, the (DE) register
 455 0299             ; must have the block size and (HL) register must
 456 0299             ; have the base address.  These values are restored
 457 0299             ; to the registers on exit from the routine.
 458 0299             ;
 459 0299 D5        ZTBK:   PUSH    D       ; Save block size
 460 029A E5                PUSH    H       ; Save base address
 461 029B 0E00              MVI     C,0
 462 029D 71        ZTBKA:  MOV     M,C     ; Write into the block
 463 029E 23                INX     H       ; Next address
 464 029F 1B                DCX     D       ; Loop control
 465 02A0 7B                MOV     A,E
 466 02A1 B2                ORA     D
 467 02A2 C29D02            JNZ     ZTBKA   ; Loop if not zeroed
 468 02A5 E1                POP     H       ; Restore registers
 469 02A6 D1                POP     D
 470 02A7 D5                PUSH    D       ; Save parameters
 471 02A8 E5                PUSH    H
 472 02A9 7E        ZTBKB:  MOV     A,M     ; Read  a cell
 473 02AA B9                CMP     C       ; Same as written?
 474 02AB C4DB01            CNZ     ADPRA   ; Log error if necessary
 475 02AE CDF301            CALL    BREAK   ; See if abort wanted
 476 02B1 23                INX     H       ; Next address
 477 02B2 1B                DCX     D       ; Loop control
 478 02B3 7B                MOV     A,E
 479 02B4 B2                ORA     D
 480 02B5 C2A902            JNZ     ZTBKB   ; Loop if more to do
 481 02B8 E1                POP     H       ; Restore base address
 482 02B9 D1                POP     D       ; Restore block size
 483 02BA C9                RET
 484 02BB             ;
```

```
 485 02BB                ; Routine WLKAD walks a single high bit thru all
 486 02BB                ; data and addresses in a controled manner.  After
 487 02BB                ; writing data, all other locations are tested for
 488 02BB                ; a zero.  When an error is detected, it is logged
 489 02BB                ; as described above.  If excessive errors occur,
 490 02BB                ; the test may be aborted by typing a ETX (Cntl C).
 491 02BB                ;
 492 02BB D5             WLKAD:  PUSH    D       ; Save block size
 493 02BC E5                     PUSH    H       ; Save address
 494 02BD 23                     INX     H       ; Set A0
 495 02BE 0E11           WLKDA:  MVI     C,11H   ; Set D0, D4 (2114)
 496 02C0 C5             WLKC:   PUSH    B       ; Save it
 497 02C1 71                     MOV     M,C     ; Write byte into memory
 498 02C2 E5                     PUSH    H       ; Save current address
 499 02C3 33                     INX     SP      ; Adjust stack to
 500 02C4 33                     INX     SP      ; find base address
 501 02C5 33                     INX     SP
 502 02C6 33                     INX     SP
 503 02C7 E1                     POP     H       ; Retreive base address
 504 02C8 E5                     PUSH    H       ; Restore it
 505 02C9 3B                     DCX     SP      ; Readjust stack
 506 02CA 3B                     DCX     SP
 507 02CB 3B                     DCX     SP
 508 02CC 3B                     DCX     SP
 509 02CD 7E             WLKB:   MOV     A,M     ; Read byte
 510 02CE 47                     MOV     B,A     ; Save byte in (B)
 511 02CF A7                     ANA     A       ; Test data
 512 02D0 EB                     XCHG
 513 02D1 E3                     XTHL            ; Get test address
 514 02D2                                        ; Save loop control
 515 02D2 C2DE02                 JNZ     DNZT    ; Non-zero data, jump
 516 02D5 CD1703                 CALL    CHLDE   ; Test addresses
 517 02D8 CCD101                 CZ      ADPRT   ; Bad cell
 518 02DB C3E802                 JMP     CONT    ; Continue test
 519 02DE                ;
 520 02DE B9             DNZT:   CMP     C       ; See if same as test data
 521 02DF C2E502                 JNZ     BADD    ; Jump if bad data
 522 02E2 CD1703                 CALL    CHLDE   ; Test addresses
 523 02E5 C4D101         BADD:   CNZ     ADPRT
 524 02E8 CDF301         CONT:   CALL BREAK      ; See if abort wanted
 525 02EB E3                     XTHL            ; Unscramble registers
 526 02EC EB                     XCHG
 527 02ED 23                     INX     H       ; Next address
 528 02EE 1B                     DCX     D
 529 02EF 7B                     MOV     A,E
 530 02F0 B2                     ORA     D       ; Done on this cell?
 531 02F1 C2CD02                 JNZ     WLKB    ; No, jump
 532 02F4 E1                     POP     H       ; Get test address
 533 02F5 C1                     POP     B       ; Get data
 534 02F6 33                     INX     SP
 535 02F7 33                     INX     SP
 536 02F8 D1                     POP     D       ; Get block size
 537 02F9 D5                     PUSH    D
 538 02FA 3B                     DCX     SP
 539 02FB 3B                     DCX     SP
```

```
 540 02FC 79                MOV     A,C     ; Get data into (A)
 541 02FD 07                RLC             ; Shift for next pattern
 542 02FE 4F                MOV     C,A
 543 02FF D2C002            JNC     WLKC    ; Not done yet
 544 0302 C1                POP     B       ; Get base address
 545 0303 D1                POP     D       ; Get block size
 546 0304 3600              MVI     M,0     ; Reset test cell
 547 0306 7D                MOV     A,L     ; Strip off base
 548 0307 91                SUB     C       ; address
 549 0308 6F                MOV     L,A
 550 0309 7C                MOV     A,H
 551 030A 98                SBB     B
 552 030B 67                MOV     H,A
 553 030C 29                DAD     H       ; Go to next address bit
 554 030D CD1703            CALL    CHLDE   ; See if done
 555 0310 F0                RP              ; Yes, return
 556 0311 09                DAD     B       ; Build next address
 557 0312 D5                PUSH    D       ; Save block size
 558 0313 C5                PUSH    B       ; Save base address
 559 0314 C3BE02            JMP     WLKDA   ; Go do it again
 560 0317           ;
 561 0317           ; Compare (HL) register to (DE) register and set
 562 0317           ; flags on result.
 563 0317           ;
 564 0317 7C        CHLDE:  MOV     A,H
 565 0318 92                SUB     D
 566 0319 C0                RNZ
 567 031A 7D                MOV     A,L
 568 031B 93                SUB     E
 569 031C C9                RET
 570 031D           ;
 571 031D           ; Routine BRNIN continuously writes a sequence of
 572 031D           ; non zero numbers into a specified memory block
 573 031D           ; and reads them back for comparing.  If errors
 574 031D           ; occur, they are logged on the console.  A running
 575 031D           ; error total number is also maintained. The test
 576 031D           ; may be terminated at any time by typing a ETX
 577 031D           ; (Cntl C).  At this time the running error total
 578 031D           ; is displayed on the console.  The test data
 579 031D           ; sequences are from 1 to 255 decimal, and then
 580 031D           ; repeats itself but always skiping 0.
 581 031D           ;
 582 031D 217703    BRNIN:  LXI     H,CBMSG ; Get message address
 583 0320 CD0C02            CALL    PARM    ; Write it, get parameters
 584 0323 E1                POP     H       ; Get base address
 585 0324 D1                POP     D       ; Get block size
 586 0325 0E01              MVI     C,1     ; Seed the data
 587 0327 0600              MVI     B,0     ; Initialize error count
 588 0329 C5        BRNA:   PUSH    B       ; Save data, error count
 589 032A D5                PUSH    D       ; Save block size
 590 032B E5                PUSH    H       ; Save base address
 591 032C 71        BRNB:   MOV     M,C     ; Write the data byte
 592 032D 0C                INR     C       ; Advance data patern
 593 032E C23203            JNZ     BRNC    ; Skip 0
 594 0331 0C                INR     C       ; Set to 1
```

```
 595 0332 23          BRNC:   INX     H       ; Go to next address
 596 0333 1B                  DCX     D       ; Do loop control
 597 0334 7B                  MOV     A,E
 598 0335 B2                  ORA     D
 599 0336 C22C03              JNZ     BRNB
 600 0339 E1                  POP     H       ; Get base address
 601 033A D1                  POP     D       ; Get block size
 602 033B C1                  POP     B       ; Get data seed, error count
 603 033C D5                  PUSH    D       ; Restore them
 604 033D E5                  PUSH    H
 605 033E 7E          BRND:   MOV     A,M     ; Read data byte
 606 033F B9                  CMP     C       ; Check it
 607 0340 CA4703              JZ      BRNE    ; Skip if OK
 608 0343 04                  INR     B       ; Error count
 609 0344 CDDB01              CALL    ADPRA   ; Log the error
 610 0347 0C          BRNE:   INR     C       ; Change test data
 611 0348 C24C03              JNZ     BRNF    ; Skip if not zero
 612 034B 0C                  INR     C       ; Reset to 1
 613 034C 23          BRNF:   INX     H       ; Next address
 614 034D 1B                  DCX     D       ; Loop control
 615 034E 7B                  MOV     A,E
 616 034F B2                  ORA     D
 617 0350 C23E03              JNZ     BRND
 618 0353 E1                  POP     H       ; Reset base address
 619 0354 D1                  POP     D       ; and block size
 620 0355 CD0901              CALL    CST     ; Time to quit
 621 0358 CA2903              JZ      BRNA    ; No, do it again
 622 035B CD0301              CALL    CONI    ; Get character
 623 035E FE03                CPI     'C'-CNTL
 624 0360                                     ; ETX (Cntl C)?
 625 0360 C22903              JNZ     BRNA    ; No, continue
 626 0363 CD1E01              CALL    CRLF
 627 0366 78                  MOV     A,B     ; Error count
 628 0367 CD3301              CALL    HEX2    ; Print it
 629 036A 217003              LXI     H,ERMSG ; Get error message address
 630 036D C3AE01              JMP     PRTWA   ; Print it and return to EXEC
 631 0370             ;
 632 0370 20455252    ERMSG:  DB      ' ERROR','S'+80H
     0374 4F52D3
 633 0377 434F4E54    CBMSG:  DB      'CONTINUOUS BURNIN',' '+80H
     037B 494E554F
     037F 55532042
     0383 55524E49
     0387 4EA0
 634 0389             ;
 635 0389             ; Routine INIT and EXEC initialize the computer and
 636 0389             ; monitors the console for a command.  When a valid
 637 0389             ; command is received, control is  transfered to
 638 0389             ; the desired routine.
 639 0389             ;
 640 0389 219002      RETN:   LXI     H,TDMSG ; Print 'TEST DONE'
 641 038C CDAB01              CALL    PRTWD
 642 038F 314000      INIT:   LXI     SP,STACK        ; Set stack pointer
 643 0392 21AC03      EXEC:   LXI     H,DIMSG ; Print diag message
 644 0395 CDAB01              CALL    PRTWD
```

2114 MEMORY DIAGNOSTIC VER 1.1

```
 645 0398 218903            LXI     H,RETN  ; Set up return address
 646 039B E5                PUSH    H
 647 039C CD0301            CALL    CONI    ; Wait for command
 648 039F FE43              CPI     'C'     ; Continuous burnin
 649 03A1 CA1D03            JZ      BRNIN
 650 03A4 FE57              CPI     'W'     ; Walking bit
 651 03A6 CA3E02            JZ      MADT
 652 03A9 C3C401            JMP     QPRT
 653 03AC           ;
 654 03AC 44494147  DIMSG:  DB      'DIAGNOSTIC:',' '+80H
     03B0 4E4F5354
     03B4 49433AA0
 655 03B8           ;
 656 03B8   0000            END

TOTAL ERRORS=00
```

2114 MEMORY DIAGNOSTIC VER 1.1

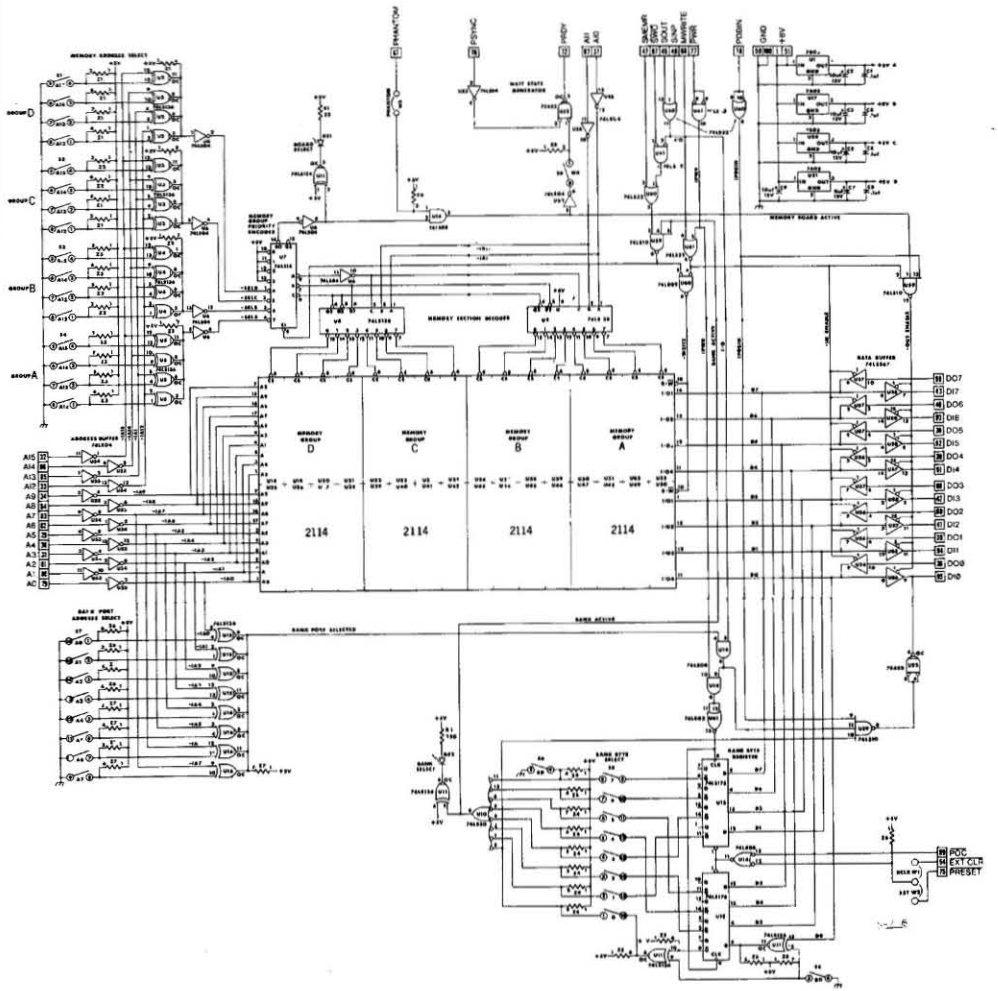
```
ABMSG  L 0207    ADMSG  L 0234    ADPRA  L 01DB    ADPRB  L 01E0
ADPRT  L 01D1    BADD   L 02E5    BAMSG  L 0230    BEMSG  L 027B
BLK    L 010F    BREAK  L 01F3    BRNA   L 0329    BRNB   L 032C
BRNC   L 0332    BRND   L 033E    BRNE   L 0347    BRNF   L 034C
BRNIN  L 031D    BZMSG  L 0224    CBMSG  L 0377    CHLDE  L 0317
CNTL   E 0040    CONI   L 0103    CONIN  E C003    CONO   L 0106
CONOU  E C006    CONST  E C373    CONT   L 02E8    CONV   L 0115
CR     E 000D    CRLF   L 011E    CRMSG  L 0127    CST    L 0109
DEPRA  L 012E    DEPRT  L 012B    DIMSG  L 03AC    DNZT   L 02DE
ECH1   L 0145    ECH2   L 0149    ECHO   L 0142    ERMSG  L 0370
ERR    L 010C    EXEC   L 0392    HEX1   L 013C    HEX2   L 0133
HLPRA  L 0151    HLPRT  L 014E    INIT   L 038F    LF     E 000A
MADT   L 023E    MADTA  L 0244    MADTB  L 0260    MADTC  L 0263
MADTD  L 026B    NIBBL  L 0199    NIO    L 01A7    PARM   L 020C
PARM1  L 0166    PARMA  L 021B    PCH2   L 015A    PCHK   L 0157
PRA    L 016B    PRB    L 016E    PRC    L 017E    PRD    L 0189
PRF    L 0191    PRM    L 0168    PRTA   L 01B6    PRTWA  L 01AE
PRTWD  L 01AB    QMSG   L 01CD    QPRT   L 01C4    RETN   L 0389
STACK  E 0040    TDMSG  L 0290    USER   E C000    WBMSG  L 027F
WLKAD  L 02BB    WLKB   L 02CD    WLKC   L 02C0    WLKDA  L 02BE
ZTBK   L 0299    ZTBKA  L 029D    ZTBKB  L 02A9

TOTAL ERRORS=00

!
```

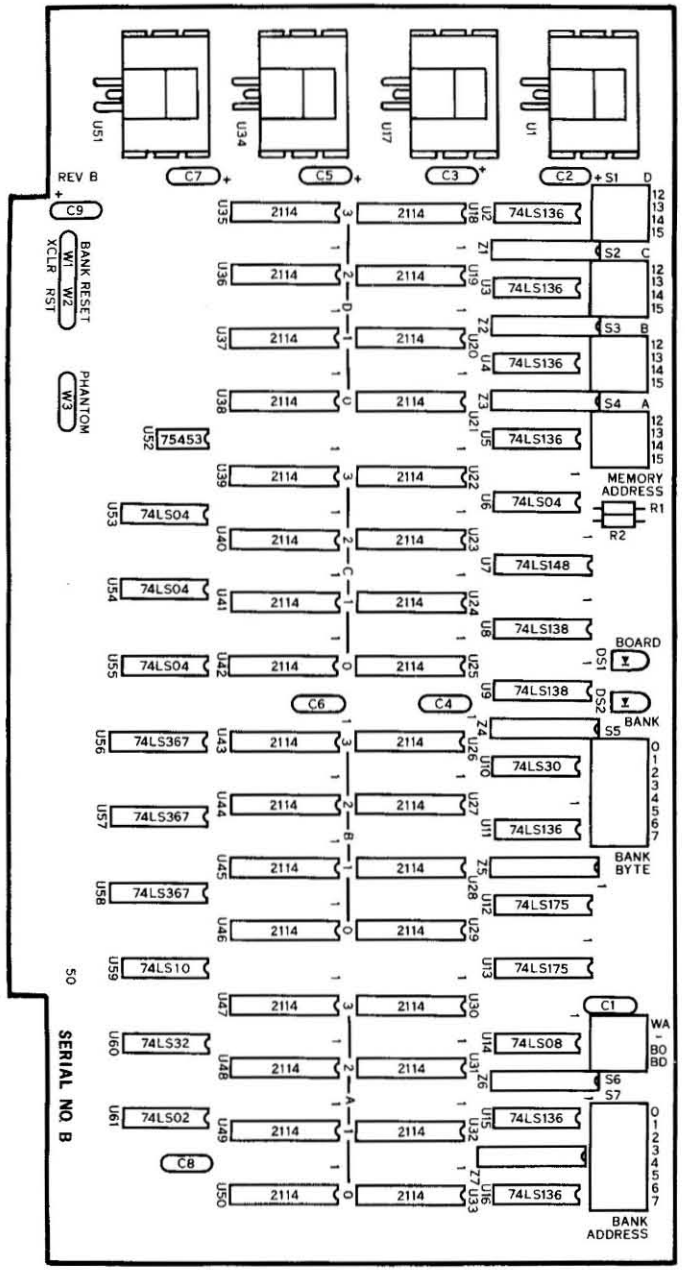
```
:03004000C38F0368
:20010000C38F03C303C0C306C0C373C3C300C0C50E20C34901E60FC69027CE4027C9E52189
:200120002701CDAE01E1C90D0A20A0CD1E017ACD33017BF50F0F0F0FCD3C01F1CD1501C3E6
:20014000450ICD0301C5E67F4FCD0601C1C9CD1E01EBCD2E01EBC9CD4201FE20C8FE2CC842
:20016000FE0D37C83FC90E01210000CD420147CD9901DA7E0129292929B56FC36B01E3E567
:2001800078CD5A01D289010DC8C2C4010DC26801C90E01210000C36E01D630D8C6E9D8C679
:2001A00006F2A701C607D8C60AB7C9CD1E01C50603CDB601C5C9784EB9C8CD06017923B770
:2001C000F2B601C921CD01CDAE01C30C013F3F3FBFCD2B01CD0F0178C3E001F5CD1E01F132
:2001E000CD3301CD0F01CD0F01CD5101CD0F0179C33301CD0901C8CD0301FE03C02107027D
:20020000CDAB01313E00C941424F52D4CDAE01212402CDAB01CD6601E1E3E5213002CDAB51
:2002200001C36601424C4F434B2053495A453AA042415345204144445245535 33AA0217F38
:2002400002CD0C02E1D17CE603B5CA6002D5217B02CDAB01213002CDAE01CD1B02C344021B
:20026000CD9902D53E04BAF26B0257CDBB02E17D936F7C9A67C8EB09C36302424144A05786
:200280004 14C4B494E472042495420544553 54A0544553542044 4F4EC5D5E50E0071231B2C
:2002A0007BB2C29D02E1D1D5E57EB9C4DB01CDF301231B7BB2C2A902E1D1C9D5E5230E115D
:2002C000C571E533333333E1E53B3B3B3B7E47A7EBE3C2DE02CD1703CCD101C3E802B9C2FC
:2002E000E502CD1703C4D101CDF301E3EB231B7BB2C2CD02E1C13333D1D53B3B79074FD24A
:20030000C002C1D136007D916F7C986729CD1703F009D5C5C3BE027C92C07D93C9217703F3
:20032000CD0C02E1D10E010600C5D5E5710CC232030C231B7BB2C22C03E1D1C1D5E57EB95C
:20034000CA470304CDDB010CC24C030C231B7BB2C23E03E1D1CD0901CA2903CD0301FE03F4
:20036000C22903CD1E0178CD3301217003C3AE01204552524F52D3434F4E54494E554F55E3
:2003800053204255524E494EA0219002CDAB013140002 1AC03CDAB01218903E5CD0301FE35
:1803A00043CA1D03FE57CA3E02C3C401444941474E4F535449433AA072
:00000000

!
```

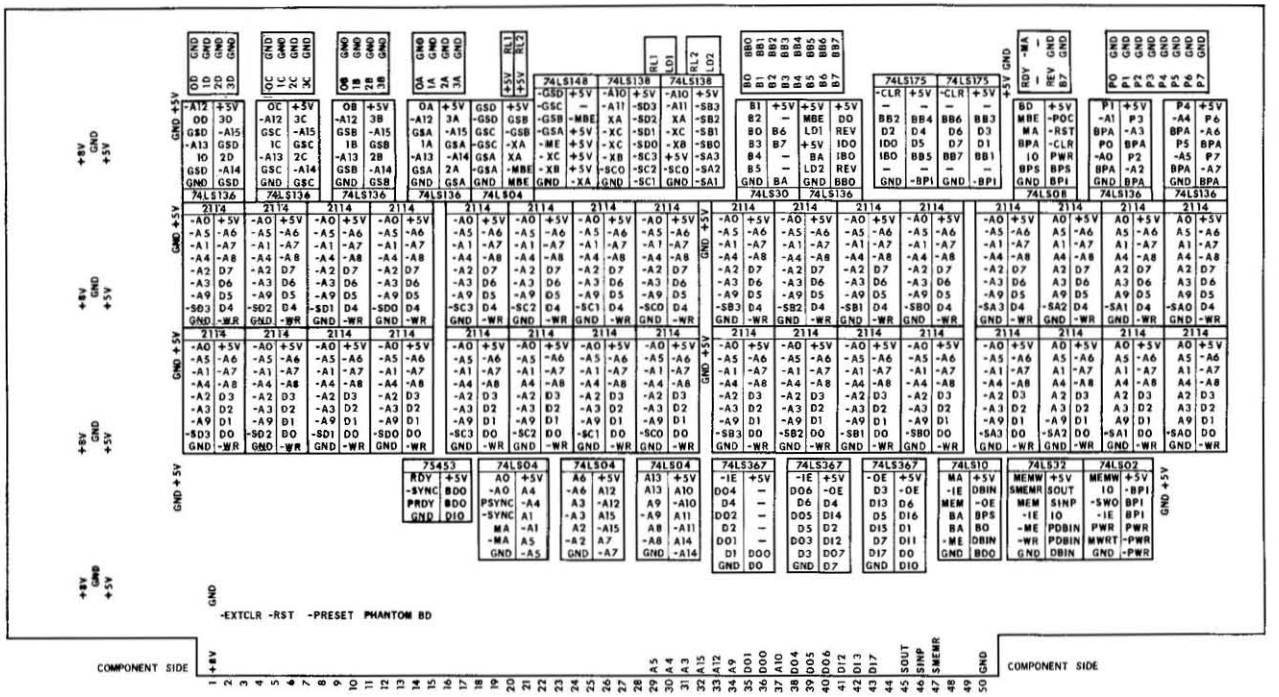
4 X 4K BANK SELECT STATIC RAM BOARD

PARTS LIST 2016B (M-XVI)

QTY	REF	NOMENCLATURE	DESCRIPTION
4	C1,4,6,8	.1UF,50V	CERAMIC CAPACITOR
5	C2,3,5,7,9	10UF,15V	TANTALUM CAPACITOR
2	DS1-2		DIODE, LIGHT EMITTING
2	R1-2	220 OHM,1/4W	RESISTER
5	S1-4,6	4PST	SWITCH, DIP: ROCKER
2	S5,7	8PST	SWITCH, DIP: ROCKER
4	U1,17,34,51	7805	IC, VOLTAGE REGULATOR, +5V
7	U2-5,11,15,16	74LS136	IC, QUAD EXCLUSIVE OR: OC
4	U6,53-55	74LS04 (74LS14)	IC, HEX INVERTER
1	U7	74LS148 (74148)	IC, OCTAL PRIORITY ENCODER
2	U8,9	74LS138 (74138)	IC, OCTAL DECODER
1	U10	74LS30 (7430)	IC, 8 INPUT NAND GATE
2	U12-13	74LS175 (74175)	IC, QUAD D REGISTER
1	U14	74LS08	IC, QUAD 2 INPUT AND GATE
8	U18-21,35-38	21L14 (2114)	IC, 1024 X 4 STATIC RAM
8	U22-25,39-42	21L14 (2114)	IC, 1024 X 4 STATIC RAM
8	U26-29,43-46	21L14 (2114)	IC, 1024 X 4 STATIC RAM
8	U30-33,47-50	21L14 (2114)	IC, 1024 X 4 STATIC RAM
1	U52	75453	IC, DUAL 2 INPUT OR: OC
3	U56-58	74LS367 (8T97)	IC, HEX 3 STATE BUS DRIVER
1	U59	74LS10	IC, TRI 3 INPUT NAND GATE
1	U60	74LS32	IC, QUAD 2 INPUT OR GATE
1	U61	74LS02	IC, QUAD 2 IN NOR GATE
7	Z1-7	7X 4.7K (2.7K)	NETWORK, RESISTOR: SIP
4	-(U1,17,34,51)		HEATSINK, TO220
4	-(U1,17,34,51)	6 X 32	NUT, HEX: W/LOCK WASHER (KEPS)
4	-(U1,17,34,51)	6 X 32 X 3/8	SCREW, PHILLIPS HEAD (SIMS)
1	XU52		IC SOCKET, 8 PIN
16	XU2-6,10-11, 14-16,53-55, 59-61		IC SOCKET, 14 PIN
8	XU7-9,12-13, 56-58		IC SOCKET, 16 PIN
32	XU18-33,35-50		IC SOCKET, 18 PIN
1		02016-0002B	PRINTED CIRCUIT BOARD
1		02016-0003B	MANUAL




REV B
SERIAL NO. B
BANK RESET
PHANTOM
MEMORY ADDRESS
BOARD
BANK
BANK BYTE
BANK ADDRESS


-EXTCLR -RST -PRESET PHANTOM BD
COMPONENT SIDE
CIRCUIT SIDE