ERN# 400 FV

Project E10002

PRODUCT TEST SPECIFICATION 88-RBM2

REJ.	APPVD.		APPROVAL	DATE
☐	☐	ENGR. MGR.		
☐	☐	PROJ. ENGR.		
☐	☐	CUST. SER. MGR.		
☐	☐	TEST MGR.		
☐	☐			
☐	☐			

PCC PACIFIC COMPUTER CORPORATION

TITLE: PRODUCT SPECIFICATION 88-RBM2

DATE | SHEET 1 OF 4

DWG. NO. 2008590-01 | REV. A

REVISION LEVEL OF THIS PAGE INDICATES THE HIGHEST REVISION LEVEL OF THIS DOCUMENTS SUBSEQUENT PAGES, AS SHOWN ON THE REVISION SHEET(S).

88-RBM2

PRODUCT SPECIFICATION

The 88-RBM2 [ROM BASIC MEMORY PCBA] provides MITS Extended Cassette BASIC programming language (see BASIC overview at end of Product Specification) in READ ONLY MEMORY. BASIC is located in the upper 16K of main memory, addresses 48K through 64K. This frees up the bottom 48K of memory for the BASIC user's programs. An automatic start-up circuit causes the computer to begin executing BASIC when either the RESET switch (or START switch) is actuated or when system power is first applied. This is accomplished by forcing a "jump to location 48K' instruction" using the Turnkey board.

In addition to BASIC in ROM, the RBM2 has additional system functions so that the need for an 8800B-T Module PCBA is eliminated.

A serial asynchronous input/output port on the RBM2 board interfaces the system console (or any terminal) to the computer. Most standard baud rates are available, as well as RS-232 or TTL signal configurations. The port can interrupt on input or output. The I/O port addresses are fixed at 020 and 021 (octal).

The RBM2 also provides the signals to the Turnkey (or ATTACHE) front panel for the control switches, "STOP/RUN" and "START" and the system indicators, 1) HLT (HALT), 2) I/O (INPUT/OUTPUT), 3) INTE (Computer interrupt enabled) 4) INT (Interrupt request) and PWR (+5 volt power OK).

SPECIFICATIONS:

FIRMWARE.....Extended Cassette BASIC in READ ONLY MEMORY (16K bytes)
(8 2K x 8 ROM IC's)

SERIAL I/O PORT..Configuration: RS-232 or TTL
Baud Rate: 110, 300, 1200, 4800, or 9600
Fixed I/O Address - 020 and 021 (octal)

SENSE SWITCHES...eight data switches read from I/O address 377 (octal)

POWER.....+8 volts @ 1 amp
±18 volts @ 40 ma

PHYSICAL.....5" x 10" printed circuit board
1 slot required

ALTAIR BASIC

(Overview)

BASIC is a high level programming language with English-like instruction, developed by Dartmouth College. Altair BASIC is one of the finest dialects of BASIC available, with numerous features not usually found in a microcomputer language. Such features include: intrinsic functions for mathematical calculations, strings manipulation and data handling, and diagnostics for program results.

ALTAIR EXTENDED BASIC

Integer variables are stored as double byte signed quantities with a range of -32768 to +32767. Variables stored in this manner occupy half as much space and are faster for arithmetic. Integers can be mixed with other variable types in an expression. Formulas containing mixed types of numeric variables are automatically converted to the dominate variable.

Altair Extended BASIC provides a PRINT USING statement to control the format of numerical output or the placement of text. PRINT USING allows the user to specify output format in scientific notation, integer, $fill, right hand +/- sign, etc., and the number of digits to be printed in a number. Strings may also be printed in specified width fields. In addition, the following aspects of Altair Extended BASIC off a wide range of program development and debugging facilities:

-Cassette - This cassette version includes provisions for loading or saving programs via CSAVE and CLOAD verbs and numeric data arrays via CSAVE* and CLOAD* verbs.

-Boolean Operators - AND, OR, and NOT, for bit manipulation and complex decision making.

-Direct Control of Input and Output and Memory - I/O ports may be read or written directly thorugh use of the INP and OUT instruction. This will allow the transfer of information between the outside world and BASIC. Any memory location may be inspected or changed via the PEEK and POKE verbs to allow transfer of data, and provide an interface with machine language subroutines.

-Strings- Variables and constants can contain as many as 255 alphanumeric characters. Functions may take substrings from the left, right or middle of long strings, or concatente long strings from shorter strings. Conversion can be performed between numbers and their string representation.

-Arrays - Numeric and string arrays may have as many dimensions as can be written on a program line and are limited in size only by available memory.

-FOR Loop Nesting - Nesting of FOR loops is limited only by available memory.

-Math Functions - Intrinsic math functions include: SIN, COS, TAN, LOG, EXP, SQR, SGN, ABS, INT, RND, POS.

-User Defined Functions - These may be defined using a single argument and are limited to the size of a single line.

-Automatic Line Numbering and Renumbering - Program lines can be numbered automatically as each program line is entered. Line numbers can also be renumbered automatically to reflect added or deleted lines.

-The IF...THEN...ELSE statement allows IF statement nesting which is limited only by available memory.

-A SWAP statement will exchange the values of two variables. This can speed up string sorts by a factor of two.

-The ERASE statement eliminates arrays from a program, freeing that memory space for other uses.

-Fancy Error messages explicitly describe user errors.

-Error Trapping facilities allows the user to write error detection and handling routines for error recovery or to provide a more complete explanation of errors than a BASIC supplied error message.

-A Trace flag is a valuable debugging aid. Each program line number is printed as the line is executed, to expose infinite loops and other programming pitfalls. This flag can be enabled or disabled as needed.

-Altair Extended BASIC allows the user to define functions which are not a part of the intrinsic function list. These may be of any type and have any number of arguments.

On the 88-RBM1 and 88-RBM2 PCBA's, Extended BASIC resides in Read Only Memory located in the top 16K section of the 64K Memory (48K - 64K).

QUANTITIES PER RBM2 BOARD

IC	QTY	IC#
74 L10	3	A, E, P
74 L04	1	B
74 LS 125	2	C, N
74 LS 153	1	D
74 L 75	1	F
74 LS 367	1	G
74 LS 161	3	H, U, V
74 04	2	J, R1
1488	1	K
1489	1	W
74 LS 02	1	L
74 LS 04	1	M
74 LS 93	1	R
74 L 00	1	S
74 LS 13	1	T
74 LS 244	5	X, Y, Z, A1, B1
74 LS 30	3	D1, E1, F1
74 LS 138	1	S1
74 367	2	X1, T1
6850	1	C1
8316 (ROM)	8	
7805	2	VR1, VR2
78L12C	1	VR5
79L12C	1	VR4

SOCKET	QTY
24 PIN	9
20 PIN	5
16 PIN	9
14 PIN	18

MISC	QTY
RP1, 2	4.7K
HEATSINK	2
SW1, 2 (8×SPST)	2

CAP	QTY	#
0.1 µf	20	SUPPRESSOR
47 µf/16V	4	C1, C2, C3, C4
1.0 µf	3	C5, C6, C7
22 µf/35V	4	C8, C9, C10, C11
27 pf	1	C12

RESISTOR	QTY	#
4.7K Ω	3	R3, 4, 5
1K Ω	1	R1
68 Ω	1	R6

MEMO

TO: Richard Dillard
John Ellis
Pat Godding
Pete Haugh
Kelly Kellams
Bruce Menn
Don Savit
Glenn Wolf
Bill Yates

FROM: John Poe

SUBJECT: Test Procedure For ROM BASIC Memory #2

DATE: May 5, 1978

---

1. This is a modified Memo and Test Procedure originally prepared by Bill Yates for the ROM Extended BASIC Board.

2. Attached please find the Test Procedure for the ROM BASIC Memory Board #2.

3. Anyone needing copies of the test PROM (RBM1-2) or the test tape (ROM Extended BASIC Functional Test) should see Bill Yates.

John J Poe

John Poe
Engineer

JP:mc

Attachment

PERTEC COMPUTER CORPORATION
MICROSYSTEMS & PERIPHERALS GROUP

PROCEDURE

SUBJECT:	PAGE of	SECTION
TEST PROCEDURE: ROM BASIC MEMORY BOARD #2	DATE REVISED	CHAPTER
	DATE EFFECTIVE	PROCEDURE

Originally Prepared By: Bill Yates 16 March 1978

Modified By: John Poe 3 May 1978

Engineering Manager _____

Manufacturing Engineer _____

QC Engineer _____

Test Supervisor _____

ROM BASIC MEMORY #2

ROM BASIC Memory #2 is intended for use in "Turnkey" machines, such as the Attache. It features Extended BASIC on ROMs, Auto START, a Serial Port, Sense Switches and Machine Status Lines available through an external cable.

User Instructions

RBM-2 has a hardwired Autostart and Non User addressable ROMs. The ROM memory is hardwired to 140000_8 and the Autostart jumps to 140000_8. RBM-2 resides at 140000_8 through 177777_8. Make sure your machine contains no other memory betweem these addresses.

I/O channel 377_8 is reserved for sense switches. These switches may be read by software to control I/O addressing, stop bits etc. Unless instructed by a software manual these switches will normally be set to produce 000_8 (all bits low). Switch Sw-2 on the RBM-2 board controls the sense data. When the switch sections are positioned to the right (On) the sense data will be low. Any switch positioned to the left will make its corresponding data bit High. Software manuals often use the word "Up" to specify a sense switch setting. If a manual requests a sense switch up move the switch section to the left. The numbers to the right of SW-2 specify which bit the switch section conrols. Software manuals refer to the sense switches as A8-A15; the RBM-2 sense switch numbers (8-15) correspond to A8-A15.

J1 is used to provide machine status to external leds located on the Attache keyboard. J1 could also be used in conjunction with the 8800b-t to provide status information to its leds. J2 is used for the serial port of the RBM-2. Both J1 and J2 are indexed to prevent installing the cable incorrectly.

The RMB-2's serial port is hardwired to channels 20_8 and 21_8. The port may be set up for TTL or RS232, but will normally be supplied set for TTL. Switch SW1 is used to set the Baud rate. Use the following table to set the desired Baud rate.

Baud Rate	Switches "On"
110	2,3,4,5,6,8
300	1,2,3,4,6,7
1200	2, 3, 4, 5
4800	1, 2, 6
9600	4,6

1. The following outlines the test procedure to be used to check out the ROM BASIC boards. There are two checks that must be made:

   a) Screening check

   b) Functional check

2. Screening Check

   a) This test checks the board under test by doing a byte-for-byte compare with a known good board. The board under test must be addressed at 140000 (octal) which is the standard configuration for the board. The known good board must be addressed to start at 100000 (octal) which requires modification of the board.

   b) Equipment required:

      (1) Attache chassis

      (2) 88-PMC: addressed at 0

      (3) Attache Video Boards

      (4) CRT Monitor

      (5) ROM BASIC TEST PROM (RBM1-2) installed in the A socket on the 88-PMC

      (6) A known good ROM BASIC board: board labelled "RBM 1 TEST ONLY"

   c) Setup the system using this equipment, connecting the CRT Monitor to the Attache Video Boards and connecting the Video Boards to the ROM BASIC Memory Board #2 to be tested.

   d) Place all switch sections of Sw-2 to the right (On), on the ROM BASIC Memory Board #2.

   e) On the RBM-2 Board place switch sections 4 and 6 of Sw-1 to the right (On).

   f) Detailed procedure

      (1) Turn off power on the Attache

      (2) Install the RBM-2 Board to be tested

      (3) Turn on power

      (4) Hit RESET

f) (cont.)

(5) "START TEST" will be printed on the CRT

(6) If no errors are found, "END OF TEST" will be printed on the CRT

(7) If any errors are found, the failing address (octal) followed by the chip number of the failing chip will be printed on the CRT. The format will be:

"ERROR AT XXXXXX / CHIP #N"

where XXXXXX is the address in octal and N is the chip number (1-8).

(8) Figure 1 shows the location of the different chips by chip number

(9) Turn off power.

(10) If the board failed the test, try replacing the ROM or ROM's that were identified as having errors and repeat the test. If there are still problems, the board should go to repair.

(11) If no message was printed, check the line/loc switch on the keyboard. It must be up. If there are still problems, indicate that the port is bad and send the board to repair.

3. Functional Check

a) This test checks the board under test by doing a quick test of several of the functions of BASIC.

b) Equipment Required

(1) An Attache and video monitor

(2) 88-UIO

(3) Cassette recorder

(4) Cassette tape with the ROM Extended BASIC FUNCTIONAL TEST

(5) 16K Memory board (static or dynamic) addressed at 0

c) Set up the system by connecting the monitor to the Attache and the recorder to the 88-UIO

d) Detailed Procedure

(1) Turn off power

(2) Install the ROM BASIC BOARD to be tested

(3) Turn on power

(4) Hit RESET

(5) BASIC should come up and "MEMORY SIZE?" should be printed on CRT. Hit CARRIAGE RETURN.

(6) "LINE PRINTER?" should be printed on the CRT. HIT C and CARRIAGE RETURN. Make sure CAPS lock key is down and LINE/LOC key is up.

(7) Using the cassette recorder and the CLOAD function, load the ROM EXTENDED BASIC FUNCTIONAL TEST.

(8) Run the test. The display shown in Figure 2 should be printed on the CRT. Be sure and check the display carefully to make sure it matches Figure 2.

e) Testing Sense Switches

(1) On the Attache keyboard type NEW;Hit Carriage Return then type 10? INP(255):GOTO 10 Hit Carriage Return.

(2) Type RUN-Hit Carriage Ruturn

(3) The CRT should display a column of zeros.

(4) While this program is running switch each section of Sw2.

(5) Do each switch section by its self. After verifying the correct result return the switch section to the On position

(6) All switch sections on should print Ø

Section	1	off	Should	Print	1
"	2	"	"	"	2
"	3	"	"	"	4
"	4	"	"	"	8
"	5	"	"	"	16
"	6	"	"	"	32
"	7	"	"	"	64
"	8	"	"	"	128

(7) If these results are not obtained indicate that the board has sense problems and sent it to Repair.

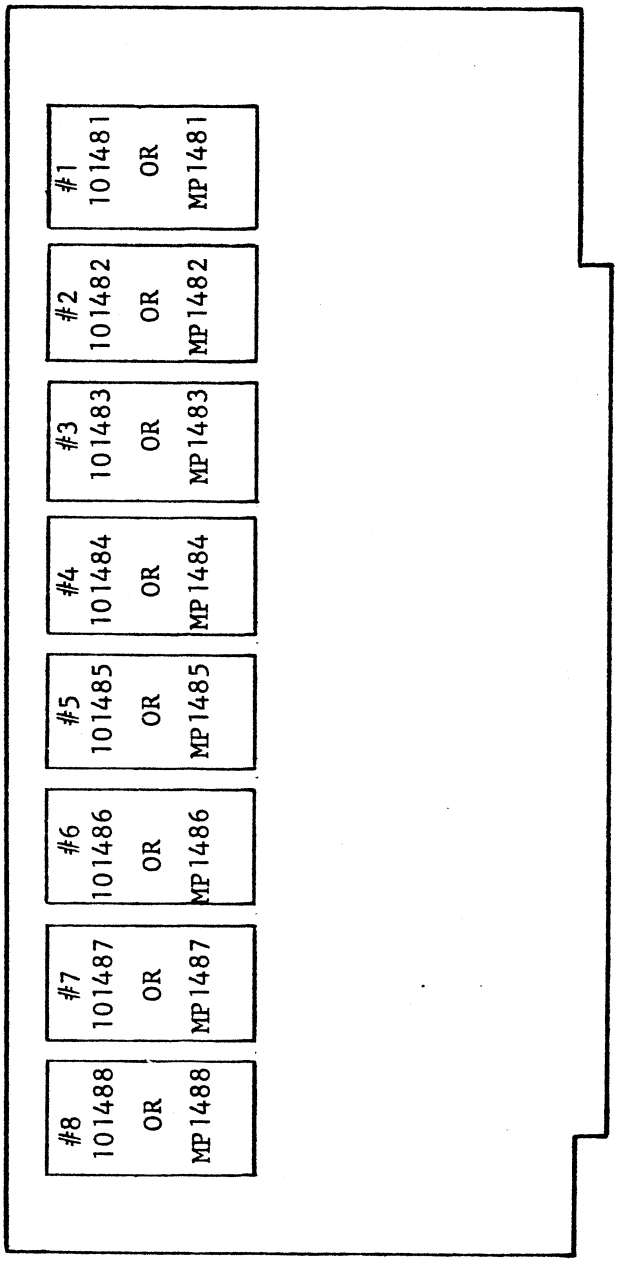

FIG 1

CHIP INSTALLATION AND CHIP NUMBERS

```
**********************************

START RØM EXTENDED BASIC LANGUAGE TEST

 2    4    6    8    10

 4.2049

 $1234.57      $4,321.76

ABCDEFGHIJKLMNØPQRSTUVWXYZ

 10  9  8  7  6  5  4  3  2  1

END ØF TEST

**********************************

ØK
```

FIG 2
PRINTOUT FOR ROM EXTENDED BASIC FUNCTIONAL TEST

TEST PROCEDURE
ROM EXTENDED BASIC BOARD
14 FEB 78
ATCH A: ROM EXTENDED BASIC FUNCTIONAL TEST LISTING


```
10 REM           RØM EXTENDED BASIC LANGUAGE TEST
20 REM                    G.W.SCHØNFELD
30 REM                    FEBRUARY 6,1978
40 CLEAR 200
50 FØR I=1 TØ 23:PRINT:NEXT
60 PRINT SPACE$(20);STRING$(32,&Ø52)
70 PRINT:PRINT
80 PRINT "START RØM EXTENDED BASIC LANGUAGE TEST"
90 PRINT:PRINT
100 DEFINT I
110 FØR I=1 TØ 10 STEP 2
120 PRINT (I*I/I)+1;SPC(2);
130 NEXT I
140 PRINT:PRINT
150 CS!=1/SIN(1)
160 CT!=1/TAN(1)
170 SC!=1/CØS(1)
180 AS!=ATN(.5/SQR(-.5*.5+1))
190 PRINT CS!+CT!+SC!+AS!
200 DEFDBL I-J
210 I=999#^5
220 J=888#^5
230 L=I*J
240 A$="ABCDEFGHIJKLMNØPQRSTUVWXYZ"
250 B$=LEFT$(A$,3)+MID$(A$,14,3)+RIGHT$(A$,3)
260 PRINT
270 I=1234.57:J=4321.77
280 PRINT USING "$$####.##";I;:PRINT TAB(14);
290 PRINT USING "$####,.##";J
300 'DEFINE FUNCTIØN TEST
310 PRINT
320 DEF FNST$(K,X$)=CHR$(K+ASC(X$))
330 FØR K=0 TØ 25
340 X$="A"
350 PRINT FNST$(K,X$);
360 NEXT K
ØK
```

TEST PROCEDURE
ROM EXTENDED BASIC BOARD
14 FEB 78
ATCH A: ROM EXTENDED BASIC FUNCTIONAL TEST LISTING
Pg 2A of 2

```
370 PRINT SPACE$(25)
380 PRINT
390 DEFINT I
400 FØR I=1 TØ 10
410 READ A(I)
420 PRINT A(I);
430 NEXT
440 RESTØRE
450 FØR I=1 TØ 10:READ A(I):NEXT I
460 DATA 10,9,8,7,6,5,4,3,2,1
470 DIM B(5):FØR I=0 TØ 5:B(I)=I:NEXT
480 ERASE B
490 DIM B(20):FØR I=0 TØ 20:B(I)=I:NEXT
500 X=123.456
510 N=FIX(X)
520 Q=SGN(X)*INT(ABS(X))
530 P=INT(X)
540 R=247 MØD 124
550 S=INT(247)-(INT(124)*(247\124))
560 T=Q+P+R+S+N
570 DEFSTR A-C
580 A="APPLE":B="BØY":C="BØY"
590 IF A<>C AND A<>B THEN GØSUB 620
600 IF A=B THEN GØTØ 660 ELSE GØSUB 640
610 GØTØ 660
620 SWAP A,C
630 RETURN
640 PRINT "NØNE ØF THEM EQUAL"
650 RETURN
660 PRINT:PRINT
670 PRINT "END ØF TEST"
680 PRINT:PRINT
690 PRINT SPACE$(20);STRING$(32,&H2A)
700 END
ØK
```

```
000000                  000100 ;               ROM EXTENDED BASIC DIAGNOSTIC
000000                  000110 ;               C.W. SCHONFELD AND C.W.VERTREES
000000                  000120 ;               VERSION 2  JANUARY 31,1978
000000                  000130 ;
000000                  000140 ;
000000                  000150 ;
000000                  000160 ;
000000                  000170 ;       THE PURPOSE OF THIS DIAGNOSTIC IS TO CHECK
000000                  000180 ;       A KNOWN GOOD ROM EXTENDED BASIC BOARD, LOCATED
000000                  000190 ;       AT 32K, AGAINST A POSSIBLY BAD BOARD LOCATED
000000                  000200 ;       AT 48K.  THE CHECK PROCEDURE IS DONE BYTE FOR
000000                  000210 ;       BYTE.  IF AN ERROR IS ENCOUNTERED, THE MES-
000000                  000220 ;       SAGE 'ERROR AT XXXXXX CHIP # X' IS PRINTED.
000000                  000230 ;       THIS PROGRAM IS DESIGNED TO RESIDE IN ONE 1702A
000000                  000240 ;       PROM LOCATED AT ADDRESS ZERO (0).  THE ONLY
000000                  000250 ;       ACTION REQUIRED OF THE OPERATOR IS TO PUT ALL
000000                  000260 ;       SENSE SWITCHES DOWN, HIT STOP, RESET AND RUN ON
000000                  000270 ;       THE FRONT PANEL.
000000                  000280 ;
000000                  000290 ;
000000                  000300 ;
000000                  000310 ;
000000                  000320 ;INITIALIZE SIO-ACIA PORT FOR 2 STOP BITS ON OCTAL PORT 20
000000                  000330 ;
000000                  000340 ;
000000 076 003           000350 INIT:   MVI     A,3             ;RESET ACIA
000002 323 020           000360         OUT     020             ;CLEAR CONTROL REGISTER
000004 076 021           000370         MVI     A,021           ;INITIALIZE FOR 2 STOP BITS
000006 323 020           000380         OUT     020             ;OUTPUT TO CONTROL REGISTER
000010 061 000311'       000390 START:  LXI     SP,STACK1+2     ;GET ADDR OF PRESET STACK
000013 041 000231'       000400         LXI     H,STRMSG        ;GET ADDR OF START MESSAGE
000016 315 000174'       000410         CALL    OUTCH           ;PRINT MSG
000021 041 137777        000420 RET1:   LXI     H,0137777       ;ADDR OF GOOD BOARD - 1
000024 061 040000        000430 LOOP:   LXI     SP,040000       ;USE (SP) AS OFFSET TO GET
000027                   000440 ;                               ;TO THE BAD BOARD
000027 043               000450         INX     H               ;INCREMENT ADDRESS
000030 104               000460         MOV     B,H             ;PUT HIGH BYTE OF (HL) IN (B)
000031                   000470 ;                               ;SO WE CAN GET CHIP NUMBER LATER
000031 175               000480         MOV     A,L             ;CHECK TO SEE IF WE STILL HAVE
000032 264               000490         ORA     H               ;DATA BY ORING (L) WITH (H)
000033 312 000215'       000500         JZ      END             ;NO MORE DATA - SO EXIT
000036 076 100           000510         MVI     A,0100          ;USE OCT 100 TO STRIP OFF 16K
000040 254               000520         XRA     H               ;TO LOOK AT LOW BOARD
000041 147               000530         MOV     H,A             ;PUT CORRECTED ADDR IN (H)
000042 176               000540         MOV     A,M             ;PUT GOOD BD DATA IN (A)
000043 071               000550         DAD     SP              ;ADD OFFSET TO LOOK AT BAD BD
000044 276               000560         CMP     M               ;COMPARE GOOD WITH BAD
000045                   000570 ;                               ;(A)=GOOD,(HL)=BAD
000045 312 000024'       000580         JZ      LOOP            ;WAS GOOD, GET NEXT BYTE
000050 061 000313'       000590         LXI     SP,STACK2+2     ;GET STACK ADDR
000053 021 000245'       000600         LXI     D,ERRMSG        ;GET ADDR OF ERROR MSG
000056 353               000610         XCHG                    ;PUT BAD ADDR IN (DE) AND
000057                   000620 ;                               ;ERRMSG ADDR IN (HL)
000057 315 000174'       000630         CALL    OUTCH           ;PRINT IT
000062                   000640 ;
000062                   000650 ;
000062                   000660 ;OUTPUT NUMBER ROUTINE.  TAKES A 16 BIT BINARY ADDRESS AND
```

```
000062                  000670 ;CONVERTS IT TO A STRAIGHT OCTAL ADDRESS (NOT SPLIT OCTAL)
000062                  000680 ;AND PRINTS IT ON THE TERMINAL.
000062                  000690 ;
000062                  000700 ;
000062 026 006          000710 OUTNUM:  MVI     D,6             ;INITIALIZE DIGIT COUNTER
000064 371              000720          SPHL                    ;STORE BAD ADDR IN [SP]
000065 257              000730          XRA     A               ;CLEAR A FOR ROTATING
000066 303 000075'      000740          JMP     FIRST           ;SHIFT LEFT 1 BIT
000071 051              000750 SECOND:  DAD     H               ;SHIFT LEFT AND
000072 027              000760          RAL                     ;ROTATE INTO [A]
000073 051              000770          DAD     H               ;AGAIN
000074 027              000780          RAL
000075 051              000790 FIRST:   DAD     H               ;AND AGAIN
000076 027              000800          RAL
000077 346 007          000810          ANI     07              ;USE LOW ORDER 3 BITS
000101 366 060          000820          ORI     060             ;MAKE IT ASCII
000103 117              000830 OUT1:    MOV     C,A             ;STORE ASCII CHAR IN [C]
000104 333 020          000840 OUT:     IN      020             ;GET PORT STATUS
000106 346 002          000850          ANI     02              ;READY YET?
000110 312 000104'      000860          JZ      OUT             ;NO - CHECK AGAIN
000113 171              000870          MOV     A,C             ;YES - GET THE CHAR
000114 323 021          000880          OUT     021             ;SEND IT TO THE TERMINAL
000116 007              000890          RLC                     ;IS IT THE CHIP NUMBER?
000117 332 000157'      000900          JC      CARET           ;YES - GO PRINT CRLF
000122 025              000910          DCR     D               ;THROUGH YET?
000123 302 000071'      000920          JNZ     SECOND          ;NO - GET NEXT DIGIT
000126 152              000930          MOV     L,D             ;[D]=0, SO CLEAR [HL] TO GET
000127 142              000940          MOV     H,D             ;BAD ADDR BACK
000130 071              000950          DAD     SP              ;[SP]=BAD ADDR, ADD TO [HL]
000131 353              000960          XCHG                    ;PUT BAD ADDR IN [DE]
000132 061 000315'      000970          LXI     SP,STACK3+2     ;GET STACK ADDR
000135 041 000256'      000980          LXI     H,CHPMSG        ;GET ADDR OF CHIP # MSG
000140 315 000174'      000990          CALL    OUTCH           ;PRINT IT
000143                  001000 ;
000143                  001010 ;
000143                  001020 ;CHIP NUMBER ROUTINE.  TAKES THE HIGH BYTE OF THE BAD
000143                  001030 ;ADDRESS (STORED IN [B]) AND USES BITS 3-5 TO CALCULATE
000143                  001040 ;AN IC CHIP NUMBER BETWEEN 1 AND 8.
000143                  001050 ;
000143                  001060 ;
000143 170              001070 CHIP:    MOV     A,B             ;PUT HIGH BYTE IN [A]
000144 037              001080          RAR                     ;ROTATE BITS 3-5 INTO
000145 037              001090          RAR                     ;THE LOW ORDER 3 BITS
000146 037              001100          RAR
000147 346 007          001110          ANI     07              ;USE LOW ORDER 3 BITS ONLY
000151 366 260          001120          ORI     0260            ;MAKE IT ASCII,BIT 7 SET
000153 074              001130          INR     A               ;MAKE IT A DECIMAL VALUE
000154 303 000103'      001140          JMP     OUT1            ;GO OUTPUT IT TO TERMINAL
000157 353              001150 CARET:   XCHG                    ;STORE BAD ADDR IN [DE]
000160 061 000317'      001160          LXI     SP,STACK4+2     ;GET STACK ADDR
000163 041 000243'      001170          LXI     H,CRLF          ;GET ADDR OF CAR RET/LINE FEED
000166 315 000174'      001180          CALL    OUTCH           ;PRINT IT
000171 303 000024'      001190 RET4:    JMP     LOOP            ;CHECK NEXT BYTE
000174                  001200 ;
000174                  001210 ;
000174                  001220 ;MESSAGE OUTPUT ROUTINE.  IF BIT 7 IS SET ON A CHARACTER
000174                  001230 ;THEN WE KNOW WE ARE AT THE END OF THE MESSAGE STRING.
```

```
000174                          001240 ;
000174                          001250 ;
000174 333 020                  001260 OUTCH:   IN      020             ;GET STATUS
000176 346 002                  001270          ANI     02              ;CHECK IT
000200 312 000174'              001280          JZ      OUTCH           ;NOT READY - CHECK AGAIN
000203 176                      001290          MOV     A,M             ;PUT THE CHAR IN (A)
000204 323 021                  001300          OUT     021             ;SEND IT TO THE TERMINAL
000206 007                      001310          RLC                     ;LAST CHAR YET? (BIT 7 HIGH)
000207 043                      001320          INX     H               ;GET NEXT CHAR
000210 322 000174'              001330          JNC     OUTCH           ;BIT 7 NOT SET-DO IT AGAIN
000213 353                      001340          XCHG                    ;PUT BAD ADDR BACK IN (HL)
000214 311                      001350          RET                     ;AND RETURN
000215                          001360 ;
000215                          001370 ;
000215                          001380 ;END ROUTINE.  CALLS 'END OF TEST' MESSAGE AND STOPS.
000215                          001390 ;
000215                          001400 ;
000215 061 000321'              001410 END:     LXI     SP,STACK5+2     ;GET STACK ADDR
000220 041 000272'              001420          LXI     H,ENDMSG        ;GET ADDR OF END MSG
000223 315 000174'              001430          CALL    OUTCH           ;PRINT IT
000226 303 000226'              001440 STOP:    JMP     STOP            ;AND HALT
000231                          001450 ;
000231                          001460 ;
000231                          001470 ;TERMINAL MESSAGES.  BIT 7 OF THE CONSOLE TERMINAL
000231                          001480 ;MESSAGES IS SET (=1) TO DENOTE END OF STRING.
000231                          001490 ;
000231                          001500 ;
000015                          001510 CR:      EQU     015
000212                          001520 LF:      EQU     0212            ;BIT 7 SET
000231 123 124 101 122          001530 STRMSG:  DB      "START TEST"
000235 124 040 124 105
000241 123 124
000243 015 212                  001540 CRLF:    DB      CR,LF
000245 105 122 122 117          001550 ERRMSG:  DC      "ERROR AT "
000251 122 040 101 124
000255 240
000256 040 040 057 040          001560 CHPMSG:  DC      "  /  CHIP # "
000262 040 103 110 111
000266 120 040 043 240
000272 105 116 104 040          001570 ENDMSG:  DB      "END OF TEST",CR,LF
000276 117 106 040 124
000302 105 123 124 015
000306 212
000307                          001580 ;
000307                          001590 ;
000307                          001600 ;PRESET STACK AREA.  THE STACK HAS BEEN PRESET WITH THE
000307                          001610 ;RETURN ADDRESSES FOR CALL INSTRUCTIONS PERMANENTLY PLACED
000307                          001620 ;SO THAT THE DIAGNOSTIC CAN EXECUTE TOTALLY IN
000307                          001630 ;PROM WITHOUT USING ANY RAM.
000307                          001640 ;
000307                          001650 ;
000307 000021'                  001660 STACK1:  DW      RET1
000311 000062'                  001670 STACK2:  DW      OUTNUM
000313 000143'                  001680 STACK3:  DW      CHIP
000315 000171'                  001690 STACK4:  DW      RET4
000317 000226'                  001700 STACK5:  DW      STOP
000321                          001710          END     INIT
```

RMB-2

MOD:

FRONT SIDE: CUT TRACE COMING FROM A9 (A9 TO E13)

BACK SIDE: CUT TRACE COMING FROM A9 (A9 TO A13)

JMPR FROM A4 TO A9

JMPR FROM E13 TO S1

OTHER JUMPERS TO INSTALL (NOT MODS):

A-A, B-B, C-C, D-D, W10, W11, W14, W15, W17

TOTAL JUMPERS ON BOARD: 11

TOTAL CUTS ON BOARD: 2

8 | 7 | 6 | 5 | 4 | 3 | 2 | 1

A0 A1 A2 A3 A4 A5 A6 A7

PINT VI0 VI1 VI2 VI3 VI4 VI5 VI6 VI7 — OW0 OW1 OW2 OW3 OW4 OW5 OW6 OW7 OW8

D1 74LS30 — L — 1/4 74LS125 — 1/3 74L10 A

SWB SINP PDBIN PWR PRDY PSYNC DI0 DI1 DI2 DI3 DI4 DI5 DI6 DI7 PDBIN SINP DO0 DO1 DO2 DO3 DO4 DO5 DO6 DO7

1/4 74LS02 — 1/6 74LS125 — 1/3 74L10 — 1/6 74LS04

X 74LS244 — A1 74LS244 — B1 74LS244

+5V — RP2 4.7K — DIPSWITCH SW2 — RP2 4.7K

SOUT PWR S SSW DSB — 1K R2 — +5V — 1-J1

SOUT MWRITE PINTE SHLTA PINT POC PRESET EXT CLR PRDY — W11 — 1/6 74LS367 — 1/6 74LS504 — W10 — 1K R1

4-J1 3-J1 2-J1 5-J1 7-J1 8-J1 6-J1 — SH. 2 1-D

MC 78L12C — VR5 — +12 V. — +18 V. — C11 22 µF 35V. — C10 22 µF 35V. — .1 µF — C9 22 µF 35V. — C8 22 µF 35V. — VR4 — -12 V. — -18 V. — MC79L12C

1/6 74LS04 — F1 74LS30

1/4 1489 — W — CTS — 6-J2 — 4.7 K R3 — +5 V. — W15 — GND — 8-J2 — W17 — W16 — 1/4 1488 — TRANSMIT — 10-J2

1/6 74LS367 — G — 1/6 7404 — J — X2 X3 X4

1/4 1488 — K — RS-232 RTS — 9-J2 — 1-J2

4.7 K R5 — +5 V. — W13 — RECEIVE — 4-J2

W12 — C5 1 µF

4.7K R4 — K3 P3 — W14 — DCD — 5-J2

1/4 1489

VCC — 74LS13 — T — 74L10 E — 74L10 P

1/2 DIPSWITCH 1 — 1/8 DIPSWITCH 1 — 4.7K'S — RP-1 4.7K'S

U 74LS161 — V 74LS161 — H 74LS161 — LD A B C D — R — CLK — RC

1/8 DIPSWITCH — RP1 4.7K

T 74LS13

C1 8250 — D0 D1 D2 D3 D4 D5 D6 D7 CTS DSR RD SCD TD SIN TR — TX RX

mits inc

RBM 2 SCHEMATIC

PDBIN
PSYNC
A13
A12
A11
A15
A14
A10
A9
A8
A7
A6
A5
A4
A3
A2
A1
A0
DI0
DI1
DI2
DI3
DI4
DI5
DI6
DI7
74L10
74L75
74LS138
74LS125
74L04
74L00
74LS125
74LS93
74367
74LS153
74LS244
8316
SH.1 A-7
+5v.
VCC
R6 68Ω
C12 27pF
Jumper W18
Jumper W19
+5V TO IC'S B,C,D,F,P,R,S,Y,Z N,L,M,N,P',R',S' ON HEAT SINK
+5V TO IC'S A,E,G,H,J,L,M,N,T,U,V,W,X A1,B1,C1,D1,E1,F1,G1,H1,J1,X1,T1 ON HEAT SINK
7805
VR1
VR2
47µF. 16v. C3
47µF. 16v. C4
47µF. 16v. C1
47µF. 16v. C2
1µF

8	7	6	5	4	3	2	1

mits inc

RBM 2 SCHEMATIC

ZONE	LTR	DESCRIPTION	DATE	BY
	A	50122 NEW	5-17-78	

DRAWN BY JEG 2-23-78
ENGINEER 3-12-78
DATE 3-14-78
SCALE NONE
DWG NO 200834 REV A