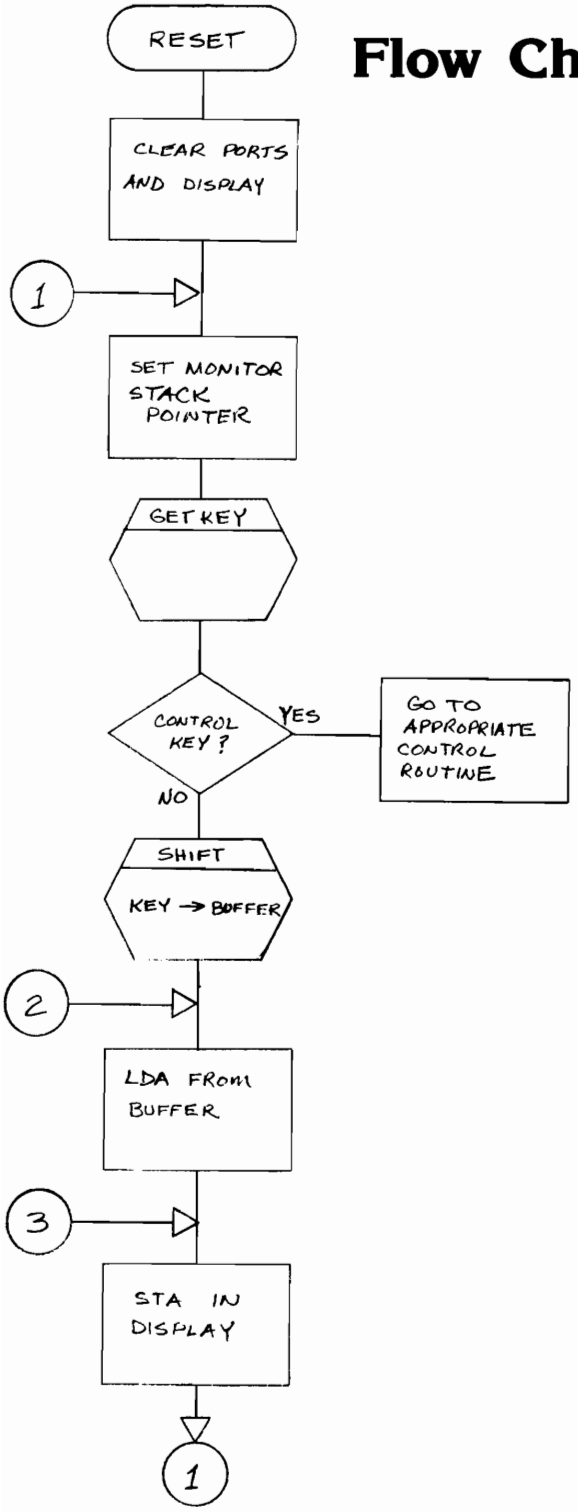
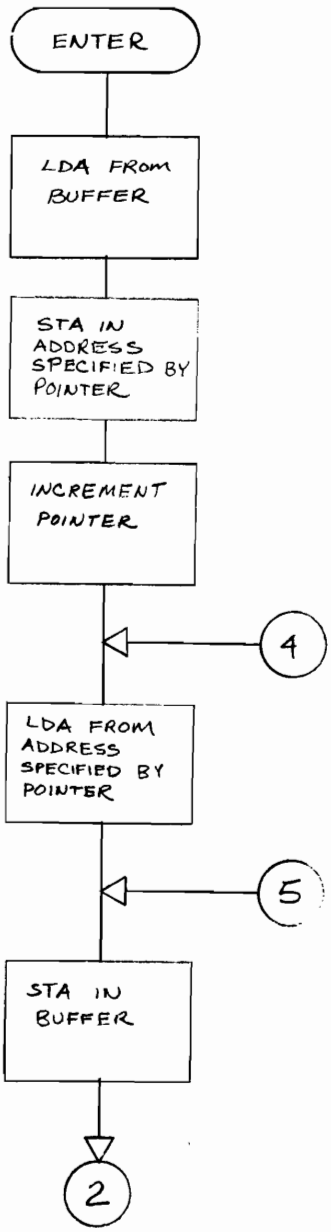
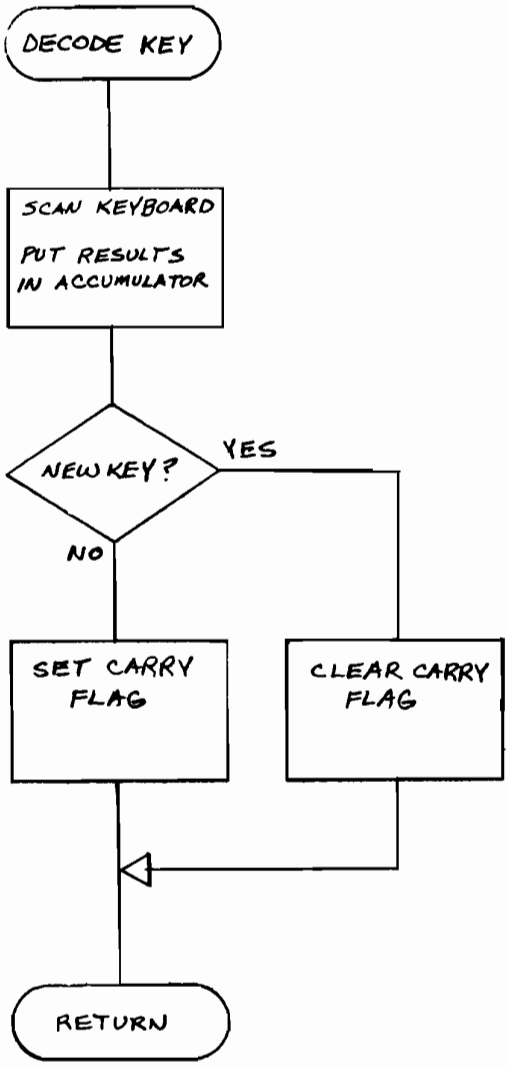
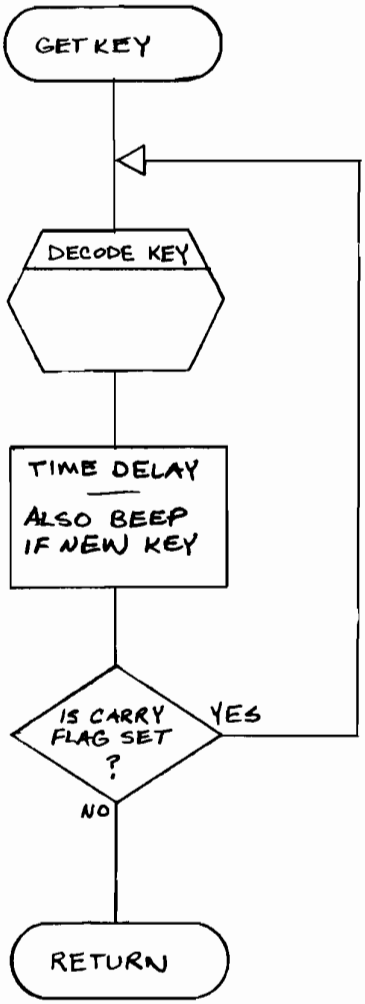
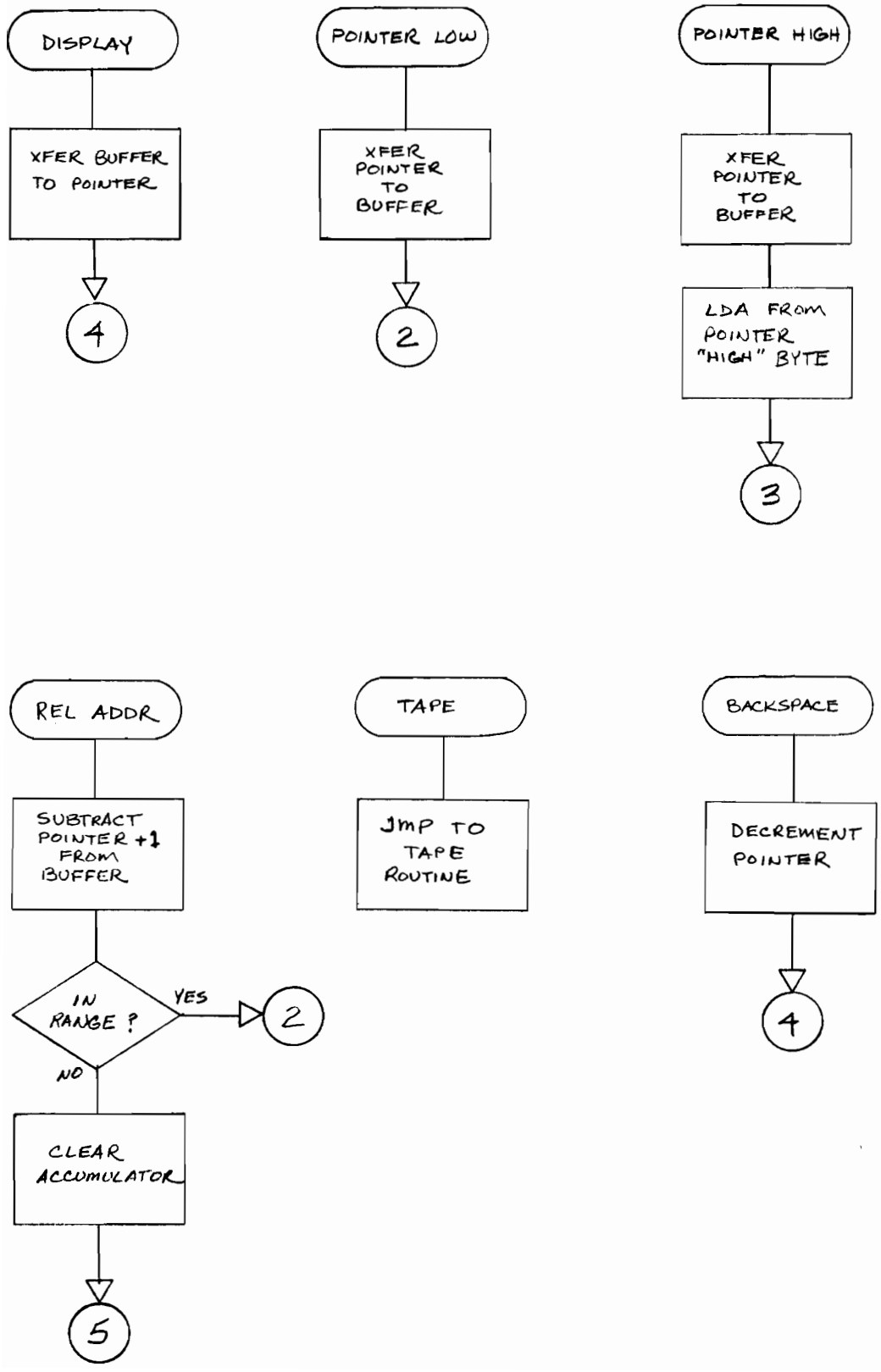
Flow Charts & Monitor Listing

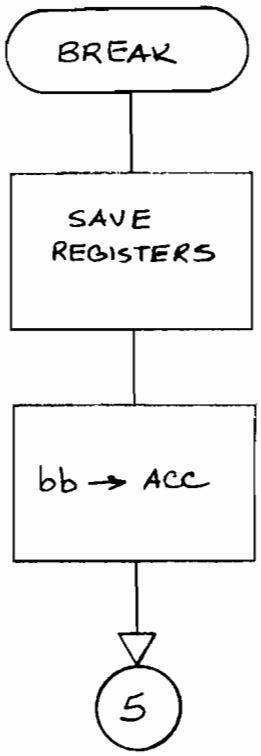


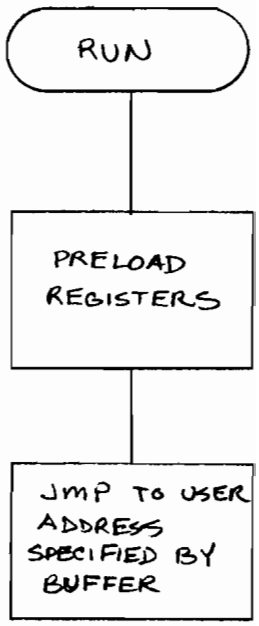


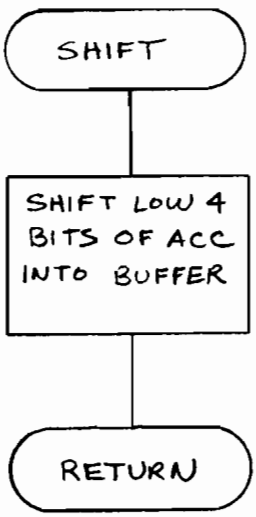

DECODE KEY
SCAN KEYBOARD
PUT RESULTS IN ACCUMULATOR
NEW KEY?
YES
NO
SET CARRY FLAG
CLEAR CARRY FLAG
RETURN


GET KEY
DECODE KEY
TIME DELAY
ALSO BEEP IF NEW KEY
IS CARRY FLAG SET ?
YES
NO
RETURN


DISPLAY
XFER BUFFER TO POINTER
4
POINTER LOW
XFER POINTER TO BUFFER
2
POINTER HIGH
XFER POINTER TO BUFFER
LDA FROM POINTER "HIGH" BYTE
3
REL ADDR
SUBTRACT POINTER +1 FROM BUFFER
IN RANGE ?
YES
2
NO
CLEAR ACCUMULATOR
5
TAPE
JMP TO TAPE ROUTINE
BACKSPACE
DECREMENT POINTER
4


BREAK
SAVE REGISTERS
bb → ACC
5


RUN
PRELOAD REGISTERS
JMP TO USER ADDRESS SPECIFIED BY BUFFER


SHIFT
SHIFT LOW 4 BITS OF ACC INTO BUFFER
RETURN

PAiA INTERACTIVE EDITOR DEBUGGER (PIEBUG)

Monitor Listing

```
0100  0200              ;
0110  0200              ;
0120  0200              ;
0130  0200              ;
0140  0200              ;
0150  0200              ;
0160  0200              ;   ****************************************
0170  0200              ;   *                                      *
0180  0200              ;   *  PIEBUG              VERSION 1.0     *
0190  0200              ;   *  PAIA INTERACTIVE EDITOR-DEBUGGER    *
0200  0200              ;   *  WRITTEN BY ROGER WALTON             *
0210  0200              ;   *  COPYRIGHT 1977 BY PAIA              *
0220  0200              ;   *    ELECTRONICS, INC.                 *
0230  0200              ;   *                                      *
0240  0200              ;   ****************************************
0250  0200              ;
0260  0200              ;
0270  0200              ;
0280  0200                      *=$0F00
0290  0F00              ;
0300  0F00              KEY     =$0800          ;BASE ADDR OF KEY PORTS
0310  0F00              TEMP    =$EE            ;TEMPORARY STORAGE
0320  0F00              LASTKE  =$F8            ;PREVIOUS KEY DECODED
0330  0F00              BUFFER  =$F0            ;KEY ENTRY BUFFER
0340  0F00              DISP    =$0820          ;LED DISPLAY
0350  0F00              MSTACK  =$ED            ;MONITOR STACK POINTER
0360  0F00              PNTER   =$F6            ;16 BIT ADDR POINTER
0370  0F00              TAPE1   =$0E00          ;START OF TAPE SYSTEM
0380  0F00              CASS    =$0900          ;CASSETTE PORT
0390  0F00              ;
0400  0F00              ;
0410  0F00              ACC     =$F9            ;REG STORAGE
0420  0F00              YREG    =$FA            ;     "
0430  0F00              XREG    =$FB            ;     "
0440  0F00              PC      =$FC            ;     "
0450  0F00              STACKP  =$FE            ;     "
0460  0F00              PREG    =$FF            ;REG STORAGE
0470  0F00              ;
0480  0F00              ;
```

```
 0490  0F00                ;      DECODE KEY SUBROUTINE
 0500  0F00                ;      THIS SUB SCANS THE ENTIRE KEYBOARD AND
 0510  0F00                ;      RETURNS WITH DECODED KEY VALUE IN A AND Y.
 0520  0F00                ;      CARRY IS CLEAR IF NEW KEY.  X IS
 0530  0F00                ;      DESTROYED.  $18 IS "NO KEY" CODE.
 0540  0F00                ;
 0550  0F00  A0 00         DECODE LDY #0           ;CLEAR RESULT REG
 0560  0F02  A2 21                LDX #$21         ;X IS PORT REG
 0570  0F04  A9 01         LOOP   LDA #1
 0580  0F06  85 EE                STA TEMP         ;SET UP MASK
0590  0F08  BD 00 08      NEXT   LDA KEY,X        ;READ CURRENT KEY PORT
 0600  0F0B  25 EE                AND TEMP         ;USE MASK TO SELECT KEY
 0610  0F0D  D0 0A                BNE RESULT       ;BRANCH IF KEY DOWN
 0620  0F0F  C8                   INY              ;SET RESULT TO NEXT KEY
 0630  0F10  06 EE                ASL TEMP         ;SHIFT MASK TO NEXT KEY
 0640  0F12  90 F4                BCC NEXT         ;BR IF MORE KEYS ON PORT
 0650  0F14  8A                   TXA
 0660  0F15  0A                   ASL A            ;SELECT NEXT PORT
 0670  0F16  AA                   TAX
 0680  0F17  90 EB                BCC LOOP         ;BRANCH IF NOT LAST PORT
 0690  0F19  C4 F8         RESULT CPY LASTKE       ;CLEAR CARRY IF NEW KEY
 0700  0F1B  84 F8                STY LASTKE       ;UPDATE LASTKEY
 0710  0F1D  98                   TYA              ;MOVE KEY TO ACC
 0720  0F1E  60                   RTS              ;RETURN
 0730  0F1F                ;
 0740  0F1F                ;
 0750  0F1F                ;
 0760  0F1F                ;      GETKEY SUBROUTINE
 0770  0F1F                ;      THIS SUB WAITS FOR A NEW KEY TO BE
 0780  0F1F                ;      TOUCHED AND THEN RETURNS WITH THE
 0790  0F1F                ;      KEY VALUE IN THE ACCUMULATOR.
 0800  0F1F                ;      X AND Y ARE CLEARED.
 0810  0F1F                ;
 0820  0F1F                ;      BEEP SUBROUTINE (EMBEDDED IN GETKEY SUB)
 0830  0F1F                ;      THIS SUB PRODUCES A SHORT BEEP AT
 0840  0F1F                ;      THE CASSETTE PORT.  CARRY MUST BE
 0850  0F1F                ;      CLEAR BEFORE ENTERING.  X AND Y
 0860  0F1F                ;      ARE CLEARED.
 0870  0F1F                ;
0880  0F1F  20 00 0F      GETKEY JSR DECODE       ;GET A KEY
0890  0F22  A2 14         BEEP   LDX #20          ;ENTER HERE FOR BEEP SUB
 0900  0F24  A0 3F         NXTX   LDY #$3F
 0910  0F26  B0 03         DELAY  BCS DLY          ;SKIP TONE IF CARRY SET
 0920  0F28  8C 00 09             STY CASS         ;GENERATE TONE
 0930  0F2B  88            DLY    DEY              ;DELAY
 0940  0F2C  D0 F8                BNE DELAY
 0950  0F2E  CA                   DEX              ;DELAY SOME MORE
 0960  0F2F  D0 F3                BNE NXTX         ;NEXT X
 0970  0F31  B0 EC                BCS GETKEY       ;BRANCH IF NOT NEW KEY
 0980  0F33  60                   RTS              ;RETURN
 0990  0F34                ;
 1000  0F34                ;
 1010  0F34                ;
 1020  0F34                ;
```

```
1030  0F34                ;      SHIFT BUFFER SUBROUTINE
1040  0F34                ;      THIS SUB SHIFTS THE LOWER 4 BITS OF
1050  0F34                ;      THE ACCUMULATOR INTO THE LEAST
1060  0F34                ;      SIGNIFICANT POSITION OF BUFFER.  THE
1070  0F34                ;      ENTIRE BUFFER IS SHIFTED 4 TIMES AND
1080  0F34                ;      THE MOST SIGNIFICANT 4 BITS ARE LOST.
1090  0F34                ;      X AND Y ARE CLEARED.  IF ON RETURN,
1100  0F34                ;      A SINGLE "ROL A" IS PERFORMED,
1110  0F34                ;      THE LOWER 4 BITS OF THE ACCUMULATOR
1120  0F34                ;      WILL CONTAIN THE 4 BITS THAT WERE
1130  0F34                ;      SHIFTED OUT OF BUFFER.
1140  0F34                ;
1150  0F34  0A            SHIFT  ASL A              ;SHIFT KEY INFORMATION
1160  0F35  0A                   ASL A              ;TO UPPER 4 BITS OF ACC
1170  0F36  0A                   ASL A
1180  0F37  0A                   ASL A
1190  0F38  A0 04                LDY #4
1200  0F3A  2A              ROTATE ROL A             ;SHIFT BIT TO CARRY
1210  0F3B  A2 FA                 LDX #$FA           ;WRAP AROUND TO $F0
1220  0F3D  36 F6           ROTNXT ROL BUFFER+6,X    ;CARRY TO BUFFER TO CARRY
1230  0F3F  E8                   INX                ;AND SO ON
1240  0F40  D0 FB                BNE ROTNXT         ;UNTIL END OF BUFFER
1250  0F42  88                   DEY                ;DONE 4 BITS?
1260  0F43  D0 F5                BNE ROTATE         ;BRANCH IF NOT
1270  0F45  60                   RTS                ;RETURN
1280  0F46                ;
1290  0F46                ;
1300  0F46                ;      RESET ENTRY POINT
1310  0F46                ;
1320  0F46  A9 00           RESET  LDA #0
1330  0F48  8D E0 08              STA $08E0          ;CLEAR DISPLAY AND PORTS
1340  0F4B  F0 08                BEQ COMAND         ;BRANCH ALWAYS
1350  0F4D                ;
1360  0F4D                ;
1370  0F4D                ;
1380  0F4D  20 34 0F      SHFTD  JSR SHIFT          ;SHIFT KEY INTO BUFFER
1390  0F50  A5 F0         DSPBUF LDA BUFFER         ;GET BUFFER
1400  0F52  8D 20 08      SEE    STA DISP           ;UPDATE DISPLAY
1410  0F55                ;
1420  0F55  A6 ED         COMAND LDX MSTACK
1430  0F57  9A                   TXS                ;SET MONITOR STACK
1440  0F58  20 1F 0F             JSR GETKEY         ;WAIT FOR KEY
1450  0F5B  C9 10                CMP #$10           ;IS IT CONTROL KEY
1460  0F5D  90 EE                BCC SHFTD          ;BRANCH IF NOT
1470  0F5F  A8                   TAY                ;CONTROL KEY INTO Y
1480  0F60  BE E2 0F             LDX TABLE-16,Y     ;GET COMMAND ADDR LOW
1490  0F63  86 EE                STX TEMP           ;SAVE IT
1500  0F65  A2 FF                LDX #$FF           ;GET COMMAND ADDR HIGH
1510  0F67  86 EF                STX TEMP+1         ;ASSEMBLE COMMAND ADDR
1520  0F69  E8                   INX                ;CLR X
1530  0F6A  6C EE 00             JMP (TEMP)         ;EXECUTE COMMAND
1540  0F6D                ;
1550  0F6D                ;
```

```
1560  0F6D  18          PHIGH  CLC
1570  0F6E  A5 F6       PLOW   LDA PNTER        ;MOVE POINTER TO BUFFER
1580  0F70  85 F0              STA BUFFER
1590  0F72  A5 F7              LDA PNTER+1
1600  0F74  85 F1              STA BUFFER+1
1610  0F76  B0 D8              BCS DSPBUF       ;BRANCH IF POINTER LOW
1620  0F78  90 D8              BCC SEE          ;BRANCH IF POINTER HIGH
1630  0F7A              ;
1640  0F7A              ;
1650  0F7A  A5 F0       DISPLA LDA BUFFER       ;MOVE BUFFER TO POINTER
1660  0F7C  85 F6              STA PNTER
1670  0F7E  A5 F1              LDA BUFFER+1
1680  0F80  85 F7              STA PNTER+1
1690  0F82  B0 14              BCS LOAD         ;BRANCH ALWAYS
1700  0F84              ;
1710  0F84              ;
1720  0F84  A5 F6       BACKSP LDA PNTER        ;DEC 16 BIT POINTER
1730  0F86  D0 02              BNE SKIP         ;BRANCH IF NO BORROW
1740  0F88  C6 F7              DEC PNTER+1
1750  0F8A  C6 F6       SKIP   DEC PNTER
1760  0F8C  B0 0A              BCS LOAD         ;BRANCH ALWAYS
1770  0F8E              ;
1780  0F8E              ;
1790  0F8E  A5 F0       ENTER  LDA BUFFER       ;GET BYTE IN BUFFER
1800  0F90  81 F6              STA (PNTER,X)    ;STORE IT IN ACTIVE CELL
1810  0F92  E6 F6              INC PNTER        ;INC 16 BIT POINTER
1820  0F94  D0 02              BNE LOAD         ;BRANCH IF NO CARRY
1830  0F96  E6 F7              INC PNTER+1
1840  0F98  A1 F6       LOAD   LDA (PNTER,X)    ;GET BYTE IN ACTIVE CELL
1850  0F9A  85 F0       STABUF STA BUFFER       ;STORE IT IN BUFFER
1860  0F9C  B0 B2              BCS DSPBUF       ;BRANCH ALWAYS
1870  0F9E              ;
1880  0F9E              ;
1890  0F9E  D8           RELADR CLD
1900  0F9F  18                  CLC              ;THIS ADDS 1 TO POINTER
1910  0FA0  A5 F0              LDA BUFFER       ;GET BUFFER LOW
1920  0FA2  E5 F6              SBC PNTER        ;SUBTRACT POINTER LOW + 1
1930  0FA4  85 F0              STA BUFFER       ;SAVE RESULTS
1940  0FA6  A5 F1              LDA BUFFER+1     ;GET BUFFER HIGH
1950  0FA8  E5 F7              SBC PNTER+1      ;SUBTRACT POINTER HIGH
1960  0FAA  A8                 TAY              ;SAVE RESULTS IN Y
1970  0FAB  A5 F0              LDA BUFFER       ;GET RESULTS LOW
1980  0FAD  B0 08              BCS POS          ;BR IF TOTAL RESULT POS
1990  0FAF  10 0A              BPL BAD          ;BR IF RESULT LOW POS
2000  0FB1  C8                 INY              ;INC RESULT HIGH
2010  0FB2  98           CHK    TYA             ;CHECK RESULT HIGH
2020  0FB3  D0 06              BNE BAD          ;BR IF NOT ZERO
2030  0FB5  F0 99              BEQ DSPBUF       ;BR ALWAYS, DISP REL ADDR
2040  0FB7  30 02       POS    BMI BAD          ;BR IF RESULT LOW NEG
2050  0FB9  10 F7              BPL CHK          ;BR ALWAYS
2060  0FBB  8A           BAD    TXA              ;CLEAR ACC
2070  0FBC  38                 SEC
2080  0FBD  B0 DB              BCS STABUF       ;BRANCH ALWAYS
2090  0FBF              ;
2100  0FBF              ;
2110  0FBF  EA                 NOP
2120  0FC0              ;
```

```
2130  OFC0                ;
2140  OFC0                ;
2150  OFC0                ;      BREAK ROUTINE ENTRY POINT
2160  OFC0                ;
2170  OFC0  85 F9         BREAK  STA ACC          ;SAVE ACCUMULATOR
2180  OFC2  84 FA                STY YREG         ;SAVE Y
2190  OFC4  86 FB                STX XREG         ;SAVE X
2200  OFC6  68                   PLA              ;GET STATUS REG
2210  OFC7  85 FF                STA PREG         ;SAVE IT
2220  OFC9  68                   PLA              ;GET PC LOW
2230  OFCA  D8                   CLD
2240  OFCB  38                   SEC
2250  OFCC  E9 02                SBC #2           ;CORRECT PC LOW
2260  OFCE  85 FC                STA PC           ;SAVE IT
2270  OFD0  68                   PLA              ;GET PC HIGH
2280  OFD1  E9 00                SBC #0           ;SUBTRACT CARRY
2290  OFD3  85 FD                STA PC+1         ;SAVE IT
2300  OFD5  BA                   TSX              ;GET USER STACK POINTER
2310  OFD6  86 FE                STX STACKP       ;SAVE IT
2320  OFD8  A9 BB                LDA #$BB         ;BREAK INDICATION
2330  OFDA  B0 BE                BCS STABUF       ;BRANCH ALWAYS
2340  OFDC                ;
2350  OFDC                ;
2360  OFDC  A6 FE         RUN    LDX STACKP       ;GET USER STACK POINTER
2370  OFDE  9A                   TXS              ;INIT STACK
2380  OFDF  A5 F1                LDA BUFFER+1     ;GET PC HIGH
2390  OFE1  48                   PHA              ;PUT IT ON STACK
2400  OFE2  A5 F0                LDA BUFFER       ;GET PC LOW
2410  OFE4  48                   PHA              ;PUT IT ON STACK
2420  OFE5  A5 FF                LDA PREG         ;GET STATUS REG
2430  OFE7  48                   PHA              ;PUT IT ON STACK
2440  OFE8  A6 FB                LDX XREG         ;RESTORE X
2450  OFEA  A4 FA                LDY YREG         ;RESTORE Y
2460  OFEC  A5 F9                LDA ACC          ;RESTORE ACCUMULATOR
2470  OFEE  40                   RTI              ;RESTORE PC & STATUS REG
2480  OFEF                ;                         FROM STACK AND EXECUTE
2490  OFEF                ;                         USER'S PROGRAM
2500  OFEF                ;
2510  OFEF                ;
2520  OFEF  4C 00 0E      TAPE   JMP TAPE1        ;EXECUTE TAPE OPTION
2530  OFF2                ;
2540  OFF2                ;
```

```
2550  0FF2                    ;       COMMAND ADDRESS TABLE
2560  0FF2                    ;       STORES LOW BYTE ONLY OF ENTRY
2570  0FF2                    ;       ADDRESS FOR EACH COMMAND
2580  0FF2                    ;
2590  0FF2  DC 0F             TABLE   .WORD RUN
2600  0FF4                            *=*-1
2610  0FF3  7A 0F                     .WORD DISPLA
2620  0FF5                            *=*-1
2630  0FF4  84 0F                     .WORD BACKSP
2640  0FF6                            *=*-1
2650  0FF5  8E 0F                     .WORD ENTER
2660  0FF7                            *=*-1
2670  0FF6  6D 0F                     .WORD PHIGH
2680  0FF8                            *=*-1
2690  0FF7  6E 0F                     .WORD PLOW
2700  0FF9                            *=*-1
2710  0FF8  EF 0F                     .WORD TAPE
2720  0FFA                            *=*-1
2730  0FF9  9E 0F                     .WORD RELADR
2740  0FFB                            *=*-1
2750  0FFA                    ;
2760  0FFA                    ;
2770  0FFA  03 00                     .WORD $0003       ;NMI VECTOR
2780  0FFC  46 0F                     .WORD RESET       ;RESET VECTOR
2790  0FFE  00 00                     .WORD $0000       ;IRQ VECTOR
2800  1000                    ;
2810  1000                    ;
2820  1000                            .END
```

```
ERRORS = 0000
```

SYMBOL TABLE

RESULT	0F19	DLY	0F2B	COMAND	0F55	LOAD	0F98
SKIP	0F8A	POS	0FB7	BAD	0FBB	TABLE	0FF2
KEY	0800	TEMP	00EE	LASTKE	00F8	BUFFER	00F0
DISP	0820	MSTACK	00ED	PNTER	00F6	TAPE1	0E00
CASS	0900	ACC	00F9	YREG	00FA	XREG	00FB
PC	00FC	STACKP	00FE	PREG	00FF	DECODE	0F00
LOOP	0F04	NEXT	0F08	GETKEY	0F1F	BEEP	0F22
NXTX	0F24	DELAY	0F26	SHIFT	0F34	ROTATE	0F3A
ROTNXT	0F3D	RESET	0F46	SHFTD	0F4D	DSPBUF	0F50
SEE	0F52	PHIGH	0F6D	PLOW	0F6E	DISPLA	0F7A
BACKSP	0F84	ENTER	0F8E	STABUF	0F9A	RELADR	0F9E
CHK	0FB2	BREAK	0FC0	RUN	0FDC	TAPE	0FEE

SCHEMATICS

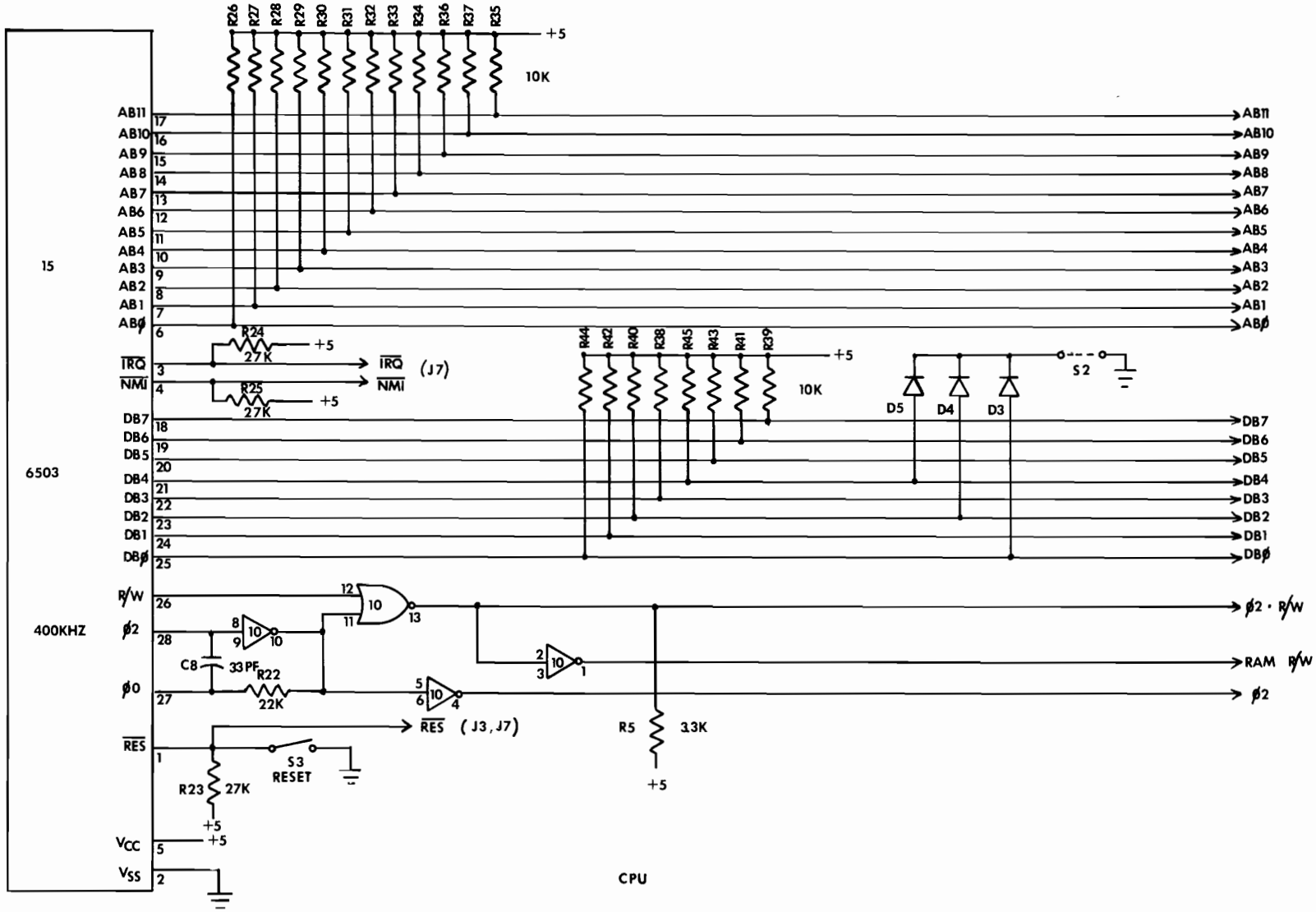
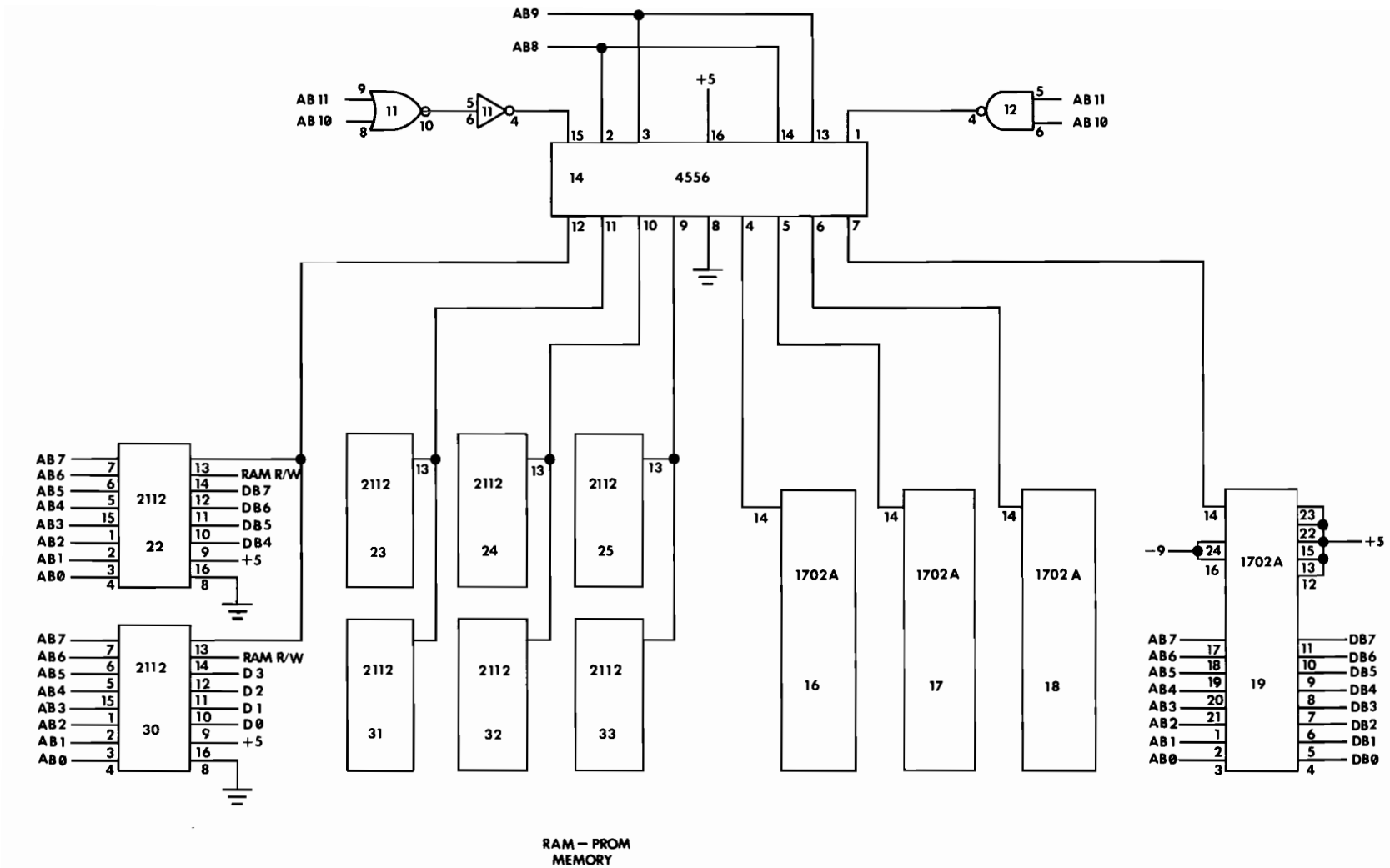

R26 R27 R28 R29 R30 R31 R32 R33 R34 R36 R37 R35
+5
10K
15
6503
400KHZ
AB11 17
AB10 16
AB9 15
AB8 14
AB7 13
AB6 12
AB5 11
AB4 10
AB3 9
AB2 8
AB1 7
ABØ 6
R24 27K +5
IRQ 3
NMI 4
R25 27K +5
IRQ (J7)
NMI
R44 R42 R40 R38 R45 R43 R41 R39
+5
10K
D5 D4 D3
S2
DB7 18
DB6 19
DB5 20
DB4 21
DB3 22
DB2 23
DB1 24
DBØ 25
R/W 26
Ø2 28
ØØ 27
C8 33PF
R22 22K
RES 1
S3 RESET
R23 27K
+5
VCC 5
VSS 2
RES (J3, J7)
R5 3.3K
+5
Ø2 · R/W
RAM R/W
Ø2
CPU

FIGURE 1



RAM – PROM
MEMORY

FIGURE 2

I/O DECODING

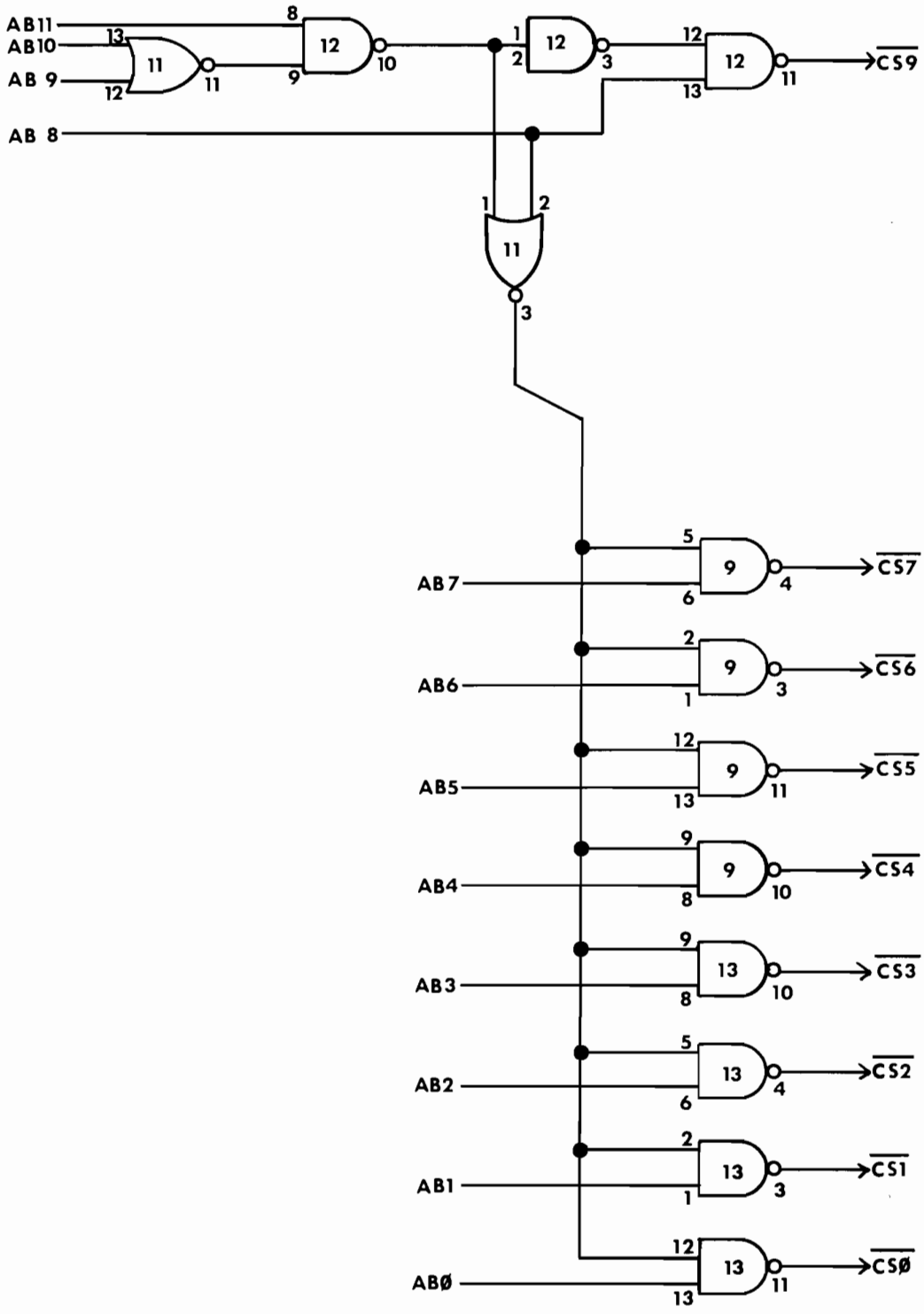


FIGURE 3

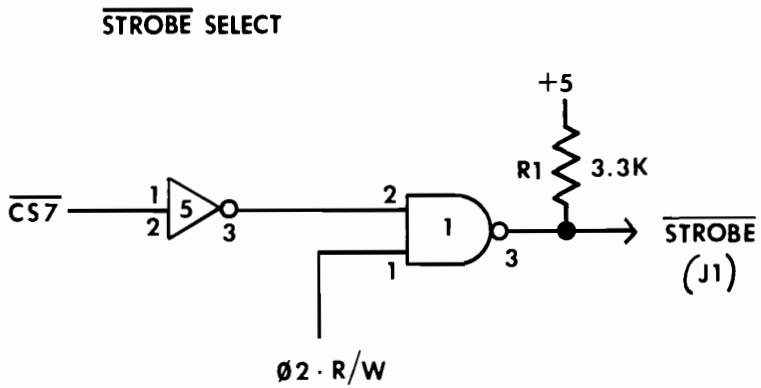


FIGURE 4

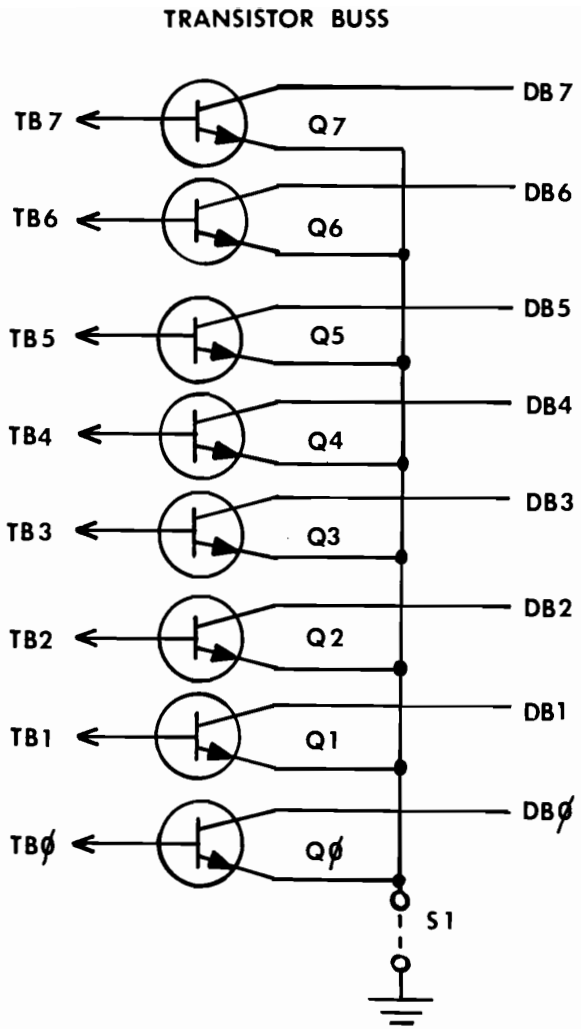
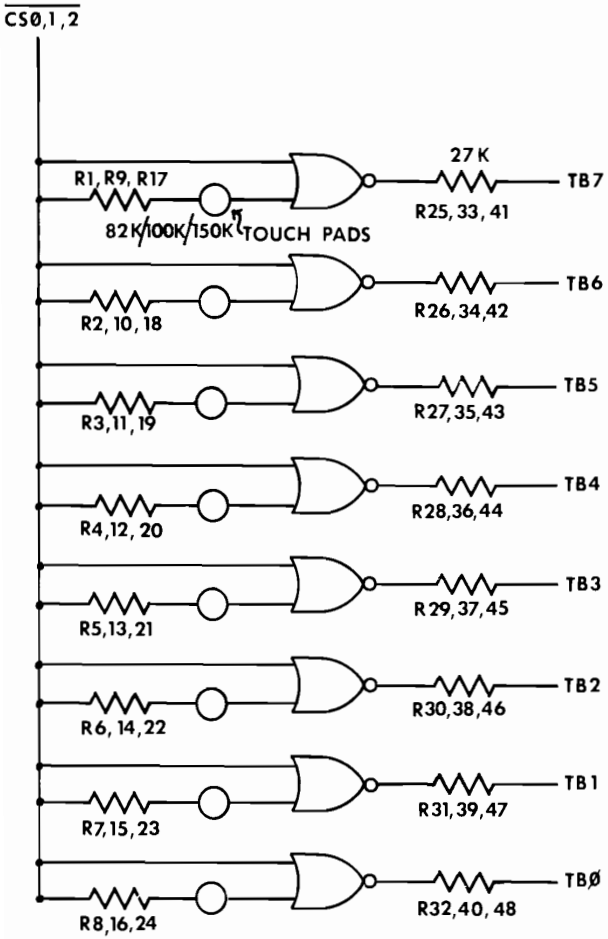
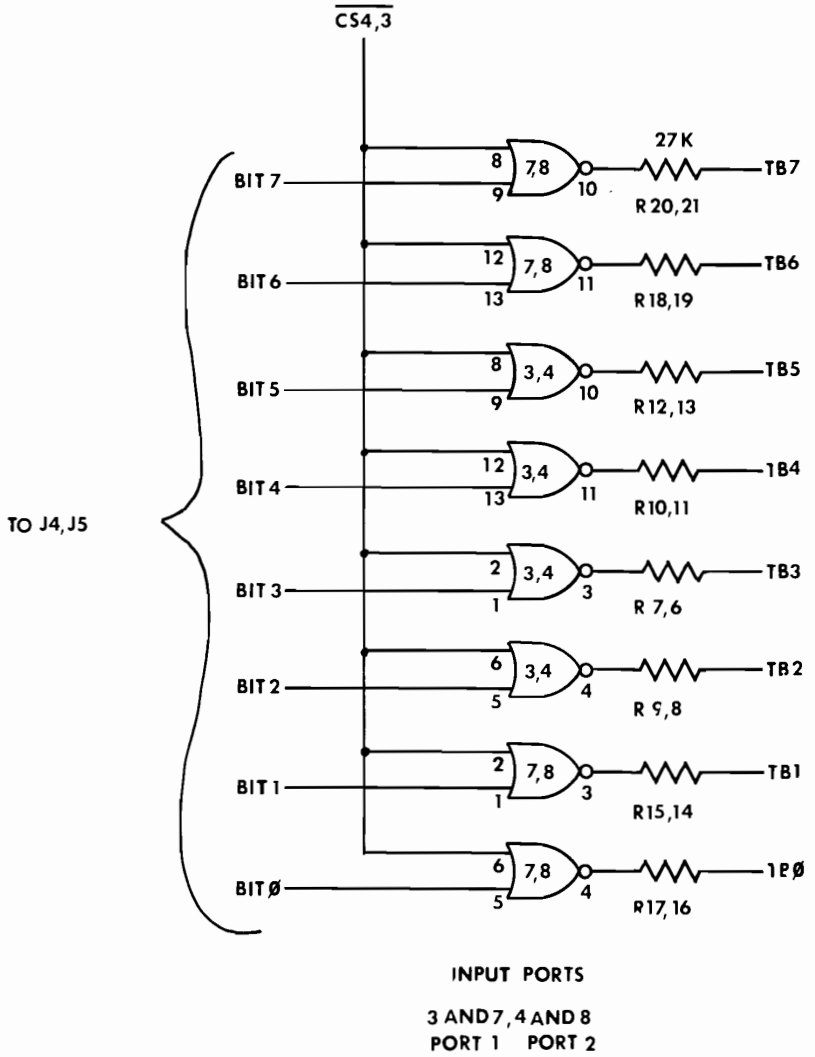


FIGURE 5



KEYBOARD

FIGURE 6



INPUT PORTS

3 AND 7, 4 AND 8
PORT 1 PORT 2

FIGURE 7

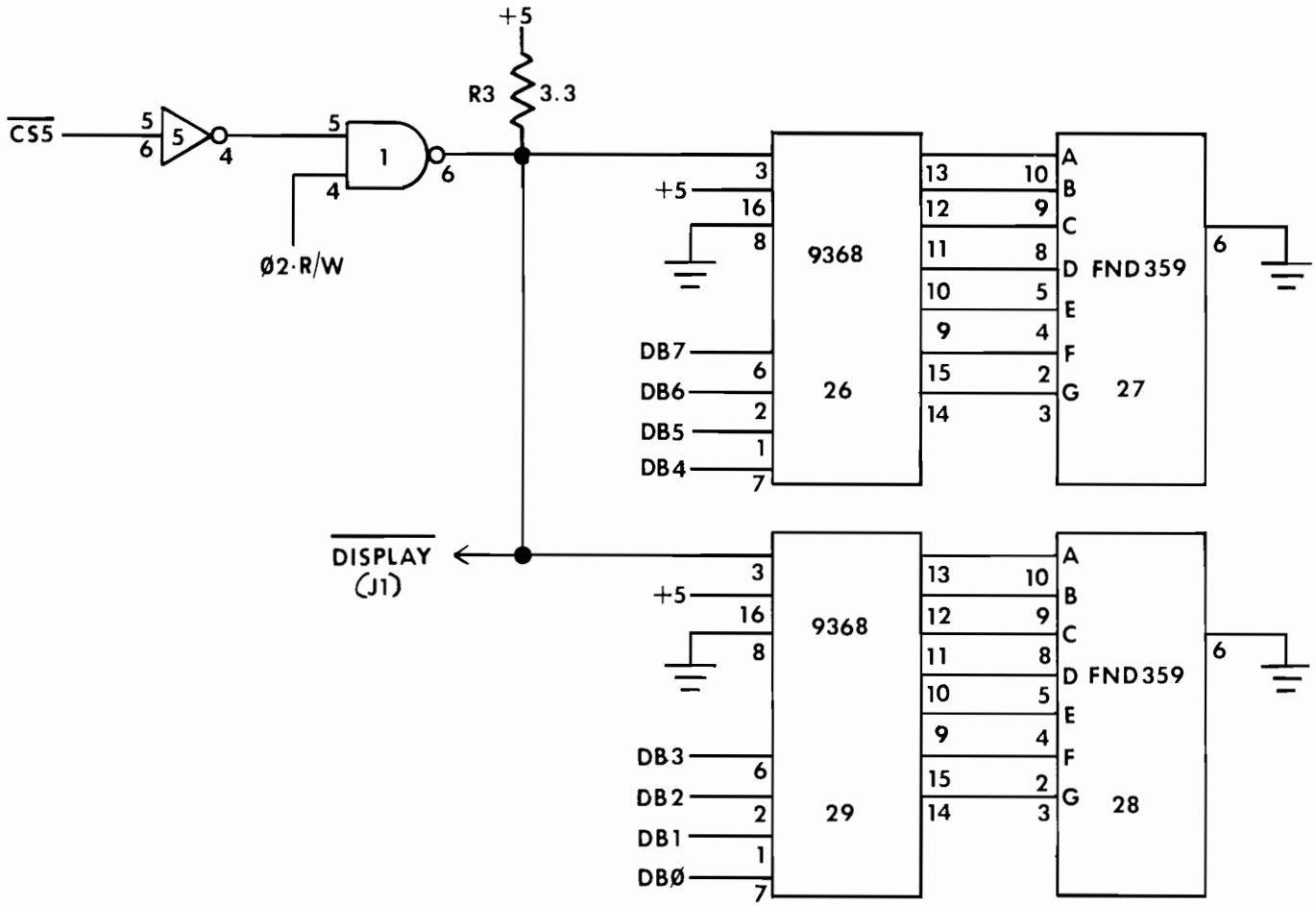

+5
R3
3.3
CS5
5
6
5
4
5
1
6
4
Ø2·R/W
+5
3
16
8
9368
26
DB7
6
DB6
2
DB5
1
DB4
7
13
12
11
10
9
15
14
10
9
8
5
4
2
3
A
B
C
D
E
F
G
FND359
27
6
DISPLAY
(J1)
+5
3
16
8
9368
29
DB3
6
DB2
2
DB1
1
DBØ
7
13
12
11
10
9
15
14
10
9
8
5
4
2
3
A
B
C
D
E
F
G
FND359
28
6

FIGURE 8

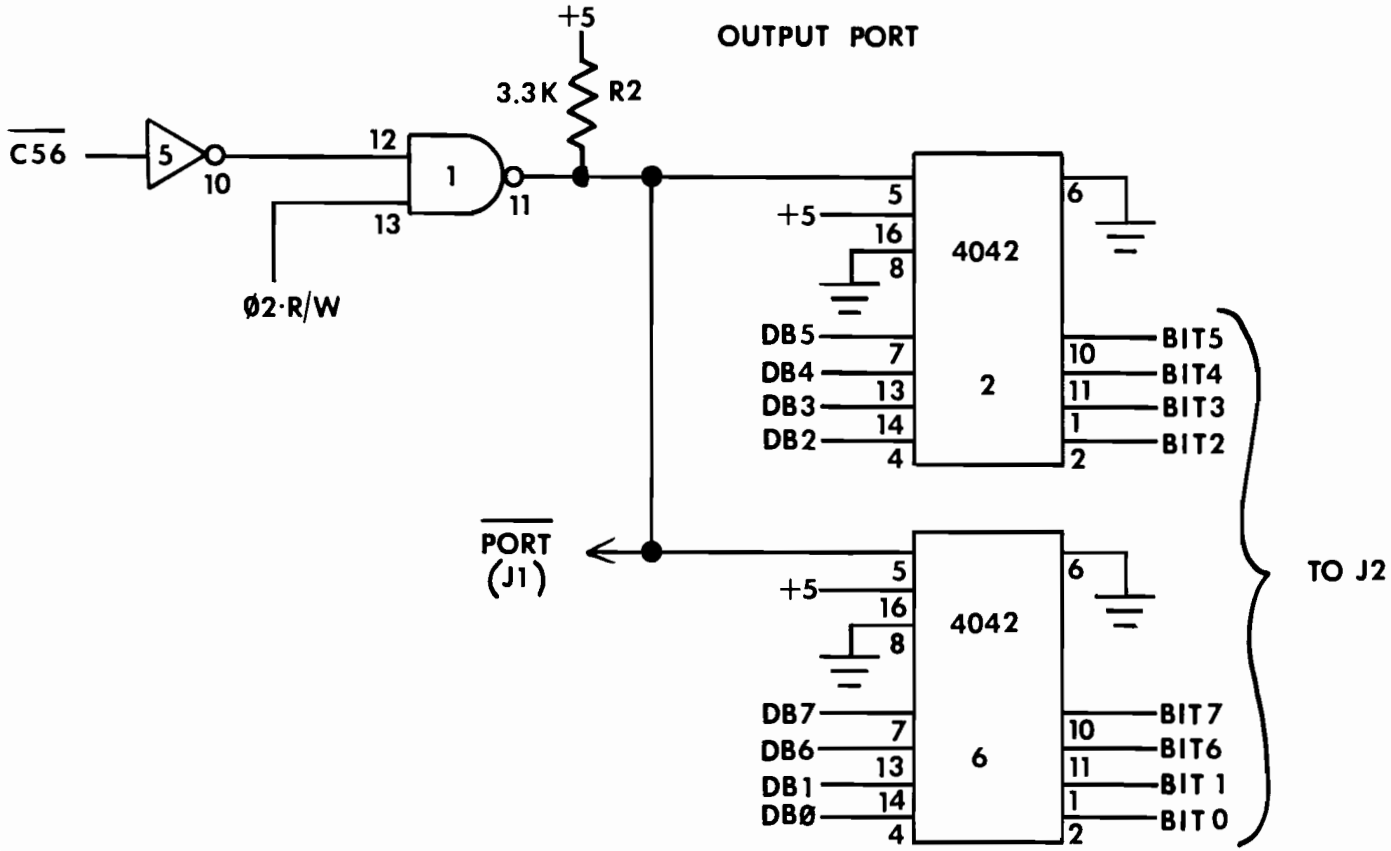

OUTPUT PORT
+5
3.3K
R2
C56
5
10
12
13
1
11
Ø2·R/W
+5
5
16
8
4042
6
DB5
DB4
DB3
DB2
7
13
14
4
2
10
11
1
2
BIT5
BIT4
BIT3
BIT2
PORT
(J1)
+5
5
16
8
4042
6
DB7
DB6
DB1
DBØ
7
13
14
4
6
10
11
1
2
BIT7
BIT6
BIT 1
BIT 0
TO J2

FIGURE 9

The PAiA 8700 Self-Test Micro-Diagnostic T.M.

There is a significant test feature built into the PAIA 8700 which, while simple in concept, provides an exceptionally powerful tool for spotting a number of potential faults associated with the Grand Buss architecture common to micro-computers. Two small circuitry details are involved in implementing this feature:

1) A means by which devices connected to the CPU's data buss may be disconnected and allowed to float.
2) A means by which a no-operation (NOP) instruction is forced onto the data buss.

Together, these two things cause a properly assembled and functioning 8700 board to operate in a very special manner.

The processor, on being reset, will fetch the first instruction from the memory location specified by the reset vector. Since all sources of data have been isolated from the data buss, the only source of instructions to the processor is from the combination of the data buss pull-up resistors R39 - R44 and the three diodes D3 - D5. The diodes clamp data buss lines D0, D2, and D4 to ground producing the binary pattern 11101010 (EA in hex) on the data buss. EA is a NOP instruction.

The processor's response to a NOP is to increment the address buss to the next address and fetch the instruction that it finds there, which is of course again a NOP. The address lines increment again and fetch the NOP, etc.

The overall result is that the address lines (all 12 of them) count in a normal binary sequence. This in turn allows for easy checking of the address lines which are operating in an easily verified and predictable manner as well as exercising all of the address decoding circuitry, making for easy checking of the various chip select lines to output ports, memory locations, etc. to see that this portion of the circuitry is operating properly.

Using the Self-Test

1) Remove all RAM (IC22 - IC25; IC30 - IC33) and ROM (IC16 - IC19) from their sockets.

2) Close the circuit board jumper S2 either by putting it in place, or, if already installed but severed, by soldering the cut ends together. This step ties the cathodes of the three diodes D3 - D5 to ground causing them to forward bias and hold the data lines D0, D2 and D4 low.

3) If the jumper S1 is in place, cut it so that no connection is made and isolate the two ends from one another. Also, tie the end of the jumper designated by the arrow on the circuit board to the +5 volt power supply line. A clip lead may be used here and the best place to pick up the +5 volts is at the left end of R5. These steps isolate the data buss by breaking the emitter leads of the transistors Q0 - Q7 and assures isolation by reverse-biasing these devices.

4) Apply power to the processor. And note that since only the ROMs require the -9 volt supply, this voltage does not have to be provided for these tests. On the other hand, it won't hurt to have it there either - whichever is easier. When the power is applied the displays should immediately light with some random digits. This is of course a quick check that the +5 volt supply is active and that there is not a direct short across the supply lines somewhere. The 9368 Display Drivers will quickly become uncomfortably warm to the touch. This is normal.

5) Reset the CPU by using a clip lead or other temporary jumper to momentarily ground the RESET line. For the purposes of these tests the RESET line is most easily accessed at pin 14 of the expansion connector J7 and ground can be picked up at the circuit board jumper S2. Since some malfunctions can cause the processor to "lock-up" (recieve an instruction that causes paralysis of the address and data busses - who knows what it's up to internally) it would be wise to have your temporary RESET switch handy during the entire procedure.

6) Check the Ø2 clock signal at pin 12 of the expansion connector J7 to see that it is:

a) present
b) swinging between essentially +5 v. and ground
c) has a period of approximately 2.5 micro-seconds ± 20%
d) has a duty factor somewhere between 30% and 70% - exact duty factor is not critical

7) Check, in sequence, the 12 address lines AB0 - AB11. These lines are most easily accessed at the expansion connectors J7 and J8 as shown below:

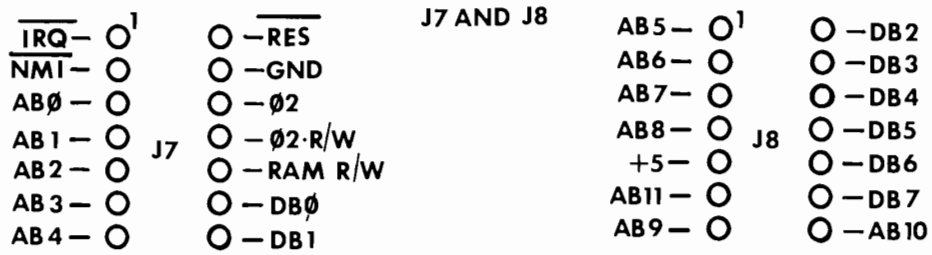


Figure 1

NOTE that AB9, AB10 and AB 11 are out of sequence at these connectors. When checking these waveforms you should observe that:

a) they are perfectly square (50% duty factor)
b) they will be slightly rounded on the rising as in figure 2:

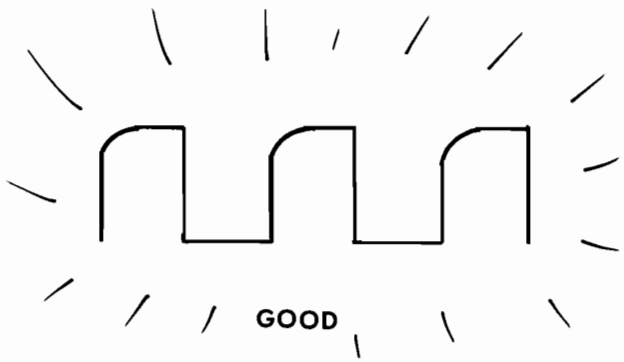
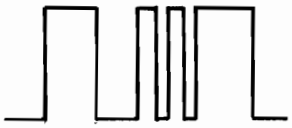


Figure 2

c) each of the lines in ascending sequence, starting with AB0, is exactly half the frequency of the preceding line in the sequence.
d) each line should swing from essentially supply to ground.
e) both the "1" state and the "0" state of the lines should be relatively free of glitches. (not more than 200 - 300 millivolts)

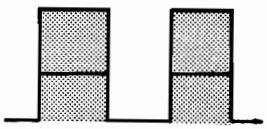
There are two problems that you will most likely spot with this test. First, that the lines do not toggle symmetrically but switch in bursts, which at low oscilloscope sweep rates will look like this:



BAD

Figure 3

which could be an indication either of a malfunctioning component or a defective conductive trace or solder joint on the circuit board.
Second, one or more of the address lines may not swing fully between supply and ground, which, again at low sweep rates, will look like this:



BAD

Figure 4

and could also be caused by a defective component but is more probably the result of a short between adjacent conductors on the circuit board. Even more specifically, this condition can most often be traced to a short between the malfunctioning address line and either a data line or one of the chip select lines. More information on these conditions can be gained with the rest of the tests.

8) Check the chip select lines individually. There are seventeen of them;

4 lines going to the RAM chips (check at pin 13 of each of the RAM locations IC22 - 25)

4 lines going to the ROM chips (pin 14 of each of the ROM locations IC16 - 19

1 line ($\overline{CS9}$) present at pin 11 of IC12

4 lines ($\overline{CS0}$, $\overline{CS1}$, $\overline{CS2}$, and $\overline{CS3}$) present at pins 11, 3, 4 and 10 respectively, of IC13

4 lines ($\overline{CS4}$, $\overline{CS5}$, $\overline{CS6}$ and $\overline{CS7}$) present at pins 10, 11, 3 and 4 respectively, of IC9

At each of these points you should see the same thing; a narrow negative pulse on the order 1 to 2 milli-seconds in duration. Like this:

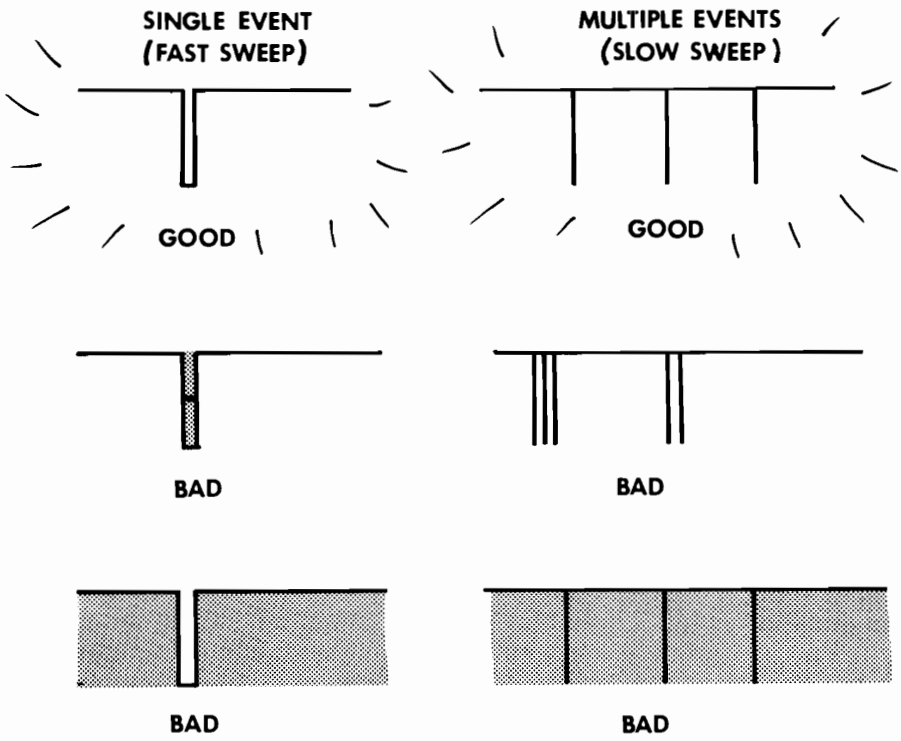


Figure 5

You should be sure that these pulses:

a) occur at a constant repetition rate
b) occur at constant time intervals
c) swing essentially from supply to ground
d) do not have faster switching events happening inside the pulse (with the exception of $\overline{CS\emptyset}$ - CS7, which will have switching inside the negative-going pulse)

9) If you have checked and successfully verified that all of the points above are as they should be, this last becomes academic. Check the data buss lines which are accessible at the expansion connectors J7 and J8. These lines should be totally static (not switching at all) and should be at the logical levels which follow:

DB0 - 0
DB1 - 1
DB2 - 0
DB3 - 1
DB4 - 0
DB5 - 1
DB6 - 1
DB7 - 1

Even if the lines are static, check to make sure that they are all within 500 millivolts of either supply or ground.

Successful completion of all of the foregoing procedures is a very strong indication that the 8700 Computer is functioning properly and will continue to do so when mated with its companion keyboard.

BEFORE LEAVING THIS SECTION BE SURE TO RESTORE THE MACHINE TO ITS ORIGINAL CONFIGURATION. Put RAM and ROM back in place, solder the ends of the jumper S1 back together and cut jumper S2 being sure to fold the ends back so they do not touch each other or surrounding circuitry.

If any of these tests failed, you must begin trouble shooting. It would be nice if we could cover all of the things which can potentially die or short to one another. We can't. Trouble shooting a system of this level of complexity is most readily accomplished in much the same manner as methods employed by medical diagnosticians:

a) consider all the symptoms (complete all tests)
b) postulate a defect that would produce some or all of the observed symptoms
c) check your hypothesis
d) probably go back and try again.

WE CAN HELP and are happy to do it. If you have any difficulties with the tests in this section, write or call:

PAIA Electronics, Inc.
1020 Wilshire Blvd.
Okla. City, OK, 73116
(405) 843-9626 9:00 am - 5:00 pm CST

Please supply information relative to the results of your tests: which lines looked OK, which didn't, etc.