CA-23

EPROM

Programer Manual

Jan. 1981

CONTENTS

CONTENTS

Introduction to the CA-23

In 1978, a new family of EPROM's were introduced by integrated circuit manufacturers. These parts are quickly becoming the industry standard for several reasons. Previous families of EPROM's required multiple voltages during the read mode and in the program mode. The multiple supply requirement of the old families almost always required an additional system power supply with the single purpose of support for these EPROM's. The new family of EPROM requires +5V only (system supply) in the read mode, thus, eliminating the support power supply. Since the introduction of these EPROM's, the family has grown at an exciting rate becoming not only larger in density to save board "real estate" and parts count but also decreasing the cost per bit. Also, as new manufacturing processes are developed yielding greater density, faster access times, and higher die yield, the die for the early generation of the family are being relegated to the new processes.

The CA-23 board supports this new family of EPROM providing a convenient method of programming and testing them. The program/data to be written to the EPROM may originate from several sources. A master EPROM could be used to make copies or code stored in memor disk may be used.

The CA-23 is designed for the new +5V family only and may not be used to program the older generation of EPROM.

Software Operation

After booting and displaying which version of the EPROM PROGRAMMER SOFTWARE is on-line, "PROG", which is the "basic" interactive program will be run. This program is used to prompt the operator and guide him through the various levels of the program. The actual programming is carried out by a machine code device handler ("HANDLR" on the directory).

The outermost level of the program is the program entry/re-start point. At this time, the CA-23 will be tested for power ON/OFF. If the power is on the operator will be asked to:

SLIDE THE ON/OFF SWITCH TO OFF

This is done to insure that no EPROM's will be inserted or removed while power is applied which could possibly damage the device. The power switch removes "all" power from the CA-23 including the programming voltage. If the power is turned off or is already off, the message

SLIDE THE PROGRAM/READ SWITCH TO READ
HIT RETURN WHEN READY?

will be displayed. The program/read switch is a hardware "fail-safe" switch used to enable/disable the programming voltage (+25V). Slide it to READ and depress carriage return (CR) when ready to continue. If the switch is already in READ, simply depress the CR key.

A list of EPROM's that the program can handle is now displayed. Select the EPROM you are working with and type the appropriate number after the prompt:

PART NUMBER YOU WISH TO WORK WITH?

Other manufacturers' parts can be cross-referenced to these part numbers if the EPROM you wish to program is not listed.

The CA-23 board incorporates a switch to reconfigure connections to the two major pin-out types of +5V only EPROM. This switch is called the TYPE switch which must be correctly set before power is applied to the CA-23 and the EPROM. At this time, the program will prompt the operator to set the switch to either position A or B:

```
SLIDE THE (TYPE) SWITCH TO POSITION (  )
HIT RETURN WHEN READY?
```

When done or if the switch is already in the correct position, depress CR.

There are six major modes of operation. These are now displayed:

```
1 = DUPLICATE
2 = VERIFY
3 = LIST
4 = PROGRAM FROM MEMORY
5 = LOAD MEMORY
6 = EDIT/SAVE MEMORY

WHAT DO YOU WISH TO DO?
```

One of the major modes is now selected by typing the appropriate number and depressing CR.

MODE 1 = DUPLICATE

This mode is used to make DUPLICATE copies of a master EPROM. The master EPROM and the EPROM to be programmed must be of the same part number or cross-referenced as previously discussed. If a copy EPROM is to be programmed and the master is a different type, then MODE 5 should be used to LOAD MEMORY from the master

and MODE 4 should be used to program the copy.

SUPPRESS LISTING DURING THIS ACTION?

Answer yes (Y) or no (N). "N" will instruct the machine code programmer routines to list the location and data being read or programmed. Answering "Y" will suppress the listing (unless an ERROR is encountered), thereby decreasing the amount of time required for an operation to be completed. If an ERROR is encountered, the location and data will be listed regardless of this software switch.

SLIDE THE TYPE SWITCH TO POSITION ( )
LOAD THE MASTER EPROM IN THE MASTER SOCKET
LOAD THE COPY EPROM IN THE COPY SOCKET
SLIDE THE ON/OFF SWITCH TO ON

A double check is now made to insure that the TYPE switch is in the correct position, and then the EPROM's should be loaded into the correct sockets. The load lever (of the sockets) must be in the closed position before activating the power on switch. When the power on switch is activited, the program will prompt the operator:

SKIP BLANK VERIFY TEST (Y OR N)?

If "N" is answered, the program will test the copy EPROM to insure that it is blank (all $FF) before attempting to program it. In all cases, "N" should be the response unless an EPROM is being edited.

SLIDE THE PROGRAM/READ SWITCH TO PROGRAM

This will apply the (+25V) programming voltage to the software controlled Tri State programming signal which applies the (+25V)

only when required to program the EPROM.

The program will now proceed to transfer the master data to the copy. Any further messages are self-explanatory. The sequence is: check copy for all $FF, load master data in memory, program copy with master data, verify copy = master.

MODE 2 = VERIFY

SUPPRESS LISTING DURING THIS ACTION?

This is the same as explained in MODE 1.

1 = VERIFY COPY TO MEMORY
2 = VERIFY COPY TO MASTER
3 = VERIFY MASTER TO MEMORY

The VERIFY MODE compares an EPROM to another EPROM or memory. A message will be generated to indicate an ERROR if a discrepancy exists and will list the location and data.

MODE 3 = LIST

1 = FROM MASTER SOCKET
2 = FROM COPY SOCKET
3 = FROM MEMORY

When one of the above three is selected, the program will either display memory immediately (#3), or prompt the operator to load the master (#1) or copy (#2) EPROM, and then list the data. In either case, all three will always be listed although they may be invalid (not installed). This is done so that if the operator wished to list all three, (#1) or (#2) could be answered, then the master and copy EPROM's loaded, and all the data for each, including memory, would be listed.

MODE 4 = PROGRAM FROM MEMORY

SUPPRESS LISTING DURING THIS ACTION?

This is explained in MODE 1.

The operator is then asked to load the copy EPROM in the copy socket and apply power and then:

SLIDE THE PROGRAM/READ SWITCH TO PROGRAM
SKIP BLANK VERIFY TEST (Y OR N)?

This is explained in MODE 1. The program will now proceed to program the copy to equal the memory data and then automatically go into the VERIFY MODE.

MODE 5 = LOAD MEMORY

SUPPRESS LISTING DURING THIS ACTION?

This is explained in MODE 1.

1 = FROM MASTER SOCKET
2 = FROM COPY SOCKET
3 = FROM DISK

This mode loads the data memory from one of the three sources. Sources (#1) and (#2) are self-explanatory. Source (#3) is from a disk file (one of seven) which has previously been saved by using the MODE 6 SAVE command. A full 8K will always be loaded regardless of the size of the EPROM as is explained in MODE 6.

MODE 6 = EDIT/SAVE MEMORY

This mode is used to EDIT the memory buffer or SAVE the memory buffer in one of seven files that are available. The computer responds with:

BASE ADDRESS(HEX)?

The computer always stores the EPROM data in a predetermined memory buffer area beginning at $5ØØØ. This actual buffer address could be confusing to the operator. Consider the example where the actual EPROM to be programmed was to be memory mapped in the destination computer to begin at $FCØØ and end at $FFFF (1K EPROM). Now assume that the EPROM data is already resident in the memory buffer area and the user wishes to edit the memory. Normally, an assembly listing would be used to determine the patch(s) and the hexadecimal address(s) of the change(s) as found in the listing. Assume that the EPROM listing begins at $FCØØ and it is desired to change location $FD12. At this point, $FCØØ should be typed. Internally, the computer will now associate $FCØØ with the first location in the buffer which is $(5ØØØ) and $FCØ1 would equal $5ØØ1. If a non-hex character is typed the input will be aborted, or, if an invalid offset is detected such as specifying the base address of a 1K EPROM as $FDØØ.

(O)PEN,(L)IST,(P)RESET,(S)AVE,E(X)IT?

OPEN is used to open a location for change. LIST is used to view memory or list on a printer, PRESET sets memory (RANGE) to an input value, SAVE saves memory to a disk file and EXIT leaves the EDIT MODE and returns to the outermost level of the program.

To EDIT location $FD12, as mentioned earlier, type "O". The program will finish typing open and ask for the location (?). Type FD12. On the left-hand side of the screen, FD12 will be printed, then a space, and then the data in that location is presented. At this time, the data to be entered should be typed in

hexadecimal. If the data is accepted, the location will be changed to the new value and the next location will be opened for change. If a non-hex character is detected, the location will be unmodified and reopened for change. To advance one location without changing the currently open location, hit LINE FEED. To back up one location type (^) which is SHIFT-N on some keyboards. To exit, hit CR.

The other modes are self-explanatory. LIST and PRESET ask for a RANGE to work with and PRESET will ask for data to fill memory with. The SAVE function will save the memory buffer in one of seven files. The full 8K byte buffer is always saved regardless of the size of the EPROM or the actual memory buffer size. In systems with 24K bytes of memory, the actual buffer size is only 4K bytes.

Data Files

Seven data files are resident on the programmer disk. These are maintained as FILE1 through FILE7. On five inch systems, two files are maintained. Each file consists of four tracks, each track storing 2048 bytes of the 8192 byte file. The 2048 limit on eight inch system file tracks retains compatability with five inch systems (eight inch systems are actually capable of saving 3072 bytes per track). The software SAVE command always saves the full 8192 byte RAM buffer ($5000-$6FFF) on the four tracks of the file regardless of the byte size of the EPROM or the actual buffer size. This is also true of the LOAD/FILE commands. That is, when loading the RAM buffer using the LOAD command, the full 8K RAM buffer is loaded from the four tracks in the file.

The SAVE (SA) command stores the buffer by effectively simulating the operating system SA command. If FILE1 (eight inch) were being saved, the following sequence occurs:

```
A*SA 33,1=5000/8
A*SA 34,1=5800/8
A*SA 35,1=6000/8
A*SA 36,1=6800/8
```

This same function could be performed manually by exiting the programmer (this can be done by depressing CR in response to any of the prompts) to the BASIC IMMEDIATE MODE and typing EXIT. The SA command may now be used as explained in detail in the OS-65D manual. The buffer is loaded in a similar manner.

```
A*CA 5000=33,1
A*CA 5800=34,1
A*CA 6000=35,1
A*CA 6800=36,1
```

The file handler of the programmer software was implemented in this fashion so that easy manipulation of files is maintained. An example could be that the user has just finished assembling a program (1K bytes) to $8C00-$8FFF and wishes to program an EPROM with this data. The assembler should be exited, "EX".

A*

The assembler disk is now removed and the programmer disk inserted in the floppy drive. To SAVE this data in FILE1:

A*SA 33,1=8C00/8

Now reset the computer and boot up the programmer disk. Use the LOAD MEMORY function to load FILE1 and then the PROGRAM FROM MEMORY function to program the EPROM copy.

Before programming the copy, it may be wise to use the EDITOR LIST command to insure that the correct data has been transferred by these sequences.

Another example might be the case where two 1K byte EPROM's are to be substituted by one 2K byte EPROM. Using the programmer, load the first EPROM to memory, exit to the IMMEDIATE MODE, and then exit to the operating system.

A*

Tracks 64-76 (eight inch) have been reserved for the following types of operations:

The first EPROM data (1K byte) is still at $5000-$53FF and could be saved by:

A*SA 64,1=5000/4

Now type RE BA to return to BASIC, run and LOAD MEMORY with the second 1K byte EPROM, then enter the operating system again and

```
A*SA 65,1=5ØØØ/4
```

The two (1K byte) EPROM's (data) are now saved on tracks 64 and 65. To load the RAM buffer with the 2K byte data:

```
A*CA 5ØØØ=64,1
A*CA 54ØØ=65,1
A*RE BA
RUN
```

The 2K byte EPROM is now programmed using the function PROGRAM FROM MEMORY and the buffer could be saved using the EDIT/SAVE command.

Power Requirements

The CA-23 requires two voltages for full functional operation:

+5V/±5% @ 500MA

+25.6V% @ 500MA

The +25V is required to program an EPROM and is not necessary for read only operations. Most adjustable +24V power supplies can be adjusted to obtain this voltage. Some users have successfully used the +24V supply in the eight inch drive cabinet for this purpose, adjusting it to +25V when programming a part and readjusting to +24V when complete.

The +5V requirement can be satisfied by either a dedicated accessory supply or by connection to the computer system power supply.

There are four zero insertion sockets supported on the CA-23. There are 24-pin and 28-pin master and copy sockets. The master sockets are used to LOAD MEMORY from a master EPROM, VERIFY it or DUPLICATE it. The master socket can never have the voltage required for programming applied to it. Therefore, the master EPROM can't be inadvertently altered by operator error or other means when it is inserted only in the master socket. The copy socket is used for programming EPROM's.

SW1 is used to remove all voltages from the CA-23, so that the EPROM's may be safely inserted or removed from the zero insertion sockets.

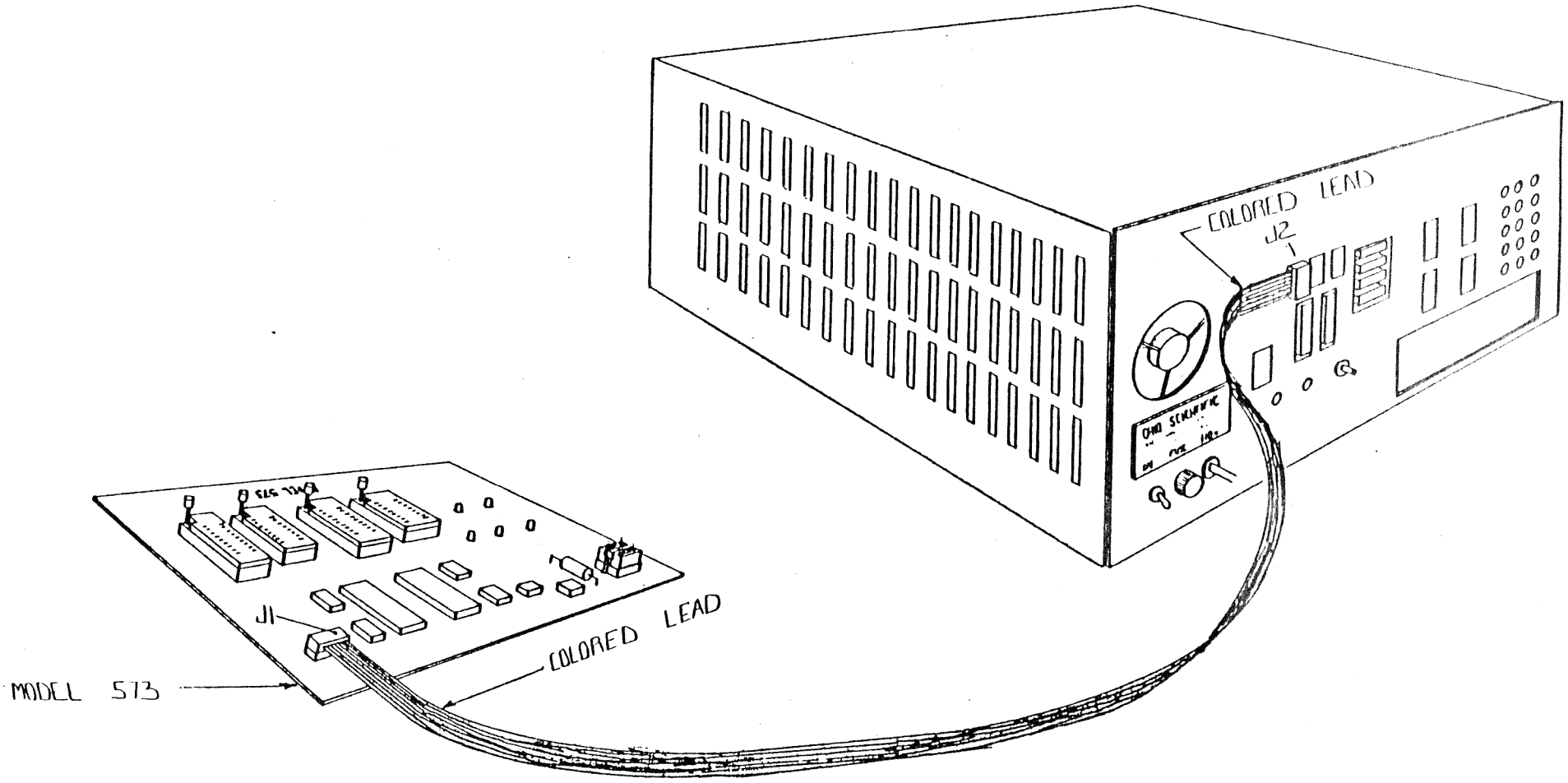
SW2 is used to mechanically inhibit or enable program voltages to the EPROM copy sockets.

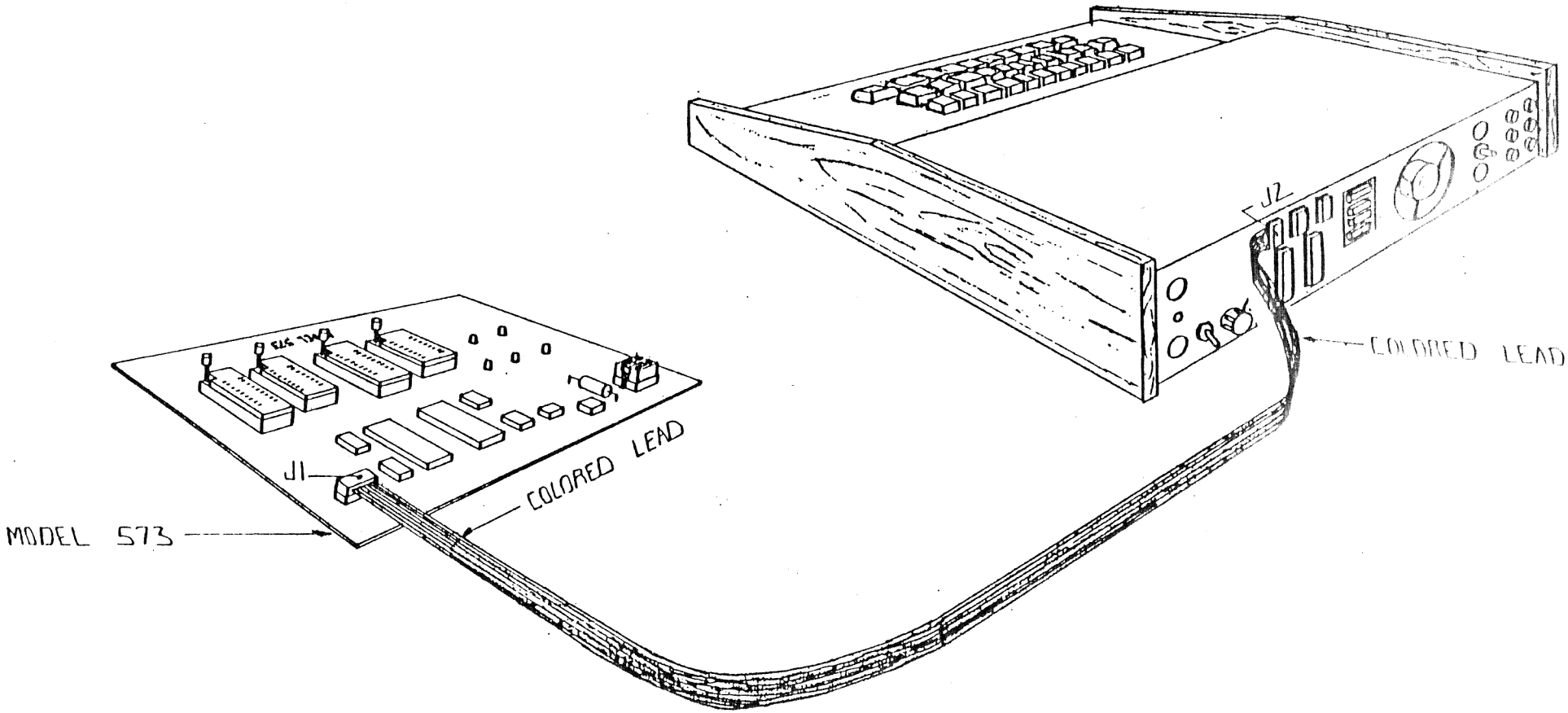
SW3 routes signals to the EPROM sockets in one of two manners depending on the type of EPROM in use.

U1D applies addresses (AØ-A7) to the sockets and either provides write data or read data (bidirectional) depending on the mode.

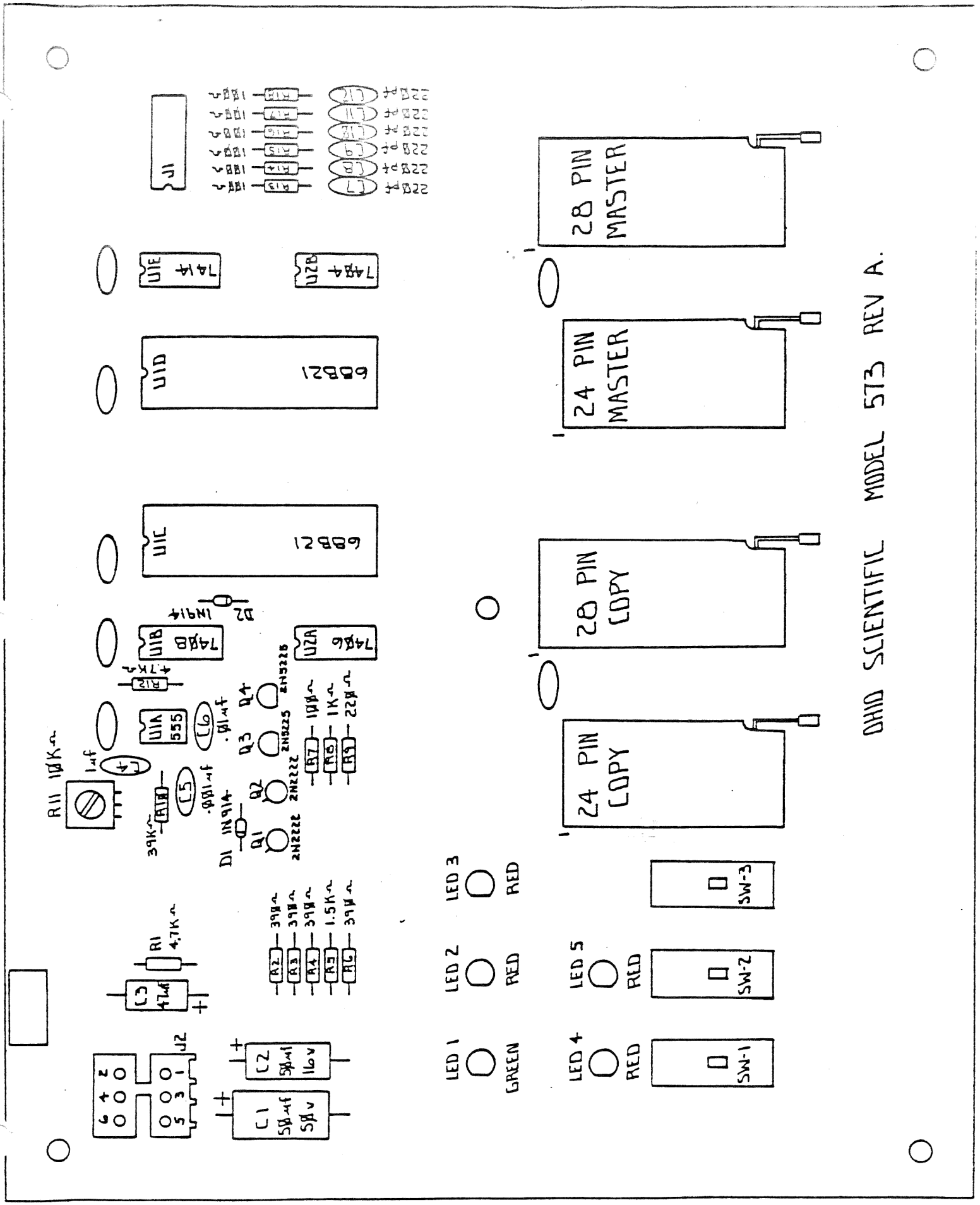
U1C applies addresses (A8-A9) and (A1Ø-A14) where required. U1D also controls several other functions.

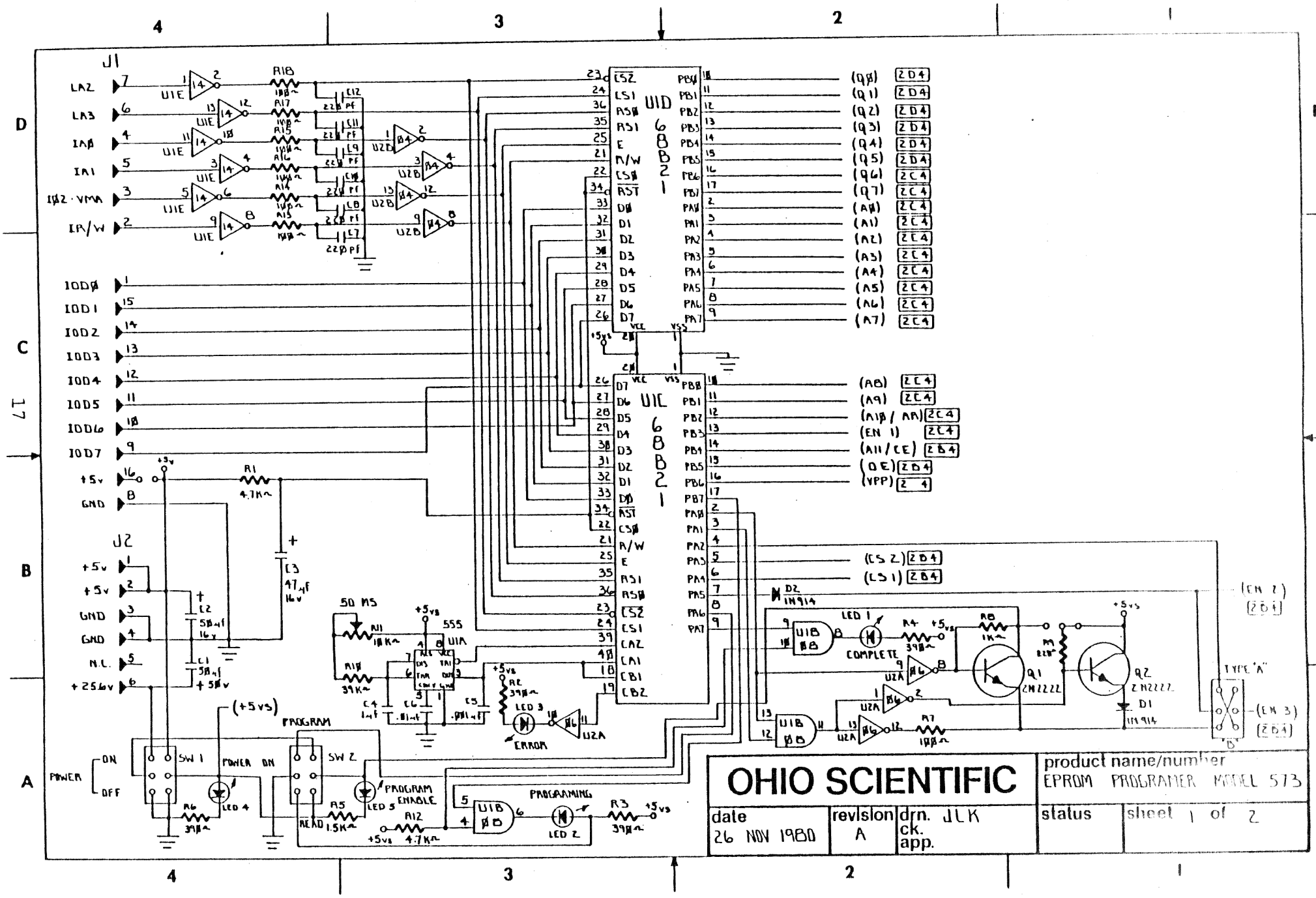
The signal VPP(SW) is capable of three levels: +25V, +5V,+.6V. These voltages are required to perform programming and reading of the various types of EPROM's by the CA-23. Q1 sources the +25V, Q2 sources +5V through D1 and U2A pin 12 sinks to ground. U1B pins 13, 12, and 11 logically prevents contention of the three devices.

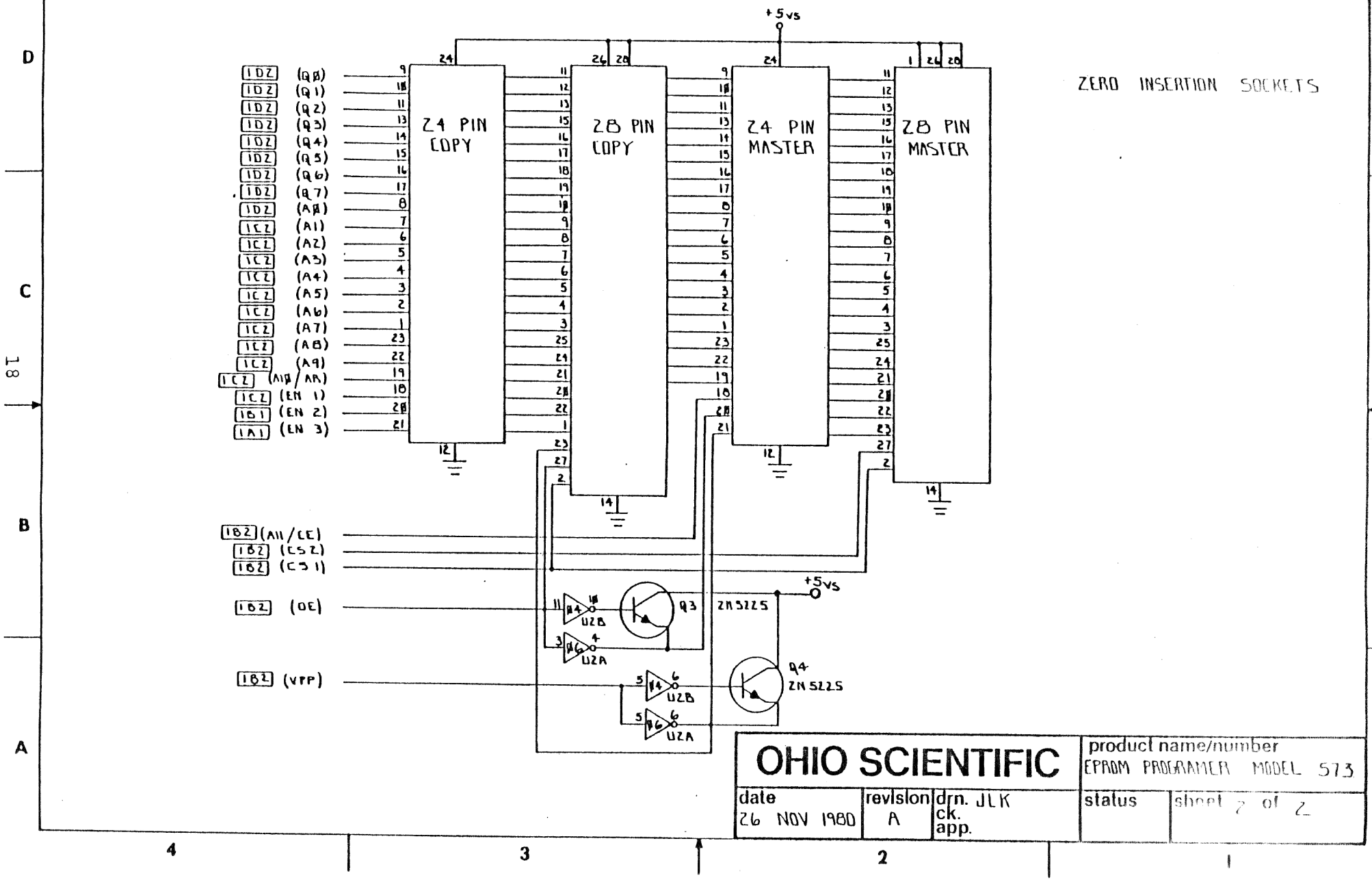

COLORED LEAD
J2
1
J1
COLORED LEAD
MODEL 573


J2
COLORED LEAD
J1
COLORED LEAD
MODEL 573

ALL UNMARKED CAPACITORS ARE .1μf BYPASS CAPACITORS




J1
J2
U1D 6821
U1E 6821
U1A 555
U1B
U2A
U2B
U2C
U2D
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10/A8
A11/CE
EN 1
OE
VPP
CS2
CS1
R/W
E
RS1
RS0
RST
CA2
CA1
CB1
CB2
D0
D1
D2
D3
D4
D5
D6
D7
IOD0
IOD1
IOD2
IOD3
IOD4
IOD5
IOD6
IOD7
+5v
GND
N.C.
+25.6v
50 MS
PROGRAM
POWER
ON
OFF
SW 1
SW 2
POWER ON
PROGRAM ENABLE
READ
PROGRAMMING
ERROR
COMPLETE
LED 1
LED 2
LED 3
LED 4
LED 5
D1 IN914
D2 IN914
Q1 2N2222
Q2 2N2222
TYPE "A"
(EN 2)
(EN 3)
(+5VS)
OHIO SCIENTIFIC
product name/number
EPROM PROGRAMER MODEL 573
date
26 NOV 1980
revision
A
drn. JLK
ck.
app.
status
sheet 1 of 2



OHIO SCIENTIFIC			product name/number EPROM PROGRAMMER MODEL 573	
date 26 NOV 1980	revision A	drn. JLK ck. app.	status	sheet 2 of 2

```
.A

 10                    ;  EPROM PROGRAMER DRIVER V1.1
 20                    ;  USE WITH BASIC PROGRAM
 30                    ;  POLLED OR SERIAL SYSTEM
 40                    ;
 50 0000=          NUS     =$0000          NOT USED
 60 0001=          P1      =NUS+1          P1=0 IF PART=1 ELSE =1
 70 0002=          P2      =P1+1
 80 0003=          P3      =P2+1
 90 0004=          P4      =P3+1
100 0005=          P5      =P4+1
110 0006=          P6      =P5+1
120 0007=          P7      =P6+1
130 0008=          P8      =P7+1
140 0009=          P9      =P8+1           P9=0 IF PART=9 ELSE =9
150 000A=          ADDL    =P9+1           EPROM LOW ADD
160 000B=          ADDH    =ADDL+1         EPROM HIGH ADD
170 000C=          DCPY    =ADDH+1         EPROM COPY READ/DATA
180 000D=          DMSTR   =DCPY+1         EPROM MSTR READ/DATA
190 000E=          PN2C    =DMSTR+1        2758 LOOP COUNT
200 000F=          QREG    =PN2C+1         TEMP STORE
210 0010=          MR      =QREG+1         TEMP
220 0011=          NR      =MR+1           TEMP
230 0012=          LT5R61  =NR+1           IF<50R=6THEN>0
240 0013=          BUFFAD  =LT5R61+1       BUFFER ADD COUNT
250 0014=          STRG    =BUFFADD+1
260 0015=          ECPY    =STRG+1
270 0016=          DISCPY  =ECPY+1
280 0017=          DISPG   =DISCPY+1
290 0018=          VIDEO   =DISPG+1
300                        ;
310                        ;EXTERNAL LINKS
320 2340=          INCH    =$2340          GET ONE ASCII CHARACTER
330 2343=          OUTCH   =$2343          OUTPUT ACC. TO SCREEN
340 2D73=          STROUT  =$2D73          OUTPUT STRING
350 2D6A=          CRLF    =$2D6A          CAR.RET LINE FEED
360                        ;
370                        ;$5000 UP = DATA BUFFER
380                        ;CONTROL BUFFER BASIC TO M.CODE
390 5000=          BUFF    =$5000
400 4FFF=          FROM    =BUFF-1         BUFFER STARTS AT PAGE
410 4FFE=          TO      =BUFF-2         BUFFER ENDS   AT PAGE
420 4FFD=          SUBACT  =BUFF-3         SUB-ACTION
430 4FFC=          ACT     =BUFF-4         MAJOR ACTION
440 4FFB=          SKIP    =BUFF-5         SKIP BLANK TEST IF ONE
450 4FFA=          PARTN   =BUFF-6         PART # AS PER BASIC TABLE
460 4FF9=          PAGES   =BUFF-7         # OF PAGES
470 4FF8=          PROG    =BUFF-8         0=READ/1=PROG/2=INIT/3=IN;4=OUT
480 4FF7=          BANK    =BUFF-9         BANK OF 2758 THAT'S GOOD
490 4FF6=          FR      =BUFF-10        LOAD FROM COPY/MASTER
500 4FF5=          VE      =BUFF-11        VERIFY TRIANGULATE
510 4FF4=          ERROR   =BUFF-12        1=ERROR
520 4FF3=          CHAR    =BUFF-13        PASS CHAR
530 4FF2=          HOASC   =BUFF-14        0=HEX,1=ASCII
540 4FF1=          LTEMP   =BUFF-15        TEMP
550 4FF0=          HTEMP   =BUFF-16
560 4FEF=          SLIST   =BUFF-17        SUPRESS LISTING IF=1
570 4FEE=          OFFSET  =BUFF-18        DISPLAY OFFSET
580 4FED=          COLD    =BUFF-19        BASIC/IF<>85 CALLS THIS PRG.IN
590                        ;
600                       ; PROGRAMER BOARD PORT ASSIGNMENTS
```

```
 610 C704=          ADD     =$C704       BEGIN OF PIA REG'S
 620 C708=          PA      =$C708
 630 C70A=          PB      =$C70A
 640 C706=          DR      =$C706       B1-DIRECTIONAL DATA
 650                ;
 660                ;
 670                        ; ENTRY FROM BASIC-PARAMETERS
 680                        ; PASSED IN BUFFER
 690                        ; CODE MUST STOP BEFORE BUFF-19
 700 4B00                    *=$4B00        BASIC STOPS AT $4CFF
 710 4B00 20534E   ENTRY    JSR RSTADD      RESET ACTUAL EPROM  ADDRESS
 720 4B03 ADF84F            LDA PROG        GET FUNCTION
 730 4B06 C903              CMP #3          EDITOR INPUT?
 740 4B08 F06E              BEQ EDT         YES
 750 4B0A C902              CMP #2          IF ITS A 2 THEN INITIALIZE
 760 4B0C F067              BEQ PTINIT      THE BOARD
 770 4B0E C904              CMP #4          IS IT HEX OUT?
 780 4B10 F069              BEQ HXO         YES
 790 4B12 A900              LDA #0          SET P1-P9
 800 4B14 AA                TAX             TO 1-9
 810 4B15 8DF44F            STA ERROR       RESET ERROR FLAG
 820 4B18 18                CLC
 830 4B19 9500     RST0     STA 0,X
 840 4B1B 6901              ADC #1
 850 4B1D E8                INX
 860 4B1E E00A              CPX #10
 870 4B20 D0F7              BNE RST0
 880 4B22 CA                DEX
 890 4B23 B500     NXTR     LDA 0,X
 900 4B25 CDFA4F            CMP PARTN        FIND PART AND SET THAT
 910 4B28 0004              BNE NPRT         P# TO 0
 920 4B2A A900              LDA #0
 930 4B2C 9500              STA 0,X
 940 4B2E CA        NPRT    DEX
 950 4B2F E000              CPX #0
 960 4B31 D0F0              BNE NXTR         DONE
 970 4B33 206A2D            JSR CRLF
 990 4B36 ADF84F            LDA PROG
1000 4B39 F009              BEQ READ        NO
1010 4B3B 20D84F            JSR WPROG       WRITE "PROGRAMMING"
1020 4B3E 206A2D            JSR CRLF
1030 4B41 4C124C            JMP PROGR       GO PROGRAM
1040 4B44 20732D   READ     JSR STROUT      WRITE
1050 4B47 52                .BYTE 'READING',0
1050 4B48 45
1050 4B49 41
1050 4B4A 44
1050 4B4B 49
1050 4B4C 4E
1050 4B4D 47
1050 4B4E 00
1060 4B4F 206A2D            JSR CRLF
1070 4B52 A502              LDA P2          IS IT 2758?
1080 4B54 D009              BNE N2758       NO
1090 4B56 ADF74F            LDA BANK        YES,WHICH BANK IS GOOD?
1100 4B59 F004              BEQ N2758
1110 4B5B A904              LDA #4          UPPER BANK
1120 4B5D 850B              STA ADDH        ADJ. ADDH
1130 4B5F 209F4E   N2758    JSR INITAD
1140 4B62 20754C   LOOP     JSR HANDLR      GO GET DATA
1150 4B65 ADFC4F            LDA ACT
1160 4B68 C901              CMP #1          DUPLICATE?
1170 4B6A D012              BNE NDUP        NO
1180 4B6C A50D              LDA DMSTR       YES
```

```
1190 4B6F A000            LDY #0
1200 4B71 9113            STA (BUFFAD),Y PUT MSTER DAT IN BUFF
1210 4B73 4CA24B          JMP NLOAD
1220 4B76 4CD24E  PTINIT  JMP INIT        GO INITIALIZE BOARD
1230 4B79 4C244E  EDT     JMP INHEX       GET HEX CHAR IN
1240 4B7C 4C4C4E  HXO     JMP HXOUT       OUTPUT ONE CHAR IN HEX
1250 4B7F ADFC4F  NOUP    LDA ACT
1260 4B82 C905            CMP #5          LOAD MEMORY?
1270 4B84 D01D            BNE NLOAD       NO
1280 4B86 ADF64F          LDA FR          YES
1290 4B89 C901            CMP #1          FROM MSTR?
1300 4B8B D009            BNE NFR1        NO
1310 4B8D A50D            LDA DMSTR       YES-STUFF BUFF W/MSTR
1320 4B8F A000            LDY #0
1330 4B91 9113            STA (BUFFAD),Y STORE IN BUFFER
1340 4B93 4CA24B          JMP NLOAD
1350 4B96 ADF64F  NFR1    LDA FR
1360 4B99 C902            CMP #2          FROM COPY?
1370 4B9B D006            BNE NLOAD       NO
1380 4B9D A50C            LDA DCPY        YES
1390 4B9F A000            LDY #0
1400 4BA1 9113            STA (BUFFAD),Y STUFF BUFF W/COPY
1410 4BA3 ADFC4F  NLOAD   LDA ACT
1420 4BA6 C901            CMP #1          DUPLICATE?
1430 4BA8 F004            BEQ SKIPTQ      YES
1440 4BAA C904            CMP #4          PROG. MODE BLANK READ
1450 4BAC D018            BNE PNT         NO
1460 4BAE ADFB4F  SKIPTQ  LDA SKIP        SKIP BLANK TEST?
1470 4BB1 C901            CMP #1          WHICH CHECKS FOR ALL FF'S
1480 4BB3 F009            BEQ PANT        YES
1490 4BB5 A50C            LDA DCPY        GET CPY DTA
1500 4BB7 C9FF            CMP #255        CHECK ALL ERASED
1510 4BB9 F00B            BEQ PNT
1520 4BBB 4C924F          JMP BLNKER      GO INFORM ERROR
1530 4BBE ADFC4F  PANT    LDA ACT         IF PROG FROM MEMORY
1540 4BC1 C904            CMP #4          AND SKIP=1 THEN
1550 4BC3 D001            BNE PNT         NOTHING ELSE
1560 4BC5 60              RTS             RETURN
1570 4BC6 20224F  PNT     JSR PRINT       PRINT ALL INFO
1580 4BC9 ADFC4F          LDA ACT
1590 4BCC C902            CMP #2          VERIFY?
1600 4BCE D02F            BNE PTST        NO
1610 4BD0 ADF54F          LDA VE
1620 4BD3 C902            CMP #2          COMPARE CPY/MSTR?
1630 4BD5 D008            BNE C1          NO
1640 4BD7 A50C            LDA DCPY        YES
1650 4BD9 C50D            CMP DMSTR       ARE THEY SAME?
1660 4BDB D030            BNE RERR        NO-REPORT
1670 4BDD F020            BEQ PTST        TEST-OK
1680 4BDF ADF54F  C1      LDA VE          VERIFY TRIANGULATE?
1690 4BE2 C901            CMP #1          CMPAR CPY/BUFFER?
1700 4BE4 D00A            BNE C3          NO
1710 4BE6 A000            LDY #0
1720 4BE8 B113            LDA (BUFFAD),Y YES
1730 4BEA C50C            CMP DCPY        SAME?
1740 4BEC D01F            BNE RERR        NO-REPORT
1750 4BEE F00F            BEQ PTST        TEST-OK
1760 4BF0 ADF54F  C3      LDA VE
1770 4BF3 C903            CMP #3          CMPAR MSTR/BUFF
1780 4BF5 D008            BNE PTST        NO
1790 4BF7 A000            LDY #0
1800 4BF9 B113            LDA (BUFFAD),Y YES
1810 4BFB C50D            CMP DMSTR       SAME?
1820 4BFD D00E            BNE RERR        NO-REPORT
```

```
1830 4BFE 205D4E   PTST    JSR INCADD      INC EPROM ADDRESS
1840 4C01 20AC4E           JSR INCBUF      INC BUFF ADDRESS
1850 4C04 A514             LDA BUFFAD+1
1860 4C06 CDFE4F           CMP TO          END OF BUFF YET?
1870 4C09 D004             BNE LOOP1       NO
1880 4C0B 60               RTS             RETURN TO BASIC
1890 4C0C 4CB24F   RERR    JMP RERR1       REPORT ERROR
1900 4C0F 4C623B   LOOP1   JMP LOOP
1910                       ;
1920                       ; PROGRAM HANDLER
1930 4C12 A509     PROGR   LDA P9          PROGRAM CONTROL
1940 4C14 D007             BNE NEXTC
1950 4C16 A901             LDA #1          ITS A MOT64K
1960 4C18 8D08C7           STA PA          TURN ON +5V
1970 4C1B D01E             BNE MOTX
1980 4C1D A507     NEXTC   LDA P7
1990 4C1F D004             BNE N7B
2000 4C21 A95F     N5ETH   LDA #255-128-32
2010 4C23 D00E             BNE NP8BC
2020 4C25 A508     N7B     LDA P8
2030 4C27 D004             BNE NP8B
2040 4C29 A93F             LDA #255-128-64
2050 4C2B D006             BNE NP8BC
2060 4C2D A505     NP8B    LDA P5
2070 4C2F F0F0             BEQ N5ETH
2080 4C31 A957             LDA #255-128-32-8
2090 4C33 8D0AC7   NP8BC   STA PB
2100 4C36 A98C             LDA #128+8+4
2110 4C38 8D08C7           STA PA
2120 4C3B 20644E   MOTX    JSR QROUT       SET UP QR AS OUTPUT
2130 4C3E E60E     PRR     INC PN2C
2140 4C40 209F4E           JSR INITAD      INIT BUFF COUNTER
2150 4C43 A000     LOOP1   LDY #0
2160 4C45 B113             LDA (BUFFAD),Y  GET BUFF DATA
2170 4C47 8D06C7           STA QR          SET UP EPROM DATA WRITE
2180 4C4A 20634F           JSR PRNTDT      PRINT PROG.DATA
2190 4C4D 20754C           JSR HANDLR      GO PROGRAM
2200 4C50 205D4E           JSR INCADD      INCREMENT EPROM ADD.
2210 4C53 20AC4E           JSR INCBUF      INCREMENT BUFF COUNTER
2220 4C56 A514             LDA BUFFAD+1    END OF BUFFER?
2230 4C58 CDFE4F           CMP TO
2240 4C5B D0E6             BNE LOOP1       NO
2250 4C5D A502             LDA P2          YES NOW SEE IF 2758
2260 4C5F D006             BNE RTR         NO-DONE
2270 4C61 A50E             LDA PN2C        2 LOOPS DONE
2280 4C63 C902             CMP #2
2290 4C65 D0D7             BNE PRR         NO-LOOP AGAIN
2300 4C67 207A4E   RTR     JSR QRIN        SET QR AS INPUTS
2310 4C6A A97F             LDA #255-128    TURN OFF +25V
2320 4C6C 8D08C7           STA PA          AND COMPLETE LAMP ON
2330 4C6F A9FF             LDA #255        DIS EPROMS
2340 4C71 8D0AC7           STA PB
2350 4C74 60               RTS
2360                       ;
2370                       ; DO READ/WRITE TO/FROM EPROM/S
2380                       ;
2390 4C75 A50A     HANDLR  LDA ADDL        GET LOW 8 BIT EPROM ADD
2400 4C77 8D04C7           STA ADD         AND SET UP
2410 4C7A A50B             LDA ADDH        COMP.ADD AND ENABLES
2420 4C7C 2907             AND #7
2430 4C7E 09A0             ORA #%10100000
2440 4C80 8510             STA MR          INIT. ADD TEMP
2450 4C82 A984             LDA #%10000100
2460 4C84 8511             STA MR
```

```
2470                            ;
2480                            ;SET UP PRELIMINARY ADDRESSING
2490 4C86 A506                  LDA P6          IS IT P#6
2500 4C88 F004                  BEQ SXOR9       YES
2510 4C8A A509                  LDA P9          NO PART#9?
2520 4C8C D012                  BNE N1          NO
2530 4C8E A50B      SXOR9       LDA ADDH        /A12 BIT LOC 7
2540 4C90 2910                  AND #16
2550 4C92 0A                    ASL A
2560 4C93 0A                    ASL A
2570 4C94 4940                  EOR #$40
2580 4C96 0510                  ORA MR
2590 4C98 8510                  STA MR
2600 4C9A 20804F                JSR A114
2610 4C9D 4CE84C                JMP N7OR8
2620 4CA0 A504      N1          LDA P4          IS IT #4
2630 4CA2 D006                  BNE N2          NO
2640 4CA4 20804E                JSR A114
2650 4CA7 4CF64C                JMP RORP
2660 4CAA ADFA4F    N2          LDA PARTN       IS THE PART #
2670 4CAD C904                  CMP #4          <4
2680 4CAF B003                  BCS NGT4        NO
2690 4CB1 4CF64C                JMP RORP
2700 4CB4 A505      NGT4        LDA P5          IS IT #5
2710 4CB6 F004                  BEQ I5OR7
2720 4CB8 A507                  LDA P7          IS IT #7
2730 4CBA D00D                  BNE N5OR7
2740 4CBC A50B      I5OR7       LDA ADDH
2750 4CBE 2908                  AND #8
2760 4CC0 4908                  EOR #8
2770 4CC2 0A                    ASL A
2780 4CC3 0A                    ASL A
2790 4CC4 0A                    ASL A
2800 4CC5 0510                  ORA MR
2810 4CC7 8510                  STA MR
2820 4CC9 A505      N5OR7       LDA P5          IS IT #5
2830 4CCB D00A                  BNE N5          NO
2840 4CCD A50B                  LDA ADDH
2850 4CCF 2908                  AND #8
2860 4CD1 4A                    LSR A
2870 4CD2 8511                  STA NR
2880 4CD4 4CF64C                JMP RORP
2890 4CD7 A507      N5          LDA P7          IS IT #7
2900 4CD9 F004                  BEQ I7OR8       YES
2910 4CDB A508                  LDA P8          IS IT #8
2920 4CDD D017                  BNE RORP        NO
2930 4CDF A50B      I7OR8       LDA ADDH        SET UP A12
2940 4CE1 2910                  AND #16
2950 4CE3 8511                  STA NR
2960 4CE5 4CF64C                JMP RORP
2970 4CE8 A509      N7OR8       LDA P9          IS IT#9
2980 4CEA D00A                  BNE RORP        NO
2990 4CEC A50B                  LDA ADDH        SET UP A12
3000 4CEE 2910                  AND #16
3010 4CF0 0908                  ORA #8
3020 4CF2 4A                    LSR A
3030 4CF3 4A                    LSR A
3040 4CF4 8511                  STA NR
3050 4CF6 ADF84F    RORP        LDA PROG        PRELIM. ADD. IS SET UP
3060 4CF9 F003                  BEQ RDHNDL      GO READ
3070 4CFB 4C764D                JMP PRHNDL      GO PROGRAM
3080                ;
3090                ;
3100                ;           GET EPROM DATA FROM BOARD
```

```
3110                    ;       PLACE MASTER DATA IN DMSTR
3120                    ;       PLACE COPY DATA IN DCPY
3130 4CFE A505  RDHNDL  LDA P5          READ HANDLER
3140 4D00 F018          BEQ I58R9       SPEC COND FOR
3150 4D02 A508          LDA P8          PARTS 5,8,9?
3160 4D04 F014          BEQ I58R9
3170 4D06 A509          LDA P9
3180 4D08 F010          BEQ I58R9
3190 4D0A A507          LDA P7
3200 4D0C D018          BNE N58R9
3210 4D0E A511          LDA NR
3220 4D10 0981          ORA #%10000001  SAVE CODE
3230 4D12 8515          STA ECPY        TO ENA CPY RD
3240 4D14 A511          LDA NR
3250 4D16 098D          ORA #%10001101  DIS CPY RD
3260 4D18 D012          BNE N58R9N      BRA
3270 4D1A A511  I58R9   LDA NR          ITS 5 OR 8 OR 9
3280 4D1C 0983          ORA #128+3      SAVE ENA CPY CODE
3290 4D1E 8515          STA ECPY
3300 4D20 A511          LDA NR
3310 4D22 0981          ORA #128+1      PASS A11 OR A12 DIS CPY RD
3320 4D24 D006          BNE N58R9N      BRA
3330 4D26 A989  N58R9   LDA #128+8+1    STORE ENA CPY
3340 4D28 8515          STA ECPY
3350 4D2A A985          LDA #%10000101
3360 4D2C 8D08C7 N58R9N STA PA
3370 4D2F 8516          STA DISCPY      DIS CPY CODE
3380 4D31 A508          LDA P8
3390 4D33 D006          BNE NP8         ITS NOT #8
3400 4D35 A510          LDA MR
3410 4D37 0940          ORA #64
3420 4D39 D00C          BNE IP8
3430 4D3B A506  NP8     LDA P6          IS TI2564?
3431 4D3D D006          BNE NP6A        NO
3432 4D3F A510          LDA MR
3433 4D41 0920          ORA #32         ENA. READ MSTR
3434 4D43 D002          BNE IP8         ALWAYS BRANCH
3435 4D45 A510  NP6A    LDA MR
3440 4D47 8D0AC7 IP8    STA PB          ENA MSTR READ
3450 4D4A AD06C7        LDA QR          WASTE TIME
3460 4D4D AD06C7        LDA QR          GET DATA
3470 4D50 850D          STA DMSTR       SAVE IT
3480 4D52 A507          LDA P7          NOW DIS MSTR
3490 4D54 D006          BNE NP7         & ENA CPY READ
3500 4D56 A510          LDA MR
3510 4D58 09B0          ORA #%10110000  /CE CPY ONLY
3520 4D5A D004          BNE NP7A         BRA
3530 4D5C A510  NP7     LDA MR
3540 4D5E 29DF          AND #255-32
3550 4D60 8D0AC7 NP7A   STA PB
3560 4D63 A515          LDA ECPY        GET ENA CODE
3570 4D65 8D08C7        STA PA          & ENA CPY
3580 4D68 AD06C7        LDA QR          WASTE TIME
3590 4D6B AD06C7        LDA QR          GET DATA
3600 4D6E 850C          STA DCPY        &SAVE IT
3610 4D70 A516          LDA DISCPY      GET DIS CODE
3620 4D72 8D08C7        STA PA          & DIS CPY
3630 4D75 60            RTS
3640                    ;
3650                    ;       PROGRAM HANDLER
3660 4D76 ADFA4F PRHNDL LDA PARTN
3670 4D79 C904          CMP #4          IS IT <4
3680 4D7B B01A          BCS NOT         NO
3690 4D7D 20924E        JSR PRTIME      YES FIRE 1-SHOT
```

```
3700 4D80 A510            LDA MR
3710 4D82 2957            AND #255-128-32-8
3720 4D84 8D0AC7          STA PB
3730 4D87 0908            ORA #8              PROG.
3740 4D89 8D0AC7          STA PB              PROGRAM
3750 4D8C 20994E          JSR MKTIME          WAIT FOR TIME-OUT
3760 4D8F A510            LDA MR
3770 4D91 2957            AND #255-128-32-8
3780 4D93 8D0AC7          STA PB
3790 4D96 60              RTS
3800 4D97 A504    NOT     LDA P4              => #4
3810 4D99 F033            BEQ ATYPE           PROGRAM PULSE A SIDE
3820 4D9B A506            LDA P6
3830 4D9D F02F            BEQ ATYPE
3840 4D9F A509            LDA P9
3850 4DA1 F043            BEQ MOT64           MOT 64K
3860 4DA3 A507            LDA P7              PARTS 5,7,8
3870 4DA5 D006            BNE NP7HN           PULSE B SIDE
3880 4DA7 A511            LDA NR
3890 4DA9 090C            ORA #8+4
3900 4DAB D002            BNE NP7HA
3910 4DAD A511    NP7HN   LDA NR
3920 4DAF 8D08C7  NP7HA   STA PA              PASS ADDRESSES & +25V
3930 4DB2 20924E          JSR PRTIME          FIRE 1-SHOT
3940 4DB5 A510            LDA MR
3950 4DB7 297F            AND #%01111111
3960 4DB9 0938            ORA #%00111000
3970 4DBB 8D0AC7          STA PB              SET UP ADD
3980 4DBE 8517            STA DISPG
3990 4DC0 2957            AND #%01010111
4000 4DC2 8D0AC7          STA PB              PROGRAM
4010 4DC5 20994E          JSR MKTIME          WAIT 50 MS.
4020 4DC8 A517            LDA DISPG           TAKE AWAY
4030 4DCA 8D0AC7          STA PB              PROG PULSE
4040 4DCD 60              RTS
4050 4DCE A510    ATYPE   LDA MR              PROG #4 OR #6
4060 4DD0 295F            AND #%01011111
4070 4DD2 8D0AC7          STA PB
4080 4DD5 20924E          JSR PRTIME          FIRE 1-SHOT
4090 4DD8 A988            LDA #128+8
4100 4DDA 8D08C7          STA PA              PROGRAM
4110 4DDD 20994E          JSR MKTIME          WAIT 50 MS.
4120 4DE0 A98C            LDA #128+8+4
4130 4DE2 8D08C7          STA PA              PROG PULSE
4140 4DE5 60              RTS
4150 4DE6 A510    MOT64   LDA MR              PROG MOTOROLA 64K
4160 4DE8 295F            AND #%01011111      REQUIRES +25V PULSING
4170 4DEA 8D0AC7          STA PB              SET UP ADDRESSES
4180 4DED 20924E          JSR PRTIME          FIRE 1-SHOT
4190 4DF0 A511            LDA NR              SET UP A12
4200 4DF2 2904            AND #4
4210 4DF4 0901            ORA #1              SET UP A12 LEAVE +5V
4220 4DF6 8D08C7          STA PA
4230 4DF9 8517            STA DISPG           SAVE DISABLE CODE
4240 4DFB 2904            AND #4
4250 4DFD 8D08C7          STA PA              +25V PULSE APPLIED
4260 4E00 20994E          JSR MKTIME          WAIT 50 MS.
4270 4E03 A517            LDA DISPG
4280 4E05 8D08C7          STA PA              TAKE AWAY PULSE
4290 4E08 60              RTS
4300              ;
4310              ;       BEGIN SUBS
4320 4E09 204023  HEXCK   JSR INCH            GET CHAR FROM KEYBOARD
4330 4E0C 8DF34F          STA CHAR            CHECK FOR VALID HEX
```

```
4340 4E0F C930          CMP #'0
4350 4E11 3035          BMI SERRR       NOT HEX
4360 4E13 C93A          CMP #':
4370 4E15 900A          BCC OK          IS HEX
4380 4E17 C947          CMP #'G
4390 4E19 B02D          BCS SERRR       NOT HEX
4400 4E1B C941          CMP #'A
4410 4E1D 9029          BCC SERRR       NOT HEX
4420 4E1F E907          SBC #7
4430 4E21 290F    OK    AND #$F         IS HEX CONVERT
4440 4E23 60            RTS
4450 4E24 20094E  INHEX JSR HEXCK       GET 2-DIG HEX CHARACTER
4460 4E27 0A            ASL A
4470 4E28 0A            ASL A
4480 4E29 0A            ASL A
4490 4E2A 0A            ASL A
4500 4E2B 8DF14F        STA LTEMP
4510 4E2E ADF24F        LDA HOASC       IS IT ASCII
4520 4E31 D014          BNE ASCII       YES
4530 4E33 20094E        JSR HEXCK
4540 4E36 8DF04F        STA HTEMP
4550 4E39 ADF24F        LDA HOASC
4560 4E3C D009          BNE ASCII
4570 4E3E ADF14F        LDA LTEMP
4580 4E41 0DF04F        ORA HTEMP
4590 4E44 8DF34F        STA CHAR
4600 4E47 60      ASCII RTS             RET BAS
4610 4E48 EEF24F  SERRR INC HOASC       SET ASCII FLAG
4620 4E4B 60            RTS
4630 4E4C ADF34F  HXOUT LDA CHAR
4640 4E4F 20B34E        JSR ASC
4650 4E52 60            RTS             RET BAS
4660 4E53 A900    RSTADD LDA #0         RESET ACTUAL EPROM ADDRESS
4680 4E55 8DF24F        STA HOASC       SET TO HEX
4690 4E58 850A          STA ADDL
4700 4E5A 850B          STA ADDH
4710 4E5C 60            RTS
4720 4E5D E60A    INCADD INC ADDL       INC EPROM ADD.
4730 4E5F D002          BNE NCY
4740 4E61 E60B          INC ADDH
4750 4E63 60      NCY   RTS
4760 4E64 A9FF    QROUT LDA #255        SET QR AS OUTPUT
4770 4E66 850F          STA QREG
4780 4E68 A900          LDA #0
4790 4E6A 8D07C7  QRX   STA ADD+3
4800 4E6D 850E          STA PN2C
4810 4E6F A50F          LDA QREG
4820 4E71 8D06C7        STA ADD+2
4830 4E74 A904          LDA #4
4840 4E76 8D07C7        STA ADD+3
4850 4E79 60            RTS
4860 4E7A A900    QRIN  LDA #0          SET QR AS INPUT
4870 4E7C 850F          STA QREG
4880 4E7E F0EA          BEQ QRX         FALL THROUGH SAVE RTS CODE
4890 4E80 A50B    A114  LDA ADDH        JUGGLE ADDRESS 11
4900 4E82 2908          AND #8
4910 4E84 0510          ORA MR
4920 4E86 8510          STA MR
4930 4E88 A50B          LDA ADDH
4940 4E8A 2908          AND #8
4950 4E8C 0A            ASL A
4960 4E8D 0510          ORA MR
4970 4E8F 8510          STA MR
4980 4E91 60            RTS
```

```
4990 4E92 AD0AC7  PRTIME LDA PB          FIRE 1-SHOT
5000 4E95 AD08C7         LDA PA
5010 4E98 60             RTS
5020 4E99 AD0BC7  MKTIME LDA PB+1        MARKTIME 50 MS.
5030 4E9C 10FB           BPL MKTIME      NOT DONE
5040 4E9E 60             RTS             TIME UP!
5050 4E9F A900    INITAD LDA #0          INIT BUFFER ADDRESS
5060 4EA1 A8             TAY
5070 4EA2 8513           STA BUFFAD
5080 4EA4 8518           STA VIDEO
5090 4EA6 ADFF4F         LDA FROM
5100 4EA9 8514           STA BUFFAD+1
5110 4EAB 60             RTS
5120 4EAC E613    INCBUF INC BUFFAD      INC BUFF ADDRESS
5130 4EAE D002           BNE NOCY
5140 4EB0 E614           INC BUFFAD+1    IF CARRY
5150 4EB2 60      NOCY   RTS
5160 4EB3 48      ASC    PHA             CONV ACC. TO 2-HEX
5170 4EB4 4A             LSR A           OUTPUT TO SCREEN
5180 4EB5 4A             LSR A
5190 4EB6 4A             LSR A
5200 4EB7 4A             LSR A
5210 4EB8 20C84E         JSR CVERT
5220 4EBB 204323         JSR OUTCH
5230 4EBE 68             PLA             RESTORE
5240 4EBF 290F           AND #$F
5250 4EC1 20C84E         JSR CVERT
5260 4EC4 204323         JSR OUTCH
5270 4EC7 60             RTS
5280 4EC8 C90A    CVERT  CMP #10         TEST 0-9
5290 4ECA 3003           BMI NOAD        GO IF 0-9
5300 4ECC 18             CLC
5310 4ECD 6907           ADC #7
5320 4ECF 6930    NOAD   ADC #$30
5330 4E01 60             RTS
5340              ;
5350              ;       INITIALIZE EPROM PROG.BOARD
5360 4ED2 207A4E  INIT   JSR QRIN        SET QR AS INPUTS
5370 4ED5 A900           LDA #0
5380 4ED7 8D05C7         STA ADD+1       INITIALIZE BOARD
5390 4EDA A9FF           LDA #255
5400 4EDC 8D04C7         STA ADD
5410 4EDF A904           LDA #4
5420 4EE1 8D05C7         STA ADD+1
5430 4EE4 A92E           LDA #46
5440 4EE6 8D09C7         STA ADD+5
5450 4EE9 A9FF           LDA #255
5460 4EEB 8D08C7         STA ADD+4
5470 4EEE 8D0AC7         STA ADD+6
5480 4EF1 A92A           LDA #42
5490 4EF3 8D09C7         STA ADD+5
5500 4EF6 A99F           LDA #159
5510 4EF8 8D08C7         STA ADD+4
5520 4EFB A92E           LDA #46
5530 4EFD 8D09C7         STA ADD+5
5540 4F00 A930           LDA #48
5550 4F02 8D0BC7         STA ADD+7
5560 4F05 A9FF           LDA #255
5570 4F07 8D0AC7         STA ADD+6
5580 4F0A A934           LDA #52
5590 4F0C 8D0BC7         STA ADD+7
5600 4F0F 60             RTS          INITIALIZED RET.BASIC
5610              ;
5620              ;       PRINT ADJUSTED BUFFER ADRESS ON SCREEN
```

```
5630                    ;       4-DIGIT HEX
5640 4F10 A514      PRNTAD LDA BUFFAD+1    GET BUFFER HIGH ADD
5650 4F12 38            SEC                ADJUST ADDRESS PRINTED
5660 4F13 E950          SBC $$50
5670 4F15 18            CLC
5680 4F16 6DEE4F        ADC OFFSET
5690 4F19 20B34E        JSR ASC            AND GO PRINT IT
5700 4F1C A513          LDA BUFFAD         GET CURRENT LOW ADD
5710 4F1E 20B34E        JSR ASC            AND GO PRINT IT
5720 4F21 60            RTS
5730 4F22 ADEF4F    PRINT  LDA SLIST       SUPRESS LISTING
5740 4F25 D03B          BNE SUP            YES
5750 4F27 20104F    DEFPNT JSR PRNTAD
5760 4F2A 20732D        JSR STROUT
5770 4F2D 20            .BYTE ' COPY=',0
5770 4F2E 43
5770 4F2F 4F
5770 4F30 50
5770 4F31 59
5770 4F32 3D
5770 4F33 00
5780 4F34 A50C          LDA DCPY           GET COPY DATA
5790 4F36 20B34E        JSR ASC
5800 4F39 20732D        JSR STROUT
5810 4F3C 20            .BYTE '  MASTER=',0
5810 4F3D 20
5810 4F3E 4D
5810 4F3F 41
5810 4F40 53
5810 4F41 54
5810 4F42 45
5810 4F43 52
5810 4F44 3D
5810 4F45 00
5820 4F46 A50D          LDA DMSTR          GET MASTER DATA
5830 4F48 20B34E        JSR ASC
5840 4F4B 20732D        JSR STROUT
5850 4F4E 20            .BYTE '  MEMORY=',0
5850 4F4F 20
5850 4F50 4D
5850 4F51 45
5850 4F52 4D
5850 4F53 4F
5850 4F54 52
5850 4F55 59
5850 4F56 3D
5850 4F57 00
5860 4F58 A000          LDY #0
5870 4F5A B113          LDA (BUFFAD),Y  GET BUFFER DATA
5880 4F5C 20834E        JSR ASC
5890 4F5F 206A2D        JSR CRLF
5900 4F62 60        SUP     RTS
5910 4F63 ADEF4F    PRNTDT LDA SLIST       SUPRESS LISTING?
5920 4F66 D01B          BNE WRKNG          YES
5930 4F68 20D84F        JSR WPROG
5940 4F6B 20104F        JSR PRNTAD
5950 4F6E 20732D        JSR STROUT
5960 4F71 20            .BYTE ' WITH ',0
5960 4F72 57
5960 4F73 49
5960 4F74 54
5960 4F75 48
5960 4F76 20
5960 4F77 00
```

```
5970 4F78 A000            LDY #0
5980 4F7A B113            LDA (BUFFAD),Y
5990 4F7C 20B34E          JSR ASC
6000 4F7F 206A2D          JSR CRLF
6010 4F82 60              RTS
6020 4F83 A504    WRKNG   LDA ADDL          INDICATE RUNNING
6030 4F85 290F            AND #15           EVERY 16
6040 4F87 D008            BNE VDRT
6050 4F89 A90D            LDA #13           CRETURN
6060 4F8B 204323          JSR OUTCH
6070 4F8E 20104F          JSR PRNTAD        PRINT ADDRESS WORKING ON
6080 4F91 60      VDRT    RTS          CONTINUE
6090 4F92 20732D  BLNKER  JSR STROUT        EPROM NOT BLANK
6100 4F95 45              .BYTE 'EPROM NOT BLANK OR ALL (FF)S',0
6100 4F96 50
6100 4F97 52
6100 4F98 4F
6100 4F99 4D
6100 4F9A 20
6100 4F9B 4E
6100 4F9C 4F
6100 4F9D 54
6100 4F9E 20
6100 4F9F 42
6100 4FA0 4C
6100 4FA1 41
6100 4FA2 4E
6100 4FA3 4B
6100 4FA4 20
6100 4FA5 4F
6100 4FA6 52
6100 4FA7 20
6100 4FA8 41
6100 4FA9 4C
6100 4FAA 4C
6100 4FAB 20
6100 4FAC 28
6100 4FAD 46
6100 4FAE 46
6100 4FAF 29
6100 4FB0 53
6100 4FB1 00
6110 4FB2 206A2D  RERRR   JSR CRLF
6120 4FB5 20732D          JSR STROUT        REPORT ERROR
6130 4FB8 45              .BYTE 'ERROR ',0
6130 4FB9 52
6130 4FBA 52
6130 4FBB 4F
6130 4FBC 52
6130 4FBD 20
6130 4FBE 00
6140 4FBF 206A2D          JSR CRLF
6150 4FC2 A900            LDA #0
6160 4FC4 8DEF4F          STA SLIST
6170 4FC7 20224F          JSR PRINT
6180 4FCA 206A2D          JSR CRLF
6190 4FCD A955            LDA #%01010101 85=ERROR
6200 4FCF 8DF44F          STA ERROR
6210 4FD2 A93C            LDA #60           ERROR LIGHT ON
6220 4FD4 8D0BC7          STA ADD+7
6230 4FD7 60              RTS               RETURN TO BASIC
6240 4FD8 20732D  WPROG   JSR STROUT
6250 4FDB 50              .BYTE 'PROGRAMMING ',0
6250 4FDC 52
```

```
6250 4FDD 4F
6250 4FDE 47
6250 4FDF 52
6250 4FE0 41
6250 4FE1 4D
6250 4FE2 4D
6250 4FE3 49
6250 4FE4 4E
6250 4FE5 47
6250 4FE6 20
6250 4FE7 00
6260 4FE8 60                RTS
6270 4FEC                   *=BUFF-20
6280                  ; INITIALIZE PASS BUFFER ONE TIME
6290 4FEC 0000              .DBYTE 0,0,0,0,0,0,0,0,0,0
6290 4FEE 0000
6290 4FF0 0000
6290 4FF2 0000
6290 4FF4 0000
6290 4FF6 0000
6290 4FF8 0000
6290 4FFA 0000
6290 4FFC 0000
6290 4FFE 0000

.!IO .,01
```

REPACK - CA23

```
1 DEF FNA(X)=10*INT(X/16)+X-16*INT(X/16)
5 POKE 2073,96
10 P0=18302
20 FORI=1TO5:PRINT:NEXT
30 PRINT"** BASIC PROGRAM REPACK UTILITY **"
31 PRINT
32 PRINT"** WARNING --- USE DRIVE A ONLY **"
35 DISK!"SE A
40 FORI=1TO5:PRINT:NEXT
50 A$="PROG":GOSUB100:REM FILE TO REPACK !!!!
51 IFA=0THENPRINT"FILE NOT FOUND":GOTO91
55 POKEP0,A:GOSUB300
70 DISK!"GO 4780"
80 FR=PEEK(P0)+256*PEEK(P0+1)
90 PRINT:PRINTFR"BYTES RECOVERED."
91 RUN"BEXEC*
100 DR=11897:A=0
105 A$=A$+"      ":A$=LEFT$(A$,6)
110 DISK!"CA 2E79=08,1":GOSUB200
120 IFA>0THENRETURN
130 DISK!"CA 2E79=08,2"
200 FORI=1TO32:N$="":FORJ=0TO5
210 N$=N$+CHR$(PEEK(DR+(I-1)*8+J))
220 NEXTJ:IFA$<>N$THENNEXTI:RETURN
230 A=PEEK(DR+(I-1)*8+6):A=FNA(A)
240 RETURN
300 PRINT:PRINT"REPACKING  FILE ("A$")":PRINT
350 A=3:POKEP0+1,A:RETURN
```

BEXEC* CA23

```
10 REM BASIC EXECUTIVE
20 REM
24 REM SETUP INFLAG & OUTFLAG FROM DEFAUL
25 X=PEEK(10950): POKE 8993,X: POKE 8994,X
27 IFPEEK(57088)=223THENPOKE9794,37
30 GOTO 100
60 A$="UNLOCK
65 IF A$="COLORS" THEN RUN "COLORS"
70 IF A$="CHANGE" THEN RUN "CHANGE"
80 IF A$="DIR"    THEN RUN "DIR    "
90 IF A$="UNLOCK" THEN 10000
100 PRINT
150 GOTO 60
10000 REM
10010 REM UNLOCK SYSTEM
10020 REM
10030 REM REPLACE "NEW" AND "LIST"
10040 POKE 741,76 : POKE 750,78
10050 REM
10060 REM ENABLE CONTROL-C
10070 POKE 2073,173
10080 REM
10085 FORI=1TO30:PRINT:NEXTI:REM SCROLL
10090 REM DISABLE "REDO FROM START"
10100 POKE 2893,55 : POKE 2894,8
10101 X=15:GOSUB11000
10102 PRINTTAB(X);"**       SYSTEM OPEN          **"
10104 GOSUB11000:GOSUB11020:GOSUB11020
10120 PRINTTAB(X);"**   EPROM PROGRAMER V1.1   **"
10123 GOSUB11020:GOSUB11020:GOSUB11000
10124 PRINT:PRINT:PRINT:PRINT:PRINT:PRINT
10125 FORI=1TO4000:NEXTI
10128 RUN"PROG
10130 RUN"PROG
11000 PRINTTAB(X);"******************************":RETURN
11020 PRINTTAB(X);"**";TAB(42);"**":RETURN
```

10122 ? Tab(x);"** By None

System Delay.

10125 INPUT "1=2716 2=OTHER"; X

10126 IF X=1 THEN RUN E2716

PROG CA23

```
10 REM
20 POKE132,255:POKE133,7
30 CLEAR:BEG=(80)%256:BG=BEG/256
40 AA=50948:CR=AA+2:DO=CR+1:SU=DO+1:CA=SU+1:AH=CA+1:CR=AH+1
50 IFPEEK(BEG-19)=85THENGOTO80
60 DISK!"CA 4800=10,1:REM GET MACHINE CODE HANDLER
70 POKEBEG-18,0:POKEBEG-17,0:POKEBEG-1,10
80 GOSUB960:GOSUB970:GOSUB1300:GOSUB760
90 GOSUB1300:GOSUB140:GOSUB740:GOSUB740:GOSUB220
100 IFAC=6THENGOTO1320
110 ONACGOTO280,440,560,570,610
120 GOTO90
130 :
140 PRINT"LIST OF PARTS THAT PROGRAM WILL HANDLE":GOSUB740
150 PRINT"1=TMS 2508":PRINT"2=TMS 2758,INT 2758
160 PRINT"3=TMS 2516,INT 2716,MCM 2716":PRINT"4=TMS 2532,MCM 2532
170 PRINT"5=INT 2732":PRINT"6=TMS 2564":PRINT"7=INT 2764
180 PRINT"8=MK  2764":PRINT"9=MCM 68764
190 :
200 INPUT"PART NUMBER YOU WISH TO WORK WITH";PN:IFPN<1ORPN>9THENGOTO140
210 GOSUB860:POKEBEG-6,PN:RETURN
220 PRINT"1 = DUPLICATE":PRINT"2 = VERIFY":PRINT"3 = LIST
230 PRINT"4 = PROGRAM FROM MEMORY
240 PRINT"5 = LOAD MEMORY":PRINT"6 = EDIT/SAVE MEMORY
250 INPUT"WHAT DO YOU WISH TO DO";AC:IFAC<1ORAC>6THENGOTO220
260 GOSUB740:POKEBEG-4,AC:RETURN
270 :
280 REM
290 GOSUB2090:GOSUB1090:GOSUB930
300 IFPN=2THENGOSUB1150
310 GOSUB790
320 POKEBEG-8,0:GOSUB1190
330 POKEBEG-8,1:GOSUB1190:AC=2:VE=2
340 POKEBEG-9,0:B0=0:B1=0
350 POKEBEG-11,VE:POKEBEG-8,0:POKEBEG-4,AC:GOSUB1190
360 X=PEEK(BEG-12):IFPN=2ANDX<>0THENB0=1
370 IFPN=2THENPOKEBEG-9,1:GOSUB1190
380 X=PEEK(BEG-12):IFPN=2ANDX<>0THENB1=1
390 GOSUB740:IFPN=2ANDB0=0THENPRINT"LOW BANK-";
400 IFPN=2ANDB1=0THENPRINT"HIGH BANK-";
410 IFPN=2ANDB0=1ANDB1=1THENPRINT"NEITHER BANK IS GOOD":GOTO2120
420 PRINT" VERIFIES OK":GOTO2130
430 :
440 GOSUB2090
450 PRINT"1 = VERIFY COPY TO MEMORY":PRINT"2 = VERIFY COPY TO MASTER"
460 PRINT"3 = VERIFY MASTER TO MEMORY"
470 INPUTVE:IFVE<1ORVE>3THEN450
480 IFPN=2THENGOSUB1150
490 POKEBEG-11,VE
500 IFVE=2THENGOSUB1090
510 IFVE=1THENGOSUB1140
520 IFVE=3THENGOSUB1130
530 POKEBEG-8,0:GOSUB1190:X=PEEK(BEG-12):IFPN=2ANDX<>0THENGOTO2120
540 GOTO2130
550 :
560 POKEBEG-17,0:GOTO620
570 GOSUB2090:GOSUB1140:GOSUB790
580 GOSUB930:IFSK=0THENPOKEBEG-8,0:GOSUB1190
590 POKEBEG-8,1:GOSUB1190:VE=1:AC=2:GOTO350
600 :
610 GOSUB2090
620 PRINT"1 = FROM MASTER SOCKET":PRINT"2 = FROM COPY SOCKET
630 IFAC=5THENPRINT"3 = FROM DISK"
```

```
640 IFAC=3THENPRINT"3 = FROM MEMORY
650 INPUTFR:IFFR<1ORFR>3THENGOTO620
660 POKEBEG-10,FR
670 IFAC=5ANDFR=3THENGOTO1930
680 IFFR=1THENGOSUB1130
690 IFFR=2THENGOSUB1140
700 IFPN=2ANDFR<>3THENGOSUB1150
710 POKEBEG-8,0:GOSUB1190:GOTO2130
720 :
730 REM
740 FORSC=1TO6:PRINT:NEXT:RETURN
750 GOSUB740:GOSUB780:RETURN
760 GOSUB750:PRINT" READ"
770 POKE2888,0:INPUT"HIT RETURN WHEN READY";A$:POKE2888,27:RETURN
780 PRINT"SLIDE THE PROGRAM/READ SWITCH TO ";:RETURN
790 GOSUB750:PRINT"PROGRAM"
800 GOSUB820:IFPR=0GOTO740
810 GOTO800
820 PR=PEEK(SW):PR=PRAND64:RETURN
830 GOSUB750:PRINT"READ"
840 GOSUB820:IFPR=64THENGOTO740
850 GOTO840
860 MZ=PEEK(8960):IFPN<4ANDMZ<95THENGOTO2160
870 IFPN<3THENPOKEBEG-2,BG+4
880 IFPN=3THENPOKEBEG-2,BG+8
890 IFPN=4ORPN=5THENPOKEBEG-2,BG+16
900 IFPN>5THENPOKEBEG-2,BG+32
910 GOSUB740:GOSUB1100:IFPN>3ANDMZ<127THENGOTO2160
920 GOSUB770:GOSUB740:RETURN
930 GOSUB740:INPUT"SKIP BLANK VERIFY TEST <Y OR N>";A$
940 POKEBEG-5,0:IFLEFT$(A$,1)="Y"THENPOKEBEG-5,1
950 RETURN
960 POKEBEG-8,2:GOSUB1190:RETURN
970 GOSUB960:GOSUB1040:IFPWR=0THENGOTO990
980 PRINT:PRINT" SLIDE THE ON/OFF SWITCH TO OFF";
990 GOSUB960:GOSUB1040:IFPWR=1GOTO990
1000 GOSUB960:GOSUB740:RETURN
1010 PRINT" SLIDE THE ON/OFF SWITCH TO ON";
1020 GOSUB960:GOSUB1040:IFPWR=1THENGOSUB960:GOSUB740:GOSUB1250:RETURN
1030 GOTO1020
1040 PWR=0:POKEAA,85:IFPEEK(AA)=85THENPWR=1
1050 RETURN
1060 REM
1070 PRINT" THE MASTER EPROM IN THE MASTER SOCKET":RETURN
1080 PRINT"LOAD THE COPY EPROM IN THE COPY SOCKET":RETURN
1090 GOSUB1100:PRINT"LOAD";:GOSUB1070:GOSUB1080:GOSUB1010:RETURN
1100 PRINT"SLIDE THE (TYPE) SWITCH TO POSITION (";
1110 IFPN<5ORPN=6ORPN=7THENPRINT"A)":RETURN
1120 PRINT"B)":RETURN
1130 GOSUB1100:PRINT"LOAD";:GOSUB1070:GOSUB1010:RETURN
1140 GOSUB1100:GOSUB1080:GOSUB1010:RETURN
1150 INPUT"IS IT A LOW OR HIGH MODE EPROM <0 OR 1>";LH
1160 IFLH=0ORLH=1THENPOKEBEG-9,LH:RETURN
1170 GOTO1150
1180 :
1190 POKEBEG-12,0:POKE8955,0:POKE8956,75
1200 X=USR(X)
1210 IFPEEK(BEG-12)=0ANDPEEK(BEG-8)<>2ANDAC<>0THENPOKE50952,255-128
1220 IFPN=2ANDAC=2THENGOTO1240
1230 IFPEEK(BEG-12)<>0THENGOTO2120
1240 RETURN
1250 X=PEEK(SW):POKESW,XOR4
1260 TM=PEEK(SW):TM=TMAND32
1270 IFPN<5ORPN=6ORPN=7THENIFTM=0THENGOTO2140
```

```
1280 IFPN=50RPN>7THENIFTM<>0THENGOTO2140
1290 POKESW,X:RETURN
1300 FORI=1TO3:GOSUB740:NEXT:RETURN
1310 :
1320 POKEBEG-8,3:GOSUB1300:PRINT"BASE ADDRESS(HEX)?";:GOSUB1870
1330 LI=PEEK(BEG-2):LI=LI-80:LI=LI%256:LI=LI+20480
1340 IFE=1THENGOTO1320
1350 A=INT(DE/256):IFA<>DE/256THENGOTO1320
1360 LL=LI-20480:LL=65536-LL:IFDE>LLTHENPRINT:PRINT"CAN'T BE":GOTO1320
1370 POKEBEG-18,A:POKEBEG-19,85
1380 OF=DE:DF=OF-BEG
1390 PRINT:PRINT"(O)PEN,(L)IST,(P)RESET,(S)AVE,E(X)IT ?";
1400 GOSUB1190:CH=PEEK(BEG-13)
1410 :
1420 IFCH=ASC("L")THENGOTO1490
1430 IFCH=ASC("O")GOTO1700
1440 IFCH=ASC("X")THENRUN
1450 IFCH=ASC("P")THENGOTO1800
1460 IFCH=ASC("S")THENPRINT"AVE";:GOTO1930
1470 GOTO1390
1480 :
1490 PRINT"1ST-";:GOSUB1500:GOTO1620
1500 PRINT"FROM? ";:GOSUB1870:IFE=1THENGOTO1500
1510 FR=DE-DF:IFFR<20480ORFR=>LITHENPRINT:PRINT"OUT OF RANGE ";:GOTO1500
1520 PRINT" TO? ";:GOSUB1870:IFE=1THENGOTO1500
1530 TT=DE-DF:IFCH=ASC("P")THENRETURN
1540 IFTT=>LIORTT<FRTHENPRINT:PRINT"OUT OF RANGE ";:GOTO1520
1550 SD=PEEK(8994)
1560 PRINT:INPUT"LIST ON LINE PRINTER";A$:IFLEFT$(A$,1)<>"Y"THENGOTO1610
1570 INPUT"TITLE";A$:IFSD=1THENPOKE8994,9:GOTO1600
1580 INPUT"DEVICE<1OR4>";DV:IFDV<1ORDV>4THENGOTO1580
1590 POKE8994,3:IFDV=4THENPOKE8994,10
1600 PRINT:PRINTA$
1610 RETURN
1620 PRINT:FORJOB=FRTOTT:GOSUB1830:IFJOB=TTTHENGOTO1690
1630 JOB=JOB+1:FORNJ=JOBTOJOB+6:PRINT" ";:Z=PEEK(JOB)
1640 POKEBEG-13,Z:GOSUB1910:IFNJ<>TTTHENJOB=JOB+1:NEXT:JOB=JOB-1
1650 PRINTTAB(13);
1660 FORJK=JO-7TOJO:A=PEEK(JK):IFA<33ORA>94THENPRINT".";:GOTO1680
1670 PRINTCHR$(A);
1680 NEXTJK
1690 PRINT:NEXTJOB:POKE8994,SD:GOTO1390
1700 PRINT"PEN?";:GOSUB1870:IFE=1THENPRINT"O";:GOTO1700
1710 PRINT:JOB=DE-DF
1720 IFJOB<20480RJOB=>LITHENPRINT"OUT OF RANGE-O";:GOTO1700
1730 GOSUB1830:PRINT" ";
1740 HI=1:X=1:GOSUB1880:IFE<>1THENGOTO1790
1750 CH=PEEK(BEG-13):IFCH=10THENJOB=JOB+1:PRINTCHR$(13);:GOTO1720
1760 IFCH=13THENGOTO1390
1770 IFCH=94THENJOB=JOB-1:PRINT:GOTO1720
1780 PRINT"?":GOTO1720
1790 POKEJOB,DE:JOB=JOB+1:PRINT:GOTO1720
1800 PRINT"RESET-";:GOSUB1500
1810 PRINT" WITH? ";:X=1:HI=1:GOSUB1880:IFE=1THENGOTO1810
1820 PRINT:PRINT"WORKING";:FORJOB=FRTOTT:POKEJOB,DE:NEXT:GOTO1390
1830 REM
1840 K=JOB+DF:H=INT(K/256):L=K-(H%256)
1850 POKEBEG-13,H:GOSUB1910:POKEBEG-13,L:GOSUB1910
1860 Z=PEEK(JOB):PRINT" ";:POKEBEG-13,Z:GOSUB1910:RETURN
1870 HI=2:X=256
1880 E=0:DE=E:FORI=1TOHI
1890 GOSUB1190:IFPEEK(BEG-14)<>0THENE=1:RETURN
1900 DE=(PEEK(BEG-13)%X)+DE:X=X-255:NEXTI:RETURN
1910 REM
```

```
1920 POKEBEG-8,4:GOSUB1190:POKEBEG-8,3:RETURN
1930 GOSUB1300:PRINT"FILES <1-7> ARE AVAILABLE"
1940 PRINT"AN OUT OF RANGE NUMBER ABORTS"
1950 POKE8955,212:POKE8956,3:
1960 IFAC=5THENPOKE8917,3:PRINT"LOAD MEMORY FROM WHICH FILE?";
1970 IFAC=6THENPOKE8917,0:PRINT"SAVE MEMORY IN WHICH FILE?";
1980 INPUTF:IFF<1ORF>7THENGOSUB740:PRINT"ABORTED":RUN
1990 POKE9822,1:POKE9823,8
2000 POKE9824,0:POKE9825,80:TR=29
2010 FORTR=TRTOTR+3
2020 POKE9826,TR+(F*4)
2030 X=USR(X)
2040 PG=PEEK(9825):POKE9825,PG+8
2050 NEXT
2060 GOSUB740:IFAC=5THENPRINT"MEMORY LOADED FROM";
2070 IFAC=6THENPRINT"MEMORY SAVED";
2080 PRINT" FILE (";F;" )":RUN
2090 INPUT"SUPPRESS LISTING DURING THIS ACTION";A$
2100 IFLEFT$(A$,1)<>"Y"THENPOKEBEG-17,0:RETURN
2110 POKEBEG-17,255:RETURN
2120 POKECB,60:PRINT"ERROR-";:GOSUB770:RUN
2130 POKE50952,255-128:PRINT"COMPLETE-";:GOSUB770:RUN
2140 PRINT:PRINT"(TYPE) SWITCH IS IN WRONG POSITION !!!"
2150 GOSUB1100:RUN
2160 GOSUB740:PRINT"NOT ENOUGH MEMORY
```