P.R.L.

TECHNICAL NOTE

TN-83-34

EPROM PROGRAMMER USING SDK-85

By

Surendra Pal and
R.S. Singh

April 1983

Physical Research Laboratory
Navrangpura, Ahmedabad-380009

AN EPROM PROGRAMMER USING SDK-85

Surendra Pal and R.S. Singh,
Physical Research Laboratory,
Ahmedabad-380009.

Abstract

A PROM programmer is developed to program and read a number of EPROMs viz. 8755, 8755-A, 2758, 2716 and 2732. The system is based on Intel's SDK-85 microcomputer and a very little hardware is required in addition to the computer. The software makes the system interactive and a number of prompting messages are incorporated to minimize any possibility of error. Each byte is verified after it is programmed and in case of error an appropriate message is sent to the console device and the decision is left to the operator to continue or terminate the further programming. Content of an EPROM can be read and printed as a whole or in part.

1. Introduction

The importance of programming an EPROM does not need any introduction, however, often it is not possible to justify to have an exclusive programming system as the number of chips need be programmed in most of the laboratories is not very large. The PROM programmer described here is most suitable for such users. The system is based on popular SDK-85 microcomputer of Intel and can be used to program or read a number of EPROMS, viz. 8755, 8755-A, 2758, 2716 and 2732. In addition to the microcomputer, a little hardware is required which is discussed in Section-4 of this document. The software, developed for this, makes the system interactive and a number of prompting messages are incorporated in order to minimize any possibility of error from the operator's side. This software resides in the memory location from 8000 (Hex) through 83C2(Hex). Some additional RAM is required to store the data to be programmed. Also a part of the basic RAM of the microcomputer is used by this program. Under this software the microcomputer runs in TTY mode so a video terminal or a teletype is required as the console device.

Any number of bytes (ofcourse less than the PROM's capacity) can be programmed in one run. Each byte is verified after it is programmed and in case of error an appropriate message is sent to the console and the decision is

left to the operator to continue or terminate the further programming. This software is discussed in Section-5 and a complete listing of the programme is given in the Appendix-3.

2. Programming Procedure

The data to be programmed should be first loaded into RAM of the microcomputer and then the PROM should be inserted in its appropriate socket for programming. After checking that the PROM is properly inserted the "EPROM Programmer" program residing at 8000 (Hex) should be executed using the 'G' command of SDK-85 monitor.

The program begins by printing the title message

EPROM PROGRAMMER (PLT-AS-1982)

EPROM TYPE#

The operator now should enter one of the following numbers depending upon the EPROM he wants to program

8755, 2758, 2716, or 2732

followed by a (CR), (see Notes 1 and 2)

Note 1. While entering these numbers the operator can enter any number of digits but only last 4 digits will be accepted.

2. If 8755-A is to be programmed, then also the operator should enter 8755 only.

The program next asks for the source field addresses by printing

SOURCE FIELD INITIAL ADDR. (HEX) ?

and the user should respond with a four-digit number followed by a (CR) (see Note 1). This number should be the lowest address of data, stored in RAM, that is to be programmed.

The final address is then asked by printing

SOURCE FIELD FINAL ADDR. (HEX) ?

This number should be highest address of the data to be programmed.

Having received the initial and final addresses of the source field, the computer asks if the operation is to program or read the PROM by printing

P, R ?

Here the response should be P (R is for reading which is described in Section-3).

Next the computer asks for the destination address by printing

DESTINATION (EPROM) INITIAL ADDR. (HEX) ?

and the operator should respond with a 4 digit number followed by (CR). This number should be the lowest address at which the first byte of data is to be programmed in PROM.

Now the computer is readyfor operation, but the final clearance is sought from the operator by printing

EXEC (Y/N) ?

Here the operator should check carefully all the numbers entered by him and also that the power supply of the 'programming unit' is switched ON. If everything is satisfactory, the operator's response should be Y (for yes). At this response the computer starts execution and the LED-s on the programming unit indicate the status. In case the operator finds something wrong, he should type N (for No). The computer will restart the whole process by printing the title message.

During the programming operation, each byte is verified after it is programmed in the PROM and in case of an error the following message is printed

PROGRAMMING ERROR IN EPROM AT (addr.)

CONTINUE (Y/N) ?

(addr.) is a four digit number indicating the address in PROM at which the error is detected, and the computer waits for the operator's response. If the operator wants to continue the further programming operation despite this error, he should respond with Y. If the operation is to be terminated, the response should be N, in this case the computer will restart by printing the title message.

When the data stored in the final address location of the source field is programmed the computer prints the following message

** DONE **

and the program restarts.

Some typical examples of the complete programming operation are given in Appendix-1.

3. Reading Procedure

The beginning of the reading procedure is similar to the one for programming. The computer first asks for the type of PROM to be read and then the source field addresses are asked. Here one should note that the source field is now in the PROM. So the initial address is the address of the first data byte to be read from the PROM and the final address is the address of the last data byte. Having received these addresses the computer asks for the mode of the operation by printing

P / R ?

The operator should respond with R (for reading) and the computer starts the execution. The data is printed on the console device in the format such that each line of the listing begins with the address of the first memory location displayed on that line, represented as 4 hexadecimal digits, followed by up to 16 memory locations, each one represented by 2 hexadecimal digits.

After the complete data is printed the computer prints

** DONE **

and the 'EPROM PROGRAMMER' program restarts by printing the title message. Some examples of reading operation are given in Appendix-2.

4. Programmer Unit (The Hardware)

The hardware required for this PROM programmer is rather simple and straight forward. The circuit diagram is given in Figure 1. In all, three sockets are used for different types of PROMs. The 40 pins DIP socket is for 8755 and 8755-A, one 24 pins DIP socket is for 2732 and the third one (24 pins DIP) is for 2716 and 2758 type PROMs. Address and data lines are made common for all the three sockets and are directly brought to a 25 pins male connector. Three address lines, A8, A9 and A10 are connected to three LEDs (number 1, 2 and 3) through buffers 4050 and 7406. The LEDs display the current page number that is being programmed.

In addition to these address and data lines, there are four control lines, viz. ALE, Prog, VPP/VDD and $\overline{RD}/\overline{OE}$. Most of these lines are also common for all the sockets. All these lines are pulled FALSE using 33 K resistances to avoid any spurious signal due to pickup when the input is not connected to the port or the port is tri-stated.

Two voltage level shifters are used to generate programming voltage VPP/VDD. Two separate level shifters are required because for some of the PROMs (viz. 8755-A, 2716 and 2758) the VPP/VDD level should be +5V during reading cycle and for other PROMs (8755 and 2732) it should be 0 volt during reading cycle. For programming 8755 or 8755-A one should carefully check that jumper J1 is for

8755-A and the jumper J2 is for 8755. Also one should be careful to check that only one PROM can be programmed at a time and the rest of the two sockets should be free.

Figure 2 shows the connection of 25 PIN connector to the SDK-85 ports.

5. Software Considerations

The 'EPROM PROGRAMMER' is written to be executed at 8000 (Hex) location. The complete program is divided into one main and 8 subroutines. A list and function of all these routines is given in Table-3. In addition to these routines some subroutines of SDK-85 monitor are also used to avoid any duplication. Table 4 gives a list of the monitor routines used by this program. Detailes of the SDK-85 monitor program is given in user manual SDK-85.

While execution, the main program receives all the information from the console (like addresses, PROM type etc) and stores it in the reserved RAM location (see Table 1). When the final clearance is received from the operator, the programmer programmes one byte at a time by calling the appropriate WRITE BYTE routine and then reads it back by calling the READ BYTE routine for verification. In case of no error, the program continues and goes for the next byte. Ofcourse before going to the next byte, it checks if the final address of the source field is reached.

Data, address and the control pulses are sent to the programming unit through four ports (see Table 2). While the control pulses for 2716, 2758 and 2732 PROMs are straight forward the 8755 family needs special consideration because its data bus is multiplexed with a part of the address bus. For this an additional pulse is generated to serve as AlE for the PROM.

A complete listing of the 'EPROM PROGRAMMER' programme is given in Appendix.3.

TABLE - 1

'EPROM PROGRAMMER' RESERVED RAM LOCATIONS

Location		Contents
20BC - 20A0	..	Stack area
20B3 - 20B5	..	Jump to READ BYTE routine
20B6 - 20B8	..	Jump to WRITE BYTE routine
20B9 - 20BA	..	Source field initial address
20BB - 20BC	..	Source field final address
20BD - 20BE	..	Destination field initial address

Table -2 : I/O PORT ASSIGNMENT

PORT NO.	Function
00	D0-D7 (Data bus for 2716, 2758 and 2732)
21	A0-A7 (Low order address bus for 2716, 2758 & 2732) AD0-AD7 (Low order address/data bus for 8755 & 8755A)
22	A8-A12 (High order address bus)
23	Control bus (for all the rchips) (LSB) - RD - VDD/VPP - Prog.Pulse - AlE (8755, 8755-A)

TABLE-3 : 'EPROM PROGRAMMER' Routines

Name	ADDRESS	DESCRIPTION
PRNTMSG	8107	A string of ASCII characters beginning at the memory location pointed by HL is printed. The end of the string is recognized by the valid delimiter ('%'). Register A,C,H,L and F's affected.
PRTERR	8113	This routine prints the error message. Reg. A, C, H, L and F's affected.
GETHX	82A8	A string of hex digits is accepted from the input stream and their value as a 16 bit integer is returned to the calling program through BC. If more than 4 hex digits are entered, only the last four are accepted. The number terminates when 'CR' is encountered. In case the first character is not 'CR' and all the characters entered are valid hex digits then carry is set true otherwise it is set false. Reg. A, B, C, D, E and F's affected.
WRITE BYTE (8755, 8755-A)	82E1	One data byte received through register D is programmed in 8755 or 8755-A at address pointed by HL register pair. ACC and CPU condition codes are affected.
READ BYTE (8755,8755A)	8315	One byte is read from the memory location in the PROM pointed by register pair HL. Data returned through acc. Only ACC affected.
READ BYTE (2716,2758, 2732).	833E	Same as 'READ BYTE' (8755, 8755-A)'
WRITE BYTE (2716,2758)	835B	One byte of data received through register D is programmed in 2716 or 2758 at memory location pointed by HL.Acc and CPU flags are affected.
WRITE BYTE 2732	838B	One byte of data received through register D is programmed in 2732 at memory pointed by HL. Acc. and CPU flags affected.

Table-4 : SDK-85 Monitor routines used by the 'EPROM PROGRAMMER

Name	Address	Description
CROUT	05EB	CROUT sends a carriage return (and hence a line feed) to the console, Register A, B, C and F's affected.
DELAY	05F1	This routine takes the 16-bit contents of register pair DE and counts down to zero, then returns to the calling program. Register A, D, E & F's affected.
FRET	06IC	FRET sets the carry flag FALSE and then returns. Only carry affected.
GETCH	06IF	GETCH returns the next character in the serial input stream through register C to the calling program. Register A, C and F's affected.
HILO	06A0	This routine compares the two 16 bit integers in HL and DE. The carry bit is set FALSE if DE > HL.
NMOUT	06C7	NMOUT converts the 8 bit, unsigned integer in the A register into 2 ASCII characters. These two characters are sent to the console at the current print position of the console. Register A, B, C and flags affected.
SRET	0732	The carry flag is set TRUE and then returns to the calling program. Only carry affected.
CO	07FA	CONSOLE OUTPUT. This routines transmits a character (in ASCII code), passed from the caller in the C register, to the console. Register A, C and F's affected.

Acknowledgement

Authors thank Mr.R.I. Patel and Mr.H.D. Parikh for technical help.

References

1. Intel Data Catalog, 1977.
2. Intel SDK-85 System Design Kit User's Mannual, 1978.
3. Intel, MCS-80/85 Family User's Mannual, 1979.

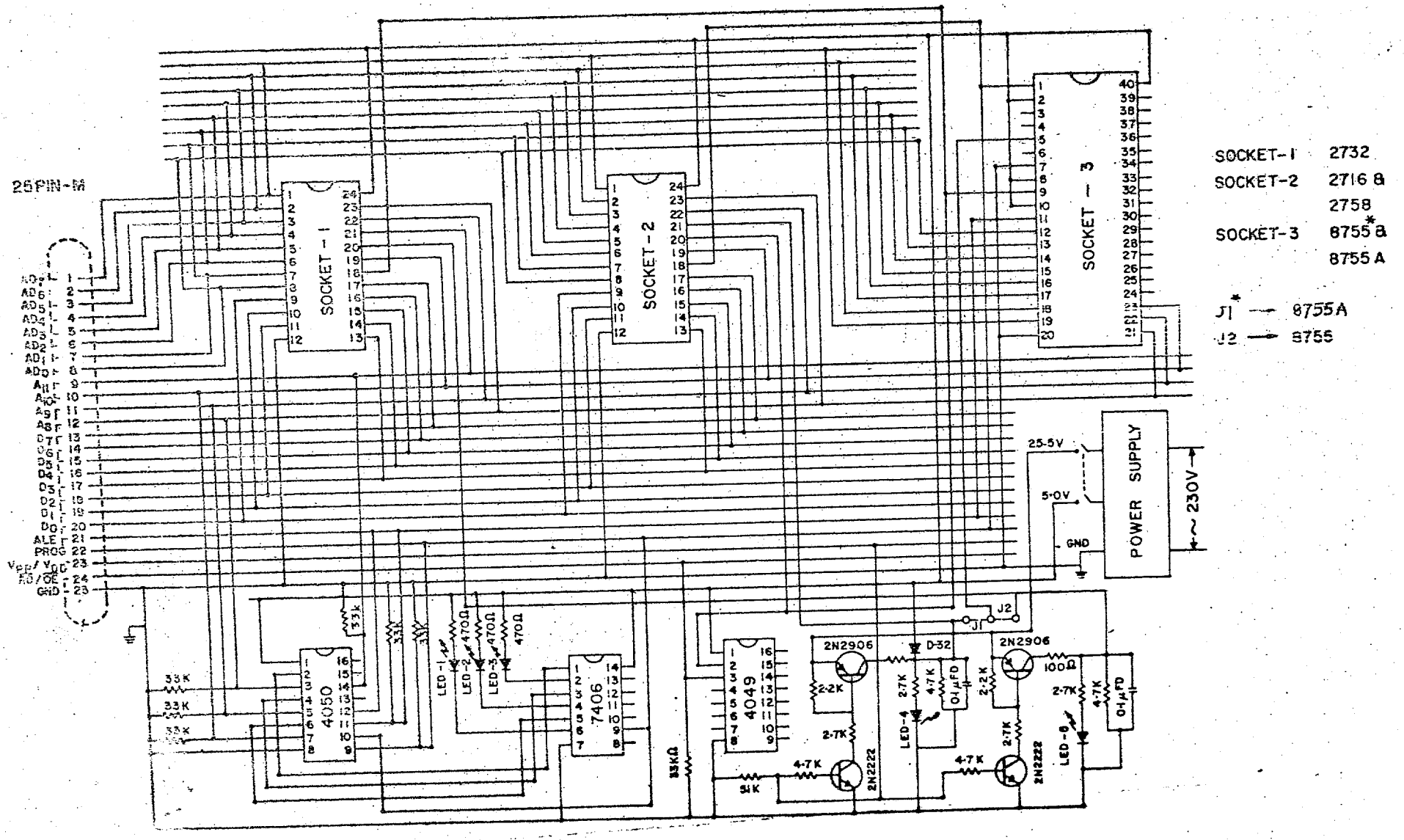
25PIN-M
SOCKET - 1
SOCKET - 2
SOCKET - 3
SOCKET-1 2732
SOCKET-2 2716 &
2758
SOCKET-3 8755 A*
8755 A
J1* — 8755A
J2 — 8755
AD7
AD6
AD5
AD4
AD3
AD2
AD1
AD0
A11
A10
A9
A8
D7
D6
D5
D4
D3
D2
D1
D0
ALE
PROG
Vpp/VDD
RD/OE
GND
25-5V
5-0V
GND
POWER SUPPLY
~ 230V
J1
J2
33K
33K
33K
LED-1
LED-2
LED-3
470Ω
470Ω
470Ω
4050
7406
4049
33KΩ
51K
2N2906
2N2222
2-2K
2-7K
4-7K
D-32
LED-4
0·1μFD
100Ω
LED-5

FIG. 1

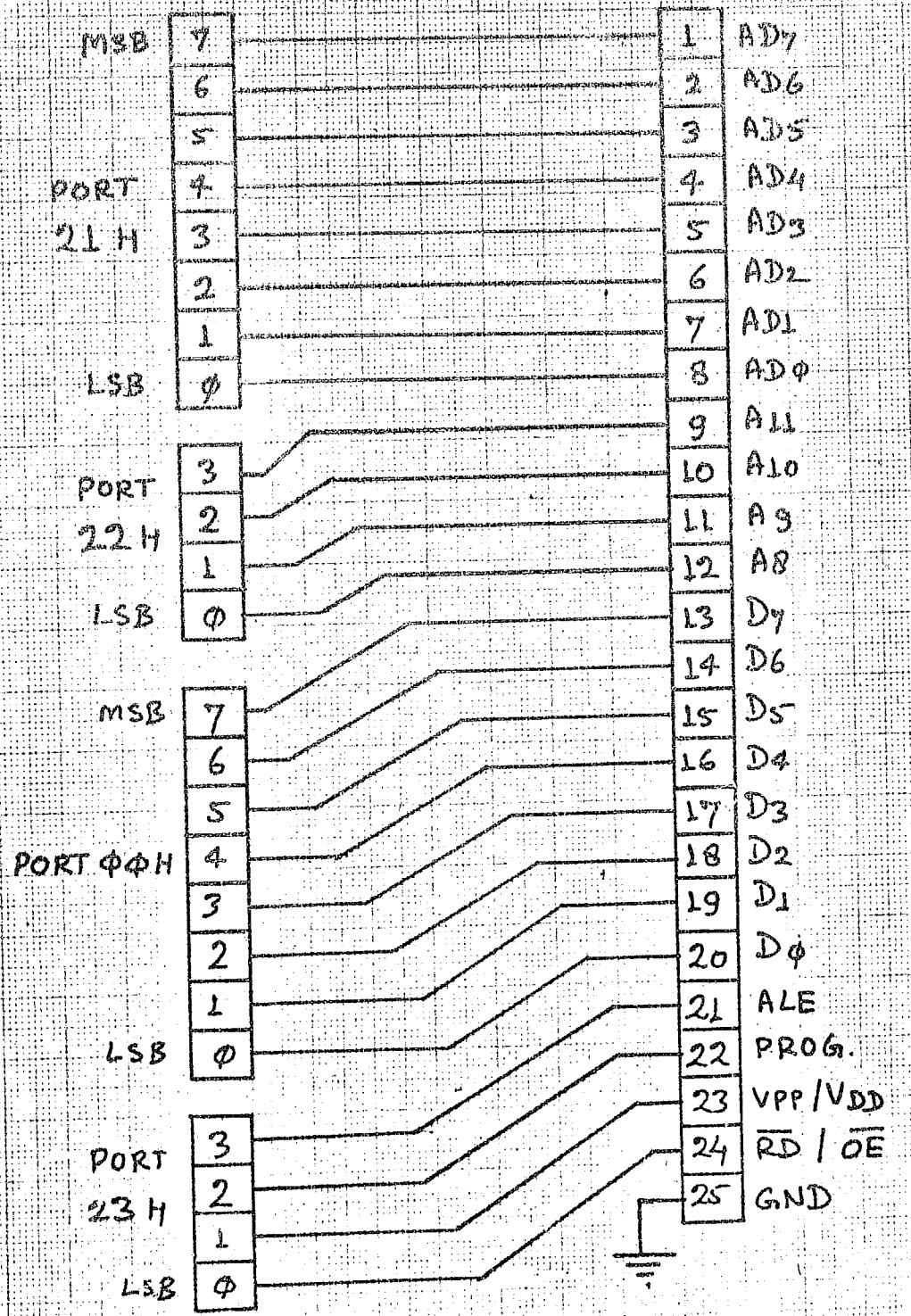

FIG. 2 PORTS CONNECTIONS

APPENDIX-1 :-

SOME PROGRAMMING EXAMPLES.

@8000

EPROM PROGRAMMER (PLT-AS-1982)

EPROM TYPE#2716

SOURCE FIELD INITIAL ADDR.(HEX)?2800

SOURCE FIELD FINAL ADDR.(HEX) ?2830

P , R ?P
DESTINATION (EPROM) INITIAL ADDR.(HEX)?0000

EXEC (Y/N)?Y

PROGRAMMING ERROR IN EPROM AT 0000
CONTINUE (Y/N)?Y

PROGRAMMING ERROR IN EPROM AT 0001
CONTINUE (Y/N)?N
EPROM PROGRAMMER (PLT-AS-1982)

EPROM TYPE#27165

ERROR
EPROM PROGRAMMER (PLT-AS-1982)

EPROM TYPE#2716

SOURCE FIELD INITIAL ADDR.(HEX)?002800

SOURCE FIELD FINAL ADDR.(HEX) ?2000

ERROR
SOURCE FIELD FINAL ADDR.(HEX) ?2830

P , R ?P
DESTINATION (EPROM) INITIAL ADDR.(HEX)?0000

EXEC (Y/N)?Y

** DONE **

EPROM PROGRAMMER (PLT-AS-1982)

EPROM TYPE#

APPENDIX- 2 :

SOME READING EXAMPLES.

```
SDK-85   VER 2.1
*08000

EPROM PROGRAMMER (PLT-AS-1982)

EPROM TYPE#7

ERROR
EPROM PROGRAMMER (PLT-AS-1982)

EPROM TYPE#8755

SOURCE FIELD INITIAL ADDR.(HEX)?0000

SOURCE FIELD FINAL ADDR.(HEX)  ?0030

P , R ?R
0000 2A BE 20 CD EB 05 7C CD C7 06 7D CD C7 06 06 04
0010 CD 14 09 7E 57 CD C7 06 7A FE 40 DA 75 08 7A FE
0020 C0 D2 71 08 06 08 CD 14 09 7A FE 76 C2 38 08 21
0030 4B
** DONE **

EPROM PROGRAMMER (PLT-AS-1982)

EPROM TYPE#2716

SOURCE FIELD INITIAL ADDR.(HEX)?0000

SOURCE FIELD FINAL ADDR.(HEX)  ?0020

P , R ?F
ERROR
P , R ?R
0000 FB 7D FE 27 DF 70 5F DD F7 C7 F3 58 FD D7 D4 BC
0010 D7 E3 5E BC 64 CE 5E FF BF D8 D1 DC F5 76 9A 17
0020 D5
** DONE **

EPROM PROGRAMMER (PLT-AS-1982)

EPROM TYPE#
```

APPENDIX-3

Listing of the 'EPROM PROGRAMMER' Program.

Total memory requirement 03C2 (Hex) bytes.

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

**  EPROM PROGRAMMER  **

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

8000    31A020      LXI SP,20A0; DEFINE STACK POINTER.
8003    C31A81      JMP 811A    ; BRANCH TO LEAVE SOME ROOM FOR
                                ; MESSAGE TABLE.

********  MESSAGE TABLE  ************

8006    0D0A 4550 524F 4D20    ; (CR)(LF)EPROM PROGRAMMER
800E    5052 4F47 5241 4D4D    ;  (PLT-AS-1982)(CR)
8016    4552 2028 504C 542D    ; (LF)(LF)EPROM TYPE#%
801E    4153 2031 3938 3229    ;
8026    0D0A 0A45 5052 4F4D    ;
802E    2054 5950 4523 25      ;

8035    0D0A 534F 5552 4345    ; (CR)(LF)SOURCE FIELD
803D    2046 4945 4C44 2049    ;  INITIAL ADDR.(HEX)?%
8045    4E49 5449 414C 2041    ;
804D    4444 522E 2848 4558    ;
8055    293F 25                ;

8058    0D0A 534F 5552 4345    ; (CR)(LF)SOURCE FIELD
8060    2046 4945 4C44 2046    ;  FINAL ADDR.(HEX)  ?%
8068    494E 414C 2041 4444    ;
8070    522E 2848 4558 2920    ;
8078    203F 25                ;

807B    0D0A 5020 2C20 5220    ; (CR)(LF)P , R ?%
8083    3F25                   ;

8085    0D0A 4445 5354 494E    ; (CR)(LF)DESTINATION
808D    4154 494F 4E20 2845    ;  (EPROM) INITIAL ADDR.(HEX)?%
8095    5052 4F4D 2920 494E    ;
809D    4954 4941 4C20 4144    ;
80A5    4452 2E28 4845 5829    ;
80AD    3F25                   ;

80AF    0D0A 4558 4543 2028    ; (CR)(LF)EXEC (Y/N)?%
80B7    592F 4E29 3F25         ;
```

```
80BD    0D0A 5052 4F47 5241   ; (CR)(LF)PROGRAMMING
80C5    4D4D 494E 4720 4552   ;  ERROR IN EPROM AT%
80CD    524F 5220 494E 2045   ;
80D5    5052 4F4D 2041 5420   ;
80DD    25                    ;

80DE    0D0A 434F 4E54 494E   ; (CR)(LF)CONTINUE (Y/N)?%
80E6    5545 2028 593F 41     ;
80EE    3F25                  ;

80F0    0D0A 4552 524F 5225   ; (CR)(LF)ERROR%

80F8    0D0A 2A2A 2044 4F4E   ; (CR)(LF)** DONE **
8100    4520 2A2A 0D0A 25     ; (CR)(LF)%
```

```
FUNCTION : PRNTMSG
DESCRIPTION : A STRING OF ASCII CHARACTERS BEGINING
            : AT THE MEMORY LOCATION POINTED BY HL
            : REGISTER PAIR IS PRINTED. THE END
            : OF THE STRING IS RECOGNIZED BY THE
            : VALID DELIMITER ("%" IN THIS CASE).
            : REG. A, C, H, L AND F'S AFFECTED.

8107    7E          MOV A,M     ; CHECK IF THE CHARACTER IS
8108    FE25        CPI 25      ; VALID DELIMITER.
810A    C8          RZ          ; RETURN IF YES.
810B    4E          C,M         ; OTHERWISE PRINT IT
810C    CDFA07      CALL 07FA   ; BY CALLING 'CO'.
810F    23          INX H       ; NEXT MEMORY LOCATION.
8110    C30781      JMP 8107    ; BRANCH TO HANDLE NEXT CHARACTER.


FUNCTION : PRTERR
DESCRIPTION : THIS ROUTINE PRINTS THE ERROR MESSAGE.
            : REG. A, C, H, L AND F'S AFFECTED.

8113    21F080      LXI H,80F0  ; DEFINE ERROR MSG POINTER.
8116    CD0781      CALL 8107   ; PRINT IT BY CALLING 'PRNTMSG'.
8119    C9          RET         ; RETURN.
```

```
FUNCTION : EPROM PROGRAMMER
DESCRIPTION: TO PROGRAM  ANY OF THE FOLLOWING EPROMS,
             : 8755, 8755A,2716, 2732, 2758.
-------------------------------------------------------------------------

811A    210680    LXI H,8006  ; SEND THE TITLE MESSAGE TO THE
811D    CD0781    CALL 8107   ; CONSOLE.
8120    CDA882    CALL 82A8   ; CALL GETHX TO RECEIVE THE TYPE NO.
8123    D23781    JNC 8137    ; CHECK IF ERROR
8126    78        MOV A,B     ; NO GO AHEAD.
8127    FE87      CPI 87      ; COMPARE THE EPROM TYPE
8129    C23281    JNZ 8132    ;
812C    79        MOV A,C     ;
812D    FE55      CPI 55      ;
812F    CA6B81    JZ 816B     ; BRANCH TO HANDLE 8755.
8132    FE27      CPI 27      ;
8134    CA3D81    JZ 813D     ; BRANCH TO HANDLE 27.. SERIES
8137    CD1381    CALL 8113   ; OTHERWISE ERROR MESSAGE.
813A    C31A81    JMP 811A    ; RESTART.
                              ;
813D    79        MOV A,C     ;
813E    FE16      CPI 16      ;
8140    CA5C81    JZ 815C     ; BRANCH TO HANDLE 2716.
8143    FE58      CPI 58      ;
8145    CA5C81    JZ 815C     ; BRANCH FOR 2758.
8148    FE32      CPI 32      ;
814A    C23781    JNZ 8137    ; BRANCH IF NOT 2732.
814D    213E83    LXI H,833E  ; SAVE THE READ BYTE ADDR. OF 2732
8150    22B420    SHLD 20B4   ;
8153    218B83    LXI H,838B  ; SAVE THE WRITE BYTE ADDR. OF 2732.
8156    22B720    SHLD 20B7   ;
8159    C37781    JMP 8177    ;
                              ;
815C    213E83    LXI H,833E  ; SAVE THE READ BYTE ADDR. FOR 2716.
815F    22B420    SHLD 20B4   ;
8162    215B83    LXI H,835B  ; SAVE THE WRITE BYTE ADDR. FOR 2716
8165    22B720    SHLD 20B7   ;
8168    C37781    JMP 8177    ;
                              ;
816B    211583    LXI H,8315  ; SAVE THE READ BYTE ADDR. FOR 8755.
816E    22B420    SHLD 20B4   ;
8171    21E182    LXI H,82E1  ; SAVE THE WRITE BYTE ADDR. FOR 8755.
8174    22B720    SHLD 20B7   ;
8177    21B320    LXI H,20B3  ; SAVE JUMP INSTRUCTION  FOR READ BYTE
817A    36C3      MVI M,C3    ; AND WRITE BYTE ROUTINES AT RESEARVED
817C    21B620    LXI H,20B6  ; RAM LOCATIONS.
817F    36C3      MVI M,C3    ;
8181    213580    LXI H,8035  ; SEND INITIAL ADDR. MESSAGE.
8184    CD0781    CALL 8107   ;
8187    CDA882    CALL 82A8   ; RECEIVE THE ADDRESS.
818A    DA9381    JC 8193     ; BRANCH IF NO ERROR.
818D    CD1381    CALL 8113   ; HANDLE ERROR.
8190    C38181    JMP 8181    ;
                              ;
```

```
8193    60          MOV H,B     ; SAVE THE INITIAL ADDRESS AT
8194    69          MOV L,C     ; RESEARVED RAM LOCATION.
8195    22B920      SHLD 20B9   ;
8198    215880      LXI H,8058  ; SEND FINAL ADDR. MESSAGE.
819B    CD0781      CALL 8107   ;
819E    CDA882      CALL 82A8   ; RECEIVE THE ADDRESS.
81A1    DAAA81      JC 81AA     ; BRANCH IF NO ERROR.
81A4    CD1381      CALL 8113   ; OTHERWISE HANDLE ERROR.
81A7    C39881      JMP 8198    ; REPEAT THE MESSAGE.
                                ;
81AA    50          MOV D,B     ; CHECK IF THE FINAL ADDRESS
81AB    59          MOV E,C     ; IS >= INITIAL ADDRESS.
81AC    2AB920      LHLD 20B9   ;
81AF    EB          XCHG        ;
81B0    CDA006      CALL 06A0   ;
81B3    D2A481      JNC 81A4    ; ERROR IF NOT.
81B6    22BB20      SHLD 20BB   ; SAVE  THE FINAL ADDRESS.
81B9    217B80      LXI H,807B  ; SEND 'P,R?' MESSAGE.
81BC    CD0781      CALL 8107   ;
81BF    CD1F06      CALL 061F   ; RECEIVE THE ANSWER FROM CONSOLE.
81C2    F5          PUSH PSW    ; SAVE IT
81C3    00          NOP         ;
81C4    CDFA07      CALL 07FA   ; ECHO IT
81C7    F1          POP PSW     ;
81C8    FE50        CPI 50      ; BRANCH IF IT IS P(PROGRAM).
81CA    CAD881      JZ 81D8     ;
81CD    FE52        CPI 52      ;
81CF    CA7782      JZ 8277     ; BRANCH IF IT IS R(READ).
81D2    CD1381      CALL 8113   ; ERROR IF NONE.
81D5    C3B981      JMP 81B9    ; REPEAT THE MESSAGE.
                                ;
81D8    218580      LXI H,8085  ; SEND THE DESTINATION ADDRESS
81DB    CD0781      CALL 8107   ; MESSAGE.
81DE    CDA882      CALL 82A8   ; RECEIVE THE ADDRESS.
81E1    DAEA81      JC 81EA     ; BRANCH IF NO ERROR.
81E4    CD1381      CALL 8113   ; HANDLE ERROR.
81E7    C3D881      JMP 81D8    ;
                                ;
81EA    60          MOV H,B     ; SAVE THE DESTINATION ADDRESS.
81EB    69          MOV L,C     ;
81EC    22BD20      SHLD 20BD   ;
81EF    21AF80      LXI H,80AF  ; SEND 'EXEC(Y/N)?' MESSAGE.
81F2    CD0781      CALL 8107   ;
81F5    CD1F06      CALL 061F   ; RECEIVE ANSWER.
81F8    F5          PUSH PSW    ;
81F9    CDFA07      CALL 07FA   ; ECHO IT.
81FC    F1          POP PSW     ;
81FD    FE59        CPI 59      ;
81FF    CA0D82      JZ 820D     ; BRANCH IF Y(YES).
8202    FE4E        CPI 4E      ;
8204    CA0080      JZ 8000     ; RESTART IF N(NO).
8207    CD1381      CALL 8113   ; ERROR IF NONE.
820A    C3EF81      JMP 81EF    ; REPEAT THE MESSAGE.
```

```
820D   CDEB05    CALL 05EB   ; SEND 'CR' 'LF'.
8210   2AB920    LHLD 20B9   ; LOAD THE BYTE TO BE PROGRAMMED
8213   56        MOV D,M     ; IN TO REGISTER D.
8214   2ABD20    LHLD 20BD   ; LOAD THE DESTINATION ADDR. IN HL.
8217   CDB620    CALL 20B6   ; CALL 'BYTE WRITE'
821A   CDB320    CALL 20B3   ; CALL 'BYTE READ'
821D   BA        CMP D       ; VERIFY THE PROGRAMMED BYTE.
821E   CA5382    JZ 8253     ; BRANCH IF CORRECT.
8221   21BD80    LXI H,80BD  ; OTHERWISE PROG. ERROR MESSAGE.
8224   CD0781    CALL 8107   ;
8227   2ABD20    LHLD 20BD   ; SEND THE ADDRESS AT WHICH
822A   7C        MOV A,H     ; THE ERROR IS DETECTED.
822B   CDC706    CALL 06C7   ;
822E   7D        MOV A,L     ;
822F   CDC706    CALL 06C7   ;
8232   21DE80    LXI H,80DE  ; SEND 'CONTINUE(Y/N)' MESSAGE.
8235   CD0781    CALL 8107   ;
8238   CD1F06    CALL 061F   ; RECEIVE ANSWER.
823B   F5        PUSH PSW    ;
823C   CDFA07    CALL 07FA   ; ECHO IT.
823F   F1        POP PSW     ;
8240   FE59      CPI 59      ;
8242   CA5082    JZ 8250     ; BRANCH IF Y(YES).
8245   FE4E      CPI 4E      ;
8247   CA0080    JZ 8000     ; RESTART IF N(NO).
824A   CD1391    CALL 9113   ; ERROR IF NONE.
824D   C33282    JMP 8232    ; REPEAT THE MESSAGE.
                             ;
8250   CDEB05    CALL 05EB   ; SEND 'CR' , 'LF'.
8253   2ABB20    LHLD 20BB   ; CHECK IF THE FINAL SOURCE ADDRESS
8256   EB        XCHG        ; IS REACHED.
8257   2AB920    LHLD 20B9   ;
825A   CDA006    CALL 06A0   ;
825D   DA6E82    JC 826E     ; BRANCH IF NO.
8260   23        INX H       ; OTHERWISE INCREASE THE SOURCE
8261   22B920    SHLD 20B9   ; AND DESTINATION ADDRESS.
8264   2ABD20    LHLD 20BD   ;
8267   23        INX H       ;
8268   22BD20    SHLD 20BD   ;
826B   C31082    JMP 8210    ; GO TO PROGRAM THE NEXT BYTE.
                             ;
826E   21F880    LXI H,80F8  ; SEND THE 'DONE' MESSAGE.
8271   CD0781    CALL 8107   ;
8274   C30080    JMP 8000    ; RESTART.
                  FUNCTION : READ EPROM
8277   00        NOP         ;
8278   00        NOP         ;
8279   2AB920    LHLD 20B9   ; LOAD INITIAL ADDRESS OF SOURCE
827C   CDEB05    CALL 05EB   ; SEND 'CR','LF' .
```

```
827F    7C          MOV A,H       ; PRINT ADDRESS.
8280    CDC706      CALL 06C7     ;
8283    7D          MOV A,L       ;
8284    CDC706      CALL 06C7     ;
8287    0E20        MVI C,20      ; LEAVE ONE SPACE.
8289    CDFA07      CALL 07FA     ;
828C    CDB320      CALL 20B3     ; READ ONE BYTE DATA.
828F    CDC706      CALL 06C7     ;  PRINT IT.
8292    23          INX H         ; NEXT ADDRESS.
8293    EB          XCHG          ; CHECK IT WITH THE END ADDR.
8294    2ABB20      LHLD 20BB     ;
8297    23          INX H         ;
8298    EB          XCHG          ;
8299    CDA006      CALL 06A0     ;
829C    DA6E82      JC 826E       ; BRANCH IF THE END IS REACHED.
829F    7D          MOV A,L       ; CHECK IF LOWER HEX OF L IS F.
82A0    E60F        ANI 0F        ;
82A2    C28782      JNZ 8287      ; GO FOR NEXT BYTE.
82A5    C37C82      JMP 827C      ; NEXT LINE .
```

```
SUBROUTINE: GETHEX
DESCRIPTION : A STRING OF TWO HEX DIGITS ACCEPTED.
            : THEN RETURNED THROUGH BC.
82A8    E5          PUSH H        ;
82A9    210000      LXI H,0000    ;
82AC    1E00        MVI E,00      ;
82AE    CD1F06      CALL 061F     ;
82B1    4F          MOV C,A       ;
82B2    CDF805      CALL 05F8     ;
82B5    79          MOV A,C       ;
82B6    FE0D        CPI 0D        ;
82B8    C2C782      JNZ 82C7      ;
82BB    51          MOV D,C       ;
82BC    E5          PUSH H        ;
82BD    C1          POP B         ;
82BE    E1          POP H         ;
82BF    7B          MOV A,E       ;
82C0    B7          ORA A         ;
82C1    C23207      JNZ 0732      ;
82C4    CA1C06      JZ 061C       ;
82C7    CD5E07      CALL 075E     ;
82CA    D2DD82      JNC 82DD      ;
82CD    CD8805      CALL 0588     ;
82D0    1EFF        MVI E,FF      ;
82D2    29          DAD H         ;
82D3    29          DAD H         ;
82D4    29          DAD H         ;
82D5    29          DAD H         ;
82D6    0600        MVI B,00      ;
82D8    4F          MOV C,A       ;
82D9    09          DAD B         ;
82DA    C3AE82      JMP 82AE      ;

82DD    E1          POP H         ;
82DE    C31C06      JMP 061C      ;
```

```
FUNCTION:WRITE BYTE (8755,8755 A)
DESCRIPTION : ONE BYTE IS PROGRAMMED IN
              8755 OR 8755A. DATA THROUGH REG. D
                ADDRESS THROUGH HL REG. PAIR.

82E1    E5          PUSH H      ; SAVE ADDRESS.
82E2    D5          PUSH D      ; SAVE DATA.
82E3    3E0F        MVI A,0F    ; DEFINE PORTS AS OUTPUT.
82E5    D320        OUT 20      ;
82E7    7D          MOV A,L     ; OUT ADDRESS.
82E8    D321        OUT 21      ;
82EA    7C          MOV A,H     ;
82EB    D322        OUT 22      ;
82ED    3E08        MVI A,08    ; OUT ALE PULSE.
82EF    D323        OUT 23      ;
82F1    3E00        MVI A,00    ;
82F3    D323        OUT 23      ;
82F5    7A          MOV A,D     ; OUT DATA.
82F6    D321        OUT 21      ;
82F8    3E04        MVI A,04    ; SEND PROG LEVEL BIT HIGH.
82FA    D323        OUT 23      ;
82FC    3E06        MVI A,06    ; SET VDD=25V.
82FE    D323        OUT 23      ;
8300    118010      LXI D,1080  ; 50 MS DELAY.
8303    CDF105      CALL 05F1   ;
8306    3E04        MVI A,04    ; SET VDD = 5V.
8308    D323        OUT 23      ;
830A    3E00        MVI A,00    ; SET PROG LEVEL BIT LOW.
830C    D323        OUT 23      ;
830E    3E00        MVI A,00    ; REDEFINE PORTS AS INPUT.
8310    D320        OUT 20      ;
8312    D1          POP D       ; RESTORE DATA.
8313    E1          POP H       ; RESTORE ADDRESS.
8314    C9          RET         ; RETURN.

FUNCTION : READ BYTE (8755,8755 A)
DESCRIPTION : ONE BYTE IS READ FROM 8755,
              OR 8755A AND DATA RETURNED THROUGH REG. D
                ADDRESS THROUGH HL PAIR.
8315    E5          PUSH H      ; SAVE ADDRESS.
8316    D5          PUSH D      ; SAVE DATA.
8317    3E0F        MVI A,0F    ; DEFINE PORTS.
8319    D320        OUT 20      ;
831B    7D          MOV A,L     ; OUT ADDRESS.
831C    D321        OUT 21      ;
831E    7C          MOV A,H     ;
831F    D322        OUT 22      ;
8321    3E08        MVI A,08    ; OUT ALE PULSE.
8323    D323        OUT 23      ;
8325    3E00        MVI A,00    ;
8327    D323        OUT 23      ;
```

```
8329    3E0E        MVI A,0E    ; REDEFINE PORT 21 AS INPIT.
832B    D320        OUT 20      ;
832D    3E01        MVI A,01    ; SET CHIP ENABLE.
832F    D323        OUT 23      ;
8331    DB21        IN 21       ; READ BYTE.
8333    57          MOV D,A     ; TRANSFER DATA TO REG. D.
8334    3E00        MVI A,00    ; REDEFINE PORTS AS INPUT.
8336    D323        OUT 23      ; AND CHIP DISABLE.
8338    D320        OUT 20      ;
833A    7A          MOV A,D     ; RESTORE DATA TO ACC.
833B    D1          POP D       ; RESTORE EARLIER DATA.
833C    E1          POP H       ; RESTORE ADDRESS.
833D    C9          RET         ; RETURN.
```

FUNCTION : READ BYTE (2716, 2758, 2732)
DESCRIPTION : ONE BYTE IS READ FROM ONE OF THE ABOVE LISTED CHIPS. DATA RETURNED THROUGH REG. D. ADDR. HL

```
833E    E5          PUSH H      ; SAVE ADDRESS.
833F    D5          PUSH D      ; SAVE DATA.
8340    3E0F        MVI A,0F    ; DEFINE PORTS AS OUT PUT.
8342    D320        OUT 20      ;
8344    7D          MOV A,L     ; SET ADDRESS.
8345    D321        OUT 21      ;
8347    7C          MOV A,H     ;
8348    D322        OUT 22      ;
834A    3E01        MVI A,01    ; SET CHIP ENABLE.
834C    D323        OUT 23      ;
834E    DB00        IN 00       ; READ THROUGH PORT 00.
8350    57          MOV D,A     ;
8351    3E00        MVI A,00    ; REDEFINE PORTS.
8353    D323        OUT 23      ;
8355    D320        OUT 20      ;
8357    7A          MOV A,D     ;
8358    D1          POP D       ; RESTORE DATA.
8359    E1          POP H       ; RESTORE ADDRESS.
835A    C9          RET         ;
```

FUNCTION : WRITE BYTE (2716, 2758)
DESCRIPTION : DATA RECEIVED THROGH D IS PROGRAMMED IN 2716 OR 2758. ADDRESS THROUGH HL PAIR.

```
835B    E5          PUSH H      ;
835C    D5          PUSH D      ;
835D    3E0F        MVI A,0F    ;
835F    D320        OUT 20      ;
8361    3EFF        MVI A,FF    ; DEFINE PORT 00 TO BE OUTPUT
8363    D302        OUT 02      ; PORT.
8365    7A          MOV A,D     ; SET DATA.
8366    D300        OUT 00      ;
8368    7D          MOV A,L     ; SET ADDRESS.
8369    D321        OUT 21      ;
836B    7C          MOV A,H     ;
836C    D322        OUT 22      ;
```

```
836E    3E02        MVI A,02    ; SET VPP = 25V.
8370    D323        OUT 23      ;
8372    3E06        MVI A,06    ; SET PGM HIGH.
8374    D323        OUT 23      ;
8376    118018      LXI D,1880  ; 50 MS DELAY.
8379    CDF105      CALL 05F1   ;
837C    3E02        MVI A,02    ; SET PGM LOW.
837E    D323        OUT 23      ;
8380    3E00        MVI A,00    ; SET VPP = 5V.
8382    D323        OUT 23      ;
8384    D320        OUT 20      ; REDEFINE PORTS.
8386    D302        OUT 02      ;
8388    D1          POP D       ; RESTORE DATA.
8389    E1          POP H       ; RESTORE ADDRESS.
838A    C9          RET         ;
```

FUNCTION : WRITE BYTE (2732).
DESCRIPTION : SIMILAR TO 2716, WITH A MINOR DIFFERENCE.

```
838B    E5          PUSH H
838C    D5          PUSH D      ;
838D    3E0F        MVI A,0F    ;
838F    D320        OUT 20      ;
8391    3EFF        MVI A,FF    ;
8393    D302        OUT 02      ;
8395    7A          MOV A,D     ;
8396    D300        OUT 00      ;
8398    7D          MOV A,L     ;
8399    D321        OUT 21      ;
839B    7C          MOV A,H     ;
839C    D322        OUT 22      ;
839E    3E02        MVI A,02    ; SET VPP = 25V.
83A0    D323        OUT 23      ;
83A2    3E03        MVI A,03    ; SET CHIP ENABLE.
83A4    D323        OUT 23      ;
83A6    118018      LXI D,1880  ; 50 MS. DELAY.
83A9    CDF105      CALL 05F1   ;
83AC    3E02        MVI A,02    ; CHIP DISABLE.
83AE    D323        OUT 23      ;
83B0    3E00        MVI A,00    ; SET VPP = 0V.
83B2    D323        OUT 23      ;
83B4    110002      LXI D,0200  ; 5 MS. DELAY FOR VPP TO COME
83B7    CDF105      CALL 05F1   ; DOWN.
83BA    3E00        MVI A,00    ; REDEFINE PORTS.
83BC    D320        OUT 20      ;
83BE    D302        OUT 02      ;
83C0    D1          POP D       ; RESTORE DATA.
83C1    E1          POP H       ; RESTORE ADDRESS.
83C2    C9          RET         ; RETURN.
```