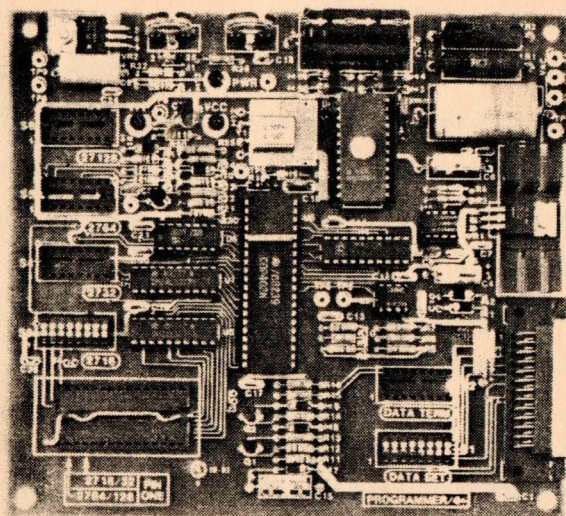


EPR0M Programmer



LEGACY

Computer Systems

26356 CARMEL RANCHO LANE • CARMEL, CA 93923 • 408-625-6562

INTRODUCTION

PROGRAMMER 4+ is a general purpose EPROM Programmer with considerable flexibility. Its' features are summarized as follows:

- FEATURES

- * Programs 2716, 2732, 2764, and 27128 EPROMs.
- * Direct connection to any RS232C terminal or computer.
- * Plug selectable as either a data set or data terminal.
- * All DC voltages are generated on the board.
- * Selectable programming voltage, 21/25VDC.
- * Power electronically switched to EPROM socket.
- * Zero insertion force (ZIF) socket for EPROM.
- * Programs, verifies and dumps in both ASCII and HEX formats.
- * Software provided for CP/M, with others; forthcoming.
- * Software on SS/SD disk, with source code.
- * Completely built and tested. (With dynamic burn-in)

This manual provides complete information about PROGRAMMER 4+ so that the user can take full advantage of these capabilities. The areas covered are Specifications, System Requirements, Installation, Operation, and Documentation.

Legacy Computer Systems warrants the PROGRAMMER 4+ to be free from parts and manufacturing defects for a period of 90 days. This warranty shall be void if the unit has been modified or subjected to improper or abnormal use. Legacy will adjust, repair or replace, at their discretion, any unit which is returned.

PROGRAMMER 4+ is a registered trademark of Legacy Computer Systems. Radio Shack is a registered trademark of Tandy Corp. CP/M is a registered trademark of Digital Research Corp.

SPECIFICATIONS**Physical**

PC board 6" X 6.5", Double sided, Plated through, Screened, and Solder masked. 4 corner mounting holes 3/16" dia.

Ambient Temp. 10 to 40 Degrees C. (25 \pm 10 During programming)

Power Reqmts. 25.2 VAC CT 60Hz, 20 Watts Max.

Indicators LED's for PWR, Vpp on, and Vcc on.

Adjustments 21V and 25V pots for Vpp calibration.

Test points TP1-TP9 provided to monitor board operation.

Interface

Electrical Serial RS232C. Handshake control signals are CTS and RTS.

Connector 25 pin female EIA, right angle PC mount.

Data Format Asynchronous bit serial; 1 start, ~~8~~⁷ data, and 1 stop

Character code ASCII upper case. (Upper and lower supported with Host software package)

Baud rate 1200 Baud fixed

Device type Data set or Data terminal, PLUG selectable.

Programming

EPROM types 2716, 2732, 2732A, 2764, and 27128. (PLUG selectable)

EPROM voltages Electrically switched ON and Off. Vpp is selectable for 21 VDC or 25 VDC by BERG style plug. Vpp and Vcc are regulated.

Program Method Fixed pulse width of 50 Millisec. with auto verify. Program time is 3 Min. - 10 Sec. for 2716.

SYSTEM REQUIREMENTS

PROGRAMMER 4+ can be used with either a standard computer terminal (stand alone mode) or as a peripheral device (computer mode). In either case, an interface cable and a low voltage AC supply is required.

The interface cable is typically a pin to pin cable with male DB25P connectors on each end. The PROGRAMMER 4+ requires the use of CTS and RTS signals for data handshaking when used as a peripheral device.

The recommended power transformer is a Radio Shack catalog number 273-1512. It is 25.2 VAC CT at 2 Amps. The user is responsible for the construction of the AC supply and is advised to use appropriate safeguards against electrical shock.

When PROGRAMMER 4+ is used as a peripheral on a host system, certain system features are necessary to insure proper operation. The host must have a serial port that can operate at 1200 Baud. The host software supplied assumes the users' operating system is CP/M 2.2 and supports BDOS calls to PUN and RDR. The serial port hardware will typically be one of the common LSI chips that support RTS and CTS. If these requirements are met, no special installation should be necessary for the software supplied. The source code is also included if special changes are required by the user.

INSTALLATION

Install PROGRAMMER 4+ by completing the following steps:

1. Construct the low voltage AC supply using the recommended power transformer (Radio Shack 273-1512) and referring to the diagram in the Documentation section. Be sure to insulate the Primary connections to prevent accidental ELECTRICAL SHOCK. The Primary should also be fused as indicated.
2. Apply power to the AC supply and verify that the secondary is providing 12.6 VAC on each side of center tap. Disconnect the supply from the AC line and connect the 3 secondary leads to PROGRAMMER 4+. See the documentation section for the solder pad locations.
3. Connect the supply to the AC line and switch on the power. The pwr LED indicator should be on. The other 2 LEDS should be off. After power has been applied for a while, check the two voltage regulators to see if they are both just noticeably warm. At this point the Vpp voltage calibrations may be checked if desired. Use a DC voltmeter to measure TP8 with respect to gnd (TP4). Measure $21 \pm .5v$ or $25 \pm .5v$. The BERG type jumper J1 determines which voltage is selected. If adjustments are required, set the 21v first using R19. then set 25v with R33. (order of setting is important).
4. Construct or purchase the serial interface cable. If the host system has a standard RS232 connector for its' port it will be a female 25 pin "D" connector (DB25S). The interface cable will then require male connectors on each end (DB25P). If a cable is to be constructed, wire it pin to pin. The required wires are pins 1 thru 8, and 20.
5. With power off both the host system and PROGRAMMER 4+, connect the serial interface cable. At this point, install the PLUG header in S1 or S2. S1 makes PROGRAMMER 4+ look like a DATA SET and S2 makes it look like a DATA TERMINAL. The idea is to set PROGRAMMER 4+ to be the opposite device of the host serial port. Data sets connect to data terminals and vice versa. If the host port is an unknown type, leave the PLUG in the data set position for the first try. If it is incorrect, PROGRAMMER 4+ will not function but no damage to either port will occur.

6. Apply power to the host system and PROGRAMMER 4+. Boot up the host and BACK UP the host software disk supplied with PROGRAMMER 4+. Put the backup disk in an available drive and log it on. Call in the host program by typing PROG4<cr>. If all is well, a menu should appear on the screen with a ":" in the lower left below the menu. The colon will appear only if Host software was able to contact PROGRAMMER 4+ via the serial port. If a colon is seen, go on to the operation section of this manual.
7. No colon appeared! Try these tips. The host system serial port must be operating at 1200 baud. The jumper PLUG on PROGRAMMER 4+ must be in the correct position (S1 or S2). The host CP/M BIOS must support PUN and RDR BDOS calls. The host serial port supports RTS and CTS. In order for PROGRAMMER 4+ to send data, pin 14 of Data set PLUG or Terminal PLUG must be +volts. It sends this data from pin 15 on the PLUG. When PROGRAMMER 4+ needs to stop data flow from host, it will change the state of pin 13 on the PLUG header to -volts. This is its' "hey CPU, I'm busy with the last data" signal. This signal must connect to the hand shake control of the host CPU port. If host CPU is a Terminal type this is typically it's CTS signal. You will find that the Host CPU will not send data to PROGRAMMER 4+ unless the hand shaking signal is in the proper state. PROGRAMMER 4+ receives data on pin 16 of the PLUGS.
8. No luck!! Contact your dealer for assistance.

OPERATION

The operation of PROGRAMMER 4+ in either direct mode or computer mode is straightforward as both modes are menu driven. The commands will be discussed in this section to provide more detailed information than is practical to put in the menus. Before giving the details of each command, here are some general statements about using them. All commands are terminated with a <CR> except the Help and Xchange menus commands. Some commands have [optional] parameters. The [] indicate an optional entry and should not be typed in with the parameter. The commands provide an interface between the host CPU system and a 16K byte RAM buffer. They also provide an interface between the buffer and the PROGRAMMER 4+. All programming and reading of EPROMs occurs between buffer and the EPROM socket. All loading and saving of disk files occurs between buffer and disk. With this in mind, here are the commands:

Computer mode

The following group of commands can only be used when PROGRAMMER 4+ is interfaced as a peripheral with a host computer system and running the PROG4 program.

Program buffer to EPROM

P<CR>

When this command is invoked the operator will be informed of what type of EPROM the PROGRAMMER 4+ is set for and how to Start or Abort the operation. If the operator elects to Start, the contents of the buffer will be programmed and verified, starting always from the first location of the buffer. Since the buffer is usually larger than the EPROM type selected, the operation will automatically stop when the first EPROM's worth of buffer has been done. At this point the operator will be prompted again to start or abort. If there is more data in the buffer to program the operator removes the first EPROM and inserts the next and starts the operation. This will cause the next section of the buffer to be programmed. This can continue until all 16K bytes of buffer has been programmed. Obviously, if only one EPROM is to be programmed, the operator aborts the command when the second prompt occurs. Another feature of this command is the "abort during programming" capability. The user may request that programming be aborted by depressing the space key. PROGRAMMER 4+ programs in complete 16 byte blocks and checks for abort after each 16 is finished. Therefore, the response to the space key is not instantaneous but occurs at the next time 16 bytes are finished.

Read EPROM to buffer

R<CR>

This command works the same way as the Program command except that the direction is reversed. That is, data is read from the EPROM and placed in the buffer. It should be clear that copying an EPROM would be done by using the Read command on the pre-programmed EPROM and the Program command on the unprogrammed EPROM. Due to the fact that both commands assume the entire buffer can be used, one can easily duplicate a set of EPROMs as well. Further, the R and P commands are independent of each other so the operator could read in a set of 2716 EPROMs and program the buffer out to 2732 EPROMs. Note that the abort during Read works here as well. Hit the space key during Read and Programmer 4+ will stop after the next 16 bytes are read.

Test an EPROM for erase

T<CR>

This command will quickly read through an EPROM and determine if it is erased or unerased. This command is usually used before the programming of a set of EPROMs is begun. The response to the operator is simply "EPROM erased" or "EPROM not erased". If more information about the unerased EPROM is desired see the Direct mode commands. It should be mentioned here that it is not mandatory to erase check the EPROMs before programming because each byte is read back and verified as it is programmed and the programming operation will stop if a no verify occurs.

Query EPROM type

Q<CR>

This command checks PROGRAMMER 4+ to see where the EPROM type jumper PLUG is inserted. It reports to the operator the EPROM type that is selected. It is a good practice to perform this command after changing the PLUG to be sure you are in the right position.

Load file to buffer

L[d:]<Filename>[.ext]<CR>

The Load command allows the operator to read a disk file to the buffer. As can be seen from the syntax, a filename is mandatory. A drive specification is optional and if left out will cause the default drive to be used. The extent on the name must agree with the directory entry of the file and cannot be left out of the command if the file does not have one. However, the Load command will always treat the file as a memory image if the extent is not ".HEX". If the file has an extent of ".HEX" it

will be converted to a memory image as it is loaded. The Address specified in the first ".HEX" data record is assumed to be the lowest address of the file. The Load command will position it in the buffer so that this data starts at the beginning of the buffer. It is the users responsibility to see that files are not larger than the available buffer (16K bytes).

Save buffer to disk

S[d:]<Filename>[.ext]<CR>

The Save command works very much like the Load command but is in the opposite direction. That is, it writes the data in the buffer to a disk file. The drive specification is optional as is the extent. If the extent is made ".HEX" the Save command will write an Intel format ".HEX" file to the disk using the actual buffer start address as the load address of the file. This causes no problem when the file is later read to the buffer using the Load command. The Load command is clever enough to place it at the start of the buffer. If the user decides to bring the Saved ".HEX" file into TPA with DDT for modification, be aware that the I(file) and R(offset) commands will have to be used. Another caution is needed. The Save command does not know how much of the buffer has valid data in it so it always saves all 16K bytes. Be sure you have enough space on the disk to save it! If the user is saving in ".HEX" format the file size is in excess of 40K bytes!! If a file is Saved that won't fit on the disk, no overwriting of other files will occur but the user will not be notified of the situation. Since the files are always a fixed length (be they hex or image) it is a simple matter to STAT the file to be sure the length is correct.

Display buffer to console

D[offset]<CR>

The Display buffer command allows the operator to inspect the contents of the buffer. An optional offset parameter is provided to allow starting the Display down in the buffer if desired. When this command is started it will automatically pause after sufficient data to fill the screen has been displayed. the Display may be resumed by depressing the space bar. If the operator wants to quit Displaying data, the CR key will abort the command. The presentation on the screen includes both a HEX data dump and an ASCII dump if codes are printable. A nice header is also displayed for each page that helps correlate the ASCII and HEX fields.

Fill buffer with FF

F<CR>

This command provides the user with the ability to initialize the buffer with the erased state of an EPROM (FF). The buffer is automatically filled with FF when the PROG4 program is first executed. IT IS NOT FILLED AGAIN UNLESS THIS COMMAND IS ISSUED.

WARNING: Remove EPROM from ZIF socket before power is applied to or removed from PROGRAMMER 4+.
It is possible to Glitch the EPROM with Vpp as 5V collapses.

Direct commands

The following commands are usable in a stand alone mode where only a terminal is available. In this case commands are terminated with Space key and abort is the CR key. These commands operate as described below when computer system is used in direct mode with PROGRAMMER 4+.

Test EPROM for erase

T[Address]<CR>

This command works very much like the Test command in computer mode. The difference is that this command allows an optional starting address and can report more information than the other. When it finds a location that is not FF the address of the location and it's non FF contents will be displayed. Use this command to find available areas of an EPROM that has some data in it already. The area is found by using several Test commands with the offset parameter. The Test operation always begins at the offset and terminates on the next non FF location or the end of the EPROM. Note that it does not report anything if EPROM is erased. All the user will observe is a delay while it checks the EPROM and a return to prompt.

Query EPROM type

Q<CR>

The Q command operates like the computer mode Q except for the report. This command reports a number that represents the EPROM type. 0,1,2, and 3 correspond to 2716, 2732, 2764, and 27128.

See ASCII data in EPROM

S[address]<CR>

This command provides a display of ASCII messages on the console device. If the code is not a printable one it is displayed as ".". Use this command for a better look at ASCII messages that are in the EPROM. More data arranged in longer lines is presented than with the computer mode Display buffer command. Also, this command does not require a Read to buffer first. One can use this command to display EPROM contents without affecting the current contents of the buffer. This command displays one screen and stops. Enter another CR to see the next page. The command is aborted by the space key.

Read EPROM to display

R[address]<CR>

This is like the See command above but displays .HEX dump instead of ASCII. Note that it also can be used without affecting the current contents of the buffer. If for example, an EPROM did not verify during a program operation, this command allows you to look at the bad data in EPROM to find out more info about the failure without changing the buffer contents. Paging and aborting are the same as the See command above.

Program EPROM byte

P[address]<CR>

The program, byte by byte, allows you to alter or patch an EPROM in a way not unlike debuggers do with RAM. When the command is started, the starting address and its' contents are displayed on the screen. The operator now has the choice of attempting a new burn to that location or to leave it alone. If a CR is depressed, the current location is left alone and the next location and its' contents are displayed. If the operator wishes to attempt a change the new values (two HEX Characters) are entered before the CR key. The PROGRAMMER 4+ will try to program the location as specified and will show an error if no verify occurs. The operation of this command is aborted by depressing the Space key.

Write to EPROM 16 bytes

W[address]<CR>

This command allows a little faster method of manually programming an EPROM. It displays the address every 16 bytes and does not bother to display the contents. The user does not have the option of skipping locations with this command as in the Program command above. The display formats accross the screen as the user enters the two HEX characters followed by a CR. When 16 bytes have been completed the display will automatically CR, LF and display another address. In this way a block format display is presented to the user. This command also verifies as it goes, so an error in programming will be reported as it happens. Abort this command with the space bar.

"DATA TERMINAL"
(COMPUTER, CRT, LPT, ETC)

**DB-265
FEMALE
25-PIN
CONNECT**

**"DATA SET"
(MODEM)**

DB-25S
FEMALE
25-PIN
CONNECTOR

**"DATA TERMINAL"
COMPUTER, CRT, LPT, ETC.)**

RECEIVE CIRCUIT
TERMINATING
IMPEDANCE = $3K - 7K$
MAX OPEN CIRCUIT
RIVER VOLTAGE = .25

**POWER ON,
PLUGGED IN,
SIGNALS VALID**

**READY TO
SEND A
CHARACTER**

COMMONLY USED	SEND CHARACTER
NOT	HI-LO "VOTE
COMMONLY	LIN

FOR DATA SET TO USE
FOR TRANSMIT SYNCH
FOR DATA TERMINAL
TO USE TO TRANSMIT

BAUD RATE - 1 BIT TIME

TYPICAL TD/RD SCOPE WAVEFORM
(ASCII SPACE-PARITY W/ 127g)

START DATA

LSB



MSB PARITY

274


ONE: < IV MARK (CLOSED)

MARK (CLOSED)

MODEM TIMING:

TC		- BIT TRANSITION
RC		- BIT CENTER

TERMINAL TIMING:

(TC)		- BIT CENTER
------	---	--------------

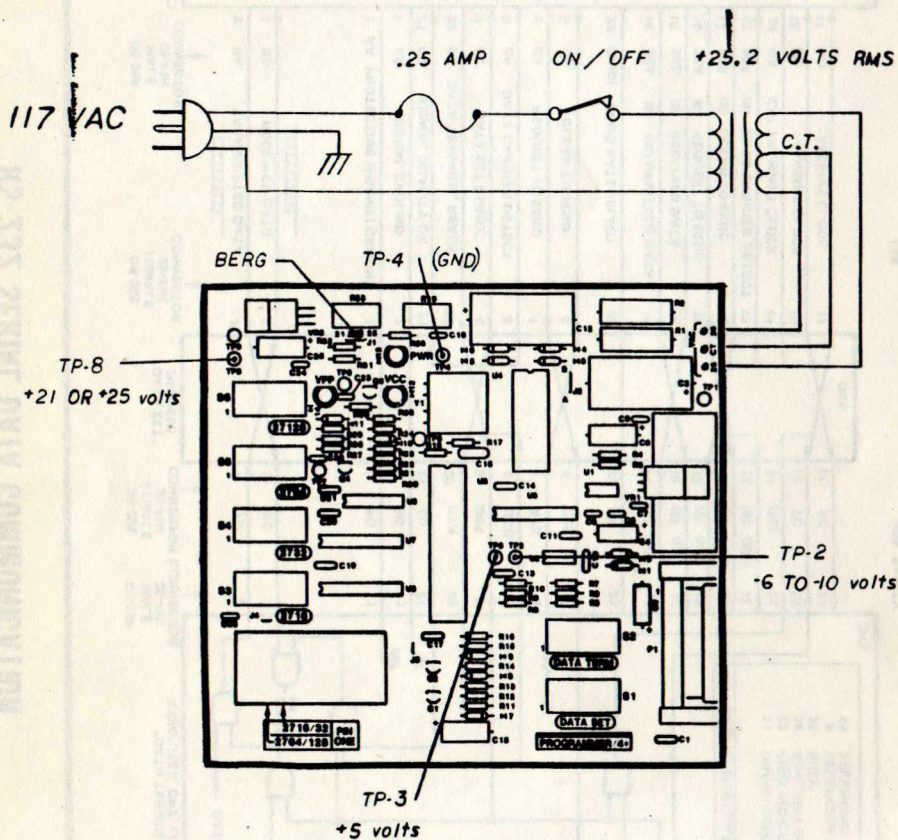
<u>ONE</u>	<u>ZERO</u>
MARK SIGNAL)	SPACE (SIGNAL)
OFF (CONTROL)	ON (CONTROL)
< +1V	< +3V

NOT USED BY MOST DATA TERMINALS

11	UNASSIGNED	R
22	RING INDICATOR (CE)	
18	RCV DIBIT CLK	
25	BUSY	
9	TELEPHONE CO TP	
10	TELEPHONE CO TP	

CT
CGK

POWER SUPPLY



INTEL-HEX FORMAT

Intel data records begin with a nine-character prefix and end with a two-character suffix. The byte count must equal the number of data bytes in the record.

Figure 2-9 simulates a series of valid data records. Each record begins with a colon (:), which is followed by

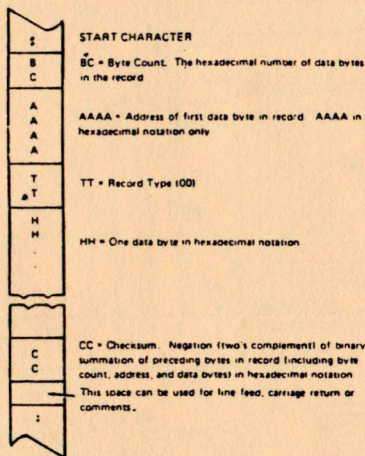
a two-character byte count. The four digits following the byte count give the address of the first data byte.

Each data byte is represented by two hex digits; the number of data bytes in each record must equal the byte count.

The suffix is a two-digit checksum, the two's complement of the binary summation of the previous bytes in the record.

INPUT

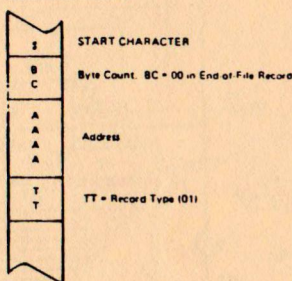
DATA RECORD



LEGEND

- | | |
|------|------------------------------------|
| | = Start Character |
| BC | = Byte Count (Data Bytes/Record) |
| AAAA | = Address Field |
| TT | = Record Type |
| H | = One Hexadecimal Digit (0-9, A-F) |
| CC | = Checksum of Record |

* END OF FILE RECORD



OUTPUT

NOTES

- 1) Number of bytes per record is variable. See Table 3.1
- 2) Each line ends with nonprinting line feed carriage return and nulls.

2 Hex characters = 1 byte

Data Records

BCAAAAATTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTCC
BCAAAAATTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTCC
BCAAAAATTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTCC
BCAAAAAT

* Alternate end of file record

:0000000000

:00000001FF

