MODEL 7956 USERS MANUAL

Document number 7956V230.MAN

DATE- MAY 1, 1986

***** READ THIS IF NOTHING ELSE *****

THE END OF THE PROGRAMMING SOCKET MARKED BOTTOM IS WHERE GROUND IS. THIS MEANS THAT PIN 12 ON A 24 PIN PART GOES AT THE BOTTOM. SO DOES PIN 14 ON A 28 PIN PART.

APPLY AC POWER BEFORE PUTTING DEVICES INTO THE PROGRAMMER.

DO NOT ATTEMPT TO READ A MASKED ROM WITHOUT CHECKING TO SEE IF VPP IS APPLIED DURING READS FOR THAT PART TYPE NUMBER.

SEE INFORMATION ABOUT BAUD RATES AND CABLES IF PROGRAMMER FAILS TO COMMUNICATE.

I

INTRODUCTION

Congratulations. You now have, what we believe to be, the most cost effective gang eprom programmer on the market today. The

design philosophy used on the 7956 allows for simple future expansion of capabilities. It may be used as a stand alone production programmer, or via its RS-232 interface from a host computer or terminal. All serial communications with the 7956 is in printable ASCII characters and it supports Intel and Motorola hex formats as well as simple block formats. Additionally, the 7956 supports the MCS-86 extended hex format, and Motorola's S record format with features for automatically split programming two eproms for use in a true 16 bit data path. Resident features include facilities for making source to eprom content comparisons, erasure checks, formatted device listings, menu driven device selection, and more.

The 7956's interrupt driven type ahead buffer allows it to program and verify in real time, while data is being sent (transparent to the user, whose sole responsibility is to send and receive data). Two user selectable algorithms are available, a standard 50ms program cycle with post verification and adaptive algorithms.

The user may elect to use the adaptive (intelligent) algorithm on 27128, 2764, 2732, 2732A device types. Adaptive algorithms are required and automatically used on the 2764A, 27128A, 27256, F27256, and 27512. MCM68764's and MCM68766's also use an adaptive algorithm. Adaptive algorithms typically offer a six fold improvement in programming time over the standard algorithm. Extended diagnostics pinpoint the cause of errors.

The 7956 is capable of programming and reading the following devices (at the time of this printing).

NMOS EPROMS:

2508,2516,2532,2564,2758A,2758B,2716,2732,2732A,2764,2764A, 27128, 27128A, i27256, F27256, 27512, 5133, 5143, MCM68766, MCM68764

CMOS EPROMS:

27C16, NMC6716, 27C32, F27C64, i27C64, 87C64, 87C256, i27C256, F27C256,i27C512,F27C512

MPU'S:

8741, 8742, 8748, 8748H, 8749, 8751, 8755

(482, 484, 512, 513, 756 ADAPTER)

EEPROM's:

Seeq 5213, Xicor X2816A, X2864A, Intel i2816A, i2817A

II

COMMANDS

Comments:

All voltages and pin configurations are set up by the onboard microprocessor and no personality modules are required. Don't try to read ROM equivalents until you have checked to see if you might damage the ROM.

Remember that the Master Socket may have programming voltage applied to pin 1. No other pin of the Master Socket gets programming voltage. This means that you may NOT copy from a 2764A to a 2764, unless you have taken action to prevent programming voltage from being applied to pin 1 of the 2764A. You can also get around the problem by copying the 2764A to a file and then making a 2764 master chip.

INITIALIZATION:

1) Remove all devices from the master and slave sockets of the programmer and apply AC power. If the dip switches are selecting anything from 02-99, the programmer will initialize to the selected device and is then ready to program. If a 00 is selected, the Leds will begin to Rotate in a binary sequence until you press either the Copy or Verify button. The part selected in this manner is then displayed on the Leds and sent via RS-232 as the prompter. A 01 will cause the Leds to Rotate also, and pressing Verify or Copy will stop them, but you must then use the Copy and Verify button to select the part you desire by the binary setting of the Leds, or use the dips to select the desired part, or communicate with the Menu command and the dip selection in position 00.

2) If the rotary dip switch is left in position 00, you can select a device by using the menu command via RS-232. This is similar to the method used in the 7128 and the 7228. See the Menu command section to do this. PGX and PGMX software is compatible with this unit.

3) When you have made your setting with the dip in the first position, move the selector back to zero to use the selection. You may now stand alone or communicate with the programmer with the eprom type you have selected.

STANDALONECOMMANDS:

1) VERIFY SLAVES FOR ERASURE - Pressing the Verify button causes the programmer to check the slave sockets for erasure. An illuminated error light indicates a non erased device.

2) COMPARE SLAVES AGAINST MASTER - Depressing both the verify and copy buttons simultaneously causes the programmer to compare the contents of the slave sockets against that of the master socket. An illuminated error light indicates a slave which does NOT compare. Note: When executing this command, both the ready and busy lights will be illuminated.

3) COPYING FROM MASTER TO SLAVES - Depressing the copy button causes all data from the master socket to be programmed into the slave devices. An illuminated error light indicates the slave did not program properly.

Note that the previous commands may be aborted by depressing either the verify or copy button while the command is in progress. A short beep is issued each time an error occurs. A longer beep is issued upon completion of the command.

RS-232SOFTWARECOMMANDS:

The following commands which program affect all slaves. Commands which upload data to the host via RS-232 get their data from slave socket 7, the one just to the right of the master socket.

P PROGRAM COMMAND.

Sending a “P”, followed optionally by an ASCII-HEX address, and a valid delimiter puts the 7956 into the program mode. Once in the program mode, ASCII-HEX data to be programmed is sent. The data may be a continuous stream or the bytes (groups of two hex characters) may be separated by valid delimiters. The program mode

is terminated upon the receipt of an ASCII dollar sign, "$" or if an error occurs.

Thus, the program command may be used to program one byte or a block of bytes at any given location. Valid delimiters are spaces, commas, carriage returns, line feeds, or dashes. It may be useful to note that the 7956 totally ignores null characters. All characters sent are echoed as they are removed from the input FIFO (type ahead buffer). NULL, XON, and XOFF characters are never put into the FIFO.

The 7956 tries to program all 8 sockets. A *WP error will be issued only if all eight slave sockets fail. YOU must clear the error lights with the "X" command before programming if you want to know if any errors occurred on a particular socket. A short beep will sound when an error occurs. Other errors are handled as they occur with messages. The following example illustrates how 33h and 23h are programmed to locations 444h and 445h in a 2716.

Example: 2716P444-33 23$

2716

[ready for next command]

: INTEL HEX PROGRAM COMMAND.

When in the command state, receipt of a colon is interpreted as the lead character in an Intel hex record. The 7956 automatically enters the program mode and programs the data contained in the hex at the address specified in the header of the hex record. The check sum is verified at the end of the hex record and the programmer then returns to the command state but does not reissue the command prompter unless the record happened to be the END record. This is done in anticipation of another hex record, i.e., all characters from the hex file, sent to the Model 7956 will be echoed back to the user with no additions or deletions.

The error light above any socket which fails to program properly will light. Remember to clear the error lights before sending your file with the "X" command. A short beep will be issued each time an error occurs, except when using adaptive algorithms. If a data error,

checksum error, or syntax error occurs during the file transfer, the programmer will issue the appropriate error message and abort back to the command state. A *WP error will be issued only if all eight slaves fail.

See the section on toggles and hex formats for clarification on how to program two devices for device use on a true 16 bit data bus. The segment base address register, maintained by the 7956, is automatically cleared when the end record is detected, or if any other command is executed other than the Intel Hex command.

S MOTOROLA HEX PROGRAM COMMAND.

This command functions precisely the same way that the Intel hex program command does, except the format is the Motorola S record format. Records may be of type S0, S1, S2, S3 OR S9.

/ TEKTRONIX HEXADECIMAL PROGRAM COMMAND.

When in the command state, receipt of a slash is interpreted as the lead character in a Tektronix hex block. The 7956 automatically enters the program mode and programs the data contained in the hex block at the address specified in the header of the hex block. The checksums are verified at the end of the hex block and the programmer then returns to the command state but does not re-issue the command prompter unless the block happened to be the termination block. This is done in anticipation of another hex block, i.e., all characters from the hex file, sent to the Model 7956 will be echoed back to the user with no additions or deletions.

R BLOCK READ COMMAND (FROM SOCKET E7 ONLY)

The R command, followed optionally by beginning and ending addresses, causes the Model 7956 to output a continuous string of ASCII-HEX characters between the specified addresses. If no addresses are specified, the 7956 will output the entire contents of the selected device. The R command may be aborted at any time by sending a dollar sign, “$”, to the programmer. The following example uses the eprom programmed in the example of the P command.

Example: 2716R444,445

3323
2716

Note: The R command is primarily for automated reading of eproms. If you execute the command line as shown in the above example, you will find that the data output overwrites the command line unless your terminal is in an auto line feed mode. (Eg. 3323_R444,445)

OI INTEL HEX FILE OUTPUT COMMAND. (FROM E7)

The OI command has the same command syntax as the R command. It differs in that the 7956 will output the device contents as an Intel hex file, including the end record, between the specified addresses or if no addresses are specified, the entire device. Again, the command may be aborted if desired with a dollar sign, "$".

OM MOTOROLA HEX FILE OUTPUT COMMAND.

The OM command functions precisely the same way the OI command does, except that the file output is in the Motorola S record format.

OT TEKTRONIX HEX FILE OUTPUT COMMAND.

The OT command works the same way as the OM and OI command does, except that the output is Tektronix Hexadecimal Format.

L LIST FORMATTED COMMAND.

The L command outputs the data, between optionally specified addresses, in a formatted fashion similar to many dump utilities. If no addresses are specified, the entire contents will be listed and the command may be aborted with the dollar sign, "$". Each line of the listing includes the beginning address in ASCII- HEX, sixteen data bytes in ASCII-HEX and the ASCIREPRESENTATION of the data.

Non printable bytes are replaced with periods in the ASCII representation field.

Example:

2716L90,AF

0090 4845 4C4C 4FFF//99FF HELLO.//..
00A0 FFFF FFFF FFFF//FFFF ....../..
2716_ [prompter indicates end of command]

Note: Unlike the R, OI, OT and OM commands, the L command will output a carriage return and line feed at the beginning of the listing. This is because the L command is primarily used when the host is functioning as a terminal and it would be irritating to have the first line of the listing overwrite the command line.

V VERIFY ERASURE COMMAND.

The V command checks the cells between the optionally specified addresses for erasure, FF's or 00's as the device type dictates. If no addresses are specified, the entire device is checked. If a non erased cell is encountered, the led above that particular socket will light and a short beep will be sounded. Remember to reset the leds before issuing the "V" command (with an "X" command). Any messages refer to the Slave socket next to the Master socket. The process continues until the end address is reached or the command is aborted with a dollar sign, "$". The programmer is left in the "compare" mode. The following example uses the same eprom used in the P and R command examples.

Example: 2716V

2716

M MENU SELECTION. (DIP POSITION 00)

You may select the device you will be working with in 2 ways. The current device type always becomes part of the command prompter. Selecting a device establishes the programming algorithm to be used, as well as the device pinout, proper programming voltage and prompter.

See note about programming voltages and 28 pin parts under comments at the beginning of the command section.

.pa

Instructions:

A) Direct Selection Method (types 02-99):

1) Move selector switch to position desired from the decimal column in the table on the following page. See Appendix D for single dip select.

B) Software Selection Method:

1) Move selector to position 00 and leave it there.

2) Press “M” on keyboard.

3) Then press the code letter for the device you want or to get a displayed menu.

.PA

Dip Switch Menu Selection Table:

(I) = intelligent mode selected by “TI”.

i = always intelligent

METHODA

NMOS PN:	DIP	NMOS PN:	DIP
2516	22	i2764A	27
2532	08	5133	15
2564	14	5143	18
2758A	19	27128	18
2758B	20	27128 -(I)-	06
2716	02	i27128A	28
2732	03	i27256	07
2732 -(I)-	16	iF27256 —*—	21
2732A	04	i27512	33
2732A -(I)-	17	i68764	09
2764	15	i68766	09
2764 -(I)-	05		

* NOTE CHANGE OF “G” SELECTION FROM
2508 TO F27256, NMOS FUJITSU TYPE.

CMOS PN:	DIP	MPU’s PN:	DIP
NMC6716	23	8741	10
27C16	24	8742	13
27C32	25	8748	10
F27C64	26	8748H	11
27C16H	30	8749	12
27C32H	31	8749H	13
iF27C256	37	(WITH 482)	

.PA

MPU’s PN:	DIP	OTHER PN:	DIP
8751	32	8755	29
(WITH 512)		(WITH 756)	

EEPROMS PN: DIP

5213 —— 34
X2816A —— 35
I2816A —— 36
I2817A —— 38
X2864A —— 39

See the Appendix section on Manufacturer's cross reference to correlate your part number with the appropriate eprom type selection. See Appendix D to select parts with the copy and verify buttons on single or double rotary dip switch programmers.

.PA

Software Selection Method:

2732M

EPROMSELECTIONMENU

NMOS	NMOS	CMOS	EEPROM	W/ADAPT
A -2758A	H -2516	L -27C16	P -5213	R -874x-1K
4 -2758B	I -2532	M -27C32	Q -X2816A	S -874x-2K
B -2716	J -2564	N -MC6716	Y -I2816A	T -874xH-1K
C -2732	K -i68766	5 -27C16H	3 -2817A	U -874xH-2K
D -2732A	G -F27256	6 -27C32H	9 -X2864A	V -8751
E -2764		O -F27C64		W -8755
1 -i2764A	Z -i27256	8 -F27C256		
F -27128	7 -i27512			
2 -i27128A				

ENTERSELECTION2
i27128A_

Results in the programmer giving you a menu of parts to select from. Refer to the appendix parts list for help in selecting the correct part. At that time, enter the menu selection number and the prompter will reflect the part number selection that you made, or dial in the right selection.

.pa

Note: The dip has precedence over the software select. If you set a device by software, you can reset it with the dip. Positions 02-99 have precedence over the hardware select position 01. If you select one of positions 02-99, the programmer will output the new selection to the terminal. The RS232 menu command is only functional when the selector is in position 00.

T TOGGLE COMMAND.

The toggle command is used as a prefix to a subset of commands. These commands are as follows:

TC - The TC toggle command is used to turn the compare mode on and off. When in the compare mode, the command prompter is prefixed by a lower case c. The compare mode is used to compare the contents of a device against that of a source file. To use the compare mode, use the TC toggle to turn on the compare mode. Then use one of the various programming commands as if you were going to program the device. Instead of programming the device, the 7956 will make a comparison of the source byte to the contents of the device. If they are not the same, the comparison error will cause the led above that particular device will be lit and the programmer will continue to check the other eproms. See Diagnostics Section for details.

TS - The TS toggle puts the 7956 into a split mode used for programming two eproms whose intended destination is for use in a true 16 bit data path environment. While in the split mode, the

command prompter is prefixed by either a lower case h or l indicating high (Odd Address) byte or low (Even Address) byte respectively. See TB command below. Note that the split mode works on either Intel Hex type files or Motorola S record type files. It is not functional from the “P” command.

TB - The TB command is used in conjunction with the split mode, TS, to target the selected device for the high (ODD) bytes or low (EVEN) bytes from an Intel Hex or Motorola S record source file.

TI - The TI command toggles the programming algorithm between intelligent and non-intelligent(dumb) on device types 2732, 2732A,2764,27128.

TN - The TN command is used to generate a 16 bit checksum from the data in the eprom. This is the sum of all the DATA bytes added together without carry. You may make a checksum between any two addresses by specifying them like the OI and V and other commands that use a start and end address. The checksum is calculated for all 9 sockets and then output to the user. See examples of this in the PGX and PGMX chapters.

.PA

TR - The TR command resets all toggles above.

' ' SPACE COMMAND

Sending a space (ascii 32 char) to the programmer causes it to reissue the command prompter.

I IDENTIFY DEVICE TYPE COMMAND.

The Model 7956 will re-output the command prompter in response to an I. This may be used by automated programs which need to have the prompter transmitted to them. Sending the 7956 a carriage return or a space yields the same results.

X CLEAR ERROR LIGHTS AND RETURN VERSION.

The X command is used to clear the error lights above the chips before programming or verifying a chip via software. The X command will return the following:

```
2716X
GTEK, INC.
MODEL 7956 Vx.xx
COPYRIGHT 1983
2716_
```

And the leds will be turned off. When ordering accessories from GTEK, please remember to include the version and serial number.

$ ABORT COMMAND

A $ sent to the programmer will abort most operations.

.pa

III

DIAGNOSTICS

General

1) All error codes to be issued by the 7956 are preceded by an asterisk, "*". This makes error trapping very easy.

2) When a non-fatal error, such as won't program, need erasing or compare occurs during programming, (WP, NE, CP) the error light for the associated chip lights.

3) FATAL errors are output on a real time basis, that is, they are output as soon as they are detected, and the programmer returns to the command state.

4) Fatal Error codes include the address at which the error occurred.

5)The error lights are cleared only when you issue the “X” command. This is so that you can issue more than one command, such as the Verify and Program, so that the error lights will be cumulative.

Fatal Error Codes

*WP ERR @ nnnn WON’T PROGRAM error

This error is issued only in the event that all eight slave sockets have errors.

.pa

*CP ERR @ nnnn COMPARISON error

Issued during comparisons and verifies, but only if ALL eight slaves fail.

*DT ERR @ nnnn DATA error

Not valid hex data.

*CS ERR @ nnnn CHECK SUM error

If checksum error is detected in hex record. Only applies to Intel and Motorola hex format program commands.

*SN ERR @ nnnn SYNTAX error

An invalid command was issued to the programmer. See COMMANDS section.

*ST ERR @ nnnn STACK error

FIFO overflow. Reduce baud rate or see the interfacing section for handshaking methods. (The 7956 can take data at 300 bps with no handshake.)

*UV ERR @ nnnn Un-aVailable error

Issued in the event the user tries to use a function of the programmer that is not available for that particular device.

.pa

Non-Fatal Errors

Non-fatal errors are indicated only by the LEDs, no message is output to the console. These errors are considered non fatal in that the process continues, i.e. you don't want to stop programming eight eproms just because one has failed.

During programming, the error lights indicate that the chip failed to program. The eprom may have needed erasing, may be no good, the wrong device type was selected or the device was mis-socketed.

During erasure verification, error lights mean that the chip is not erased.

During comparisons, error lights mean that the eprom contents differ from the source.

Overload Conditions

If a programming voltage overload condition occurs, the programmer will issue a continuous tone, indicating an overload on the programming voltage pin. The tone can be aborted by pressing the copy or verify button if in the stand alone condition or by issuing a "$" or other command from the keyboard if in the communication mode.

Usually, the programmer will have aborted to the command state and have issued a *WP ERR @ nnnn and lit all the error lights, since the programming voltage will have been cut short during the last programming cycle.

Remove the shorted part(s) or make the proper selection for the chips you are programming or correct the condition that caused the overload and CAUTIOUSLY try them again. Shorted Vcc pins will not cause the same kind of overload. Chips will usually fail on the first byte with this kind of overload, without the continuous tone.

To find a part causing the problem out of 8 parts, take out 4 chips and try again. If it still errors out, take 2 chips out and try again. If it still errors out that means that one of the two parts is bad. Take 1 out and try again. If it still errors out, that must be the bad chip; put the remainder of the chips back into the programmer and begin again. The Master chip could cause the problem if it is a 28 pin part.

Remember that the Master Socket may have programming voltage applied to pin 1. No other pin of the Master Socket gets programming voltage. This means that you may NOT copy from a 2764A to a 2764, unless you have taken action to prevent programming voltage from being applied to pin 1 of the 2764A. You can also get around the problem by copying the 2764A to a file and then making a 2764 master chip.

.pa

IV

INTERFACING NOTES

The Model 7956 is surprisingly easy to interface to and there are several methods of handshaking which can be utilized if it is desired to operate at the higher baud rates. The following section describes some of the methods.

1.Software handshake. This is perhaps the easiest method of all. When you begin to send data to be programmed, send the first byte but don't wait for it to be echoed. That would effectively cut your communication rate in half. Instead, send the second byte, receive the first, send the third byte, receive the second, etc. This technique will allow you to

program as fast as the algorithm in use permits. Some devices program faster, some slower! See an example of this in Fig. 4.1

2. CTS/DTR hardware handshaking. The Model 7956 is configured as data terminal equipment, which means that the CTS (clear to send) line is an input to the programmmer which when pulled low forces the programmer to stop sending. On the other hand, the DTR (data terminal ready) line is an output from the programmer, which will go low when the buffer contains 4 or more characters and high again when there are less than 2 characters in the FIFO. If you are using hardware handshake and the DTR line goes low, you should stop sending to the 7956. The RTS line is pulled high whenever the programmer is plugged in. See Specifications for Cable.

3. Xon/Xoff software handshaking. If you do not monitor the DTR line, the 7956 will transmit an XOFF character if there gets to be 9 characters in the FIFO. When the FIFO level drops below 6 characters, an XON will be transmitted. Likewise, when the programmer is sending you data, you may send an XOFF character, which will stop the programmer from sending until it receives an XON character. XON's and XOFF's, are not put into the FIFO, but are processed as soon as they are received. Even if you don't use XON/XOFF handshaking, you will find it useful when using the L, list command, to stop and start the data flow to your screen. XON and XOFF are the keyboard equivalents of control-Q and control-S respectively.

4.Please note that the 7956 may communicate at 4 different baud rates. To initialize at the new baud rate, simply start sending at least 4, and up to 6 spaces to the programmer at the new baud rate to to communicate at the new baud rate. The programmer will begin reissuing the prompter in response to the spaces when locked on again.

.pa

Glen:

figure 4.1 which is the flowchart goes here!

.pa

V

AUTOMATION HINTS

When you automate the transfer of data from your computer to the 7228, you should examine the echoed characters to see if an asterisk, "*" has been sent. If you receive one, it means that an error message will follow and that the programmer will return to the command state. Any automation software should take this into account.

The effective addressing range of a device is determined by it's size. If a 2K byte device is being used, then it only has 11 significant address lines and only the lowest 11 bits of the address field are significant. Thus, as far as the 7228 is concerned, 000H is equivalent to 800H or F000H in a 2K device.

You don't need to compare the characters that are echoed to what you sent. The characters are echoed to the host as they are removed from the FIFO, and would not reflect a programming error. However, the 7228 will detect any programming error and the host need only trap the error message. The PGX utilities for CP/M and MSDOS based computers send echoed characters to the screen (console). PGMX, due to its high baud rates, does not attempt to display all the information being transferred.

The programmer is in the command state after the prompter is sent. The prompter always ends with a ''. You can use this character to let your program know that an R, OI, OM, OT, V, or L command has finished.

You should probably have one mode of operation where you communicate directly with the programmer (turn your computer into a terminal). This will give you easy use of the L, V, P, and M commands.

.PA

SPECIFICATIONS

DIMENSIONS: (H x W x D)

2.5" x 12.0" x 7.5"

(63.5mm x 304.8mm x 190.5mm)

POWER:

120VAC, 60HZ, 25 VA (240Vac, 50Hz, option)

INTERFACE:

DB25P - data terminal equipment (see below).

DATAWORD:
1 Start, 8 Data, 1 Stop, No parity

BAUDRATE:
Auto select 300, 600, 1200, 2400

WEIGHT:
5 Pounds (2.4 KG)

OPERATINGENVIRONMENT:
45 - 95 DEG F. (7 - 35 DEG C.)
5% TO 95% non-condensing relative humidity

CABLE:
DTE programmer dte computer dce
1- Equip Ground (EG)—1 (EG) - 1 (EG)
2- Transmit Data (TXD)——3 (RXD) - 2 (TXD)
3- Receive Data (RXD)- 2 (TXD) - 3 (RXD)
4- Ready To Send (RTS)——6 (DSR) - 4 (RTS)
5- Clear To Send (CTS)-20 (DTR) - 5 (CTS)
6- Data Set Rdy (DSR)——4 (RTS) - 6 (DSR)
7- Signal Ground (SG)—7 (SG) - 7 (SG)
20- Data Term Rdy (DTR)——5 (CTS) - 20 (DTR)
.pa
MAKING A CABLE.

Refer to the Specifications section for information on making a cable for other than an IBM PC.

IBM PC DB25 FEMALE PROGRAMMER DB25 FEMALE

EG 1————————————1 EG
TXD 2————————————3 RXD
RXD 3————————————2 TXD
CTS5————————————20DTR
SG 7————————————7 SG
DTR 20———————————5 CTS
DSR 6————————————4 RTS
RTS 4————————————6 DSR

HOOK UP FOR IBM PC-AT 9 PIN DB TO 25 PIN DB
AT DB9 MALE: PROGRAMMER DB-25 FEMALE
CD 1—-nc
RXD 2————————2 TXD
TXD 3————————3 RXD
DTR 4————————5 CTS
SG 5————————7 SG
DSR 6————————4 RTS
RTS 7————————6 DSR
CTS 8————————20 DTR
RD 9—-nc
.PA

VII

HEX FORMATS

INTEL FORMAT

DATA RECORD

Byte
1 Colon (:)
2..3 Number of binary data bytes

4..5 Load address, high byte
6..7 Load address, low byte
8..9 Record type
10..x Data bytes, 2 ASCII-HEX characters
x+ 1..x+ 2 Checksum, two ASCII-HEX characters
x+ 3..x+ 4 CR,LF

END RECORD

Byte
1 Colon (:)
2..3 Record length, must be “00"
4..7 Execution address
8..9 Record type
10..11 Check sum
12..13 CR,LF

EXTENDEDADDRESSRECORD(MCS-86HEXFORMAT)
Byte
1 Colon (:)
2..3 Record length, should be “02"
4..7 Load address field, should be “0000"
8..9 Record type, must be “02"
10..13 USBA
14..15 Check sum
16..17 CR,LF
.PA

START ADDRESS RECORD (MCS-86 FORMAT)
Byte
1 Colon (:)
2..3 Record length, “04"

4..7 “0000"
8..9 Record type, “03"
10..13 8086 CS value
14..17 8086 IP value
18..19 Check sum
20..21 CR,LF

The checksum is the two’s compliment of the 8- bit sum, without carry, of all the data bytes, the two bytes in the load address, and the byte count.

MOTOROLA FORMAT

COMMENT RECORD

Byte
1..2 “S0"
3..n comment field
x+ 1..x+ 2 CR,LF

DATA RECORDS

Byte
1..2 “S1"
3..4 Number of data bytes + 3.
5..6 Load address, high byte.
7..8 Load address, low byte.
9..x Data bytes, 2 characters each.
x+ 1..x+ 2 Checksum.
x+ 3..x+ 4 CR,LF.
.pa
Byte

1..2 “S2"
3..4 Number of data bytes + 4.
5..10 Load address, 24 bits
11..x Data bytes, 2 characters each.
x+ 1..x+ 2 Checksum.
x+ 3..x+ 4 CR,LF.

Byte
1..2 “S3"
3..4 Number of data bytes + 5.
5..12 Load address, 32 bits
13..x Data bytes, 2 characters each.
x+ 1..x+ 2 Checksum
x+ 3..x+ 4 CR,LF.

END RECORD

Byte
1..2 “S9"
3..4 CR,LF.

In the above S records, the byte count includes the load address and checksum. Thus the byte count is equal to the number of data bytes plus the following; 3 for S1, 4 for S2 and 5 for S3 type records. The checksum is the one’s compliment of the 8-bit sum, without carry, of the byte count, the two bytes of the load address, and the data bytes.

.pa

TEKTRONIX HEX FORMAT

DATA BLOCKS

Byte
1 Header which is a Slash (/)

2..5 Location counter which is 4 ascii hex bytes representing the load address of the data bytes.

6..7 Byte count which is 2 ascii hex bytes specifying # of binary data bytes in the data field of the block.

8..9 First Checksum which is 2 ascii hex bytes specifying the HEX SUM of the values of the previous six digits. (location counter and the byte count)

10..X Binary data bytes which are each represented as 2 ascii hex digits. (in other words 16 binary bytes are represented as 32 ascii bytes.)

X+ 1..X+ 2 Second Checksum. 2 ascii hex bytes representing the SUM modulo 256 of the binary values of the ascii data bytes. (8 bit sum without carry.)

X+ n Always a carriage return.

TERMINATION BLOCK

Byte

1 Header (/) slash

2..5 Transfer address which is the address for execution of code, probably 0000 in most cases.

6..7 Byte count, always 00 for a termination block.

8..9 Checksum of the six digits that make up the transfer and byte count which is probably 00 in most cases.

10 Always a carriage return.

ABORT BLOCK

Byte

1 Header (/) slash

2 Header (/) slash

3..X+ 69 Message up to 69 characters for error information etc.

X+ 70 Always a carriage return.

Example of Data block and 1 Abort block

/0000101001020304050607080 90A0B0C0D0E0F0038
//THIS IS AN ERROR MESSAGE HERE

Note: programmer will issue a *DT error on the second “/” mark and return to the command state without displaying the abort message...

Example... of Data block and 1 Termination block

/0000101001020304050607080 90A0B0C0D0E0F0038
/00000000 (CARRIAGE RETURN)

NOTE: Most terminals will display Tektronix data only on one line, since the format calls for only a carriage return at the end of a record.

.pa

VIII

USING INTERFACE PROGRAM PGX

INSTALLATION OF PGX

PGX (for MS/PC DOS or CP/M) is sold as an option to the programmer. It is a communication program which enables you to do 3 basic things:

1) Communicate with the programmer. We call this the interactive mode.

2) Program Eproms from an Intel Hex file.

3) Read Eprom Data into an Intel Hex file.

The GHEX program allows you to convert a binary or executable file into an Intel Hex file to send to the programmer.

On the DOS PGX program disk you will have several files:

1- PGX12.EXE Use with COM1: and 7128
2- PGX12ALT.EXE Use with COM2: and 7128
3- PGX24.EXE COM1: WITH 7956, 7956, 9000
4- PGX24ALT.EXE COM2: WITH 7956, 7956, 9000
5- GHEX.EXE Convert binary to HEX
6- PGX.OBJ Main object file
7- UART1200.OBJ RS-232 driver object file
8- READPGX.ME Instructions for use
9- RUN7128.BAT Batch file to install .OBJ's
10- DEBUG.TXT Instruction Tutorial
11- PARTS.LST Parts Cross-Reference

On the CP/M program disk you will have different programs depending on what type of computer you are using. A "generic" CP/M program disk will have at least 3 programs:

1- PGX.COM Communication(Rdr/Pun I/O)
2- GHEX.COM Convert Binary to Hex
3- READPGX.ME Installation Instructions
4- SID.TXT Tutorial with SID.COM
5- DDT.TXT Tutorial with DDT.COM
6- PARTS.LST Parts Cross-Reference
7- RDRPUN.HEX Reader/Punch driver (inst.)
8- IOB.ASM Source code for I/O Byte dvr.
9- ZILOG.ASM Source code for Uart driver.
10- PGXBAT.COM Assembled Batch I/Ob driver.
11- PGXCRT.COM Assembled CRT I/Obyte driver.
12- PGXTTY.COM Assembled TTY I/Obyte driver.
13- PGXUC1.COM Assembled UC1 I/Obyte driver.

14- RDRPUN.SUB Submit file to install RDR/PUN

15- IOBYTE.SUB Submit file to install IOB.HEX

16- ZILOG.SUB Submit to install ZILOG.HEX

The first thing you should do is back up your master disk. If you have a hard disk, we suggest that you make a sub-directory and copy the files from your master disk to the newly created sub-directory. (See your DOS manual on how to make a sub-directory.)

Under CP/M, PIP the master disk to your hard disk or create a new disk and PIP the programs to it.

Example for DOS:

C:\MD GTEK

C:\CD GTEK

C:\GTEKCOPY A:*.*

Example for CP/M:

APIP B:=A:*.*[R

The end result will be having a copy of the programs from the master disk on a working area of the hard disk or on a working disk. Store the original copy of our programs in a safe place. Make a note of the serial number from the label to have it handy when you call GTEK for upgrade information or to ask questions about the operation of the program. A good place would be in the NOTES section in the back of this manual.

Now, on your working copy, you may delete the programs that you don’t need to use the programmer.

1- Select the program that you will be using with the programmer, PGX24.EXE for com1: and a 7956 for instance. Under CP/M, select the program that you need to use or install PGX.COM for the driver that you need to use. PGX.COM under CP/M comes installed with Reader/Punch drivers. See the READPGX.ME file for installation instructions under CP/M.

2- Rename the program so that the instructions match the name of the program on your disk:

DOS: CREN PGX24.EXE PGX.EXE

CP/M: AREN PGX.COM = PGXUC1.COM

.pa

3- Copy the programs to the disk or sub- directory that you will be using it from.

DOS:C:\ GTEKCOPY PGX.EXE \ WORKAREA

or A:COPY PGX.EXE B:

CP/M: APIP B:= A:PGX.COM

When you are using DOS on an IBM or Compatible, you should be aware that PGX is an interrupt driven program. CP/M PGX is a “polled” version.

IRQ4 is used in conjunction with an interrupt service routine for COM1: when PGX is invoked. This is a hardware line on your PC to give the system an interrupt whenever a character is received. If you know that something else in your computer is using this hardware interrupt line, then you should use the other com line, which uses IRQ3 (COM2:).

IRQ3 is also used in the same manner for COM2: when PGX is invoked. If you know something in your system uses IRQ3 for interrupts, then you must use the other com port.

OPERATION

PGX is a “command” driven program as opposed to a “MENU” driven program, which means that everything you do is done by entering a “command” from the DOS or CP/M command line instead of selecting the command from a menu. This makes the program very fast when you have learned what the commands are.

.pa

In most cases the commands are exactly the same command as what the programmer is expecting so the selection of the command is somewhat intuitive.

When you invoke the PGX program from the following examples, the log-on messages may not be identical to the messages displayed here, but they will be very similar.

There are 3 modes from the DOS or CP/M command line that you may enter:

1- Communication or Interactive mode:

```
CPGX

GTEK Programmer Interface Package Version 5.04
Copyright 1983, GTEK, INC.
X
GTEK,INC.
MODEL 7956 V2.30
COPYRIGHT 1983
xxxx_
```

The above example when invoked puts you into a mode where you may simply communicate with the programmer to issue commands

such as M, L, V, P, etc. Control returns to DOS or CP/M when you issue a Control-C from the command prompter or during any programmer operation being performed from this mode.

.pa

2- Program mode:

```
CPGX filename [options]
GTEK Programmer Interface Package Version 5.04
Copyright 1983, GTEK, INC.
X
GTEK,INC.
MODEL 7956 V2.30
COPYRIGHT 1983
xxxxoptions
EPtype:10000000etc....
:0000000000
type
C_
```

The above example shows how an eprom may be programmed from the DOS or CP/M command line. See the following text about OPTIONS to understand what they are. The above operation may be aborted to DOS or CP/M at any time by a control-C. You may invoke this operation from the DOS or CP/M command line ONLY, not from within PGX (interactive mode).

.pa

3- Read mode:

```
CPGX filename [options]
GTEK Programmer Interface Package Version 5.04
Copyright 1983, GTEK, INC.
X
GTEK,INC.
```

MODEL 7956 V2.30
COPYRIGHT 1983
xxxx options
type OI

:10000000 etc....
:0000000000
type
C_

The above example shows how an Intel Hex disk file may be created from an eprom in the programmer. See the following text about OPTIONS to understand what they are.

DEFINITIONS

Please note that the listed commands are generally passed on to the programmer unchanged except for the order in which they appear in the command line. PGX will send the commands specified to the programmer in the following order:

1 - menu command
2 - toggle commands (except TN is done last)
(TN not available under CP/M)
3 - verify erasure
4 - program or read
5 - checksum (tn)
.pa
Some options, particularly the “R” option, work differently from the programmer “R” command.

The “@” command is not a valid command for the programmer except on the DOS or CP/M command line. The “@” command is used to give PGX information, not the programmer.

You may not specify any command more than once inside the brackets except for 2 toggle commands under DOS and 1 toggle command under CP/M.

sssss= starting address, up to 24 bits worth of ascii-hex characters (16 bits for CP/M).

eeeee = ending address, up to 24 bits worth of ascii-hex characters (16 bits for CP/M).

oooo = offset amount, 16 bits worth of ascii- hex characters

Delimiter = a dash (-), a comma (,), a space ( ), a carriage return (ascii character number 0dh), or a line feed (ascii character number 0ah). Carriage return and line feed are represented by a
or .

FILENAME = a valid DOS filename to be used by PGX to look for a file on the disk. PGX will automatically supply a .HEX extension and look for a .HEX even if you specified an extension.

AND REMEMBER!

The effective addressing range of a device is determined by it’s size. If a 2K byte device is being used, then it only has 11 significant address lines and only the lowest 11 bits of the address field are significant. Thus, as far as the 9000 is concerned, 000H is equivalent to 800H or F000H in a 2K device.

VALID OPTIONS FOR PGX

1- Any valid programmer command except OI, OM, OT, R

2-@sssss-eeeee

An @ symbol followed by the starting address (sssss) followed by a dash (-) followed by the ending address (eeeee) will cause the incoming data from the Intel Hex File to be sent to the specified locations. The @ symbol means that PGX will search the Intel Hex file for data located between the start address (sssss) and the end address (eeeee) and send that data to the same locations within the eprom. Remember that if the address range for that eprom is exceeded by the specified range of addresses then the data will “wrap-around” to the beginning of the eprom or to an address modulo eprom size.

The OPTIONS for PGX are always issued on the DOS or CP/M command line and are as follows:

R - read file. (default is program mode)

m bounds

M - menu selection

T - toggle command

V - verify erasure

These OPTIONS must be within two square brackets separated from the filename by a space. The general usage is as follows:

CPGX filename [OPTIONS]

INTERACTIVE EXAMPLES:

CPGX

From the DOS command line, this establishes communication with the programmer and after the log-on message, displays the Programmer Command Prompter, which is the currently selected eprom type or xxxx, which means that you must first select an eprom type. The programmer at this point is waiting to accept a valid command.

2732TN

F000 F000 F000 F000 F000 F000 F000 F000 F000
2732_

In this example, the checksum command was issued from the programmer command state, causing it to do a 16 bit checksum on the selected part (empty sockets or blank parts, which have a checksum of F000).

2732VF00-FFF

2732_

In this example, the programmer was commanded to check the part to see if it was blank within the range of F00 up to and including address FFF. Error lights indicate unerased.

PROGRAMMING EXAMPLES:

The options are always set off by an opening square bracket ([) and the ending square bracket (]). Invalid commands result in an error message and a return to DOS or CP/M. It should be noted that any ascii-hex characters used in boundaries should be upper case.

CPGXfilename[TI,ME,V,@00000-00FFF]

(or)

CPGX filename [@0-FFF, ME, TI, V]

(same results)

In the above examples, PGX will first log-on with the programmer and send the following (in bold face):

(log-on message)
GTEK,INC.
MODEL 7956 V2.30
COPYRIGHT 1983
xxxxME
2764TI
i2764V0000-0FFF
(sends nothing if blank, otherwise it will send *CP ERR @ address /lf) if all 8 fail.
i2764:10000000xx... Intel Hex data here
if at least 1 eprom was blank, up to end record...
:0000000000
i2764
/lf
C_

PGX first sent the eprom Menu command to select the eprom type, then sent the Verify command @ the specified addresses 000 through and including FFF to the programmer. If the eproms were not blank, the result would be for the programmer to output the message *CP ERR @ nnnn to the console if all fail, and you will return to the DOS or CP/M command prompter. If at least one eprom was blank, (the rest have error lights lit above them) as each option is finished, PGX sends the next option in order of priority. If a mistake is made within the OPTIONS bracket PGX will issue an OPTION ERROR message and return to DOS or CP/M. This command may be aborted by typing a control - C on the keyboard.

READING EXAMPLE:

CPGX filename [R,MF]

In the above example, PGX will first select the eprom type (MF) and then issue an OI (Output Intel) to the programmer. The programmer will respond by selecting the eprom type (27128) and then outputting 16 byte long INTEL.HEX records read from socket E7 until the end of the eprom is reached (0000-3FFF). PGX will then write the INTEL.HEX records to the disk. If there is not enough room in memory to hold the entire file, a handshake will cause the programmer to stop sending and write the data to the disk. This command may not be aborted by a control - C after it is started.

The CP/M version of PGX will not allow you to read an eprom bigger than about a 27128. If you do it will tell you to read shorter segments. With a 27256 for instance, read it from 0-3FFF and then from 4000-7FFF.

MORE EXAMPLES:

To program 3 2716's from a binary file that contains 1093H bytes. User input is underlined and PGX input is boldface:

CGHEXTEST.BIN

CPGXTEST[@0-7FF,MB,V]

(Log on message)
xxxxMB
/If
2716V0000,07FF
/If
2716:10 etc...

:0000000000
/lf
2716
/lf
Cpgx test [@800-FFF,V]

(Log on message)
2716V0800,0FFF

2716:10etc...
:0000000000
/lf
2716
/lf
CpgxTEST [@1000-1FFF]
(Log on message)
2716:10etc...
:0000000000
/lf
2716
/lf
C_

In the above example, we first created an INTEL.HEX file from the binary file TEST.BIN. We then told PGX to send the Menu command for a 2716 eprom (MB) to the programmer, Verify that the part was blank between the load addresses 0000H through 07FFH. PGX then sent all the data that was found in the file TEST.HEX between the locations 0000H through 07FFH to the programmer.

After the first section was completed, we then told PGX to Verify that the part is blank between the addresses 0800H through 0FFFH within the eprom (remember that the address 0800H is not significant to this eprom- 0000H is really the same address as 0800H in this eprom). PGX then sent all the data that was found in the file TEST.HEX between the

locations 0800H through 0FFFH to the programmer. Remember that the programmer keeps the type selection in it's prompter, so you don't have to keep re-selecting the eprom type every time, unless you are using different eproms every time. If you do keep selecting the eprom type, it is a cheap form of insurance.

After the second section was completed, we then told PGX to send all the data that was found in the file TEST.HEX between the locations 1000H through 1FFFH. You might remember that there are only 93H bytes left to send. When PGX finishes sending what's left in the file (1000H to 1093H), it encounters an END record in the file, which causes PGX to return to DOS (or CP/M).

Automating the process could be accomplished with a batch file such as this:

```
TEST.BAT
PGXTEST[@0-7FF,V,MB]
pause remove eprom, insert new blank
PGXTEST[@800-FFF,V]
pause remove eprom, insert new blank
PGXTEST[@1000-1FFF]
echo now you are done.
```

In particularly large files, PGX may take some time to search through the whole file, so if you are programming a 2732 from a file that is 32K long and specify that you want to send 7000H through 7FFFH (@7000-7FFF), it could take a while for PGX to scan from the beginning through to 7000H.

See the Chapters on GHEX and DEBUG also.

.PA

USAGE OF GHEX.EXE

GHEX.EXE is a program provided for you to be able to convert a binary file into an INTEL.HEX file. This capability is built-in to the PGMX.COM program, but you may want to use it for convenience.

General usage is:

GHEXfilename.ext

.uj 0
OR
GHEX filename.ext offset

Offset is an ASCII-HEX number that specifies where you want your code to begin in the HEX file.

GHEXfiletest.bin

Will result in an INTEL.HEX file being created on your disk by the name filetest.hex. The load addresses begin at 0000H since no offset was specified. GHEX does not destroy the input file.

GHEXfiletest.binAA55

Will result in an INTEL.HEX file being created on your disk by the name filetest.hex, just like before except the load addresses start at AA55H.

.PA

X

USING DEBUG.COM

You may use DEBUG.COM (supplied with PC-DOS) in conjunction with our GHEX.EXE to modify an INTEL.HEX file without worrying about the checksums in the INTEL.HEX file.

The following is a short tutorial to modify a 4K byte INTEL.HEX file with DEBUG. The procedure is to run DEBUG first.

DEBUG

-_

From the - prompter within DEBUG use the N command to specify the name of your INTEL.HEX file.

-Nfilename.HEX

-_

Use the L command to load the hex file with an offset (if it begins at 0000H). You must do this since if it starts loading at 0000H within the segment, it will overwrite your file control block at 5Ch.

-L 100

-_

The CX register now contains the number of bytes read into memory with an offset of 100. You may have to modify the CX register to properly reflect the correct number bytes you must write back to the disk. Remember that this is going to write from CS:CX when you issue the command.

-RCX

CX:1000

-_

Your data is now loaded into the memory of the computer at offset 100H. Use the E command to modify the bytes you need to modify. An example of modifying locations starting at 0A55H with data is shown. Locations A55H through A57H contain FFH.

-EA55 01 02 03

-_

Now specify a new file name to write to the disk with since you can't use an extension of HEX with the file you are writing. You want to call it a BIN or IMG file instead since that is what the data really is anyway.

-NNEWFILE.BIN

-_

Now you can use the Write command to write the new data to the disk. DEBUG will write an exact image of CS:CX bytes to the disk starting at an offset of 0100H bytes.

-W

Writing 1000H bytes

-_

Now use GHEX to make it an INTEL.HEX file, or use PGMX's binary file transfer.

.PA

XI

USING INTERFACE PROGRAM PGMX

INSTALLATION of PGMX

PGMX, sold separately, is a high speed communication program which runs on IBM PC's and AT's. It allows flexible manipulation, transmission and reception of Intel HEX files and binary files.

On the PGMX program disk you will have 3 programs: PGMX.COM, PINSTALL.COM and GHEX.EXE. PGMX is the program used to communicate with your programmer. PINSTALL is the program that you must run to install the serial drivers in PGMX so that you can communicate with the programmer.

If you try to run the PGMX program without installing the serial drivers, it will tell you to run the PINSTALL program. Remember that the PGMX license is a single user license. If you buy the programmer now and later on wish to purchase the software, be sure to have the CORRECT serial number ready.

Insert GTEK program disk in drive A: and copy the programs to your hard disk with:

CCOPY A:*.*

This will copy all the programs on the GTEK disk over to the subdirectory that you are logged on to on your hard disk. If you don't have a hard disk, use DISKCOPY or COPY to the B: drive. Refer to the DOS manual for specific instructions on using the COPY command. The desired end result is a backing up of the original GTEK copy. Store the original program disk in a safe place.

Now you should insert the backup copy in the drive A: and/or go to the subdirectory where PINSTALL and PGMX are located. You must first run the PINSTALL program to install the serial drivers for PGMX.

CPINSTALL

After the copyright and version number appears, you are asked to select a letter which corresponds to the type of installation you wish to perform.

Most people will probably select to set up to communicate at 2400 baud on computer serial port COM1: or the selection for 2400 baud on COM2:

IRQ4 is used in conjunction with an interrupt service routine for COM1: when PGMX is invoked if you installed it for COM1:. This is a hardware line on your PC to give the system an interrupt whenever a character is received. If you know that something else in your computer is using this hardware interrupt line, then you should use the other com line, which uses IRQ3 (COM2:).

IRQ3 is also used in the same manner for COM2: when PGMX is invoked if you installed it for COM2:. If you know something in your system uses IRQ3 for interrupts, then you must use the other com port.

The next selection that you have to make is where your line printer is located, on parallel port 1, 2, or 3 (lpt1:, lpt2: or lpt3:). This has to be done so that PGMX knows where to send printed data.

After you have made that last selection, you are returned to the DOS command prompter and PGMX is set up to run under those conditions that you specified.

See the example for CPGMX
later in the manual.

OPERATION

PGMX is a “command driven program” as opposed to a “MENU driven program” which means that everything you do is done by entering a “command” on the command line instead of “selecting” the command from a menu. This makes the program very fast when you have learned what the commands are.

In most cases the commands are exactly the same command as what the programmer is expecting, so the selection of the command is somewhat intuitive.

There are 2 ways that commands may be given to PGMX:

1- From the PC or MS DOS command line.

2- From within PGMX.

Commands executed from DOS return to DOS upon completion. Commands executed from within PGMX return to PGMX upon completion. Command lines may be entered from within PGMX by depressing control F.

EXAMPLES:

CPGMX

Enter PGMX and establish communication with the programmer (assuming everything is hooked up properly).

CPGMX FILENAME

Results in communication being established with the programmer and sending FILENAME.HEX (Intel Hex Format) from the disk to the programmer. When PGMX is through, you are returned to the DOS system prompt.

CPGMX FILENAME [OPTIONS]

Results in PGMX establishing communication with the programmer, and then performing according to selected options.

Programming the eprom in binary or Intel Hex format or Reading the eprom in the same formats may be accomplished by giving the proper options. OPTIONS are always enclosed in square brackets and separated by comma's. Invalid commands result in an appropriate and descriptive ERROR message.

VALID OPTIONS

R - read file. (default is program mode)

%ooooo - binary mode select (default is HEX)

m bounds

Mx - menu selection

Tx - toggle command (3 max on command line)

V - verify erasure

MORE EXAMPLES

PGMX

from the DOS command line establishes communication with the programmer, and after log-on displays the Programmer Command Prompter, which is the currently selected eprom type.

LOG ON MESSAGE EXAMPLE

High Speed Interface Package Version 2.01/
Copyright 1983, 1984, 1986 GTEK, INC.
All Rights Reserved, worldwide.
I/O Hardware Driver Vers 1.01 - IBM PC/AT
Serial port - COM1, 2400 bps
Printer port - LPT1:

GTEK,INC.
MODEL 7956 V2.30
COPYRIGHT 1983,1986
xxxx_

The programmer is ready and waiting for a command at this point. If you want to do a Menu command, pressing an M and the code necessary will select an eprom type or press M
to get a menu:
.pa
2732M

EPROMSELECTIONMENU

NMOS	NMOS	CMOS	EEPROM	W/ADAPT
A -2758A	H -2516	L -27C16	P -5213	R -874x-1K
4 -2758B	I -2532	M -27C32	Q -X2816A	S -874x-2K
B -2716	J -2564	N -MC6716	Y -I2816A	T -874xH-1K
C -2732	K -i68766	5 -27C16H	3 -2817A	U -874xH-2K
D -2732A	G -F27256	6 -27C32H	9 -X2864A	V -8751
E -2764		O -F27C64	W -8755	
1 -i2764A	Z -i27256	8 -F27C256		

F -27128 7 -i27512
2 -i27128A

ENTERSELECTION2
i27128A_

Results in the programmer giving you a menu of parts to select from. Refer to the appendix parts list for help in selecting the correct part. At that time, enter the menu selection number and the prompter will reflect the part number selection that you made, or dial in the right selection.
.pa
i27128ATN

C000 C000C000 C000 C000 C000C000C000 C000
i27128A_

Results in the programmer giving you a 16 bit addition of all the 8 bit bytes of all the parts in all 9 sockets, without carry. Blank 27128s give you C000 for the checksum.

i27128A(control)

Control- generally means to press and hold the CONTROL key on your keyboard and press a command letter. Valid command letters are P, F and C. The ESCape key is also a valid control command key, but you do not hold the control key down to press ESC. The ESC key is a valid control character already. The escape control command may also be obtained by pressing CONTROL [on the IBM keyboard or by holding down the ALT key and entering027 on the numeric keypad. Pressing and holding the CONTROL key is represented by a caret and the letter that must also be pressed, eg. ^C.

The definitions of the CONTROL commands are:

^P -start sending / stop sending (toggle) data simultaneously to the printer.

^ F-enter a command line. Examples follow.

^C -Abort most programmer commands and return to the DOS command prompter or PGMX. This command will work even though you may be in the process of programming, reading, verifying, etc., an eprom.

ESC or ^ [- Escape from program. This command is used as an alternative to control- alt-del and is not normally used. This is an EMERGENCY command and the results could be unpredictable.

USING control F

2716^F

Enter Command line FILENAME [@0-1FF,V,TN

Results in PGMX doing a blank check on the eprom between 0 and 1FF inclusive. Then FILENAME.HEX is opened and any hex data falling between the specified boundaries is sent. During data transfer, PGMX displays the load addresses of the hex records that it is sending. Finally, the checksum is calculated between the specified addresses and displayed.

The options are always set off by an opening square bracket ([) and the ending square bracket (]) is optional. Invalid commands result in an error message and a return to the programmer command prompter. You could do the same example above with PGX, except you must issue the command line from the DOS prompter.

DEFINITIONS

Please note that the listed commands are generally passed on to the programmer unchanged except for the order in which they appear in

the command line. PGMX will send the commands specified to the programmer in the following order:

.pa

1 - menu command

2 - toggle commands (except TN is done last)

3 - blank check or verify erasure

4 - program or read

5 - checksum (tn)

Some commands, particularly the “R” command, work differently from the programmer command “R”. The “%” and the “@” command are not valid commands for the programmer except on the command line. They are used to give PGMX information, not the programmer. You may not specify any command more than once inside the brackets except the toggle commands, and you are allowed a maximum of 3 of those.

sssss = 24 bit starting address, Hex chars.

eeeee = 24 bit ending address, Hex chars.

ooooo = 24 bit offset amount, Hex Chars

Delimiter = a dash (-), a comma (,), a space ( ), a carriage return (ascii character number 0dh), or a line feed (ascii character number 0ah). Carriage return and line feed are represented by a
or .

FILENAME = a valid DOS filename to be used by PGMX to look for a file on the disk. In case that a percent (%) sign was specified then the filename specified will be taken literally. In other words you must be explicit and give the extension of the filename also. If the percent sign was not specified then PGMX will automatically supply a .HEX extension and look for a .HEX even if you specified an extension.

.pa

EXT = a valid DOS extension for the filename in your directory. You are allowed to use any extension you wish here, (in the binary % mode)

and the data will be sent to the programmer UNCHANGED. The EXT will only be valid when you have specified a percent sign (%) within the brackets.

AND REMEMBER!

The effective addressing range of a device is determined by it's size. If a 2K byte device is being used, then it only has 11 significant address lines and only the lowest 11 bits of the address field are significant. Thus, as far as the 7956 is concerned, 000H is equivalent to 800H or F000H in a 2K device.

VALID COMMANDS FOR PGMX

1- Any valid programmer command except OI, OM, OT, R. This applies to PGX also.

2-@sssss-eeeee

An @ symbol followed by the starting address (ssss) followed by a dash (-) followed by the ending address (eeee) will cause the incoming data from the Binary or Intel Hex File to be sent to the specified locations. In the case of a binary file (specified by a % on the same command line only), the @ symbol means that the data specified by the % sign will go to the ssss-eeee specified by the @ sign within the eprom. In the case of an Intel Hex file (no %), the @ symbol means that PGMX will search the Intel Hex file for data located between the start address (ssss) and the end address (eeee) and send that data to the same locations within the eprom. PGX supports this, except for the binary transfer usage.

3- %ooooo

A percent sign (%) followed by an offset (you may omit specifying an offset of 0, but PGMX may warn you that you did not specify it, just in

case you forgot) will cause PGMX to treat the EXTension you specified literally (and not add a .HEX extension). PGX does not support this.

.uj 0

EXAMPLES:

To program 3 2716's from a binary file that contains 1093H bytes:

xxxxMB

2716^F

EnterCommandline—TEST.BIN[%0,@0-7FF

Causes PGMX to look for a file called TEST.BIN on the disk, and when found start sending from relative offset location 0 within TEST.BIN to locations 0 through 7FFh within the eprom. The number of bytes sent is the number of bytes between 0 to 7FFh inclusive. If you don't specify boundaries, you will "Wrap Around" to location 000H at location 800H because you are still sending data to the programmer through PGMX.

.pa

2716^F

Entercommandline—TEST.BIN[%800,@0-7FF

Causes PGMX to look for a file called TEST.BIN on the disk, and when found start sending from relative offset 800H from within TEST.BIN to locations 0 to 7FFh within the eprom.

2716^F

Entercommandline-TEST.BIN[%1000,@0-7FF

.uj 0

Causes PGMX to look for a file called TEST.BIN on the disk and when found start sending from relative offset 1000H from within the TEST.BIN to locations 0 through 7FFh within the eprom. However, the program

will terminate when it encounters the end of the file you are sending from, since there are only 94H bytes left in the file TEST.BIN left to send.

Automating the process could be accomplished with a batch file such as this:

TEST.BAT

pgmxtest.bin[mb,v,@0-7ff,%%0,tn

pause remove eprom, insert new blank

pgmxtest.bin[v,@0-7ff,%%800,tn

pause remove eprom, insert new blank

pgmxtest.bin[v,@0-7ff,%%1000,tn

echo now you are done.

(use 2 percents (%%) in a batch file)

Reading an eprom to a disk file is accomplished with the 'R' option.

pgmxfilename[r

Results in reading the selected eprom to the Intel hex disk file, FILENAME.HEX.

pgmx filename[r,%

.uj 0

Results in reading the selected eprom to a binary disk file whose name is FILENAME. (no extension was specified.). Notice an offset value included with the % has no meaning

during a read operation.

pgmx [tn,ma or ^ F[tn,ma from within PGMX

Results in selecting 2758 (note menu selection has side effect of resetting all toggles) and calculating the checksum.

ADVANCED EXAMPLE

pgmx filename[mz,ts,v,tn,@20000-2FFFF

Results in selecting 27256, split mode, doing a blank check, programming the eprom with hex data residing between the 20 bit addresses of 20000 and 2FFFF inclusive, and calculating it's checksum.

.uj 0

This particular file is big. Don't be afraid that PGMX has hung up. It has to check the load addresses of every record in the file, and it would take a minute before it reached records at load address 20000, unless the file was created with an "exotic" compiler in such a manner that segment records with apparently random addresses are placed at apparently random locations every few records in the file. No joke intended.

.pa

The boundaries specified cover a 64k range, but the eprom is only 32k. The reason for this is that in the split mode, the 2 eproms are considered as one eprom of twice the size. However, if an error message is issued during programming in the split mode, the address given by the error message is the physical address in the single eprom.

.PA

WARRANTY AND SERVICE

LIMITED WARRANTY

GTEK, INC., warrants to the original purchaser of this GTEK, INC., product that it is to be in good working order for a period of 90 days from the date of purchase from GTEK, INC., or an authorized GTEK,

INC., dealer. Should this product, in GTEK, INC.'s opinion, malfunction during the warranty period, GTEK will, at its option, repair or replace it at no charge, provided that the product has not been subjected to misuse, abuse, or non-GTEK authorized alterations, modifications, and / or repairs.

Products requiring Limited Warranty service during the warranty period should be delivered to GTEK with proof of purchase. If the delivery is by mail, you agree to insure the product or assume the risk of loss or damage in transit. You also agree to pre-pay the shipping charges to GTEK. GTEK agrees to pay return freight (in the continental USA) by UPS surface on warranty work.

ALL EXPRESS AND IMPLIED WARRANTIES FOR THIS PRODUCT INCLUDING, BUT NOT LIMITEDTO, THE WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE LIMITED IN DURATION TO THE ABOVE 90 DAY PERIOD. Some states do not allow limitations on how long an implied warranty lasts, so the above limitations may not apply to you.

UNDER NO CIRCUMSTANCES WILL GTEK, INC. BE LIABLE IN ANY WAY TO THE USER FOR DAMAGES, INCLUDING ANY LOST PROFITS, LOST SAVINGS, OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF, OR INABILITY TO USE, SUCH PRODUCT. Some states do not allow the exclusion or limitation of incidental or consequential damages for consumer products, so the above limitations or exclusion may not apply to you.

THIS WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS, AND YOU MAY ALSO HAVE OTHER RIGHTS WHICH MAY VARY FROM STATE TO STATE.

The limited warranty applies to hardware products only.

.PA

SERVICE

For warranty service or non warranty service, contact GTEK, INC. at (601) 467-8048 to obtain an RMA (Return of Material Authorization number). We will need the serial number and date of purchase. Send the programmer, freight prepaid to:

GTEK,INC.

P. O. Box 289

307 COLEMAN Avenue

WAVELAND MS. 39576

Be sure to include the RMA on and in the package so we will know what to do with it. GTEK will pay return freight (within the Continental United States) on in-warranty service. Out of warranty service charges are determined on an hourly labor plus materials basis.

.pa

PGX AND PGMX SOFTWARE LICENSE AGREEMENT

"This software is a proprietary product of GTEK, Inc. It is protected by copyright and trade secret laws. It is licensed (not sold) for use on a single micro-computer system, and is licensed only on the condition that you agree to this LICENSE AGREEMENT." GTEK, INC. provides this program and licenses its use worldwide. You assume responsibility for the use of this software to achieve your intended results, and for the installation, use and results obtained from the software.

LICENSE

The Licensee may:

a. use the program on a single machine;

b. copy the program into any machine readable or printed form for backup or modification purposes in support of your use of the program on the single machine;

c. modify the program and/or merge it into another program for your use on the single machine (Any portion of this program merged into another program will continue to be subject to the terms and conditions of this

Agreement.): and,

d. transfer the program and license to another party if the other party agrees to accept the terms and conditions of this Agreement. If you transfer the program, you must at the same time either transfer all copies whether in printed or machine- readable form to the same party or destroy any copies not transferred; this includes all modifications and portions of the program contained or merged into other programs.

You must reproduce and include the copyright notice on any copy, modification or portion merged into another program.

YOU MAY NOT USE, COPY, MODIFY, OR TRANSFER THE PROGRAM, OR ANY COPY, MODIFICATION OR MERGED PORTION, IN WHOLE OR IN PART, EXCEPT AS EXPRESSLY PROVIDED FOR IN THIS LICENSE. IF YOU TRANSFER POSSESSION OF ANY COPY, MODIFICATION OR MERGED PORTION OF THE PROGRAM TO ANOTHER PARTY, YOUR LICENSE IS AUTOMATICALLY TERMINATED.

TERM

The license is effective until terminated. You may terminate it at any other time by destroying the program together with all copies, modifications and merged portions in any form. It will also terminate upon conditions set forth elsewhere in this Agreement or if you fail to comply with any term or condition of this Agreement. You agree upon such termination to destroy the program together with all copies, modifications and merged portions in any form.

.pa

PGX AND PGMX LIMITED WARRANTY

THIS PRODUCT IS NOT A CONSUMER PRODUCT WITHIN THE MEANING OF THE UNIFORM COMMERCIAL CODE AND APPLICABLE STATE LAW. THE PROGRAM IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU (NOT GTEK, INC.) ASSUME THE ENTIRE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION. SOME STATES DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES, SO THE ABOVE EXCLUSION MAY NOT APPLY TO YOU. THIS WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS AND YOU MAY ALSO HAVE OTHER RIGHTS WHICH VARY FROM STATE TOSTATE.

GTEK, Inc. does not warrant that the functions contained in the program will meet your requirements or that the operation of the program will be uninterrupted or error free. However, GTEK, Inc. warrants the diskette(s) on which the program is furnished, to be free from defects in materials and workmanship under normal use for a period of ninety (90) days from date of delivery to you as evidenced by a copy of your receipt.

Licensee herein acknowledges that the software licensed hereunder is of the class which inherently cannot be tested against all contingencies by Licensor. Licensee acknowledges Licensee's obligation to test all programs produced by the licensed software to determine suitability and correctness prior to use.

LIMITATIONSOFREMEDIES

GTEK, Inc.'s entire liability and your exclusive remedy shall be:

1. the replacement of any diskette(s) not meeting GTEK's "Limited Warranty" and which is returned to GTEK, Inc. with a copy of your receipt, or

2. if GTEK, Inc. or the dealer is unable to deliver a replacement diskette(s) which is free of defects in materials or workmanship, you may terminate this Agreement by returning the program and your money will be refunded.

IN NO EVENT WILL GTEK, INC. BE LIABLE TO YOU FOR ANY DAMAGES, INCLUDING ANY LOST PROFITS, LOST SAVINGS OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE SUCH PROGRAM EVEN IF GTEK, INC. HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, OR FOR ANY CLAIM BY ANY OTHER PARTY.

SOME STATES DO NOT ALLOW THE LIMITATION OR EXCLUSION OF LIABILITY FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES SO THE ABOVE LIMITATION OR EXCLUSION MAY NOT APPLY TO YOU.

.pa

GENERAL

You may not substitute, assign or transfer the license or the program except as expressly provided in this Agreement. Any attempt otherwise to sublicense, assign or transfer any of the rights, duties or obligations hereunder is void.

This Agreement will be governed by the laws of the State of Mississippi.

Should you have any questions concerning this Agreement, you may contact GTEK, Inc. by writing to GTEK, Inc. Sales and Service, P.O.Box 289, Waveland, MS, 39576.