Model 7228 user's manual
Document number &7228.PUB

Date- 18 December 1987

Fourth Edition

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

The end of the programming socket marked Ground is where ground is. This means that pin 12 of a 24 pin part and pin 14 of a 28 pin part goes at this end.

Apply AC power before putting devices into the 7228.

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.

—Notes—

Table of Contents

—Notes—

INTRODUCTION

Congratulations. You now have what we believe to be the most cost effective and advanced eprom programmer on the market today. The design philosophy used on the 7228 allows for simple future expansion of capabilities. All communications with the 7228 is in printable ASCII characters and it supports Intel, Motorola and Tektronix hex formats as well as simple block formats. Additionally, the 7228 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, a Zap command for chip erasing EEPROMs, and more.

The 7228's interrupt driven type ahead buffer allows it to program and verify in real time, while data is being sent. The model 7228 programs and verifies in real time transparent to the user, whose sole responsibility is to send and receive data. The standard algorithm prereads cells prior to programming, skips the cell if it is not necessary to program it, and post verifies the cells to assure that it is properly programmed. Extended diagnostics pinpoint the cause of errors.

In addition to this algorithm, the user may elect to use an adaptive algorithm on the 27128, 2764, 2732, 2732A. Adaptive algorithms are required and automatically used on the 27256 and MC68766/64's for instance. Adaptive algorithms typically offer a six fold improvement in programming time over the standard algorithm.

The Model 7228 V8.26 may be used with hardware CTS/DTR handshake. Baud rate selection is done automatically through your interface program, PGMX. The 7228 defaults to 2400 baud on power-up. Used in conjunction with any terminal or computer with an RS-232 port, the 7228 is capable of programming and reading the devices listed in Appendix C of this manual. 40 pin devices require socket adapters.

All voltages and pin configurations are set up by the onboard mpu and no personality modules are required. ROMs may be read safely only with certain eprom selections, such as i27512, i68766, F27C64 and 27C32.

COMMANDS

P Block Program

Sending a 'P', followed optionally by an ascii–hex address, and a valid delimiter puts the 7228 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 ascii–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 7228 totally ignores null characters. All characters sent are echoed as they are removed from the input FIFO (type ahead buffer). Xon, and Xoff characters are never put into the FIFO. The following example illustrates how 33h and 23h are programmed to locations 444h and 445h in a 2716:

Example: 2716> **P444—33 23$**
2716>_

[Ready for next command. This command is handled by PGMX's binary mode.]

: Intel hex program

When in the command state, receipt of a colon is interpreted as the lead character in an Intel hex record. The 7228 automatically enters the program mode and programs the data contained in the hex record 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 7228 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 7228 will be echoed back to the user with no additions or deletions.

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

7228, is automatically cleared if a programming error occurs, the end record is detected, or if any other command is executed other than the Intel Hex command.

S Motorola hex program

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 hex program

When in the command state, receipt of a slash is interpreted as the lead character in a Tektronix hex block. The 7228 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. All characters from the hex file, sent to the Model 7228 will be echoed back to the user with no additions or deletions.

R Block Read (Ascii–Hex Output)

The R command, followed optionally by beginning and ending addresses, causes the Model 7228 to output a continuous string of ASCII-HEX characters between the specified addresses. If no addresses are specified, the 7228 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 uses the eprom programmed in the example of the P command.

Example: 2716> **R444,445<cr>**
3323$
2716>_
[terminated by cr,lf, followed by prompt]
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 over writes the command line unless your terminal is in an auto line feed mode.

Example: 3323> **R444,445<cr>**
2716>_

The "R" command is handled by PGMX with the binary (%) read mode. Don't confuse it with PGMX's "R" command, which is to read Intel Hex.

OI Intel Hex Output

The OI command has the same command syntax as the R command. It differs in that the 7228 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 Output

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

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 Output

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

```
Example:  2716> L90,AF< cr>
0090  4845 4C4C 4FFF  // FFAB 99FF H E L L O . // . . . .
00A0  FFFF FFFF FFDD // FFFF  FFFF .  .  .  .  . // . . . .
2716>_
[prompter indicates end of command]
```

Note: The lines are shortened at the "//" to allow printing on this page. Unlike the R, OI, 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

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, its contents and address are output. The process continues until the end address is reached or the command is aborted with a dollar sign, '$'. The following example uses the same eprom used in the P and R command examples.

Example:
```
2716> V<cr>
33 @ 0444
23 @ 0445
2716>_
```

Note the command verifies the whole selected part because boundaries were not specified. The command outputs a cr/lf before each unerased byte is displayed so the command line is not overwritten. A new prompter indicates end of command.

M Menu

The Menu command is used to select the device type you intend to work with. The current device type always becomes part of the command prompter. Sending an M<cr> causes a menu to be output, from which the desired device is then selected. If the code letter for the device is already known, then just send M<code> and the device will be selected. Selecting a device establishes the programming algorithm to be used, as well as the device pinout, proper programming voltage and prompter.

Menu Command Examples follow:

xxxx> **MD**
2732A>_
Note that a cr,lf and new prompt are output subsequent to sending the code letter D which selects the 2732A device type.

2732A> **M<cr>**

```
EPROMSELECTIONMENU
NMOS        NMOS        CMOS        EEPROM      W/ADAPT
A– 2758     H– 2516     L– 27C16    P– 5213     R– 874x-1K
B– 2716     I–  2532    M– 27C32    Q– X2816A   S– 874x-2K
4– 2716B    +– 2532A                X– 48016    T– 874xH-1K
C– 2732     J– 2564     O– F27C64   Y– I2816A   U– 874xH-2K
D– 2732A    K– i68766   0– I27C64   9– X2864A   V– 8751
N– 2732B                            3– I2817A   W– 8755
E– 2764     Z– i27256   8– F27C256              !– 874xAH
1– i2764A   G– F27256
F– 27128    7– i27512
2– i27128A
Enter Selection –>E
2764>TI
i2764>_
```

You must give the 7228 a device type on power up before attempting to perform any commands other than Menu. An *SL ERR will be generated if this is not done.

xxxx>_ [default command prompter for V8.26]
See the selection chart in the appendix to select parts that are not on the 7228's menu but can be programmed using this programmer.

T Toggle command set.

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

TC Compare

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 7228 will make a comparison of the source byte to the contents of the device. If they are not the same, a comparison error will be issued and the 7228 will return to the command state. See Diagnostics Section for details.
Example: i2764A> **TC**
ci2764A>_

TI Intelligent

The TI toggle command is used to select between adaptive and standard programming. When using adaptive programming the prompt will indicate this with a small "i" before the prompt. For example, if you were programming a 2732 and you wanted to use the adaptive algorithm, press "TI" and the prompt would indicate "i2732>". This works on the following chips: 2732, 2732A, 2764, and 27128. It is used automatically on the 2764A, 27128A, 27256 and other parts. Press TI again and the small "i" disappears.

T Split

The TS toggle puts the 7228 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. It should be noted, that if a programming error occurs while in the split mode, that the address of the error given by the 7228 will be the address within the eprom being programmed, not the address in the hex file. See TB command.

TB Byte

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.

TR Reset

The TR command resets any previous toggle command except the adaptive command for the 2764A, 68766 (68764), 27128A, and 27256, which is permanent on those chips.

TN Checksum

The TN command is used to generate a 16 bit checksum from the data in the eprom. This is the sum of all the (8 bit) DATA bytes added together without carry. You may make a checksum between any two addresses by specifying them like the OI and V and the other commands that use a start and end address.

TZ Zap

The TZ command is the chip erase command used in conjunction with EEPROMS. The device is erased and erasure is verified. The UV error is issued in the event that the user attempts to execute the TZ command when the device is not capable of supporting it. The SEEQ 5213, Intel 2816A and Xicor X2816A (and others) are capable of modifying bytes randomly, and Zapping is not necessary. The Hitachi 48016, however, does not have this feature. Thus, in order to reprogram a 48016, the chip erase (Zap) command must first be executed.

' ' Space

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

I Identify device

The Model 7228 will reissue the command prompter in the same way that the space command does.

X Return Version Number

The X command will return the version number of the firmware installed in the chip shown as follows:

```
2716> X
GTEK,INC.
Model 7228 Vx.xx
Copyright  1987
2716>_
```

$ Abort Character.

Causes the operation in progress to abort to the command prompter when received dur ing operations of the OI, OM, OT, R, P, L, V, U. Refer to those commands for further information.

—Notes—

DIAGNOSTICS

General

1. All error codes to be issued by the Model 7228 are preceded by an asterisk, '*'. This makes error trapping very easy.
2. After an error occurs and the error mes sage is output, the input FIFO is cleared and the programmer returns to the command state.
3. Errors are output on a real time basis, i.e., they are output as soon as they are detected.
4. Error codes include the address, (nnnn), where the error occurred, except attempt to program on power up before selecting the eprom type (*SL err).

Error Codes

*WP ERR @nnnn— cell Won't Program. Eprom is **no good** or the **wrong device type** was selected or the device was **improperly inserted**, in which case, it **may or may not be any good now** !!

*NE ERR @nnnn— cell Needs Erasing. Can't program the cell because there are some bits that can't be moved from a 0 to a 1 state.

*CP ERR @nnnn— ComPare error. Issued in the event of a difference between source code and destination contents of target device. See TC command.

*DT ERR @nnnn— DaTa error. Not valid hex data. Character is not a 0 through 9 or A through F or CR or LF.

*CS ERR @nnnn— CheckSum error. Data check sum does not add up to what was sent in the HEX record. Only applies to Intel and Motorola hex format program commands.

*SN ERR @nnnn— SyNtax error. Not a valid programmer command. See commands.

*SL ERR @nnnn— SeLect error. No such menu code, or if no "@nnnn" present, you tried to program after power up without selecting an eprom type.

*ST ERR @nnnn— STack error. FIFO overflow. Reduce baud rate or see interfacing section for handshaking methods. The 7228 version 7.xx can take data at 300 bps with no handshake on any algorithm. The 7228 version 8.xx requires hardware handshaking.

*UV ERR @nnnn— Un-aVailable error. Issued in the event the user tries to use a function of the 7228 that is not available for that particular device, such as the Zap command on the I2816.

INTERFACING NOTES

The Model 7228 is surprisingly easy to interface 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. You can use only the second method with the 7228 version 8.xx.

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 figure 4.1 for flowchart.

2–CTS/DTR hardware handshaking. The Model 7228 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. Version 7.xx DTR will go low when the buffer is about 50% full and high again when the buffer is about 30% full. Version 8.xx has about the same amount of buffering, but DTR is constantly toggling to obtain the higher baud rates. If you are using hardware hand shake and the DTR line goes low, you should stop sending Immediately to the 7228. 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 7228 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 re ceived. 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. Version 8.xx does not send Xon/Xoff, but will accept it.

4–Please note that the 7228 may communicate at many different baud rates. To initialize to a new baud rate, send a "break" signal to the programmer for more than 100 milliseconds, and then at least 5 milliseconds after you restore from the break, send an 80H character at the baud rate you wish to begin sending. After that, a space command will cause the prompter to be reissued.

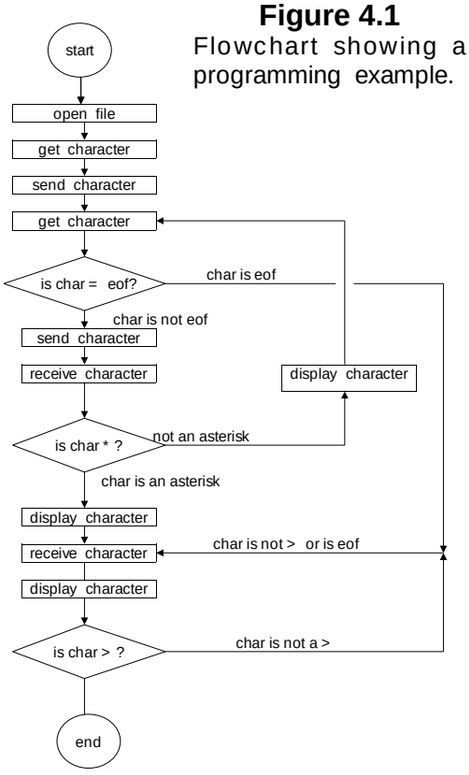


Figure 4.1
Flowchart showing a programming example.

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, unless the "D" option has been specified.

The 7228 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 7228 (turn your computer into a terminal). This will give you easy use of the L, V, P, and M commands.

—Notes—

SPECIFICATIONS

Dimensions (H x W x D):
2.75" x 5.25" x 6.75"
(70mm x 133mm x 171mm)

Power Requirements:
120VAC, 60HZ, 10VA
(240VAC, 50HZ optional)

Interface Connector:
DB25P configured as Data Terminal Equipment.

Data word size:
1 Start bit, 8 data, 1 stop bit, no parity

Auto Select Baud Rate:
1200–19200 (version 8.xx)
300–2400 (version 7.xx)

Weight:
3 pounds (1.4 KG)

Operating Environment:
45 - 95 deg F. (7 - 35 deg C.)
5% to 95% non-condensing relative humidity.

PROGRAMMER INTERFACE

The model 7228 has a DB25P connector configured as Data Terminal Equipment (DTE).
7228:

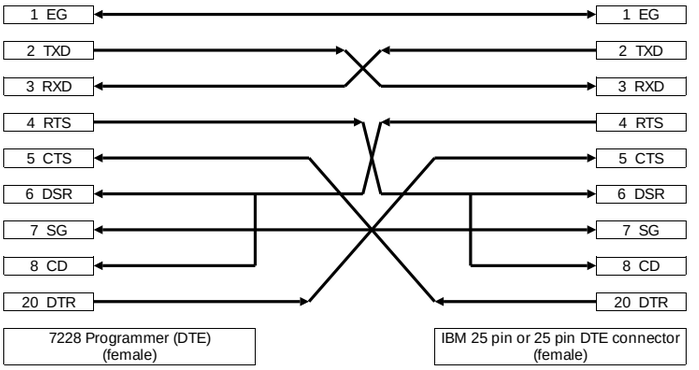
Pin#	Direction	Function
1—(EG)	<—>	Equipment Ground.
2—(TXD)	—>	Transmit Data.
3—(RXD)	<—	ReceiveData.
4—(RTS)	—>	Request To Send. Always active when power is on.
5—(CTS)	<—	Clear To Send. High enables 7228 to transmit data Pulled high internally.
6—(DSR)	<—	Data Set Ready. Not used.
7—(SG)	<—>	Signal Ground.
20–(DTR)	—>	Data Terminal Ready. High when programmer willing to accept data.

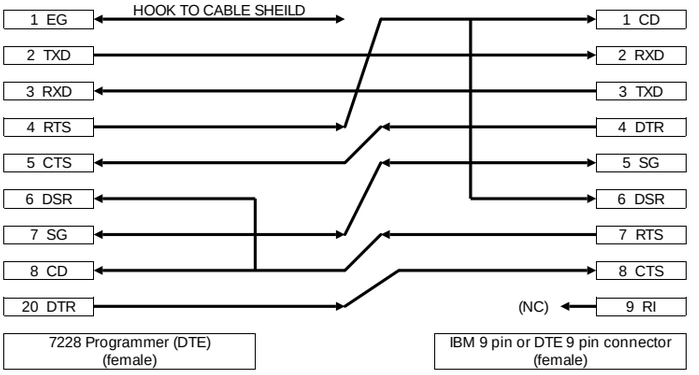
MAKING A CABLE.

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

IBM PC/XT/AT DB25 (female) to 7228 (female– Gtek pn RSMDTE)

IBM — 7228
EG –1 — 1—EG
TXD–2 — 3—RXD
RXD–3 — 2—TXD
CTS–5 —20—DTR
SG—7 — 7—SG
DTR–20 — 5—CTS
DSR–6 — 4—RTS
RTS–4 — 6— DSR





AT DB9 (male) to 7228 DB25 (female)

AT — 7228
CD—1 — 8—CD
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

HEX FORMATS

Intel Format

Data Record

Byte Number	
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 Number	
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

Extended Address Record (MCS-86 hex format)

Byte Number

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

Start Address Record (MCS-86 hex format)

Byte Number

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'scompliment** 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 Number

1—2	"S0"
3—n	Comment field
x+1—x+2	CR,LF

Data Records

Byte Number

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.

Byte Number

1—2	"S2"
3—4	Number of data bytes + 4. (2 characters)
5—10	Load address, 24 bits (6 characters)
11—x	Data bytes, 2 characters each.
x+1—x+2	Checksum (2 characters).
x+3—x+4	CR,LF.

Byte Number

1—2 "S3"

3—4 Number of data bytes + 5.

5—12 Load address, 32 bits (8 characters)

13—x Data bytes, 2 characters each.

x+1—x+2 Checksum

x+3—x+4 CR,LF.

End Record

Byte Number

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.

Tektronix Hex Format

Data Blocks

Byte Number	
1	Header (which is a forward slash- /)
2—5	Location counter which is 4 ascii-hex characters representing the load address of the data bytes.
6—7	Byte count which is 2 ascii hex bytes specifying the number 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-hex 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. (CR)

Termination Block

Byte Number	
1	Header (forward slash /)
2—5	Transfer address which is the address for execution of code.
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.
10	Always a carriage return. (CR)

Abort Block

Byte Number	
1	Header forward slash /
2	Header forward slash /
3—X+69	Message up to 69 characters for error information etc.
X+70	Always a carriage return. (CR)

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

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.

Using Interface Program PGMX and PGMX7

Installation of PGMX and PGMX7

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

On the PGMX program disk you will have at least 3 programs: PGMX.COM, PINSTALL.COM and GHEX.EXE. PGMX is the program used to communicate with your 7228. PINSTALL is the program that you must run to install the serial drivers in PGMX so that you can communicate with the 7228. Other programs and document files are provided to allow conversion from Motorola format to Intel hex and other programs to split and interleave to and from 8, 16 and 32 bit binary formats.

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.

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

C> **COPY 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.

C> **PINSTALL<cr>**

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 with Version 8.xx will probably select to set up to communicate at 19,200 baud on computer serial port COM1: or the selection for 19,200 baud on COM2:. People with Version 7.xx will use 2400.

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 C>**PGMX<cr>** 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:

C> **PGMX<cr>**

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

C> **PGMX FILENAME<cr>**

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.

C> **PGMX FILENAME [OPTIONS]<cr>**

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)
binary mode select (default is HEX)	
@sssss-eeeee	read bounds
Mx	menu selection
Tx	toggle command (3 max on command line)
Vsssss-eeeee	verify erasure
D	display data as it is being received from the 7228

MORE EXAMPLES

PGMX<cr> from the DOS command line establishes communication with the 7228, and after log-on displays the 7228 Command Prompter, which is the currently selected eprom type.

LOG ON MESSAGE EXAMPLE

(remember these are examples and your display may not be exactly like this one!)

C> **pgmx<cr>**
High Speed Interface Package Version 9.33
Copyright 1983, 1984, 1986, 1987 GTEK, INC.
All Rights Reserved, worldwide.
I/O Hardware Driver Vers 1.01 - IBM PC/AT
Serial port - COM1, 19,200 bps
Printer port - LPT1:

GTEK,INC.
MODEL 7228 V8.26

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<cr>** to get a menu:

2732> **M<cr>**

EPROMSELECTIONMENU

NMOS	NMOS	CMOS	EEPROM	W/ADAPT
A– 2758	H– 2516	L– 27C16	P– 5213	R– 874x-1K
B– 2716	I– 2532	M– 27C32	Q– X2816A	S– 874x-2K
4– 2716B	+– 2532A		X– 48016	T– 874xH-1K
C– 2732	J– 2564	O– F27C64	Y– I2816A	U– 874xH-2K
D– 2732A	K– i68766	0– I27C64	9– X2864A	V– 8751
N– 2732B		3– I2817A	W– 8755	
E– 2764	Z– i27256	8– F27C256		!– 874xAH
1– i2764A	G– F27256			
F– 27128	7– i27512			
2– i27128A				

Enter Selection —>**2**
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.

i27128A> **TN< cr>**
C000
i27128A> _

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

i27128A>(**control-F**)

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 entering 027 on the numeric keypad. Pressing and holding the CONTROL - C key for instance 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 or PGMX command prompter. This command will work even though you may be in the process of programming, reading, verifying, etc., an eprom in the automated (control-F) mode.

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<cr>**

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 7228 command 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:

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 7228 command "R". The "%" and the "@" command are not valid commands for the 7228 except on the PGMX command line. They are used to give PGMX information, not the 7228. 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 characters (0-9 and A-F).
eeeee = 24 bit ending address, Hex characters.
ooooo = 24 bit offset amount, Hex Characters

A delimiter is a dash (—), a comma (,), a space ( ), a carriage return, or a line feed (ascii characters 2Dh, 2Ch, 20h, 0Dh or 0Ah). Carriage return and line feed are re presented by a <cr> or <lf>.

A FILENAME is a valid DOS filename to be used by PGMX to look for a file on the disk. In the case where a percent (%) sign is specified, 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.

An EXT is 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 7228 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.

2- @sssss-eeeee. An @ symbol followed by the starting address (ssss) followed by a dash (-) followed by the ending address (eeee) will cause PGMX to search through the specified FILENAME to find the specified locations inclusive to be sent to the 7228. 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 (offset), will go to the ssss-eeee specified by the @ sign within the eprom, and eeee less ssss bytes will be sent. 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) inclusive, and send that data to the same locations within the eprom.

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). Any offset you specify (ooooo) will cause PGMX to scan up to that location in the file before sending any data to the 7228.

EXAMPLES:

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

xxxx> **MB**
2716> **^F**
Enter Command line -->**TEST.BIN[%0,@0-7FF<cr>**

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.

2716> **^F**

Enter command line-->**TEST.BIN[%800,@0-7FF<cr>**

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**

Enter command line ->**TEST.BIN[%1000,@0-7FF<cr>**

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.

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

C> **pgmx filename[r<cr>**

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

C> **pgmx filename[r,%<cr>**

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. Use the @ command to read between specified locations within an eprom.

C> **pgmx [tn,ma< cr>**
or
2716> **^F**
enter selection —>**[tn,ma <cr>** (from within PGMX)

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

ADVANCED EXAMPLE

C> **pgmxfilename[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.

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.

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.

Batch file automation

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

TEST.BAT

```
pgmx  test.bin[mb,v,@0-7ff,%%0,tn
pause remove eprom, insert new blank
pgmx  test.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)

Error return codes for batch file processing:

These error return codes may be used by a calling batch file or process which drives a chip handler like those manufactured by EXATRON. **PGMX7** can **NOT** be used like this. **ONLY PGMX!**

ERCODE=1 for any 7228 error messages (like *NE, or *WP)
ERCODE=2 for PGMX aborted by user with control-C
ERCODE= 5 for PGMX aborted by a disk error like "file not found"
or disk full or any command syntax error like "option error"
ERCODE=6 for PGMX when it was expecting a response from the
7228 and a timeout occured before any response was
received.

ERROR.BATprogram:

```
echo off
pgmx %1
if errorlevel 6 goto :lostcom
if errorlevel 5 goto :syssner
if errorlevel 2 goto :abort
if errorlevel 1 goto :badpart
echo This part programmed ok.
goto :enbat

:lostcom
echo You have lost communications with the programmer
goto :enbat

:syssner
echo There is a disk system error or a syntatical error.
echo Example, PGMX cannot find the file you specifyed or
echo  you are trying to use a command that does not exist
echo  or if you are reading a file maybe the disk is full!
goto :enbat

:abort
echo Someone typed a control C while the file was transferring. The
echo   program has been aborted.
goto :enbat
```

```
:badpart
echo The Eprom programmer issued an error such as *WP or
echo *NE or *DT or any other error which it might issue.
echo In any case you should reject echo this part.
goto :enbat

:enbat 1
echo  done
```

The above batch file will allow you to automatically program an eprom and abort if there are any problems. Add to it any other commands or programs nececssary for your specific application.

Other programs available:

Note the following programs are written so you can compile them easily with QuickBasic. We don't guarantee these programs to be error free, but they should present no problem to the experienced user.

CBIN.BAS:
A program to calculate a checksum from a binary file. The file must contain the exact number of bytes that fits in the eprom for you to get the same checksum as the TN command will give you (unless you specify boundaries with TN).

CHEX.BAS:
A program to calculate a checksum from an Intel Hex file. The file must contain data for every byte in the eprom. A file that does not fully program the eprom will not give the same checksum as the TN command unless you know what part and how much of the eprom is not programmed.

INTR16.BASandINTR32.BAS
Programs to combine 2 (intr16) or 4 (intr32) 8 bit BINARY files into 1 binary file.

SPLIT16.BASandSPLIT32.BAS
Programs to split 1 BINARY file into 2 (split16) or 4 (split32) binary files to program sets of eproms. The 7228 already has a 16 bit split mode,

and it may be faster to split the 16 bit file with the 7228 since basic runs so slowly (unless it's compiled). You would have to split a 32 bit file with the basic program first to obtain 4 8 bit files.

S_TO_HEX.COM
Program to take a Motorola Hex file and convert it to an Intel Hex file. It takes input from the keyboard and outputs it to the console. To modify whole files, use the DOS redirection commands:

Example: C>S_TO_HEX <moto.mik >intel.hex
will take a Motorola mik or ptp file by the name of MOTO.MIK and convert it to an Intel hex file by the name intel.hex.

—Notes—

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:

C> **GHEX filename.ext< cr>**
OR
C> GHEX **filename.ext offset**

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

C> **GHEX filetest.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.

C> **GHEX filetest.bin AA55**

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.

—Notes—

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.

C> **DEBUG<cr>**
—_

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

—**Nfilename.HEX<cr>**
—_

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<cr>**

—_

The CX register now contains the number of bytes read into memory with an offset of 100 (hex). 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<cr>**
CX: **1000<cr>**
—_

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< cr>**
—_

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<cr>**

—_

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<cr>**
Writing 1000H bytes
—_

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

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 1 year 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.

ALL EXPRESS AND IMPLIED WARRANTIES FOR THIS PRODUCT INCLUDING, BUT NOT LIMITED TO, THE WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE LIMITED IN DURATION TO THE ABOVE 1 YEAR 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.

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 along with the invoice number or a copy of the old invoice. Send the programmer, freight prepaid to:

GTEK,INC.
RMA Number #####
399 Highway 90
Bay St. Louis, MS 39520

Be sure to include the RMA on the shipping label and in the package so we will know what to do with it. Out of warranty service charges are determined on an hourly labor plus materials basis.

PGX, PGMX and PGMX7 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 condiions 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.

PGX, PGMX and PGMX7 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 TO STATE.

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.

Limitations of Remedies

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.

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 2310
Bay St. Louis, MS 39521-2310

—Notes—

Appendix A

Introduction

Parts in the follwing list are listed by manufacturer. In most cases you could use a "generic" selection directly from the menu you get from the 7228, except for the notable exceptions of the 27256.

If you don't see your part on the list, you may send a data sheet to GTEK or try calling GTEK to see if we can tell you about a particular part. Be sure to have a data sheet handy when calling unless you have not been able to get one, in which case we may or may not be able to tell you if it will program or how to program it.

General Rules

1- "A", "B", "H", or "AH" version parts program at lower voltages than their predecssors. If you try to program, verify, list or output (read) one of these parts using the wrong algorithm, the part will probably be destroyed in microseconds due to over voltage on the programming pin. The part may appear to be OK and may even still contain data that you had in the part previously, but if you erase it and then try to program it, you will probably get a *WP err @ 0000. This goes for MPU's also.

2- Cmos eproms generally use different algorithms to program than the nmos parts, but if the voltage is the same, you might try the nmos equivalent algorithm if you want to try programming the part adaptively. Some of the newer cmos eproms use the same algorithm as the nmos part, like the Intel 27C276; use menu selection Z.

3- Roms are generally readable on the 7228 if you take precautions to not use a selection that is going to use the verify mode to read it. If you are not sure, use a spec sheet for the menu selection/part number you would like to use and check the Vpp pin for that part during reads (OI or L commands) to see if programming voltage appears there. This is done with NO part in the socket of course. Some of the parts which do not use Vpp during reads are 27512, F27C256, 68766, F27C64, 27C32,

and 2532. This may not always hold true on the 7228, however. Rom equivalents of MPU's may only be read after modifying the calibration of the 7228 and in some cases the adapter socket.

4-Roms may be masked to use what would be address lines on eproms, as chip select lines. This means that a rom may have several chip enables. You may be able to read the part between the addresses of an eprom which forms the appropriate chip enable combination. Some roms may have no eprom equivalents with respect to the additional chip enable lines.

Appendix B

Adapters for the 7228

481— for MCS-48 family parts: 8741, 8742, 8742H, 8748, 8748H, 8749, 8749H

483—for MCS-48A family parts: 8742AH

511—for MCS-51 family parts: 8744, 8751, 8751H
Do Not try to program an 87C51 on the 7228. Use a Model 9000 programmer instead.

755- for 8755, 8755A Pio chip

Adapters for these parts are made so that the handle of the adapter matches the handle of the programming socket as per the illustration below.

Manufacturer's Cross Reference vs Menu Selection

GTEK believes that the information contained in this list is correct. However, GTEK assumes no responsibility or liability for the accuracy of this list.

These Notes are beside the Eprom selections on the following pages:

1–Standard algorithm only is available for this part, which is typically 50ms. EEproms may use a 10ms algorithm as the standard algorithm.

2–Adaptive algorithm only is available for this part, which is typically 1ms X total number of cycles to program (15-25) + overprogram pulse of 3 or 4 times the total number of cycles.

3–Can use Standard or Adaptive algorithm. Some parts may not program with Adaptive algorithm. The selection shown is for the recommended algorithm. If your part will not program, try the alternative algorithm.

4–Use Model 481 adapter with this part and selection on the 7228 programmer. Uses Standard algorithm only.

5–Use Model 483 adapter with this part and selection. Uses Adaptive algorithm only. Programming the security byte on the 8742AH chip is accomplished by programming data 0FFh at location FF1Fh.

6–Use Model 511 adapter with this part and selection. Uses Standard algorithm only. Programming the security byte on the 8751 or 8744 chip is accomplished by programming data 00h at location 0FFFFh. The data in location 0FFFh in the 8751 may be anything but zero, or else the security byte will not program.

7–This Fujitsu 12.5 volt algorithm selection is different from the Intel selection by the use of the -CE pin. This selection was 2508 on earlier versions, (before V7.11).

8–7228 Version 7.12 or later.

9–Use Model 755 adapter with this part and selection. Uses Standard algorithm only.

10–This CMOS part would normally use the standard algorithm. You can use the NMOS equivalent part selection to program this part Adaptively (TI command).

11–These parts are programmed using a 705 adapter (programmer) through programming a 2732 or 2732A on a GTEK (or any) programmer and then putting the 2732 into the 705, so the 68705 can copy the data from the 2732 into the Eprom of the 68705.

12–The "C" after the part number in this case deontes the case style. There are other "C" parts now available that program with Intelligent algorithms with 12.5 volts rather than the 21 volts of this part. Using the 21 volt selection will damage a 12.5 volt part beyond use. If you have a "C" part, call GTEK for details on programming it.

13–These parts may REQUIRE the Adaptive algorithm. TI started producing chips using a fast algorithm without changing their part numbers. You may not be able to determine which algorithm to use with these parts. To be safe, always use the Adaptive algorithm with these parts. Programming with the dumb algorithm might damage the part.

GTEK believes that the information contained in this list is correct. However, GTEK assumes no responsibility or liability for the accuracy of this list.

To select a part using a GTEK programmer, do the following: From the default power up prompter you type "M" and then you can either type the selection or hit return:

xxxx> **ME**
i2764> **M**

Menu appears here. Make your part selection.

enter selection --> **E**
<i2764>_

The part you select can usually use several different algorithms. Different GTEK models use different algorithms on power-up default. The 7228 uses the standard algorithm as default unless you change it on all parts capable of the adaptive algorithm, except for the parts that

require the adaptive algorithm like the 2764A. For instance a 2764 in most cases can be programmed adaptively.

```
2764> Ti
i2764> ME
2764>
```

Some parts default to the adaptive algorithm, like 2764A for example:

```
2764> M1
i2764A>_
```

In the following list under the menu selections, the suggested algorithm is shown. When using the interactive terminal mode or PGMX to select a part, a "ti" might also be shown. That means to also select the intelligent algorithm for that part.

AMD

EPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
AM2716	25.0	N	mb	2K	3
AM2716B	12.5	N	m4	2K	2
AM2732	25.0	N	mc,ti	4K	3
AM2732A	21.0	N	md,ti	4K	3
AM2732AP	21.0	N	md	4K	3
AM2732B	12.5	N	mn	4K	2
AM2764	21.0	N	me,ti	8K	3
AM2764P	21.0	N	me,ti	8K	3
AM2764A	12.5	N	m1	8K	2
AM2764AP	12.5	N	m1	8K	2
AM27128	21.0	N	mf,ti	16K	3
AM27128A	12.5	N	m2	16K	2
AM27256	12.5	N	mz	32K	2
AM27512	12.5	N	m7	64K	2

MPU'S:

Part #	Volts	Type	Menu	Size	notes
8741	25.0	N	mr	1K	4
8742H	21.0	N	mu	2K	4
8748	25.0	N	mr	1K	4
8748H	21.0	N	mt	1K	4
8749	21.0	N	mu	2K	4
8749H	21.0	N	mu	2K	4

ATMEL

EPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
AT27256	12.5	N	mz	32K	2
AT27C256	12.5	C	mz	32K	2

DALLAS SEMICONDUCTOR
BATTERY BACKED STATIC RAM:

N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
DS1220	TTL	N	mq	2K	1
DS1225	TTL	N	m9	8K	1

FUJITSU
EPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
MBM2764	21.0	N	me,ti	8K	3
MBM27C64	21.0	C	me,ti	8K	3
MBM27128	21.0	N	mf,ti	16K	3
MBM27C128	21.0	C	mf,ti	16K	3
MBM27256	12.5	N	mg	32K	7
MBM27C256	21.0	C	m8	32K	2
MBM27C256A	12.5	C	mg	32K	7
MBM27C512	12.5	C	m7	64K	2

MPU'S:

8742H	21.0	N	mu	2K	4

GENERALINSTRUMENT
EPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
27C64	12.5	C	m1	8K	2
27HC64	12.5	C	m1	8K	2
27C128	12.5	C	m2	16K	2
27256	12.5	C	mz	32K	2
27C256	12.5	C	mz	32K	2

HITACHI
EPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
HN482716G	25.0	N	mb	2K	3
HN482732G	25.0	N	mc,ti	4K	3
HN482732AG	21.0	N	md,ti	4K	3
HN482764G	21.0	N	me,ti	8K	3
HN482764P	21.0	N	me,ti	8K	3
HN27C64	21.0	C	me,ti	8K	3
HN4827128P	21.0	N	mf,ti	16K	3
HN27128A	12.5	N	m2	16K	2
HN27256G	12.5	N	mz	32K	2
HN27512	12.5	N	m7	64K	2

EEPROMS:

HN58064P	TTL	N	m9	8K	1

INTEL

EPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
2758	25.0	N	ma	1K	3
2716	25.0	N	mb	2K	3
2732	25.0	N	mc,ti	4K	3,17
2732A	21.0	N	md,ti	4K	3,17
P2732A	21.0	N	md,ti	4K	3,17
2764	21.0	N	me,ti	8K	3,17
2764A	12.5	N	m1	8K	2,17
P2764A	12.5	N	m1	8K	2,17
27C64	12.5	C	m1	8K	2,17
87C64	12.5	C	m1	8K	2,17
27128	21.0	N	mf,ti	16K	3,17
27128A	12.5	N	m2	16K	2,17
27256	12.5	N	mz	32K	2,17
P27256	12.5	N	mz	32K	2,17
27C256	12.5	C	mz	32K	2,17
87C256	12.5	C	mz	32K	2,17
27512	12.5	N	m7	64K	2,17
P27512	12.5	N	m7	64K	2,17

EEPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
2816A	TTL	N	my	2K	1
2817A	TTL	N	m3	2K	1
2864	TTL	N	m9	8K	1

Intel MPU'S:

Part #	Volts	Type	Menu	Size	notes
8741	25.0	N	mr	1K	4
8742H	21.0	N	mu	2K	4
8742AH	12.5	N	m!	2K	5
8748	25.0	N	mr	1K	4
8748H	21.0	N	mt	1K	4
8749H	21.0	N	mu	2K	4
8751	21.0	N	mv	4K	6
8751H	21.0	N	mv	4K	6
8744H	21.0	N	mv	4K	6
OTHER:					
8755	25.0	N	mw	2K	9

MITSUBISHI EPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
M5L2716K	25.0	N	mb	2K	3
M5L2732K	25.0	N	mc,ti	4K	3
M5L2764K	21.0	N	me,ti	8K	3
M5L27128	21.0	N	mf,ti	16K	3
M5M27C128	21.0	C	mf,ti	16K	3
M5L27256	12.5	N	mz	32K	2
M5M27C256K	12.5	C	mz	32K	2
M5L27512	12.5	N	m7	64K	2

MOTOROLA
EPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
MCM2716	25.0	N	mb	2K	3
MCM2532	25.0	N	mi	4K	1
MCM68732	25.0	N	mc,ti	4K	3
MCM68764	25.0	N	mk	8K	2
MCM68766	25.0	N	mk	8K	2

EEPROM:

MCM2833	TTL	N	m9	4K	1
MCM2864	TTL	N	m9	8K	1

MPU'S:

MC68705P3	21.0	N	(note)	1K	11
MC68705P5	21.0	N	(note)	1K	11
MC68705R3	21.0	N	(note)	2K	11
MC68705R5	21.0	N	(note)	2K	11
MC68705U3	21.0	N	(note)	2K	11
MC68705U5	21.0	N	(note)	2K	11

NATIONAL
EPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
MM2758	25.0	N	ma	1K	1
MM2716	25.0	N	mb	2K	3
NMC27C16	25.0	C	mb	2K	3
NMC27C32	25.0	C	mc,ti	4K	3
NMC27C64	12.5	C	m1	8K	2
NMC27CP128	12.5	C	mz	16K	2
NMC27C256	12.5	C	mz	32K	2
NMC27C512	12.5	C	m7	64K	2

EEPROM:

NMC98C64A	TTL	N	m9	8K	1

NEC

EPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
uPD2716D	25.0	N	mb	2K	3
uPD2732D	25.0	N	mc,ti	4K	3
uPD2732C	25.0	N	mc,ti	4K	3,12
uPD2732AD	21.0	N	md,ti	4K	3
uPD27C32D	21.0	N	md,ti	4K	3
uPD2764D	21.0	N	me,ti	8K	3
uPD2764C	21.0	N	me,ti	8K	3,12
uPD27C64D	21.0	C	me,ti	8K	3
uPD27C64C	21.0	C	me,ti	8K	3
uPD27128D	21.0	N	mf,ti	16K	3
uPD27128C	21.0	N	mf,ti	16K	3,12
uPD27256D	21.0	N	m8	32K	2
uPD27256C	21.0	N	m8	32K	2,12
uPD27C256D	21.0	C	m8	32K	2
uPD27C256C	21.0	C	m8	32K	2,12

MPU'S:

8741	25.0	N	mr	1K	4
8742H	21.0	N	mu	2K	4
8748	25.0	N	mr	1K	4
8748H	21.0	N	mt	1K	4
8749H	21.0	N	mu	2K	4

OKI

EPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
MSM2764	21.0	N	me,ti	8K	3
MSM27128	21.0	N	mf,ti	16K	3

ROCKWELL

EPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
87C64	12.5	C	m1	8K	2

SEEQ
EPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
5133	21.0	N	me	8K	3
5133H	21.0	N	me	8K	3
5143	21.0	N	mf	16K	3
27256	12.5	N	mz	32K	3
27C256	12.5	N	mz	32K	3

EEPROMS:

DQ2816A	TTL	N	my	2K	1
DQ2817A	TTL	N	m3	2K	1
DQ2864	TTL	N	m9	8K	1
DQ28C64	TTL	C	m9	8K	1
5212	TTL	N	mp	1K	1
5213	TTL	N	mp	2K	1
52B13	TTL	N	mp	2K	1
52B23	TTL	N	m9	4K	1
52B33	TTL	N	m9	8K	1
52B13H	TTL	N	m9	2K	1
52B23H	TTL	N	m9	4K	1
52B33H	TTL	N	m9	8K	1

SIGNETICS
EPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
27C64	12.5	C	m1	8K	2
87C64	12.5	C	m1	8K	2
27C256	12.5	C	mz	32K	2
87C256	12.5	C	mz	32K	2

SGS
EPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
M2716	25.0	N	mb	2K	3
M2716P	25.0	N	mb	2K	3
M2732A	21.0	N	md,ti	4K	3
M2732AP	21.0	N	md,ti	4K	3
M2764	21.0	N	me,ti	8K	3
M2764P	21.0	N	me,ti	8K	3
M2764A	12.5	N	m1	8K	2
M2764AP	12.5	N	m1	8K	2
M27128A	12.5	N	m2	16K	2
M27256	12.5	N	mz	32K	2
M27512	12.5	N	m7	64K	2

SMOS
EPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
27C64	21.0	C	me,ti	8K	3
27128	21.0	N	mf,ti	16K	2
27C256	12.5	C	mz	32K	2

EEPROM:

2864	TTL	N	m9	8K	1

TEXASINSTRUMENTS

EPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
TMS2516	25.0	N	mh	2K	3
TMS2532	25.0	N	mi	4K	1
TMS2532A	21.0	N	m+	4K	2
TMS2732	25.0	N	mc,ti	4K	13
TMS2732A	21.0	N	md,ti	4K	13
TMS27P32A	21.0	N	md,ti	4K	13
TMS2564	25.0	N	mj	8K	13
TMS2764	21.0	N	me,ti	8K	13
TMS27P64	21.0	N	me,ti	8K	13
TMS27C64	12.5	C	m1	8K	2
TMS27C128	12.5	C	m2	16K	2
TMX27PC128	12.5	C	m2	16K	2
TMS27C256	12.5	C	mz	32K	2
TMX27PC256	12.5	C	mz	32K	2
TMX27C512	12.5	C	m7	64K	2

TOSHIBA

EPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
TMM2764D	21.0	N	me,ti	8K	3
TMM2764DI	21.0	N	me,ti	8K	3
TMM2764AD	12.5	N	m1	8K	2
TMM2464AP	12.5	N	m1	8K	2
TMM27128D	21.0	N	mf,ti	16K	3
TMM27128DI	21.0	N	mf,ti	16K	3
TMM27128AD	12.5	N	m2	16K	2
TMM24128AP	12.5	N	m2	16K	2
TMM27256D	21.0	N	m8	32K	2
TMM27256DI	21.0	N	m8	32K	2
TMM27256AD	12.5	N	mz	32K	2
TMM24256AP/F	12.5	N	mz	32K	2
TC57256D	21.0	N	m8	32K	2
TMM27512D	12.5	N	m7	64K	2

VLSI
EPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
VT27C64	12.5	C	m1	8K	2
VT27C128	12.5	C	m2	16K	2
VT27256	12.5	C	mz	32K	2

XICOR
EEPROMS: N=nmos, C=cmos

Part #	Volts	Type	Menu	Size	notes
X2816A	TTL	N	mq	2K	1
X2864A	TTL	N	m9	8K	1
X28C64	TTL	C	m9	8K	1

GTEK believes that the information contained in this list is correct. However, GTEK assumes no responsibility or liability for the accuracy of this list.

Appendix D

GTEK is a registered trademark and PGMX, PGX, GHEX, Model 9000, Model 7228 are trademarks of GTEK, Inc.

AMD is a registered trademark of Advanced Micro Devices, Inc.

ATMEL is a registered trademark of ATMEL Corporation.

CP/M is a registered trademark of Digital Research Incorporated.

Cypress is a registered trademark of Cypress Semiconductor Corporation.

Dallas Semiconductor is a registered trademark of Dallas Semiconductor Corp.

Exel is a registered trademark of Exel Microelectronics, Inc., a subsidiary of Exar Corporation.

Fujitsu is a registered trademark and Quick Pro is a trademark of Fujitsu Microelectronics Incorporated.

GI, General Instrument are registered trademarks of General Instrument Corporation.

Hitachi is a registered trademark of Hitachi America, Ltd.

IBM is a registered trademark, and PC, XT, AT, PS/2 are trademarks of International Business Machines Corporation.

ICT is a registered trademark of International CMOS Technology, Inc.

Intel is a registered trademark and Intelligent, MCS–86, QuickPulse are trademarks of the Intel Corporation.

MS–DOS is a registered trademark and DOS and QuickBasic are trademarks of Microsoft Corporation.

Mitsubishi is a registered trademark of Mitsubishi Electronics America, Inc.

Motorola is a registered trademark of Motorola Inc.

National is a registered trademark of National Semiconductor Corporation.

NEC is a registered trademark of NEC Electronics Inc.

OKI is a registered trademark of OKI Semiconductor Inc.

Rockwell is a registered trademark of Rockwell International Corp.

Samsung is a registered trademark of Samsung Semiconductor Inc.

Seeq is a registered trademark of Seeq Technology Inc.

Sidekick is a trademark of Borland, International.

Signetics is a registered trademark of Signetics Corporation.

SGS is a registered trademark of the SGS Group.

ST is a trademark of SGS–Thomson Microelectronics

SMOS is a registered trademark

Tektronix is a registered trademark of Tektronix, Inc.

Texas Instruments is a registered trademark of Texas Instruments, Inc.

Textool is a registered trademark of 3M.

Thomson–Mostek is a registered trademark of Thomson Components – Mostek Corporation.

Toshiba is a registered trademark of Toshiba America Inc.

VLSI is a registered trademark of VLSI Technology Inc.

WaferScale is a registered trademark and RPROM is a trademark of WaferScale Integration Inc.

Xicor is a registered trademark of Xicor, Inc.