TS 803H USER'S MANUAL ADDENDUM

Addendum Part No.:	127223-00 Rev. A
Date Issued:	August 9, 1984
Addendum For:	TS 803H User's Manual Date Issued: 15 November 1983 Part Number: 2317700 Rev. A

This addendum contains the following information:

1. A change in the switch setting information for two and four head drive systems.

2. The write-protect label on the system diskette must be removed before inserting it into the TS 803H.

3. A change in the GSX driver files supporting different printers and plotters. The list of printers that are supported has been expanded. The ASSIGN.SYS and GENGRAF.COM files have been changed. Other files have been removed from the diskette and new files added to the diskette.

4. The PARK utility program.

5. In addition, note that the hard disk read routine in the BIOS has changed. The hard disk read routine originally checked the BUSY status before reading in data from the hard disk controller. Now however, the hard disk read routine checks the BUSY status twice and also checks the data request bit before reading in data from the controller.

6. Information about the interrupt modem port driver for the MOST 3801 STI UART/TIMER chip.

Take the appropriate pages out of your manual and replace them with the corresponding pages in this addendum. Add the PARK utility program to the end of the appendices in your manual.

Figure 2-8
Rear Panel Switch

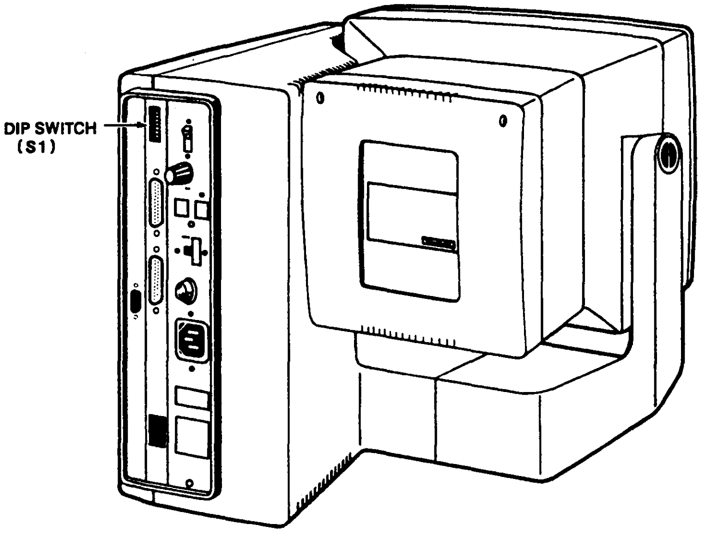


Figure 2-9
Default Switch Sections
(as set by TeleVideo)

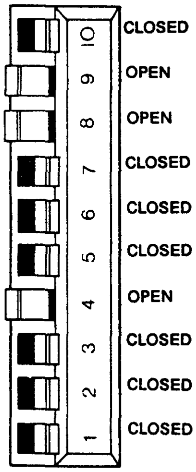


Table 2-2 lists all the possible settings for the DIP switch. Read through the table, and set the sections according to your requirements. The starred (*) settings are default settings (set by TeleVideo).

Table 2-2
Switch Settings

Section	Setting (set to right or left)	Function
1	closed (right)	*Baud rate (see Table 2-3)
2	closed (right)	*Baud rate (see Table 2-3)
3	closed (right)	*Baud rate (see Table 2-3)
4	closed (right) open (left)	TS 803 *TS 803H
5	closed (right) open (left)	*Local Remote
6	closed (right) open (left)	**Two head drive **Four head drive
7	closed (right) open (left)	*60Hz 50Hz
8	open (left)	Required
9	open (left) closed (right)	*Green on black screen (default) Black on green screen
10	closed (right)	*Not used

* Default settings
** Two head drive has one LED light on the drive
Four head drive has two LED lights on the drive

NOTE! If you are not connecting a printer or modem and are ready to use your system, go to the Checklist of Installation Procedures at the end of this chapter before continuing with Chapter 3, Power On.

If you wish to connect a printer or modem to your system, continue reading this chapter.

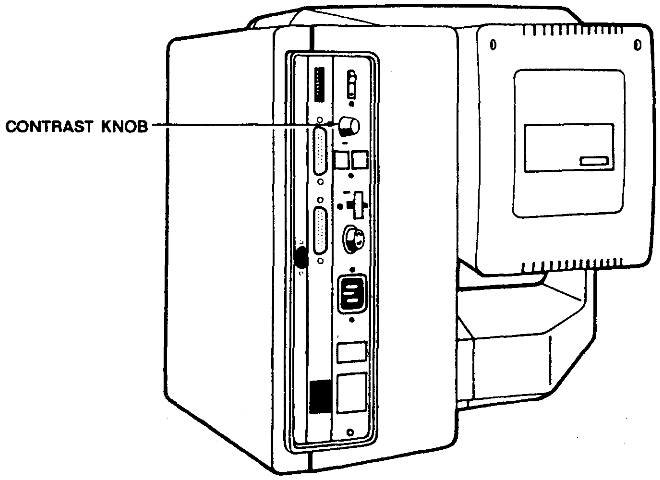
If you are adding a plotter or a mouse, please follow the installation instructions that accompany those devices. Also see Appendix K.

Prompts vary from one computer system to another but almost always indicate that the operating system is ready for your response. While using an applications program, you may see a different prompt symbol; this will be explained in the applications program instructions. The letter before the prompt itself (**A>**) tells which drive is currently being used. This is called the **logged drive.**

Adjusting the Contrast

Adjust the contrast by turning the contrast knob on the rear panel of the system for the desired screen intensity. See Figure 3-4.

Figure 3-4
Contrast Knob



Logical Drives

The TS 803H can be booted from either the floppy drive or the hard drive. The computer automatically first looks to the floppy drive for the system boot files. If they are not in the floppy drive, it looks to the hard drive.

You have booted the TS 803H from the system diskette in the floppy drive but for continual use, you should transfer the system diskette files to the hard drive and store the system diskette as a back-up copy.

There is a special utility program called SUBMIT that correctly and easily transfers all the necessary data from the system diskette to the hard drive. Follow the SUBMIT program carefully. The instructions are written in the script form as described in the Notation Convention section of Chapter 1.

Transferring the Software onto the Hard Disk

The SUBMIT program is used to transfer all of the software from the system diskette onto the hard disk. You need to have the BUILDDSK.SUB file, which is on the system diskette, to run this program. By using the BUILDDSK.SUB and SUBMIT programs, you format the hard disk, run the FIXDISK program, and transfer all the software from the floppy onto the hard disk.

USER: 1. Boot the system from the floppy drive (turn on the power, insert the system diskette with the write-protect label removed into the floppy drive, and close the drive).

SYSTEM: 2. Displays

```
TeleVideo System TS 803H Vx.y
(c) 1983 TeleVideo Systems, Inc.
Boot from floppy disk.
xxK CP/M version x.y

A>
```

USER: 3. Enter

SUBMIT BUILDDSK<CR>

SYSTEM: 4. Displays

```
     A>; TS-803H SOFTWARE INSTALLATION SUBMIT FILE
     A>; FOR INSTALLATION OF THE TS-803H
     A>;
     A>; FORMAT THE TS-803H HARD DISK - TYPE '<CR>' FOR INTERNAL DISK
     A>HFORMAT

     TS 803H Hard Disk FORMAT Vx.y
     10Mb Miniscribe III hard disk
     (c) TeleVideo Systems, Inc.

** Caution:  All data on the integral hard disk will be destroyed. **

               Type RETURN when ready (ESC or CTRL-C to abort)
```

USER: 5. Enter

<CR>

GDOS loads at run-time with your graphics applications program, so it consumes system memory space only when required, leaving the normal Transient Program Area for non-graphic programs.

GDOS performs coordinate scaling so that your program can specify points in a normalized coordinate space. It uses device specific information to translate the normalized coordinates into the corresponding values for your particular graphics device.

Multiple graphics devices can be supported under GSX-80 within a single application. By referring to devices with a workstation identification number, graphics information can be sent to any of the several resident devices. GDOS dynamically loads a specific device driver when requested by the applications program, overlaying the previous driver. This technique minimizes memory size requirements since only one driver is resident at any time.

Graphic Input/Output System

The Graphic Input/Output System (GIOS) is similar to the basic I/O system or BIOS. It provides the device specific code required to interface your particular graphics device to GDOS.

GIOS consists of a set of device drivers that communicate directly with the graphic devices through the appropriate host ports. A unique device driver is required for each different graphics device on your system. The term GIOS refers to the collection of available device drivers as well as the particular driver that is loaded into memory when required by your application. Although a single program can use several graphic devices, only one driver is loaded by GDOS at a time.

GIOS performs the graphics primitives of GSX-80 consistent with the inherent capabilities of your graphics device. In some cases a device driver will emulate standard GDOS capabilities which are not provided by the graphics device hardware. For example, some devices may require that dashed lines be simulated by a series of short vectors generated in the device driver.

GSX-80 is supplied with drivers for many of the most popular graphics devices for microcomputer systems.

Later in this document you will find a list of devices supported by the GSX supplied by TeleVideo for the TS 803, TS 1603, and TS 1602. Some manufacturers of graphics devices supply drivers for their devices.

GENGRAF UTILITY

The GENGRAF utility is used to combine your application program and the GSX loader into one executable .COM file. The GSX loader is a small program that loads the GDOS and GIOS into memory at run time and establishes the links between your applications program and GDOS. The GSX loader is attached to your applications program after it has been compiled/assembled and linked with the required external routines and libraries.

HOW GSX FITS INTO YOUR SYSTEM

Your applications program may be written in any language provided the GDOS protocol is observed. You may compile/assemble and link your application in the normal manner, yielding a .COM executable file. One additional step must be performed, however, before executing your graphics program: the GSX Loader must be attached to the front of your program so that it can prepare the operating system environment for your graphics application.

The GENGRAF utility (provided with the GSX-80 distribution) allows you to attach the loader to your program with one simple command:

```
     GENGRAF <filename>
```

For example, if your graphics application program were in an executable file named MYFILE.COM, then the following command string would attach the GSX Loader and place the result into file MYFILE.COM.

```
GENGRAF MYFILE
```

The resulting MYFILE.COM file would be ready to run.

You should be aware of the total memory space available to your application program in the TPA. This will be less for graphics applications than for normal programs because of the GDOS and device driver requirements.

GSX Supported Peripherals

TeleVideo supplies the following device drivers for their systems with graphic capabilities, like the TS 803, TS 1602, and TS 1603:

GSX supports:

- Printers
  - Anadex DP-9001A
  - Anadex DP-9501
  - Centronics 351
  - Centronics 352
  - Centronics 353
  - C.ITOH 8501A Low Resolution
  - datasouth DS180
  - DEC LA50
  - DEC LA100
  - Epson Low Resolution
  - Epson High Resolution
  - OKIDATA Microline
- Plotters
  - Hewlett-Packard 7470
- CRT device drivers
  - TeleVideo TS 803
  - TeleVideo TS 803H

GSX-86 supports:

- Printers
  - Epson MX-80 with graftrax Plus
  - Epson MX-80 type III
  - Epson FX-80
  - IDS MicroPrism 480
  - IDS Prism 80/132 (monochrome)
  - IDS Prism 80/132 (color)
  - Okidata Microline 92
  - Printronix MPV
  - Printronix P300/P600
- Plotters
  - Hewlett-Packard 7470A
  - Hewlett-Packard 7220
  - Houston Instruments HiPlot DMP-3/4-443
  - Houston Instruments HiPlot DMP-6/7
  - Strobe Model 100
- CRT device drivers
  - TeleVideo TS 1602
  - TeleVideo TS 1603 (plus graphic option)

GRAPHICS PRIMITIVES DRIVER

The EPROM firmware in the TS 803H serves three functions:

System boot and diagnostic routines
Alphanumerics Generator/Terminal Attributes Emulator
Graphics Primitives Driver

The Graphics Primitives Driver is the lowest level interface to the graphics memory. The Graphics Primitives Driver accesses the 20K segment of the 32K graphic memory block as a 640 x 240 pixel bit map through the CPU, mapping the graphics to be displayed by the video controller. The CPU updates the display on alternate cycles with video controller access.

The programmer can have programs access the display memory with calls to the Graphics Primitives Driver. The following sections describe the procedures by which calls can be made to the graphics driver.

Figure 9-1
Graphics Display

```
                  (Y)
                   ^
                   |
                   |---------------------   (639, 239)
                   |                     |
                   |       Screen        |
                   |                     |
                   |                     |
         (0, 0,)    ------------------------->(X)
```

X, Y coordinates in unit of pixel

The TS 803H is also equipped with an optical MOUSE interface. The mouse has ten software-support functions which are described in the Mouse Support Functions section of this chapter. These functions can be directly accessed by the user through a single-entry point.

CALLING THE GRAPHICS DRIVER

The following procedure enables you to access the graphics driver.

1. Load register "C" with the desired graphics function identifier as listed in Table 9-1.

2. Load the parameters as specified by Table 9-1 either into registers HL, and DE or into a **parameter array.**

NOTE! The address of the parameter array mentioned above is contained in address 0FF00h and 0FF01h.

0FF00h contains low byte of address.
0FF01h contains high byte of address.

3. Perform a Restart instruction (RST 28h in Z80 or RST 5 in 8080). The restart instructions direct the call to a location in the BIOS. It then jumps to the graphics entry point in the EPROM where the graphics function will be carried out.

Notice that the array for an extended parameter list is stored in high RAM memory so that it is still available after the memory bank switches from graphics mode to alpha mode and vice versa.

The following is an example of how to access the graphics driver in Z80 code. A complete list of examples of how to access the graphics driver is listed in the section of this chapter called "Sample Graphics Access Program Using 8080 Assembly Code."

Suggestion: it is very useful to create a macro for access to the

15 C = 15 **Fill Interior Style.** This routine sets the type of interior style to be used in the horizontal scan routine. The style selected is passed on the first word of the parameter array:

```
                -----------
array          | Style  #  | word 1
                -----------
```

Table 9-2
Fill Interior Styles

Style #	**Style**
0	Hollow
1	Solid
2	Pattern
3	Cross-hatch

16 C = 16 **Fill-Style Index.** This routine sets the style index of the currently-selected interior style. The style index only affects the pattern and cross-hatch interior styles. The style index is passed on the first word of the parameter array.

```
                 -------------
array          | Style Index | word 1
                 -------------
```

Table 9-3
Graphics Styles

Current Interior Style	**Style Index**	**Comment**
Hollow	None	Style Index has no effect.
Solid	None	Style Index has no effect.
Pattern	1	Low Intensity
	2	Low Intensity
	3	Low Intensity
	4	Low Intensity
	5	High Intensity
	6	High Intensity

Cross-Hatch	1	Vertical lines
	2	Horizontal lines
	3	+45 degree lines
	4	-45 degree lines
	5	Both vertical and horizontal cross-hatch
	6	Both +45 and -45 degree cross-hatch

17 C = 17 **Fill Color Index.** This routine affects the area style to be written into memory in the following way:

1 = white
2 = black

```
              -------------
array        | Color Index | word 1
              -------------
```

17 RESERVED RESERVED

19 C = 19
HL = y scan line value

GetPix. Returns all of a scan line of pixels. Returns the scan line in the array. Bytes in the array are packed with the most significant bit corresponding to the lower x value. The bytes are loaded into the array in order of increasing x (from left to right) on the scan line.

THE MOUSE

The TS 803H provides ten mouse support functions which you can access. The SETUP program allows you to attach or detach the mouse from the system. Once the mouse is attached, it must be initialized (function 1) before all the support functions become operational. The mouse opens a completely new dimension in human interface to the computer.

TS 803 VIDEO DISPLAY ATTRIBUTES DO NOT CORRESPOND TO RECENTLY-CHANGED SWITCH SETTINGS	Reset the system by pressing ^RESET so software can scan new DIP switch settings.
TS 803H "LOCKED UP" (DOES NOT RESPOND TO KEYBOARD)	Turn the drive latches to the vertical position and power the TS 803H off and back on. Turn the latches back to the horizontal position. Check the keyboard connector.
TS 803H SCREEN IS DEAD; CURSOR MIGHT APPEAR	Turn the TS 803H off and check the fuses.
PRINTER DOES NOT PRINT WHAT IS TYPED	Reconnect the printer cable, being sure it is completely on the pin connector. Check the printer cable configuration. Check for printer fault (out of paper or ribbon).
HARDWARE ERROR MESSAGE APPEARS	Reset by pressing ^RESET.
ESCAPE AND CONTROL COMMANDS DO NOT FUNCTION AS ANTICIPATED	Reenter commands, paying particular attention to whether uppercase or lowercase characters, ones or zeroes, are needed.
THE MESSAGE "DISK WRITE ERROR, TEMPORARY FILE : GSX.TMP" APPEARS	Indicates that there's not enough space on the system diskette to send the TS 803H demonstration program to a printer or plotter. Make a backup copy of the system diskette, erase the unneeded printer driver files from the system diskette you are using, and run the demonstration program.

CHANGING THE FUSE

The TS 803H has one exterior fuse which is accessible from the rear panel.

To check or change the exterior fuse, turn the power off, pull the plug out of the power outlet, and unscrew the fuse holder (see Figure 10-1).

The fuse is a small glass cylinder. If the thin wire inside the fuse is still intact, the fuse should be functional (see Figure 10-2). If the thin strip is broken and/or if the glass is slightly black, the fuse has blown and must be replaced (see Figure 10-3). If the glass is totally black, do not replace the fuse. The problem could be with the system or with the power outlet. Call your computer store.

Figure 10-1
Exterior Fuse Location

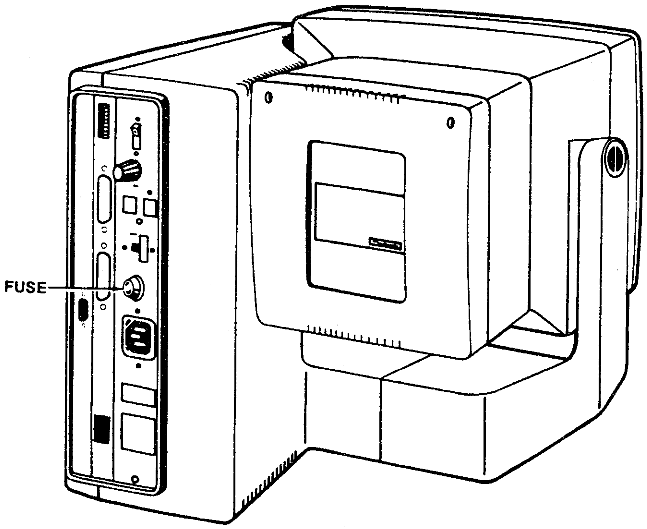


Figure 10-2
Good Fuse

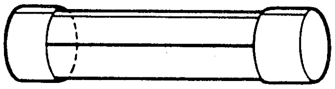
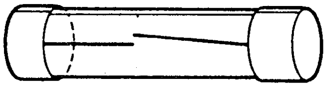


Figure 10-3
Blown Fuse



To replace the fuse, slip the old fuse from the fuse holder and insert a new fuse. The fuse rating is shown on a label above the fuse holder. See the section called Inserting the Fuse in Chapter 2 for more information about fuses. You can purchase new fuses from your computer store.

If the newly-replaced fuse blows out immediately, do not replace it with another fuse. Call your computer store.

APPENDIX F
COMMUNICATIONS PORT DRIVER

The TS 803 (BIOS 1.5 and above), TS 803H, and TeleVideo Portable Computer (TPC I) include an interrupt modem port driver for the MOSTEK 3801 STI UART/TIMER chip.

The following functions are added to the standard BIOS functions:

1. Modem Buffer Ready (this is the modem port input buffer status check)

```
    To BIOS:        C = 18

    From BIOS:      Z = 1, A = 0    (if modem input buffer is empty)

                    Z = 0, A = FF   (modem input buffer not empty)
```

2. Get Character From Buffer (get character from modem input buffer)

```
    To BIOS:        C = 19

    From BIOS:      Z = 0, A = char (input character if buffer is
                                      not empty)

                    Z = 1, A = undefined
                                     (modem input buffer empty)
```

```
NOTE!               Z = 0      Zero condition flag = false

                    Z = 1      Zero condition flag = true
```

3. Set modem REC_INT_VEC (resets the internal buffer or sets the modem receive interrupt vector to the user routine)

```
    a.   To BIOS:  C = 20
                  HL = 0          (clears the modem input buffer
                                  and resets the modem receive
                                  interrupt vector to the BIOS
                                  interrupt handling routine)

    b.   To BIOS:  C = 20
                  HL = user subroutine address
                                  (sets the modem interrupt
                                  vector to the user handling
                                  routine)

         From BIOS:  none
```

PROGRAMMING EXAMPLE #1

This is used to demonstrate how to use functions 18, 19, and 20. The program displays all incoming characters through the modem port onto the console.

Use the internal modem Rx buffer in the BIOS.

```
               BDOS EQU 5     CP/M BDOS ENTRY
START:         LXI  H,0       RESET MODEM RECEIVE BUFFER
               MVI  C,20
               RST  6
NOT_READY:     MVI  C,18      CHECK RECEIVE READY
               RST  6
               JZ   NOT_READY  WAIT IF NOT READY
               MVI  C,19      IF CHARACTER READY THEN
               RST  6         READ IT
               MOV  E,A       DISPLAY RCV CHARACTER
               MVI  C,2       ON CONSOLE
               CALL BDOS
               JMP NOT_READY
```

PROGRAMMING EXAMPLE #2

The following information provides an example of how to program the MOSTEK 3801 STI UART/TIMER chip on the TS 803, (BIOS version 1.5), TS 803H, and TPC I. This example uses its own interrupt driven modem receive handler, enabling the TS 803, TS 803H and TPC I to receive data from the modem port at data rates up to 9600 bps.

3801 STI REGISTER ADDRESSES

```
IDR       EQU       20H       INDIRECT DATA REG
GPIP      EQU       21H       GENERAL PURPOSE I/O REG
PVR       EQU       28H       POINTER/VECTOR REG
TABCR     EQU       29H       TIMER A AND B CONTROL REG
TADR      EQU       2BH       TIMER A DATA REG
UCR       EQU       2CH       USART CONTROL REG
RSR       EQU       2DH       RECEIVER STATUS REG
TSR       EQU       2EH       TRANSMITTER STATUS REG
UDR       EQU       2FH       USART DATA REGISTER
```

COMMUNICATIONS PORT DRIVER INTERFACE WITH SYSTEM BIOS

```
CHK$IN    EQU  18   CHECK MODEM PORT RCV. READY

GET$MD    EQU  19   GET A CHARACTER FROM MODEM RCV. BUFFER

SET$MIV   EQU  20   RESET/SET MODEM RCV. INTERRUPT VECTOR ADDRESS
```

COMMUNICATIONS PORT INITIALIZATION

The applications software calls this routine to initialize the communications port. This might be done when the program is first loaded, whenever configuration data is changed, when it is necessary to raise or lower DTR, etc.

On entry to the communications port initialization routine, registers B through E specify how the communications port is to be set up as listed Table 1.

Table 1
Registers B Through E

B = BAUD RATE	C = PARITY
0 = 150	0 = NONE
1 = 300	1 = ODD
2 = 600	2 = EVEN
3 = 1200	D = DATA BITS
4 = 2400	0 = 8 BITS/CHAR
5 = 4800	1 = 7 BITS/CHAR
6 = 9600	E = DTR STATUS

Whenever the communications port initialization is done, any waiting input characters should be discarded.

```
COMIN:    PUSH D         SAVE D AND E
          MVI D,0
          IN TABCR       READ CURRENT TIMER
          STA TABD       SAVE FOR LATER
          LXI H,BRBTB    GET BIT PATTERN FOR CORRECT BAUD RATE
          MOV E,B
          DAD D
          MOV A,M
          STA TADRD      BAUD RATE
          POP D
          MVI B,0
          LXI H,PTYTB
          DAD B
          MOV A,M        GET PARITY
          LXI H,DTBTB
          MOV C,D
          DAD B
          ORA M          COMBINE WITH DATA BITS
          ORI 88H        DIVIDE BY 16 AND 1 START/STOP BIT
          STA UCRD
          LXI H,DTRD     INITIAL DTR STATUS
          MOV A,E
          ORA A
          JZ CMIN6
          MVI A,20H      ASSERT DTR
CMIN6:    ORA M
          STA DDRD
          IN  PVR
          ANI 0E0H       GET STI INTERRUPT VECTOR AND MASK OFF DO
                         NOT CARE BITS
          MOV B,A        SAVE IT INTO REG-B
          ORI 6
          OUT PVR        POINT TO INDIRECT REG #6
          LDA DDRD       SET DDR
          OUT IDR
          SUB A
          OUT GPIP       SET HANDSHAKE LINES
          MOV A,B        RESTORE SYSTEM INTERRUPT VECTOR TO REG-A
          ORI 7          SET INDIRECT REG. POINTER #7
          OUT PVR
          IN  IDR        SAVE TIMER C AND D REGS
          ORI 088H       RESET TIMER A AND B (CHANNELS C AND D
                         ARE UNCHANGED)
          OUT IDR        RESET CHANNEL A AND B
          MVI A,0
          OUT TABCR      STOP A AND B
          LDA TADRD
          OUT TADR       SET BAUD RATE FOR MODEM PORT
          LDA TABD
          OUT TABCR      RESTART CHANNELS A AND B
          LDA UCRD
          OUT UCR        SET FORMAT
          MVI A,01H
          OUT RSR        TURN ON RXD
          MVI A,05H
          OUT TSR        TURN ON TXD
          LXI  H,0
          SHLD RCV$BUFF  CLEAR RECEIVER BUFFER (INPUT POINTER)
          SHLD RCV$BUFF+2(OUTPUT POINTER)
          LXI  H,MOD$RCV HL = RECEIVER INTERRUPT HANDLER ROUTINE
                         ADD.
          MVI  C,20      SET INTERRUPT VECTOR TO HL
          RST  6
          SUB A
          RET

TADRD:    DB 0           BAUD RATE STORAGE
TABD:     DB 0           A AND B CONTROL WORD STORAGE
UCRD:     DB 0           FORMAT STORAGE
DDRD:     DB 0           DTR AND RTS BITS
```

Table 2
Baud Rate Bits

```
BRBTB:    DB 64       150
          DB 32       300
          DB 16       600
          DB 8       1200
          DB 4       2400
          DB 2       4800
          DB 1       9600
```

Table 3
Parity Bits

```
PTYTB:    DB 0       NONE
          DB 04H     ODD
          DB 06H     EVEN
```

Table 3
Character Length Bits

```
DTBTB:    DB 0       8 DATA BITS
          DB 20H     7 DATA BITS

DTRD:     DB 04H     RTS HI DTR LOW
```

GET COMMUNICATIONS PORT INPUT STATUS

The application calls this routine whenever it wishes to determine if there is an incoming byte on the communications port available to be read from the UART. On return from this routine, register A should be zero if no byte is available, and should be non-zero if a byte is available.

```
COMIS:     LHLD      RCV$BUFF
           XCHG                     DE = INPUT OFFSET POINTER
           LHLD      RCV$BUFF+2
           XRA       A              CLEAR A AND CARRY FLAG
           DB        0EDH,52H       *** Z80 ***   SBC HL,DE
           RZ                       RETURN WITH A = 0 IF Z = 1
           CMA                      ELSE A = 0FFH
           RET
```

READ COMMUNICATIONS PORT INCOMING BYTE

The application calls this routine to read the incoming data byte into register A. This routine is read only if a previous call has indicated that there is a byte available.

```
COMIB:     JMP       GET$CHR        INPUT BYTE
```

GET COMMUNICATIONS PORT OUTPUT STATUS

The application calls this routine whenever it wishes to determine if the communications port UART is ready to accept another byte to be transmitted. On return from this routine, register A should be zero if the UART is not ready, and should be non-zero if the UART is ready.

```
COMOS:     IN TSR         READ STATUS REGISTER
           ANI 80H        LOOK ONLY AT RELEVANT STATUS BIT
           RET
```

OUTPUT BYTE TO THE COMMUNICATIONS PORT

The application calls this routine whenever it wishes to output a byte through the communications port UART. The byte itself should be supplied in register A (all 8 bits are sent.) The application will obviously only call this routine if it has been determined that the communication port is ready to accept a byte by calling the status routine above.

```
COMOB:     OUT UDR        OUTPUT BYTE
           RET
```

MODEM RECEIVE INTERRUPT HANDLER

The application uses the following interrupt handler for receiving data from the communications port UART. This is to enable receiving all incoming data, even when the Z80 is performing other functions such as absolute cursor positioning, screen clearing, etc., and to ensure that data is not lost because the processor may be performing these functions.

```
MOD$RCV:
          IN   RSR            CHECK RECEIVER STATUS FOR ERROR
          ANI  11110000B
          RP                  EXIT IF NO DATA AVAILABLE
          IN   UDR            READ RCV$BYTE FROM MODEM PORT TO REG-A
PUT$CHR:
          LXI  H,RCV$BUFF     HL = BUFFER POINTER
          LXI  B,RCV$BSIZ     BC = BUFFER SIZE
          PUSH H
          PUSH PSW
          MOV  E,M
          INX  H
          MOV  D,M
          PUSH D              SAVE INPUT OFFSET POINTER ONTO STACK
          INX  D
          CALL CMP$DEBC
          JC   PUT$C1
          LXI  D,0
PUT$C1:
          INX  H
          MOV  C,M
          INX  H
          MOV  B,M
          CALL CMP$DEBC
          POP  B
          JZ   PUT$C2
          INX  H
          DAD  B
          POP  PSW
          MOV  M,A
          POP  H
          MOV  M,E
          INX  H
          MOV  M,D
          ORA  A         SET CY = 0 (NO OVERFLOW)
          RET
```

BUFFER OVERFLOW

```
PUT$C2:
          POP  PSW
          POP  H
          STC            SET CARRY = 1 (BUFFER OVERFLOW)
          RET
```

Compares 16 bits integer DE and BC.

```
     Returns:  CY = 1   Z = 0 IF DE < BC
               CY = 0   Z = 1 IF DE = BC
               CY = 0   Z = 0 IF DE > BC
```

```
CMP$DEBC:
          MOV  A,D
          CMP  B
          RNZ
          MOV  A,E
          CMP  C
          RET
```

Gets character from the RCV_BUFFER.

Returns: A = CHARACTER

```
GET$CHR:
          LXI  H,RCV$BUFF+2
          MOV  E,M              GET CURRENT OUTPUT POINTER TO DE
          INX  H
          MOV  D,M
          INX  H
          DAD  D                INDEX TO CHARACTER IN BUFFER
          MOV  L,M              GET A BYTE TO REG-L
          INX  D                COMPUTE NEXT BYTE LOCATION IN BUFFER
          LXI  B,RCV$BSIZ
          CALL CMP$DEBC
          JNZ  GET$C1
          LXI  D,0
GET$C1:
          MOV  A,L              RESTORE CHARACTER INTO REG-A
          XCHG
          SHLD RCV$BUFF+2       ADDRESS OUTPUT POINTER
          RET
```

MODEM RECEIVER BUFFER AREA

Buffer Structure

```
RCV$BUFF:
          DW   0                MUST INITIALIZE TO ZERO (INPUT
                                OFFSET POINTER)
          DW   0                MUST INITIALIZE TO ZERO (OUTPUT
                                OFFSET POINTER)
RCV$BSIZ  EQU  (0600H-RCV$BUFF)-4  SIZE OF RECEIVER BUFFER
          DS   RCV$BSIZ          RING BUFFER DATA AREA
```

CBIOS LISTING OF MODEM DRIVER

Listed below is a segment of the TS 803 CBIOS listing which highlights the modem driver sections referenced in the example given above.

From 803CONS.MAC:

MODEM RECEIVER ROUTINE 6/9/83

```
MRX_INT:                           SYSTEM INTERRUPT DISABLE FROM HERE
          LD   (MRX_STACK),SP USER STACK
          LD   SP,MRX_STACK   SET STACK POINTER TO LOCAL
          PUSH AF             SAVE ALL REGISTERS (A,BC,DE,HL AND
                              FLAGS)
          PUSH BC
          PUSH DE
          PUSH HL
          LD   A,1
          OUT  (BANKSW),A     SWITCH BACK TO BANK #1 (TPA MEMORY)
```

If you want to handle the RCV-Buffer or error by using software, you must modifiy the three bytes of call instruction at location "MRX_IDR" to their subroutine. At the end, you must return to this routine handler.

User routines must use their own stack if using more than ten levels of stack. You must not alter any of the following registers: I, R, IX, IY, AF', BC', DE', HL', SP, and you must keep the system interrupt disabled.

```
MRX_IDR:
          CALL MRX_CHR        GET CHARACTER OUT OF RECEIVER REG.

          LD   A,(C_BANK)     GET PREVIOUS BANK #
          OUT  (BANKSW),A     RESTORE PREVIOUS BANK SELECT
          POP  HL             RESTORE ALL REGS (A, BC, DE, HL AND
                              FLAGS)
          POP  DE
          POP  BC
          POP  AF
          LD   SP,(MRX_STACK) RESTORE SYSTEM STACK POINTER
          EI                  ENABLE SYSTEM INTERRUPT
          RETI
```

MODEM RECEIVER ROUTINE 6/9/83

```
MRX_CHR:
          IN   A,(MODRSR)     CHECK RCV. STATUS
          AND  11111000B      READ STATUS TO CLEAR ERROR FLAGS.
          CP   80H            BUFFER READY AND ERROR ?
          RET  C              RETURN IF BUFFER NOT READY
          IN   A,(MODMDP)     READ RX. DATA TO A-REG
```

If Z = 0, then there is no error
IF Z = 1, then there is an error (OV, PE, FE, BREAK)??? *
You can put the RCV_CHR into the buffer anyway.

```
          LD   HL,MRX_BUFF    ADDRESS MODEM RECEIVER BUFFER
          LD   BC,MRX_BSIZE   BUFFER SIZE
```

Put the character into the ring buffer. If the buffer overflows, then ignore the input character.

```
     Accepts:  A  = INPUT CHAR.
               HL = BUFFER POINTER
               BC = BUFFER SIZE

     Returns:  CY = 0 IF CHAR --> BUFFER
               CY = 1 IF BUFFER FULL

PUT_CHR:
          PUSH HL             SAVE BUFFER POINTER
          PUSH AF             SAVE INPUT CHARACTER
          CALL CHK_OVFW       IS BUFFER OVERFLOW?
          JR   Z,PUT_C2       YES, OVERFLOW CANNOT STORE CHARACTER
          INC  HL
          ADD  HL,BC          INDEX TO CHARACTER SPACE
          POP  AF             RESTORE CHARACTER
          LD   (HL),A
          POP  HL
          LD   (HL),E         RESTORE START BUFFER POINTER
          INC  HL
          LD   (HL),D         UPDATE INPUT BUFFER POINTER
          OR   A
          RET

PUT_C2:
          POP  AF
          POP  HL
          SCF
          RET

CHK_OVFW:
          LD   E,(HL)         DE = INPUT POINTER
          INC  HL
          LD   D,(HL)
          PUSH DE             SAVE OLD INPUT INDEX POINTER
          INC  DE             BC = BUFFER SIZE
          CALL CMP_DEBC       DE > BC ?
          JR   C,PUT_C1       SKIP IF DE < BC
          LD   DE,0           ELSE DE = 0
PUT_C1:
          INC  HL
          LD   C,(HL)         BC = OUTPUT POINTER
          INC  HL
          LD   B,(HL)
          CALL CMP_DEBC       INPUT POINTER = OUTPUT POINTER ?
          POP  BC             RESTORE OLD INPUT INDEX POINTER TO
                              BC-REG

          RET
```

Check to see if the receiver buffer is empty.

```
     Returns:  Z = 1 If buffer is empty
               Z = 0 If buffer is not empty

CHK_MRX:
          LD   DE,(MRX_BUFF)
          LD   BC,(MRX_BUFF+2)
```

Compares 16 bits uns. integer.

```
     Accepts:  DE = 16 uns. int
               BC = 16 uns. int

     Returns:  Z = 0   CY = 1 IF DE < BC
               Z = 1   CY = 0 IF DE = BC
               Z = 0   CY = 0 IF DE > BC

CMP_DEBC:
          LD   A,D
          CP   B
          RET  NZ
          LD   A,E
          CP   C
          RET
```

Gets the character from the ring buffer.

```
     Accepts:  HL = Buffer pointer
               BC = Buffer size
     Returns:   Z = 1 if the buffer is empty
               ELSE A = Returns character

GET_CHR:
          PUSH BC              SAVE BUFFER SIZE
          LD   E,(HL)          GET INPUT BUFFER INDEX TO DE
          INC  HL
          LD   D,(HL)
          INC  HL
          LD   C,(HL)          GET OUTPUT BUFFER INDEX TO BC
          INC  HL
          LD   B,(HL)
          CALL CMP_DEBC        INPUT INDEX = OUTPUT INDEX ?
          POP  DE              RESTORE BUFFER SIZE TO DE (NOT BC)
          RET  Z               EXIT IF BUFFER EMPTY
          DI                   **** DISABLE SYSTEM INTERRUPT ****
          PUSH HL              ELSE SAVE OUTPUT INDEX BUFFER POINTER
          INC  HL
          ADD  HL BC           INDEX TO RX. CHARACTER IN BUFFER
          LD   L,(HL)          GET CHARACTER TO L-REG
          INC  BC              ADJUST OUTPUT INDEX POINTER TO NEXT
          CALL CMP_DEBC        OUTPUT INDEX = BUFFER SIZE ?
          JR   NZ,GET_C1       SKIP IF NOT EQUAL
          LD   BC,0            ELSE RESET OUPUT INDEX POINTER
```

```
GET_C1:
          LD   A,L              RESTORE CHARACTER INTO A-REG
          POP  HL               RESTORE OUTPUT INDEX BUFFER POINTER
          LD   (HL),B           UPDATE OUTPUT INDEX POINTER
          DEC  HL
          LD   (HL),C
          LD   C,0
          DEC  C
          RET
```

APPENDIX G
PIN CONNECTOR ASSIGNMENTS

Table G-1
Board Connectors

Connector No.	Description
P1	RS-232C Modem Connector
P2	Serial Printer Connector
P5	Power Connector
P6	Video Connector
P7	Floppy Disk Drive Connector
P8	WDC Board Connector
P9	Comp. Video Connector
P11	Keyboard Connector

Table G-2
RS-232C Modem Connector

Pin No.	RS-232C Designator	Description	I/O
1	AA	Frame Ground	G
2	BA	Transmit Data	O
3	BB	Receive Data	I
4	CA	Request to Send	O
5	CB	Clear to Send	I
7	BA	Signal Ground	C
8	CF	Data Carrier Detect	I
15	DB	Transmit Clock	I
17	DD	Receive Clock	I
20	CD	Data Terminal Ready	O
24	DA	Transmit Clock	O

Legend:

G = AC chassis ground
I = Input
O = Output
C = Signal common

APPENDIX L
SYSTEM DISKETTE FILE LIST

This is a list of the files that were on the TS 803H system diskette when it was initially released. TeleVideo reserves the right to change these files without notification to the customer. If you have any questions about the files on your diskette, contact your computer store.

Table L-1
System Diskette Files

Filenames	**Description**
MOVCPM.COM*	Program to relocate the CCP and BDOS to generate a CPMxx.COM file.
DDT.COM*	Debug program (Dynamic Debug Tool) for the assembly language.
SYSGEN.COM*	CP/M utility program to copy system tracks 0 and 1.
COPYDISK.COM	Utility program to copy an entire diskette.
GENGRAF.COM*	Utility to combine the applications program and the GSX loader into an executable command file.
DDTS803.PRL	GSX graphics driver to support the TS 803H.
PIP.COM*	CP/M utility program to copy files.
ED.COM*	CP/M text editing program.
LOAD.COM*	Program to generate an executable program (command file) from the HEX file generated by the assembler.
SETUP.COM	Utility program to reconfigure the system parameters.
DEMOSCRN	Contains the data for the TS 803H demo program.
GSX.SYS*	Graphics System Extension file (GSX loader).
SURF1	Graphics data for the TS 803 demo program.
STAT.COM*	CP/M utility program. For the capabilities of the STAT command, see Chapter 5.

SUBMIT.COM* CP/M utility program for submitting CP/M commands in a batch mode. Works with XSUB.COM.

DUMP.COM* Program to display a file in HEX numbers.

TOD.COM Time of Day program.

ASSIGN.SYS Graphics file that works with the demo program for determining the output device of the demo.

DEMOTXT2.BIN Data file used by the system demo.

ASM.COM* 8080 assembler program. Generates a HEX file which can be used by the load program.

XSUB.COM* Works with the SUBMIT file submitting CP/M commands.

FORMAT.COM Utility program to format a diskette.

GSXREAD.ME Explanation about GSX (Graphics System Extension) and the TS 803H demonstration program.

803HDEMO.COM System demo program.

PARK.COM Utility program to lock the hard disk.

HFORMAT.COM Utility program to format the hard disk.

FIXDISK.COM Utility program to inspect the hard disk for defective data areas.

BUILDDSK.SUB Utility program used with SUBMIT to install the operating system on the hard disk.

DDANADXM.PRL* GSX graphics driver to support Anadex DP-9501 and DP-9001A printers.

DDCNTXM.PRL* GSX graphics driver to support Centronics 351, 352, and 353 printers.

DDCITOLR.PRL* GSX graphics driver to support the C.ITOH 8501A Low Resolution Printer.

DDDS180.PRL* GSX graphics driver to support the datasouth DS180 printer.

DDLA50.PRL* GSX graphics driver to support the DEC LA50 printer.

DDLA100.PRL* GSX graphics driver to support the DEX LA100 printer.

DDFXLR8.PRL* GSX graphics driver to support the Epson Low Resolution (8-bit) printer.

DDFXHR8.PRL* GSX graphics driver to support the Epson High Resolution (8-bit) printer.

DDOKI84.PRL* GSX graphics driver to support the OKIDATA Microline printer.

DDHP7470.PRL* GSX graphics driver to support the Hewlett-Packard 7470 plotter.

* Digital Research, Inc. (DRI) files

CLEAR

Clear all to spaces	ESC + or ^Z
Clear all to nulls	ESC :
Clear all to half-intensity spaces	ESC ,
Clear all to nulls and reset half-intensity	ESC *

FUNCTION KEYS

Program function keys	SHIFT/LOC ESC | p1 p2 message ^Y

(p1=number of function key
p2=1
^Y=termination character)

Table M-2
Switch Setting List

Section	Setting	Function
1	closed (right)	Baud rate
2	closed (right)	Baud rate
3	closed (right)	Baud rate
4*	closed (right)	
5*	closed (right)	
6*	closed (right)	
7*	closed (right)	
8	Not used	
9	open (left)	Green on black screen (default)
10*	Not used	

* required settings

Table M-3
Baud Rate List

Switch Section			Baud Rate
1	2	3	
C	C	C	9,600
O	C	C	4,800
C	O	C	2,400
O	O	C	1,200
C	C	O	600
O	C	O	300
C	O	O	150
O	O	O	75

C=closed
O=open

APPPENDIX N
TS 803H DEMONSTRATION PROGRAM

The Digital Research GSX (Graphics System eXtension) gives programs the ability to easily display images on a number of different graphics devices. It provides a standard interface so that different graphics devices look the same to applications programs.

We have included a demonstration program to show some of the capabilities of the GSX on the TS 803H. To run the program, you need the following files on your logged disk:

803hdemo.com	Demonstration program itself
assign.sys	Explained below
demotxt2.bin	Data for text fonts
surfl	Data for a curved surface
ddts803.prl	Software to talk to the TS 803H

NOTE! Plotters must be connected to the modem port and printers must be connected to the printer port.

If you intend to send the TS 803H demonstration program to a printer or plotter, you must make a backup copy of the system diskette, erase unneeded printer driver files from your system diskette, and then run the demo program to send the output to a printer or plotter. If you don't do this and you attempt to send the demonstration program to a printer or plotter, the following error message appears:

```
Disk Write Error, Temporary File : GSX.TMP
A>
```

It indicates there's not enough space on the system diskette.

To run the program, enter

803Hdemo<CR>

The program asks you if you want the demonstration displayed on the CRT, a printer, or a plotter. To see the demonstration on the TS 803H screen, follow these instructions.

USER: Enter

1<CR>

SYSTEM: Displays

reading in data

After about 30 seconds, the program starts displaying a series of different screens giving information about the TS 803H. The program runs unattended, and continues to run until you press the D key. The program then finishes drawing the current page and returns to CP/M.

If you want to look at any single page, press any key other than D while the page is being drawn. That page stays on the screen until you press any other key.

Hardcopy Options

For the following printers, you can see the screens from the demonstration on the printer by choosing option number 21 when the demonstration program starts running. The Epson must be plugged into the TS 803H printer port, located on the back panel of the main unit, and the baud rates on the printer and computer should be set to 1200 baud. See the section in Chapter 2 on baud rates.

Epson Low Resolution (8-bit) printer
Epson High Resolution (8-bit) printer

For the following plotter, you must plug it into the modem port of the TS 803H, and select number 11 when the program starts. If you do not get output on the plotter, you can use the Setup Program to check the baud rates and data formats and make sure that the RDR: and PUN: logical devices are set up to be the UR1: and PTP: physical devices respectively.

Hewlett-Packard 7470 plotter

How the Assign.sys File Works

If you have a printer or plotter and wish to see output on it, you must make sure that software support is provided for that device. See Tables 1 and 2 for the printers and plotter that are supported.

Table 1
Supported Printers

Printer Name	File for Device
Anadex DP-9001A AND DP-9501	ddanadxm.PRL
Centronics 351, 352, and 353	ddcntxm.PRL
C.ITOH 8501A Low Resolution	ddcitolr.PRL
datasouth DS180	ddds180.PRL
DEC LA50	ddla50.PRL
DEC LA100	ddla100.PRL
Epson Low Resolution (8-bit)	ddfxlr8.PRL
Epson High Resolution (8-bit)	ddfxhr8.PRL
OKIDATA Microline	ddoki84.PRL

Table 2
Supported Plotter

Plotter Name	File for Device
Hewlett-Packard 7470*	ddhp7470.PRL

* The Hewlett-Packard 7470 plotter and the Epson FX80 printer using the Epson Low Resolution driver have been tested by TeleVideo to be fully compatible to the TS 803H. The other drivers have not been tested by TeleVideo Systems.

The file named "assign.sys" determines which device will get output. The file comes set up to talk to the TS 803H screen or the Hewlett-Packard 7470 plotter or Epson Low Resolution printers, and should look like:

```
21 @ddfxlr8
01 @ddts803
11 @ddhp7470
```

The "dd" portion of the filename stands for device driver. The "@" (at-sign) stands for the default drive. For example, if you are running from drive A, CP/M takes the "@" to mean "A". The file must be in the format of two digits specifying the device number, followed by a space, the disk letter, and file name. GSX assumes a ".prl" extension, so you do not need to include the extension in the specification of the filename.

Note that the numbers at the beginning of each line correspond to the numbers asked for by the demonstration program. When you specify a number to the demonstration program, it tells the GSX to look for that number in the assign.sys file. The GSX uses the filename associated with that number to send graphics commands to the device associated with the file.

The GSX assumes that device 1 is the default console, so the TS 803H should always be set to 1. Devices 1-10 are CRTs, 11-20 are plotters, and 21-30 are printers. The assign.sys file that comes with the system uses devices 1, 11, and 21.

GSX assumes that the printer has been set up as the LST: device and the plotter as the PUN: device. You can use the Setup Program to make sure that the TS 803H is set up correctly.

Changing the Assign.sys File

If you have a printer other than the Epson Low Resolution (8-bit) printer or a plotter other than the H-P 7470, you must edit the assign.sys file using a text editor. For example, the Centronics 351 printer uses the ddcntxm.prl file. You should change the first line in the file to read:

```
21 @:ddcntxm
```

and save the new version of the file. The largest driver should be the first line in the assign.sys file, so use the stat program to check the driver sizes if you change the assign.sys file. You may plug the printer into the printer port or modem port on the back of the TS 803H, and check the baud rates and data formats. See the SETUP utility program for information on checking baud rates, data formats, and default output devices.

APPENDIX O
PARK UTILITY PROGRAM

The **PARK** utility positions the R/W heads of the TS 803H hard disk over the safety zone (usually the innermost cylinder). Thus, data is protected from destruction or erasure.

There are two distinct configurations of the TS 803H hard disk: the Miniscribe hard disk and Rodime hard disk configurations. Since the location of the safety zone is different, PARK detects the location by reading a DIP switch setting. The corresponding DIP switch setting is as follows:

DIP Switch Setting for Section 6:

closed (right)	Miniscribe hard disk configuration (2 R/W heads)
open (left)	Rodime hard disk configuration (4 R/W heads)

STOP! Arbitrarily changing the setting of section 6 may cause deterioration of the hardware stepping function or destroy data on the disk. WE STRONGLY SUGGEST THAT YOU DO NOT CHANGE THE FACTORY SETTING OF THIS SWITCH.

USER: 1. Insert the system diskette into the floppy drive and turn on the TS 803H.

2. If the system prompt is not displayed, reset the TS 803H by pressing the ^RESET button.

SYSTEM: 3. Displays

```
TeleVideo System TS-803H Vx.x
(c)1983 TeleVideo Systems, Inc.
xxk CP/M version y.y

A>
```

USER: 4. Enter

park<cr>

SYSTEM: 5. Displays

```
PARK Program for TS-803H  V1.1
Copyright 1984 TeleVideo Systems, Inc.

Do you want to park R/W heads?(y/n)
```

USER: 6. Enter

y

SYSTEM: 7. Displays

```
Parking complete.
REMOVE DISKETTE from drive and TURN OFF THE POWER.
```

USER: 8. Turn off the power switch on the rear of the TS 803H and disconnect the power cord. You may now safely move the TS 803H.