SOLOS™

CUTER™

USER'S MANUAL

Processor Technology Corp.
6200 Hollis St.
Emeryville, CA 94608
(415) 652-8080

Software Technology Corporation
P.O. Box 5260
San Mateo, CA 94402
(415) 349-8080

M0100

SOLOS(tm)/CUTER(tm)

USER'S MANUAL

PROCESSOR TECHNOLOGY CORP.
6200 Hollis Street
Emeryville, CA 94608

(415) 652-8080

SOFTWARE TECHNOLOGY CORP.
P. O. Box 5260
San Mateo, CA 94402

(415) 349-8080

IMPORTANT NOTICE

SOFTWARE WARRANTY

Software Technology Corporation warrants this Software Product to be free from defects in material and workmanship for a period of three months from the date of original purchase.

This warranty is made in lieu of any other warranty expressed or implied and is limited to repair or replacement, at the option of Software Technology Corporation, transportation and handling charges excluded.

To obtain service under the terms of this warranty, the defective part must be returned, along with a copy of the original bill of sale, to Software Technology Corporation within the warranty period.

The warranty herein extends only to the original purchaser and is not assignable or transferable and shall not apply to any software product which has been repaired by anyone other than Software Technology Corporation or which may have been subject to alterations, misuse, negligence, or accident, or any unit which may have had the name altered, defaced or removed.

P R E F A C E

This manual describes the use and operation of either SOLOS(tm) or CUTER(tm). SOLOS is a program designed to be a personality module in a Sol(tm). CUTER is a program designed to provide much of the power of SOLOS for the non-Sol user. Because SOLOS and CUTER have been designed to be compatible operating systems, this manual will refer to SOLOS meaning the SOLOS/CUTER operating system. The few differences between SOLOS and CUTER will be stated explicitly.

(tm) SOLOS, CUTER and Sol are trademarks of Processor Technology Corporation.

TABLE OF CONTENTS

TABLE OF CONTENTS (cont.)

I. INTRODUCTION

SOLOS is a 2048 byte program that configures the Sol-20 and one or two cassette tape recorders into a powerful, stand-alone computing system. SOLOS takes advantage of the Sol-20's built-in hardware peripherals and the 8080 instruction set to optimize the convenience and power of the inherent computer capabilities of the Sol.

Outstanding features of SOLOS include...

- **STANDARDIZED I/O SOFTWARE PROTOCOL** which makes all Sol-20 I/O (keyboard, display, serial, parallel and cassette) accessible to external programs from one entry point--a standard feature in all future Sol system software products that will require less memory than would normally be used for I/O routines.
- **SOFTWARE INTERFACE** permits user defined routines for custom applications.
- **"INDUSTRY STANDARD-SETTING" CASSETTE I/O CONTROL** includes methods for loading and saving programs and commands that execute programs after automatic loading.
- **EXCLUSIVE CASSETTE I/O ROUTINES** allow cassette files to be accessed on a byte-by-byte basis as though each file were a byte-by-byte device. Thus, data transfer to and from cassettes appears as normal I/O--and two cassettes can be used simultaneously to assemble and edit programs.
- **NEW DISPLAY CONTROL** features found only in expensive video terminals--including ESCAPE sequences for cursor positioning and character speed control.
- **19 COMMANDS** to access the basic requirements of the Sol-20 control cassette tape recorders and set up special conditions in SOLOS. (See the "Quick Command Reference List".)

Definition of Terms

In this manual:

addr means word address hexadecimal characters, (Ø-FFFF) range

data means hexadecimal characters, (Ø-FF) range

file means a collection of data

name means any one to five character identification for a file

port means a SOLOS pseudoport from 0 to 3

unit means a number of 1 or 2 corresponding to the appropriate tape recorder

( ) means optional parameters

Only the first two letters of the command expressions must be typed when entering a command expression. (The underscored letters in the following Quick Command Reference List.)

Quick Command Reference List

COMMAND	FUNCTION
Console	
EXEC addr	Begin program execution at 'addr'
ENTR addr	Enter data into memory starting at 'add
DUMP addr1 (addr2)	Dump memory data, 'addr1' to 'addr2'
TERM (portin (portout))	Enter Terminal Mode
CUST name (addr)	Insert or remove a custom command
Tape	
GET (name(/unit) (addr))	Get a tape file into memory
SAVE name (/unit) addr1 addr2 (addr3)	Save a file from memory to tape
XEQ (name(/unit) (addr))	Get then execute a tape file
CAT (/unit)	Catalog tape files
Set	
SET S=data	Screen character rate
SET I=port	Input port to SOLOS
SET O=port	Output port to SOLOS
SET N=data	Number of NULLS following CRLF
SET XEQ addr	Auto-execute addr
SET TAPE Ø or 1	Ø=12ØØ baud, 1=3ØØ baud
SET TYPE data	Type 'byte' header
SET COUT addr	Custom output addr
SET CIN addr	Custom input addr
SET CRC data	Allows ignoring of tape CRC Read Errors

With a Sol, or CUTER on a Processor Technology GPM board, a power-on performs a reset which causes a SOLOS system reset. The Sol user may initiate this system reset anytime by simultaneously pressing the upper case and repeat keys.

A SOLOS system reset enters SOLOS into COMMAND mode. When in COMMAND mode, SOLOS will do a Carriage Return-Line Feed (CRLF) followed by a prompt (>). SOLOS then awaits the entry of a COMMAND. A COMMAND is processed upon receipt of a Carriage Return (CR). Pressing the MODE (or Control-@) key while awaiting a COMMAND causes the current COMMAND input line to be ignored and return to COMMAND mode. CUTER also resets the current I/O pseudo port selections to the system default.

The MODE (or Control-@) key is also used to abort the execution of most commands. This use of the MODE (or Control-@) key turns off both tape machines (if on) and returns to COMMAND mode.

II. CONSOLE COMMANDS

Console Commands in Brief

SOLOS has five console commands. They are:

Command	Function
EXEC addr	Begin program execution at 'addr'.
ENTR addr	Enter data into memory starting at 'addr'.
DUMP addr1 (addr2)	Dump memory data, 'addr1' to 1addr2'.
TERM (portin (portout))	Enter Terminal Mode (available under SOLOS only)
CUST name (addr)	Insert or remove a custom command.

Console Commands in Detail

Execute Command EXEC addr

This command begins program execution at memory location specified by (addr).

Example: EXEC 2ØØ

Enter Command ENTR addr

Example: ENTR 5ØØ

: C3 ØØ Ø1 1ØØØ: Ø5/

Result: Beginning at memory location 5ØØ, the following data was entered: C3 ØØ Ø1. The new memory location of 1ØØØ: was selected to enter the data 51. The slash (/) terminated the ENTR command and returned to command mode.

Dump Command DUMP addr1 (addr2)

This command displays sequential memory data on the screen starting at location (addr1) and ending with (addr2).

Example: DUMP CØ2E CØ37

Result: CØ2E E1 DB FA 2F E6 Ø1 C8 DB FC C9

Dumped the SOLOS keyboard input routine. (See listing.) Starting at memory location CØ2E and ending at memory location CØ37.

Terminal Command TERM (port-I (port-O)) (Available under SOLOS onl

This command causes the Sol system to become a video terminal for connection to an external computer or modem. This command begins by automatically setting the I/O pseudo ports to the specified values. An omitted port parameter will be set to 1. Execution then proceeds by sending all Sol keyboard entries (except cursor control) to the specified Output pseudo port. Any input available from the Input pseudo port will be processed by the SOLOS display driver.

Example: TERM

Result: Keyboard data will be sent to the serial port and all data from the serial port will appear on the display screen.

Custom Command CUST name (addr) definition/removal

When a non-SOLOS command is entered, a separate table of custom commands (in RAM) will be searched. The CUST command is used to enter and remove up to six custom command names from the custom command table. (Only the first two letters of the name are significant.) When the name (2 to 5 letters) specified by the CUST command is not already in the custom command table, a new custom command will be entered into the table having an execute address as specified. When the addr is not specified, the beginning address of SOLOS will be used.

When the name specified on the CUST command already exists in the custom command table, this table entry will be replaced with an 'end-of-table' indicator. Therefore, not only will the specified name be removed, but any other custom command names following in the table will also be removed.

Example: CUST BASIC Ø
CUST ALS8 EØ6Ø

Result: Two new custom commands are now known.
ALS8 at location EØ6Ø, and
BASIC at location Ø.

III. TAPE COMMANDS

Tape commands are used to control the tape cassette recorders. In these commands, unit selection is optional, with a default selecting unit 1. When a unit is specified, however, it must be separated from the file identification name with a slash (/) and without spaces in between: e.g., TARGT/2.

Tape Header

At the start of each tape file is header information. This information includes the following data:

- name: name of file, 5 ASCII characters or less
- type: number is specified by user at time file is created
- addr: starting address of file
- size: number of data bytes in file
- XEQ addr: auto-execute address word (See Set Commands - Section IV)

Error Messages

Cassette error messages are printed in this format:

"ERROR (name) (type) (addr) (size)"

Reasons for an error message are:

1. bad read of file (tape error or CRC ERROR)
2. MODE (or Control-@) key used for escaping while reading a tape file
3. XEQ command given to a non-executable file.

Tape Commands in Brief

SOLOS has four tape commands. They are:

GET (name (/unit) (addr))	Get a file from tape to memory
SAVE name (/unit) addr1 addr2 (addr3)	Save file
XEQ (name (/unit) (addr))	Get, then execute, a file
CAT (/unit)	Catalog of tape files

Tape Commands in Detail

Get a file from tape — GET (name(/unit) (addr))

This command transfers the specified or next tape file into memory. If a (name/unit) is given, this command will search forward on the cassette until that file is found. The (addr) parameter, if given, specifies the memory location at which the file will be loaded. If the addr is omitted, the file will be loaded as specified in the header.

Example: GET TARGT/2

Result: Gets the program "TARGT" from tape unit #2 into memory as specified by the tape file header information. Returns to SOLOS command mode.

Get, then Execute — XEQ (name(/unit) (addr))

This command is an extension of the GET command which gets a tape file and executes as specified by the header information. The (/unit) and (addr) are optional and operate the same as with the GET command.

Example: XEQ FOCAL

Result: Gets, then executes, a program named "FOCAL" from tape unit 1.

Save a file — SAVE name (/unit) addr1 addr2 (addr3)

This command transfers program or data onto a tape cassette file name (name) starting at (addr1) and ending at (addr2). The name of the file becomes part of the tape's header information. SET TYPE and SET XEQ commands affect the header information on the tape file. The optional addr3 specifies the address (if different than addr1) to be entered in the tape header.

Example: SAVE CHASE/2 Ø 1FF

Result: Saves onto tape unit 2 a program named "CHASE" starting at location ØØØØ and ending at location 1FF.

Catalog of files — CAT (/unit)

This command will start the tape unit specified and list each tape file header information.

Example: CAT /2

Result:
```
SLOPE Ø5ØØ Ø2ØØ
HUM   Ø5ØØ ØBØØ
```

III. TAPE COMMANDS (cont.)

Note: A very useful feature of the CAT command is to apply power to the tape units when needed to rewind tape. Depressing the MODE (or Control-@) key will remove power from tape unit and return to COMMAND mode.

IV. SET COMMANDS

SOLOS has 10 set commands. They are:

SET S=data	Screen character rate
SET I=port	Input port to SOLOS
SET O=port	Output port to SOLOS
SET N=data	Number of NULLS following CRLF
SET XEQ addr	Auto-execute addr
SET TAPE Ø or 1	Ø=12ØØ baud, 1=3ØØ baud
SET TYPE data	Type 'byte' header
SET COUT addr	Custom output addr
SET CIN addr	Custom input addr
SET CRC data	Allows ignoring of tape CRC Read errors

Set Commands In Detail

Set Speed of Display — SET S=Ø-FF

This command determines character display rate to the screen:

data = Ø - Fastest

data = FF - Slowest

Input/Output Command Parameters

The next two SET commands affect SOLOS input and output command parameters.

Set Out Command — SET O=port

This command selects the output driver routine to which SOLOS routes data. Under SOLOS, COMMAND mode text is always sent to the display screen. Under CUTER, all output goes to the current Output pseudo port. In all cases, the output from each command is sent to the current output pseudo port.

IV. SET COMMANDS (cont.)

The Output Pseudo ports command parameter values are:

Ø = Video Display

1 = Serial Output Port

2 = Parallel Output Port

3 = User Defined by SET COUT command

Example: SET O=1
DUMP Ø 2F

Result: Select serial output port. 'Dump Ø 2F' would be displayed, but the data would go to the serial output port.

Set In Command SET I=port

This command selects the input driver routine to SOLOS. All future input commands would come from the new selected input pseudo port.

The Input Pseudo port parameter values are:

Ø = Keyboard

1 = Serial Input Port

2 = Parallel Input Port

3 = User defined by SET CIN command

Example: SET I=1

Result: SOLOS would expect the next command to come from the serial port input routine. The Sol keyboard would have no affect except to simultaneously hit repeat and upper case keys to reset the computer.

Cassette Tape Parameter Commands

The Following SET commands affect the cassette tape parameters:

Set Tape Command SET TAPE Ø or 1

This command selects one of two standard speeds.

Ø = 12ØØ baud high speed

1 = 3ØØ baud low speed

Normally set to Ø.

IV. SET COMMANDS (cont.)

Set Type Command — SET TYPE data

This command sets (data) values into the 'type' byte in the tape header information when used in conjunction with the SAVE command. The 'type' byte data is entered as a hexadecimal value, but it will appear on the screen as an ASCII character when displayed by the GET or CAT command. Only displayable characters should be used for type values (data). The most significant bit of the type value determines if the tape file can be executed automatically by an XEQ command. (Ø = Auto-execute, 1 = Not executable.) Typing of tape files can be very useful in grouping common files.

Example: SET TYPE 47

47 = 'G' character for GAME FILES
Sign Bit = Ø, auto-execute

SET TYPE 5Ø

5Ø = 'P' character for PROGRAM FILES
Sign Bit = Ø, auto-execute

SET TYPE C4

C4 = 'D' character for DATA FILES
Sign Bit = 1, non-execute

Set Execute Command — SET XEQ addr

This command sets the auto-execute address (addr) word into the tape header information when used in conjunction with the SAVE command. This address word is used by the XEQ command after loading a tape file to begin program execution at location specified by tape header information (addr). Note that the 'TYPE' byte determines if the file is of the auto-execute type.

Example: SET XEQ 2ØØ

Result: The auto-execute address of 2ØØ Hex will be written onto the tape header when the next SAVE command is issued.

Custom Input/Output Commands

The next SET commands set address pointers to custom input and output driver routines when 'SET I=3' and/or 'SET O=3' are used. These custom I/O drivers must meet the SOLOS I/O drivers requirements. See the SOLOS software listing for model input routine.

Set Custom Output Command — SET COUT addr

This command informs SOLOS software where the user defined output routine specified by 'addr' is located.

IV. SET COMMANDS (cont.)

The Custom Output driver requirements are:

1. The 'addr' (address) word in the SET COUT command will equal the starting address of the output routine.

2. It is the user's responsibility to save registers prior to any modification of the register.

3. The "B" register will contain the data passed from SOLOS for output routine.

4. The output routine will end with a 'RET' instruction or equivalent.

Set Custom Input Command SET CIN addr

This command informs SOLOS software where the user defined input routine specified by 'addr' is located.

The Custom Input driver requirements are:

1. The 'addr' address word in the SET CIN command will equal the starting address of the input routine.

2. It is the user's responsibility to save registers prior to any modification of the register.

3. The input routine combines actually inputting the character along with STATUS. The routine returns either a zero flag indicating no character is available or the character in Register "A" with a non-zero flag. The calling program can then take appropriate action based on a zero or non-zero condition.

Set CRC Error Checking SET CRC data

This command is used to specify whether or not the standard CRC error checking routines are to be used. When a value of FF is specified, all further tape reads will ignore CRC errors. Any value other than FF indicates standard error checking is to be in effect. This command is very useful to allow a tape to be read in which would otherwise not be readable. When CRC errors are being ignored, it must be remembered that the data read in may not be valid.

Example: SET CRC FF

Result: CRC error checking will be set to ignore all CRC errors.

Set Number of NULLS SET N=data

This command sets the number of nulls (binary zeroes) to be output following a carriage return-linefeed (CRLF) sequence. The value is

initialized to zero but may be set to any number up to FF (hex). This command is useful when using output devices requiring a delay following a carriage return.

Example: SET N=3

Result: Every CRLF issued by SOLOS will be followed by three nulls.

V. SUBROUTINES

A. Introduction to the SOLOS Machine Language Interface

The Machine Language Interface with SOLOS is based on:

1. A predefined set of 'pseudo' I/O ports allowing software compatibility as well as providing an easy means of supporting any I/O device.

2. A system defined register usage when interfacing with SOLOS.

3. A system jump table of entry points.

First are the pseudo ports. Built into SOLOS are four input and four output pseudo ports. I/O requests made to a pseudo port are converted internally to a request either to a specific device, a built-in routine, or a user written routine. All non-tape I/O requests made to SOLOS are made with reference to one of the following pseudo ports.

PSEUDO PORTS FOR SOLOS

Pseudo Port	Input	Output
0	Keyboard	VDM driver
1	Serial port	Serial port
2	Parallel Port	Parallel Port
3	User written routine	User written routine

PSEUDO PORTS FOR CUTER

Pseudo Port	Input	Output
0	Keyboard data from parallel port 3,not KDR status, on port 0, bit 0.	VDM driver
1	Serial port 1, RDA status on port 0, bit 6.	Serial port 1, TBE status on port 0, bit 7.
2	Parallel port 2 with not-PDR status on port 0, bit 4. 1	Parallel port 2 with not-PXDR status on port 0, bit 2.
3	User written routine.	User written routine.

V. SUBROUTINES (cont.)

Second are the defined register usages when interfacing at the machine language level with SOLOS.

Whenever a machine program is executed by SOLOS (via the EXEC or XEQ command, or via a custom command), the stack pointer and HL registers are predefined by SOLOS >. The stack pointer is set such that the user may perform stacking operations which will use the SOLOS stack. The SOLOS stack begins at the end of the SOLOS RAM area and works its way down from there. Excessive use of this stack can destroy data maintained by SOLOS within its RAM area. The stack is also prepared so that the user may issue a standard RET instruction to return control to SOLOS command mode processor.

*

The HL register pair is initialized to point to the very beginning of SOLOS. It is at this point that the SOLOS jump table begins. The user program may then use the address presented in the HL register pair as the beginning of the jump table.

This address is provided for two reasons:

1. CUTER may be located at any address in memory, providing the means for programs to function with CUTER located at any address, and

2. the first byte of the jump table for SOLOS is different from the first byte for CUTER, providing an easy means of distinguishing between SOLOS and CUTER.

Third is the SOLOS jump table (see next page). All requests to SOLOS should be made based on this jump table and not to the actual routine addresses as scattered throughout SOLOS. By using only this jump table the user can be assured of maintaining compatibility between SOLOS and CUTER. **

Page 15: Paragraph two describing the Stack Pointer and the Stack. Add the following:

* The Stack itself should be established such that:

(a) A "Ret" instruction can be used as an exit by the executing program.

(b) The locations at Stack Pointer -1 and -2 in memory contain the address of the executed program itself. This information can be accessed by machine code similar to:

```
LXI   H,-1    A constant minus one.
DAD   SP      HL=SP-1 now.
MOV   A,M     A=our own high address.
```

Code such as this can be used to allow a routine to be made self-relocating to a 256 byte boundary.

Page 15: Paragraph three describing the HL register pair. Add the following:

** The HL register pair is initialized to point to the very beginning of SOLOS. Because the jump table may appear at any 256 byte boundary, register L will be zero.

V. SUBROUTINES (cont.)

JUMP TABLE

Address	Label	Length	Function
C000	START	1	This byte allows power-on reset of SOLOS. It is 00 for SOLOS and 7F for CUTER, providing an easy means of differentiating the exact operating system in use.
C001	INIT	3	This is a "JMP" to the power-on reset.
C004	RETRN	3	Enter at this point to return control to SOLOS command mode processor.
C007	FOPEN	3	Enter here to open a tape file.
C00A	FCLOS	3	Enter here to close a tape file.
C00D	RDBYT	3	Enter here to read a byte from an open tape file.
C010	WRBYT	3	Enter here to write a byte to an open tape file.
C013	RDBLK	3	Enter here to read one tape block into memory based on a header.
C016	WRBLK	3	Enter here to write one tape block from memory based on a header.
C019	SOUT	3	Enter here to output the character in register "B" to the current system output pseudo port. This is always an "LDA" pointing to the byte containing the current system output pseudo port value.
C01C	AOUT	3	Enter here to output the character in register "B" to the pseudo port specified in register "A".
C01F	SINP	3	Enter here to obtain status/character from the current system input pseudo port into register "A". This is always an "LDA" to the byte containing the current system input pseudo port value.
C022	AINP	3	Enter here to obtain status/character from the input pseudo port specified in the "A" register. On return, register "A" will contain the character with the flags set to indicate whether a character is present or not.

V. SUBROUTINES (cont.)

B. System Entry Points

There are actually only two system entry points within the SOLOS jump table. Entry at these points does not require that any register be initialized. The first (at either label "START" or "INIT") is used to perform a complete power-on system reset. As a part of the system reset, the system RAM area data used by SOLOS will be cleared. The only reason for entering via "START" or "INIT" is that the power-on circuitry requires a one byte instruction to allow various circuits to stabilize. The other use of the byte labeled "START" is to determine if a user program is being executed under SOLOS or is CUTER controlled. When under SOLOS, this byte will be zero. When under CUTER, this byte will be non-zero.

The other system entry point ("RETRN") is used to return to SOLOS command mode. This entry point does <u>not</u> perform a system reset.

C. SOLOS Input Entry Points

SINP entry point address C01F

This entry point will set register "A" to the current system input pseudo port. The current system input pseudo port is changed by tl "SET I=" command. After setting register "A", this command procee by executing an "AINP". (See below.)

AINP entry point address C022

This entry point is used to input one character or status from any pseudo port. Register "A" on entry indicates the desired input pseudo port from 0 to 3. Because this entry point is a combination status/get-character routine, it is the user's responsibility to interpret return flags properly. When a character is <u>not</u> avail able, the zero flag will be set and the character will be placed into register "A".① What this means is that, if the user wants to wait for a character to be entered, simply follow the CALL AINP (or SINP) with a "JZ" jump-if-zero instruction back to the call. A combined status/get-character routine is very important when allowing user written input routines.

D. SOLOS Output Entry Points

SOUT entry point address C019

This entry point will set register "A" to the current system output pseudo port. The current system output pseudo port is changed by using the "SET O=" command. After setting register "A", this command proceeds by executing an "AOUT". (See next definition.)

① When a character <u>is</u> available, the zero flag is <u>reset</u> and the character is in register "A".

V. SUBROUTINES (cont.)

AOUT entry point address C01C

This entry point is used to output one character to any pseudo port. Register "A" is assumed to be a binary value from 0 to 3 indicating the desired output pseudo port. Register "B" will contain the character to be output. On return, the PSW and Register "A" are undefined. All other registers are as they were on entry.

E. SOLOS VDM Display Driver

Because the VDM is much more powerful than a standard hardcopy device, the built-in VDM driver supports many expanded functions. The following characters, when sent to the VDM driver (output pseudo port 0), cause special functions to be performed:

Hex	Character		Function
01	Control-A	(SOH)	Move cursor left (wrap mode) one position.
0B	Control-K	(VT)	Clear screen; position cursor at home.
OD	Control-M	(CR)	Clear remainder of line; then move cursor to beginning of same line.
13	Control-S	(DC3)	Move cursor right (wrap mode) one position.
17	Control-W	(ETB)	Move cursor up (wrap mode) one line.
1A	Control-Z	(SUB)	Move cursor down (wrap mode) one line.

The escape key (hex code 1B) is also a special character to the VDM driver. It initiates what is known as an escape sequence. The escape character is always followed by one or two hexadecimal values (bytes) which indicate what expanded function is to be performed. The following lists the escape sequences and corresponding results. Where a third byte must follow the escape, this will be represented by (##), indicating that this third byte actually contains a value being passed to the VDM driver.

Escape sequence	Function
1B 01 ##	Place the cursor onto position (##) of the current display line. (##) is in the range 00 - 3F.
1B 02 ##	Place the cursor onto line number (##) of the display screen. (##) is in the range 00 - 0F, with the topmost line being line 00.
1B 03	Pass back the current cursor line/character position in Registers BC. Register "B" is set to the character position (00-3F), and Register "C" is set to the line position (00-0F).
1B 04	Pass back the memory address of the current cursor location into Registers "BC".

more escape sequences . . .

V. SUBROUTINES (cont.)

Escape sequence	Function
1B 05 ## 1B 06 ## 1B 07 ##	The third byte is output to the VDM at the current cursor position exactly as is, regardless of this byte's value. No check is made of this character (##). Being a control character, it is only placed into the VDM memory as-is, and the cursor is advanced one position.
1B 08 ##	The display speed is set to the value (##) specified. The speed ranges from 00 (fastest) to FF (slowest).
1B 09 ##	This functions the same as escape sequence 01. The cursor is positioned to character position ## of the current display line.

F. Cassette Tape Entry Points to SOLOS

SOLOS contains subroutines to handle data transfer to and from two cassette units. Both block-by-block and byte-by-byte access are available. While performing any tape read, the user can return to the present calling software program by pressing the MODE (or Control-@) key.

In block transfers, each request results in tape movement and a transfer of an information block to or from a location in memory. SOLOS uses block-by-block access to provide the tape commands.

In byte transfers, on the other hand, SOLOS buffers the data into 256 byte blocks, doing cassette operations only once per 256 transfers. BASIC uses byte-by-byte access for data files. Other programs--such as editors, assemblers or special user-written programs--can also call the byte-by-byte routines if a few specific conventions and calling sequences are followed.

File Header

The file header for SOLOS provides specific attributes to a file. These attributes consist of a five ASCII character name and a file type.

File name serves two functions:

1. It permits easy human identification of the file, and

2. It provides the identification for which SOLOS searches to find the correct file.

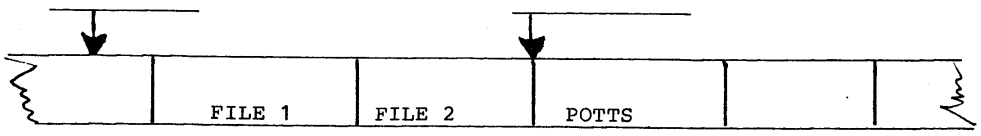
File type is used in SOLOS to prevent certain operations, such as automatic XEQ, if the file is not of the proper type.

V. SUBROUTINES (cont.)

When calling open the register, pair "HL" should point to a memory location that contains the header. Following is the layout of a SOLOS file header:

```
NAME  ASC  '12345'     A five character name with trailing binary zeroes
      DB   0           Should always be zero.
TYPE  DB   'B'+80H     File type. If Bit 7=1, then this is a data file
                       (not executable).
SIZE  DW   LENGTH      Length of file in number of bytes.
ADDR  DW   FROM        Address at which file is to be read to or from
                       which it is to be written.
XEQ   DW   EXEC        Auto execute address (ignored for data files).
      DS   3           Space - not currently used by SOLOS.
```

As previously mentioned, SOLOS uses the name to find the correct data for the file operations. Assume you were about to read data from a file named POTTS, for example, and you had correctly opened the file with a header pointing to that name. SOLOS, when you first requested a data transfer, would read past File 1 and File 2 (as shown below) and then read data from the POTTS file.



Block Access

The Block Access method invokes no management by the system. Each 'call' to the 'Read' or 'Write' routines performs a complete cassett operation. Read and Write routines are used by SOLOS for GET and SAVE commands and serve as examples of the calling conventions for RDBLK and WRBLK routines.

Read Tape Block Routine RDBLK

The entry point for RDBLK is C013.

On entry: Register A contains Unit and Speed data with bit 5 (speed) Ø for 12ØØ baud (or 1 for 3ØØ baud); bit 7=1 for Tape 1; bit 6=1 for Tape 2; and all other bits=Ø.

Registers H & L contain the address of file header information.

Registers D & E contain the address of where the file is to be loaded into memory. (If set to Ø, this information is taken from file header information on tape

On exit: Normal return: Carry Flag is cleared, and data has been transferred into memory.

Error return: On errors, or user pressing MODE (or Control-@) from keyboard, the Carry Flag is set.

Write Tape Block Routine WRBLK

The entry point for WRBLK is CØ16.

On entry: Register A contains unit and speed with the same bit values as specified for RDBLK.

Registers H & L contain file header address. The file header information will be written onto the specified tape unit followed by the data.

On exit: Normal return: Carry Flag is cleared, and data has been transferred to tape.

There are no error returns.

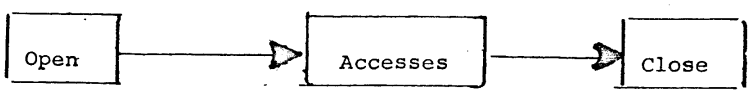
Byte Access

Data stored on, or about to be stored on, a tape should be considered a file. In a SOLOS file, data is stored one byte at a time as a string of bytes along the tape with no assumed meaning or structure. It is simply a collection of bytes that can be accessed by someone with responsibility for the intelligence of the data.

When writing to tape, SOLOS records the data in a form that allows the data to be read from the tape later. When reading from tape, SOLOS provides the management to access each byte sequentially.

SOLOS also provides start and stop control of two units. File operations view Unit 1 as File 1 and Unit 2 as File 2. Thus, data in Unit 1 is associated with File 1, and data in Unit 2 is associated with File 2.

When using Byte Access, two important user management operations are necessary. As shown in Figure below, the first is to open a file to tell SOLOS you want to access the file. The second is to close a file to inform SOLOS you are finished with it.



V. SUBROUTINES (cont.)

SOLOS provides entry points to Open, Read, Write and Close tape files. Each of these routines requires that certain conventions be followed to ensure accurate data transfers.

File Open Routine FOPEN

The Open routine sets up certain internal parameters to keep track of data requests. This operation should be called only once prior to the first access of the file. The File Header information is the same format as in the Block Access mode and is used in both reading and writing of files. If the Byte Accesses are of the Read type, SOLOS will search the tape file until the correct file 'name' is found as specified by the File Header information. On the next Read access, SOLOS will transfer the first data byte of the file. If the Byte Accesses are of the Write type, the File Header information will be transferred onto the file.

The entry point for FOPEN is CØØ7.

On entry: Register A contains File # (1 or 2) same as tape unit (1 or 2).

Registers H & L contain address of the File Header information.

On exit: Normal return: All registers are altered and file is ready for accesses.

Error return: The Carry Flag is set. Reason for error: file already open.

Write Byte Routine WRBYT

The Write Byte routine writes a single byte of data into a buffer file. SOLOS stores this data until it contains 256 bytes. It then writes this block onto the tape, followed by a CRC character (error checking character). SOLOS then resets the buffer file for the next 256 bytes of data.

The entry point for WRBYT is CØ1Ø.

On entry: Register A contains File # (1 or 2).

Register B contains the byte of data to be transferred onto tape.

On exit: Normal return: Carry Flag cleared.
Error return: Carry Flag set - errors caused by:

1. file NOT open, or
2. file previously used for reading.

V. SUBROUTINES (cont.)

Read Byte Routine RDBYT

The Read Byte routine reads a single byte of data from a buffer file. SOLOS fills this buffer as needed per read request. Each time SOLOS fills the file buffer (reads a block), the CRC character is checked for data accuracy.

The entry point for RDBYT is CØØD.

On entry: Register contains file # (1 or 2)

On exit: Normal return: Register A contains data byte. Carry and Minus Flags set mean 'end of file'.

Error return: Carry Flag set. Errors are caused by:

1. file NOT open
2. file previously used for writing
3. CRC character error
4. pressing MODE (or Control-@) while actually reading from the tape.

Close File Routine FCLOS

The Close file routine oloses the current file and resets the internal parameters for the next open operation. It is very important to close the file after all data transfers are completed. Failure to do so could result in lost data and prevent further open operations.

The entry point for FCLOS is CØØA.

On entry: Register A contains File # (1 or 2) to be closed.

On exit: Normal return: Carry Flag cleared.

Error return: Carry Flag set. (Error is caused by file NOT open.)

VI. LOADING & EXECUTING CUTER (Applicable to CUTER only)

CUTER is available (1) on cassette tape with its own loader which can be loaded at any memory address from 0200 through F400, or (2) in ROM at the address C000. In order to load CUTER from cassette tape, perform the following steps. When CUTER is being used in ROM, the procedure is much simpler: make sure the sense switches are set according to H below prior to executing location C000.

A. Verify that the hardware is connected and functioning properly.

B. Enter the following bootstrap routine into memory beginning at location 0. The following is presented in a format similar to that produced by a "DUMP" command with an address shown every 10 (hex) bytes:

```
0000: 21 40 00 F9  45 4D 3E 80  D3 FA E7 05  C2 0A 00 E7
0010: 3D C2 0F 00  E7 02 03 FE  DD C2 14 00  E9 00 00 00
0020: DB FA A5 CA  20 00 DB FB  C9
```

C. Verify that the above bootstrap is in memory exactly as presented.

D. Set the sense switches to the address at which CUTER is to be loaded. The sense switches will be the hi-order byte of the memory address, with the lo-order byte zero. As an example: Sense switches set to 34 hex will cause CUTER to be loaded into memory beginning at location 3400 hex. For convenience, a memory address should be selected that also specifies the default I/O pseudo ports (see "H" below). The address specified must be between 0200 and F400. Remember, however, that CUTER occupies 2K of memory and uses 1K of RAM beyond that.

E. Make sure that the CUTER tape is rewound and placed into the proper cassette machine. The CUTER bootstrap will activate the motor control for tape unit one. If your cassette machine motor control is attached as tape unit one, you may now place the machine into "PLAY" mode.

F. Execute location zero (the bootstrap). This can be accomplished by allowing a "Reset" to specify an address of zero. At this time, be certain that the cassette machine is in "PLAY" mode and is activated.

G. When completed, the CUTER loader program will "HALT". This is not an error condition. When completed, the motor control will also be turned off.

VI. LOADING & EXECUTING CUTER (cont.) (Applicable to CUTER only)

H. Via sense switches, select the default I/O pseudo ports as follows:

```
        X X X X   I I O O
Bit     7 6 5 4   3 2 1 0
```

Where: X X X X doesn't matter

I I which pseudo port from 0 - 3 (00-11 binary) is to be the default input pseudo port.

O O which pseudo port from 0 - 3 (00-11 binary) is to be the default Output pseudo port.

NOTE: Whenever CUTER does a full system reset (begins execution at its beginning memory address), the sense switches will be accessed to determine the default I/O pseudo ports.

I. If either Input or Output default is to be pseudo port 3 (user written routine), verify the following:

(i) The appropriate user written routine is in memory.

(ii) The address of the appropriate I/O routine is entered into the CUTER system RAM area. The system RAM area begins exactly 2K (800 hex) after the beginning of CUTER. The first word of this area is used to contain the address for the user Input routine. The second word will contain the address of the user Output routine. Addresses are entered in lo-hi order.

J. Execute location ZERO. The CUTER loader will have properly prepared this location to either transfer control to the CUTER just loaded or to indicate an error while loading CUTER. If there was no error, CUTER will now be in control.

Remember to turn off the cassette machine and remove the CUTER tape.

K. IF your computer halts again, this means one of the following errors has occurred. Display memory location ONE to determine the error code. The error code will be one of the following:

Error Code in Hex	Meaning
00	The specified load address was not within the range 0200-F400, or the tape file loaded was not CUTER.
01	A tape read error was detected.
02	There was no tape read error, but the CRC (error checking) character was invalid.
40	The file was loaded, but it was not CUTER.

PROCESSOR TECHNOLOGY CORP.
6200 HOLLIS STREET
EMERYVILLE, CALIF. 94608

MANUAL DATA
CUTS READ AND WRITE ROUTINES

```
0001 *
0002 *
0003 *
0004 *          TAPE READ AND WRITE ROUTINES FOR CUTS BOARDS
0005 *
0006 *
0007 *   THESE ROUTINES WERE EXTRACTED FROM "SOLOS"™ AND "CUTER"™
0008 * TO ILLUSTRATE THE REQUIREMENTS OF THE CUTS BOARD FOR
0009 * READING AND WRITING TO THE CASSETTE TAPE.  BOTH SOLOS AND
0010 * CUTER ALSO HAVE FILE BUFFERING ROUTINES TO PROVIDE BYTE
0011 * BY BYTE TRANSFERS TO THE CASSETTE TAPE.
0012 *
0013 *   CUTER resides on cassette tape and is available at your
0014 * local Processor Technology Dealer for $11.00 or ROM
0015 * resident for use on the GPM module.  The CUTER program
0016 * requires 2K of memory plus 1k of RAM work area.  A short
0017 * bootstrap program is used to load CUTER from cassette
0018 * tape.
0019 *
0020 *   CUTER is relocatable to any 256 byte boundary and has a
0021 * built in command processor and support for serial,
0022 * parallel, keyboard and VDM I/O as well as CUTS cassette
0023 * routines.  The CUTER software is necessary for all major
0024 * Processor Technology or Software Technology programs.
0025 *
0026 *        CUTER COMMAND LIST
0027 *
0028 *  DUMP        Dump memory
0029 *  ENTER       Enter to memory
0030 *  EXECUTE     Execute a program
0031 *  TLOAD       Load programs or data from cassette
0032 *  TSAVE       Save programs or data to cassette
0033 *  TXEQ        Load and run program from tape
0034 *  TCAT        List names of files on tape
0035 *  CUST        Enter or delete custom command name
0036 *  SET di      Set display speed
0037 *  SET nu      Set output nulls
0038 *  SET ta      Set tape speed
0039 *  SET in      Set input port  (0-3)
0040 *  SET out     Set output port  (0-3)
0041 *  SET cin     Set custom input driver address
0042 *  SET cout    Set custom output driver address
0043 *  SET xeq     Set auto execute address to header
0044 *  SET type    Set type in header
0045 *
0046 *
0047 *
0048 *
0049 *
0050 *
0051 *                    <<=-   TAPE READ ROUTINES   -=>>
```

PROCESSOR TECHNOLOGY CORP.
6200 HOLLIS STREET
EMERYVILLE, CALIF. 94608

MANUAL DATA
CUTS READ AND WRITE ROUTINES

```
                        0052 *
                        0053 *
                        0054 *      ON ENTRY:     A   - HAS UNIT AND SPEED
                        0055 *                    HL  - POINT TO HEADER BLOCK
                        0056 *                    DE  - HAVE OPTIONAL PUT ADDRESS
                        0057 *
                        0058 *      ON EXIT:      CARRY IS SET IF ERROR OCCURRED
                        0059 *                    DE HAVE SIZE OF BLOCK READ
                        0060 *                    TAPE UNITS ARE OFF
                        0061 *
                        0062 *
0000 D5                 0063 RTAPE    PUSH    D           SAVE OPTIONAL ADDRESS
0001 06 03              0064          MVI     B,3         SHORT DELAY
0003 CD 1D 01           0065          CALL    TON
0006 DB FB              0066          IN      TDATA       CLEAR THE UART FLAGS
                        0067 *    LOOP HERE UNTIL VALID HEADER IS FOUND
0008 E5                 0068 PTAP1    PUSH    H           HEADER ADDRESS
0009 CD 5F 00           0069          CALL    RHEAD       GO READ HEADER
000C E1                 0070          POP     H
000D DA 49 00           0071          JC      TERR        IF AN ERROR OR ESC WAS RECEIVED
0010 C2 08 00           0072          JNZ     PTAP1       IF VALID HEADER NOT FOUND
                        0073 *    FOUND A VALID HEADER NOW DO COMPARE
0013 E5                 0074          PUSH    H           GET BACK AND RESAVE ADDRESS
0014 11 2D 01           0075          LXI     D,THEAD
0017 CD 0F 01           0076          CALL    DHCMP       COMPARE DE-HL HEADERS
001A E1                 0077          POP     H
001B C2 08 00           0078          JNZ     PTAP1       DIDN T COMPARE...GO BACK TO LOOP
                        0079 *    FOUND IT.. NOW ADJUST REGISTERS FOR READ
001E D1                 0080          POP     D           OPTIONAL  PUT" ADDRESS
001F 7A                 0081          MOV     A,D
0020 B3                 0082          ORA     E           SEE IF DE IS ZERO
0021 2A 34 01           0083          LHLD    BLOCK       GET BLOCK SIZE
0024 EB                 0084          XCHG    .            ..TO DE
                        0085 *  DE HAS HBLOCK....HL HAS USER OPTION
0025 C2 2B 00           0086          JNZ     RTAP        IF DE WAS ZERO GET TAPE LOAD ADDRESS
0028 2A 36 01           0087          LHLD    LOADR       GET TAPE LOAD ADDRESS
                        0088 *
                        0089 *
                        0090 *      THIS ROUTINE READS "DE" BYTES FROM THE TAPE
                        0091 *      TO ADDRESS HL.  THE BYTES MUST BE FROM ONE
                        0092 *      CONTIGUOUS PHYSICAL BLOCK ON THE TAPE.
                        0093 *
                        0094 *          HL HAS "PUT" ADDRESS
                        0095 *          DE HAS SIZE OF TAPE BLOCK
                        0096 *
002B D5                 0097 RTAP     PUSH    D           SAVE SIZE FOR RETURN TO CALLING PROGRAM
                        0098 *
002C 7A                 0099 LOLOOP   MOV     A,D         GET COUNT
002D B3                 0100          ORA     E
002E CA 5A 00           0101          JZ      RTOFF       COUNT IS ZERO-TURN OFF TAPE AND RETURN
0031 EB                 0102          XCHG                GET COUNT TO HL
                        0103 *
0032 01 00 FF           0104          LXI     B,-256      THIS MANY PRIOR TO CRC TEST
```

PROCESSOR TECHNOLOGY CORP.
6200 HOLLIS STREET
EMERYVILLE, CALIF. 94608

MANUAL DATA
CUTS READ AND WRITE ROUTINES

```
0035 09             0105          DAD     B          A LITTLE MATH
0036 D2 4E 00       0106          JNC     LBLK       NO CARRY  IT S THE LAST BLOCK
0039 06 00          0107          MVI     B,0        256 TO READ
003B D3 FC          0108          OUT     0FCH       DING THE PORT FOR INSECURE ROBERTS
                    0109 *
003D 0E 00          0110 RDBLK    MVI     C,0        ZERO THE CRC
003F EB             0111          XCHG    .          ROUND ROBIN
                    0112 *
0040 CD 7D 00       0113 RTLOP    CALL    RHED1      READ IN THIS BLOCK
0043 DA 49 00       0114          JC      TERR       IF ERROR OR ESC
0046 CA 2C 00       0115          JZ      LOLOOP     CONTINUE LOOP IF CRC TEST IS OK
                    0116 *
                    0117 *  ERROR RETURN
                    0118 *
0049 AF             0119 TERR     XRA     A
004A 37             0120          STC     .          SET ERROR FLAGS
004B C3 5B 00       0121          JMP     RTOF1
                    0122 *
                    0123 *  LAST BLOCK--PUT FINAL COUNT IN B
                    0124 *
004E 45             0125 LBLK     MOV     B,L        GET LOWER PORTION OF COUNT
004F 21 00 00       0126          LXI     H,0        TELL DE WE ARE FINISHED
0052 C3 3D. 00      0127          JMP     RDBLK
                    0128 *
                    0129 *
0055 06 01          0130 TOFF     MVI     B,1        SHORT DELAY AFTER WRITE
0057 CD 1F 01       0131          CALL    DELAY
                    0132 *
005A AF             0133 RTOFF    XRA     A
005B D3 FA          0134 RTOF1    OUT     STAPT      TURN OFF THE TAPE
005D D1             0135          POP     D          RETURN BYTE COUNT
005E C9             0136          RET
                    0137 *
                    0138 *
                    0139 *
                    0140 *   READ THE HEADER
                    0141 *
005F 06 0A          0142 RHEAD    MVI     B,10       FIND 10 NULLS
0061 CD 8F 00       0143 RHEA1    CALL    STAT
0064 D8             0144          RC                 IF ESCAPE
0065 DB FB          0145          IN      TDATA      IGNORE ERROR CONDITIONS
0067 B7             0146          ORA     A          ZERO?
0068 C2 5F 00       0147          JNZ     RHEAD
006B 05             0148          DCR     B
006C C2 61 00       0149          JNZ     RHEA1      LOOP UNTIL 10 IN A ROW
                    0150 *
                    0151 *    WAIT FOR THE START CHARACTER
                    0152 *
006F CD 9D 00       0153 SOHL     CALL    TAPIN
0072 D8             0154          RC      .          ERROR OR ESCAPE
0073 3D             0155          DCR     A
0074 C2 6F 00       0156          JNZ     SOHL       WAIT FOR A '1
                    0157 *
```

PROCESSOR TECHNOLOGY CORP.
6200 HOLLIS STREET
EMERYVILLE, CALIF. 94608

MANUAL DATA
CUTS READ AND WRITE ROUTINES

```
                        0158 *
                        0159 *    NOW GET THE HEADER
                        0160 *
0077 21 2D 01           0161          LXI     H,THEAD     POINT TO BUFFER
007A 01 00 10           0162          LXI     B,HLEN*256  LENGTH OF HEADER IN  B ,C<0
                        0163 *
007D CD 9D 00           0164 RHED1    CALL    TAPIN       GET BYTE
0080 D8                 0165          RC
0081 77                 0166          MOV     M,A         STORE IT
0082 23                 0167          INX     H           INCREMENT ADDRESS
0083 CD E6 00           0168          CALL    UDCRC       NOW CALCULATE THE CRC
0086 05                 0169          DCR     B           WHOLE HEADER YET?
0087 C2 7D 00           0170          JNZ     RHED1       LOOP UNTIL DONE
                        0171 *
                        0172 *   THIS ROUTINE GETS THE NEXT BYTE AND COMPARES IT
                        0173 * TO THE VALUE IN REGISTER C.  THE FLAGS ARE SET ON
                        0174 * RETURN.
                        0175 *
008A CD 9D 00           0176 CRCCK    CALL    TAPIN       GET CRC BYTE
008D A9                 0177          XRA     C           COMPARE IT WITH CALCULATED (CLEAR CARRY)
008E C9                 0178          RET
                        0179 *
                        0180 *
                        0181 *    THIS ROUTINE GETS THE NEXT AVAILABLE BYTE FROM THE
                        0182 *  TAPE.  WHILE WAITING FOR THE BYTE THE KEYBOARD IS TESTED
                        0183 *  FOR AN ESC COMMAND.  IF RECEIVED THE TAPE LOAD IS
                        0184 *  TERMINATED AND A RETURN TO THE COMMAND MODE IS MADE
                        0185 *
008F DB FA              0186 STAT     IN      STAPT
0091 E6 40              0187          ANI     TDR
0093 C0                 0188          RNZ     .           WHEN CHARACTER IS READY
0094 DB 01              0189          IN      KDATA
0096 FE 1B              0190          CPI     MODE         ESC ?
0098 C2 8F 00           0191          JNZ     STAT
009B 37                 0192          STC     .           SET ERROR FLAG
009C C9                 0193          RET                 AND RETURN
                        0194 *
                        0195 *
                        0196 *
009D CD 8F 00           0197 TAPIN    CALL    STAT        WAIT UNTIL A CHARACTER IS AVAILABLE
00A0 D8                 0198          RC
                        0199 *
00A1 DB FA              0200 TREDY    IN      STAPT
00A3 E6 18              0201          ANI     TFE+TOE     DATA ERROR?
00A5 DB FB              0202          IN      TDATA       GET THE DATA
00A7 C8                 0203          RZ      .           IF NO ERRORS
00A8 37                 0204          STC     .           SET ERROR FLAG
00A9 C9                 0205          RET
                        0206 *
                        0207 *
                        0208 *
                        0209 *
                        0210 *                  WRITE TAPE BLOCK ROUTINE
```

MANUAL DATA
CUTS READ AND WRITE ROUTINES

PROCESSOR TECHNOLOGY CORP
6200 HOLLIS STREET
EMERYVILLE CALIF. 94608

```
                        0211 *
                        0212 *   ON ENTRY:   A - HAS UNIT AND SPEED
                        0213 *              HL - HAVE POINTER TO HEADER
                        0214 *
                        0215 *
00AA E5                 0216 WTAPE    PUSH     H           SAVE HEADER ADDRESS
00AB CD ED 00           0217          CALL     WHEAD       WRITE THE HEADER
00AE E1                 0218          POP      H
00AF 11 07 00           0219          LXI      D BLKOF     OFFSET TO BLOCK SIZE IN HEADER
00B2 19                 0220          DAD      D           HL POINT TO BLOCK SIZE
                        0221 *  GET ADDRESS AND SIZE FROM HEADER
00B3 5E                 0222          MOV      E,M
00B4 23                 0223          INX      H
00B5 56                 0224          MOV      D,M         DE HAVE SIZE
00B6 23                 0225          INX      H           POINT TO STARTING ADDRESS
00B7 7E                 0226          MOV      A,M
00B8 23                 0227          INX      H
00B9 66                 0228          MOV      H,M
00BA 6F                 0229          MOV      L,A         HL HAVE STARTING ADDRESS
00BB E5                 0230 WRLO1    PUSH     H           FOR STACK CLEAN UP ON TURN OFF
                        0231 *
                        0232 *
                        0233 *    THIS ROUTINE WRITES ONE PHYSICAL BLOCK ON THE
                        0234 *  TAPE "DE" BYTES LONG FROM ADDRESS "HL .
                        0235 *
00BC 7A                 0236 WRLOP    MOV      A,D
00BD B3                 0237          ORA      E           TEST IF COUNT IS ZERO
00BE CA 55 00           0238          JZ       TOFF
00C1 01 00 FF           0239          LXI      B,-256      SUBTRACT 256 FROM IT
00C4 EB                 0240          XCHG
00C5 09                 0241          DAD      B
00C6 D2 D4 00           0242          JNC      WLBLK       IF 256 WEREN T LEFT
00C9 06 00              0243          MVI      B,0
                        0244 *
00CB 0E 00              0245 WDBLK    MVI      C,0         CRC STARTS WITH ZERO
00CD EB                 0246          XCHG                 RESTORE COUNT TO DE.. ADDRESS TO HL
00CE CD 02 01           0247 WDBL1    CALL     WLOOP       WRITE OUT THE BLOCK
00D1 C3 BC 00           0248          JMP      WRLOP       AND GO BACK TO MAJOR LOOP
                        0249 *
00D4 45                 0250 WLBLK    MOV      B,L         REMAINDER OF COUNT
00D5 21 00 00           0251          LXI      H,0         TELL DE WE ARE DONE
00D8 C3 CB 00           0252          JMP      WDBLK
                        0253 *
00DB F5                 0254 WRBYT    PUSH     PSW         SAVE CHARACTER
00DC DB FA              0255 WRWAT    IN       0FAH        GET UART STATUS
00DE E6 80              0256          ANI      80H
00E0 CA DC 00           0257          JZ       WRWAT       WAIT UNTIL IT IS READY
00E3 F1                 0258          POP      PSW
00E4 D3 FB              0259          OUT      0FBH        OUTPUT THE CHARACTER
                        0260 *
                        0261 *   THIS ROUTINE UPDATES THE CRC
                        0262 *
00E6 91                 0263 UDCRC    SUB      C           FORM PARTIAL
```

PROCESSOR TECHNOLOGY CORP
6200 HOLLIS STREET
EMERYVILLE CALIF 94608

MANUAL DATA
CUTS READ AND WRITE ROUTINES

```
00E7 4F          0264          MOV     C,A       SAVE IT
00E8 A9          0265          XRA     C         NOW BEND IT OUT
00E9 2F          0266          CMA     .         GET A FF
00EA 91          0267          SUB     C         CRC+1-1 IS NOT THE SAME
00EB 4F          0268          MOV     C,A       AND RESAVE IT
00EC C9          0269          RET
                 0270 *
                 0271 *
                 0272 *   THIS ROUTINE WRITES THE HEADER POINTED TO BY
                 0273 *   HL TO THE TAPE.
                 0274 *
00ED CD 1B 01    0275 WHEAD    CALL    WTON      TURN ON THE TAPE AND DELAY
00F0 16 32       0276          MVI     D,50      WRITE 50 ZER0S
00F2 AF          0277 NULOP    XRA     A
00F3 CD DB 00    0278          CALL    WRBYT
00F6 15          0279          DCR     D
00F7 C2 F2 00    0280          JNZ     NULOP
                 0281 *
00FA 3E 01       0282          MVI     A,1       50 ZER0S FOLLOWED BY A ONE
00FC CD DB 00    0283          CALL    WRBYT
00FF 01 00 10    0284          LXI     B,HLEN*256  HEADER LENGTH TO B, ZERO TO C
                 0285 *
0102 7E          0286 WLOOP    MOV     A,M       GET CHARACTER
0103 CD DB 00    0287          CALL    WRBYT     WRITE IT TO THE TAPE
0106 05          0288          DCR     B
0107 23          0289          INX     H
0108 C2 02 01    0290          JNZ     WLOOP
010B 79          0291          MOV     A,C       GET CRC
010C C3 DB 00    0292          JMP     WRBYT     PUT IT ON THE TAPE AND RETURN
                 0293 *
                 0294 *
                 0295 *   THIS ROUTINE COMPARES THE HEADER IN THEAD TO
                 0296 *   THE USER SUPPLIED HEADER ADDRESS IN HL.
                 0297 *   ON RETURN IF ZERO IS SET IF THE TWO NAMES COMPARED
                 0298 *
010F 06 05       0299 DHCMP    MVI     B,5       COMPARE FIVE CHARACTERS
0111 1A          0300 DHLOP    LDAX    D         GET ONE PART
0112 BE          0301          CMP     M         COMPARE IT WITH THE OTHER
0113 C0          0302          RNZ     .         RETURN IF NOT THE SAME
0114 05          0303          DCR     B
0115 C8          0304          RZ      .         IF ALL FIVE COMPARED
0116 23          0305          INX     H         COMPARE THE NEXT
0117 13          0306          INX     D
0118 C3 11 01    0307          JMP     DHLOP
                 0308 *
                 0309 *
011B 06 04       0310 WTON     MVI     B,4       SET LOOP DELAY
011D D3 FA       0311 TON      OUT     STAPT     TURN ON THE SELECTED DRIVE
                 0312 *
011F 11 00 00    0313 DELAY    LXI     D,0
0122 1B          0314 DLOP1    DCX     D
0123 7A          0315          MOV     A,D
0124 B3          0316          ORA     E
```

MANUAL DATA
CUTS READ AND WRITE ROUTINES

PROCESSOR TECHNOLOGY CORP
6200 HOLLIS STREET
EMERYVILLE, CALIF. 94608

```
0125 C2 22 01          0317          JNZ     DLOP1       LOOP HERE UNTIL DE ARE ZERO
0128 05                0318          DCR     B
0129 C2 1F 01          0319          JNZ     DELAY       LOOP HERE UNTIL B IS ZERO
012C C9                0320          RET
                       0321 *
                       0322 *
                       0323 *
                       0324 *
                       0325 *
                       0326 *
                       0327 *          PORT ASSIGNMENTS
                       0328 *
     00 FA             0329 STAPT    EQU     0FAH        STATUS PORT GENERAL
     00 FB             0330 TDATA    EQU     0FBH        TAPE DATA
     00 01             0331 KDATA    EQU     1           KEYBOARD DATA PORT FOR ESCAPE TEST
     00 1B             0332 MODE     EQU     1BH         ESCAPE KEY
                       0333 *
                       0334 *
                       0335 *
                       0336 *          BIT ASSIGNMENT MASKS
                       0337 *
                       0338 *
     00 08             0339 TFE      EQU     8           TAPE FRAMING ERROR
     00 10             0340 TOE      EQU     16          TAPE OVERFLOW ERROR
     00 40             0341 TDR      EQU     64          TAPE DATA READY
     00 80             0342 TTBE     EQU     128         TAPE TRANSMITTER BUFFER EMPTY
                       0343 *
     00 40             0344 TAPE1    EQU     64          TAPE ONE  OFF  BIT
     00 80             0345 TAPE2    EQU     128         TAPE TWO  OFF  BIT
                       0346 *
                       0347 *
                       0348 *
                       0349 *
                       0350 *                GLOBAL AREA
                       0351 *
                       0352 *
                       0353 *
                       0354 *          SYSTEM PARAMETER AREA
                       0355 *
                       0356 *
                       0357 *
012D                   0358 THEAD    DS      5           NAME
0132                   0359          DS      1           THIS BYTE MUST BE ZERO FOR AUTO EXECUTE
0133                   0360 HTYPE    DS      1           TYPE
0134                   0361 BLOCK    DS      2           BLOCK SIZE
0136                   0362 LOADR    DS      2           LOAD ADDRESS
0138                   0363 XEQAD    DS      2           AUTO EXECUTE ADDRESS
013A                   0364 HSPR     DS      3           SPARES
                       0365 *
     00 10             0366 HLEN     EQU     $-THEAD     LENGTH OF HEADER
     00 07             0367 BLKOF    EQU     BLOCK-THEAD  OFFSET TO BLOCK SIZE
                       0368 *
                       0369 *
```

PROCESSOR TECHNOLOGY CORP.
6200 HOLLIS STREET
EMERYVILLE, CALIF. 94608

MANUAL DATA
CUTS READ AND WRITE ROUTINES

0370 *

PROCESSOR TECHNOLOGY CORP.
6200 HOLLIS STREET
EMERYVILLE CALIF. 94608

MANUAL DATA
CUTS READ AND WRITE ROUTINES

WRWAT	00DC	0257
WTAPE	00AA	
WTON	011B	0275
XEQAD	0138	

PORT Ø ASSIGNMENTS – INPUT

PRESENT BIT		FOR PORT #:	I/O NEW	PROPOSED		
Ø	SELECTRIC CASE	3	OUT	Ø	$\overline{KDR}$	PORT 3 Keyboard in
1	– UNASSIGNED –			①	$\overline{PDR}$	PORT 2
2	TAPE READER (XDRA)	2	IN	②	$\overline{PXDR}$	PORT 2
3	– UNASSIGNED –			③	UNASSIGNED	
4	TAPE PUNCH (XDRA)	2		④	~~UNASSIGNED~~	SELECTRIC CASE - Port 3
5	SELECTRIC (XDRB)	3	OUT OK	⑤	XDRB	Port 3 (Selectric ready)
6	TVT (RDA)	1	OK	⑥	RDA	Port 1 } TVT, NO CHANGE
7	TVT (TBE)	1	OK	⑦	TBE	Port 1 }

CHANGES

1. Move selectric case bit from bit Ø to bit ④
   Modify selectric I/O routine (papertape) to accommodate bit 4
2. Move PTR ready from bit 2 to bit 1 (Ø4 to Ø2)
   Modify Intel monitor in PROM to accommodate
3. Move PTP ready from bit 4 to bit 2 (1Ø to Ø4)
   Modify Intel monitor in PROM to accommodate
4.