User's Manual
PROGRAM
THE
STAND-ALONE
OPERATING
SYSTEM

093-000062-03

ABSTRACT

Data General's Stand-alone Operating System (SOS) provides I/O support to programs in a non-disk environment and includes utility programs for software development. SOS can be used in systems with at least 8K of core. Systems having cassette or magnetic tape drives can use a special set of utility programs that includes a command line interpreter (CLI). The CLI performs certain file maintenance tasks and allows mnemonic loading of utility programs from a master cassette or magnetic tape reel.

Ordering No. 093-000062

Printed in the United States of America
Rev. 03, June 1973

Original Release - October 1971
First Revision - July 1972
Second Revision - February 1973
Third Revision - June 1973

This revision to the Stand-alone Operating System User's Manual, 093-000062-03, supersedes 093-000062-02 and constitutes a minor revision, a complete list of changes can be found at the back of the manual.

INTRODUCTION

DATA GENERAL's Stand-alone Operating System performs input/output for programs that execute in a non-disk environment. Included in SOS are a number of utility programs including the relocatable assembler, extended relocatable loader, symbolic text editor, SYSGEN, and the library file editor. SOS routines perform I/O on an interrupt driven basis using core buffers unique to each declared device.

The user communicates with SOS through system command words built into his program. Using these system command words, the user can: open and close files, get file attributes, read and write a line, determine the value of and change the value of NMAX, etc. The program interface used to communicate with SOS is similar to that for the Real Time Disk Operating System; in this regard, SOS may be considered a subset of RDOS. The RDOS to SOS Interface Program (DSOSI) makes SOS program interface RDOS compatible.

The SOS main program supports by default a full teletype. A full teletype is the teletype printer, keyboard, reader and punch. Alternatively, support for the teletype printer and keyboard alone may be obtained. Other system devices available are the high-speed paper tape reader, high-speed paper tape punch, line printer (both 80-column and 132-column), card reader, mark sense card reader, plotter, up to eight cassette drives, and up to eight magnetic tape drives.

For systems supporting at least one cassette or magnetic tape unit, included in SOS are a Command Line Interpreter (CLI) and a Core Image Loader/Writer. These systems may also benefit from the added convenience of operation from a master cassette or magnetic tape reel.

SOS contains two User Application Routines. The first is called SAVRE, which is the Save-Restore Program. This program maintains a user-supplied stack to save the caller's registers when a subroutine is called, and the program restores the information on the stack upon returns.

The second program is the Command Table Builder (CTB). This program reads a command line from the teletypewriter into a user-supplied core block and dissects the line into a table of string (byte) pointers and flag bit settings. This table is the SOS equivalent to the RDOS COM.CM file.

For those systems with a Command Line Interpreter (those with one or more cassette or magnetic tape units) the user may use the CLI to load the assembler, an absolute binary paper tape, cassette file, the text editor, the library file editor, a magnetic tape file, the relocatable loader and SYSGEN. Additionally, the CLI may be used to initialize or release a cassette or magnetic tape unit, make a save file, or transfer a file.

CHAPTER 5 - SOS UTILITY PROGRAMS (Continued)

CHAPTER 6 - CONFIGURING SOS UTILITY PROGRAMS

APPENDIX A - UTILITY PROGRAM OPERATION

APPENDIX B - ADDING DEVICE HANDLERS TO SOS

The Stand-alone Operating System (SOS) performs I/O for programs that execute in a non-disk environment. SOS routines perform I/O on an interrupt driven basis, using core buffers unique to each declared device. The program interface used to communicate with SOS is similar to that for the Real Time Disk Operating System (RDOS) as described in DGC Manual 093-000075, Chapter 4. In this regard, SOS may be considered a subset of RDOS.

Included in SOS are a number of utility programs including relocatable assembler, extended relocatable loader, symbolic text editor, library file editor, and SYSGEN. A command line interpreter (CLI) and core image loader/writer are included in SOS for systems with one or more cassette or magnetic tape drives. This CLI allows mnemonic loading of utility programs from a master cassette, as well as performing a number of file maintenance tasks.

SOS TAPES

The SOS library tape (099-000010) contains the main SOS program and the drive routines for all SOS I/O devices except cassette and magnetic tape drives. In addition, the SOS library contains the RDOS to SOS interface program (DSOSI) and two SOS user application routines: the save-restore program (SAVRE) and the command table builder (CTB); SAVRE and CTB are discussed in Chapter 4. Driver routines for cassette drives are supplied as a separate cassette library (099-000041). Driver routines for magnetic tape drives are supplied as a separate magnetic tape library (099-000042).

For systems with neither cassette drives nor magnetic tape transports, only the following SOS tapes are supplied.

Name	Number
SOS Library	099-000010
SOS Text Editor (RB)	089-000104
SOS Extended Assembler (RB)	089-000106
SOS Extended Assembler (AB)	091-000069
SOS Library File Editor (RB)	089-000081
SOS Library File Editor (AB)	091-000057
SOS SYSGEN (RB)	089-000122
SOS SYSGEN (AB)	091-000070
RDOS User Parameters	090-000883
SOS Stand-alone Parameters	090-000498
SOS User Application Parameters	090-000889
Extended Assembler Command Definitions	
Nova Basic Instructions	090-001482
Floating Point Interpreter	090-001483
Operating Systems	090-001484

(RB = relocatable binary; AB = absolute binary)

The absolute binary programs listed above are configured to support a full teletype, high-speed paper tape reader and punch, and line printer. The

SOS TAPES (Continued)

relocatable binary tapes can be used to produce absolute binary tapes (or core image file for systems with cassette or magnetic tape drives) with arbitrary I/O configurations. The procedures for configuring these programs is described in Chapter 6, Configuring SOS Utility Programs.

For systems with one or more cassette drives, the following additional SOS tapes are supplied:

Name	Number
SOS Cassette Driver Library	099-000041
Cassette Core Image Loader/Writer(AB)	091-000067
SOS Relocatable Loader (RB)	089-000120
SOS Command Line Interpreter (RB)	089-000121
SOS SYSGEN with Cassette Support (AB)	091-000071
SOS CLI with Cassette Support (AB)	091-000072
SOS Relocatable Loader with Cassette Support (AB)	091-000073
SOS 132-column Line Printer Driver	089-000148

The absolute binary programs with cassette support are configured to support cassette units 0, 1, and 2, as well as a full teletype, high-speed paper tape reader and punch, and line printer.

For systems with one or more magnetic tape transports, the following additional SOS tapes are supplied:

Name	Number
SOS Magnetic Tape Driver Library	099-000042
Magnetic Tape Core Image Loader/ Writer (AB)	091-000068
SOS Relocatable Loader (RB)	089-000120
SOS Command Line Interpreter (RB)	089-000121
SOS SYSGEN with Magnetic Tape Support (AB)	091-000074
SOS CLI with Magnetic Tape Support(AB)	091-000075
SOS Relocatable Loader with Magnetic Tape Support (AB)	091-000076
SOS 132-column Line Printer Driver	089-000148

The absolute binary programs with magnetic tape support are configured to support magnetic tape units 0, 1, and 2, as well as a full teletype, high-speed paper tape reader and punch, and line printer.

LOADING SOS

SOS I/O support software is loaded by the extended stand-alone relocatable loader (091-000038) or the SOS relocatable loader. To load the main SOS program, the symbol .SOS must have been declared as external normal in a previously loaded program. When the RDOS to SOS Interface Program (DSOSI) is used, the symbol .DSI must be declared external normal instead of .SOS; DSOSI when

LOADING SOS (Continued)

loaded causes the SOS main program to be loaded as well.

The SOS main program supports a full teletype (teletype keyboard, printer, reader, and punch) by default. Support for the other system devices (high-speed paper tape reader, high-speed paper tape punch, line printer, card reader, plotter, up to eight cassette drives, and up to eight magnetic tape transports) may be obtained by loading the appropriate drivers. Support for teletype printer and keyboard (instead of a full teletype) may be obtained by loading the small teletype driver. Table 1 lists the symbols that must be declared external normal in order to load each of these optional drivers.

The cassette and magnetic tape drivers are contained in two libraries separate from the SOS library: the SOS cassette library and the SOS magnetic tape library. Loading any control table/core buffer for a cassette or magnetic tape unit automatically results in loading the associated driver (CTADR or MTADR) and control table/core buffers for all units whose number is less than the specified number. Thus loading CTU4 automatically results in loading CTADR, CTU0, CTU1, CTU2 and CTU3 as well.

SOS I/O DEVICES

A primary difference between RDOS and SOS is that under RDOS files are assigned by file name to temporary channel numbers, while under SOS channel numbers are fixed and always denote the same device (file). File names are not recognized by SOS.

The RDOS to SOS Interface Program (DSOSI) makes the SOS program interface RDOS compatible. Using DSOSI, user programs open devices (files) by name on RDOS channel numbers and reference them thereafter by RDOS channel numbers (0-7). Mixing of calls within a program is not permitted; each user program must be consistent in issuing either RDOS or SOS commands (but not both) to perform I/O. Programs that use DSOSI should contain only RDOS commands and must declare .DSI (not .SOS) as an external normal. When the program is loaded, external normal .DSI will cause DSOSI to be loaded from the SOS library.

MASTER CASSETTE OR MAGNETIC TAPE OPERATION

Systems that include one or more cassette or magnetic tape drives may benefit from the added convenience of operation from a master cassette or magnetic tape reel. A master reel contains in core image form a core image loader/writer, command line interpreter, text editor, extended assembler, extended relocatable

MASTER CASSETTE OR MAGNETIC TAPE OPERATION (Continued)

loader, library file editor, and SYSGEN. When the master reel is loaded on cassette or magnetic tape unit 0, the core image loader/writer can be bootstrapped and the other programs can be loaded by mnemonic commands to the command line interpreter. The command line interpreter also performs a number of basic file maintenance tasks.

To obtain the maximum convenience from master reel operation in software development, more than one cassette or magnetic tape unit should be available.

TABLE 1

Program (Title)	Tape	External Symbol For Loading	Function
STTY	SOS library	.STTY	Small teletype driver (keyboard and printer only.)
PTRDR	SOS library	.PTRD	High-speed paper tape reader driver.
PTPDR	SOS library	.PTPD	High-speed paper tape punch driver.
LPTDR	SOS library	.LPTD	Line printer driver.
CDRDR	SOS library	.CDRD	Card reader driver.
PLTDR	SOS library	.PLTD	Plotter driver.
CTADR	Cassette library	.CTAD	Cassette driver containing control table/core buffer to support cassette unit 0.
CTU1	Cassette library	.CTU1	Control table/core buffer to support cassette unit 1.
CTU2	Cassette library	.CTU2	Control table/core buffer to support cassette unit 2.
.	.	.	.
.	.	.	.
.	.	.	.
CTU7	Cassette library	.CTU7	Control table/core buffer to support cassette unit 7.
MTADR	Magnetic tape library	.MTAD	Magnetic tape drive containing control table/core buffer to support magnetic tape unit 0.
MTU1	Magnetic tape library	.MTU1	Control table/core buffer to support magnetic tape unit 1.
MTU2	Magnetic tape library	.MTU2	Control table/core buffer to support magnetic tape unit 2.
.	.	.	.
.	.	.	.
.	.	.	.
MTU7	Magnetic tape library	.MTU7	Control table/core buffer to support magnetic tape unit 7.

TABLE OF EXTERNAL NORMAL SYMBOLS FOR DRIVERS

SOS can provide to user programs I/O support for teletype keyboard input ($TTI), teletype printer ($TTO), teletype paper tape reader ($TTR), teletype paper tape punch ($TTP), high-speed paper tape reader ($PTR), high-speed paper tape punch ($PTP), line printer ($LPT), card reader ($CDR), plotter ($PLT), eight cassette drives (CT0, CT1, ... CT7) and eight 9-track magnetic tape drives (MT0, MT1, ... MT7). Additional support capability may be added to SOS through the addition of driver routines to the SOS libraries as described in Appendix B.

SOS DEVICE DRIVER ROUTINES

SOS device driver routines are contained in three libraries: the SOS library, the SOS cassette library, and the SOS magnetic table library. The SOS Library contains the following routines:

Name	Purpose
CTB	command table builder (SOS user subroutine).
SAVRE	save/return routine (SOS user subroutine).
DSOSI	RDOS to SOS interface program.
PLTDR	plotter driver.
CDRDR	card reader driver.
PTRDR	paper tape reader driver.
LPTDR	line printer driver.
PTPDR	paper tape punch driver.
STTYDR	small teletype driver ($TTI, $TTO only).
SOS	main program of SOS.
BTTYDR	full teletype driver ($TTI, $TTO, $TTR, $TTP)

The SOS cassette library contains the cassette unit driver modules:

Name	Purpose
CTU1	Buffer and control table for cassette unit 1 (CT1)
CTU2	Buffer and control table for cassette unit 2 (CT2)
.	.
.	.
.	.
CTU7	Buffer and control table for cassette unit 7 (CT7)
CTADR	Cassette driver (includes buffer and control table for cassette unit 0 (CT0)).

The SOS magnetic tape library contains the magnetic tape unit driver modules:

Name	Purpose
MTU1	Buffer and control table for magnetic tape unit 1 (MT1).
MTU2	Buffer and control table for magnetic tape unit 2 (MT2).
.	.
.	.
.	.
MTU7	Buffer and control table for magnetic tape unit 7 (MT7).
MTADR	Magnetic tape driver (includes buffer and control table for magnetic tape unit 0 (MT0)).

SPECIFYING DEVICES BY NAME OR BY CHANNEL

Under SOS, user programs can perform I/O through commands to SOS. These commands are discussed in Chapter 3. When running programs under the RDOS to SOS interface program (DSOSI) devices are specified in commands by their device names. When DSOSI is not used, devices are specified by their fixed channel numbers. The following is a table of these numbers:

Device Name	Channel Number
$PLT	6
$TTP	10
$CDR	11
$TTO	12
$TTI	13
$LPT	14
$PTR	15
$PTP	16
$TTR	17
MT0	20
MT1	21
.	.
.	.
MT7	27
CT0	30
CT1	31
.	.
.	.
CT7	37

SOS UTILITY PROGRAM I/O

Each of the utility programs included in SOS (except the core image loader/writer and text editor) is supplied in two forms: a relocatable binary form and an absolute binary form. The relocatable binary form must be configured as explained in Chapter 6, and may be configured with arbitrary I/O support. The absolute binary is preconfigured with support for full teletype ($TTI, $TTO, $TTR, $TTP), high-speed paper tape reader/punch ($PTR, $PTP) and line printer ($LPT). For systems that include one or more cassette drives, they are also configured with support for three cassette drives (CT0, CT1, and CT2). For systems that include one or more magnetic tape drives, they are also configured for three magnetic tape drives (MT0, MT1, and MT2).

The user communicates with the Stand-alone Operating System through system command words assembled into his program.

SYSTEM COMMAND FORMAT

All SOS commands have the same format:

```
.SYSTM
command
error return      ;STATUS IN AC2
normal return     ;AC'S AND CARRY PRESERVED
```

The mnemonic .SYSTM and the SOS command words are recognized as legal mnemonics by the DGC Stand-alone Relocatable Assembler and the RDOS Relocatable Assembler.

The mnemonic .SYSTM must immediately precede the command. Appearance of the mnemonic .SYSTM results in the assembly of a

```
JSR @ 17
```

instruction which allows system communication through the main system entry address stored in page zero. The system command word must be assembled as the word following the .SYSTM.

Once system action is completed, normal return is made to the second instruction after the system command word. If an exceptional condition is detected, return is made to the first instruction following the system command word.

System commands have the form either of a mnemonic or a mnemonic followed by a channel number:

command or command n

where n is a digit that represents the fixed I/O channel (device) number ($0\text{-}76_8$). When no I/O channel is needed for command execution, the command word, command, appears alone in the instruction. If the command requires arguments, these are passed in the accumulators.

One argument commonly passed in an accumulator is a byte pointer. A byte pointer contains the word address in bits 0-14, which contain or will receive the byte. Bit 15 specifies which half (0 left; 1 right). Note that this is the reverse of the byte pointer as specified in "How to Use the Nova Computers." To use the subroutine shown on page 2-21 of the manual, change the MOV 0 0 SZC instruction to a MOV 0 0 SNC instruction.

Any command requiring a channel number n need not specify this number in the command itself. By specifying octal 77 (the device code of the CPU), as the channel number in the instruction, the system will use instead the number passed in AC2.

SYSTEM COMMAND FORMAT (Continued)

For example, the following instructions specify a write to channel 16:

```
        LDA     2, C16
        .SYSTM
        .WRS    CPU     ;CPU=OCTAL 77
        JSR     EOF
        .
        .
        .
C16:    16
```

STATUS ON RETURN FROM SYSTEM

Status of the accumulators upon return from the system is as follows:

If the system returns no information as a result of the call, the Carry and all accumulators except AC3 will be preserved.

AC2 is used when an exceptional return is made to return a numeric error code. Error codes are listed by number at the end of this chapter and the applicable codes are listed for each command.

AC3 is destroyed by .SYSTM (as it is by the use of JSR). On return from the system, however, AC3 is loaded from the contents of memory location 00016_8. This location is defined as a permanent symbol by the DGC Assembler and has the name USP (User Stack Pointer). A convenient method of saving AC3 is to store it in location 00016_8 before issuing .SYSTM.

LIST OF COMMAND WORDS

The following is a list of the command word mnemonics:

.SYSI	Initialize SOS devices.
.OPEN	Open a file.
.CLOSE	Close a file.
.RESET	Close all open files.
.GTATR	Get the file attributes.
.RDS	Read sequential characters.
.RDL	Read sequential lines.
.WRS	Write sequential characters.
.WRL	Write sequential lines.
.GCHAR	Read a character from the TTI.
.PCHAR	Write a character to the TTO.
.MEM	Determine available memory space.
.MEMI	Allocate an increment of memory.

The SOS commands above are a subset of the RDOS commands. All other RDOS commands, including .RTN, result in an error return with the error code:

2 (ERICM -- Illegal system command)

returned in AC2.

INPUT/OUTPUT COMMANDS

All I/O is handled by system I/O commands. These commands require a channel number to be given in the second field of the command word. If the channel number is 77, then AC2 must contain the desired channel number. The system provides two basic modes for reading and writing files.

The first mode is the line mode, where data read or written is assumed to consist of ASCII character strings terminated by either carriage returns, form feeds or nulls. In this mode, the system handles all device-dependent editing at the device driver level. For example, line feeds are ignored on paper tape input devices and supplied after carriage returns on all paper tape output devices. Further, reading and writing never require byte counts, since reading continues until a carriage return is read and writing proceeds until a carriage return is written. The line mode commands are .RDL and .WRL.

The second mode is unedited sequential mode. In this mode, data is transmitted exactly as read from the file or device. No assumption is made by the system as to the nature of this information. Thus, this mode would always be used for processing binary files. This mode requires the user program to specify byte counts necessary to satisfy a particular read or write request. The sequential mode commands are .RDS and .WRS.

Initialization of Communications (.SYSI)

The .SYSI command must be issued before any other SOS commands are used. It initializes all tables, clears each SOS device, restores NMAX to its value at load time, and enables interrupts from all devices. Additional .SYSI commands can be given if the user wishes to clear devices on restarts.

The format of the command is:

```
.SYSTM
.SYSI           ;INITIALIZE SOS
error return    ;NEVER TAKEN
normal return   ;AC'S AND CARRY RESERVED
```

Open a File (.OPEN)

Before other I/O commands can be used, a device must be opened with the .OPEN command. Issuance of the .OPEN command initially links a channel to a particular file. This association between a file and channel number can be broken by using the .CLOSE command; and all currently open files can be closed using the .RESET command. The contents of AC1 are used as a mask to control the device characteristics that are set for the duration of the .OPEN. For every bit set in the word, the corresponding device characteristic (See .GTATR command) is inhibited. If AC1 contains 0, then no characteristics are inhibited. If the device is attribute protected (see .GTATR command) the mask is ignored.

Open a File (.OPEN) (Continued)

The .OPEN command results in the initialization of the control table for the device, the output of leader on paper tape devices, or a prompt message for input devices requiring user intervention. The format of the .OPEN command is:

```
.SYSTM
.OPEN n         ;OPEN CHANNEL n
error return
normal return
```

Possible errors resulting from the .OPEN command are:

AC2	Mnemonic	Meaning
0	ERFNO	Illegal channel number.
3	ERICD	Illegal command for device.
12	ERDLE	File doesn't exist.
31	ERSEL	Unit not properly selected.

Close a File (.CLOSE)

After use, files may be closed to insure an orderly ending sequence. The format of the .CLOSE command is:

```
.SYSTM
.CLOSE n        ;CLOSE CHANNEL n
error return
normal return
```

Possible errors resulting from a .CLOSE command are:

AC2	Mnemonic	Meaning
0	ERFNO	Illegal channel number.
15	ERFOP	File not open.

Close All Files (.RESET)

The command causes all currently open files to be closed. The format of the .RESET command is:

```
.SYSTM
.RESET
error return
normal return
```

The error return from this command is never taken.

Get File Attributes (.GTATR)

This command obtains the attributes of a file or the characteristics of a device. To obtain attributes, the file must be opened (see .OPEN command). The number of the channel is given in the system command. The format of the .GTATR command is:

```
.SYSTM
.GTATR n        ;CHANNEL n
error return
normal return
```

Upon return, AC0 contains the file attributes. The attribute settings and their meanings are given below:

Bit	Mnemonic	Meaning
1B0	ATRP	Device is read protected.
1B1	ATCHA	Device is attribute protected; a file or device with this attribute contains unsuppressable device characteristics (e.g., the $TTI is attribute protected and hence always contains the DCKEY characteristic while it is open.)
1B14	ATPER	Permanent file. All devices are permanent files.
1B15	ATWP	Device is write protected.

AC1 contains the device characteristics of the file. The bit/characteristic correspondence used in interpreting the bit configuration returned in AC1 is shown below:

Bit	Mnemonic	Meaning
1B0	DCDIR	Directory device. SOS recognizes the characteristic as indicative of a data channel block transfer device.
1B1	DCC80	An 80-column device. Applicable to the card reader and line printer.
1B2	DCLTU	Device changing lower case ASCII to upper case.
1B3	DCFFO	Device requiring form feeds on opening.
1B4	DCFWD	Full word device (reads or writes more than one byte.)
1B6	DCLAC	Output device requiring line feeds after carriage returns.
1B7	DCPCK	Input device requiring a parity check. Output device requiring parity to be computed.

Get File Attributes (.GTATR) (Continued)

Bit	Mnemonic	Meaning
1B8	DCRAT	Output device requiring a rubout after every tab.
1B9	DCNAF	Output device requiring nulls after every form feed.
1B10	DCKEY	A keyboard input device.
1B11	DCTO	A keyboard output device.
1B12	DCCNF	Output device without form feed hardware.
1B13	DCIDI	Device requiring operator intervention.
1B14	DCCGN	Output device without tabbing hardware.
1B15	DCCPO	Output device requiring leader/trailer.

Possible errors resulting from a .GTATR command are:

AC2	Mnemonic	Meaning
0	ERFNO	Illegal channel number.
15	ERFOP	Attempt to get attributes of an unopened file.

Read a Line (.RDL)

This command causes an ASCII line, having even parity, to be read. AC0 must contain a byte pointer to the starting byte address within the user area into which the line will be read.

Reading will terminate normally after transmitting either a carriage return, null, or a form feed to the user. Reading will terminate abnormally after transmission of 132 characters (decimal) without detecting a carriage return, null, or form feed, upon detection of a parity error, or upon end of file. In all cases, the byte count read will be returned in AC1. If the read is terminated because of a parity error, the character having incorrect parity will be stored (high order bit zero) as the last character read. The byte pointer to the character can always be computed as:

C(AC0) + C(AC1) - 1

where: C(x) means "the contents of x."

Read a Line (.RDL) (Continued)

The format of the .RDL command is:

```
.SYSTM
.RDL n          ;READ FROM CHANNEL n
error return
normal return
```

Possible errors resulting from a .RDL command are:

AC2	Mnemonic	Meaning
0	ERFNO	Illegal channel number.
3	ERICD	Illegal command for device.
6	EREOF	End of file.
7	ERRPR	Attempt to read a read protected file.
15	ERFOP	File not open.
22	ERLLI	Line limit (132 characters) exceeded.
24	ERPAR	Parity error.
30	ERFIL	File read error.

Write a Line (.WRL)

The command assumes an ASCII file. AC0 must contain a byte pointer to the starting byte address within the user area from which characters will be read.

Writing will terminate normally upon writing a null, carriage return, or a form feed, and abnormally after transmission of 132 (decimal) characters without detection of a carriage return, a null, or a form feed. In either case, AC1 will contain, upon termination, the number of bytes read from the user area to complete the request. The termination of a write line on a null allows for formatting output without forcing a carriage return.

The format of the .WRL command is:

```
.SYSTM
.WRL n          ;WRITE TO CHANNEL n
error return
normal return
```

Possible errors resulting from the .WRL command are:

AC2	Mnemonic	Meaning
0	ERFNO	Illegal channel number.
3	ERICD	Illegal command for device.
10	ERWPR	Attempt to write a write protected file.
15	ERFOP	File not open.
22	ERLLI	Line limit (132 characters) exceeded.

Read Sequential (.RDS)

Sequential mode transmits data exactly as read from the file. AC0 must contain a byte pointer to the starting byte address within the user area into which the data will be read and AC1 must contain the number of bytes to be read. The format of the .RDS command is:

```
.SYSTM
.RDS n          ;READ FROM CHANNEL n
error return
normal return
```

Possible errors resulting from a .RDS command are:

AC2	Mnemonic	Meaning
0	ERFNO	Illegal channel number.
3	ERICD	Illegal command for device.
6	EREOF	End of file.
7	ERRPR	Attempt to read a read-protected file.
15	ERFOP	File not open.
30	ERFIL	File read error.

Upon end of file the partial count will be returned in AC1.

Write Sequential (.WRS)

.WRS transmits data exactly as read from the user area. AC0 must contain a byte pointer to the starting address of the data within the user area and AC1 must contain the number of bytes to be written. The format of the .WRS command is:

```
.SYSTM
.WRS n          ;WRITE TO CHANNEL n
error return
normal return
```

Possible errors resulting from a .WRS command are:

AC2	Mnemonic	Meaning
0	ERFNO	Illegal channel number.
3	ERICD	Illegal command for device.
10	ERWPR	Attempt to write a write protected file.
15	ERFOP	File not open.

TELETYPEWRITER COMMANDS

Buffered transfer of single characters between the teletype and AC0 is handled by the commands .GCHAR and .PCHAR. No channel number is required for the commands, and the teletype is always considered "open" to them.

Get a Character (.GCHAR)

The command returns a character typed from the teletype in AC0. The character is right-justified in AC0 with bits 0-8 cleared. No channel is required; the TTI is always used as input for this command. The format of the .GCHAR command is:

```
.SYSTM
.GCHAR
error return
normal return
```

No error return is possible from this command.

Put a Character (.PCHAR)

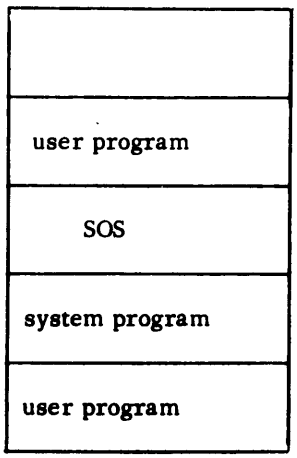
This command transmits a character in AC0, bits 9-15, to the teletypewriter. No channel is required; the TTO is always used as output for this command. The format of the .PCHAR command is:

```
.SYSTM
.PCHAR
error return
normal return
```

No error return is possible from this command.

MEMORY COMMANDS

Upon the completion of a relocatable load, the Stand-alone Operating System resides in lower memory, among various user or system programs comprising the load module. Memory then looks, essentially, as follows:



← top of memory HMA (highest memory address available)
← NMAX (first location available above loaded programs)
← bottom of memory

MEMORY COMMANDS (Continued)

The highest memory address available (HMA) is usually the first word below the Binary Loader. If a user symbol table has been loaded at the high end of user memory, the high memory address will be the first word below the user symbol table.

The .MEM command returns both the current value of NMAX and HMA. The .MEMI command allows the user to adjust the value of NMAX.

Determine Available Memory (.MEM)

This command returns the current value of NMAX in AC1 and the value of HMA in AC0. HMA represents either the bottom of the binary loader or the end of the user symbol table. A SUB 1,0 determines the limit of memory available to the user program. The format of the .MEM command is:

```
.SYSTM
.MEM
error return
normal return
```

There are no error returns from this command.

Change NMAX (.MEMI)

This command allows the user to increase or decrease the value of NMAX. The increment or decrement (in two's complement) is passed in AC0. The command causes the value of NMAX to be updated in the User Status Table and the new NMAX to be returned in AC1. The format of the command is:

```
.SYSTM
.MEMI
error return
normal return
```

NMAX will not be changed if the new value of NMAX would be higher than HMA. No check is made as to whether or not the user decreases NMAX below its original value as determined at relocatable load time.

Whenever a program requires memory space above the loaded program, a .MEMI should be executed first to allocate the number of words needed. The allocated memory space may be used by programs for buffers, user stacks, temporary storage, etc. There is one error return from a .MEMI command:

AC2	Mnemonic	Meaning
26	ERMEM	Attempt to allocate more memory than available.

ERROR MESSAGES

Code	Mnemonic	Meaning	Applicable Commands
0	ERFNO	Illegal channel number.	.OPEN .CLOSE .GTATR .RDL .RDS .WRL .WRS
2	ERICM	Illegal system command.	- - -
3	ERICD	Illegal command for device.	.RDL .RDS .WRL .WRS .OPEN
6	EREOF	End of file.	.RDL .RDS
7	ERRPR	Attempt to read a read protected file.	.RDL .RDS
10	ERWPR	Attempt to write a write protected file.	.WRL .WRS
12	ERDLE	File does not exist.	.OPEN
15	ERFOP	File not opened.	.GTATR .CLOSE .RDL .RDS .WRL .WRS
22	ERLLI	Line limit exceeded on read or write line.	.RDL .WRL
24	ERPAR	Parity error on read line.	.RDL
26	ERMEM	Attempt to allocate more memory than available.	.MEMI
30	ERFIL	File read error.	.RDL .RDS
31	ERSEL	Unit not properly selected.	.OPEN

DEVICE RESPONSE TO SOS COMMANDS

This section describes the functions performed by the SOS I/O commands, as applied to each of the devices supported by DGC.

$PLT

.OPEN	Device is initialized.
.CLOSE	Device is reinitialized after outstanding I/O is complete.
.WRS	The specified bytes are output to the device, unedited.
.WRL	Illegal command to this device.

DEVICE RESPONSE TO SOS COMMANDS (Cont'd)

$TTP

.OPEN	Device is initialized; leader is punched.
.CLOSE	Device is reinitialized after outstanding I/O is complete; trailer is punched.
.WRS	The specified bytes are output to the device, unedited.
.WRL	The ASCII string is output to the device with rubout characters inserted after tabs, a line feed inserted after carriage returns, and nulls inserted after form feeds.

$CDR

.OPEN	Device is initialized; a prompt message is written and a response is necessary for the program to continue.
.CLOSE	Device is reinitialized.
.RDS	The specified bytes are read into the user area from the device, unedited. Each pair of bytes read represents one full column of the card. Bits 4-7 of the first byte represent card rows 12, 11, 0, and 1 respectively. Bits 0-7 of the second byte represent card rows 2-9 respectively. A one in bit 0 of the first byte of a pair indicates end of card. No other meaningful data is included in this byte pair.
.RDL	The 80-character ASCII string is read into the user area from the device. If the characteristic DCCB0 was suppressed on the .OPEN, then 72 columns are read. The translation from Hollerith is performed in the card reader driver. A 12-11-0-1-2-3-4-5-6-7-8-9 punch causes end of file. The byte count returned to the user reflects the last non-blank character on the card.

$TTO

.OPEN	Device is initialized.
.CLOSE	Device is reinitialized after outstanding I/O is complete.
.WRS	The specified bytes are output to the device, unedited.
.WRL	The ASCII string is output to the device with simulated tabbing, a line feed inserted after carriage return, and nulls inserted after form feeds.

DEVICE RESPONSE TO SOS COMMANDS (Cont'd)

$TTI

.OPEN	Device is initialized.
.CLOSE	Device is reinitialized.
.RDS	The specified bytes are read into the user area from the device, unedited.
.RDL	The ASCII string is read into the user area from the device. The input stream is is echoed on the $TTO. A rubout character deletes the previous input character and causes a back arrow to be echoed. The shift L character causes the entire input string to be deleted. Line feeds are ignored.

$TTR

.OPEN	Device is initialized; a prompt message is written and a response is necessary for the program to continue.
.CLOSE	Device is reinitialized.
.RDS	The specified bytes are read into the user area from the device, unedited.
.RDL	The ASCII string is read into the user area from the device. Rubouts, line feeds and nulls are ignored.

CTA (see also Appendix C, SOS Cassette and Magnetic Tape Files)

.OPEN	The specified file is located on the tape reel, and the read/write head positioned to the file mark preceding.
.CLOSE	Device is reinitialized. Following write operations, the last partial block is output to the file (padded with nulls if necessary) and two end of file marks are written.
.RDS	The specified bytes are read into the user area from the file, unedited.
.RDL	The ASCII string is read into the user area from the file. Rubouts, line feeds and nulls are ignored.
.WRS	The specified bytes are output to the file, unedited.
.WRL	The ASCII string is output to the file.

$LPT

.OPEN	Device is initialized.
.CLOSE	Device is reinitialized; a form feed character is output.
.WRS	The specified bytes are output to the device, unedited.
.WRL	The ASCII string is output to the device with simulated tabbing, and line feeds are inserted after carriage returns.

DEVICE RESPONSES TO SOS COMMANDS (Cont'd)

$LPT (Cont'd)

NOTE: The system can accept 132-column printouts. Tape number 089-000148 must be loaded in order to accomplish this. The tape can be loaded by editing it into the system library, by use of LFE, thereby deleting the 80-column device driver. Or the tape (#089-000148) can be loaded before the system library tape, thus replacing the 80-column device driver for the 132-column device driver.

$PTR

.OPEN	Device is initialized; a prompt message is written and a response is necessary for the program to continue.
.CLOSE	Device is reinitialized.
.RDS	The specified bytes are read into the user area from the device, unedited.
.RDL	The ASCII string is read into the user area from the device. Rubouts, line feeds, and nulls are ignored.

$PTP

.OPEN	Device is initialized; leader is punched.
.CLOSE	Device is reinitialized after outstanding I/O is complete; trailer is punched.
.WRS	The specified bytes are output to the device, unedited.
.WRL	The ASCII string is output to the device with rubouts inserted after tabs, a line feed after a carriage return, and nulls after a form feed.

MTA (See also Appendix C, SOS Cassette and Magnetic Tape Files)

.OPEN	The specified file is located on the tape reel, and the read/write head positioned to the file mark preceding.
.CLOSE	Device is reinitialized. Following write operations, the last partial block is output to the file (padded with nulls if necessary) and two end of files marks are written.
.RDS	The specified bytes are read into the user area from the file, unedited.
.RDL	The ASCII string is read into the user area from the file. Rubouts, line feeds and nulls are ignored.
.WRS	The specified bytes are output to the file, unedited.
.WRL	The ASCII string is output to the file.

SAVE-RESTORE PROGRAM (SAVRE)

This program maintains a user-supplied stack to save the caller's registers (3 accumulators, 2 temporary storage locations, and the caller's return location) when a subroutine is called, and the program restores the information on the stack upon returns. The User Stack Pointer (USP) always points to the executing subroutine's stack frame. This subroutine may then access any of the caller's registers, as well as any of its own, by using the stack displacements defined in the SOS User Application Parameter Tape (Appendix D).

The user must supply the stack before any of the SOS user application routines can be used. To provide the stack, the user merely stores a beginning address for the stack into the USP. The address stored in the cell must point to a core block large enough to meet the user stack requirements. No stack overflow check is made. If the stack is used only with the SOS library routines, then the value of SOSEC in the SOS User Application Parameter Tape gives the maximum number of 6-location frames necessary.

The various features of the SAVRE program are illustrated in the following example of step-by-step use of the stack:

1. To supply the stack, the user may allocate a fixed block of core starting at NMAX:

```
        :
        .SYSTM          ;GET NMAX INTO AC1
        .MEM
        JMP .           ;ERROR
        STA 1, USP      ;INITIALIZE START OF STACK
        LDA 0, STKSZ
        .SYSTM          ;UPDATE NMAX
        .MEMI
        JMP .           ;NOT ENOUGH CORE AVAIL-
          .             ;ABLE
          :
STKSZ:  SOSEC+3*SSEL    ;TOTAL STACK SIZE = SOS
                        ;ENTRY COUNT + USER'S
                        ;ENTRY COUNT * ENTRY
                        ;LENGTH
```

   Or, instead of starting at NMAX, the user may alternatively allocate any fixed block of core:

```
        LDA 0, STACK    ;LOAD STACK POINTER
        STA 0, USP      ;STORE IN USP
          :
          .
STACK:  .+1
        .BLK SOSEC*SSEL ;SOS SIZE IS ADEQUATE
```

SAVE-RESTORE PROGRAM (SAVRE) (Continued)

2. To save the caller's registers when a subroutine is entered via a JSR SUBR instruction:

```
        .EXTN SAVR, RETR
SUBR:   STA 3, @USP     ;SAVE-RETURN LOCA-
                        ;TION IN STACK
        SAVR            ;SAVE REGISTERS
          :             ;AC3 → SUBR's FRAME
```

3. To access the caller's registers (which are intact when the subroutine is entered):

```
        STA 2, OAC0, 3  ;RETURN AC2 IN CALLER'S AC0
        ISZ OAC2, 3     ;INCREMENT THE RETURNED AC2
        ISZ ORTR, 3     ;INCREMENT RETURN LOCA-
                        ;TION COUNTER
        DSZ OT1, 3      ;DECREMENT CALLER'S
                        ;TEMPORARY
        STA 1, T0, 3    ;STORE MY OWN TEMPORARY
```

4. To return to the calling subroutine:

```
        RETR            ;RETURN TO CALLER
```

5. To pop multiple stack levels, i.e., make an exceptional return, the external command, ERETR, has been defined. To illustrate use of this feature, consider five subroutines: A, B, C, D, and E, executing at three levels: 1, 2, and 3.

   Subroutine A executes at level 1. Subroutines B, C, and D execute at level 2. Subroutine E executes at level 3. Subroutine A calls B, C, and D, each of which may call E.

   When a special condition is encountered by B, C, D, or E, subroutine A may want to regain immediate control. This is achieved by performing the ERETR in subroutine 3, provided that subroutine A has performed the logic shown below and on the page following.

```
        .EXTD ERAD, ERUSP
        .EXTN ERETR
A:      STA 3, @USP         ;SAVE REGISTERS
        SAVR
          :
        LDA 0, AERAD
        STA 0, ERAD         ;STORE EXCEPTIONAL
                            ;RETURN ADDRESS
        STA 3, ERUSP
          :
```

SAVE-RESTORE PROGRAM (SAVRE) (Continued)

5. To make an exceptional return (continued)

```
        .
        .
        .
        JSR  B
        .
        .
        .
        JSR  C
        .
        .
        .
        JSR  D
        .
        .
        .
AERAD:  SPECR           ;EXCEPTIONAL RETURN
                        ;ADDRESS
        .
        .
SPECR:  LDA  0, VAR     ;REGAIN CONTROL HERE
        STA  0,OAC1, 3  ;ON EXCEPTIONAL CON-
        .               ;DITION AND CONTINUE
        .
        .
VAR:    .BLK  1
```

This mechanism returns control unconditionally to location SPECR upon the execution of an ERETR by subroutines B, C, D, or E. The stack pointer will be A's original pointer and the caller's registers are unchanged but the registers returned to A are indeterminate. One case in which this feature could be used is in string processing, where special characters may require special processors or signal the termination of a set of processors.

Following is a list summarizing the external declarations necessary to use the Save-Restore program:

```
.EXTN  SAVR, RETR       ;USE OF THESE FEATURES
                        ;IS OPTIONAL

.EXTN  ERETR            ;USE OF THIS FEATURE IS
                        ;OPTIONAL. HOWEVER, IF
                        ;USED, IT REQUIRES THE
                        ;EXTERNAL DISPLACE-
                        ;MENTS THAT FOLLOW

.EXTD  ERUSP, ERAD      ;MUST BE DECLARED AND
                        ;INITIALIZED TO USE
                        ;EXCEPTIONAL RETURN
```

COMMAND TABLE BUILDER (CTB) PROGRAM

This program reads a command line from the $TTI into a user-supplied core block and dissects the line into a table of string (byte) pointers and flag bit settings. Blank characters in the line are considered string separators. The table is the effective SOS equivalent of the RDOS COM.CM file (see 093-000075, Appendix D). The table is intended only for the SOS user employing the RDOS to SOS Interface Program, described in Chapter 1.

The table produced by this program, the translate table used to derive the flag bit settings, and the input buffer for reading the command line must be supplied by the calling program. The SOS user stack (see Save-Restore Program) must also be supplied by the caller.

The command line that is read may be continued by typing a SHIFT N (up arrow) character one position before the carriage return. The RUBOUT key (echoed as a back arrow) causes the preceding character to be deleted from the line. The current line may be deleted by typing SHIFT L (backslash). A facility to output a prompt message before reading the command line also exists in the program.

One command table entry is created for each unique string in the input line. Each of these strings may be modified by multiple alphabetic switches (/a where a is a single letter) and/or one numeric switch (/n where n is a digit 0-9). The slash which designates the switch character may follow the string directly or be separated by one or more blanks. The same applies to succeeding switches modifying the same string. The switch character must directly follow the slash, however. If a blank follows a slash, the next character examined will be considered the start of a new string. Any characters which follow the switch character are ignored. These features are illustrated by the following equivalent argument strings:

```
$PTR/2  /F  /A

$PTR/A/F/2

$PTR/A  /FINAL  /2

$PTR  /ASCII  /F  /2
```

COMMAND TABLE BUILDER (CTB) (Continued)

The layout of the command table built by the program is given in the SOS User Application Parameter Tape, 090-000889. Each unique string scanned in the command line causes a two-word entry to be added to the table. The first word at displacement CTBP from the entry pointer is a byte pointer to the string. The second word at displacement CTSW contains bit settings which correspond with switches that modified the string. This correspondence is established in part by another user-supplied table. Bits 11-15 of this word are reserved for the following use:

Bits 12-15 - A four-bit octal number derived from the numeric switch modifying the string. If no numeric switch is specified, no bits are set. Maximum value is octal 11.

Bit 11 - Set if one or more undefined switches modify the string.

The user-supplied translate table (TRT) establishes the definition of bits 0-10 of this table. This allows eleven user specified alphabetic switches. The address of the translate table is one of the arguments passed to the program. The table occupies at most 11 locations which specify the bit position/alphabetic switch correspondence. The following examples illustrate the use of the table:

```
TRT: "C      ;/C = BIT 0
     "E      ;/E = BIT 1
     "B      ;/B = BIT 2
     "X      ;/X = BIT 3
     "A      ;/A = BIT 4
     "M      ;/M = BIT 5
     "T      ;/T = BIT 6
     "Z      ;/Z = BIT 7
     "H      ;/H = BIT 8
     "P      ;/P = BIT 9
     "S      ;/S = BIT 10
     -1      ;TERMINATES TRANSLATE TABLE
```

The above example of a translate table defines eleven alphabetic switches - C, E, B, X, A, M, T, Z, H, P, and S - to map respectively to bits 0-10 of the switch word. All other alphabetic switches result in the setting of bit 11 of the switch word.

```
TRT: "A      ;/A = BIT 0
     "L      ;/L = BIT 1
     "B      ;/B = BIT 2
     "N      ;/N = BIT 3
     -1      ;TERMINATES TABLE
```

COMMAND TABLE BUILDER (CTB) (Continued)

The previous translate table defines four alphabetic switches - A, L, B, and N - to map to bits 0-3 of the switch word. All other alphabetic switches result in the setting of bit 11 of the switch word. The utility of the table may be increased by inserting the following equivalences into the program:

```
SWA = SW0        SWB = SW2
SWL = SW1        SWN = SW3
```

Symbols SW0, SW1, SW2, and SW3 are defined in the SOS User Application Parameter Tape.

The following sample sequence of code may then be executed to examine the flag bits in each table entry:

```
        LDA  2, .CT      ;AC2 → COMMAND TABLE
        LDA  1, MASK     ;AC1 = MASK WORD
LOOK:   LDA  0, CTSW, 2  ;AC0 = SWITCH WORD
        AND  1, 0, SZR   ;SWITCH "A" OR "N" ON?
        JMP  OUT         ;YES
        INC  2, 2        ;NO
        INC  2, 2        ;LOOK AT NEXT ENTRY
        JMP  LOOK
OUT:    .
        .
        .
MASK:   SWA+SWN          ;"A" AND "N" SWITCH BITS
.CT:    CT               ;ADDRESS OF COMMAND
                         ;TABLE
```

The calling procedure necessary to invoke the program is the following:

```
JSR  .CTB
arg1
arg2
arg3
arg4
arg5
return location
```

The arguments passed are the following:

arg1 The byte address of any prompt message to be typed on the $TTO. If this argument is a -1, then no message is typed.

arg2 The byte address of the input buffer used to read the command line.

arg3 The maximum byte size of the input buffer. If this length is exceeded through use of continued input lines, the read is terminated and reissued beginning with the prompt message.

COMMAND TABLE BUILDER (CTB) (Continued)

arg4 The beginning address of the command table to be created by the program. The maximum size of this table is determined by the maximum input buffer length. The table may have at most one entry for every word in the input buffer. This is the case in which every other input character is a string separator.

arg5 The beginning address of the translate table to be used to interpret the string switches.

The arguments returned are:

1. The new Command Table, formatted as in the above description.

2. The Command Table entry count in AC0.

Accumulators 1 and 2 are unchanged; accumulator 3 contains USP. All channels will be closed. Channels 0 and 1 are used to read and write the command line and must therefore be closed when the program is entered. The SOS user stack must have been supplied and initialized as shown in the preceding section.

A sample program which calls the CTB program and repeats the call if less than three unique strings are input is coded in the next column.

COMMAND TABLE BUILDER (CTB) (Continued)

```
        .EXTN .CTB          ;.CTB MUST BE DECLARED
          :
        LDA  1, CN3         ;LOAD COMPARAND
START:  JSR  @CTB           ;CALL CT BUILDER
        PRMPT*2             ;BYTE ADDRESS OF
                            ;PROMPT MESSAGE
        IBUF*2              ;BYTE ADDRESS OF INPUT
                            ;BUFFER
        300                 ;MAXIMUM COMMAND
                            ;LINE LENGTH
        CT                  ;COMMAND TABLE ADDRESS
        STRT                ;TRANSLATE TABLE ADDRESS
        SUBZ# 1, 0, SNC     ;3 OR MORE TABLE ENTRIES?
        JMP  START          ;NO.. RE-PROMPT
        STA  0, T0, 3       ;YES. SAVE COUNT IN STACK
          .
          .
          .
CN3:    3
CTB:    .CTB
PRMPT:  .TXT  /** <15> /
IBUF:   .BLK  140
CT:     .BLK  140
STRT:   "A                  ;ONLY 4 SWITCHES RECOG-
        "B                  ;NIZED
        "C
        "D
        -1
```

Four utility programs are included in SOS:

SOS Text Editor
SOS Extended Assembler
SOS Library File Editor
SOS SYSGEN

For systems with at least one cassette or magnetic tape drive, three additional utility programs are included in SOS:

SOS Relocatable Loader
SOS Command Line Interpreter
Core Image Loader/Writer

All except the SOS Text Editor and the Core Image Loader/Writer are supplied in both relocatable binary and absolute binary forms. The SOS Text Editor is supplied in relocatable binary form only, and the Core Image Loader/Writer is supplied in absolute binary form only. Relocatable binary forms must be configured with I/O support as described in Chapter 6, Configuring SOS Utility Programs. Absolute binary forms are preconfigured as described in Chapter 1.

Details of operation of the SOS Text Editor, SOS Extended Assembler, SOS SYSGEN, and SOS Relocatable Loader are explained in Appendix A. Details of operation of the SOS Library File Editor are contained in the Library File Editor manual, 093-000074. Details of operation of the SOS Command Line Interpreter and Core Image Loader/Writer are explained later in this chapter.

PAPER TAPE OPERATION

SOS utility programs can be operated using the binary loader to load absolute binary paper tapes from the high-speed paper tape reader or the teletype paper tape reader. The procedures for loading programs in this manner are described in Section 2.8 of "How to Use the Nova Computers."

MASTER CASSETTE OR MAGNETIC TAPE OPERATION

Systems with one or more cassette or magnetic tape drives are provided with two special utility programs: a command line interpreter (CLI) and a core image loader/writer. The core image loader/writer transfers core image files between core and either cassette or magnetic tape. The CLI implements mnemonic loading of other utility programs into core from a master cassette or magnetic tape reel. In addition, the CLI performs a number of basic file maintenance tasks for the user.

Master Cassette or Magnetic Tape Reel

A master cassette or magnetic tape reel has the following format:

File 0: Core Image Loader/Writer
File 1: Command Line Interpreter (CLI)
File 2: Text Editor
File 3: Extended Assembler
File 4: Extended Relocatable Loader
File 5: Library File Editor
File 6: SYSGEN

To be used as a master, this cassette or magnetic tape reel must be mounted on cassette drive 0 (CT0) or magnetic drive 0 (MT0), respectively. Each utility program on the master reel must be in executable core image form. Procedures for generating a master reel are discussed in Chapter 6, Configuring SOS Utility Programs.

CORE IMAGE LOADER/WRITER

The core image loader/writer is a utility program that performs two functions: it loads core image files from cassette or magnetic tape into core and produces core image files on cassette or magnetic tape from the contents of core. There are 2 versions of the core image loader/writer--one for use with cassette drives and another for use with magnetic tape drives. The cassette version works only with cassettes and the magnetic tape version works only with magnetic tape. Both versions when loaded occupy the last 400_8 locations in core.

Bootstrap Procedure

The core image loader/writer can be bootstrap loaded from file 0 of the master cassette or magnetic tape reel. The master reel must be loaded on cassette or magnetic tape unit 0 and must be rewound manually. This can be done by pressing the REWIND button on the drive unit.

For machines without the Program Load option, deposit 060134 (for cassette units) or 060122 (for magnetic tape units) into location 376_8, and deposit 000377 into location 377_8. Press the console switches for RESET and then START.

For machines with the Program Load option, set the data switches on the console to 100034 (for cassette units) or 100022 (for magnetic tape units), press RESET and then PROGRAM LOAD.

The core image loader/writer is read into page zero initially and then relocates itself to the high end of

Bootstrap Procedure (Continued)

memory. At the end of the relocation process, the loader outputs a prompt (#) on the teletype. This prompt indicates that the core image loader is ready to accept a command. Whenever the core image loader/writer is resident in core, the core image loader may be restarted by setting the data switches to the address of the last location in memory, pressing RESET, and then pressing START. The core image writer can be started by setting the data switches to the address of the next to last location in core, pressing RESET and then pressing START.

Core Image Loader Operation

Having issued the # prompt on the teletype, the core image loader waits for an operator response of a device number (0-7) and a file number (0-99) separated by a colon. Device 0 need not be specified. For example:

#2:7↵ (unit 2, file 7)

#4↵ (unit 0, file 4)

The indicated cassette or magnetic tape file is loaded from the specified device into memory starting at address 0. If data switch 0 on the console is set (up), the loader will halt after loading is complete. If switch 0 is reset (down), the loader will transfer control to the address at location 405_8 at the end of the loading process, unless this location contains -1. If location 405_8 contains -1, the loader will simply halt.

If the core image loader encounters a non-recoverable error while trying to load a core image file, it will type

*ERR

and halt with the cassette or magnetic tape status word in AC0. The following list describes the error conditions assigned to each bit in the status word.

Bit	Meaning
1	Data late (perhaps due to a long indirect access chain or a faster device preempting the channel.)
3	Illegal command
5	Lateral parity error in a word
6	Addressed tape is beyond the EOT marker
8	Addressed tape is at load point
10	Bad tape (e.g., data is found in an interrecord gap)
13	Unit is write locked
14	Odd number of bytes detected in a read or write attempt

Core Image Loader Operation (Continued)

If rewinding and substituting a fresh cassette or magnetic tape does not cure an error condition, a hardware malfunction is indicated; run the appropriate diagnostic program.

Core Image Writer Operation

The core image writer operates in a manner similar to that of the core image loader. When the core image writer is started it outputs a # prompt and waits for specification of a device number and a file number separated by a colon. Unit 0 need not be specified. After the file has been specified, the core image writer will request specification of the upper core address (NMAX) to be written onto tape. It does this by typing:

NMAX:

on the teletype. The operator must then respond with the highest core address (in octal) whose contents he wants written into the core image cassette or magnetic tape file. Upon detection of a non-recoverable error, the core image writer proceeds in the same way as the core image loader.

COMMAND LINE INTERPRETER (CLI)

The Command Line Interpreter (CLI) is a utility program which performs certain file maintenance chores for the user and implements mnemonic loading of other utility programs from a master cassette or magnetic tape. The CLI accepts commands typed by the operator on the teletype. When it is ready to receive a command, the CLI types on the teletype a prompt consisting of R followed by a carriage return.

In order to fully utilize the CLI, the core image loader/writer should be resident in core, and a master cassette or magnetic tape should be loaded on cassette drive 0 (CT0) or magnetic tape drive 0 (MT0). A master cassette or magnetic tape has the following format:

File 0: Core image loader/writer (bootstrappable)
File 1: Command Line Interpreter (CLI)
File 2: Text Editor
File 3: Extended Assembler
File 4: Extended Relocatable Loader
File 5: Library File Editor
File 6: SYSGEN

The CLI can be loaded from the master cassette or magnetic tape reel using the core image loader/writer. Many of the commands the CLI accepts result in the CLI being overwritten in core. After these commands, return can be made to the CLI only by loading it again.

COMMAND LINE INTERPRETER (CLI) (Continued)

The following are descriptions of the commands accepted by the CLI:

Load the Assembler

Format: ASM

This command causes file 3 from the master cassette unit (CT0) or magnetic tape unit 0 (MT0) to be loaded. The effect is the same as for a 0:3 command to the core image loader. The CLI is overwritten in core by the extended assembler. If data switch 0 on the console is reset (down) the assembler will type the prompt ASM on the teletype when the loading is complete.

Load an Absolute Binary Paper Tape

Format: BLDR $PTR or
BLDR $TTR

This command will load an absolute binary tape with the binary block loader (contained in the CLI), using either the high-speed paper tape reader ($PTR) or the teletypewriter reader ($TTR). The loading will overwrite both the core image loader/writer and the CLI in core.

Load a Cassette File

Format: CTx:yy

where: x is the cassette drive number (0-7) and
yy is the file number (0-99) to be loaded from that drive.

Core image file yy on cassette drive x is loaded into core, overwriting the CLI. If no file yy exists on cassette drive x the error message FILE NON-EXISTENT is printed on the teletype. In systems operating from magnetic tape (the magnetic tape core image loader/writer is used) this command is not valid and results in the error message NON-EXISTENT FILE.

Load the Text Editor

Format: EDIT

File 2 from the master cassette unit (CT0) or magnetic tape unit (MT0) is loaded into core. The effect is the same as for a 0:2 command to the core image loader. The CLI is overwritten in core by the text editor. If data switch 0 on the console is reset (down) the text editor will type the prompt * on the teletype when loading is complete.

Initialize Cassette or Magnetic Tape

Format: INIT devicename

where: devicename is one of CT0, CT1,...CT7, or MT0, MT1,...MT7.

The cassette or magnetic tape unit specified is rewound. If there is no such unit in the system, the error message ILLEGAL FILE NAME will be printed on the teletype.

Load the Library File Editor

Format: LFE

File 5 from the master cassette unit (CT0) or magnetic tape unit (MT0) is loaded into core. The effect is the same as for a 0:5 command to the core image loader. The CLI is overwritten in core by the library file editor. If data switch 0 on the console is reset (down) the library file editor will type the prompt LFE on the teletype when loading is complete.

Make a Save File

Format: MKSAVE absolute-binary-filename output filename

Input from file absolute-binary-filename is converted to a core image (save) file. This core image file becomes file output filename. Possible error messages are:

NOT ENOUGH ARGUMENTS
ILLEGAL FILE NAME
ILLEGAL COMMAND FOR DEVICE
DEVICE IS READ PROTECTED
FILE NON-EXISTENT
CHECKSUM ERROR
PHASE ERROR

Load a Magnetic Tape File

Format: MTx:yy

where: x is the magnetic tape drive number (0-7) and
yy is the file number (0-99) to be loaded from that drive.

Core image file yy on magnetic tape drive x is loaded into core, overwriting the CLI. If no file yy exists on magnetic tape drive x, the error message NON-EXISTENT FILE is printed on the teletype. In systems operating from cassette (the cassette core image loader/writer is used), this command is not valid and results in the error message NON-EXISTENT FILE.

Release Cassette or Magnetic Tape

Format: RELEASE devicename

where: devicename is one of CT0, CT1,...,CT7, or MT0, MT1, ...,MT7

The cassette or magnetic tape unit specified is rewound. If there is no such unit in the system, the error message ILLEGAL FILE NAME will be printed on the teletype.

Load the Relocatable Loader

Format: RLDR

File 4 from the master cassette unit (CT0) or magnetic tape unit (MT0) is loaded into core. The effect is the same as for a 0;4 command to the core image loader. The CLI is overwritten in core by the relocatable loader. If data switch 0 on the console is reset (down) the relocatable loader will type the prompt RLDR on the teletype when it has been loaded.

Load SYSGEN

Format: SYSG

File 6 from the master cassette unit (CT0) or magnetic tape unit (MT0) is loaded into core. The effect is the same as for a 0;6 command to the core image loader. The CLI is overwritten in core by SYSGEN. If data switch 0 on the console is reset (down) SYSGEN will type the prompt SYSG on the teletype when loading is complete.

Transfer a File

Format: XFER source filename destination filename

XFER/A source filename destination filename

This command causes file source filename to be transferred to file destination filename. Transfer is made in binary mode unless the switch /A is given. When switch /A is given, the source file data is interpreted as even parity ASCII. Special action is taken at the destination file depending upon the nature of the destination device.

The relocatable binary versions of SOS utility programs can be used to produce executable versions of these programs configured with arbitrary I/O support. This allows the user to produce executable utility programs with the necessary I/O support for the devices in his particular system, without wasting core space for device drivers that never are used.

The process of configuring a utility program can be divided into two main tasks: producing a trigger which specifies the desired I/O support, and performing a relocatable load of the trigger, the appropriate SOS libraries, and the relocatable binary form of the utility program.

PRODUCING A TRIGGER (FOR ALL SOS SYSTEMS)

Triggers are produced by the SYSGEN program. The SYSGEN program accepts from the teletype a command line containing device driver entry and outputs a file (the trigger) containing external normal references to the named device drivers. These external normal references cause the named device drivers to be loaded from SOS libraries when the trigger precedes those libraries as input to the relocatable loader.

The first step in producing a trigger is to load and start the SYSGEN program. This can be done using the binary loader to load an absolute binary SYSGEN paper tape (091-000070, 091-000071, or 091-000074). SYSGEN can also be loaded from cassette or magnetic tape using the core image loader/writer.

When the SYSGEN program is started, it outputs to the teletype the prompt

SYSG

and waits for the user to type a command line. The SYSGEN command line has the following format:

(SYSG) driver$_1$ ... driver$_n$ .DSI [.CTB]
output-device/O [trigger-name/T]

where: driver$_i$ is a device driver entry symbol.

.DSI is the RDOS to SOS interface program and must appear in every SYSGEN command line.

.CTB is the optional command table builder. The .CTB must be specified only for triggers to be used in configuring the assembler, relocatable loader, the CLI, or the SYSGEN program itself.

output-device is the name of the device to which the user wishes the trigger to be output. This device name must be followed by the /O switch.

trigger-name is the optional name of the trigger. If this name is omitted, by default the trigger will be named .MAIN. If the name is present, it must be followed by the /T switch.

PRODUCING A TRIGGER (Continued)

Driver Entry Symbol	Device
.CDRD	card reader
.CTAD	cassette unit 0 only
.CTU1	cassette units 0 and 1
.	.
.	.
.	.
.CTU7	cassette units 0 through 7
.PTPD	high-speed paper tape punch
.PTRD	high-speed paper tape reader
.LPTD	line printer
.MTAD	magnetic tape unit 0 only
.MTU1	magnetic tape units 0 and 1
.	.
.	.
.	.
.MTU7	magnetic tape units 0 through 7
.PLTD	plotter
.STTY	teletype printer and keyboard only

If .STTY is not specified, a full teletype driver (including teletype paper tape reader and punch) will be loaded by default.

For example, to produce a trigger on paper tape for the assembler to provide support for a full teletype, high-speed reader and punch and cassette units 0 and 1, the following SYSGEN command line could be typed:

(SYSG) .PTRD .PTPD .CTU1 .DSI .CTB $PTP/O

The trigger produced will have the title .MAIN since no trigger name was specified. The trigger will be output by the high-speed paper tape punch.

After the trigger has been output, the SYSGEN program will again type its prompt (SYSG) and wait for another command. If more than one utility program is to be configured requiring different triggers, it is most convenient to generate all the necessary triggers before overwriting the SYSGEN program in core.

A discussion of the SYSGEN program, including its associated error messages, is included in Appendix A.

CONFIGURATION PROCEDURE (FOR PAPER TAPE SYSTEMS)

The following is a step by step description of a procedure for configuring any SOS utility program (except the assembler) using only paper tapes as input files. The

CONFIGURATION PROCEDURE (FOR PAPER TAPE SYSTEM) (Continued)

end product is an absolute binary paper tape. Before starting this procedure, the trigger to be used should be generated on paper tape. Each typed command in this procedure must be terminated by a carriage return.

1. Using the binary loader, load the extended relocatable loader (tape 091-000038).

2. Mount the trigger in the teletype reader and type 1, or in the high-speed paper tape reader and type 2.

3. Mount the SOS library (099-000010) in the teletype reader and type 1, or in the high-speed paper tape reader and type 2. If the trigger specifies support for cassette or magnetic tape drives, perform this process first for the SOS cassette library (099-000041) or the SOS magnetic tape library (099-000042), and then the SOS library.

4. Mount the relocatable binary version of the program to be configured in the teletype reader and type 1, or in the high-speed paper tape reader and type 2.

5. Type 5 and note the value of NMAX output by the relocatable loader on the teletype; this number will be used in Step 12.

6. Mount the relocatable binary punch program (089-000080) on the teletype reader and type 1, or on the high-speed paper tape reader and type 2.

7. Type 6 and note the value of RBFP output by the relocatable loader on the teletype; this number will be used in Step 10.

8. Type 8 to terminate the loading process.

9. Examine the contents of location 405_8; note this value for use in Step 13.

10. Enter RBFP (from Step 7) into the data switches on the computer console, press RESET and then press START.

11. Type 0H for output on the teletype punch or 1H for output on the high-speed paper tape punch.

12. Type 1,nmaxP where nmax is the value of NMAX noted in Step 5.

13. Type cE where c is the value (in octal) of the contents of location 405_8 noted in Step 9.

CONFIGURING THE ASSEMBLER (FOR PAPER TAPE SYSTEMS)

The following is the procedure for configuring the SOS extended assembler. It is similar to the preceding procedure for the other SOS utility programs.

1. Using the binary loader, load the extended relocatable loader (tape 091-000038).

2. Mount the trigger in the teletype reader and type 1, or in the high-speed paper tape reader and type 2.

3. Mount the SOS library (099-000010) in the teletype reader and type 1, or in the high-speed paper tape reader and type 2. If the trigger specifies support for cassette or magnetic tape drives, perform this process first for the SOS cassette library (099-000041) or the SOS magnetic tape library (099-000042), respectively, and then mount the SOS library.

4. Mount the relocatable binary version of the SOS extended assembler (089-000106) in the teletype reader and type 1, or in the high-speed paper tape reader and type 2.

5. Enter 016500_8 in the data switches on the computer console and type 3.

6. Mount the relocatable binary punch program (089-000080) on the teletype reader and type 1, or on the high-speed paper tape reader and type 2.

7. Type 6 and note the value of RBFP output by the relocatable loader on the teletype; this number will be used in Step 13.

8. Type 8 to terminate the loading process.

9. Press CONTINUE on the computer console to start the assembler.

10. Mount the extended assembler command definitions tape (090-000890) in the teletype reader and type 0 $TTR, or in the high-speed paper tape reader and type 0 $PTR.

11. When the assembler halts, examine AC0 and note its contents (NMAX); this value will be used in step 15.

12. Examine the contents of location 405_8; note this value for use in Step 16.

13. Enter RBFP (from step 7) into the data switches on the computer console, press RESET, and then press START.

CONFIGURING THE ASSEMBLER (FOR PAPER TAPE SYSTEMS) (Continued)

14. Type 0H for output on the teletype punch or 1H for output on the high-speed paper tape punch.

15. Type 1,nmaxP where nmax is the value of NMAX noted in step 11.

16. Type cE where c is the value (in octal) of the contents of location 405_8 noted in Step 12.

PRODUCING A MASTER REEL

The following is a step by step description of a procedure for configuring SOS utility programs and at the same time producing a master cassette or magnetic tape reel. The assumption is made that only one cassette or magnetic tape drive is available and that the programs used are available only on paper tapes. Two cassette and magnetic tape reels are required for this procedure; when mounted, cassette reels should have their file protect tabs in place and magnetic tapes reels should contain a write permit ring so that files may be written on them. Before starting this procedure, the triggers to be used should have been generated on paper tapes. Each typed command in this procedure must be terminated by a carriage return.

1. Using the binary loader, load and start the absolute binary version of the command line interpreter (091-000072 for cassette; 091-000075 for magnetic tape). When started the CLI will type the prompt R on the teletype.

2. Mount the cassette or magnetic tape reel that is not to become the master reel on unit 0.

3. Mount the absolute binary version of the SOS relocatable loader (091-000073) for cassette; 091-000076 for magnetic tape) in the high-speed paper tape reader (or the teletype reader) and type the command line:

MKSAVE $PTR CT0:0 (for cassette)

MKSAVE $PTR MT0:0 (for magnetic tape)

If the teletype reader is used, substitute $TTR for $PTR.

4. When the CLI again types the R prompt, remove the first reel and mount the master reel on unit 0.

5. Mount the core image loader/writer (091-000067 for cassette; 091-000068 for magnetic tape) in the high-speed paper tape reader (or teletype reader) and type the command line:

PRODUCING A MASTER REEL (Continued)

5. (Continued)

MKSAVE $PTR CT0:0 (for cassette)

MKSAVE $PTR MT0:0 (for magnetic tape)

If the teletype reader is used, substitute $TTR for $PTR.

6. Rewind the master reel by pressing REWIND on the drive unit and bootstrap the core image loader. The bootstrap process was explained in Chapter 5.

7. After the core image loader has output its prompt (#), remove the master reel and mount the other reel on unit 0.

8. Set data switch 0 on the computer console to the zero position (down) and type 0. The core image loader will load the relocatable loader into core.

9. After the relocatable loader has typed its prompt (RLDR) on the teletype, mount the master reel on unit 0.

10. Type the following command line to the relocatable loader:

CT0:1/S $PTR/4 (for cassette)

MT0:1/S $PTR/4 (for magnetic tape)

If the teletype reader is to be used, substitute $TTR for $PTR. As the relocatable loader requests them, mount the following paper tapes in order:

1. The trigger to be used for the command line interpreter.
2. The SOS cassette library (099-000041) for cassette or the SOS magnetic tape library (099-000042) for magnetic tape.
3. The SOS library (099-000010).
4. The relocatable binary version of the command line interpreter (089-000121).

The loader will produce a core image file of the CLI on the master cassette and will type OK when finished.

11. Restart the core image loader by setting the address of the last location in core into the data switches on the computer console, pressing RESET, and then pressing START. Repeats Steps 7, 8, and 9. Repeat Step 10 making the following substitutions:

PRODUCING A MASTER REEL (Continued)

11. (Continued)

1. CT0:2 or MT0:2 instead of CT0:1 or MT0:1 respectively, in the relocatable loader command line.
2. The trigger for the SOS text editor instead of the trigger for the CLI.
3. The relocatable binary version of the SOS text editor (089-000104) instead of that for the CLI.

12. Restart the core image loader. Repeat steps 7, 8, and 9. Repeat Step 10 making the following substitutions:

1. CT0:3 or MT0:3 instead of CT0:1 or MT0:1 respectively, in the relocatable loader command line.
2. The trigger for the SOS extended assembler instead of the trigger for the CLI.
3. The relocatable binary version of the SOS extended assembler (089-000106) instead of that for the CLI.

13. Press CONTINUE on the computer console to start the assembler.

14. Mount the extended assembler command definitions tape (090-000890) in the high-speed paper tape reader (or teletype reader). After the assembler has typed its prompt (ASM), type the command line:

0 $PTR

If the teletype reader is used, substitute $TTR for $PTR. The assembler will perform a one pass assembly of the command definitions and will then halt.

15. Examine the contents of AC0 and note them for use in Step 16.

16. Start the core image writer by setting the address of the next to last location in core into the data switches on the computer console, pressing RESET, and then pressing START. After the core image writer types its prompt (#), type 3. The core image writer will type NMAX:. Respond by typing (in octal) the value noted in Step 15. The core image writer will produce a core image file on the master reel and type its prompt (#) when finished.

PRODUCING A MASTER REEL (Continued)

17. Restart the core image loader. Repeat Steps 7, 8, and 9. Repeat Step 10 making the following substitutions:

1. CT0:4 or MT0:4 instead of CT0:1 or MT0:1 respectively, in the relocatable loader command line.
2. The trigger for the relocatable loader instead of the trigger for the CLI.
3. The relocatable binary version of the SOS relocatable loader (089-000120) instead of that for the CLI.

18. Restart the core image loader. Repeat Steps 7, 8, and 9. Repeat Step 10 making the following substitutions:

1. CT0:5 or MT0:5 instead of CT0:1 or MT0:1 respectively in the relocatable loader command line.
2. The trigger for the library file editor instead of the trigger for the CLI.
3. The relocatable binary version of the library file editor (089-000081) instead of that for the CLI.

19. Restart the core image loader. Repeat Steps 7, 8, and 9. Repeat Step 10 making the following substitutions:

1. CT0:6 or MT0:6 instead of CT0:1 or MT0:1 respectively in the relocatable loader command line.
2. The trigger for SYSGEN instead of the trigger for the CLI.
3. The relocatable binary version of SOS SYSGEN (089-000122) instead of that for the CLI.

This appendix provides details of the operation of four SOS utility programs: SOS extended assembler, SOS relocatable loader, SOS text editor, and SOS SYSGEN.

The conventions used in defining command line formats are:

All upper case letters represent valid command line elements.

Items in a command line printed in lower-case indicate either command information or device names which must be supplied.

Elements enclosed in brackets ([ ]) are optional. Stacked items indicate alternate choices.

The ellipsis (...) is used to indicate that preceding command information may be repeated if desired.

Command names enclosed in parentheses are prompts output by system programs. These prompts are not significant command line elements, but merely indicate that the system program is loaded and ready to accept command line input.

ASSEMBLER

Format: (ASM) 0 file name$_1$ ...

(ASM) 1 file name$_1$ ...

(ASM) 2 file name$_1$ ...

Purpose: To assemble one or more ASCII source files. Output may be an absolute or relocatable binary file, with an optional listing file. Input files are assembled in the order specified in the command line. A cassette or magnetic tape unit may not be used for both input and output, nor may it be used for more than one output file. A cassette or magnetic tape unit may be used for more than one input file.

In the above formats, 0, 1, and 2 are keys describing the number of passes required. Global switches listed below modify the key in a given command line. Action taken by the assembler depends upon the key specified in the command line:

0 - Perform pass one on the specified input source file(s), then halt with the highest symbol table address (SST) in AC0. Normally the source file is a Command Definition tape. The user may then invoke the core image writer to preserve a copy of this assembler on any cassette file.

1 - Perform pass one and pass two on the specified input files, producing the specified binary and listing files. At the completion of pass two, the assembler outputs a new prompt, ASM, and awaits a new command line.

2 - Perform pass two only on the specified input files producing the specified binary and listing files. The symbol table used for this pass is that produced by the most recently executed pass one assembly. At the completion of this pass, the assembler outputs a new prompt, ASM, and awaits a new command line.

ASSEMBLER (Continued)

Purpose (Continued)

Switches

Global:

/E suppress assembly error messages normally output to the $TTO.

/T suppress the listing of the symbol table.

/U include local (i.e., user) symbols in the binary output file.

Local:

/B relocatable or absolute binary file is output on the given device.

/L any output device to which the listing is directed.

/N any input file which is not to be listed on pass 2.

/P pause before accepting a file from a device. The message:

PAUSE - NEXT FILE, devicename

is output by the assembler which waits until any key is struck on the teletypewriter console.

/S skip this source file during pass two.

/n repeat the given input source file n times, where n is a digit from 2 to 9.

Errors:

NO. END (No . END statement in any source file.)

I/O ERROR nn nn is an error code. The following codes and their meanings are possible:

1 ILLEGAL FILE NAME
7 ATTEMPT TO READ A READ-PROTECTED FILE.
10 WRITE-PROTECTED FILE
12 NON-EXISTENT FILE

ASSEMBLER (Continued)

Examples:

1 CT1:0/B $LPT/L CT0:0/S CT0:1 CT0:2/N ↵

causes a two-pass assembly to be executed using cassette files 0, 1, and 2 on unit 0 as input files. A binary file is produced on unit 1, file 0, and a listing file is printed on the line printer. On pass 2, input file CT0:0 is skipped, and input file CT0:2 is not listed.

2/U CT0:13 CT0:14 CT0:18 CT0:8 CT1:1/B CT2:0/L ↵

causes the second pass of an assembly to be executed using input files 13, 14, 18 and 8 (in that order) on cassette unit 0. The binary, containing user symbols, is produced on file 1 of cassette unit 1, and the listing is produced on file 0 of cassette unit 2.

1/E CT0:16 CT0:17 CT1:0 CT1:1 $LPT/L ↵

causes a two-pass assembly to be executed on input files CT0:16, CT0:17, CT1:0, and CT1:1 with a listing printed on the line printer. Error messages normally output to the $TTO are suppressed, and no binary file is produced.

1 CT0:0/S CT0:0/P CT1:1/B ↵

causes a two-pass assembly to be performed on two files, both of which will be read from unit 0. The first file is a parameter list which will be read during the first pass only. After this parameter list is read, the pause message is output, and a new file is placed in cassette unit 0. The first file of this new reel is scanned for both pass 1 and pass 2 to complete the assembly. File 1 of unit 1 receives the binary output; no listing is produced.

RELOCATABLE LOADER

Format: (RLDR) file name$_1$ ...

Purpose: To produce from relocatable binary files an executable core-resident file and a core-image (or save) file. Both files start at address zero. The symbol table is included in these files only if the debugger has been loaded, and then the table always follows the program immediately.

The debugger provided for cassette and magnetic tape systems is Extended Debug III. This debugger is found on relocatable binary tape 089-000073.

RELOCATABLE LOADER (Continued)

Purpose: (Continued)

The loading of user ZREL begins at location 50_8, and user NREL loading starts at location 440_8. Locations 400-437 contain the User Status Table. Location 405_8 contains the starting address of the loaded program.

The maximum size of each loaded program cannot exceed the maximum core address less 1325_8. The relocatable loader will type symbol tables for programs whose size exceeds that maximum value, yet the loading process will not be completed for such programs.

Upon completion of each successful load, the message "OK" is output and the system halts with the loaded program in core.

Switches

Local:

/A causes a listing of the symbol table with symbols ordered alphabetically. (The local switch /L must also be given to define an output listing device.)

/L causes a listing of the symbol table on the output file or device whose name precedes the switch. Symbols in the table will be ordered numerically by symbol value.

/N set current NMAX (the starting load address) forcing the file name following the switch to the absolute address immediately preceding this switch.

/P pause before opening this file. The message: PAUSE-NEXT FILE, devicename is output by the loader which waits until any key is struck on the teletypewriter console.

/S output the save file to the device (either cassette or magnetic tape unit) whose name precedes this switch.

/U load user symbols appearing within the file whose name precedes this switch.

/n load the preceding file n times, where n is a digit from 2 to 9.

Errors:

NO INPUT FILE SPECIFIED.
NO SAVE FILE SPECIFIED (i.e., no core image output device has been specified with a /S switch.)
SAVE FILE IS READ/WRITE PROTECTED (The save file device must permit both reading and writing: only cassette and magnetic tape units are permitted as save file devices.)

RELOCATABLE LOADER (Continued)

Errors: (Continued)

I/O ERROR nn (nn is an error code). The following codes and their meanings are possible:

1 - ILLEGAL FILE NAME
7 - ATTEMPT TO READ A READ-PROTECTED FILE
10 - WRITE-PROTECTED FILE
12 - NON-EXISTENT FILE

Example:

The command line:

$TTO/L/A CT2:0/S $PTR CT1:6 16500/N CT1:0 ↵

causes the relocatable loader to load programs into core from the paper tape reader ($PTR), file 6 on cassette drive 1 (CT1:6), and file 0 on cassette drive 1 (CT1:0). The program from CT1:0 is loaded starting at address 16500_8 (16500/N). A core image file containing the loaded programs is written to file 0 of cassette unit 2 (CT2:0/S). An alphabetically ordered symbol table is output on the teletype printer ($TTO/L/A).

SYMBOLIC TEXT EDITOR

Format: EDIT

Purpose: To load the symbolic text editor by means of the CLI core image load command. The operation of the text editor is as described for the DOS editor in the Nova Text Editor Manual, 093-000018, except for the following.

When the H command is issued, control returns to the beginning of the editor program just as though the user had depressed RESET, then START with 377 in the console data switches.

As with the RDOS editor, CTRL A interrupts the activity of the editor. Also as with the RDOS editor, all output files must be closed with the GC command. Execution of the P or E commands does not insure that the final input page will be written onto the output file.

Switches:

Not applicable.

SYMBOLIC TEXT EDITOR (Continued)

Errors:

See errors given for the Core Image Loader command and in the Text Editor manual.

Example:

After loading and activating the text editor, input and output files must be assigned. For example:

GRCT2:3$GWCT1:3$$

causes a source file to be read from file 2 of cassette unit 3, and an output file to be written as file 3 of unit 1.

SYSGEN

Format: (SYSG) $driver_1 \ldots driver_n$ .DSI [.CTB]
output-device/O [trigger-name/T] ↵

Purpose: The SYSGEN program generates triggers for use in configuring SOS utility programs. Each driver in the command line is an entry symbol for a program in one of the SOS libraries. For each of these entry symbols included in the command line, an external normal reference to the program is included in the trigger. The trigger is entirely made up of these external normal references, including external normal references for .DSI and .CTB (if specified).

Switches:

/O the preceding specifies the output device

/T the preceding is to be assigned as the title of the trigger

Errors:

NOT ENOUGH ARGUMENTS (At least two are needed.)
OUTPUT FILE WRITE PROTECTED, FILE: filename
NO OUTPUTFILE SPECIFIED
ILLEGAL SYMBOL NAME: symbolname (Invalid character in command line.)
FILE DOES NOT EXIST, FILE: filename
UNEXPECTED SYSTEM ERROR (Computer halts with the system error code in AC2).

Example:

.PTRD .PTPD .MTU4 .STTY .DSI MT1:3/O HORSE/T ↵

would cause the SYSGEN program to output a

SYSGEN (Continued)

Example: (Continued)

trigger named HORSE containing external normal references to the driver routines for the high-speed paper tape reader, high-speed paper tape punch, magnetic tape unit 4, and teletype keyboard and printer, and to the RDOS to SOS interface program (.DSI).

APPENDIX B - ADDING DEVICE HANDLERS TO SOS

This appendix is intended to ease the task of adding a special device driver to the SOS library. There are really two levels of SOS compatibility available to the user who is incorporating additional devices:

1. A level that permits servicing of an interrupt from one or more special devices, while selectively enabling interrupts from other devices.

2. A level that permits complete control of the device via standard SOS commands.

The complexity of the required user program increases with the level. Level 1 requires a Device Control Table (DCT) for each device and a Device Priority Table. Level 2 requires a DCT, a Device Priority Table, and a Channel-Number-to-Device Map. All of these tables, as well as the associated driver code, may be assembled in one relocatable binary program. This program may then be included in the relocatable load module along with the SOS library. The SOS Parameters and the RDOS User Parameters may be included in the assembly to assure the proper definition of the required tables.

The critical requirements for these levels are:

Level 1: Provide a means of clearing the device on system initializations and resets, and provide interrupt save storage compatible with the SOS scheme.

Level 2: In additon to the requirements for Level 1, provide a SOS Channel-Number-to-Device Control Table link and include the appropriate subset of I/O Dispatch routines: open, close, line, and sequential.

If the tables are correctly defined in the user program, the SOS MAIN program will perform these critical functions, using the supplied information.

No source level changes to the SOS library programs are necessary. The links to the user supplied tables currently exist in the SOS MAIN program in the form of unresolved external declarations. If these unresolved externals become resolved in the relocatable load module, then SOS assumes the presence of an additional device or devices. The user must be careful, therefore, not to resolve these externals inadvertently with his own global symbols.

The remainder of this appendix describes the mechanisms that may be used to achieve Level 1 or Level 2 SOS compatibility with optional devices.

DEVICE CONTROL TABLE (DCT)

Each SOS device requires a control table. Although some elements of the table may not be used by an added driver, the table must be defined exactly as in the following description, so that the critical elements reside at the correct displacements. This table requires 33 octal locations (displacements 0-32 from the DCT layout description in the SOS Parameter Tape). These displacements and their meanings are as follows:

Equivalence	Displacement	Meaning
0	DCTCD	The octal device code. Must be assembled into the table.
1	DCTMS	The mask of all lower priority devices, including this device. This mask is used to disable interrupts from all lower priority devices while processing an interrupt from this device. This mask should reflect the priorities established by the Device Priority Table (page B-3). The mask bits are defined in the SOS parameter tapes, and must be assembled into the table.
2	DCTCH	The active device characteristics from the DOS User Parameter Tape. This word is derived by masking the complement of the user's AC1 on a .OPEN command with the device's fixed characteristics (see DCTFC). The device characteristics must be assembled into the table; they are not referenced for Level 1 devices.
3	DCTLK	The link to the next priority device, a pointer to its control table. This word is initialized by a .SYSI. The priorities are established by SOS table .OPPP.
4	DCTIS	The address of the interrupt service routine. The address must be assembled into the table. (See SOS Interrupt Handling, page B-4.)
5	DCTIL	The interrupt frame links. This points to the DCT of the last interrupted device. This word is maintained by the SOS interrupt service routine.

DEVICE CONTROL TABLE (DCT) (Continued)

Equivalence	Displacement	Meaning
6	DCTDT	The Command Dispatch Table address for this device. The Command Dispatch Table must be ordered in the following manner: 0 - open routine address 1 - close routine address 2 - read/write sequential routine address 3 - read/write line routine address Any of the before mentioned functions that are illegal for a device should contain a -1 in their location. The address must be assembled into the table. See Device Start, Stop, and Dispatch Routines.
7	DCTST	The address of the device start routine. The address must be assembled into the table. See Device Start, Stop, and Dispatch Routines.
10	DCTSP	The address of the device stop routine. The address must be assembled into the table. See Device Start, Stop, and Dispatch Routines.
11	DCTFL	The device flags. These flag bits are maintained by the global SOS subroutines. Three flags are currently defined: DCACT = 1B15 - Device is active (executing I/O). Must be off to perform a SOS reset. DCACP = 1B8 - A keyboard input device may accept a character. DCKMD = 1B0 - A keyboard input device is in echo mode. Echo the input character.

DEVICE CONTROL TABLE (DCT) (Continued)

Equivalence	Displacement	Meaning
12	DCTBS	The size of the device buffer (in bytes for character devices, in words for full word devices). It must be assembled into the table.
13	DCTBF	Buffer first byte (word) address. If the device is a full-word device (DCFWD characteristic), then this location must contain the beginning word address of the buffer. For character devices, this word must contain the beginning byte address of the buffer.
14	DCTBL	Buffer last byte (word) address.
15	DCTIP	Buffer current input pointer. For an output device, this is the byte address at which to store the next byte sent to the device from the user program. For an input device, this is the byte address at which to store the next byte received from the device. This word is maintained by the global SOS subroutines.
16	DCTOP	Buffer current output pointer. For an output device, this is the byte address from which to fetch the next byte for output. For an input device, this is the byte address from which to fetch the next byte requested by the user program. This word is maintained by the global SOS subroutines.
17	DCTCN	Count of active data in the buffer, i.e., bytes not yet sent to the device or bytes not yet moved to the user program for output and input devices respectively. This word is maintained by the global SOS subroutines.

DEVICE CONTROL TABLE (DCT) (Continued)

Equivalence	Displacement	Meaning
20	DCTTO DCTCC	Timeout constant (all input devices). Column counter (all output devices). For input devices this word represents the maximum time interval during which they may have outstanding data following a start pulse. The parameter "SCTIM" defined on the RDOS User Parameter Tape corresponds to a time of 1 millisecond on the Supernova SC. Then, if a device requires 6 milliseconds to timeout, the word can be assembled as: 6*SCTIM For output devices, this word is maintained by the global SOS subroutines.
21	DCTRC DCTLC	Restart constant (all input devices). Line counter (all output devices). For input devices, if the active data count is less than this constant, another start pulse should be sent to the device. This word must be assembled into the table. For output devices, this word is maintained by the global SOS subroutines.
22	DCTAT	Device attributes. Fixed bit settings always returned to the user in AC0 on a .GTATR command. Attributes include attribute protected, permanent, read protected, and write protected. See RDOS User Parameter Tape.
23	DCTFC	Device fixed characteristics. These characteristics, from the DOS User Parameter Tape, always become the active characteristics (DCTCH) after the device is .OPENed, unless they are suppressed by the AC1 mask.
24-32		Device Interrupt Frame. The machine state at the time of a device interrupt is saved in these DCT

DEVICE CONTROL TABLE (DCT) (Continued)

Equivalence	Displacement	Meaning
24-32 (Continued)		locations. The layout of the interrupt frame is as follows:
24	IAC0	Saved AC0.
25	IAC1	Saved AC1.
26	IAC2	Saved AC2.
27	IAC3	Saved AC3.
30	IPC	Program Counter. (Location 0 when the interrupt was taken.)
31	IRLOC	Volatile SOS linkage cell.
32	IMSK	Interrupt enable mask when the interrupt was taken. The carry bit is saved in bit 14 of this word.

DCT displacements 6-23 (DCTDT - DCTFC) are not referenced by Level 1 devices, with the exception of DCTFL, bit 15, which must be off to perform a SOS reset command, and DCTSP, which is executed on .SYSI and .RESET commands. For Level 2 devices, the Dispatch Table (DCTDT) must be defined. Use of the remaining elements depends on the definitions of this table; if any of the global SOS routines are invoked, then any or all of these elements may be referenced.

DEVICE PRIORITY TABLE

This table is referenced by the SOS initialization procedure (system call .SYSI) in order to establish the Device Control Table link words (DCTLK). The order in which the DCT's are linked determines the order in which the SOS devices are searched for a matching code on an interrupt. This table is normally embedded in SOS-MAIN. If the external normal .OPPP in SOS is resolved, however, then a user supplied table is used to establish the links. The SOS-MAIN table is set up as follows:

```
.PTRP
.CDRP
.MTAP
.CTAP
.TTRP
.PTPP
.LPTP
.PLTP
.TTOP
0
```

DEVICE PRIORITY TABLE (Continued)

Each of the symbols ending in "P" is declared an entry in the SOS-MAIN program. The table is always terminated with a zero word. This table reflects a descending priority level of the SOS devices, beginning with the $PTR and ending with the $TTO.

As an example of a user supplied table, consider the addition of devices XXX and YYY, where XXX should be the highest priority device and YYY has a priority less than the MTA but higher than the $TTR. The critical program declarations to achieve this priority scheme would appear as follows:

```
        .ENT    .OPPP
        :
        .NREL
        .EXTN   .PTRP, .CDRP, .TTRP, .PTPP, .LPTP
        .EXTN   .TTOP, .MTAP, .CTAP

.OPPP:  .XXXP
        .PTRP
        .CDRP
        .MTAP
        .CTAP
        .YYYP
        .TTRP
        .PTPP
        .LPTP
        .PLTP
        .TTOP
        0
        :
.XXXP:  XXXDC   :POINTER TO XXX DCT
.YYYP:  YYYDC   :POINTER TO YYY DCT
```

This table must be defined for Level 1 and Level 2 SOS devices.

CHANNEL-NUMBER-TO-DEVICE MAP

This table is referenced by the SOS command dispatch routine. If a SOS I/O command has referenced a channel number outside of the legal SOS range and the SOS external normal, .OPTP has been resolved, then this table is expected to point to a list of Device Control Table addresses. These addresses must be ordered by channel number, beginning at channel number 30 (HCHNO+1 from the SOS parameters), Thus if devices XXX and YYY were being incorporated into SOS for Level 2 compatibility and XXX were assigned to channel 30 and YYY to channel 31, the critical program declarations would be as shown in the next column.

CHANNEL-NUMBER-TO-DEVICE MAP (Continued)

```
        .ENT    .OPTP
        :
        .NREL
.OPTP:  .+1
.XXXP:  XXXDC       ;POINTER TO XXX DCT
.YYYP:  YYYDC       ;POINTER TO YYY DCT
```

This table must be supplied for Level 2 compatibility. The table is expandable from channel number "HCHNO+1" to number 76.

SOS INTERRUPT HANDLING

When an interrupt is taken, a SOS module preserves the the interupted machine state. The DCT interrupt frame (IAC0 - IMSK) is utilized in these procedures, and when the device interrupt handler gains control, the "save" is complete and the mask in the device's Control Table (DCTMS) is active. The following functions are the responsibility of the interrupt routine (DCTIS):

1. Clearing the done flip-flop in the device.
2. Storing/retrieving the next character in the device buffer.
3. Restarting the device when appropriate.
4. Returning to the SOS interrupt module.

The SOS stack mechanism may not be invoked at interrupt processing time. The SOS modules .IBUF and .OBUF may be called, however, since they do not require a stack frame (see SOS global subroutines).

Two simple interrupt routines, one for output device $PTP and one for input device $PTR, illustrate the above points:

```
PTRS:   DIAC    1,PTR       ;RETRIEVE CHARACTER/
                            ;CLEAR DONE
        JSR     @ADRIB      ;STORE CHARACTER IN THE
                            ;DEVICE'S BUFFER
        JMP     .           ;AN IMPOSSIBLE RETURN -
                            ;BUFFER ALREADY FULL
        NIOS    PTR         ;RESTART $PTR. THE
                            ;BUFFER IS NOT YET FULL
        JMP     @.+1        :DON'T RESTART $PTR. THE
                            ;BUFFER BECAME FULL
        .EXTN   .DISM
        .DISM               ;RETURN TO THE SOS
                            ;INTERRUPT MODULE
        .EXTN   .IBUF
ADRIB:  .IBUF               ;ENTRY POINT
```

SOS INTERRUPT HANDLING (Continued)

```
PTPS:   NIOC    PTP     ;CLEAR DONE.
        JSR     @ADROB  :RETRIEVE NEXT CHARAC-
                        :TER FROM THE DEVICE'S
                        ;BUFFER.
        DOAS    1,PTP   ;RESTART DEVICE AND
                        ;SEND THIS CHARACTER IF
                        :THE RETURN CAME HERE.
        JMP     @.+1    ;OTHERWISE DON'T
                        ;RESTART.
        .DISM           ;RETURN TO THE SOS
                        :INTERRUPT MODULE.
        .EXTN   .OBUF
ADROB:  .OBUF           ;ENTRY POINT.
```

DEVICE START, STOP, AND DISPATCH ROUTINES

Device Start Routine

The address of this routine is at displacement DCTST in the DCT. For output devices, this routine should send a start pulse plus the character from AC1. If the device will not interrupt as a result of this action, return to one location beyond the normal return location. Otherwise, return to the normal location. AC3 points to the normal return location. As an illustration, consider the line printer start routine:

```
LPTST:  DOAS    1,LPT   ;START LPT, OUTPUT THE
                        ;CHARACTER.
        SKPBZ   LPT     ;WILL IT INTERRUPT?
        JMP     0, 3    ;YES
        JMP     1, 3    ;NO
```

For input devices, this routine should send a start pulse and return to the normal return location. For example, the $PTR start routine is:

```
PTRST:  NIOS    PTR     ;SEND START PULSE.
        JMP     0, 3    ;RETURN
```

Device Stop Routine

The address of this routine is at displacement DCTSP in the DCT. This routine should simply send a clear pulse and return to the normal return location. Using the $PTR as an example:

```
PTRSP:  NIOC    PTR     ;SEND CLEAR PULSE.
        JMP     0, 3    ;RETURN
```

This routine is executed for each device on any .RESET or .SYSI.

Device Dispatch Routines

The device dispatch table address is at displacement DCTDT in the DCT, as previously described. If the global SOS routines are not invoked for any of the four functions, then the following points should be noted in the dispatch routine:

1. AC3 points to the error return location. Increment by one for a success return.

2. The contents of the user accumulators are:

   AC0 - page zero displacement "CAC0".
   AC1 - page zero displacement "CAC1".
   AC2 - UST displacement "USTA2".

   If an accumulator is being returned to the user, then the appropriate location must be changed.

3. AC2 points to the DCT when the dispatch routine gains control.

If the global SOS routines are used, then they may be invoked directly (by assembling their addresses into the dispatch table.)

SOS LINKAGE

A simple stack mechanism is employed in SOS at the non-interrupt level. This mechanism provides a means of saving the calling routine's accumulators and of operating on variables stored in the stack. No stack mechanism is provided for interrupt processing, but the state of the current stack must be preserved whenever an interrupt is processed. The SOS interrupt dispatch routine performs this service.

The linkage scheme used in SOS makes use of several page zero locations and of a fixed block of core assembled into the SOS module. The size of this core block permits a "depth" of six calls from the common SOS entry point at the start of the program. The page zero locations and equivalences that are used in the linkage mechanism include:

SAVE - (JSR @3) invokes the routine which saves accumulators and updates the stack pointer.

RTRN - (JMP @4) invokes the routine which restores the caller's accumulators and returns to him.

SOS LINKAGE (Continued)

CSP - (Page Zero Location) always points to the stack frame currently in use.

RLOC - (Page Zero Location) used as a temporary by the SAVE routines; may also be used as a temporary by any routine in lieu of allocating a stack frame.

The stack frame is a fixed size with the following displacements defined:

RTLOC - The return location in the current subroutine (location which it last "called".)

AC0 - Contents of accumulator 0 of the current subroutine at the last call which it made.

AC1 - Same as above; accumulator 1.

TMP - Available for use by the current subroutine as a temporary.

OAC0 - Contents of caller's accumulator 0.

OAC1 - Contents of caller's accumulator 1.

OTMP - Caller's temporary.

ORTN - Caller's return location.

Following a SAVE or a RTRN, AC3 always points to the current stack frame, and each of the above locations may be referenced as displacements from it. Otherwise, the CSP may be loaded into an index accumulator in order to reference the locations. (Accumulator 2 is never saved since it usually contains a Device Control Table address and is passed as an argument from subroutine to subroutine.)

The typical procedures executed in using the mechanism are as follows:

1. A routine (B) is called by another routine (A) through the JSR instruction:

```
        JSR   B
                or
        JSR   @ADDRB
        :
        :
ADDRB:  B
```

SOS LINKAGE (Continued)

2. Routine B saves the caller's return location when it begins execution with:

```
STA     3, @CSP
```

3. Routine B may, at any time thereafter, perform a:

```
SAVE
```

to save the caller's accumulators, allocate a stack frame, and update the CSP appropriately.

4. To return to A, routine B simply performs a:

```
RTRN
```

If no frame is required by B, it may save the caller's return location in RLOC:

```
STA     3, RLOC
```

It may then use any accumulator and operate on A's frame (by loading CSP into an index accumulator) if it wishes. It should perform the return in the following manner:

```
LDA     3, CSP
JMP     @RLOC
```

so that when A regains control, accumulator 3 is pointing to its frame. Note that the accumulators upon this return are exactly as they were left by routine B, rather than as they were when A called B.

GENERALIZED SOS SUBROUTINES

The global routines defined as entries in SOS are as follows:

.OPN - Open
.CLS - Close
.WRSE - Write sequential
.WRLI - Write line
.RDSE - Read sequential
.RDLI - Read line
.ACHR - Send a character
.RCHR - Read a character
.IBUF - Input a character to buffer
.OBUF - Output a character from buffer
.STB - Store a byte
.LDB - Load a byte
.DISM - Dismiss an interrupt

GENERALIZED SOS SUBROUTINES (Continued)

These routines are available for use with any programs loaded with SOS. The calling procedures and brief descriptions are given below:

.OPN

Calling Sequence: JSR .OPN

All references to JSR .XXX instructions, where .XXX is an entry point, are equivalent to the following instruction sequence:

```
JSR    @XXX
 :
.XXX
```

Arguments: AC0 = Byte address of the file name if the device is an intervention device. Otherwise, AC0 is ignored.
AC2 = Pointer to the DCT.

Return Sequence: Always returns to calling location+2 with the accumulators unchanged.

Description: The device's active characteristics are derived by ANDing the complement of the user's AC1 and the device's fixed characteristics (DCTFC); they are stored at DCTCH. Then, if the device being opened is an intervention device, a prompt message is typed. If the device requires leader/trailer, it is punched. The device is always cleared (DCT stop routine) and the Device Control Table is initialized.

.CLS

Calling Sequence: JSR .CLS

Arguments: AC2 = Pointer to the Device Control Table.

Return Sequence: Always returns to calling location+2 with the accumulators unchanged.

Description: If the device being closed requires leader/trailer, it is punched. When the device is no longer active, it is cleared and its Device Control Table is initialized.

GENERALIZED SOS SUBROUTINES (Continued)

.WRSE

Calling Sequence: JSR .WRSE

Arguments: "CAC0" * = Beginning byte address for transfer
"CAC1" * = Byte count for transfer
AC2 = Pointer to Device Control Table

Return Sequence: Always returns to calling location+2 with the accumulators unchanged.

Description: The specified number of bytes are inserted into the device's buffer for output.

.WRLI

Calling Sequence: JSR .WRLI

Arguments: CAC0 = Beginning byte address for the transfer
AC2 = Pointer to the Device Control Table.

Return Sequence: The accumulators are unchanged, except CAC1 which contains the count of bytes written. A return to the calling location+1 indicates the maximum line length was exceeded. A return to calling location+2 is normal.

Description: The specified ASCII characters are inserted into the device's buffer for output. The character string that is output is terminated by:

1. carriage return
2. form feed
3. null byte

Line editing is done, based on the characteristics of the device.

* Page zero displacements from the SOS Parameter Tape.

GENERALIZED SOS SUBROUTINES (Continued)

.RDSE

Calling Sequence: JSR .RDSE

Arguments: CAC0 = Beginning byte address for the transfer
CAC1 = Byte count for the transfer
AC2 = Pointer to Device Control Table

Return Sequence: A return to calling location+1 indicates either end of file or illegal command for the device. A return to calling location+2 is normal. The accumulators are unchanged. CAC1 contains the partial count read on an EOF return.

Description: The specified number of bytes are placed in the user area from the device's buffer.

.RDLI

Calling Sequence: JSR .RDLI

Arguments: CAC0 = Beginning byte address for the transfer
AC2 = Pointer to Device Control Table

Return Sequence: A return to calling location+1 indicates either end of file, line length exceeded, or parity error. In this case, AC1 contains the partial count of bytes read and CAC2 ** contains the error code. A return to calling location+2 is normal, with accumulators unchanged except CAC1 which contains the count of characters read.

Description: The specified ASCII characters are inserted into the user area from the device's buffer. The character string is terminated by:

1. carriage return
2. form feed

Nulls, rubouts, and line feeds are ignored.

** UST displacement from the SOS Parameters (PARA.SR).

.RCHR

Calling Sequence: JSR .RCHR

Arguments: AC2 = Pointer to Device Control Table

Return Sequence: A return to calling location+1 indicates device timeout (end of file). In this case, the accumulators are unchanged with the error code in CAC2. A return to calling location+2 is normal. In this case, the right justified byte (word)* that is read is in AC1 with the other accumulators unchanged.

Description: The next available byte is read from the device's buffer into AC1. If necessary, a start pulse is issued to the device.

.ACHR

Calling Sequence: JSR .ACHR

Arguments: AC1 = Right justified byte to be sent.
AC2 = Pointer to the Device Control Table

Return Sequence: A return is always made to calling location+1 with the accumulators unchanged.

Description: The byte is inserted into the device's buffer for output. If necessary, a start pulse is issued to the device.

.IBUF

Calling Sequence: JSR .IBUF

Arguments: AC1 = Right justified byte to be inserted into buffer
AC2 = Pointer to Device Control Table

Return Sequence: A return to the calling location+1 indicates the buffer is already full. A return to calling location+2 indicates the character was inserted and the buffer is not full. A return to calling location+3 indicates the character was inserted and the buffer became full. In every case, the accumulators are unchanged, except AC0, which is destroyed.

* Input devices with the characteristic, DCFWD, always operate on words rather than bytes.

GENERALIZED SOS SUBROUTINES (Continued)

.IBUF (Continued)

Description: The byte (word) is placed in the appropriate buffer slot and the Device Control Table is updated accordingly. This routine is used at the interrupt processing level by input devices and at the non-interrupt level by output devices.

.OBUF

Calling Sequence: JSR .OBUF

Arguments: AC2 = Pointer to Device Control Table

Return Sequence: A return to calling location+1 indicates the buffer is empty. A return to calling location+2 indicates the buffer is not empty. In this case, the next available byte (word) is fetched from the buffer and returned right justified in AC1. In both cases, AC0 is destroyed and the other accumulators are unchanged.

Description: The byte (word) is fetched from the appropriate buffer slot into AC1 and the Device Control Table is updated accordingly. This routine is used at the interrupt processing level by output devices (to fetch their next byte for output) and at the non-interrupt level by input devices (to retrieve bytes from their buffers).

.STB

Calling Sequence: JSR .STB

Arguments: AC0 = Destination byte address
AC1 = Right justified byte

Return Sequence: The return is always made to calling location+1 with AC0 incremented and the other accumulators unchanged.

Description: The passed byte is stored at the specified address.

GENERALIZED SOS SUBROUTINES (Continued)

.LDB

Calling Sequence: JSR .LDB

Arguments: AC0 = Source byte address

Return Sequence: The return is always made to calling location+1 with the right justified byte in AC1. The other accumulators are unchanged.

Description: The byte at the specified address is loaded and returned in AC1.

.DISM

Calling Sequence: JMP .DISM

Arguments: None.

Return Sequence: No return.

Description: This routine restores the machine to the state it was in before the device interrupted. Control is passed to this point when the interrupt from the device has been serviced.

Up to eight cassette drives and eight magnetic tape drives are permitted per system. Magnetic tape drives must be 9-track and set to high density (800 bpi). Each cassette or magnetic tape drive used at a given time must be supported by a separate support program. The number of cassette or magnetic tape drives supported by an SOS program is determined by the number of support programs included in the configuration of the program. The units supported in a given configuration are numbered consecutively starting with 0. For example, if three cassette drives are supported they will be CT0, CT1, and CT2. (See Chapter 6, Configuring SOS Utility Programs).

When running programs under SOS with the DSOSI program, cassette and magnetic tape units are referenced by file name. For magnetic tape units:

```
MT0
MT1
.
.
.
MT7
```

For cassette units:

```
CT0
CT1
.
.
.
CT7
```

When running programs under SOS without the DSOSI program, cassette and magnetic tape units are referenced by channel number. For magnetic tape units:

```
20 = MT0
21 = MT1
.
.
.
27 = MT7
```

For cassette units:

```
30 = CT0
31 = CT1
.
.
.
37 = CT7
```

Cassette or magnetic tape files are referenced by file number under SOS. Each cassette or magnetic tape unit may have up to 100 files numbered 0 through 99. Files must be written consecutively starting with file 0.

The device name and file number are specified at the time the file is opened. When performing a .OPEN command under SOS with the DSOSI program. AC0 must contain a byte pointer to the unit name/file number. The string has the format:

MTx:yy for magnetic tape units

CTx:yy for cassette units

where: x is the unit number (0 - 7)

yy is the file number (0 - 99)

File numbers may be given in single-digit or double digit format. For example:

MT1:4 and MT1:04

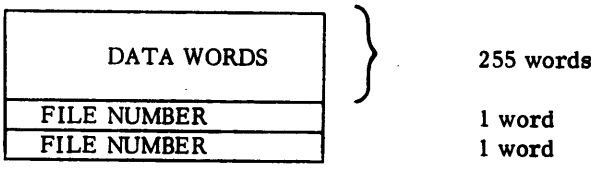
are equivalent references to the fifth file on magnetic tape unit 1.

When performing a .OPEN command without DSOSI, the channel number upon which the file is opened (20-27 for magnetic tape units; 30-37 for cassette units) is given in the command. AC0 contains the file number, which in this case may be any positive number ($0 - 99_{10}$).

When performing a .OPEN command with or without DSOSI, the characteristics mask is ignored since all cassette and magnetic tape files are attribute protected (see .GTATR command, page 3-3). The other SOS I/O commands (.RDL, .RDS, .WRL, .WRS, and .CLOSE) are used with cassette or magnetic tape files in the standard manner discussed in Chapter 3.

> NOTE: No magnetic tape file may ever be written unless the write permit ring is inserted in the tape reel. No cassette file may ever be written unless the file protect tabs are in place on the cassette.

Data is written onto and read from cassette and magnetic tapes in fixed length blocks of 257_{10} 16-bit words. The first 255 words of each block are data and the last two contain the file number:



Files are variable in length, each consisting of as many fixed length blocks as required. If the last block is not completely used, the remainder is padded with nulls. Consecutive files are separated by end-of-file (EOF) marks. Two EOF marks follow the last file written. All tapes have the following format:

Start of reel
File 0 (n_0 blocks)
EOF mark
File 1 (n_1 blocks)
EOF mark
.
.
.
File K (n_k blocks)
EOF mark
EOF mark

Whenever a file is written, two EOF marks are written following it. Since a pair of consecutive EOF marks always indicate the end of tape to SOS, overwriting a file makes inaccessible any succeeding files that were written previously. For example, if file 3 were overwritten on a tape containing 13 files (files 0 through 12), files 4 through 12 on that tape would become inaccessible to SOS. The next file written on that tape would be a new file 4. Attempts to write any other file at that point would result in a "FILE NON-EXISTENT" error code.

```
;SOS USER APPLICATION PARAMETERS (PARUA)

;STACK DISPLACEMENTS

.DUSR   SSEL=   6               ;ENTRY LENGTH
.DUSR   SOSEC=  5               ;ENTRY COUNT FOR SOS USER ROUTINES
.DUSR   RTN=    0               ;FRAME LAYOUT:
.DUSR   T0=     1               ;   RETURN LOCATION
.DUSR   T1=     2               ;   TEMPORARIES
.DUSR   SAC0=   3               ;   SAVE ACCUMULATORS
.DUSR   SAC1=   4
.DUSR   SAC2=   5
.DUSR   OAC0=   SAC0-SSEL       ;THESE DISPLACEMENTS PERMIT
.DUSR   OAC1=   SAC1-SSEL       ;  "CALLEE" TO GET AT "CALLER'S"
.DUSR   OAC2=   SAC2-SSEL       ;    REGISTERS
.DUSR   ORTN=   RTN-SSEL
.DUSR   OT0=    T0-SSEL
.DUSR   OT1=    T1-SSEL

        ;COMMAND TABLE DISPLACEMENTS

.DUSR   CTBP=   0               ;STRING BYTE POINTER
.DUSR   CTSW=   1               ;SWITCH WORD
.DUSR   CTNBP=  2               ;NEXT STARTING BYTE POINTER
.DUSR   CTEL=   2               ;ENTRY LENGTH

        ;COMMAND TABLE SWITCHES

.DUSR   SW0=    1B0             ;   EACH OF THESE BIT SETTINGS
.DUSR   SW1=    1B1             ;MAY BE EQUIVALENCED TO A MEAN-
.DUSR   SW2=    1B2             ;INGFUL VALUE IN THE USER APPLI-
.DUSR   SW3=    1B3             ;CATION PROGRAM.  THE ARRANGEMENT
.DUSR   SW4=    1B4             ;OF THE TRANSLATE TABLE (TRT)
.DUSR   SW5=    1B5             ;DETERMINES THE PRECISE MEANING
.DUSR   SW6=    1B6             ;OF EACH SWITCH.
.DUSR   SW7=    1B7
.DUSR   SW8=    1B8
.DUSR   SW9=    1B9
.DUSR   SW10=   1B10
.DUSR   SW11=   1B11
```

```
;
;       SOS PARAMETERS
;

;               LINKAGE

.DUSR   SAVE=   JSR     @3
.DUSR   RTRN=   JMP     @4
.DUSR   RTLOC=  0
.DUSR   AC0=    1
.DUSR   AC1=    2
.DUSR   TMP=    3
.DUSR   SLGT=   TMP+1
.DUSR   OAC0=   AC0-SLGT
.DUSR   OAC1=   AC1-SLGT
.DUSR   OTMP=   TMP-SLGT
.DUSR   ORTN=   RTLOC-SLGT
.DUSR   NFRAM=  6.
.DUSR   SSZ=    NFRAM*SLGT

;               PAGE ZERO

.DUSR   RLOC=   6
.DUSR   CMSK=   7
.DUSR   CSP=    10
.DUSR   CDCT=   12      ;IN SERVICE DCT
.DUSR   BDCT=   13      ;BEGINNING OF DCT CHAIN
.DUSR   CAC0=   14
.DUSR   CAC1=   15

;               ADDITIONAL UST DEFINITIONS

.DUSR   USTA0=  20
.DUSR   USTA1=  USTA0+1
.DUSR   USTA2=  USTA1+1
.DUSR   USTA3=  USTA2+1
.DUSR   USTCY=  USTA3+1
.DUSR   USTIS=  USTCY+1
.DUSR   USTWA=  USTIS+1
.DUSR   USTRS=  USTWA+1

.DUSR   UFTEC=  10      ; ENTRY COUNT OF UFT'S
.DUSR   UFT=    UST+UFPT+UFTEC  ; START OF UFT TABLE

;               ADDITIONAL DEVICE CHARACTERISTICS

.DUSR   DCDIR=  1B0     ; SOS DATA CHANNEL DEVICE
```

```
; DEVICE CONTROL TABLE (DCT) LAYOUT

; COMMON TO ALL DEVICES

.DUSR DCTCD=    0       ; DEVICE CODE
.DUSR DCTMS=    1       ; MASK OF LOWER PRIORITY DEVICES

        ;DEFINE THE MASK BITS

        .DUSR   MSTTO=  1B15
        .DUSR   MSTTI=  1B14
        .DUSR   MSPTP=  1B13
        .DUSR   MSLPT=  1B12
        .DUSR   MSCDR=  1B10
        .DUSR   MSPLT=  1B12
        .DUSR   MSMTA=  1B10
        .DUSR   MSPTR=  1B11

.DUSR DCTCH=    2       ; DEVICE CHARACTERISTICS
.DUSR DCTLK=    3       ; LINK TO NEXT DCT
                        ; (-1 TERMNATES THE CHAIN)
.DUSR DCTIS=    4       ; INTERRUPT SERVICE ROUTINE ADDRESS
.DUSR DCTIL=    5       ; INTERRUPT MACHINE STATE LINK


.DUSR DCTDT=    6       ; COMMAND DISPATCH TABLE ADDRESS WORD
.DUSR DCTST=    7       ; ADDRESS OF DEVICE START ROUTINE
.DUSR DCTSP=    10      ; ADDRESS OF DEVICE STOP ROUTINE
.DUSR DCTFL=    11      ; FLAGS (ACTIVE, ATTACHED, ETC.)

        ; DEFINE THE FLAGS
        .DUSR   DCACT=  1B15    ; ACTIVE FLAG
        .DUSR   DCACPT= 1B8     ; ACCEPT CHARACTER FLAG
        .DUSR   DCKMD=  1B0     ; TTY KEYBOARD MODE FLAG


; COMMON TO DEDICATED DEVICES (I.E. SINGLE USER/SINGLE BUFFER)

.DUSR DCTBS=    12      ; BUFFER SIZE ( BYTES )
.DUSR DCTBF=    13      ; BUFFER FIRST ADDRESS (BYTE )
.DUSR DCTBL=    14      ; BUFFER LAST  ADDRESS
.DUSR DCTIP=    15      ; BUFFER INPUT POINTER (BYTE )
.DUSR DCTOP=    16      ; BUFFER OUTPU POINTER
.DUSR DCTCN=    17      ; COUNT OF ACTIVE DATA
.DUSR DCTTO=    20      ; TIMEOUT WORD (ALL INPUT DEVICES)
.DUSR DCTCC=    20      ; COLUMN COUTER (ALLOUTPUT DEVICES)
.DUSR DCTRC=    21      ; RESTART CONSTANT (ALL INPUT DEVICES)
.DUSR DCTLC=    21      ; LINE COUNTER (ALL OUTPUT DEVICES)
.DUSR DCTAT=    22      ;DEVICE ATTRIBUTES
.DUSR DCTFC=    23      ;DEVICE FIXED CHARACTERISTICS

.DUSR LCHNO=    6       ; LOWEST LEGAL CHANNEL #
.DUSR HCHNO=    37      ; HIGHEST LEGAL CHANNEL #
                        ; NOTE - ONE OR BOTH OF THESE EQUI-
                        ;        VALENCES MAY BE CHANGED TO ADD
                        ;        DEVICE DRIVERS
```

```
; MAG TAPE PARAMETERS

.DUSR   MTBW2=  377     ;BUFFER WORD SIZE
.DUSR   MTBB2=  MTBW2*2 ;BUFFER BYTE SIZE
.DUSR   CTBW2=  MTBW2
.DUSR   CTBB2=  MTBB2


;               INTERRUPT FRAME TEMPLATE

.DUSR   IAC0=   DCTFC+1
.DUSR   IAC1=   IAC0+1
.DUSR   IAC2=   IAC1+1
.DUSR   IAC3=   IAC2+1
.DUSR   IPC=    IAC3+1
.DUSR   IRLOC=  IPC+1
.DUSR   IMSK=   IRLOC+1
.DUSR   IFRL=   7       ;INTERRUPT FRAME LENGTH


; DEFINE THE CLI STACK DISPLACEMENTS

.DUSR SSLGT=    -7      ; VARIABLE LENGTH OF CALLING'S FRAME
.DUSR SSOSP=    -6      ; PREVIOUS STACK POINTER
.DUSR SSRTN=    -5      ; RETURN ADDRESS OF CALLING PROGRAM
.DUSR SSEAD=    -4      ; ENTRY ADDRESS OF CALLED ROUTINE
.DUSR SSCRY=    -3      ; CARRY
.DUSR SSAC0=    -2      ; SAVE STORAGE FOR CALLING'S ACCUMULATORS
.DUSR SSAC1=    -1
.DUSR SSAC2=    0       ; (DON'T MODIFY THIS DISPLACEMENT!!)

; DEFINE THE CLI SIZE

.DUSR CLIN= 13015       ;CLI NREL

;
; UFT ENTRY
;

.DUSR UFTFN=0           ;FILE NAME
.DUSR UFTEX=5           ;EXTENSION
.DUSR UFTAT=6           ;FILE ATTRIBUTES
.DUSR UFTBK=7           ;NUMBER OF LAST BLOCK IN FILE
.DUSR UFTBC=10          ;NUMBER OF BYTES IN LAST BLOCK
.DUSR UFTAD=11          ;DEVICE ADDRESS OF FIRST BLOCK (0 UNASSIGNED)
.DUSR UFTDL=12          ;DCT LINK

.DUSR UFTDC=13          ;DCT ADDRESS
.DUSR UFTUN=14          ;UNIT NUMBER
.DUSR UFTCA=15          ;CURRENT BLOCK ADDRESS
.DUSR UFTCB=16          ;CURRENT BLOCK NUMBER
.DUSR UFTST=17          ;FILE STATUS
.DUSR UFTNA=20          ;NEXT BLOCK ADDRESS
.DUSR UFTLA=21          ;LAST BLOCK ADDRESS
.DUSR UFTDR=22          ;SYS.DR DCB ADDRESS
.DUSR UFTFA=23          ;FIRST ADDRESS

.DUSR UFTBN=24          ;CURRENT FILE BLOCK NUMBER
.DUSR UFTBP=25          ;CURRENT FILE BLOCK BYTE POINTER
.DUSR UFTCH=26          ;DEVICE CHARACTERISTICS
.DUSR UFTCN=27          ;ACTIVE REQ COUNT
                        ;B0 INDICATES 0 -0=DS01,1=DS02

.DUSR UFTEL=UFTCN-UFTFN+1       ;UFT ENTRY LENGTH
.DUSR UFDEL=UFTDL-UFTFN+1       ;UFD ENTRY LENGTH
```

```
;
; SYSTEM FILE ENTRY
;

.DUSR SFKEY=-5          ;KEY
.DUSR SFLK=-4           ;MAP.DR LINK (-1 IF NOT DSK DVC)
.DUSR SFNX=-3           ;NEXT ENTRY IN CHAIN
.DUSR SFBK=-2           ;NUMBER OF LAST BLOCK IN FILE
.DUSR SFBC=-1           ;BYTE IN LAST BLOCK
.DUSR SFDCB=0           ;DCB ENTRY

.DUSR UDBAT=UFTAT-UFTDC ;NEGATIVE DISP. TO ATTRIBUTES
.DUSR UDBAD=UFTAD-UFTDC ;NEGAVIE DISP. TO FIRST ADDRESS
.DUSR UDBBK=UFTBK-UFTDC ;NEGATIVE DISP. TO LAST BLOCK
.DUSR UDBBN=UFTBN-UFTDC ;POSITIVE DISP. TO CURRENT BLOCK



;
; FILE ATTRIBUTES
;

.DUSR ATRP=1B0          ;READ PROTECTED
.DUSR ATCHA=1B1         ;CHANGE ATTRIBUTE PROTECTED
.DUSR ATSAV=1B2         ;SAVED FILE
.DUSR ATLNK=1B3 ;PART DISK LINK
.DUSR ATPAR=1B4 ;PARTITIONED
.DUSR ATCON=1B12        ;CONTIGUOUS FILE
.DUSR ATRAN=1B13        ;RANDOM FILE
.DUSR ATPER=1B14        ;PERMANENT FILE
.DUSR ATWP=1B15         ;WRITE PROTECTED

;
; FILE STATUS
;

.DUSR STER=1B15         ;ERROR DETECTED
.DUSR STIOP=1B14        ;I/O IN PROGRESS
.DUSR STFWR=1B13        ;FIRST WRITE FLAG
.DUSR STINI=1B1         ;NO INIT BIT
.DUSR STCMK=1B0         ;SET = READ (FILIO)
                        ;(INIT/RELEASE SWTCH FOR SYS.DR DCB)

;
; BUFFER STATUS
;

.DUSR QTMOD=1B15        ;HAS BEEN MODIFIED
.DUSR QTER=1B14         ;ERROR DETECTED
.DUSR QTIOP=1B12        ;I/O IN PROGRESS
.DUSR QTLCK=1B11        ;BUFFER LOCKED
.DUSR QTIND=1B10        ;INDIRECT MODE (ADDRESS IN BQNXT)
.DUSR QTEMD=1B9         ;ERROR MODE (MAG TAPE)
```

```
;
; SYSTEM CONSTANTS.
;

.DUSR SCWPB=255.          ;WORDS PER BLOCK
.DUSR SCLLG=132.          ;MAX LINE LENGTH
.DUSR SCAMX=24.           ;MAX ARGUMENT LENGTH IN BYTES
.DUSR SCFNL=UFTEX-UFTFN+1 ;FILE NAME LENGTH
.DUSR SCMER=10.           ;MAX ERROR RETRY COUNT
.DUSR SCSTR=16            ;SAVE FILE STARTING ADDRESS
.DUSR SCTIM=-80.          ;RINGIO 1 MS. LOOP TIME (SN)
.DUSR SCSYS=1             ;DEVICE ADDRESS FOR SYS.DR
.DUSR SCSVB=SCSYS+1       ;4 CONTIGUOUS BLOCKS FOR CORE IMAGES
.DUSR SCPNM=4             ;NUMB PUSH LEVELS
.DUSR SCEXT=UFTEX-UFTFN   ;EXTENSION OFFSET IN NAME AREA
.DUSR SCRRL=64.           ;WORDS PER RANDOM RECORD
.DUSR SFINT=1B0           ;INTERRUPT FLAG
.DUSR SFCRD=1B13          ;CRITICAL READ ERROR
.DUSR SFPRD=1B14          ;PANIC ON READ ERROR
.DUSR SFBRK=1B15          ;BREAK FLAG
.DUSR FADZ=40             ;MAP LOCATION IN BOOTSTRAP
.DUSR SCZMX=FADZ+1
.DUSR SCFUL=SCZMX+1
.DUSR SCPAR=SCFUL+1
.DUSR SCPOV=SCPAR+1       ;PARTIAL INIT WITH OVERLAYS
.DUSR SCKEY=SCPOV+1
.DUSR SCCLI=SCKEY+1       ;END OF CLI LOCATION IN BOOTSTRAP
.DUSR SCIDV=SCCLI+1       ;DEVICE CODE OF BOOTSTRAP DEVICE

; DEFINE THE EXCEPTIONAL STATUS CODES

.DUSR ERFNO=    0         ; ILLEGAL CHANNEL NUMBER
.DUSR ERFNM=    1         ; ILLEGAL FILE NAME
.DUSR ERICM=    2         ; ILLEGAL SYSTEM COMMAND
.DUSR ERICD=    3         ; ILLEGAL COMMAND FOR DEVICE
.DUSR ERSV1=    4         ; NOT A SAVED FILE
.DUSR ERWR0=    5         ; ATTEMPT TO WRITE AN EXISTENT FILE
.DUSR EREOF=    6         ; END OF FILE
.DUSR ERRPR=    7         ; ATTEMPT TO READ A READ PROTECTED FILE
.DUSR ERWPR=    10        ; WRITE PROTECTED FILE
.DUSR ERCRE=    11        ; ATTEMPT TO CREATE AN EXISTENT FILE
.DUSR ERDLE=    12        ; A NON-EXISTENT FILE
.DUSR ERDE1=    13        ; ATTEMPT TO ALTER A PERMANENT FILE
.DUSR ERCHA=    14        ; ATTRIBUTES PROTECTED
.DUSR ERFOP=    15        ; FILE NOT OPENED
.DUSR ERUFT=    21        ; ATTEMPT TO USE A UFT ALREADY IN USE
.DUSR ERLLI=    22        ; LINE LIMIT EXCEEDED 0
.DUSR ERRTN=    23        ; ATTEMPT TO RESTORE A NON-EXISTENT IMAGE
.DUSR ERPAR=    24        ; PARITY ERROR ON READ LINE
.DUSR ERCM3=    25        ; TRYING TO PUSH TOO MANY LEVELS
.DUSR ERMEM=    26        ; NOT ENUF MEMORY AVAILABLE
.DUSR ERSPC=    27        ; OUT OF FILE SPACE
.DUSR ERFIL=    30        ; FILE READ ERROR
.DUSR ERSEL=    31        ; UNIT NOT PROPERLY SELECTED
.DUSR ERADR=    32        ; ILLEGAL STARTING ADDRESS
.DUSR ERRD=     33        ; ATTEMPT TO READ INTO SYSTEM AREA
.DUSR ERDIR=    35        ; FILES SPECIFIED ON DIFF. DIRECTORIES
.DUSR ERDNM=    36        ; ILLEGAL DEVICE NAME
.DUSR EROVN=    37        ; ILLEGAL OVERLAY NUMBER
.DUSR EROVA=    40        ; ILLEGAL OVERLAY FILE ATTRIBUTE
.DUSR ERTIM=    41        ; USER SET TIME ERROR
.DUSR ERNOT=    42        ; OUT OF TCB'S
.DUSR ERXMT=    43        ; SIGNAL TO BUSY ADDR
.DUSR ERSQF=    44        ; SQUASH FILE ERROR
.DUSR ERIBS=    45        ; DEVICE ALREADY IN SYSTEM
.DUSR ERICB=    46        ; INSUFFICENT CONTIGUOUS BLOCKS
.DUSR   ERSIM=  47        ;QTY ERROR
.DUSR ERQTS=    50        ;ERROR IN USER TASK QUEUE TBL
```

```
; CLI ERROR CODES

.DUSR CNEAR=100          ;NOT ENOUGH ARGUMENTS
.DUSR CILAT=101          ;ILLEGAL ATTRIBUTE
.DUSR CNDBD=102          ;NO DEBUG ADDRESS
.DUSR CNCTD=103          ;NO CONTINUATION ADDRESS
.DUSR CNSAD=104          ;NO STARTING ADDRESS
.DUSR CCKER=105          ;CHECKSUM ERROR
.DUSR CNSFS=106          ;NO SOURCE FILE SPECIFIED
.DUSR CNACM=107          ;NOT A COMMAND
.DUSR CILBK=110          ;ILLEGAL BLOCK TYPE
.DUSR CSPER=111          ;NO FILES MATCH SPECIFIER
.DUSR CPHER=112          ;PHASE ERROR
.DUSR CTMAR=113          ;TOO MANY ARGUMENTS

; DEFINE THE PANICS

.DUSR   PNOP=    010     ; NOP MAGIC
.DUSR   POFFS=   1B11    ; OFFSET

.DUSR   PNSDE=   21*POFFS+PNOP   ;SOMEBODY RAPED MY SYS.DR
.DUSR   PNCSO=   22*POFFS+PNOP   ; SYSTEM STACK OVERFLOW
.DUSR   PNQER=   23*POFFS+PNOP   ;QUEUE ERROR-ILLEG BLK
.DUSR   PNCDE=   25*POFFS+PNOP   ; CRITICAL DISK READ/WRITE ERROR
.DUSR   PNCRR=   26*POFFS+PNOP   ; RUNAWAY READER
.DUSR   PNDPE=    27*POFFS+PNOP  ;MOVING HEAD DISK ERROR
.DUSR   PNDPE= 27*POFFS+PNOP     ;DISK PACK ERROR

        ; DEFINE THE CHARACTERISTICS

.DUSR   DCCPO=   1B15    ; DEVICE REQUIRING LEADER/TRAILER
.DUSR   DCCGN=   1B14    ; GRAPHICAL OUTPUT DEVICE WITHOUT TABBING
                         ; HARDWARE
.DUSR   DCIDI=   1B13    ; INPUT DEVICE REQUIRING OPERATOR INTERVENTION
.DUSR   DCCNF=   1B12    ; OUTPUT DEVICE WITHOUT FORM FEED HARDWARE
.DUSR   DCTO=    1B11    ; TELETYPE OUTPUT DEVICE
.DUSR   DCKEY=   1B10    ; KEYBOARD DEVICE
.DUSR   DCNAF=   1B9     ; OUTPUT DEVICE REQUIRING NULLS AFTER FORM FEEDS
.DUSR   DCRAT=   1B08    ; RUBOUTS AFTER TABS REQUIRED
.DUSR   DCPCK=   1B07    ; DEVICE REQUIRING PARITY CHECK
.DUSR   DCLAC=   1B06    ; REQUIRES LINE FEEDS AFTER CARRIAGE RTN
.DUSR   DCSPO=   1B05    ; SPOOLABLE DEVICE
.DUSR   DCFWD=   1B04    ; FULL WORD DEVICE (ANYTHING GREATER THAN
.DUSR   DCFFO=   1B03    ; FORM FEEDS ON OPEN
.DUSR   DCLTU=   1B02    ; CHANGE LOWER CASE ASCII TO UPPER
.DUSR   DCC80=   1B01    ; READ 80 COLUMS
.DUSR   DCSPC=   1B00    ; SPOOL CONTROL
                         ; SET = SPOOLING ENABLED
                         ; RESET = SPOOLING DISABLED
```

```
; USER STATUS TABLE (UST) TEMPLATE

.DUSR   UST=    400     ; START OF BACKGROUND USER STATUS AREA

.DUSR   USTP=12         ;%ZERO LOC FOR UST POINTER
; NOTE- USTP MUST CORRESPOND TO PARS %ZERO ALLOCATIONS

.DUSR   USTPC=  0
.DUSR   UST%M=  1       ; %MAX
.DUSR   USTSS=  2       ; START OF SYMBOL TABLE
.DUSR   USTES=  3       ; END OF SYMBOL TABLE
.DUSR   USTNM=  4       ; NMAX
.DUSR   USTSA=  5       ; STARTING ADDRESS
.DUSR   USTDA=  6       ; DEBUGGER ADDRESS
.DUSR   USTHU=  7       ; HIGHEST ADDRESS USED
.DUSR   USTCS=  10      ; FORTRAN COMMON AREA SIZE
.DUSR   USTIT=  11      ; INTERRUPT ADDRESS
.DUSR   USTBR=  12      ; BREAK ADDRESS
.DUSR   USTIN=  13      ; INITIAL START OF NREL CODE
.DUSR   USTCT=  14      ; CURRENTLY ACTIVE TCB
.DUSR   USTAC=  15      ; START OF ACTIVE TCB CHAIN
.DUSR   USTFC=  16      ; START OF FREE TCB CHAIN
.DUSR   USTCH=  17      ; NUMBER OF CHANNELS
.DUSR   USTOD=  20      ; OVLY DIRECTORY ADDR
.DUSR   USTSV=  21      ; FORTRAN STATE VARIABLE SAVE ROUTINE (OR 0)
.DUSR   USTEN=  USTSV   ; LAST ENTRY

; DEFINE 8 SPARE WORDS

.DUSR UFPT=30   ;USER FILE POINTER TABLE


.DUSR TPC=0     ;USER PC + CARRY
.DUSR TAC0=1    ;AC0
.DUSR TAC1=2    ;AC1
.DUSR TAC2=3    ;AC2
.DUSR TAC3=4    ;AC3
.DUSR TPRST=5   ;STATUS BITS + PRIORITY
.DUSR TSYS=6    ;SYSTEM CALL WORD
.DUSR TLNK=7    ;LINK WORD
.DUSR TUSP= 10  ;USP
.DUSR TELN=11   ;TCB EXTENTION ADDR
.DUSR TLN=TELN-TPC+1

; STATUS BITS: 1B0 = TASK PENDED, 2B0 = WAITING FOR OVERLAY AREA


.DUSR OVNDS= 0  ;DIRECTORY NODE TABLE START
.DUSR   OVDAD= 10       ;DEV ADDR OF INDEX
.DUSR   OVIDX= 1        ;OFFSET FOR INDEX 1 IN IV INDEX
.DUSR   OVNAD= 2        ;NODE POINT
.DUSR OVNLN= OVNAD-OVIDX+1      ;LENGTH OF A NODE GLOP
; OFFSETS FOR USER TASK QUEUE TABLE
.DUSR   QPC= 0          ;STARTING PC
.DUSR   QNUM= 1         ;NUMBER OF TIMES TO EXEC
.DUSR   QTOV= 2         ;OVERLAY
.DUSR   QSH= 3          ;STARTING HOUR
.DUSR   QSMS= 4         ;STARTING SEC IN HOUR
        QPRI=TPRST      ;MUST BE SAME
.DUSR   QRR= 6          ;RERUN TIME INC IN SEC
        QTLNK=TLNK      ;MUST BE SAME
.DUSR   QOCH= 10        ;CHAN OVERLAYS OPEN ON
.DUSR   QCOND= 11       ;TYPE OF OVLY LOAD
.DUSR   QTLN= QCOND-QPC+1
```

Available to users with one or more cassette drives are cassettes numbered 085-000002, and 085-00003. Supplied to users with one or more magnetic tape drives is magnetic tape unit number 085-000004. Below are the order of files contained on each.

CASSETTES (Model #3235)

085-000002:

Core Image Loader/Writer	File 0
Command Line Interpreter	File 1
Text Editor	File 2
Assembler	File 3
Relocatable Loader	File 4
Library File Editor	File 5
SYSGEN	File 6

085-000003:

Cassette Library	File 0
SOS Library	File 1
Command Line Interpreter (RB)	File 2
Text Editor (RB)	File 3
Assembler (RB)	File 4
Relocatable Loader (RB)	File 5
Library File Editor (RB)	File 6
SYSGEN	File 7
Extended Assembler Command Definitions:	
Nova Basic Instructions	File 8
Floating Point Interpreter	File 9
Operating Systems	File 10
RDOS User Parameters	File 11
SOS Stand-alone Parameters	File 12
SOS User Application Parameters	File 13

MAGNETIC TAPE (Model #3236)

085-000004:

Core Image Loader/Writer	File 0
Command Line Interpreter	File 1
Text Editor	File 2
Assembler	File 3
Relocatable Loader	File 4
Library File Editor	File 5
SYSGEN	File 6
SOS Magnetic Tape Library	File 7
SOS Library	File 8
Command Line Interpreter (RB)	File 9
Text Editor (RB)	File 10
Assembler (RB)	File 11
Relocatable Loader (RB)	File 12
Library File Editor (RB)	File 13
SYSGEN (RB)	File 14
Extended Assembler Command Definitions:	
Nova Basic Instructions	File 15
Floating Point Interpreter	File 16
Operating Systems	File 17
RDOS User Parameters	File 18
SOS Stand-alone Parameters	File 19
SOS User Application Parameters	File 20

cut along dotted line

DATA GENERAL CORPORATION
PROGRAMMING DOCUMENTATION
REMARKS FORM

DOCUMENT TITLE _____

DOCUMENT NUMBER (lower righthand corner of title page) _____

TAPE NUMBER (if applicable) _____

Specific Comments. List specific comments. Reference page numbers when applicable. Label each comment as an addition, deletion, change or error if applicable.

General Comments and Suggestions for Improvement of the Publication.

FROM: Name: _____ Date: _____
Title: _____
Company: _____
Address: _____
_____

FOLD DOWN FIRST FOLD DOWN

FIRST
CLASS
PERMIT
No. 26
Southboro
Mass. 01772

BUSINESS REPLY MAIL

No Postage Necessary If Mailed In The United States

Postage will be paid by:

Data General Corporation

Southboro, Massachusetts 01772

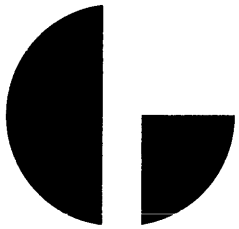
ATTENTION: Programming Documentation

FOLD UP SECOND FOLD UP

STAPLE

CHANGES FROM REVISION 02 TO REVISION 03 OF THE STAND-ALONE OPERATING SYSTEM USER'S MANUAL

Page	Nature of Change
i	Introduction has been added to the manual.
1-1	Three new tape numbers have been added to replace tape number 089-000890, Extended Assembler Command Definitions.
1-1	A new device driver tape has been added for the support of a 132-column Line Printer Driver.
3-1	Description of the system command formats has been expanded; and further description is included concerning passing channel numbers through the use of AC2.
3-1	First sentence, in third paragraph under STATUS ON RETURN FROM SYSTEM now reads: "...(as it is by the user of JSR)."
3-2	Further description of the .OPEN command has been added.
3-2, 3-3	Text has been changed to indicate that character strings are terminated by either carriage returns, form feeds, or nulls.
3-6	.GTATR has been added to list of applicable commands for error code 0; and .RDL, .WRL, .RDS, .WRS have been added to applicable commands for error code 15.
3-6	Further description is included on the functions performed by the $CDR on a .RDS.
3-6	On a .RDL using the $CDR, a 12-11-0-1-2-3-4-5-6-7-8-9 punch causes end of file.
3-7	The line printer may be used for either 80-column or 132-column printouts.
4-3	First sentence, second paragraph now reads, "The user-supplied translate table (TRT) establishes the definition of bits 0-10 of this table."
5-3	Description of Loading an Absolute Binary Paper Tape has been further expanded.



DATA GENERAL
CORPORATION
Southboro,
Massachusetts 01772
(617) 485-9100

ADDENDUM TO

THE STAND-ALONE OPERATING

SYSTEM USER'S MANUAL

093-000062-03

New features have been implemented for the Stand-alone Operating System (SOS). Additionally, changes have been made to existing features of SOS. The additions and changes are described below.

ADDED INFORMATION

Special Keyboard Interrupts

The User Status Table displacements USTIT and USTBR are now available for use to transfer control as a result of interrupts from special keyboard characters. These characters are CTRL A and CTRL C, respectively. Whenever the $TTI interrupt handler detects the receipt of a CTRL A character, it checks the contents of UST + USTIT; whenever it detects a CTRL C character, it checks the contents of UST + USTBR. In both cases if the contents of the UST location are 0 or -1, the character (C or A) is treated as part of the normal input stream and no further action is taken.

If the UST location contains anything else, control is unconditionally transferred to the address contained in that location. All I/O will be reset and interrupts will be enabled if the break character was a CTRL A and disabled if the character was CTRL C.

The contents of UST + USTIT are never altered by SOS. When the SOS Relocatable Loader builds a load module, that location is initialized to -1. To receive control after CTRL A interrupts, the user program must change the location appropriately. The contents of UST + USTBR are set to the highest memory address by SOS whenever a .SYSI command is executed. Therefore, if this location is left unchanged by the user, a CTRL C break results in a jump to the last memory address. The last memory location in a cassette or magnetic tape environment is usually the start of the core image loader. And, in a paper tape environment, the last memory location is the start of the absolute binary loader.

UST+USTBR may point to any location, however. Whichever location it does point to is the address to which control will be passed when a CTRL C is detected (with all I/O cleared and interrupt disabled). Note that use of one of the control keys to stop a cassette or magnetic tape write operation could cause an incomplete file to be written to the tape. Therefore, subsequent errors in file positioning may occur unless the user performs a manual rewind.

Utility Programs' Treatment of CTRL Characters

All SOS Utility Programs, except the Editor, treat CTRL C and CTRL A interrupts in the following manner:

CTRL C - terminate utility program
- invoke either the core image loader or the absolute binary loader

CTRL A - terminate current program activity, restart utility program

The action taken by the Editor upon receipt of these control characters is:

CTRL C - never causes any kind of a break under any circumstances

CTRL A - If one of the following Editor commands is in progress:

T, Y, N, E, or P

causes a termination of the operation and a restart of the Editor with all I/O reset but with the input buffer intact.
- Otherwise treated as part of the normal input stream and may be ignored, recognized as a legal Insert character or as an illegal command depending upon its context in the input stream.

CHANGED INFORMATION

The "H" command under the Text Editor now causes a jump to the core image or absolute binary loaders.

In the USTP, location 12 now points to UST.

A new switch has been added to the CLI command, INIT. The /F switch causes two EOF records to be written to the appropriate cassette or magnetic tape unit, starting from the beginning of the tape.

On page C-4 a reference is made to tape number 090-000890 under SYSGEN procedures for the Assembler. This tape has been replaced with three instruction tapes, numbered as follows.

Extended Assembler Command Definitions	
Instruction definitions-Nova Basic Instructions	090-001482
Instruction definitions-Floating Point Interpreter	090-001483
Instruction definitions-Operating Systems	090-001484