DATA GENERAL
CORPORATION

Southboro,
Massachusetts 01772
(617) 485-9100

PROGRAM

DOS-COMPATIBLE STAND-ALONE OPERATING SYSTEM USER'S MANUAL

TAPE

Library Binary:	099-000071

ABSTRACT

Data General's DOS-Compatible Stand-alone Operating System can be used with Nova-line computers to perform all input/output device handling except disk I/O. I/O is interrupt-driven, using core buffers unique to each device.

093-000094-00

Original Release September, 1973

INTRODUCTION

The DOS-Compatible Stand-alone Operating System (DSOS) can be used to perform all I/O device handling except disk I/O. I/O is performed on an interrupt driven basis, using core buffers unique to each declared device.

Under DSOS channel numbers are fixed and always denote the same device/file. The DSOS channel number assigned to each device is as follows:

Device	Channel Number
$PLT	6
$TTP	10
$CDR	11
$TTO	12
$TTI	13
$LPT	14
$PTR	15
$PTP	16
$TTR	17

All of the devices listed above are supported by DSOS. Additional device drivers may be added by users following the procedure outlined in Appendix A

DSOS is supplied to DGC customers as a library tape containing the relocatable binary programs:

CDRDR	Card Reader Driver
PLTDR	Plotter Driver
SOS	Main Program of DSOS
DSOSI	DOS to SOS Interface Program

DOS-Compatible SOS is loaded with the Extended Relocatable Loader, 091-000038. To load the main program of DSOS (SOS), .SOS must be declared an external normal in a previously loaded program. The main program of DSOS supports all DGC system devices except the card reader and the plotter. To load the card reader driver, .CDRD must be declared an external normal; to load the plotter driver, .PLTD must be declared an external normal.

The DOS to SOS interface program (DSOSI) permits users to call DSOS while using the DOS channel number scheme. Under DOS, files are assigned by file name to temporary channel numbers while under DSOS channel numbers always denote the same file or device and file names are not recognized. The DOS to SOS interface program is loaded by declaring .DSI as an external normal instead of .SOS. The

I/O commands for this program are identical to the Disk Operating System (DOS) I/O commands as described in Chapter 4 of the DOS User's Manual, 093-000048. The legal channel numbers are 0-7 under DOS; the file names that are passed are used by DSOSI to derive SOS channel numbers.

DSOS program commands are described in Chapter 1. The properties of each DSOS device are given in Chapter 2. Appendix A contains a detailed internal description of DSOS which permits users to add their own device drivers. Appendix B contains ASCII listings of the DSOS System Parameter Tape and the DOS User Parameter Tape.

TABLE OF CONTENTS

CHAPTER 1

DSOS - PROGRAM COMMUNICATION

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

INITIALIZATION OF COMMUNICATION (.SYSI)

The command .SYSI must be issued before any other DSOS commands can be used. It clears all devices and allocates to DSOS additional memory for device buffers, etc. Memory allocation is only performed on the first .SYSI. Additional .SYSI commands can be given if the user wishes to clear devices.

The format of the command is:

```
        .SYSI                   ;INITIALIZE DSOS
        normal return           ;AC'S AND CARRY ARE PRESERVED
```

There is no error return from a .SYSI command.

SYSTEM COMMAND FORMAT

After communication is intialized using the command .SYSI, all other commands have the general format:

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

The mnemonic .SYSTM and the command words are recognized as legal mnemonics by the DGC Relocatable Assembler.

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

```
                JSR @2
```

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.

SYSTEM COMMAND FORMAT (Continued)

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.

When no I/O channel is needed in command execution, the command word 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.

The channel (device) number for any system command requiring a channel number can also be passed in AC2. Specifying octal 77 as the channel number in the instruction causes the system to use the number passed in AC2. For example, the following instructions specify a write to channel 16:

```
        LDA     2,C16
        .SYSTM
        .WRS    CPU
        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.

STATUS ON RETURN FROM SYSTEM (Continued)

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 a JSR). On return from the system, however, AC3 is loaded from the contents of memory location 00016. 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 before issuing the .SYSTM.

LIST OF COMMAND WORDS

The command word mnemonics are:

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

DSOS commands are a subset of DOS commands. All other DOS commands, including .RTN, result in return of the error code "ILLEGAL SYSTEM COMMAND" 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 or form feeds. 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 to all paper tape output devices. Further, reading and writing 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.

Open a File (.OPEN)

Before other I/O commands can be used, a device must be opened with the .OPEN command. This 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.

```
.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.

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.

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.

Read a Line (.RDL)

The command causes an ASCII line, having even parity, to be read. ACØ 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 or a form feed to the user. Reading will terminate abnormally after transmission of 132 characters (decimal) without detecting a carriage return or a 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 *

The format of the .RDL command is:

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

* C(x) means "the contents of x".

Read a Line (.RDL) (Continued)

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.
22	ERLLI	Line limit (132 characters) exceeded.
24	ERPAR	Parity error.

Write a Line (.WRL)

This 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, a 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.
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:

Read Sequential(.RDS) (Continued)

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

Possible errors resulting from a .RDS command are:

AC0	Mnemonic	Meaning
0	ERFNO	Illegal channel number.
3	ERICD	Illegal command for device.
6	EREOF	End of file.

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

Write Sequential (.WRS)

This command transmits data exactly as read from the user area. AC0 must contain a byte pointer to the starting byte 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.

TYPEWRITER COMMANDS

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

Get a Character (.GCHAR)

This command returns a character typed from the teletypewriter in AC0. The character is right-adjusted in AC0 with bits 0-8 cleared. No channel is required; the TTI

Get a Character (.GCHAR) (continued)

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)

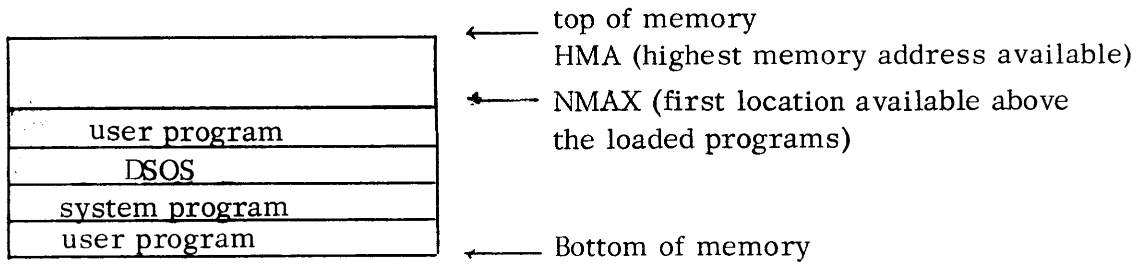
This command transmits a character in AC0, bits 9-15, to the teletypewriter. No channel is required; the TTO is always used as output for this command. The format of the .PCHAR command is:

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

No error return is possible from this command.

MEMORY COMMANDS

Upon completion of a relocatable load, the DOS-Compatible 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 the loaded programs)
user program
DSOS
system program
user program
Bottom of memory

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 may represent either the bottom of the Binary Loader or the end of the user symbol table. A SUB 1,0 instruction 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 .MEMI 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 user 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 resulting from a .MEMI command:

Change NMAX (.MEMI) (Continued)

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 .RDS .WRS .CLOSE .RDL .WRL
2	ERICM	ILLEGAL SYSTEM COMMAND	--
3	ERICD	ILLEGAL COMMAND FOR DEVICE	.RDS .WRS .RDL .WRL
6	EREOF	END OF FILE	.RDS .RDL
7	ERRPR	ATTEMPT TO READ A READ PROTECTED FILE	.RDS .RDL
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

CHAPTER 2

DSOS DEVICE PROPERTIES

This chapter describes the functions performed by the DSOS I/O commands as applied to each of the devices supported by DGC.

$PLT

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

$TTP

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

$CDR

.OPEN command	-	Device is initialized; a prompt message is written and a response is necessary for the program to continue.
.CLOSE command	-	Device is reinitialized.
.RDS command	-	The specified bytes are read into the user area from the device, unedited.
.RDL command	-	The 80 character ASCII string is read into the user area from the device. The translation from Hollerith is performed in the card reader driver. A 12-11-0-1 punch causes end of file. The byte count returned to the user reflects the last non-blank character on the card.

$TTO

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

$TTI

.OPEN command	-	Device is initialized.
.CLOSE command	-	Device is reinitialized.
.RDS command	-	The specified bytes are read into the user area from the device, unedited.
.RDL command	-	The ASCII string is read into the user area from the device. The input stream is echoed to 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.

$LPT

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

$PTR

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

$PTP

.OPEN command	-	Device is initialized; leader is punched.
.CLOSE command	-	Device is reinitialized after outstanding I/O is complete; trailer is punched.
.WRS command	-	The specified bytes are output to the device, unedited.
.WRL command	-	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.

$TTR

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

APPENDIX A

ADDING DEVICE HANDLERS TO DSOS

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

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

2. A level which 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 DSOS library. The DSOS Parameters and the DOS 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 DSOS implementation.
Level 2	In addition to the requirements of 1, provide a DSOS 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 main program of DSOS (SOS) will perform these critical functions, using the supplied information.

No source level changes to the DSOS library rograms are necessary. The links to the user supplied tables currently exist in the DSOS main program in the form of unresolved external declarations. If these unresolved externals become resolved in the relocatable load module, then DSOS 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 DSOS compatibility with optional devices.

DEVICE CONTROL TABLE (DCT)

Each DSOS 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 24 octal locations (displacements 0-23 from the DCT layout description in the DSOS Parameter Tape.) These displacements and their meanings are as follows:

Equivalence	Displacement	Explanation
Ø	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 (see page A-5.) The mask bits are defined in the DSOS parameter tapes. Must be assembled into the table.
2	DCTCH	The device characteristics from the DOS User Parameter Tape. Must be assembled into the table. 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 DSOS table .OPPP.
4	DCTIS	The address of the interrupt service routine. Must be assembled into the table. (See DSOS Interrupt Handling, page A-7.)
5	DCTIL	The interrupt frame links. Points to the device control table of the last interrupted device. This word is maintained by the DSOS interrupt service routine.
6	DCTDT	The Command Dispatch Table address for this device. The Command Dispatch Table must be ordered in the following manner:

6	(Continued)	Ø - 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. Must be assembled into the table; see Device Start, Stop, and Dispatch Routines.
7	DCTST	The address of the device start routine. Must be assembled into the table. (See Device Start, Stop, and Dispatch routines.)
10	DCTSP	The address of the device stop routine. 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 DSOS subroutines. Three flags are currently defined: DCACT = 1B15 - Device is active (executing I/O). Must be off to perform a DSOS reset. DCACP = 1B8 - A keyboard input device may accept a character. DCKMD = 1BØ - A keyboard input device is in echo mode. Echo the input character.
12	DCTBS	The size of the device buffer (in bytes for character devices, in words for full word devices). Must be assembled into the table.
13	DCTBF	Buffer first byte (word) address. If assembled as -1, then DSOS will allocate a buffer area (using the buffer size parameter given above),

13	DCTBF (Continued)	and initialize both this word and the next word on a .SYSI. If other than -1, this word is assumed to be the appropriate beginning buffer address.
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 DSOS 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 DSOS 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 DSOS subroutines.
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 DOS 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 DSOS subroutines.

DEVICE CONTROL TABLE (DCT) (Continued)

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 DSOS subroutines.
22	DCTSØ	Interrupt frame pointer. This word points to a 7-word block of memory in which the machine state is saved whenever an interrupt from this device is serviced. This word is initialized by a .SYSI.
23	DCTS1	Spare word. Not currently used in DSOS.

DCT displacements 6-21 (DCTDT - DCTRC) are not referenced by Level 1 devices, with the exception of DCTFL, bit 15, which must be off to perform a DSOS reset command. 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 DSOS routines are invoked then any or all of these elements may be referenced.

DEVICE PRIORITY TABLE

This table is referenced by the DSOS initialization routine (.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 DSOS devices are searched for a matching code on an interrupt. This table is normally embedded in the DSOS main program. If the external normal .OPPP in the DSOS main program is resolved, however, then a user supplied table is used to establish the links. The DSOS main program table is set up as follows:

```
.PTRP
.CDRP
.TTRP
.PTPP
.LPTP
.PLTP
.TTOP
 Ø
```

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

DEVICE PRIORITY TABLE (Continued)

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 $CDR 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, .PLTP, .TTOP

        .OPPP:  .XXXP
                .PTRP
                .CDRP
                .YYYP
                .TTRP
                .PTPP
                .LPTP
                .PLTP
                .TTOP
                 Ø

                .
                .
                .
        .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 DSOS command dispatch routine. If a DSOS I/O command has referenced a channel number outside of the legal DSOS range and the DSOS 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 20 (HCHNO + 1 from the DSOS parameters). Thus if devices XXX and YYY were being incorporated into DSOS for Level 2 compatibility and XXX were assigned to channel 20 and YYY to channel 21, the critical program declarations for this table would be as follows:

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.

DSOS INTERRUPT HANDLING

When an interrupt is taken, a DSOS module preserves the interrupted machine state. The interrupt frame pointer, DCTSØ, 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 DSOS interrupt module.

The DSOS stack mechanism may not be invoked at interrupt processing time. The DSOS modules .IBUF and .OBUF may be called, however, since they do not require a stack frame (see DSOS 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 THE CHARACTER IN THE DEVICE'S BUFFER.
          JMP     .           ;AN IMPOSSIBLE RETURN-BUFFER ALREADY FULL.
          NIOS    PTR         ;RESTART THE $PTR; THE BUFFER IS NOT FULL YET.
          JMP     @.+1        ;DON'T RESTART $PTR; THE BUFFER BECAME FULL.
          .EXTN   .DISM
          .DISM               ;RETURN TO THE DSOS INTERRUPT MODULE.
          .EXTN   .IBUF
ADRIB:    .IBUF               ;ENTRY POINT
```

```
PTPS:    NIOC     PTP        ;CLEAR DONE.
         JSR      @ADROB     ;RETRIEVE THE NEXT CHARACTER
                             ;FROM THE DEVICE'S BUFFER.
         DOAS     1,PTP      ;RESTART THE DEVICE AND SEND
                             ;THIS CHARACTER IF THE RETURN CAME
                             ;HERE.
         JMP      @.+1       ;OTHERWISE DON'T RESTART.
         .DISM               ;RETURN TO THE DSOS 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       Ø,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       Ø,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       Ø,3       ;RETURN
```

Device Dispatch Routines

The device dispatch table address is at displacement DCTDT in the DCT, previously described. If the global DSOS subroutines 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 as follows:

ACØ	-	page zero displacement, "CACØ".
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 DSOS routines are used, then they may be invoked directly (by assembling their addresses into the dispatch table.)

DSOS LINKAGE

A simple stack mechanism is employed in DSOS 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 DSOS interrupt dispatch routine performs this service.

The linkage scheme used in DSOS makes use of several page zero locations and of a fixed block of core assembled into the DSOS module. The size of this core block permits a "depth" of six calls from the common DSOS 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.
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".)
ACØ	-	Contents of accumulator Ø of 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.
OACØ	-	Contents of caller's accumulator Ø.
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
```

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

```
STA        3,@CSP
```

3. Routine B may anytime 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 DSOS SUBROUTINES

The global routines defined as entries in the DSOS main program, 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

These routines are available for use with any programs loaded with DSOS. 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:    .XXX
```

Arguments: ACØ = Byte address of the file name if the device is an intervention device. Otherwise, ACØ is ignored.

AC2 = Pointer to the DCT.

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

Description: 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.

.WRSE

Calling Sequence:	JSR .WRSE
Arguments:	"CACØ"* = 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:	CACØ = 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 DSOS parameter tape.

.RDSE

Calling Sequence: JSR .RDSE

Arguments:

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

Return Sequence: A return to calling location + 1 indicates either end of file or illegal command for 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:

CACØ:	=	Beginning byte address for the transfer.
AC2	=	Pointer to the 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 the 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. a carriage return
2. a form feed

Nulls, rubouts, and line feeds are ignored.

* UST Displacement from the DOS User Parameters.

.RCHR

Calling Sequence:	JSR .RCHR
Arguments:	AC2 = Pointer to the 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 ACØ which is destroyed.

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

.IBUF (Continued)

Description: The byte (word) is placed into 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 ouput 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 ACØ 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: ACØ = Destination byte address.
AC1 = Right justified byte.

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

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

.LDB

Calling Sequence:	JSR .LDB
Arguments:	ACØ = 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.

```
;********************************************************************
;
; NAME: PARA.SR                                  PART NUMBER: 090-002278
;
;
; DESCRIPTION: STAND-ALONE PARAMETERS
;
;
; DOCUMENTATION REFERENCES:
;
;       TITLE                                    DOCUMENT NO.
;
;       DOS-COMPATIBLE SOS                       007-000751
;
;
; REVISION HISTORY:
;
;       REV.            DATE
;
;       00              08/17/73
;
;
; COPYRIGHT (C) DATA GENERAL CORPORATION, 1973
; ALL RIGHTS RESERVED.
;********************************************************************
```

```
;
;       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

;                    INTERRUPT FRAME TEMPLATE

.DUSR   IAC0=   0
.DUSR   IAC1=   1
.DUSR   IAC2=   2
.DUSR   IAC3=   3
.DUSR   IPC=    4
.DUSR   IRLOC=  5
.DUSR   IMSK=   6
.DUSR   IFRL=   7       ;INTERRUPT FRAME LENGTH
```

```
; 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

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

.DUSR DCTBS=     12       ; BUFFER SIZE ( BYTES )
.DUSR DCTBF=     13       ; BUFFER FIRST ADDRES (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 DCTS0=     22       ; DEVICE SPECIAL WORD 0
.DUSR DCTS1=     23       ; DEVICE SPECIAL WORD 1

.DUSR LCHNO=     6        ; LOWEST LEGAL CHANNEL #
.DUSR HCHNO=     17       ; HIGHEST LEGAL CHANNEL #
                          ; NOTE - ONE OR BOTH OF THESE EQUI-
                          ;        VALENCES MAY BE CHANGED TO ADD
                          ;        DEVICE DRIVERS
```

DOS USER PARAMETERS

```
; DEFINE THE SYSTEM 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 ACCUMULATOR
.DUSR SSAC1=     -1
.DUSR SSAC2=     0        ; (DON'T MODIFY THIS DISPLACEMENT!!)
```

```
;
; 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 UNASSI
.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
                           ;(LEAVE "UFTCH" AS LAST WORD!)

.DUSR UFTEL=UFTCH-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 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/0 IN PROGRESS
.DUSR QTLCK=1B11         ;BUFFER LOCKED
.DUSR QTCMD=1B10         ;COMMAND = 1 = READ, 0 = WRITE
.DUSR QTEMO=1B9          ;ERROR MODE (MAG TAPE)
.DUSR QTIND=1B8          ;INDIRECT DRIVER MODE SW.


;
; 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=-60.         ;RINGIO 1 MS. LOOP TIME (SN)
.DUSR SCSYS=1            ;DEVICE ADDRESS FOR SYS.DR
.DUSR SCMAP=2            ;DEVICE ADDRESS FOR MAP.DIR
.DUSR SCSVB=SCMAP+1      ;4 CONTIGUOUS BLOCKS FOR CORE IMAGES
.DUSR SCSNO=4            ;NUMBER OF 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 CADZ=40            ;CA LOCATION IN BOOTSTRAP
.DUSR LADZ=CADZ+1        ;LA LOCATION IN BOOTSTRAP
.DUSR SCFUL=LADZ+1
.DUSR SCPAR=SCFUL+1
.DUSR SCKEY=SCPAR+1
```

```
; 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 D
.DUSR ERRTN=     23         ; ATTEMPT TO RESTORE A NON-EXISTENT IMAG
.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
```

```
; 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   PNCUI=  21*POFFS+PNOP   ; UNKNOWN INTERRUPT
                                ; DEVICE CODE IN AC0
.DUSR   PNCSO=  22*POFFS+PNOP   ; SYSTEM STACK OVERFLOW
.DUSR   PNCDW=  23*POFFS+PNOP   ; CRITICAL DISK WRITE ERRORS
.DUSR   PNCDR=  24*POFFS+PNOP   ; CRITICAL DISK READ  ERRORS
.DUSR   PNCDE=  25*POFFS+PNOP   ; CRITICAL DISK READ/WRITE ERROR
.DUSR   PNCRR=  26*POFFS+PNOP   ; RUNAWAY READER
.DUSR   PNCMT=  27*POFFS+PNOP   ; MTA CONTROLER ERROR
```

```
        ; DEFINE THE CHARACTERISTICS

.DUSR   DCCPO=  1B15    ; DEVICE REQUIRING LEADER/TRAILER
.DUSR   DCCGN=  1B14    ; GRAPHICAL OUTPUT DEVICE WITHOUT TABBIN
                        ; HARDWARE
.DUSR   DCIDI=  1B13    ; INPUT DEVICE REQUIRING OPERATOR INTERV
.DUSR   DCCNF=  1B12    ; OUTPUT DEVICE WITHOUT FORM FEED HARDWA
.DUSR   DCTO=   1B11    ; TELETYPE OUTPUT DEVICE
.DUSR   DCKEY=  1B10    ; KEYBOARD DEVICE
.DUSR   DCNAF=  1B9     ; OUTPUT DEVICE REQUIRING NULLS AFTER FO
.DUSR   DCRAT=  1B08    ; RUBOUTS AFTER TABS REQUIRED
.DUSR   DCPCK=  1B07    ; DEVICE REQUIRING PARITY CHECK
.DUSR   DCLAC=  1B06    ; REQUIRES LINE FEEDS AFTER CARRIAGE RTN
.DUSR   DCFWD=  1B04    ; FULL WORD DEVICE (ANYTHING GREATER THA
.DUSR   DCFFO=  1B03    ; FORM FEEDS ON OPEN
.DUSR   DCLTU=  1B02    ; CHANGE LOWER CASE ASCII TO UPPER
.DUSR   DCC80=  1B01    ;READ 80 COLUMS
.DUSR   DCDIR=  1B00    ; DIRECTORY DEVICE
```

```
; USER STATUS TABLE (UST) TEMPLATE

.DUSR   UST=     400     ; START OF USER STATUS AREA

.DUSR   USTPC=   0       ; PROGRAM COUNTER (LEAVE AT DISPLACEMENT
.DUSR   USTZM=   1       ; ZMAX
.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   USTIS=   14      ;INTERRUPT SERVICE WORD
.DUSR   USTWA=   15      ; I/O WAIT RETURN
.DUSR   USTRS=   16      ; I/O COMPLETION RESTORE
                         ; DEFINE 4 SPARE WORDS
.DUSR   USTA0=   23      ; SAVE STORAGE FOR AC0
.DUSR   USTA1=   24      ; AC1
.DUSR   USTA2=   25      ; AC2
.DUSR   USTA3=   26      ; AC3
.DUSR   USTCY=   27      ; CARRY

.DUSR   USTEL=   30      ; ENTRY LENGTH
.DUSR   USTEC=   1       ; ENTRY COUNT

.DUSR MXFNO=10           ;MAX NUMBER OF FILE TABLES
.DUSR UFPT=UST+USTEL     ;USER FILE POINTER TABLE
.DUSR UFTEC=MXFNO        ;ENTRY COUNT
.DUSR UFT=UFPT+UFTEC     ;UFT'S
```

INDEX

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: ____________________	
	____________________	

cut along dotted line

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

DATA GENERAL
CORPORATION

Southboro,
Massachusetts 01772
(617) 485-9100