DNOS

Reference Handbook

Part No. 2270505-9701 *B
March 1985

TEXAS INSTRUMENTS

LIST OF EFFECTIVE PAGES

INSERT LATEST CHANGED PAGES AND DISCARD SUPERSEDED PAGES

Note: The changes in the text are indicated by a change number at the bottom of the page and a vertical bar in the outer margin of the changed page. A change number at the bottom of the page but no change bar indicates either a deletion or a page layout change.

DNOS Reference Handbook (2270505-9701)

Original Issue . . . . . October 1982
Revision . . . . . November 1983
Change 1 . . . . . March 1985

Total number of pages in this publication is 250 consisting of the following:

The computers, as well as the programs that TI has created to use with them, are tools that can help people better manage the information used in their business; but tools—including TI computers—cannot replace sound judgment nor make the manager's business decisions.

Consequently, TI cannot warrant that its systems are suitable for any specific customer application. The manager must rely on judgment of what is best for his or her business.

PAGE NO.	CHANGE NO.
Cover	1
Effective Pages	1
Eff. Pages (Cont.)	1
iii - xiii	0
ix	1
x - xii	0
1-1 - 1-2	0
1-3	1
1-4 - 1-5	0
1-6	1
1-7 - 1-24	0
2-1 - 2-4	0
2-5	1
2-6	0
2-6A/2-6B	1
2-7 - 2-14	0
3-1 - 3-32	0
3-33	1
3-34 - 3-46	0
4-1 - 4-8	0
4-9	1
4-10	0
4-11	1
4-12	0
5-1 - 5-18	0
5-19	1
5-20 - 5-24	0
5-25	1
5-26 - 5-29	0
5-30 - 5-36	1
5-37 - 5-38	0
5-39	1
5-40 - 5-41	0
5-42	1
5-43	0
5-44	1
5-45	0
5-46 - 5-48	1
5-49 - 5-51	0
5-52	1
5-53 - 5-55	0
5-56	1
5-57 - 5-71	0
5-72	1
5-73 - 5-75	0
5-76	1
5-77 - 5-78	0
5-79	1
5-80 - 5-92	0
5-93	1
5-94 - 5-96	0
6-1 - 6-6	0
7-1	0
7-2	1
7-3 - 7-4	0
A-1 - A-18	0
Index-1 - Index-10	0
User's Response	1
Business Reply	1
Inside Cover	1
Cover	1

Preface

Before using this handbook, you should have a basic understanding of DNOS operations, directory structures, and pathnames. If you need detailed information about specific items, refer to the manual or manuals suggested in each section.

This handbook contains the following sections:

Section

1 General Operations — Describes log-on and log-off procedures and useful command sequences and procedures.

2 Using SCI — Lists SCI function keys and SCI primitives.

3 Assembly Language Instructions and Formats — Gives assembly language instructions and formats and the object code format.

4 Program Development — Provides information on how to set up and execute a program and lists link editor commands and debugger commands.

5 SVC Blocks — Presents the structure of each SVC block and lists the type of information you must provide for each.

6 ASCII Character Codes — Lists ASCII characters and their equivalent hexadecimal codes.

7 Job and Task States — Lists job and task state codes.

Appendix

A Keycap Cross-Reference — Lists the generic key names and shows the corresponding keycap labels on each terminal supported by DNOS. Throughout this handbook, keys are referred to by generic name.

Contents

2 — Using SCI

Appendixes

Illustrations

Tables

1

General Operations

1.1 INTRODUCTION

This section describes general operations, logging on, logging off, and useful command sequences and procedures.

1.2 GENERIC KEY NAMES

The names used in this handbook for individual keys on terminals are generic key names and apply to all terminals. In some cases the names on the keycaps of some terminals match the generic names, but in many cases they do not. Table A-1 in Appendix A includes the generic key names and the corresponding keys for each terminal supported by DNOS. Appendix A also contains a reverse reference table showing commonly used keys on the Model 911 VDT and the corresponding generic key names.

1.3 LOGGING ON

Before you can use DNOS at a terminal, you must activate the System Command Interpreter (SCI). Since you can customize activation procedures to a particular application, the prompts shown in this paragraph do not appear in every case. These steps show the procedure for activating SCI at a terminal:

1. Turn on the terminal if it is not already on.

2. Press the Attention key.

3. Enter the exclamation mark (!).

4. If this is the first terminal to log on after loading DNOS, enter the year, month, day, hour, and minute as requested.

5. DNOS responds by displaying or printing the following message:

   DNOS 1.X.XX

   where:

   1.X.XX is the release version of DNOS.

6. If user identification is required, DNOS displays the following prompts:

   ```
   USER ID:
   PASSCODE:
   ```

   Type in your assigned user ID and press the Return key to signal DNOS that an entry has been made. The user ID must be all uppercase characters. Next, type in your assigned passcode and press the Return key to signal DNOS that another entry has been made. To preserve passcode security, the characters of the passcode are not displayed.

7. DNOS may respond by displaying the following prompt (if it is not predefined by the Modify Terminal Status (MTS) command).

   ```
   ACCOUNT ID:
   ```

Type in the assigned account ID and press the Return key to signal that an entry has been made. Your account ID must be a valid number found in the file .S$ACCVAL.

8. DNOS may respond by displaying the following prompt (if it is not predefined by the MTS command):

   JOB NAME:

   Type in a job name and press the Return key to signal DNOS that an entry has been made. A job name can be any alphanumeric string (maximum of eight characters) that starts with an alphabetic character or a dollar sign ($) and consists of only uppercase characters.

9. DNOS may respond by displaying the following prompt (if it is set for display by the MTS command):

   SYNONYM FILE PATHNAME:

   Enter the pathname of the file that contains the synonyms and logical names you want to use, or press the Return key to accept the default pathname (the file in which the synonyms for your user ID are normally kept). If a file is supplied, it must have been originally established by the Snapshot Name Definitions (SND) command.

10. If the job name entered is already in use with the same user ID, DNOS may respond with the following prompt:

    RECONNECT?:

    Type in YES or NO and press the Return key to signal that an entry has been made. YES specifies that the terminal is also to be associated with the job in use. NO specifies that the terminal is to be associated with a new job. An ACCOUNT ID validation is not performed when YES is specified.

11. If the log-on is successful, DNOS either displays the main menu and the SCI prompt ([ ]) or it displays the news file if one exists. If a news file is displayed, SCI waits for the Command key to be pressed. After the Command key is pressed, SCI displays the main menu and SCI prompt ([ ]) as shown in Figure 1-1. The main menu can be changed at the option of the systems programmer.

```
****************************************
**  T E X A S  I N S T R U M E N T S  **
**  D N O S  S Y S T E M               **
****************************************

 Command Groups:

/DEBUG  - Interactive Debugger          /OPER   - Operator Interface
/DEVICE - I/O Devices                   /PREXEC - Program Execution
/DIR    - Directories                   /PRFILE - Program Files
/EDIT   - Text Editor                   /SECURE - File Security
/FILE   - File Management               /SPOOL  - Spooler
/JOB    - Job Management                /STATUS - Status Reports
/LANG   - Language Support              /SYSCON - System Configuration
/LUNO   - Logical Unit Numbers          /SYSGEN - System Generation
/MSG    - Message Facilities            /SYSMGT - System Management
/NAME   - Synonyms and Logical Names    /TPD    - Teleprinter Devices
/NET    - Networking Support            /VOLUME - Disk Volumes

[ ]
```

Figure 1-1. SCI Main Menu

12. To begin operating the terminal, enter the SCI commands that are available to you. If you enter a command that is not authorized for your user ID, SCI will display an appropriate error message.

13. While executing SCI commands, do not turn off the terminal. If the terminal is turned off, device errors are written to the system log and the system may loop in an attempt to complete the command.

1.4 LOGGING OFF

Follow these steps to log off.

1. Enter the Quit (Q) command.

2. If the text editor or debugger is active at the terminal, the Q command automatically prompts for the Quit Editor (QE) or Quit Debug (QD) command responses. If not, SCI terminates at this point.

3. After you enter the necessary responses to either the QE or QD command, enter the Q command again to terminate SCI at your terminal.

4. If a pending foreground operation is active when you enter the Q command, this message appears:

   U SCI — 0048 QUIT OPERATION NOT VALID WITH <yyy> PENDING

   where:

   <yyy> represents the name of the operation(s), such as SCU (System Configuration Utility).

   To erase the message, press the Return key. You must complete the operation by using the command that terminates the pending operation, such as Quit Configuration Utility Session (QSCU). Reenter the Q command to terminate SCI.

5. If a pending background task is active when you enter the Q command, the following message appears:

   U SCI — 0029 CANNOT QUIT WITH BACKGROUND TASK PENDING

   To erase the message, press the Return key. Allow the background task to complete. Reenter the Q command to terminate SCI at the terminal.

1.5 TEXT EDITOR FUNCTION KEYS

On the VDT keyboard, designated keys activate all of the edit control functions. These functions consist of operations that change the editing position in the file, insert and delete data, move the cursor, display or suppress line numbers, and change editing modes. Table 1-1 lists the edit control functions and their associated keys.

Table 1-1. Edit Control Functions on a VDT

Function	Generic Key Name
Return to command mode	Command
Roll (display) up	F1
Roll (display) down	F2
Enable/disable word wrap	F3[1]
Duplicate to tab	F4
Clear to tab	F5
Display/suppress line numbers	F6[2]
Edit mode/compose mode	F7[1]
Restore line	F8
New line (next line)	Return

Table 1-1. Edit Control Functions on a VDT (Continued)

Function	Generic Key Name
Forward tab	Forward Tab
Erase rest of line	Skip
Forward space one character	Next Character
Backspace one character	Previous Character
Back tab	Back Tab or Previous Field
Right margin	Next Field[3]
Left margin	Enter[3]
Erase characters on line	Erase Field
Delete line	Erase Input
Insert line	Initialize Input
Repeat	Repeat
Insert character	Insert Character
Delete character	Delete Character
Move cursor up	Previous Line
Move cursor down	Next Line
Move cursor right	Next Character
Move cursor left	Previous Character
Move cursor home	Home

Table 1-1. Edit Control Functions on a VDT (Continued)

Function	Generic Key Name
Enable/disable uppercase	Uppercase Lock[1]
Control character codes	Control

Notes:

[1] Alternates modes each time this key is pressed.

[2] Alternates display of line numbers (74 data characters per line) with no display of line numbers (80 data characters per line) each time this key is pressed.

[3] The Next Field key and Enter key also function as horizontal scroll keys.

For further information, refer to the *DNOS Text Editor Manual.*

1.6 USEFUL COMMAND SEQUENCES AND PROCEDURES

Some procedures require a special sequence of commands. The following paragraphs identify some of these procedures. If you need to view the SCI prompts, enter the command at your terminal. If you want a detailed explanation of the prompts, refer to the *DNOS System Command Interpreter (SCI) Reference Manual*.

1.6.1 Printing a KIF

You cannot print the contents of a Key Indexed File (KIF) with the Print File (PF) command. Copy the KIF to a sequential file by using the Copy KIF to Sequential File (CKS) command. The CKS command copies the contents of a KIF to a sequential file in the order you specify. To specify the order, name a key by which the CKS command can read the file. For example, if you specify the number 2 for the KEY prompt, and that key contains zip codes, CKS reads the KIF by zip code as it is copied to the sequential medium. If you specify a device name in the form STxx or LPxx (where xx represents an integer) for the OUTPUT ACCESS NAME prompt, only the first 80 characters of each record are written to the device.

After you copy the KIF to a sequential file, you can execute the PF command and print the sequential file.

1.6.2 Showing a KIF

The Show File (SF) command allows you to view a KIF in the order of the primary key. If you want to view the file in another order, use the CKS command. The previous paragraph discusses this command in greater detail.

1.6.3 Spooler Recovery

An unusual series of events can cause the spooler to stop functioning without allowing you to restart it. You must reinitialize the spooler environment by performing these steps:

1. Use the Show Output Status (SOS) command to see which files are waiting to be printed and to see the current devices available to the spooler and the class names assigned to each. Write down this information for later use.

2. Enter the Execute Operator Interface (XOI) command and then press the Command key to return to SCI.

3. Use the List Jobs (LJ) command to find the ID of the spooler job.

4. Use the Kill Job (KJ) command to kill the spooler job.

5. Use the Delete File (DF) command to delete the spooler queue file (known by the logical name S$SDTQUE).

6. Restart the spooler job by using the Execute Job (XJ) command. Examine the file .S$ISBTCH to see what responses your system uses to start the spooler job. The most common responses are shown in the following example. If your system uses different parameters from the parameters in this example, see the discussion concerning the spooler in the *DNOS Systems Programmer's Guide* for additional details.

```
[ ]XJ
EXECUTE JOB
                              SITE NAME:
                              JOB NAME:  SPOOLER
                 PROGRAM FILE PATHNAME:  S$UTIL
                       TASK ID OR NAME:  04D
                                 PARM1:  1
                                 PARM2:  1
                            STATION ID:
            SYNONYM TABLE PATHNAME:  .S$USER.SPOOLER.SYN
                              PRIORITY:  5
                              JCA SIZE:  MEDIUM

EXECUTE JOB
                    USE CURRENT USER ID?:  NO
```

EXECUTE JOB

USER ID: SPOOLER
PASSCODE:
ACCOUNT ID:

7. Use the Modify Spooler Device (MSD) command and the list you made in step 1 to reassign all devices and classes to the spooler.

8. Execute the PF command for each of the files that the SOS command showed as waiting to be printed.

9. If you no longer have need to be the system operator, enter the Quit Operator Interface (QOI) command.

1.6.4 Enabling Devices for the Spooler

Until you assign device names, class names, and forms for each printer available to the spooler, the spooler cannot process print requests. Use the MSD command to assign printers and classes of printers to the spooler.

If you are uncertain of the assigned parameters, you can view the attributes of spooler devices by executing the SOS command. This command lists the device, form, status, and class name of a specified device or of all devices assigned to the spooler. If you want to change any of these attributes, such as adding a device to or removing one from use by the spooler, use the MSD command.

1.6.5 Recovering Lost Output After a Paper Jam

If paper jams in the printer, you can recover damaged pages by following these steps:

1. Execute the Halt Output at Device (HO) command to discontinue queued output temporarily.

2. Unjam the paper.

3. Execute the Resume Output at Device (RO) command. Respond to the PAGE SKIP prompt with the number of pages the output is to skip backward. Be sure to precede the number with a minus sign (−). If you enter a null response to this prompt, the output resumes at the point at which it was discontinued. Otherwise, the output resumes and begins printing at the page that you specified.

1.6.6 Changing the Number of Directory Entries

Use the following procedure to increase or decrease the number of directory entries:

1. Use the Create Directory File (CFDIR) command to create a new directory with the desired number of entries.

2. Use the Copy Directory (CD) command to copy your original directory into the new directory created in step 1. The CD command automatically duplicates the files and directories in your original directory.

3. Use the Delete Directory (DD) command to delete the original directory.

4. Use the CFDIR command to create the original directory with the desired number of directory entries.

5. Use the CD command to copy the new directory into the original directory. The original directory now contains the original files and subdirectories and has the desired number of entries.

6. Use the DD command to delete the new directory you created in step 1.

NOTE

In a file security environment, changing the number of directory entries with this procedure causes any secured files to become secured differently. Your creation access group will have all access rights to the files, but no other groups will have any access rights. You will need to reestablish file security.

1.6.7 Copying a Program File

Use the CD command to copy a program file. Specify the program file you want to copy as the input pathname. Specify the destination directory as the output pathname. You receive the following message if you made only insertions (no deletions) to the program file you are copying and if a destination file does not exist:

PROGRAM FILE COPIED USING FAST COPY

1.6.8 Copying a Crash File

Since .S$CRASH is an image file, use the CD command to copy it. Specify .S$CRASH as the INPUT PATHNAME and a destination directory as the OUTPUT PATHNAME. This example creates the file .TEMPDIR.S$CRASH and copies from .S$CRASH to .TEMPDIR.S$CRASH.

```
[ ]CD
        COPY DIRECTORY
                    INPUT PATHNAME:   .S$CRASH
                   OUTPUT PATHNAME:   .TEMPDIR
              CONTROL ACCESS NAME:
               LISTING ACCESS NAME:
                           OPTIONS:   ADD
              EXECUTION MODE (F,B):   FOREGROUND
```

1.6.9 Recognizing a System Crash

You know the system has crashed when:

- No terminal in the system responds to keyboard input.
- The POWER, FAULT, IDLE, and RUN indicator lights on the front panel of the computer are lit.

You know the system is in a continuous loop when:

- No terminal in the system responds to keyboard input.
- The POWER and RUN indicator lights on the front panel of the computer are lit.

The pattern in the data lights on the front panel after a system crash represents a hexadecimal crash code. The *DNOS Messages and Codes Reference Manual* lists the crash codes.

System crashes occur as the result of critical hardware errors or failures in system software. When a crash occurs, analyze the crash dump to determine the cause. If you cannot determine the cause, send a copy of the dump on magnetic tape or disk, copies of the system link maps, and a description of the events preceding the crash to your customer representative.

1.6.10 Dumping the Contents of Memory to .S$CRASH

When a system crash occurs, the system idles, waiting for you to take action. To analyze the cause of the system crash, dump the contents of memory to the crash file as follows:

1. Press HALT on the front panel.
2. Press RUN on the front panel.

The file .S$CRASH now receives the contents of memory. When the dump is complete, the IDLE light comes on again. Perform an initial program load (IPL) as detailed in the next paragraph. You can examine the crash dump by using the Execute Crash Analysis Utility (XANAL) command. For more information about this command, refer to the *DNOS System Command Interpreter (SCI) Reference Manual* and the *DNOS System Programmer's Guide*.

If you cannot analyze the crash dump immediately, you can copy the contents of .S$CRASH to a file in another directory. Refer to Copying a Crash File in this section. Make this copy as soon as possible after you perform the IPL on the system. By having a copy of .S$CRASH, you can safely perform another dump, if necessary, without losing the information from the previous one.

1.6.11 Performing an IPL After a Crash

After you dump the contents of memory following a system crash, you must perform an IPL to reactivate the system. Use this procedure:

1. Press HALT on the front panel.
2. Press LOAD on the front panel. DNOS is now loaded into memory.
3. Log on to a terminal.

4. Optionally, enter the PF command to print the system log files (.S$LOG1 and .S$LOG2).

5. Optionally, copy the system crash file (.S$CRASH) to another file by using the CD command. Refer to Copying a Crash File in this section.

If the problem occurs again, contact your customer representative. You should send the representative a copy of the crash file on magnetic tape or disk, system link maps, and a description of the events preceding the crash.

1.6.12 Forcing a System Crash

Certain problems in the system do not result in a system crash, but they prevent useful work on the system. The system malfunctions, but the FAULT light is not lit. For example, a system routine may hang in a loop or a deadlock condition. In such cases, force a system crash to obtain the crash dump for analysis.

Model 990 and Business System minicomputers have slightly different procedures for forcing a system crash.

1.6.12.1 Model 990 Minicomputers. The following procedure forces a crash on Model 990 minicomputers:

1. Press HALT twice.
2. Press the following in sequence:
   a. PC DISPLAY
   b. MA ENTER
   c. CLR
   d. MDE
   e. RUN
3. The system crashes with a crash code of >62.

4. Dump the contents of memory to the .S$CRASH file by pressing HALT and then RUN. Refer to Dumping The Contents of Memory to .S$CRASH in this section.

5. Perform an IPL.

1.6.12.2 Business Systems Minicomputers. The following procedure forces a crash on Business Systems minicomputers:

1. Press HALT 11 times.

2. Press RUN.

3. Dump the contents of memory to the .S$CRASH file by pressing HALT and then RUN. Refer to Dumping the Contents of Memory to .S$CRASH in this section.

4. Perform an IPL.

For further information, refer to the *DNOS Operations Guide*.

2

Using SCI

2.1 INTRODUCTION

SCI provides a set of commands and utilities for use with DNOS. SCI commands are grouped by function into menus. The SCI main menu displays the names of the command groups. By entering a command group menu name in response to the SCI prompt ([ ]), you can view the associated commands of a group. When using SCI, various function keys are used for the entry and control of SCI commands. Also, new SCI commands can be built with SCI primitives. This section describes SCI function keys and lists SCI primitives.

2.2 SCI FUNCTION KEYS

Table 2-1 lists the keys that have special terminal control functions when SCI is accepting user responses to field prompts on a terminal in VDT mode. Table 2-2 shows the special terminal control keys that can be used when an SCI command is executing while the terminal is in VDT mode. Table 2-3 describes additional procedures that can be used for Terminal Local File displays.

Table 2-1. VDT Mode Command Entry Keys

Key	Function
Home	Positions the cursor at the beginning of the response field of the first field prompt displayed on the screen. You can reenter responses to all of the field prompts. The Home key does not allow you to go back to a previously displayed set of field prompts of the same command.
Return	Used to accept the current response to the field prompt. It is also used to position the cursor in the response field of the next field prompt or to start command processing if the command has no more field prompts.
Skip	Deletes all characters to the right of the cursor when it is positioned in the response field of the current field prompt and moves the cursor position to the start of the response field of the next field prompt.
Next Field	Moves the cursor position to the start of the response field of the next field prompt. If there are no more field prompts, the command executes.
Previous Field	Moves the cursor position to the start of the previous response field.

Table 2-1. VDT Mode Command Entry Keys (Continued)

Key	Function
Erase Field	Deletes an entire response to a field prompt including any initial value supplied by the system. You can then enter a new response to the same field prompt.
Erase Input	Resets any initial values and goes to the first field prompt displayed on the terminal. The initial values are reset only for the set of field prompts currently displayed.
Command	Aborts a command currently accepting user responses and returns the terminal to command mode.
F4	Causes duplication of the corresponding field from the previous line. This key does not cause field termination. You can either accept the duplicated field by pressing the Return key or modify the duplicated field before accepting it.
Back Tab	Positions the cursor to the beginning of the response field for the current field prompt.

Table 2-1. VDT Mode Command Entry Keys (Continued)

Key	Function
Forward Tab	Moves the cursor to the next pathname component in the current response field. At the end of the pathname, the cursor is positioned to the beginning of the current response field.
Previous Line	Positions the cursor to the same character position in the response field of the previous field prompt. If there is no previous field prompt, the cursor position is unchanged.
Next Line	Positions the cursor to the same character position in the response field of the next field prompt. If there is no next field prompt, the cursor position is unchanged.
Previous Character	Moves the cursor one position to the left in the response field of the current field prompt.
Next Character	Moves the cursor one position to the right in the response field of the current field prompt.

Table 2-1. VDT Mode Command Entry Keys (Continued)

Key	Function
Enter	Enters responses to all field prompts as displayed. If a syntax error occurs, the cursor is positioned to the first character position of the field prompt response field containing the syntax error.
Repeat	Repeats the key function when used in conjunction with any key.
Insert Character	Allows you to insert new characters, beginning at the current cursor position in the response field of the current field prompt. Any characters currently in the response field are shifted to the right. Characters can be inserted until the response field is full.
Delete Character	Deletes a character beginning at the cursor position and shifts the remaining characters in the response field of the field prompt one character position to the left.

Table 2-2. Control Keys for Commands Executing in VDT Mode

Key	Function
Attention	Temporarily stops output at a terminal. Pressing any alphabetic or numeric key continues output.
Attention then Return	Returns an error to the executing task. Most DNOS tasks programmed by Texas Instruments abort when the error is returned to the task. However, you can program a task to continue execution if you desire.
Attention then Control and X	Aborts the executing commands that can be aborted to return control to the user. One task is killed each time the break key sequence is used. If only SCI is active, it is killed, the job at the terminal terminates, and the terminal is again available for use. If tasks other than SCI are active, the order in which tasks are killed is: foreground tasks, background tasks, and SCI. If this control key sequence is issued to a terminal not logged on to SCI, the log-on prompts appear, but no task is bid and the log-on prompts remain on the screen. (To log on SCI, perform the normal log-on procedure of pressing the attention key followed by the exclamation point.)

Table 2-2. Control Keys for Commands Executing in VDT Mode (Continued)

Key	Function
Attention then Print	Prints the contents of a video display terminal (VDT) on the nominated hardcopy printer attached to your DNOS system. Each VDT can have a different printer nominated as its output device. This task works with 911, 931, and 940 VDTs without any modification to the device service routines (DSRs) or the operating system. You can optionally add header information to your printout. Refer to the *DNOS System Programmer's Guide* for more information on the header option and printer selection. This currently does not work across a local area network.

Table 2-3. Additional Procedures for Terminal Local File Displays

Key	Function
+ Integer then Return	Entering a plus sign (+), an integer (decimal or hexadecimal), then pressing the Return key, positions the display of the file contents ahead the number of lines specified by the integer.
Integer then Return	Entering an integer (decimal or hexadecimal) and pressing the Return key positions the display of the file contents to the line specified by the integer.
– Integer then Return	Entering a minus sign (–), an integer (decimal or hexadecimal), and then pressing the Return key, positions the display of the file contents back the number of lines specified by the integer.
F1	Causes the display of the next portion of a file (the amount displayed depends on the type of terminal). When the end-of-file is reached, pressing the F1 key has no further effect.
F2	Causes the display of a previous portion of a file. If the beginning of the file is displayed, the F2 key has no effect.

Table 2-3. Additional Procedures for Terminal Local File Displays (Continued)

Key	Function
F3	Causes the display to scroll horizontally 10 columns to the left. Scrolling may continue through column 170. The bottom line of the VDT screen displays a scaling line, unless the display is positioned in column 1. The last character which can be shown is in character position 240 of the input record. The terminal must be in default VDT mode (see the MTS command) for this function to operate as described.
F4	Causes the display to scroll horizontally 10 columns to the right. The bottom line of the VDT screen displays a scaling line, unless the display is positioned in column 1. The terminal must be in default VDT mode (see the MTS command) for this function to operate as described.
F6	Causes line numbers to be displayed in the leftmost columns of the VDT screen. Pressing the F6 key again causes the line numbers to be removed from the screen. The terminal must be in default VDT mode (see the MTS command) for this function to operate as described.

For further information, refer to the *DNOS SCI Reference Manual*.

2.3 SCI PRIMITIVES

Table 2-4 lists the SCI primitive notations, and Table 2-5 lists the available primitives and their associated parameters. Primitives are used to build batch streams and new SCI commands.

Table 2-4. SCI Primitive Notation

Notation	Meaning
Uppercase	Enter the item as shown.
Lowercase	Enter an item of this type.
No marks	Indicates required items.
[ ]	Indicates optional items.
Item...item	You can use more than one item of this type. Separate items with commas.
/	Indicates alternate items.

Table 2-5. SCI Primitives

Primitive Command	Parameters
.PROC	*name*[(*full name*)][= *int*][,*field prompt list*]
.EOP	
.PROMPT	[(*full name*)][= *int*][,*field prompt list*]
.SYN	*name* = "*value*" ...*name* = "*value* "
.EVAL	[*mode*][= YES/NO,] *name* = *value*
.SPLIT	LIST = (*list*)[,FIRST = *name*][,REST = *name*] or LIST = "*string*" [,FIRST = *name*][,REST = *name*] [,CHARACTER = "*string*"][,POSITION = *int*][,STATUS = *name*]
.SVC	[*$name*]DATA/BYTE/TEXT = *value(s)*... [*$name*]DATA/BYTE/TEXT = *value(s)*

Table 2-5. SCI Primitives (Continued)

Primitive Command	Parameters
.IF	*op1, relation, op2*
.ELSE	
.ENDIF	
.LOOP	
.WHILE	*op1, relation, op2*
.REPEAT	
.UNTIL	*op1, relation, op2*
.EXIT	

Table 2-5. SCI Primitives (Continued)

Primitive Command	Parameters
.BID	TASK = *namelint*[,LUNO = *int*][,CODE = *int*] [,PROGRAM FILE = *acnm*][,PARMS = (*string...string*)] [,UTILITY]
.DBID	TASK = *namelint*[,LUNO = *int*][,CODE = *int*] [,PROGRAM FILE = *acnm*][,PARMS = (*string...string*)] [,UTILITY]
.QBID	TASK = *namelint*[,LUNO = *int*][,CODE = *int*] [,PROGRAM FILE = *acnm*][,PARMS = (*string...string*)] [,UTILITY]
.DATA	[*acnm*][,EXTEND[= YES/NO]][,SUBSTITUTION[= YES/NO]] [,REPLACE[= YES/NO]]
.EOD	
.STOP	[TEXT = *string*][,CODE = *int*]

Table 2-5. SCI Primitives (Continued)

Primitive Command	Parameters
.USE	[*pathname...pathname*]
.OPTION	[PROMPT[= *string*]][,MENU[= *name*]] [,PRIMITIVES[= YES/NO]][,LOWERCASE[= YES/NO]]
.MENU	[*menu name*]
.SHOW	*filename...filename*

For further information refer to the *DNOS Systems Programmer's Guide*.

3

Assembly Language Instructions and Formats

3.1 INTRODUCTION

The assembly language instructions and formats in this section are for the Model 990 and Business Systems computers. This section lists assembly language instructions numerically by hexadecimal operation code (opcode), and alphabetically by mnemonic opcode; and it includes the assembly language instruction formats. It also includes assembly language addressing modes, the status register, workspace pointer and registers, macro language statements, macro language variable components, and object code format.

3.2 ASSEMBLY LANGUAGE INSTRUCTION FORMATS

Figure 3-1 shows the assembly language instruction formats.

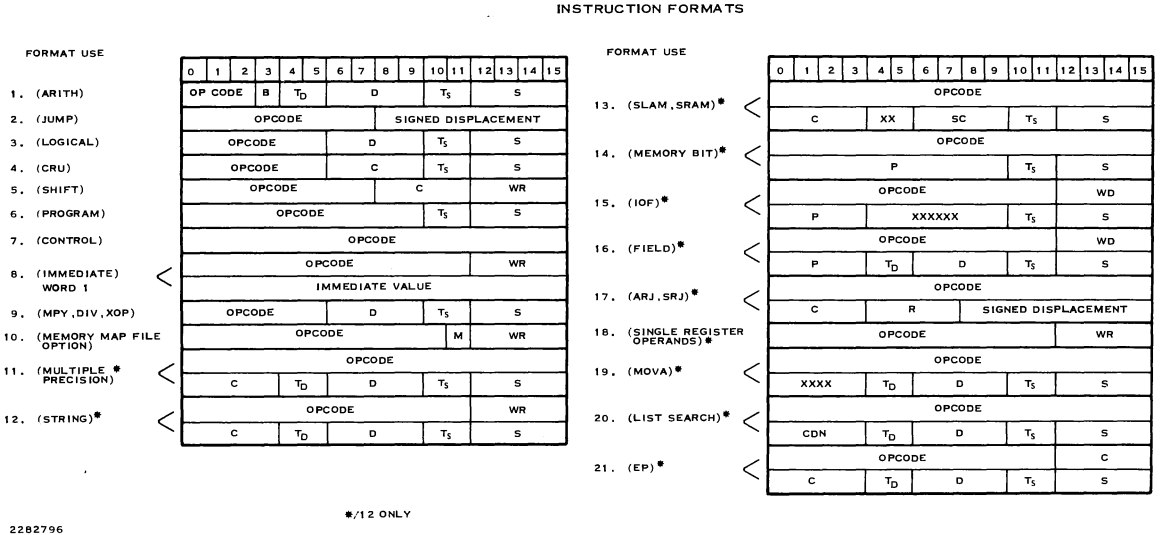


Figure 3-1. Assembly Language Instruction Formats

3.3 ADDRESSING MODES

The assembly language features eight addressing modes. Three of these modes—program counter relative addressing, CRU bit addressing, and immediate addressing—are special purpose addressing modes. For more information on these modes, read the *Assembly Language Reference Manual*. The remaining five modes in the instructions specify a general address for the source or destination operand. Table 3-1 lists these modes and shows how to use each in assembly language. Each addressing format describes the T field as Ts and Td.

Table 3-1. Addressing Modes

Addressing Mode	T Field Value	Example	Notes
Workspace Register	0	R5	
Workspace Register Indirect	1	*R7	
Workspace Register Indirect Autoincrement	3	*R7+	

Table 3-1. Addressing Modes (Continued)

Addressing Mode	T Field Value	Example	Notes
Symbolic Memory	2	@LABEL	1, 2
Indexed Memory	2	@LABEL(5)	1, 3

Notes:

1. The instruction requires an additional word for each T field value of 2. This word contains a memory address.
2. The assembler sets the S or D field to 0.
3. You cannot use workspace register 0 for indexing.

3.4 ASSEMBLY LANGUAGE BY OPCODE

Table 3-2 lists all assembly language instructions by hexadecimal opcode and includes the status bits affected. Numbers listed in the format column refer to the instruction formats presented in Figure 3-1.

Table 3-2. Hexadecimal Instruction Table

Hexadecimal Operation Code	Mnemonic Operation Code	Name	Format	Status Bits Affected
001C	SRAM	Shift Right Arithmetic Multiple Precision	13	0-3
001D	SLAM	Shift Left Arithmetic Multiple Precision	13	0-4
001E	RTO	Right Test for One	11	2
001F	LTO	Left Test for One	11	2
0020	CNTO	Count Ones	11	2
0021	SLSL	Search List Logical Address	20	2

Table 3-2. Hexadecimal Instruction Table (Continued)

Hexadecimal Operation Code	Mnemonic Operation Code	Name	Format	Status Bits Affected
0022	SLSP	Search List Physical Address	20	2
0023	BDC	Binary to Decimal ASCII Conversion	11	0-2, 4
0024	DBC	Decimal to Binary ASCII Conversion	11	0-2, 4
0025	SWPM	Swap Multiple Precision	11	0-2
0026	XORM	Exclusive OR Multiple Precision	11	0-2
0027	ORM	OR Multiple Precision	11	0-2
0028	ANDM	AND Multiple Precision	11	0-2
0029	SM	Subtract Multiple Precision Integer	11	0-4
002A	AM	Add Multiple Precision Integer	11	0-4
002B	MOVA	Move Address	19	0-2
002D	EMD	Execute Micro-Diagnostic	7	0-15

Table 3-2. Hexadecimal Instruction Table (Continued)

Hexadecimal Operation Code	Mnemonic Operation Code	Name	Format	Status Bits Affected
002E	EINT	Enable Interrupts	7	—
002F	DINT	Disable Interrupts	7	—
0030	STPC	Store Program Counter	18	—
0040	CS	Compare Strings	12	0-2
0050	SEQB	Search String for Equal Byte	12	0-2
0060	MOVS	Move String	12	0-2
0070	LIM	Load Interrupt Mask	18	12-15
0080	LST	Load Status Register	18	0-15
0090	LWP	Load Workspace Pointer	18	—
00A0	LCS	Load Writable Control Store	18	—
00B0	BLSK	Branch Immediate and Push Link to Stack	8	—

Table 3-2. Hexadecimal Instruction Table (Continued)

Hexadecimal Operation Code	Mnemonic Operation Code	Name	Format	Status Bits Affected
00C0	MVSR	Move String Reverse	12	0-2
00D0	MVSK	Move String From Stack	12	0-2
00E0	POPS	Pop String From Stack	12	0-2
00F0	PSHS	Push String From Stack	12	0-2
0140	BIND	Branch Indirect	6	—
0180	DIVS	Divide Signed	6	0-2, 4
01C0	MPYS	Multiply Signed	6	0-2
0200	LI	Load Immediate	8	0-2
0220	AI	Add Immediate	8	0-4
0240	ANDI	AND Immediate	8	0-2
0260	ORI	OR Immediate	8	0-2

Table 3-2. Hexadecimal Instruction Table (Continued)

Hexadecimal Operation Code	Mnemonic Operation Code	Name	Format	Status Bits Affected
0280	CI	Compare Immediate	8	0-2
02A0	STWP	Store Workspace Pointer	18	—
02C0	STST	Store Status	18	—
02E0	LWPI	Load Workspace Pointer Immediate	8	—
0300	LIMI	Load Interrupt Mask Immediate	8	12-15
0320	LMF	Load Memory Map File	10	—
0340	IDLE	Idle	7	—
0360	RSET	Reset	7	12-15
0380	RTWP	Return With Workspace Pointer	7	0-15
03A0	CKON	Clock On	7	—
03C0	CKOF	Clock Off	7	—

Table 3-2. Hexadecimal Instruction Table (Continued)

Hexadecimal Operation Code	Mnemonic Operation Code	Name	Format	Status Bits Affected
03E0	LREX	Load or Restart Execution	7	12-15
03F0	EP	Extended Precision	21	0-2
0400	BLWP	Branch and Load Workspace Pointer	6	—
0440	B	Branch	6	—
0480	X	Execute	6	—
04C0	CLR	Clear	6	—
0500	NEG	Negate	6	0-4
0540	INV	Invert	6	0-2
0580	INC	Increment	6	0-4
05C0	INCT	Increment by Two	6	0-4
0600	DEC	Decrement	6	0-4

Table 3-2. Hexadecimal Instruction Table (Continued)

Hexadecimal Operation Code	Mnemonic Operation Code	Name	Format	Status Bits Affected
0640	DECT	Decrement by Two	6	0-4
0680	BL	Branch and Link	6	—
06C0	SWPB	Swap Bytes	6	—
0700	SETO	Set to One	6	—
0740	ABS	Absolute Value	6	0-4
0780	LDS	Long Distance Source	6	—
07C0	LDD	Long Distance Destination	6	—
0800	SRA	Shift Right Arithmetic	5	0-3
0900	SRL	Shift Right Logical	5	0-3
0A00	SLA	Shift Left Arithmetic	5	0-4
0B00	SRC	Shift Right Circular	5	0-3

Table 3-2. Hexadecimal Instruction Table (Continued)

Hexadecimal Operation Code	Mnemonic Operation Code	Name	Format	Status Bits Affected
0C00	CRI	Convert Real to Integer	7	0-4
0C01	CDI	Convert Double Precision Real to Integer	7	0-4
0C02	NEGR	Negate Real	7	0-2
0C03	NEGD	Negate Double Precision	7	0-2
0C04	CRE	Convert Real to Extended Integer	7	0-4
0C05	CDE	Convert Double Precision Real to Extended Integer	7	0-4
0C06	CER	Convert Extended Integer to Real	7	0-2
0C07	CED	Convert Extended Integer to Double Precision Real	7	0-2
0C08	NRM	Normalize	11	0-2
0C09	TMB	Test Memory Bit	14	2

Table 3-2. Hexadecimal Instruction Table (Continued)

Hexadecimal Operation Code	Mnemonic Operation Code	Name	Format	Status Bits Affected
0C0A	TCMB	Test and Clear Memory Bit	14	2
0C0B	TSMB	Test and Set Memory Bit	14	2
0C0C	SRJ	Subtract From Register and Jump	17	—
0C0D	ARJ	Add to Register and Jump	17	—
0C0E/ 0C0F	XIT	Exit From Floating Point Interpreter	7	—
0C10	INSF	Insert Field	16	0-2
0C20	XV	Extract Value	16	0-2
0C30	XF	Extract Field	16	0-2
0C40	AR	Add Real	6	0-4
0C80	CIR	Convert Integer to Real	6	0-2

Table 3-2. Hexadecimal Instruction Table (Continued)

Hexadecimal Operation Code	Mnemonic Operation Code	Name	Format	Status Bits Affected
1600	JNE	Jump if Not Equal	2	—
1700	JNC	Jump if No Carry	2	—
1800	JOC	Jump on Carry	2	—
1900	JNO	Jump if No Overflow	2	—
1A00	JL	Jump if Logical Low	2	—
1B00	JH	Jump if Logical High	2	—
1C00	JOP	Jump if Odd Parity	2	—
1D00	SBO	Set CRU Bit to Logic One	2	—
1E00	SBZ	Set CRU Bit to Logic Zero	2	—
1F00	TB	Test Bit	2	2
2000	COC	Compare Ones Corresponding	3	2

Table 3-2. Hexadecimal Instruction Table (Continued)

Hexadecimal Operation Code	Mnemonic Operation Code	Name	Format	Status Bits Affected
0EC0	SD	Subtract Double Precision Real	6	0-4
0F00	MD	Multiply Double Precision Real	6	0-4
0F40	DD	Divide Double Precision Real	6	0-4
0F80	LD	Load Double Precision Real	6	0-2
0FC0	STD	Store Double Precision Real	6	0-2
1000	JMP	Unconditional Jump	2	—
1100	JLT	Jump if Less Than	2	—
1200	JLE	Jump if Low or Equal	2	—
1300	JEQ	Jump if Equal	2	—
1400	JHE	Jump if High or Equal	2	—
1500	JGT	Jump if Greater Than	2	—

Table 3-2. Hexadecimal Instruction Table (Continued)

Hexadecimal Operation Code	Mnemonic Operation Code	Name	Format	Status Bits Affected
0CC0	SR	Subtract Real	6	0-4
0D00	MR	Multiply Real	6	0-4
0D40	DR	Divide Real	6	0-4
0D80	LR	Load Real	6	0-2
0DC0	STR	Store Real	6	0-2
0E00	IOF	Invert Order of Field	15	—
0E10	SNEB	Search String for Not Equal Byte	12	0-2
0E20	CRC	Cyclic Redundancy Code Calculation	12	2
0E30	TS	Translate Strings	12	0-2
0E40	AD	Add Double Precision Real	6	0-4
0E80	CID	Convert Integer to Double Precision Real	6	0-2

Table 3-2. Hexadecimal Instruction Table (Continued)

Hexadecimal Operation Code	Mnemonic Operation Code	Name	Format	Status Bits Affected
0EC0	SD	Subtract Double Precision Real	6	0-4
0F00	MD	Multiply Double Precision Real	6	0-4
0F40	DD	Divide Double Precision Real	6	0-4
0F80	LD	Load Double Precision Real	6	0-2
0FC0	STD	Store Double Precision Real	6	0-2
1000	JMP	Unconditional Jump	2	—
1100	JLT	Jump if Less Than	2	—
1200	JLE	Jump if Low or Equal	2	—
1300	JEQ	Jump if Equal	2	—
1400	JHE	Jump if High or Equal	2	—
1500	JGT	Jump if Greater Than	2	—

Table 3-2. Hexadecimal Instruction Table (Continued)

Hexadecimal Operation Code	Mnemonic Operation Code	Name	Format	Status Bits Affected
0CC0	SR	Subtract Real	6	0-4
0D00	MR	Multiply Real	6	0-4
0D40	DR	Divide Real	6	0-4
0D80	LR	Load Real	6	0-2
0DC0	STR	Store Real	6	0-2
0E00	IOF	Invert Order of Field	15	—
0E10	SNEB	Search String for Not Equal Byte	12	0-2
0E20	CRC	Cyclic Redundancy Code Calculation	12	2
0E30	TS	Translate Strings	12	0-2
0E40	AD	Add Double Precision Real	6	0-4
0E80	CID	Convert Integer to Double Precision Real	6	0-2

Table 3-2. Hexadecimal Instruction Table (Continued)

Hexadecimal Operation Code	Mnemonic Operation Code	Name	Format	Status Bits Affected
2400	CZC	Compare Zeros Corresponding	3	2
2800	XOR	Exclusive OR	3	0-2
2C00	XOP	Extended Operation	9	6
3000	LDCR	Load Communication Register	4	0-2, 5
3400	STCR	Store Communication Register	4	0-2, 5
3800	MPY	Multiply	9	—
3C00	DIV	Divide	9	4
4000	SZC	Set Zeros Corresponding	1	0-2
5000	SZCB	Set Zeros Corresponding, Byte	1	0-2, 5
6000	S	Subtract Words	1	0-4
7000	SB	Subtract Bytes	1	0-5

Table 3-2. Hexadecimal Instruction Table (Continued)

Hexadecimal Operation Code	Mnemonic Operation Code	Name	Format	Status Bits Affected
8000	C	Compare Words	1	0-2
9000	CB	Compare Bytes	1	0-2, 5
A000	A	Add Words	1	0-4
B000	AB	Add Bytes	1	0-5
C000	MOV	Move Word	1	0-2
D000	MOVB	Move Byte	1	0-2, 5
E000	SOC	Set Ones Corresponding	1	0-2
F000	SOCB	Set Ones Corresponding, Byte	1	0-2, 5

3.5 ALPHABETICAL LIST OF ASSEMBLY LANGUAGE INSTRUCTIONS

Table 3-3 lists all assembly language instructions alphabetically by mnemonic opcode. Numbers listed in the format column refer to the instruction formats presented in Figure 3-1. An asterisk (*) preceding a mnemonic opcode indicates that the instruction is available only on the 990/12 or Business System 800 computer; two asterisks (**) indicate that the instruction is available on the 990/12, the 990/10A, and the Business System computers.

Table 3-3. Alphabetical List of Assembly Language Instructions

Mnemonic Operation Code	Name	Hexadecimal Operation Code	Format	Notes
A	Add Words	A000	1	
AB	Add Bytes	B000	1	
ABS	Absolute Value	0740	6	5
* AD	Add Double Precision Real	0E40	6	
AI	Add Immediate	0220	8	
* AM	Add Multiple Precision Integer	002A	11	
ANDI	AND Immediate	0240	8	

Table 3-3. Alphabetical List of Assembly Language Instructions (Continued)

Mnemonic Operation Code	Name	Hexadecimal Operation Code	Format	Notes
* ANDM	AND Multiple Precision	0028	11	
* AR	Add Real	0C40	6	
* ARJ	Add to Register and Jump	0C0D	17	
B	Branch	0440	6	
* BDC	Binary to Decimal ASCII Conversion	0023	11	
** BIND	Branch Indirect	0140	6	
BL	Branch and Link	0680	6	
* BLSK	Branch Immediate and Push Link to Stack	00B0	8	
BLWP	Branch and Load Workspace Pointer	0400	6	
C	Compare Words	8000	1	
CB	Compare Bytes	9000	1	
* CDE	Convert Double Precision Real to Extended Integer	0C05	7	
* CDI	Convert Double Precision Real to Integer	0C01	7	
* CED	Convert Extended Integer to Double Precision Real	0C07	7	
* CER	Convert Extended Integer to Real	0C06	7	

Table 3-3. Alphabetical List of Assembly Language Instructions (Continued)

Mnemonic Operation Code	Name	Hexadecimal Operation Code	Format	Notes
CI	Compare Immediate	0280	8	
* CID	Convert Integer to Double Precision Real	0E80	6	
* CIR	Convert Integer to Real	0C80	6	
CKOF	Clock Off	03C0	7	1, 4
CKON	Clock On	03A0	7	1, 4
CLR	Clear	04C0	6	
* CNTO	Count Ones	0020	11	
COC	Compare Ones Corresponding	2000	3	
* CRC	Cyclic Redundancy Code Calculation	0E20	12	
* CRE	Convert Real to Extended Integer	0C04	7	
* CRI	Convert Real to Integer	0C00	7	
* CS	Compare Strings	0040	12	
CZC	Compare Zeros Corresponding	2400	3	
* DBC	Decimal to Binary ASCII Conversion	0024	11	
* DD	Divide Double Precision Real	0F40	6	
DEC	Decrement	0600	6	

Table 3-3. Alphabetical List of Assembly Language Instructions (Continued)

Mnemonic Operation Code	Name	Hexadecimal Operation Code	Format	Notes
DECT	Decrement by Two	0640	6	
* DINT	Disable Interrupts	002F	7	1
DIV	Divide	3C00	9	
** DIVS	Divide Signed	0180	6	
* DR	Divide Real	0D40	6	
* EINT	Enable Interrupts	002E	7	1
* EMD	Execute Micro-Diagnostic	002D	7	1
* EP	Extended Precision	03F0	21	
IDLE	Idle	0340	7	1, 4
INC	Increment	0580	6	
INCT	Increment by Two	05C0	6	
* INSF	Insert Field	0C10	16	
INV	Invert	0540	6	
* IOF	Invert Order of Field	0E00	15	
JEQ	Jump if Equal	1300	2	
JGT	Jump if Greater Than	1500	2	
JH	Jump if Logical High	1B00	2	

Table 3-3. Alphabetical List of Assembly Language Instructions (Continued)

Mnemonic Operation Code	Name	Hexadecimal Operation Code	Format	Notes
JHE	Jump if High or Equal	1400	2	
JL	Jump if Logical Low	1A00	2	
JLE	Jump if Low or Equal	1200	2	
JLT	Jump if Less Than	1100	2	
JMP	Unconditional Jump	1000	2	
JNC	Jump if No Carry	1700	2	
JNE	Jump if Not Equal	1600	2	
JNO	Jump if No Overflow	1900	2	
JOC	Jump on Carry	1800	2	
JOP	Jump if Odd Parity	1C00	2	
* LCS	Load Writable Control Store	00A0	18	
* LD	Load Double Precision Real	0F80	6	
LDCR	Load Communication Register	3000	4	
LDD	Long Distance Destination	07C0	6	1
LDS	Long Distance Source	0780	6	1
LI	Load Immediate	0200	8	1
* LIM	Load Interrupt Mask	0070	18	1

Table 3-3. Alphabetical List of Assembly Language Instructions (Continued)

Mnemonic Operation Code	Name	Hexadecimal Operation Code	Format	Notes
LIMI	Load Interrupt Mask Immediate	0300	8	1
LMF	Load Memory Map File	0320	10	1
* LR	Load Real	0D80	6	
LREX	Load or Restart Execution	03E0	7	1, 4
** LST	Load Status Register	0080	18	2
* LTO	Left Test for One	001F	11	
** LWP	Load Workspace Pointer	0090	18	
LWPI	Load Workspace Pointer Immediate	02E0	8	
* MD	Multiply Double Precision Real	0F00	6	
MOV	Move Word	C000	1	
* MOVA	Move Address	002B	19	
MOVB	Move Byte	D000	1	
* MOVS	Move String	0060	12	
MPY	Multiply	3800	9	
** MPYS	Multiply Signed	01C0	6	
* MR	Multiply Real	0D00	6	
* MVSK	Move String From Stack	00D0	12	

Table 3-3. Alphabetical List of Assembly Language Instructions (Continued)

Mnemonic Operation Code	Name	Hexadecimal Operation Code	Format	Notes
* MVSR	Move String Reverse	00C0	12	
NEG	Negate	0500	6	
* NEGD	Negate Double Precision	0C03	7	
* NEGR	Negate Real	0C02	7	
* NRM	Normalize	0C08	11	
ORI	OR Immediate	0260	8	
* ORM	OR Multiple Precision	0027	11	
* POPS	Pop String From Stack	00E0	12	
* PSHS	Push String From Stack	00F0	12	
RSET	Reset	0360	7	1, 4
* RTO	Right Test for One	001E	11	
RTWP	Return With Workspace Pointer	0380	7	2
S	Subtract Words	6000	1	
SB	Subtract Bytes	7000	1	
SBO	Set CRU Bit to Logic One	1D00	2	
SBZ	Set CRU Bit to Logic Zero	1E00	2	
* SD	Subtract Double	0EC0	6	

Table 3-3. Alphabetical List of Assembly Language Instructions (Continued)

Mnemonic Operation Code	Name	Hexadecimal Operation Code	Format	Notes
* SEQB	Search String for Equal Byte	0050	12	
SETO	Set to One	0700	6	
SLA	Shift Left Arithmetic	0A00	5	
* SLAM	Shift Left Arithmetic Multiple Precision	001D	13	
* SLSL	Search List Logical Address	0021	13	
SLSP	Search List Physical Address	0022	20	1
* SM	Subtract Multiple Precision Integer	0029	11	
* SNEB	Search String for Not Equal Byte	0E10	12	
SOC	Set Ones Corresponding	E000	1	
SOCB	Set Ones Corresponding, Byte	F000	1	
* SR	Subtract Real	0CC0	6	
SRA	Shift Right Arithmetic	0800	5	
* SRAM	Shift Right Arithmetic Multiple Precision	001C	13	
SRC	Shift Right Circular	0B00	5	
* SRJ	Subtract from Register and Jump	0C0C	17	
* SRL	Shift Right Logical	0900	5	
STCR	Store Communication Register	3400	4	3

Table 3-3. Alphabetical List of Assembly Language Instructions (Continued)

Mnemonic Operation Code	Name	Hexadecimal Operation Code	Format	Notes
* STD	Store Double Precision Real	0FC0	6	
* STPC	Store Program Counter	0030	18	
* STR	Store Real	0DC0	6	
STST	Store Status	02C0	18	
STWP	Store Workspace Pointer	02A0	18	
SWPB	Swap Bytes	06C0	6	
* SWPM	Swap Multiple Precision	0025	11	
SZC	Set Zeros Corresponding	4000	1	
SZCB	Set Zeros Corresponding, Byte	5000	1	
TB	Test Bit	1F00	2	
* TCMB	Test and Clear Memory Bit	0C0A	14	
* TMB	Test Memory Bit	0C09	14	
* TS	Translate Strings	0E30	12	
* TSMB	Test and Set Memory Bit	0C0B	14	
X	Execute	0480	6	
* XF	Extract Field	0C30	16	
* XIT	Exit From Floating Point Interpreter	0C0E/ 0C0F	7	

Table 3-3. Alphabetical List of Assembly Language Instructions (Continued)

Mnemonic Operation Code	Name	Hexadecimal Operation Code	Format	Notes
XOP	Extended Operation	2C00	9	
XOR	Exclusive OR	2800	3	
* XORM	Exclusive OR Multiple Precision	0026	11	
* XV	Extract Value	0C20	16	

Notes:

1. The instruction is privileged.
2. When the privileged instruction bit of the status register (bit 7) is set to 1, RTWP and LST return only status bits 0 through 5 to the status register.
3. When the privileged instruction bit of the status register (bit 7) is set to 1, and the effective CRU address is greater than >E00, an error interrupt occurs and the instruction is not executed.
4. This instruction is not implemented in the TMS9900.
5. Status bits 0, 1, 2, and 4 are set before the operation is performed.

3.6 INTERRUPT TRAPS AND ERROR INTERRUPT STATUS BITS

Figure 3-2 shows the interrupt traps. There are 16 interrupt levels available, several in use by the hardware as noted. Other interrupt levels are used for device interrupts according to the system hardware configuration.

Interrupt level 2 is defined as the system error interrupt. Eleven conditions can cause a system error interrupt. These are mapped by DNOS into crash codes in the range of >60 through >6F. These conditions are listed in Figure 3-3. For further details about the error conditions, refer to the *DNOS Messages and Codes Reference Manual*.

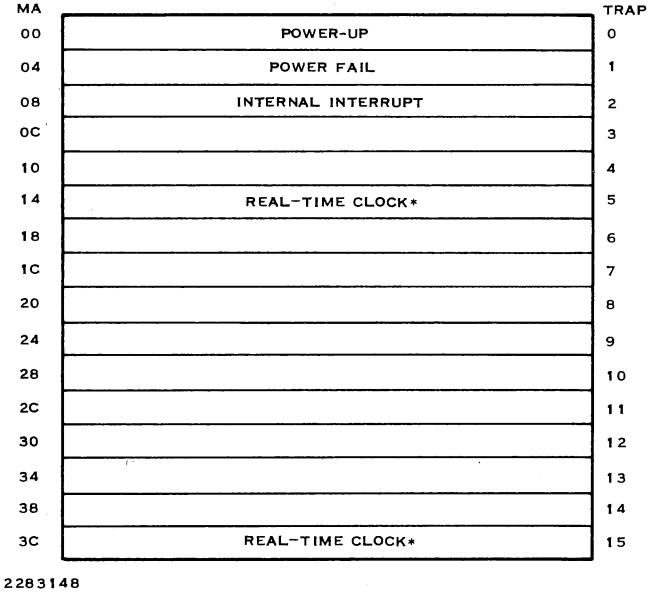


Figure 3-2. Interrupt Traps

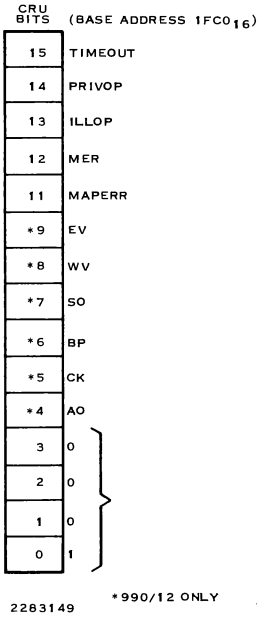
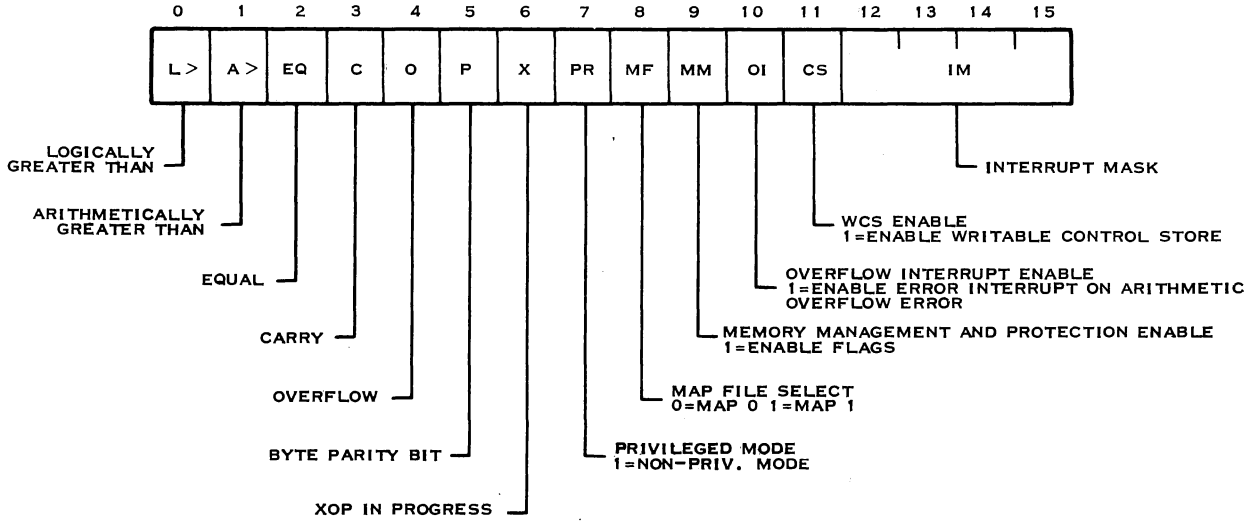


Figure 3-3. Error Interrupt Status Bits

3.7 COMPUTER STATUS REGISTER

Figure 3-4 shows the status register of the Model 990 and Business System computers. Table 3-4 shows which of the bits in the status register are used by which systems.



2282795

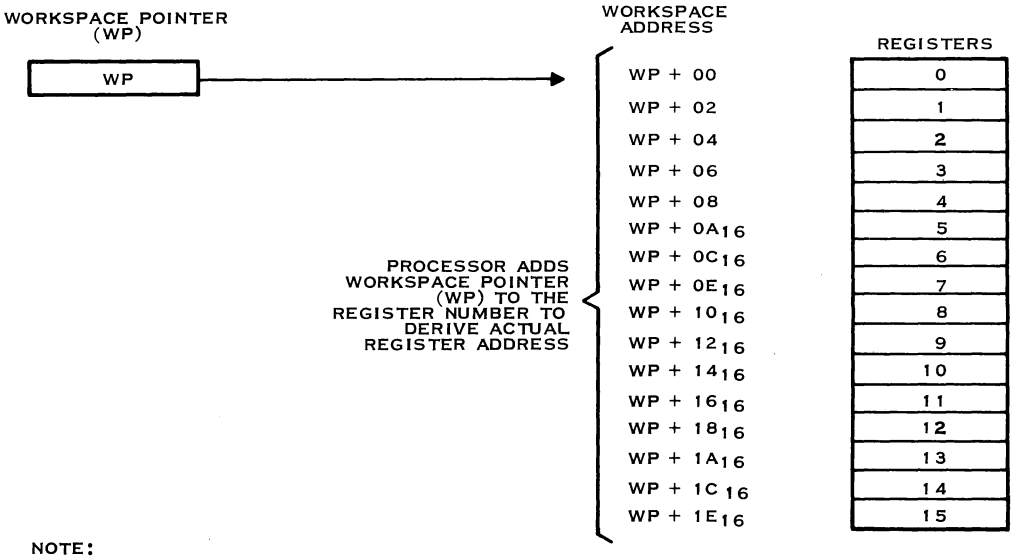
Figure 3-4. Computer Status Register

Table 3-4. Status Register Bit Usage by System

Bit	System
0–7	990/10, 990/10A, 990/12, Business Systems 300, 600, 800
8	990/10A, 990/12, Business Systems 300, 600, 800
9–11	990/12, Business System 800
12–15	990/10, 990/10A, 990/12, Business Systems 300, 600, 800

3.8 WORKSPACE POINTER AND REGISTERS

The workspace pointer (WP) register contains the address of the first byte in the currently active set of workspace registers. A workspace register file occupies 32 contiguous memory bytes in the program memory area. The address of the first byte must be even. Figure 3-5 shows the workspace pointer and registers. Table 3-5 shows the dedicated workspace registers.



NOTE:

ALL MEMORY WORD ADDRESSES ARE EVEN.

2282900

Figure 3-5. Workspace Pointer and Registers

Table 3-5. Dedicated Workspace Registers

Register	Contents	Used During
0 (bits 12-15)	Shift count (optional)	Shift instructions (SLA, SRA, SRC, SRL, SRAM, SLAM)
0	Count field (optional)	Byte string instructions (990/12 and Business System 800 only—CRC, CS, MOVS, MVSK, MVSR, POPS, PSHS, SEQB, SNEB, TCMB, TMB, TS, TSMB)
0 (bits 4-7)	Count field (optional)	(990/12 and Business System 800 only—INSF, IOF, SLAM, SRAM, XF, XV)
0 (bits 12-15)	Count field (optional)	Multiple precision instructions (990/12 and Business System 800 only—AM, ANDM, BDC, CNTO, DBC, INSF, IOF, LTO, NRM, ORM, RTO, SLAM, SM, SRAM, SWPM, XF, XORM, XV)
0	Floating point accumulator	Conversion instructions (990/12 and Business System 800 only—CDI, CRI)

Table 3-5. Dedicated Workspace Registers (Continued)

Register	Contents	Used During
0-1	Floating point accumulator	Floating point instructions (990/12 and Business System 800 only—AR, CDE, CER, CIR, CRE, CRI, DR, LR, MR, NEGR, SR, STR)
0-1	Destination operand	Signed instructions (990/12 and Business System 800 only—DIVS, MPYS)
0-3	Floating point accumulator	Floating point instructions (990/12 and Business System 800 only—AD, CDE, CDI, CED, CID, DD, LD, MD, NEGD, SD, STD)
11	Return address	Branch and link (BL) instruction
11	Effective address	Software implemented XOP instruction
12 (bits 3-14)	CRU base address	CRU instructions (SBO, SBZ, TB, LDCR,STCR)

Table 3-5. Dedicated Workspace Registers (Continued)

Register	Contents	Used During
13	Saved WP register	Context switching (BLWP, RTWP, software XOP, recognized interrupt, LOAD, RSET)
14	Saved PC register	Context switching (BLWP, RTWP, software XOP, recognized interrupt, LOAD, RSET)
15	Saved ST register	Context switching (BLWP, RTWP, software XOP, recognized interrupt, LOAD, RSET)

3.9 MACRO LANGUAGE

The elements of macro language are labels, strings, constants, operators, variables, parameters, symbols, keywords, and verbs. A macro definition contains statements including macro language verbs and model statements. Table 3-6 shows the syntax of the statements containing the macro language verbs. Table 3-7 defines the macro variable components and the symbol components.

Table 3-6. Macro Language Table

Statement	Syntax
MACRO	<macro name>...$MACRO...[<parm>][,<parm>]...[<comment>]
VARIABLE	...$VAR...<var>[,<var>]...[<comment>]
ASSIGN	...$ASG...<expression>...TO...<var>...[<comment>] <string>
NAME	<label>...$NAME...[<comment>]
GO TO	...$GOTO...<label>...[<comment>]
EXIT	...$EXIT...[<comment>]
CALL	...$CALL...<macro name>...[<comment>]
IF	...$IF...<expression>...[<comment>]
ELSE	...$ELSE...[<comment>]
END IF	...$ENDIF...[<comment>]
END	[<label>]...$END...[<comment>]
SUBSTRING	...$SBSTG...<var>,<expr>,<expr>...TO...<var>...[<comment>]

Note:

Use the comment field on the $MACRO statement only if you specify at least one parameter.

Table 3-7. Macro Component Table

Qualifier	Meaning
Macro Variable Components:	
S	The string component of the variable
A	The attribute component of the variable
V	The value component of the variable
L	The length component of the variable
Symbol Components:	
SS	The string component of a linked symbol
SV	The value component of a linked symbol
SA	The attribute component of a linked symbol
SL	The length component of a linked symbol
SU	The user attribute component of a linked symbol
SG	The segment component of a linked symbol

3.10 OBJECT CODE FORMAT

Table 3-8 lists the tag characters and the field values associated with each tag. Unless otherwise noted in the table, the size of each field is either four characters (ASCII object format) or four binary digits (compressed object format). Variations in this size are noted by an integer value in parentheses following the field value description. The tags are listed in groups, where the group heading indicates the general function of the tags in that group.

Table 3-8. Object Record Format and Tags

Tag	Field 1	Field 2	Field 3
Module Definition Tags:			
0	PSEG length	Module name (8)	—
M	DSEG length	$DATA	0000
M	Blank common length	$BLANK	Common no.
M	CSEG length	Common name (6)	Common no.
M	CBSEG length	$CBSEG	CBSEG No.

Table 3-8. Object Record Format and Tags (Continued)

Tag	Field 1	Field 2	Field 3
Entry Point Definition Tags:			
1	Absolute address	—	—
2	P/R address	—	—
Load Address Tags:			
9	Absolute address	—	—
A	P/R address	—	—
S	D/R address	—	—
Data Tags:			
B	Absolute value	—	—
C	P/R address	—	—
T	D/R address	—	—
N	C/R address	Common or CBSEG no.	—

Table 3-8. Object Record Format and Tags (Continued)

Tag	Field 1	Field 2	Field 3
External Definition (DEF) Tags:			
6	Absolute value	Symbol (6)	—
5	P/R address	Symbol (6)	—
W	D/R or C/R address	Symbol (6)	Common no.
External Reference (REF) Tags:			
3	P/R chain address	Symbol (6)	—
4	Absolute chain address	Symbol (6)	—
X	D/R or C/R chain address	Symbol (6)	Common no.
E	Symbol index number	Absolute offset	—
Secondary External Reference Tags:			
V	P/R chain address	Symbol (6)	—
Y	Absolute chain address	Symbol (6)	—
Z	D/R or C/R chain address	Symbol (6)	Common no.
U	0000	Symbol (6)	—

Table 3-8. Object Record Format and Tags (Continued)

Tag	Field 1	Field 2	Field 3
Symbol Definition Tags:			
G	P/R address	Symbol (6)	—
H	Absolute value	Symbol (6)	—
J	D/R or C/R address	Symbol (6)	Common no.
Checksum and End-of-Record Tags:			
7	Value	—	—
8	Any value	—	—
F	—	—	—
Load Bias Tag:			
D	Absolute address	—	—
Repeat Count Tag (FORTRAN Only):			
R	Value	Repeat count	—

Table 3-8. Object Record Format and Tags (Continued)

Tag	Field 1	Field 2	Field 3
Program ID (IDT) Tag (Link Editor Only):			
I	P/R address	Module name (8)	—
CBSEG Reference Tag (COBOL Only):			
Q	Record offset	CBSEG no.	—

Notes:

P/R indicates program relocatable (within a PSEG).

D/R indicates data relocatable (within a DSEG).

C/R indicates common relocatable (within a CSEG).

CBSEG indicates COBOL segment.

For further information, refer to the *Assembly Language Reference Manual*.

4

Program Development

4.1 INTRODUCTION

This section outlines steps necessary to develop and execute a program. It also provides information about development tools such as the link editor and debugger.

4.2 PROGRAM DEVELOPMENT STEPS

Use the following steps to create, link, install, and execute a program. Commands relating to each step appear in parentheses at the end of the step.

1. Create your source file using the text editor.

   XE

2. Compile or assemble your source file using the compiler or the assembler. The compiler executes in either background or foreground.

3. Link your program if necessary.

   XLE

   a. Check errors and investigate all unresolved references.

   b. Relink if necessary.

      XLE

4. During linking, if you do not specify the FORMAT IMAGE option, you must install your overlay, procedure, segment, or task.

   IO, IP, IPS, or IT

5. Establish the appropriate environment of synonyms, logical names, and LUNOs by setting up command procedures that:

   a. Bid your task in either foreground or background mode.

   b. Restore the environment as it was before your program executed.

6. Execute your task in either background or foreground mode.

4.3 LINK EDITOR COMMANDS

Table 4-1 lists the link editor commands and their syntax statements alphabetically. A brief description of the syntax rules follows the command list.

Table 4-1. Link Editor Commands

Command List	Notes
ABSOLUTE	
ADJUST [*n*]	
ALLGLOBAL	
ALLOCATE	
AUTO	
COMMON *base,name*[,*name*...,*name*]	
DATA *base*	
DUMMY	
END	
ERROR	
FIND [*name*...,*name*]	
FORMAT {ASCII/COMPRESSED/IMAGE[,REPLACE][,*priority*]}	
GLOBAL [*symbol*...,*symbol*]	
INCLUDE [*name*...,*name*]	See Note 1.
LIBRARY *name*[,*name*...,*name*]	
LOAD	
MAP {REFS/NO$'*string*'[,NO$'*string*'...,NO$'*string*']}	
NOAUTO	

Table 4-1. Link Editor Commands (Continued)

Command List	Notes
NOERROR	
NOLOAD	
NOMAP	
NOPAGE	
NOSYMT	
NOTGLOBAL [*symbol...,symbol*]	
PAGE	
PARTIAL	
PHASE *level,name*[,PROGRAM *base*][,ID *n*]	See Note 2.
PROCEDURE *name*	
PROGRAM *base*	
SEARCH [*name...,name*]	
SEGMENT *map,name*[,PROGRAM *base*][,ID *n*]	
SHARE *name,name*[,*name...,name*]	
SYMT	
TASK [*name*][,PROGRAM *base*]	

Table 4-1. Link Editor Commands (Continued)

Notes:

1. If you do not specify a *name* operand, the object module(s) to be included must immediately follow the INCLUDE command in the control stream.
2. If you use the PHASE command with the ABSOLUTE command, the PROGRAM *base* operand is required and can be followed by an optional DATA *base* operand.

The following syntax rules apply to the link editor commands.

1. Enter commands by typing in the command name, followed by at least one blank and then any operands required.
2. Enter either the full command name or only the first four characters of the name. This also applies to the PROGRAM operand in the PHASE, SEGMENT, and TASK commands.
3. Separate multiple operands by commas.

4. Place each command on a separate line (record).

5. Precede comments by a semicolon (;). Place comments on a separate line or after a command and its operand(s).

6. You can use synonyms and/or logical names to specify pathnames.

7. The notations used in the syntax definitions are as follows:

   a. Enter items in uppercase exactly as shown except for the command names and PROGRAM operand, which you can enter in the four-character abbreviated form.

   b. Replace items in lowercase italics with a specific operand of the appropriate type indicated.

   c. You may enter items in square brackets ([ ]); you must enter items not enclosed in square brackets.

d. You can enter one of several operands in braces ({ }). The choices are separated by slashes (/).

e. You can repeat an operand preceding an ellipsis (...) as many times as necessary. You must separate the operands with commas.

For further information, refer to the *DNOS Link Editor Manual*.

4.4 DEBUGGER COMMANDS

Table 4-2 lists the SCI commands that are most frequently used in debugging. Table 4-3 lists the debugger commands that allow control and trace execution of instructions in a task.

Table 4-2. General Debugger Commands

SCI Command	Command Name
Data Display Commands:	
LB	List Breakpoints
LM	List Memory
LSM	List System Memory
SIR	Show Internal Registers
SP	Show Panel
SV	Show Value
SWR	Show Workspace Registers
Data Modification Commands:	
MIR	Modify Internal Registers
MM	Modify Memory
MSM	Modify System Memory
MWR	Modify Workspace Registers

Table 4-2. General Debugger Commands (Continued)

SCI Command	Command Name
Breakpoint Commands:	
AB	Assign Breakpoint(s)
DB	Delete Breakpoint(s)
DPB	Delete/Proceed from Breakpoint(s)
PB	Proceed from Breakpoint
Task Control Commands:	
AT	Activate Task
HT	Halt Task
QD	Quit Debug Mode
RT	Resume Task
XD	Execute in Debug Mode
XHT	Execute and Halt Task

Table 4-2. General Debugger Commands (Continued)

SCI Command		Command Name
Search Commands:		
	FB	Find Byte
	FW	Find Word
Pascal Commands:		
	ABP	Assign Breakpoint — Pascal
	DBP	Delete Breakpoint — Pascal
	DPBP	Delete and Proceed from Breakpoint — Pascal
	LBP	List Breakpoints — Pascal
	LPS	List Pascal Stack
	PBP	Proceed from Breakpoint — Pascal
	SPS	Show Pascal Stack

Table 4-3. Debugger Commands for Simulated Mode

SCI Command	Command Name
ASB	Assign Simulated Breakpoint
DSB	Delete Simulated Breakpoint(s)
LSB	List Simulated Breakpoints
RST	Resume Simulated Task
ST	Simulate Task

For further information, refer to the *DNOS Assembly Language Programmer's Guide*.

5

SVC Blocks

5.1 INTRODUCTION

This section lists supervisor call (SVC) blocks by hexadecimal opcode and provides a line drawing of each call block. These line drawings show the type of information necessary for each call block. Applicable system flags, user flags, and operations for each sub-opcode are summarized in various tables. Table 5-1 lists the SVCs by hexadecimal opcode.

Table 5-1. SVC Opcodes

SVC #	Name
00	I/O Operations Resource-Independent Sub-opcodes: 00 — Open 01 — Close 02 — Close, Write EOF 03 — Open and Rewind 04 — Close and Unload 05 — Read Device Status 06 — Forward Space 07 — Backward Space 09 — Read ASCII 0B — Write ASCII 0D — Write EOF 0E — Rewind 0F — Unload Resource-Specific Sub-opcodes: See Table 5-4.

Table 5-1. SVC Opcodes (Continued)

SVC #	Name
01	Wait for I/O
02	Time Delay
03	Get Date and Time
04	End of Task
06	Suspend Task
07	Activate Suspended Task
09	Extend Time Slice
0A	Convert Binary to Decimal
0B	Convert Decimal to Binary
0C	Convert Binary to Hexadecimal
0D	Convert Hexadecimal to Binary
0E	Activate Time-Delayed Task
0F	Abort I/O Request by LUNO
10	Get Common Data Address
11	Change Task Priority

Table 5-1. SVC Opcodes (Continued)

SVC #	Name
12	Get Memory
13	Release Memory
14	Load Overlay
17	Get Task Bid Parameters
1B	Return Common Data Address
1C	Put Data
1D	Get Data
1F	Scheduled Bid Task
20	Install Disk Volume
21	System Log Queue Request
22	Disk Management
24	Suspend for Queue Input
25	Install Task Segment
26	Install Procedure/Program Segment
27	Install Overlay
28	Delete Task
29	Delete Procedure/Program Segment
2A	Delete Overlay

Table 5-1. SVC Opcodes (Continued)

SVC #	Name
2B	Execute Task
2C	Read/Write TSB
2D	Read/Write Task
2E	Self-Identification
2F	Get End Action Status
31	Map Program Name to ID
33	Kill Task
34	Unload Disk Volume
35	Poll Status of Task
36	Wait for Any I/O
37	Assign Program File Space
38	Initialize New Disk Volume
3B	Set Date and Time
3D	Semaphore Operations
	Sub-opcodes: 00 — Signal 01 — Wait

Table 5-1. SVC Opcodes (Continued)

SVC #	Name
	02 — Test 03 — Initialize 04 — Modify
3E	Reset End Action Status
3F	Retrieve System Data
40	Segment Management
	Sub-opcodes: 00 — Change Segment 01 — Create Segment 02 — Reserve Segment 03 — Release Segment 04 — Check Segment Status 05 — Force Write Segment 06 — Reserved 07 — Set/Reset Not Modified and Releasable

Table 5-1. SVC Opcodes (Continued)

SVC #	Name
	09 — Load Segment 0A — Unload Segment 0B — Set Exclusive Use of a Segment 0C — Reset Exclusive Use of a Segment
41	Initiate Event
42	Wait for Event
43	Name Management Sub-opcodes: 00 — Determine Name's Value 02 — Set Name's Value 04 — Delete Name 0F — Restore Name Segment
45	Get Encrypted Value

Table 5-1. SVC Opcodes (Continued)

SVC #	Name
46	Get Decrypted Value
47	Log Accounting Entry
48	Job Management
	Sub-opcodes: 01 — Create Job 02 — Halt Job 03 — Resume Halted Job 04 — Change Job Priority 05 — Map Job Name to Job ID 06 — Kill Executing Job 07 — Delete Job 09 — Get Job Information
49	Get Accounting Information
4C	Return Code Processor
4F	Post Event

5.2 SVC CALL BLOCKS

The line drawings of the SVC blocks are arranged by hexadecimal opcode and use the following notations:

Notation	Meaning
Greater than sign (>)	Identifies hexadecimal values.
Angle brackets (< >)	Enclose items returned to the supervisor call block.
Reserved	Designates a call block field or flag that must be set to 0.
[Reserved]	Designates a call block field that is reserved but may contain any value.

These notations apply to the supervisor call block diagrams used in this manual. The opcode and name of the SVC are printed on the top left of each SVC explanation. To the right and on the same line, the requirements and attributes of the SVC are printed. Additional lines are provided when several requirements and attributes apply. The requirements and attributes are:

- Align on word boundary
- Privileged tasks only
- System tasks only
- Can be initiated as an event

The I/O Operations SVC, opcode >00, is common to all I/O operations. SVC >00 contains sub-opcodes that are defined for each I/O operation. Some of the I/O operations require extensions to the basic SVC >00 block. These sub-opcodes and extensions are discussed in subsequent paragraphs.

The following line drawings represent the basic supervisor call blocks. For further information, refer to the *DNOS SVC Reference Manual.*

SVC >00 -- I/O OPERATIONS

ALIGN ON WORD BOUNDARY
CAN BE INITIATED AS AN EVENT

DEC	HEX		
0	0	>00	<RETURN CODE>
2	2	SUB-OPCODE	LUNO
4	4	<SYSTEM FLAGS>	USER FLAGS
6	6	DATA BUFFER ADDRESS	
8	8	READ CHARACTER COUNT	
10	A	WRITE CHARACTER COUNT/<ACTUAL READ COUNT>	

2279470

For more information on I/O operations, refer to I/O Operations SVC >00 Details in this section.

SVC > 01 -- WAIT FOR I/O

ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	> 01	<RETURN CODE>
2	2	ADDRESS OF BYTE 2 OF I/O CALL BLOCK	

2279471

SVC > 02 -- TIME DELAY

ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	>02	<RETURN CODE>
2	2	TIME DELAY COUNT	

2279455

SVC > 03 -- GET DATE AND TIME

ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	>03	<RETURN CODE>
2	2	BUFFER ADDRESS	

2279738

SVC > 04 -- END OF TASK

DEC	HEX	
0	0	>04

2279462

SVC > 06 -- Unconditional Suspend

Dec	Hex	
0	0	>06

2279458

SVC > 07 -- Activate Suspended Task

Dec	Hex		
0	0	>07	<Task State Code>
2	2	Run-Time ID	

2279459

SVC >09 -- Extend Time Slice

Dec	Hex		
0	0	>09	Time Unit Count

2279460

SVC >0A -- Convert Binary to Decimal ASCII

Dec	Hex		
0	0	>0A	<Return Code>
2	2	<ASCII Minus/Blank>	<ASCII Digit>
4	4	<ASCII Digit>	<ASCII Digit>
6	6	<ASCII Digit>	<ASCII Digit>

2279716

Register 0 contains the value to be converted.

SVC > 0B -- CONVERT DECIMAL ASCII TO BINARY

DEC	HEX		
0	0	>0B	<RETURN CODE>
2	2	ASCII SIGN	ASCII DIGIT
4	4	ASCII DIGIT	ASCII DIGIT
6	6	ASCII DIGIT	ASCII DIGIT

2279717

Register 0 receives the converted value.

SVC >0C -- Convert Binary to Hexadecimal ASCII

Dec	Hex		
0	0	>0C	<Return Code>
2	2	<ASCII Digit>	<ASCII Digit>
4	4	<ASCII Digit>	<ASCII Digit>

2279718

Register 0 contains the value to be converted.

SVC >0D -- Convert Hexadecimal to Binary

Dec	Hex		
0	0	>0D	<Return Code>
2	2	ASCII Digit	ASCII Digit
4	4	ASCII Digit	ASCII Digit

2279719

Register 0 receives the converted value.

SVC >0E -- Activate Time Delay Task

Dec	Hex		
0	0	>0E	<Task State Code>
2	2	Run-Time ID	

2279456

SVC >0F -- ABORT I/O PRIVILEGED TASK*

DEC	HEX		
0	0	>0F	<RETURN CODE>
2	2	FLAGS	LUNO
4	4	ZERO, OR ADDRESS OF TSB	
6	6	ZERO, OR ADDRESS OF JSB	

2286051

Byte 2 Flags

0 Do not close

1 — Do not close files and devices
0 — Close open files and devices

1-7 Reserved

*Task must be software privileged if aborting I/O in another job.

SVC >10 -- GET COMMON DATA ADDRESS

DEC	HEX		
0	0	>10	<RETURN CODE>

2279755

The address is returned to register 9.

The size of the area is returned to register 8.

SVC >11 -- CHANGE TASK PRIORITY

DEC	HEX		
0	0	>11	NEW PRIORITY LEVEL <OLD PRIORITY LEVEL>

2279457

SVC >12 -- GET MEMORY

ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	>12	<RETURN CODE>
2	2	NUMBER OF BEETS	

2279724

Register 9 receives the beginning address of the first beet.

SVC >13 -- RELEASE MEMORY ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	>13	<RETURN CODE>
2	2	[RESERVED]	

2279725

All memory beyond the address in register 9 is released.

SVC >14 -- LOAD OVERLAY

ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	>14	<RETURN CODE>
2	2	LOAD ADDRESS	
4	4	OVERLAY ID	
6	6	PROGRAM FILE LUNO	

2279447

Load Address:

If 0, use the natural load address. If the nonzero load address differs from the natural load address and the overlay is relocatable, then relocation is performed.

Program File LUNO:

If 0, use .S$SHARED proram file. If >FF, use the same program file as the requesting task. Otherwise, specifies LUNO previously assigned to correct program file.

SVC >17 -- GET TASK BID PARAMETERS

ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	>17	<RETURN CODE>
2	2	<PARAMETER 1>	
4	4	<PARAMETER 2>	

2279741

SVC >1B -- RETURN COMMON DATA ADDRESS

DEC	HEX	
0	0	>1B

2279756

SVC >1C -- PUT DATA

ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	>1C	<RETURN CODE>
2	2	RESERVED	MESSAGE ID
4	4	BUFFER ADDRESS	
6	6	[RESERVED]	
8	8	MESSAGE LENGTH	
10	A	RESERVED	

2279757

SVC >1D -- GET DATA — ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	>1D	<RETURN CODE>
2	2	FLAGS	MESSAGE ID
4	4	BUFFER ADDRESS	
6	6	BUFFER LENGTH	
8	8	ACTUAL MESSAGE LENGTH	
10	A	RESERVED	

2286060

SVC >1F -- SCHEDULED BID TASK — ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	>1F	<RETURN CODE>
2	2	INSTALLED ID	YEAR
4	4	DAY	
6	6	HOUR	MINUTE
8	8	SECOND	FLAGS
10	A	PARAMETER 1	
12	C	PARAMETER 2	
14	E	STATION ID	PROGRAM FILE LUNO
16	10	JOB ID	

2286048

Byte 2 Flags

0 Delete flag

1 — Delete all messages in the queue without return
0 — Return the oldest message and delete that message from the queue

1-7 Reserved

Byte 9 Flags

0 Station ID/LUNO

1 — Station ID and LUNO in bytes 14-15
0 — Station ID of calling task and system program file

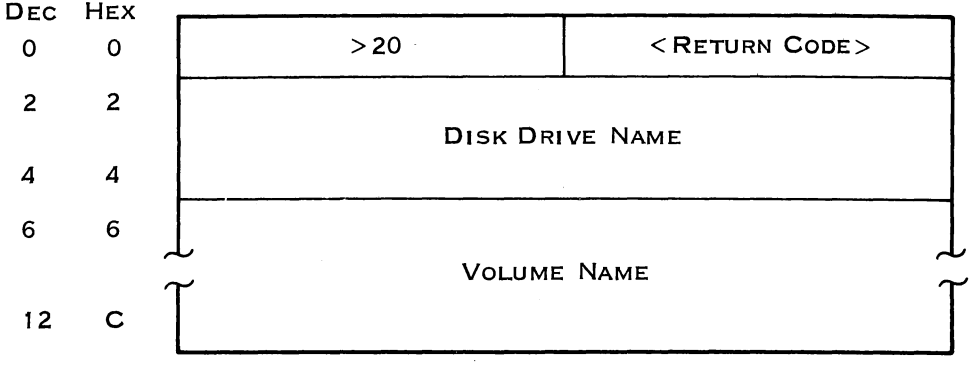
1 Job ID flag

1 — Job ID specified in bytes 16-17
0 — Same job as calling task

2-7 Reserved

SVC > 20 -- INSTALL DISK VOLUME

ALIGN ON WORD BOUNDARY
PRIVILEGED TASK ONLY



2279714

SVC >21 -- SYSTEM LOG

ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	>21	<RETURN CODE>
2	2	RESERVED	
4	4	BUFFER ADDRESS	
6	6	RESERVED	

2279742

SVC >22 -- Disk Management

Align on word boundary
Privileged task only

Dec	Hex		
0	0	>22	<Return Code>
2	2	Sub-Opcode	Reserved
4	4	Disk PDT Address	
6	6	Block Size (Bytes)	
8	8	Number of Blocks or Number of ADUs	
10	A		
12	C	ADU Number	

2279750

The return fields after a successful allocation are:

DEC	HEX		
8	8	ADU/BLOCK	BLOCKS/ADU
10	A	NUMBER OF ADUS ALLOCATED	
12	C	STARTING ADU NUMBER	

2279751

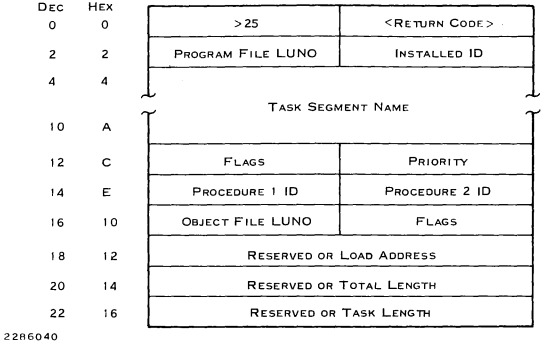
SVC > 24 -- SUSPEND FOR QUEUE INPUT SYSTEM TASK ONLY

DEC	HEX	
0	0	> 24

2279752

SVC >25 -- INSTALL TASK SEGMENT

ALIGN ON WORD BOUNDARY
PRIVILEGED TASKS ONLY



Program File LUNO:

If 0, use .S$SHARED program file. If FF, use same program file as bidding task.

Byte 12 Flags

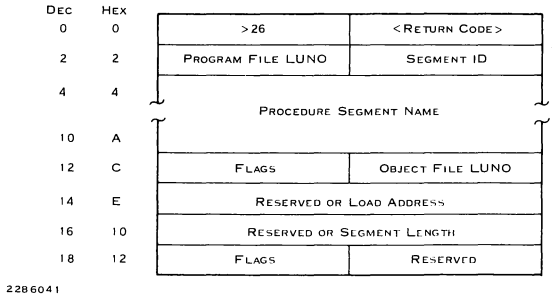
0 Privileged
1 System
2 Memory resident
3 Delete protected
4 Replicatable
5 Proc 1 on .S$SHARED
6 Proc 2 on .S$SHARED
7 Special install (used by system)

Byte 17 Flags

0 Overflow
1 WCS
2 Execute protect
3 Software privileged
4 Updatable
5 Reusable
6 Copyable
7 Reserved; set to zero

SVC >26 -- INSTALL PROCEDURE/PROGRAM SEGMENT

ALIGN ON WORD BOUNDARY
PRIVILEGED TASKS ONLY



Byte 12 Flags

0 Segment flag

  1 — Program seg
  0 — Procedure seg

1 For program seg only. May only be accessed by system task

2 Memory resident
3 Delete protected
4 Uses Writable Control Store (procedure seg); replicatable (program seg)
5 Execute protected (procedure seg); not sharable (program seg)
6 Write protected (procedure seg); reserved (program seg)
7 Special install; set to 0

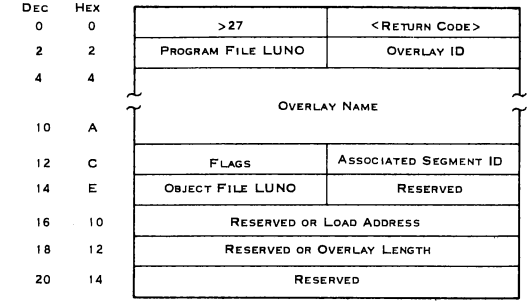
Byte 18 Flags

For procedure seg only.
For a program seg, byte 18 should be set to 0.

0 Reserved
1 Writable Control Store
2 Execute protect
3 Write protect
4 Updateable
5 Reusable
6 Copyable
7 Reserved; set to 0

SVC > 27 -- INSTALL OVERLAY

ALIGN ON WORD BOUNDARY
PRIVILEGED TASKS ONLY



2286042

Byte 12 Flags

0 Relocate when loaded
1-2 Reserved
3 Delete protect
4-5 Reserved
6 Task/Segment flag

1 — Associated seg ID is a program segment ID
0 — Associated seg ID is a task segment ID

7 Special install; set to 0

SVC >28 — DELETE TASK

ALIGN ON WORD BOUNDARY
PRIVILEGED TASK ONLY

DEC	HEX		
0	0	>28	<RETURN CODE>
2	2	PROGRAM FILE LUNO	INSTALLED ID
4	4	FLAGS	RESERVED

2286043

Byte 4 Flags

0-6 Reserved
7 Program file LUNO is open

SVC >29 -- DELETE PROCEDURE/PROGRAM SEGMENT

ALIGN ON WORD BOUNDARY
PRIVILEGED TASK ONLY

DEC	HEX		
0	0	>29	<RETURN CODE>
2	2	PROGRAM FILE LUNO	SEGMENT ID
4	4	FLAGS	RESERVED

2286044

Byte 4 Flags

0-6 Reserved
7 Program file LUNO is open

SVC >2A -- DELETE OVERLAY

ALIGN ON WORD BOUNDARY
PRIVILEGED TASK ONLY

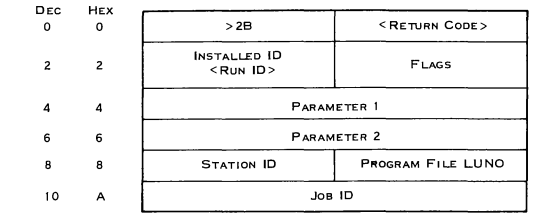
DEC	HEX		
0	0	>2A	<RETURN CODE>
2	2	PROGRAM FILE LUNO	OVERLAY ID
4	4	FLAGS	RESERVED

2286045

Byte 4 Flags

0-6 Reserved
7 Program file LUNO is open

SVC > 2B -- EXECUTE TASK ALIGN ON WORD BOUNDARY



Byte 3 Flags

0 Job ID flag

1 — Job ID in bytes 10-11
0 — Same job

1 Reserved
2 Reserved
3 Background
4 Terminate calling task
5 Reserved
6 Unconditionally suspend task
7 Suspend calling task until called task terminates

Station ID:

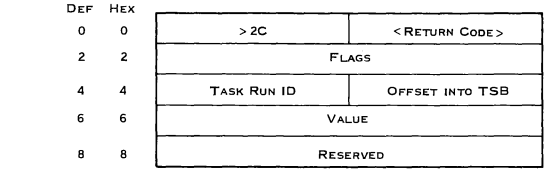
If 0, use the same ID as the calling task. If >FF, use no station ID.

Program File LUNO:

If 0, use the .S$SHARED program file. If >FF, use the same program file as the requesting task.

SVC > 2C -- READ/WRITE TSB

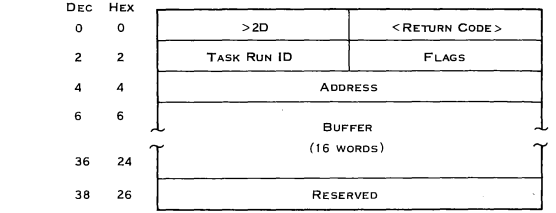
ALIGN ON WORD BOUNDARY
PRIVILEGED TASK ONLY



2286058

SVC > 2D -- READ/WRITE TASK

ALIGN ON WORD BOUNDARY
PRIVILEGED TASK ONLY



2286059

Byte 2-3 Flags

0-14 Reserved
15 Set to 1 for write;
Set to 0 for read.

Byte 3 Flags

0 Set to 1 for write;
Set to 0 for read.
1 ID flag

0 — Task run ID in byte 2
1 — System overlay ID in byte 2

2-3 Reserved
4-7 Number of words to be read or written;
specify 0 to read or write 16 words.

SVC > 2E -- Self-Identification

Dec	Hex		
0	0	>2E	<Run-Time ID>
2	2	<Installed ID>	<Station ID>
4	4	<Job ID>	

2279743

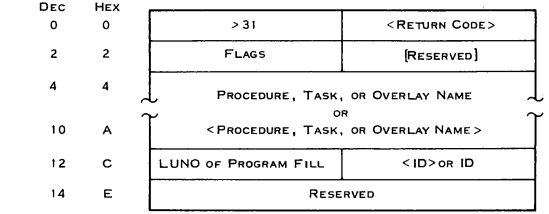
SVC >2F -- GET END ACTION STATUS

ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	>2F	<ERROR CODE>
2	2	<WORKSPACE POINTER>	
4	4	<PROGRAM COUNTER>	
6	6	<STATUS REGISTER>	
8	8	RESERVED	

2279744

SVC >31 -- MAP PROGRAM NAME TO ID ALIGN ON WORD BOUNDARY



Byte 2 User Flags

0-1 Type of segment

00 — Task
01 — Procedure
10 — Overlay

2 Defines operation

1 — Return name
0 — Return ID

3-6 Reserved

7 Current state of LUNO

1 — LUNO opened by this task
0 — LUNO not opened by this task (0 for LUNO >00 or >FF)

SVC >33 -- KILL TASK

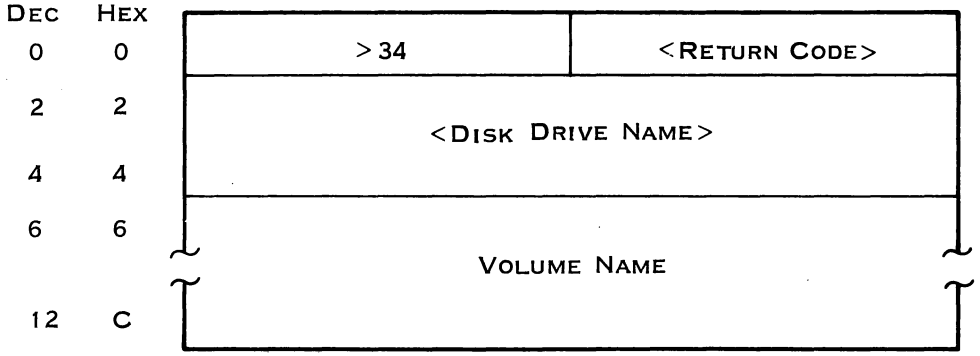
ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	>33	<RETURN CODE>
2	2	RUN-TIME ID	STATION ID
4	4	<TASK STATE>	RESERVED
6	6	RESERVED	

2279461

SVC >34 -- UNLOAD DISK VOLUME

ALIGN ON WORD BOUNDARY
PRIVILEGED TASK ONLY



2279715

SVC > 35 -- POLL STATUS OF TASK

DEC	HEX		
0	0	>35	<RETURN CODE>
2	2	FLAGS	<TASK STATE CODE>
4	4	STATION ID	RUN-TIME ID
6	6	RESERVED	

2286056

Byte 2 Flags

0 Set to 1 when byte 4 is station ID; set to 0 when station ID is station ID of calling task
1-7 Reserved

SVC >36 -- WAIT FOR ANY I/O

ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	> 36	<RETURN CODE>

2279472

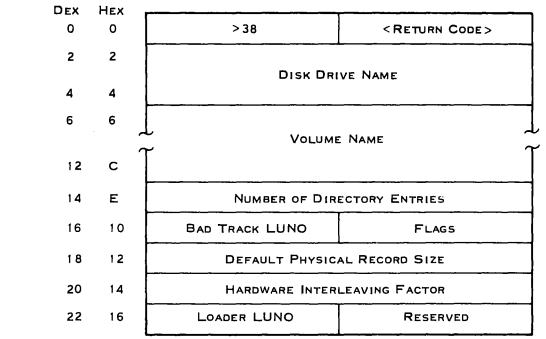
SVC >37 -- ASSIGN PROGRAM FILE SPACE

ALIGN ON WORD BOUNDARY
PRIVILEGED TASK ONLY

DEC	HEX		
0	0	>37	<RETURN CODE>
2	2	PROGRAM FILE LUNO	RESERVED
4	4	LENGTH	
6	6	<RECORD NUMBER>	
8	8	RESERVED	

2279451

SVC >38 -- INITIALIZE NEW DISK VOLUME ALIGN ON WORD BOUNDARY
PRIVILEGED TASK ONLY



2286053

Byte 17 Flags

0 Set to 1
1 1 — Write track one loader
  0 — Do not write track one loader
2 1 — Do not format disk
  0 — Format disk
3-7 Reserved

SVC >3B -- Set Date and Time

Align on word boundary
Privileged task only

Dec	Hex		
0	0	>3B	<Return Code>
2	2	Buffer Address	
4	4	Reserved	

2279739

SVC > 3D -- SEMAPHORE

ALIGN ON WORD BOUNDARY
CAN BE INITIATED AS AN EVENT

DEC	HEX		
0	0	>3D	<RETURN CODE>
2	2	SUB-OPCODE	SEMAPHORE NUMBER
4	4	SEMAPHORE VALUE	
6	6	RESERVED	

2286055

Byte 2 Sub-Opcodes

0 Signal; increment semaphore value
1 Wait; decrement semaphore and wait if negative
2 Test; returns value of semaphore
3 Initialize; initialize semaphore
4 Modify

SVC >3E -- RESET END ACTION STATUS

DEC	HEX		
0	0	>3E	<RETURN CODE>
2	2	RUN-TIME ID	RESERVED

2279745

SVC >3F -- RETRIEVE SYSTEM DATA ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	>3F	<RETURN CODE>
2	2	DATA STRUCTURE TYPE	FLAGS
4	4	INDEX	
6	6	OFFSET (BYTES)	
8	8	BUFFER LENGTH (BYTES)	
10	A	LENGTH RETURNED (BYTES)	
12	C	BUFFER ADDRESS	
14	E	RESERVED	

2286057

Byte 2 Data Structure Type

>81 PDT (Physical Device Table)
>82 TPCS (TILINE Peripheral Device Space)
>83 CSEG LGLCOM (Log Common Data)
>84 CSEG NFCLKD (Clock Data)
>85 CSEG NFDATA (Nucleus Data)
>86 CSEG NFPTR (System Printers)
>87 CSEG PMDATA (Program Management Data)
>88 CSEG NFJOBC (Job Management Data)

Byte 3 Flags

0 Block format

1 — Returns one word in bytes 10-11
0 — Returns data in specified buffer

1 Address

1 — Value in bytes 6-7 used as indirect address
0 — Value in bytes 6-7 used as offset (relative to first byte of structure)

2-7 Reserved

SVC > 40 -- SEGMENT MANAGEMENT ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	>40	<RETURN CODE>
2	2	SUB-OPCODE	SEGMENT GROUP LUNO
4	4	FLAGS	
6	6	SEGMENT ID ONE (INSTALLED OR RUN-TIME ID)	
8	8		
10	A	SEGMENT ID TWO (RUN- TIME ID)	
12	C	<SEGMENT ADDRESS IN ADDRESS SPACE>	
14	E	<SEGMENT LENGTH>	
16	10	ATTRIBUTES	
18	12	|RESERVED|	

2286054

Byte 2 Sub-Opcodes and Applicable Flags

00	Change segment	>FF83
01	Create segment	>F983
02	Reserve segment	>9803
03	Release segment	>9803
04	Check segment status	>9C03
05	Force write segment	>F803
07	Set/Reset not modified and releasable flags	>F803
09	Load segment	>9803
0A	Unload segment	>9803
0B	Set exclusive use	>9803
0C	Reset exclusive use	>9803

Byte 16 Attributes

0 1 — For task: privileged; otherwise, readable segment
1 1 — System segment
2 1 — Memory resident
3 Reserved
4 1 — Replicatable segment
5 1 — For task: Proc 1 on S$SHARED; otherwise, share protected
6 1 — For task: Proc 2 on S$SHARED; otherwise, reserved
7 Reserved

Byte 16 Attributes

8 1 — For task: overflow protection; otherwise, reserved
9 1 — Uses WCS
10 1 — Execute protected
11 1 — For task: software privileged; otherwise, write protected
12 1 — Updatable segment
13 1 — Reusable segment
14 1 — Copyable
15 Reserved

Byte 4-5 Flags

0 1 — Installed ID
0 — Run-time ID
1 1 — Not modified
2 1 — Releasable
3 1 — Memory based
4 1 — Old segment specified by run-time ID
0 — Old segment specified by position in bits 14-15
5 1 — Segment is a task segment
6 1 — Verify the load address
7 1 — Use bit 8
8 1 — Set exclusive use
0 — Reset exclusive use

9-13 Reserved
14-15 See bit 4

SVC >41 -- INITIATE EVENT

ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	>41	<RETURN CODE>
2	2	RESERVED	EVENT NUMBER
4	4	REQUEST BLOCK ADDRESS	
6	6	RESERVED	

2279735

SVC >42 -- WAIT FOR EVENT

ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	>42	<RETURN CODE>
2	2	MAXIMUM WAIT TIME	
4	4	<EVENT BITS>	
6	6		
8	8	EVENT MASK	
10	A		
12	C	RESERVED	

2279736

SVC >43 -- NAME MANAGEMENT ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	>43	<RETURN CODE>
2	2	SUB-OPCODE	FLAGS
4	4	ADDRESS OF NAME	
6	6	ADDRESS OF VALUE	
8	8	ADDRESS OF PARAMETER LIST	
10	A	SEGMENT ID/<PATHNAME FLAG>	
12	C	RESERVED	

2286049

Byte 2 Sub-Opcodes

00 Determine name's value
02 Set name's value
04 Delete name

Byte 3 Flags

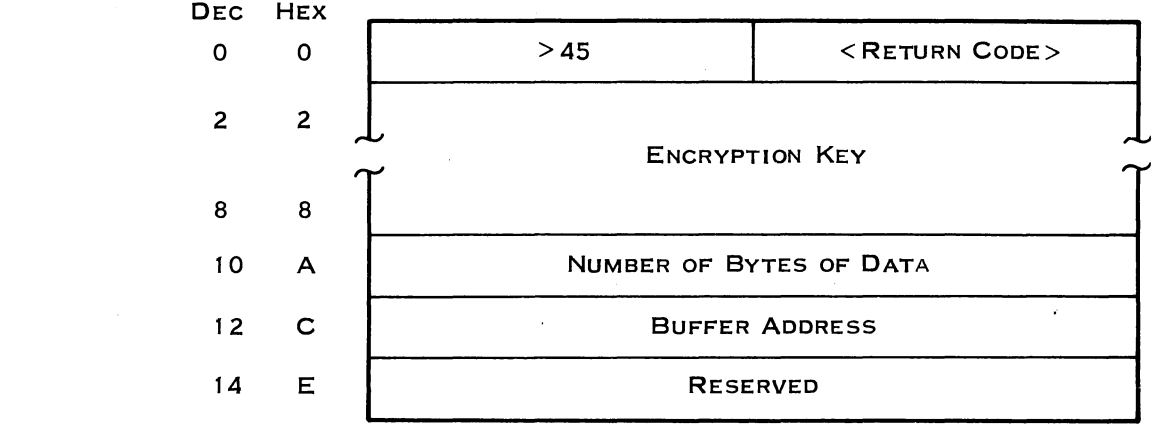
0 Name type

1 — Logical name
0 — Synonym

1-2 Reserved
3 Global name
4-7 Reserved

SVC > 45 -- GET ENCRYPTED VALUE

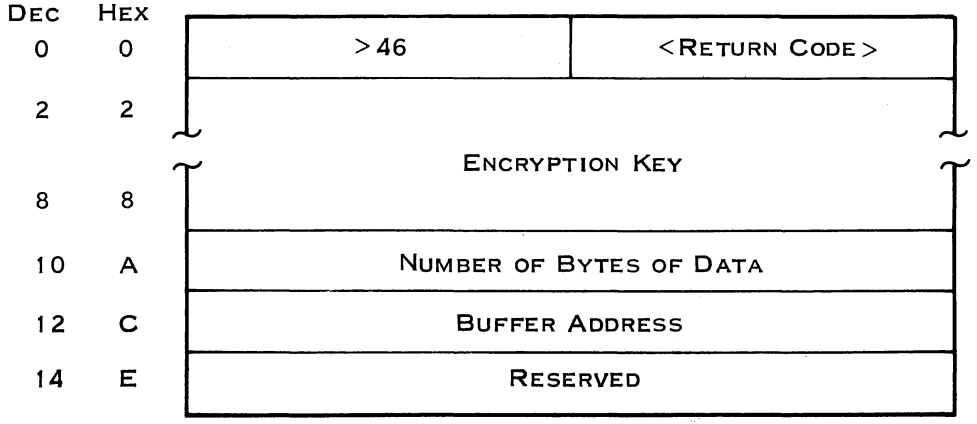
ALIGN ON WORD BOUNDARY



2279720

SVC > 46 -- GET DECRYPTED VALUE

ALIGN ON WORD BOUNDARY



2279721

SVC > 47 -- LOG ACCOUNTING ENTRY

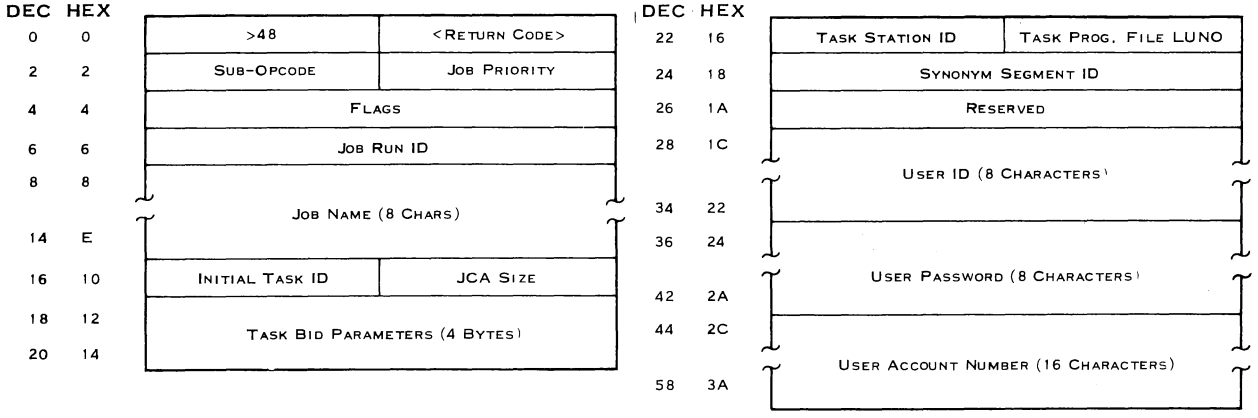
ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	>47	<RETURN CODE>
2	2	BUFFER ADDRESS	
4	4	RESERVED	

2279722

SVC>48 --JOB MANAGEMENT REQUEST

ALIGN ON WORD BOUNDARY



2286039

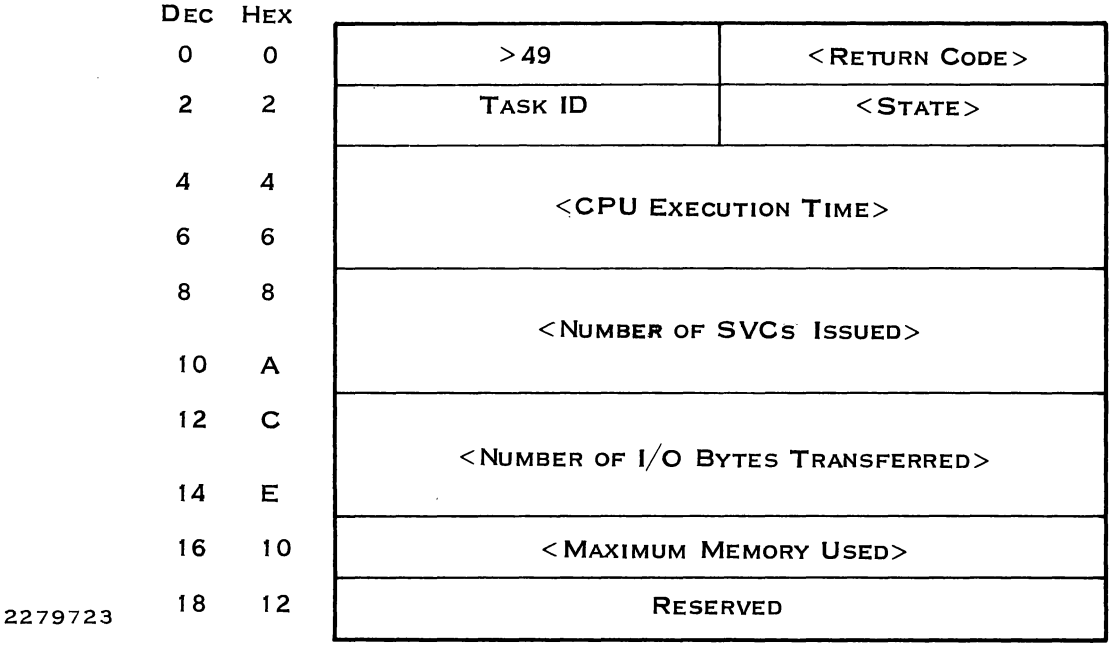
Byte 2 Sub-Opcodes

01 Create job
02 Halt job
03 Resume halted job
04 Change job priority
05 Map name to ID
06 Kill job
07 Delete job
09 Get job information

Byte 4-5 Flags

0 Use bytes 28-59 for new user ID, password, and account
1 Do not validate request
2 Batch job
3-15 Reserved

SVC > 49 -- GET ACCOUNTING INFORMATION ALIGN ON WORD BOUNDARY



SVC >4C -- RETURN CODE PROCESSING

ALIGN ON WORD BOUNDARY

DEC	HEX		
0	0	>4C	<RETURN CODE>
2	2	SVC BLOCK ADDRESS	
4	4	BUFFER ADDRESS	
6	6	<MESSAGE NUMBER>	
8	8	RESERVED	

2279747

SVC > 4F -- Post Event

Align on word boundary

Dec	Hex		
0	0	>4F	<Return Code>
2	2	Run ID	Event Number
4	4	Job ID	
6	6	Reserved	

2283150

5.3 I/O OPERATIONS SVC >00 DETAILS

SVC >00 contains sub-opcodes that are defined for each I/O operation.

5.3.1 Basic SVC >00 Call Block Variant

The basic call block variant for SVC >00 is a 12-byte call block that includes system flags and user flags. The following line drawing represents the basic supervisor call block. Table 5-2 lists system flag bits and their descriptions. Table 5-3 lists user flag bits and their descriptions.

SVC >00 -- I/O Operations

Align on word boundary
Can be initiated as an event

Dec	Hex		
0	0	>00	<Return Code>
2	2	Sub-Opcode	LUNO
4	4	<System Flags>	User Flags
6	6	Data Buffer Address	
8	8	Read Character Count	
10	A	Write Character Count/ < Actual Read Count >	

2279470

Table 5-2. Bit Description for System Flags

Bit	Meaning
0	Busy
1	Error
2	End-of-file
3	Event key
4–7	Reserved

Table 5-3. Bit Description for User Flags

Bit	Meaning
0	Initiate request
1	Output with reply Read with validation Key specified flag Buffer has write interleaved format
2	Security access rights requested
3	Offset enabled (Read by track)
3	Resolve logical names (Master read)
3-4	Access privileges (Open operations) 00 — Exclusive write 01 — Exclusive all 10 — Shared 11 — Read only

Table 5-3. Bit Description for User Flags (Continued)

Bit	Meaning
4	Offset forward (Read by track)
5	Lock (Read operations to files) Unlock (Write operations to files) Do not suspend (Master Read, Read Call Block) Transfer inhibit (Read by track) Immediate open (for a TPD) Do not replace (Open operations to files, Rename file operation)
6	Extended call block
7	Blank adjust Set event mode (Open operations) No retries (Read by track)

Table 5-4 lists applicable system flags, user flags, and operations for each sub-opcode. An asterisk notation in Table 5-4 in a column for a particular I/O resource indicates which operations are supported by that I/O resource. All system and user flags which may be applicable for a particular sub-opcode are shown in the table. The applicable system and user flags for a particular sub-opcode are a function of the I/O resource in use. In most cases, an understanding of the bit descriptions for the system and user flags (Tables 5-2 and 5-3) is sufficient to determine which flags apply for a specific I/O resource. Refer to the *DNOS Supervisor Call (SVC) Reference Manual* for further information.

Table 5-4. Applicable Operations for I/O Resources

SUB-OPCODE	DESCRIPTION	SYSTEM FLAGS				USER FLAGS								KIF	REL REC FILE	SEQ FILE	MASTER/SLAVE CHAN	SYMMETRIC CHAN	CARD READER	CASSETTE	DIRECT DISK	LINE PRINTER	MAG TAPE	TPD	733 ASR	911 VDT	940 OR 931 VDT
>00	OPEN	0	1			0			3	4	5		7	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
>01	CLOSE	0	1			0								Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
>02	CLOSE, WRITE EOF	0	1			0									Y	Y		Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
>03	OPEN AND REWIND	0	1			0			3	4	5		7	Y	Y	Y		Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
>04	CLOSE AND UNLOAD	0	1			0									Y	Y		Y	Y	Y		Y	Y	Y	Y	Y	Y
>05	READ DEVICE STATUS	0	1			0											Y	Y				Y	Y	Y	Y	Y	Y
>05	READ FILE CHARACTERISTICS	0	1			0		2						Y	Y	Y											
>05	READ FORMAT	0	1			0															Y						
>05	REMOTE GET EVENT CHARACTER	0	1		3	0	1					6												Y	Y	Y	Y
>06	FORWARD SPACE	0	1	2		0								Y	Y	Y				Y			Y				
>07	BACKWARD SPACE	0	1	2		0								Y	Y	Y				Y			Y				
>07	ISSUE RESTORE TO DISK	0	1			0															Y						
>08	WRITE FORMAT	0	1			0															Y						
>09	READ ASCII	0	1	2	3	0					5		7	Y	Y	Y			Y	Y			Y	Y	Y	Y	Y
>09	READ BY ADU	0	1			0															Y						
>09	SYMMETRIC READ	0	1	2		0												Y									
>0A	READ BY TRACK	0	1			0			3	4	5		7								Y						
>0A	READ DIRECT	0	1	2		0							7	Y	Y	Y		Y	Y	Y			Y	Y		Y	Y

2285057 (1/3)

Table 5-4. Applicable Operations for I/O Resources (Continued)

SUB-OPCODE	DESCRIPTION	SYSTEM FLAGS				USER FLAGS								KIF	REL REC FILE	SEQ FILE	MASTER/SLAVE CHAN	SYMMETRIC CHAN	CARD READER	CASSETTE	DIRECT DISK	LINE PRINTER	MAG TAPE	TPD	733 ASR	911 VDT	940 OR 931 VDT
>08	WRITE ASCII	0	1			0	1								Y	Y				Y		Y	Y	Y	Y	Y	Y
>08	WRITE BY ADU	0	1			0	1				5		7								Y						
>08	SYMMETRIC WRITE	0	1			0												Y									
>0C	WRITE BY TRACK	0	1			0															Y						
>0C	WRITE DIRECT	0	1			0							7		Y	Y		Y		Y		Y	Y	Y		Y	Y
>0D	WRITE EOF	0	1	2		0										Y		Y		Y		Y	Y	Y			Y
>0D	WRITE LOGICAL EOF	0	1	2		0									Y												
>0E	REWIND	0	1			0								Y	Y	Y				Y		Y	Y	Y		Y	Y
>0E	STORE REGISTERS	0	1			0															Y						
>0F	UNLOAD	0	1			0														Y			Y	Y			
>0F	READ FORMAT	0	1			0															Y						
>10	REWRITE	0	1			0					5		7		Y	Y											
>10	WRITE DELETED SECTOR	0	1			0															Y						
>11	MODIFY ACCESS PRIVILEGES	0	1			0			3	4				Y	Y	Y											
>11	READ DELETED SECTOR	0	1			0								Y							Y						
>12	OPEN EXTEND	0	1			0			3	4	5				Y	Y		Y									Y
>12	WRITE FORMAT WITH INTERLEAVING	0	1			0	1														Y						

2285057 (2/3)

Table 5-4. Applicable Operations for I/O Resources (Continued)

SUB OPCODE	DESCRIPTION	SYSTEM FLAGS				USER FLAGS							KIF	REL REC FILE	SEQ FILE	MASTER/SLAVE CHAN	SYMMETRIC CHAN	CARD READER	CASSETTE	DIRECT DISK	LINE PRINTER	MAG TAPE	TPD	733 ASR	911 VDT	940 OR 931 VDT
>15	DEVICE DEPENDENT COMMUNICATION CONTROL	0	1			0						7											Y			
>19	MASTER READ	0	1			0			3		5					Y										
>1A	READ CALL BLOCK	0	1			0					5					Y										
>1B	MASTER WRITE	0	1			0										Y										
>1C	REDIRECT ASSIGN LUNO	0	1			0										Y										
>40	OPEN RANDOM	0	1			0			3	4	5			Y												
>41	READ GREATER	0	1			0	1				5			Y												
>42	READ BY KEY	0	1			0	1				5			Y												
>42	READ CURRENT	0	1			0	1				5			Y												
>44	READ GREATER OR EQUAL	0	1			0	1				5			Y												
>45	READ NEXT	0	1			0					5			Y												
>46	INSERT	0	1			0								Y												
>47	REWRITE	0	1			0					5			Y												
>48	READ PREVIOUS	0	1			0					5			Y												
>49	DELETE BY KEY	0	1			0	1							Y												
>49	DELETE CURRENT	0	1			0	1							Y												

2285057 (3/3)

Table 5-4. Applicable Operations for I/O Resources (Continued)

SUB-OPCODE	DESCRIPTION	SYSTEM FLAGS				USER FLAGS								KIF	REL REC FILE	SEQ FILE	MASTER/SLAVE CHAN	SYMMETRIC CHAN	CARD READER	CASSETTE	DIRECT DISK	LINE PRINTER	MAG TAPE	TPD	733 ASR	911 VDT	940 OR 931 VDT
>4A	UNLOCK	0	1			0								Y	Y	Y											
>50	SET CURRENCY EQUAL	0	1			0								Y													
>51	SET CURRENCY GREATER OR EQUAL	0	1			0								Y													
>52	SET CURRENCY GREATER	0	1			0								Y													
>59	MULTIPLE RECORD READ	0	1	2		0					5				Y	Y											
>58	MULTIPLE RECORD WRITE	0	1			0					5				Y	Y											

2285270

Table 5-5. Utility Flags

Bit	Description
0	File created by assign
1, 2	File Usage Flags 00—No special usage 01—Directory file 10—Program file 11—Image file
3, 4	LUNO scope 00—Task local 01—Job local 10—Global 11—Shared
5	Autogenerate LUNO
6	Autocreate file

SVC >00 -- I/O OPERATIONS
(UTILITY SUB-OPCODES)

ALIGN ON WORD BOUNDARY
CAN BE INITIATED AS AN EVENT

DEC	HEX			DEC	HEX	
0	0	>00	<RETURN CODE>	18	12	DEFINED LOGICAL RECORD LENGTH
2	2	SUB-OPCODE	LUNO	20	14	DEFINED PHYSICAL RECORD LENGTH
4	4	<SYSTEM FLAGS>	USER FLAGS	22	16	PATHNAME ADDRESS
6	6	<RESOURCE TYPE>		24	18	RESERVED
8	8	RESERVED		26	1A	
10	A			28	1C	INITIAL FILE ALLOCATION
12	C	KEY DEF. BLOCK ADDR/DEF. PHYS. REC. SIZE		30	1E	
14	E	RESERVED		32	20	SECONDARY FILE ALLOCATION
16	10	UTILITY FLAGS		34	22	

2286050

2279469

Table 5-5 lists utility flag bits and their descriptions. Table 5-6 lists applicable utility flags and operations for each SVC >00 sub-opcode.

Table 5-5. Utility Flags

Bit	Description
0	File created by assign
1, 2	File Usage Flags 00—No special usage 01—Directory file 10—Program file 11—Image file
3, 4	LUNO scope 00—Task local 01—Job local 10—Global 11—Shared
5	Autogenerate LUNO
6	Autocreate file

SVC >00 -- I/O OPERATIONS
(UTILITY SUB-OPCODES)

ALIGN ON WORD BOUNDARY
CAN BE INITIATED AS AN EVENT

DEC	HEX		
0	0	>00	<RETURN CODE>
2	2	SUB-OPCODE	LUNO
4	4	<SYSTEM FLAGS>	USER FLAGS
6	6	<RESOURCE TYPE>	
8	8	RESERVED	
10	A		
12	C	KEY DEF. BLOCK ADDR/DEF. PHYS. REC. SIZE	
14	E	RESERVED	
16	10	UTILITY FLAGS	

DEC	HEX	
18	12	DEFINED LOGICAL RECORD LENGTH
20	14	DEFINED PHYSICAL RECORD LENGTH
22	16	PATHNAME ADDRESS
24	18	RESERVED
26	1A	
28	1C	INITIAL FILE ALLOCATION
30	1E	
32	20	SECONDARY FILE ALLOCATION
34	22	

2286050

2279469

Table 5-5 lists utility flag bits and their descriptions. Table 5-6 lists applicable utility flags and operations for each SVC >00 sub-opcode.

5.3.2 Extensions of the Basic SVC >00 Call Block

Some sub-opcodes of the I/O Operations SVC >00 require an extension to the basic supervisor call block. These extensions, represented by line drawings, are discussed in the following paragraphs.

5.3.2.1 I/O Utility SVC Block Extension. The sub-opcodes of the I/O Operations SVC >00 that perform I/O utility functions such as support device I/O, file I/O, and interprocess communication, require an extension to the basic SVC block. The following SVC block applies to all utility functions except creating and deleting IPC channels.

Table 5-5. Utility Flags (Continued)

Bit	Description
7	Parameters present
8	Valid logical record length
9	Temporary file
10	Immediate write-to-disk file
11, 12	Data format 00—Normal record image 01—Blank suppressed 10—Reserved 11—Reserved
13	Allocation may grow

Table 5-5. Utility Flags (Continued)

Bit	Description
14, 15	File type 00—Reserved 01—Sequential file 10—Relative record file 11—Key indexed file

Table 5-6. I/O Utility Functions

SUB-OPCODE	DESCRIPTION	UTILITY FLAGS																DEVICE	IPC CHAN	KIF	REL REC FILE	SEQ FILE
>90	CREATE FILE		1	2					7	8	9	10	11	12	13	14	15			Y	Y	Y
>91	ASSIGN LUNO	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Y	Y	Y	Y	Y
>92	DELETE FILE								7											Y	Y	Y
>93	RELEASE LUNO				3	4			7									Y	Y	Y	Y	Y
>95	RENAME FILE								7											Y	Y	Y
>96	REMOVE FILE OR CHANNEL PROTECTION								7										Y	Y	Y	Y
>97	WRITE PROTECT FILE OR CHANNEL								7										Y	Y	Y	Y
>98	DELETE PROTECT FILE OR CHANNEL								7										Y	Y	Y	Y
>99	VERIFY PATHNAME OR DEVICE NAME		1	2			5						11	12	13	14	15	Y	Y	Y	Y	Y
>9A	ADD ALIAS	NONE																		Y	Y	Y
>9B	DELETE ALIAS	NONE																		Y	Y	Y
>9C	DEFINE WRITE MODE											10								Y	Y	Y

NOTES:

IF A UTILITY FLAG DOES NOT APPLY, SET TO ZERO

BITS 8-15 APPLY ONLY IF BIT 6=1 AND IF THE FILE DOES NOT ALREADY EXIST. IF THE FILE ALREADY EXISTS, BITS 1 AND 2, AND 11 THROUGH 15 ARE SET BY THE SYSTEM.

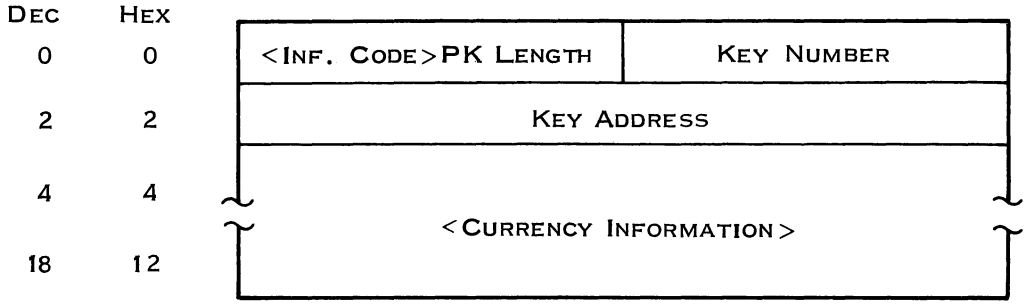
THESE BITS ARE SET BY THE SYSTEM.

2285058

5.3.2.2 Key Indexed File Currency SVC Block Extension. The key indexed file resource-specific operations use a single-word extension to the basic supervisor call block, as follows:

2279648

The currency block address is the word boundary address of a block containing a key that specifies the record on which the operation is to be performed. The currency block also contains the address of an area in which file management returns information, enabling access to the record by commands that do not supply a key. The structure of the currency block is:



2279649

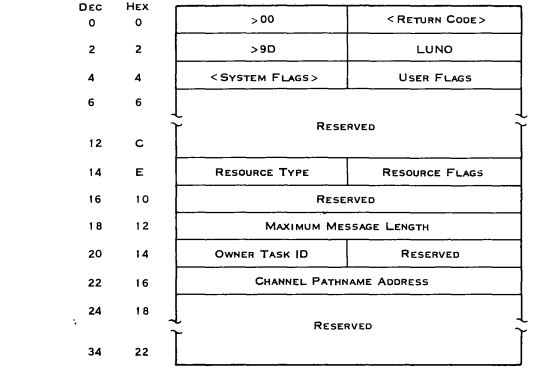
The key or partial key for the operation is in a block or buffer. The address of the block is placed in the currency block.

All resource-specific operations except Open Random and Unlock return an informative code in byte 0 of the currency block. The informative code is a code that gives the status of an operation.

5.3.2.3 IPC Channel Utility SVC Block Extensions. To create an IPC channel, a program executes an I/O Operations SVC with sub-opcode >9D. This is the extended supervisor call block for this operation:

SVC > 00 -- I/O OPERATIONS
(UTILITY SUB-OPCODE > 9D)

ALIGN ON WORD BOUNDARY
CAN BE INITIATED AS AN EVENT



2286052

Byte 5 User Flags

0-1 Channel scope flag

00 — Task-local scope
01 — Job-local scope
10 — Global scope

2 Shared channel

3 Channel type flag

1 — Symmetric
0 — Master/slave

4 Owner task processes Assign LUNO operations (Master/slave only)

5 Owner task processes Abort I/O operation (Master/slave only)

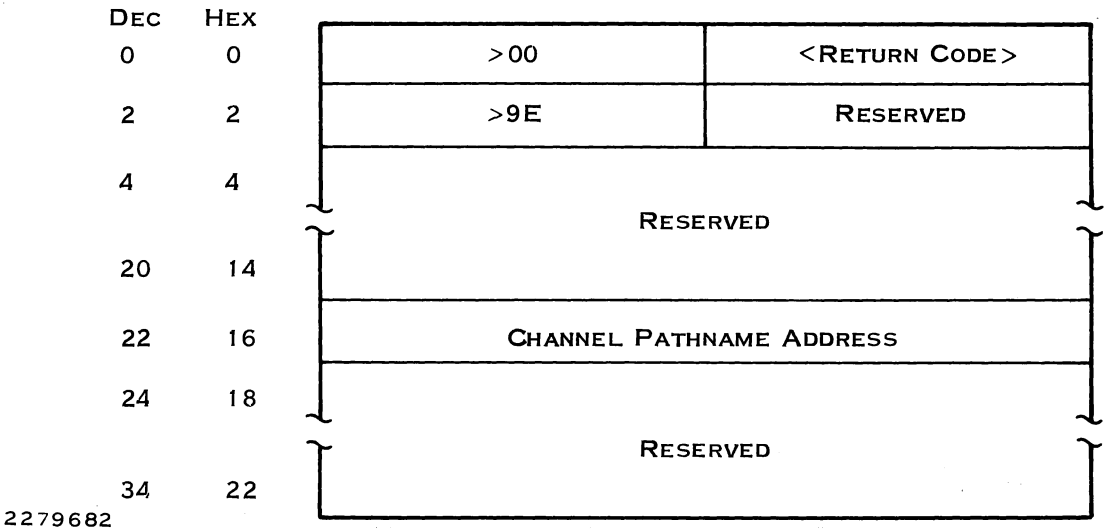
6 Owner task processes all I/O utility operations (Master/slave only)

7 Reserved

To delete an IPC channel, a program executes an I/O Operations SVC with sub-opcode >9E. This is the extended supervisor call block for this operaton:

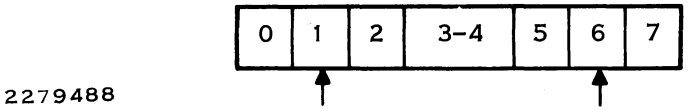
SVC >00 -- I/O OPERATION
(UTILITY SUB-OPCODE>9E)

ALIGN ON WORD BOUNDARY
CAN BE INITIATED AS AN EVENT



5.3.2.4 Terminal Devices SVC Block Extensions. Most of the TPD, 733 ASR, 911, 931, and 940 VDT resource-specific I/O operations use an extended SVC block. The sub-opcodes for the resource-independent operations apply, but the operations are modified by the states of flags in the extended user flags field.

The extended call flag in the user flag field (byte 5) of the SVC block must be set to one for resource-specific I/O operations. Otherwise, the system does not use the extensions to the SVC block. These flags in the user flag field apply to resource-specific I/O operations:



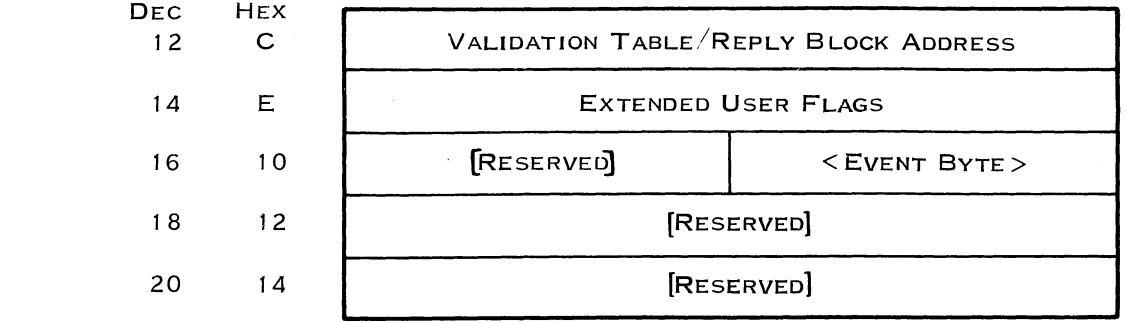
1 Reply flag. Set as follows:

1 — Write with Reply, Remote Get Event Character, or Read with Validation
0 — All other operations

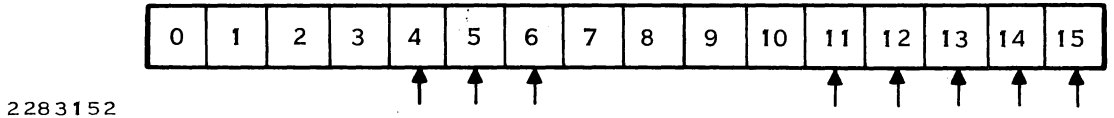
6 Extended call flag. Set as follows:

1 — Extended call block (required for resource-specific I/O)
0 — Basic supervisor call block (used for resource-independent)

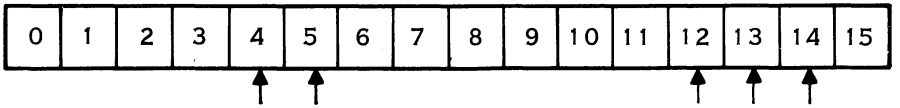
This is the extension to the basic SVC block for TPD and 733 ASR operations:



These are the extended user flags for TPD operations:



These are the extended user flags for 733 ASR operations:



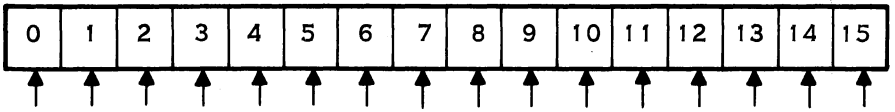
2279511

The following is the extension to the basic SVC block for 911, 931, and 940 VDT operations:

DEC	HEX		
12	C	VALIDATION TABLE/REPLY BLOCK ADDRESS	
14	E	EXTENDED USER FLAGS	
16	10	FILL CHARACTER	<EVENT BYTE>
18	12	CURSOR POSITION ROW	COLUMN
20	14	FIELD BEGINNING ROW	COLUMN

2283153

These are the extended user flags for 911, 931, and 940 VDT operations:



2279490

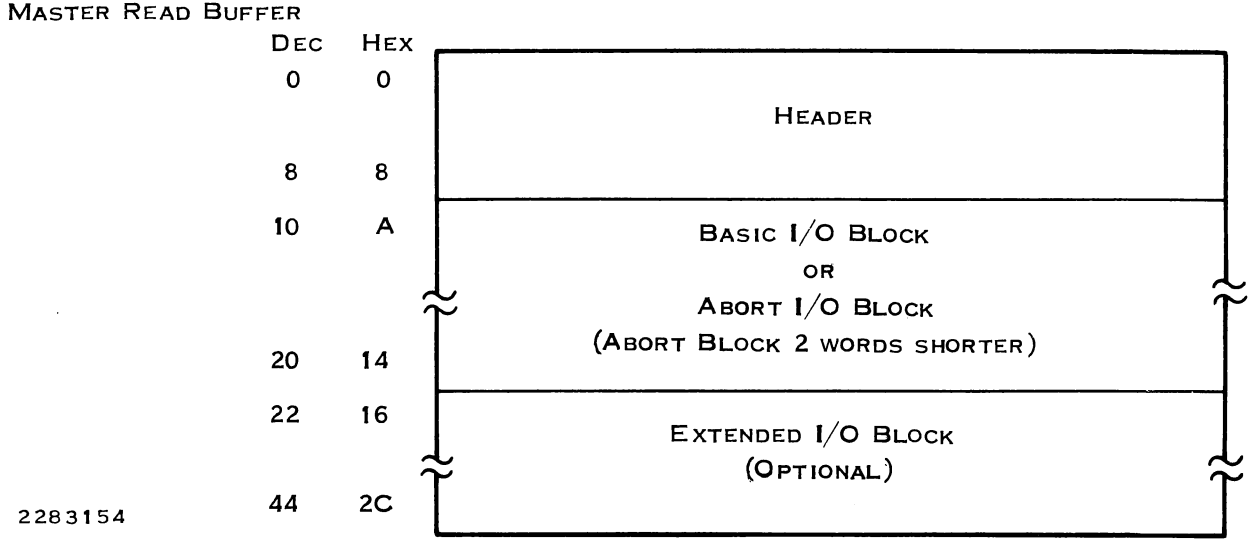
The following lists the flags and the I/O operations in which the user-extended flags are effective:

Bit	Definition	Used in Operations
0	Field start position	All
1	Intensity	All
2	Blink cursor	Read ASCII
3	Graphics	Read/Write
4	Eight-bit ASCII	Read/Write Direct
5	Task edit	Read
6	Beep	Read/Write

Bit	Definition	Used in Operations
7	Right boundary	Read
8	Cursor position within a read field	Read
9	Fill character	Read
10	Do not initialize field	Read
11	Return on termination character	Read
12	No echo	Read
13	Character validation	Read
14	Validation error mode	Read
15	Warning beep	Read

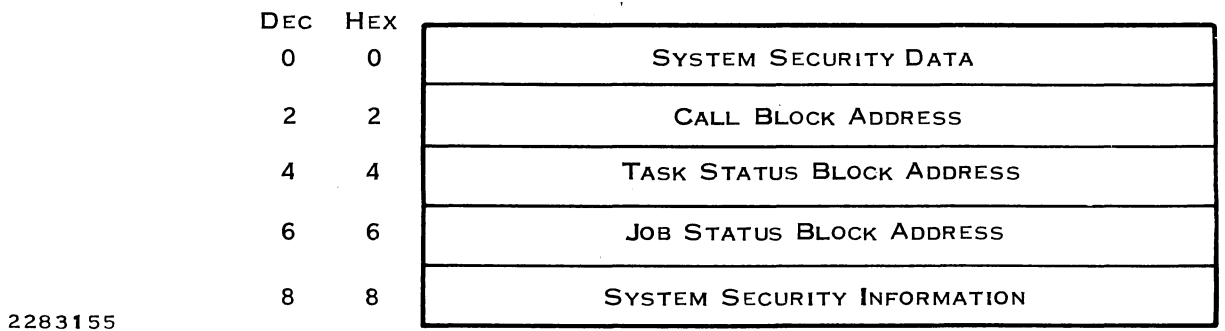
5.3.2.5 Master Read SVC Block Extension. The Master Read operation returns information from the supervisor call block, preceded by a header that contains information about the calling task. The master task supplies data in a buffer when processing a read operation requested by a slave task, except when zero characters are to be returned to the slave task. The data structure consisting of the header, the call block contents, and any required buffer is called the master read buffer (MRB).

Because the SVC block within the MRB varies according to the requested operation and the default type, the MRB has several variations. The following block shows the contents of the MRB:



The following line drawings represent the contents of each component of the MRB. A description of the content of specific bytes is included.

CALLING TASK HEADER



The calling task header of the MRB contains this information:

Byte	Contents
0-1	System security information.
2-3	Call block address. Address of the supervisor call block in the slave task.
4-5	Task status block address. Address of the task status block of the slave task.
6-7	Job status block address. Address of the job status block for the job to which the slave task belongs.
8-9	Security information.

Basic I/O Block

10	A	>00	<Return Code>
12	C	Sub-opcode	LUNO
14	E	<System Flags>	User Flags
16	10	Data Buffer Offset	
18	12	Read Character Count	
20	14	Write Character Count/<Actual Read Count>	

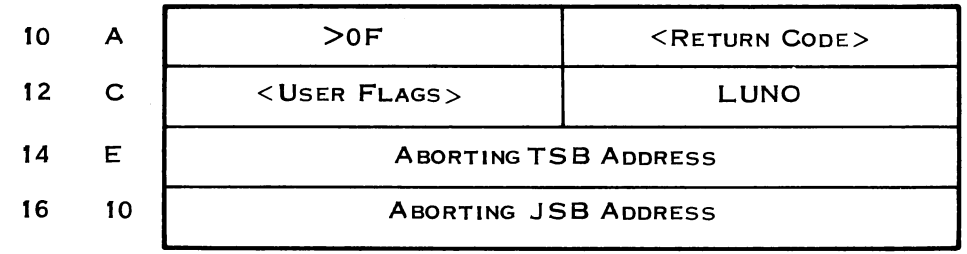
2283156

The basic I/O block of the MRB contains this information:

Byte	Contents
10	Opcode, >00.
11	Return code. The master task may return 0 when the operation completes satisfactorily. The task may return an error code when the operation completes in error.
12	Sub-opcode for desired operation. This field contains >91 for an assign LUNO operation.
13	Logical unit number (LUNO).
14	System flags. DNOS sets the busy flag (bit 0) before returning the call block to the master task.
15	User flags. You can set these flags for utility operations. The flags for the operations that use this field are specified and described in the paragraph on each operation.

Byte	Contents
16-17	Data buffer offset. Not used (reserved) for an assign LUNO operation.
18-19	Read character count. Not used (reserved) for an assign LUNO operation.
20-21	Write character count. Not used (reserved) for an assign LUNO operation.

ABORT I/O BLOCK



2283157

The abort I/O block of the MRB contains this information:

Byte	Contents
10	Opcode, >0F.
11	Return code. The master task may return 0 when the operation completes satisfactorily. The task may return an error code when the operation completes in error.
12	User flags. Set as follows: 0 Do not close flag. 1 — Do not close the LUNO if the task aborts. 0 — Close the LUNO if the task aborts. 1-7 Reserved.
13	Logical unit number (LUNO).
14-15	Aborting TSB address or 0 if task is aborting its own I/O.
16-17	Aborting JSB address or 0 if task is aborting its own I/O.

Extended I/O Block

22	16	Key Definition Block Offset
24	18	Reserved
26	1A	Utility Flags
28	1C	Defined Logical Record Length
30	1E	Defined Physical Record Length
32	20	Pathname Offset
34	22	Parameter Offset
36	24	Reserved
38	26	Initial File Allocation
40	28	
42	2A	Secondary File Allocation
44	2C	

2286061

The MRB contains the following:

Byte	Contents
15	User flags. You can set these flags for utility operations. The flags for those operations that use this field are specified and described in the paragraph pertaining to each operation.
16-17	Data buffer offset. Not used (reserved) for an assign LUNO operation.
18-19	Read character count. Not used (reserved) for an assign LUNO operation.
20-21	Write character count. Not used (reserved) for an assign LUNO operation.
22-23	Reserved.
24-25	Reserved.
26-27	Utility flags. You can set these flags for utility operations. The flags for each operation are specified and described in the paragraph pertaining to each operation.

Byte	Contents
28–29	Logical record length. Applies to Create operations.
30–31	Physical record length. Applies to Create operations.
32–33	Pathname offset. For all operations except Release LUNO, contains the relative address of a buffer that contains the pathname. The address is relative to byte 0 of the MRB. The format of the pathname buffer is: 0 — Length n of pathname in bytes 1–n — Pathname
34–35	Parameter offset. Address of the parameter buffer relative to byte 0 of the MRB. The parameters are logical name parameters, in the format described in the Name Management SVC description.
36–37	Reserved.
38–41	Initial file allocation. Applies to a Create operation.
42–45	Secondary file allocation. Applies to a Create operation for an expandable file.

RKN704

6

ASCII Character Codes

Table 6-1. ASCII Character Codes

Character	ASCII (Hexadecimal)	Character	ASCII (Hexadecimal)
NUL	00	BEL	07
SOH	01	BS	08
STX	02	HT	09
ETX	03	LF	0A
EOT	04	VT	0B
ENQ	05	FF	0C
ACK	06	CR	0D

Table 6-1. ASCII Character Codes (Continued)

Character	ASCII (Hexadecimal)	Character	ASCII (Hexadecimal)
SO	0E	US	1F
SI	0F	Space	20
DLE	10	!	21
DC1	11	"	22
DC2	12	#	23
DC3	13	$	24
DC4	14	%	25
NAK	15	&	26
SYN	16	'	27
ETB	17	(	28
CAN	18	)	29
EM	19	*	2A
SUB	1A	+	2B
ESC	1B	,	2C
FS	1C	-	2D
GS	1D	.	2E
RS	1E	/	2F

Table 6-1. ASCII Character Codes (Continued)

Character	ASCII (Hexadecimal)	Character	ASCII (Hexadecimal)
0	30	A	41
1	31	B	42
2	32	C	43
3	33	D	44
4	34	E	45
5	35	F	46
6	36	G	47
7	37	H	48
8	38	I	49
9	39	J	4A
:	3A	K	4B
;	3B	L	4C
<	3C	M	4D
=	3D	N	4E
>	3E	O	4F
?	3F	P	50
@	40	Q	51

Table 6-1. ASCII Character Codes (Continued)

Character	ASCII (Hexadecimal)	Character	ASCII (Hexadecimal)
R	52	c	63
S	53	d	64
T	54	e	65
U	55	f	66
V	56	g	67
W	57	h	68
X	58	i	69
Y	59	j	6A
Z	5A	k	6B
[	5B	l	6C
\	5C	m	6D
]	5D	n	6E
^	5E	o	6F
_	5F	p	70
`	60	q	71
a	61	r	72
b	62	s	73

Table 6-1. ASCII Character Codes (Continued)

Character	ASCII (Hexadecimal)	Character	ASCII (Hexadecimal)
t	74		
u	75		
v	76		
w	77		
x	78		
y	79		
z	7A		
{	7B		
|	7C		
}	7D		
~	7E		
DEL	7F		

7

Job and Task States

Table 7-1. Job State Codes

Code	Job State
01	Being created
02	Executable state
03	Halted
04	Terminating
05	JCA being expanded

Table 7-2. Task State Codes

Code (Hexadecimal)	Significance
00	In ready state
01	Awaiting memory
02	Job in a nonexecutable state
03	Task being loaded into memory
04	Terminated
05	In time delay
06	Unconditionally suspended
07	Awaiting Ten X processor completion
09	Suspended for I/O
0F	Suspended for aborted I/O
14	Awaiting overlay load services
17	Awaiting co-routine activation
19	Awaiting completion of initiated I/O
1E	Awaiting system semaphore
1F	Awaiting scheduled task bid
20	Awaiting install volume completion
22	Awaiting disk management services
24	Awaiting queue input

Table 7-2. Task State Codes (Continued)

Code (Hexadecimal)	Significance
25	Awaiting task installation
26	Awaiting procedure installation
27	Awaiting overlay installation
28	Awaiting completion of task deletion
29	Awaiting completion of procedure deletion
2A	Awaiting completion of overlay deletion
2B	Suspended by Bid Task SVC
2D	Awaiting read/write task completion
30	Awaiting system table area
31	Awaiting map program name to ID completion
34	Awaiting unload volume completion
36	Awaiting any I/O
37	Awaiting assignment of program file space
38	Awaiting initialize new volume completion
3D	Suspended by Semaphore SVC
40	Awaiting segment management services
42	Awaiting event completion

Table 7-2. Task State Codes (Continued)

Code (Hexadecimal)	Significance
43	Awaiting name management services
48	Awaiting job management services
4A	Awaiting forced swap completion
4C	Awaiting return code processing completion

Appendix A

Keycap Cross-Reference

Generic keycap names that apply to all terminals are used for keys on keyboards throughout this manual. This appendix contains specific keyboard information to help you identify individual keys on any supported terminal. For instance, every terminal has an Attention key, but not all Attention keys look alike or have the same position on the keyboard. You can use the terminal information in this appendix to find the Attention key on any terminal.

The terminals supported are the 931 VDT, 911 VDT, 915 VDT, 940 EVT, the Business System terminal, and hard-copy terminals (including teleprinter devices). The 820 KSR has been used as a typical hard-copy terminal. The 915 VDT keyboard information is the same as that for the 911 VDT except where noted in the tables.

Appendix A contains three tables and keyboard drawings of the supported terminals.

Table A-1 lists the generic keycap names alphabetically and provides illustrations of the corresponding keycaps on each of the currently supported keyboards. When you need to press two keys to obtain a function, both keys are shown in the table. For example, on the 940 EVT the Attention key function is activated by pressing and holding down the Shift key while pressing the key labeled PREV FORM NEXT. Table A-1 shows the generic keycap name as Attention, and a corresponding illustration shows a key labeled SHIFT above a key named PREV FORM NEXT.

Function keys, such as F1, F2, and so on, are considered to be already generic and do not need further definition. However, a function key becomes generic when it does not appear on a certain keyboard but has an alternate key sequence. For that reason, the function keys are included in the table.

Multiple key sequences and simultaneous keystrokes can also be described in generic keycap names that are applicable to all terminals. For example, you use a multiple key sequence and simultaneous keystrokes with the log-on function. You log on by *pressing the Attention key, then holding down the Shift key while you press the exclamation (!) key.* The same information in a table appears as *Attention/(Shift)!*.

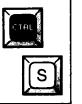
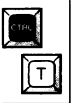
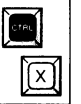
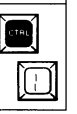
Table A-2 shows some frequently used multiple key sequences.

Table A-3 lists the generic names for 911 keycap designations used in previous manuals. You can use this table to translate existing documentation into generic keycap documentation.

Figures A-1 through A-5 show diagrams of the 911 VDT, 915 VDT, 940 EVT, 931 VDT, and Business System terminal, respectively. Figure A-6 shows a diagram of the 820 KSR.

2274834 (1/14)

Table A-1. Generic Keycap Names

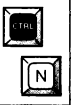
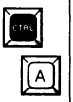
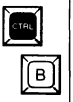
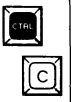
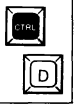
Generic Name	911 VDT	940 EVT	931 VDT	Business System Terminal	820[1] KSR
Alternate Mode	None				None
Attention[2]					
Back Tab	None			None	
Command[2]					
Control					
Delete Character					None
Enter					
Erase Field					

Notes:

[1]The 820 KSR terminal has been used as a typical hard-copy terminal with the TPD Device Service Routine (DSR). Keys on other TPD devices may be missing or have different functions.

[2]On a 915 VDT the Command Key has the label F9 and the Attention Key has the label F10.

Table A-1. Generic Keycap Names (Continued)

Generic Name	911 VDT	940 EVT	931 VDT	Business System Terminal	820[1] KSR
Erase Input					
Exit					
Forward Tab					
F1					
F2					
F3					
F4					

Notes:

[1]The 820 KSR terminal has been used as a typical hard-copy terminal with the TPD Device Service Routine (DSR). Keys on other TPD devices may be missing or have different functions.

2284734 (3/14)

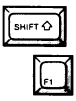
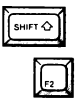
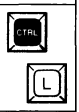
Table A-1. Generic Keycap Names (Continued)

Generic Name	911 VDT	940 EVT	931 VDT	Business System Terminal	820[1] KSR
F5					
F6					
F7					
F8					
F9					
F10					

Notes:

[1]The 820 KSR terminal has been used as a typical hard-copy terminal with the TPD Device Service Routine (DSR). Keys on other TPD devices may be missing or have different functions.

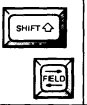
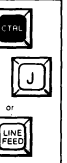
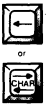
Table A-1. Generic Keycap Names (Continued)

Generic Name	911 VDT	940 EVT	931 VDT	Business System Terminal	820[1] KSR
F11					
F12					
F13					
F14					
Home					
Initialize Input					

Notes:

[1]The 820 KSR terminal has been used as a typical hard-copy terminal with the TPD Device Service Routine (DSR). Keys on other TPD devices may be missing or have different functions.

Table A-1. Generic Keycap Names (Continued)

Generic Name	911 VDT	940 EVT	931 VDT	Business System Terminal	820[1] KSR
Insert Character					None
Next Character	 or 				None
Next Field					None
Next Line					
Previous Character					None
Previous Field					None

Notes:

[1]The 820 KSR terminal has been used as a typical hard-copy terminal with the TPD Device Service Routine (DSR). Keys on other TPD devices may be missing or have different functions.

2284734 (6/14)

Table A-1. Generic Keycap Names (Continued)

Generic Name	911 VDT	940 EVT	931 VDT	Business System Terminal	820[1] KSR
Previous Line					
Print					None
Repeat		See Note 3	See Note 3	See Note 3	None
Return					
Shift					
Skip					None
Upperoase Lock					

Notes:

[1]The 820 KSR terminal has been used as a typical hard-copy terminal with the TPD Device Service Routine (DSR). Keys on other TPD devices may be missing or have different functions.

[3]The keyboard is typamatic, and no repeat key is needed.

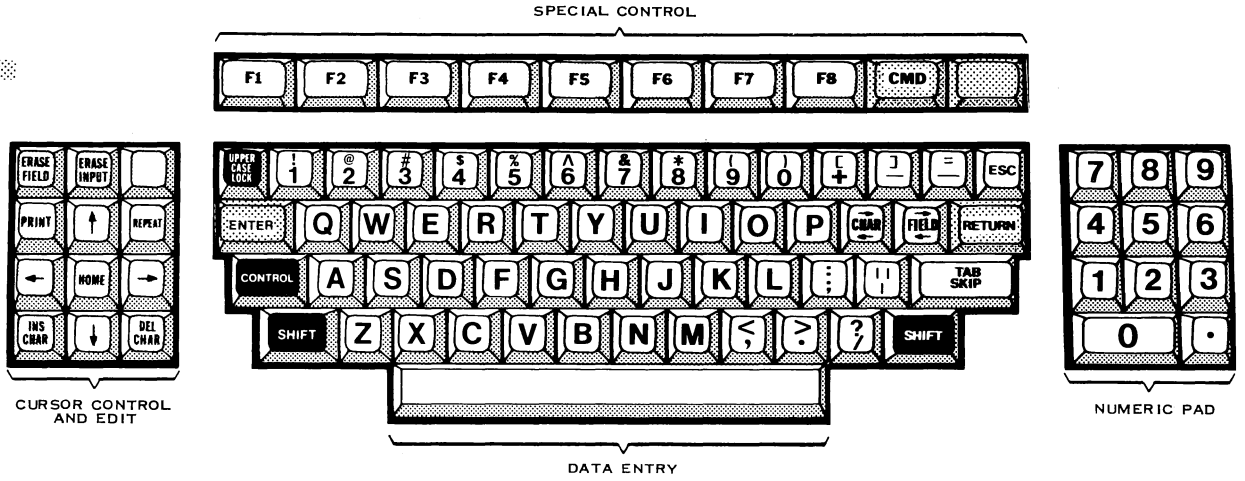
Table A-2. Frequently Used Key Sequences

Function	Key Sequence
Log-on	Attention/(Shift)!
Hard-break	Attention/(Control)x
Hold	Attention
Resume	Any key

Table A-3. 911 Keycap Name Equivalents

911 Phrase	Generic Name
Blank gray	Initialize Input
Blank orange	Attention
Down arrow	Next Line
Escape	Exit
Left arrow	Previous Character
Right arrow	Next Character
Up arrow	Previous Line

2284734 (8/14)



2284734 (9/14)

Figure A-1. 911 VDT Standard Keyboard Layout

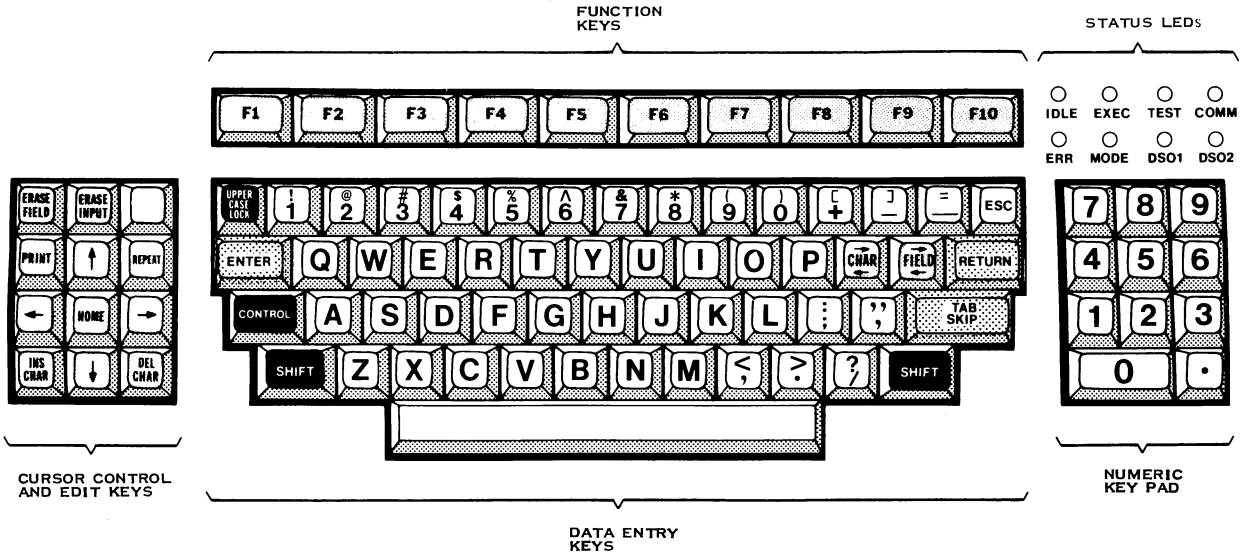
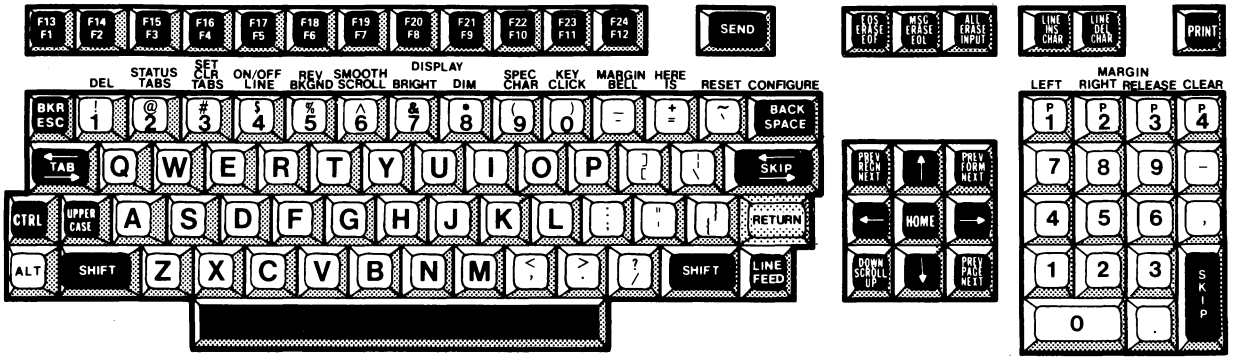
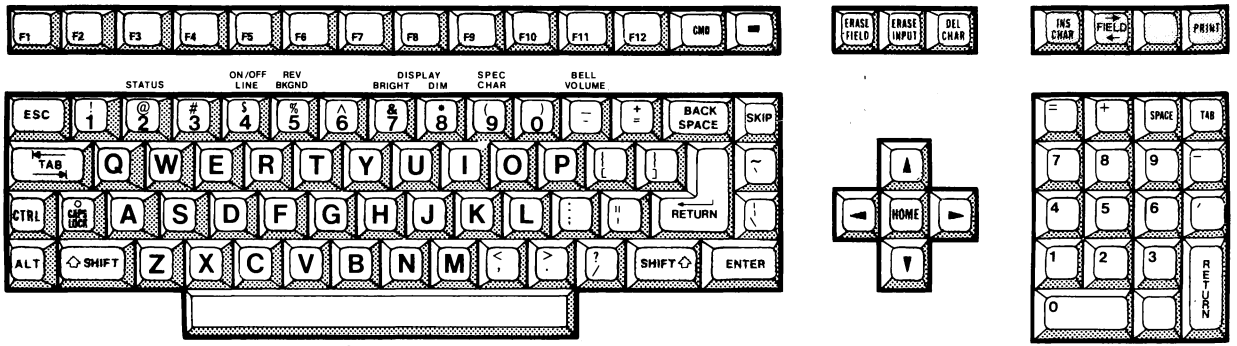


Figure A-2. 915 VDT Standard Keyboard Layout



2284734 (11/14)

Figure A-3. 940 EVT Standard Keyboard Layout



2284734 (12/14)

Figure A-4. 931 VDT Standard Keyboard Layout

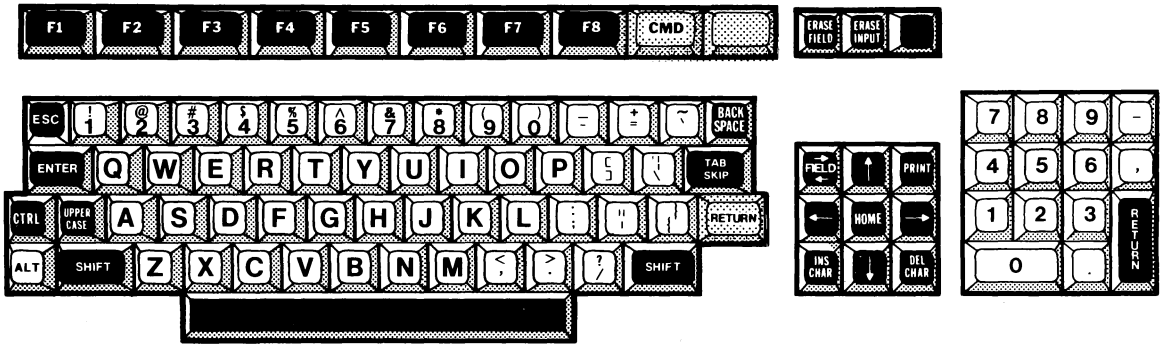
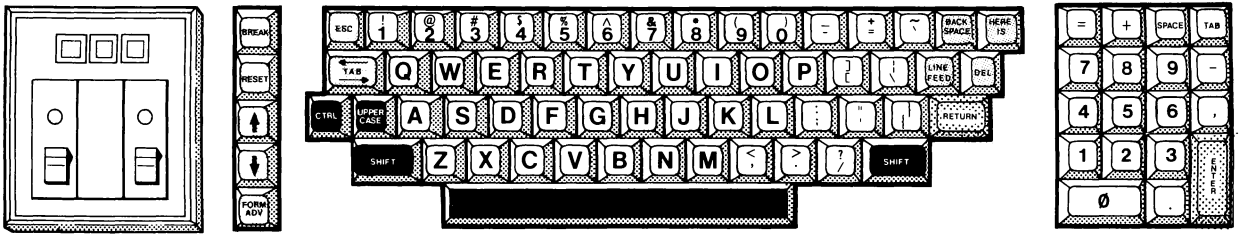


Figure A-5. Business System Terminal Standard Keyboard Layout



2284734 (14/14)

Figure A-6. 820 KSR Standard Keyboard Layout

Alphabetical Index

Introduction

HOW TO USE INDEX

The index, table of contents, list of illustrations, and list of tables are used in conjunction to obtain the location of the desired subject. Once the subject or topic has been located in the index, use the appropriate paragraph number, figure number, or table number to obtain the corresponding page number from the table of contents, list of illustrations, or list of tables.

INDEX ENTRIES

The following index lists key words and concepts from the subject material of the manual together with the area(s) in the manual that supply major coverage of the listed concept. The numbers along the right side of the listing reference the following manual areas:

- Sections — Reference to Sections of the manual appear as "Sections x" with the symbol x representing any numeric quantity.

- Appendixes — Reference to Appendixes of the manual appear as "Appendix y" with the symbol y representing any capital letter.

- Paragraphs — Reference to paragraphs of the manual appear as a series of alphanumeric or numeric characters punctuated with decimal points. Only the first character of the string may be a letter; all subsequent characters are numbers. The first character refers to the section or appendix of the manual in which the paragraph may be found.

- Tables — References to tables in the manual are represented by the capital letter T followed immediately by another alphanumeric character (representing the section or appendix of the manual containing the table). The second character is followed by a dash (-) and a number.

  Tx-yy

- Figures — References to figures in the manual are represented by the capital letter F followed immediately by another alphanumeric character (representing the section or appendix of the manual containing the figure). The second character is followed by a dash (-) and a number.

  Fx-yy

- Other entries in the Index — References to other entries in the index preceded by the word "See" followed by the referenced entry.

NOTES

NOTES

USER'S RESPONSE SHEET

Manual Title: DNOS Reference Handbook (2270505-9701)

Manual Date: March 1985 **Date of This Letter:** _____

User's Name: _____ **Telephone:** _____

Company: _____ **Office/Department:** _____

Street Address: _____

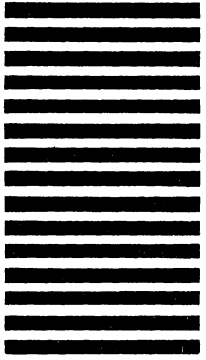
City/State/Zip Code: _____

Please list any discrepancy found in this manual by page, paragraph, figure, or table number in the following space. If there are any other suggestions that you wish to make, feel free to include them. Thank you.

Location in Manual	Comment/Suggestion
_____	_____
	_____
_____	_____
	_____

NO POSTAGE NECESSARY IF MAILED IN U.S.A.

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES



BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 7284 DALLAS, TX

POSTAGE WILL BE PAID BY ADDRESSEE

TEXAS INSTRUMENTS INCORPORATED

DIGITAL SYSTEMS GROUP

ATTN: TECHNICAL PUBLICATIONS
P.O. Box 2909 M/S 2146
Austin, Texas 78769