DNOS

Concepts and Facilities

Part No. 2270501-9701 *C
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INSERT LATEST CHANGED PAGES AND DISCARD SUPERSEDED PAGES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 Concepts and Facilities (2270501-9701 *C)

Original Issue . . . . . August 1981
Revision . . . . . October 1982
Revision . . . . . November 1983
Change 1 . . . . . March 1985

Total number of pages in this publication is 108 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.

© 1981, 1982, 1983, 1985, Texas Instruments Incorporated. All Rights Reserved

Printed in U.S.A.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of Texas Instruments Incorporated.

Manual Update

MANUAL: DNOS Concepts and Facilities (2270501-9701 *C)

MCR/CHANGE NO.: MCR 004679/Change 1

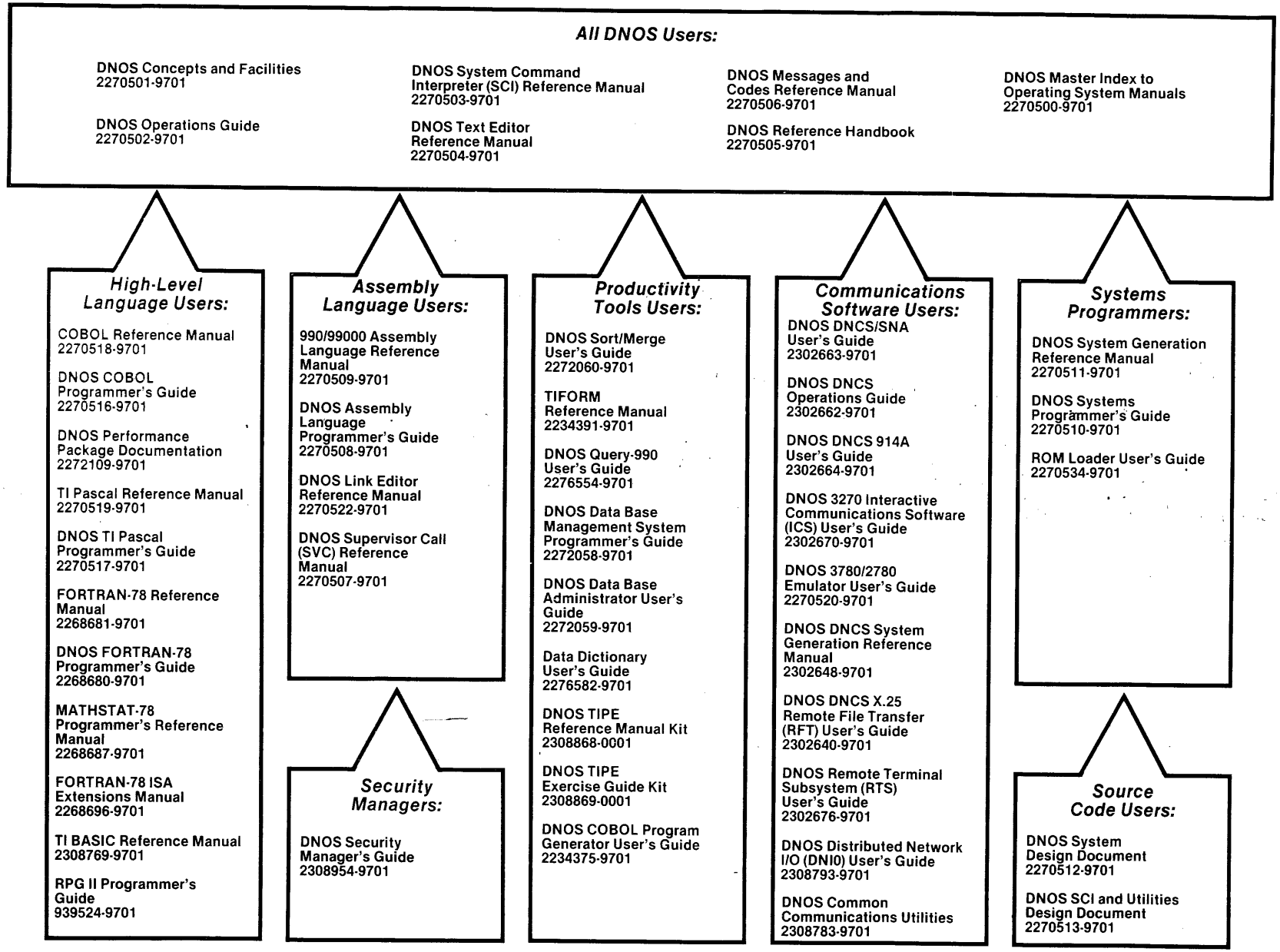
EFFECTIVITY DATE: 20 March 1985

This change package contains information necessary to update your current manual. Please remove the obsolete pages from your existing manual and replace them with the changed pages as follows:

Remove Obsolete Pages	Insert Change 1 Pages
Cover/Manual Revision History	Cover/Effective Pages
iii - iv	iii - iv
vii - viii	vii - viii
1-1 - 1-2	1-1 - 1-2
1-5 - 1-8	1-5 - 1-8
2-3/2-4	2-3/2-4
3-1 - 3-4	3-1 - 3-4
3-11 - 3-14	3-11 - 3-16
8-3 - 8-4	8-3 - 8-4
Index-1 - Index-2	Index-1 - Index-2
User's Resp./Bus. Reply	User's Resp./Bus. Reply
Inside Cover/Cover	Inside Cover/Cover

DNOS Software Manuals

This diagram shows the manuals supporting DNOS, arranged according to user type. Refer to the block identified by your user group and all blocks above that set to determine which manuals are most beneficial to your needs.



DNOS Software Manuals Summary

Concepts and Facilities
Presents an overview of DNOS with topics grouped by operating system functions. All new users (or evaluators) of DNOS should read this manual.

DNOS Operations Guide
Explains fundamental operations for a DNOS system. Includes detailed instructions on how to use each device supported by DNOS.

System Command Interpreter (SCI) Reference Manual
Describes how to use SCI in both interactive and batch jobs. Describes command procedures and gives a detailed presentation of all SCI commands in alphabetical order for easy reference.

Text Editor Reference Manual
Explains how to use the Text Editor on DNOS and describes each of the editing commands.

Messages and Codes Reference Manual
Lists the error messages, informative messages, and error codes reported by DNOS.

DNOS Reference Handbook
Provides a summary of commonly used information for quick reference.

Master Index to Operating System Manuals
Contains a composite index to topics in the DNOS operating system manuals.

Programmer's Guides and Reference Manuals for Languages
Contain information about the languages supported by DNOS. Each programmer's guide covers operating system information relevant to the use of that language on DNOS. Each reference manual covers details of the language itself, including language syntax and programming considerations.

Performance Package Documentation
Describes the enhanced capabilities that the DNOS Performance Package provides on the Model 990/12 Computer and Business System 800.

Link Editor Reference Manual
Describes how to use the Link Editor on DNOS to combine separately generated object modules to form a single linked output.

Supervisor Call (SVC) Reference Manual
Presents detailed information about each DNOS supervisor call and DNOS services.

DNOS System Generation Reference Manual
Explains how to generate a DNOS system for your particular configuration and environment.

User's Guides for Productivity Tools
Describe the features, functions, and use of each productivity tool supported by DNOS.

User's Guides for Communications Software
Describe the features, functions, and use of the communications software available for execution under DNOS.

Systems Programmer's Guide
Discusses the DNOS subsystems and how to modify the system for specific application environments.

ROM Loader User's Guide
Explains how to load the operating system using the ROM loader and describes the error conditions.

DNOS Design Documents
Contain design information about the DNOS system, SCI, and the utilities.

DNOS Security Manager's Guide
Describes the file access security features available with DNOS.

Preface

This book provides general background information about DNOS features, concepts, and facilities. It is one of several documents that describes the operational characteristics and features of DNOS. You should become acquainted with these documents, as necessary, to prepare and execute application programs under DNOS. The DNOS family of documents is shown in the frontispiece of this book.

This book is organized into nine sections and a glossary.

Section

1 Introduction — Introduces DNOS and some of the primary features of the system.

2 System Command Interpreter — Explains how to communicate interactively with DNOS.

3 Application Development — Describes the application tools that are available with DNOS.

4 Program Management — Introduces the job and task concepts and also details several concepts used to provide program management.

5 DNOS Memory Organization — Describes, in general, how memory is organized and managed in DNOS.

6 I/O Resource Management — Explains the allocation of resources, input/output (I/O) methods, logical names, and the spooling process.

7 Interprocess Communication (IPC) — Introduces IPC and describes its use.

8 File Organization — Details the various file types supported by DNOS, describes file features, and explains file security.

9 Messages and Exception Handling — Describes the message structure, diagnostics, and power failure handling capabilities of DNOS.

Appendix

A Keycap Cross-Reference — Gives the generic key names in a chart that relates the generic name to the key on a specific terminal.

DNOS Glossary

Provides a list of terms that describes operating systems in general, and DNOS in particular.

Contents

4 — Program Management

5 — DNOS Memory Organization

6 — I/O Resource Management

7 — Interprocess Communication (IPC)

Illustrations

Tables

1

Introduction

1.1 DNOS FEATURES

The Distributed Network Operating System (DNOS) is a general-purpose, multitasking operating system designed to operate with the Texas Instruments Business System minicomputers. DNOS includes a sophisticated file management package which provides support for key indexed files, sequential files, and relative record files. DNOS is a multiterminal system that is capable of making each of several users appear to have exclusive control of the system. DNOS supports output spooling and program-accessible accounting data. Job-level and task-level operations enable more efficient use of system resources.

In addition to multiterminal applications, DNOS provides support for advanced program development. Users communicate with DNOS by entering commands at a terminal or by providing a file of commands. The System Command Interpreter (SCI) processes those commands and directs the operating system to initiate the action specified by a command. A text editor allows the user to enter source programs or data into the system. A macro assembler is provided for assembly language programs. Several high-level languages, including FORTRAN, COBOL, BASIC, RPG II, and Pascal, are supported. A link editor and extended debugging facilities are also provided. A variety of utility programs and productivity tools support access to and management of information contained in a data base, design of specific forms on the screen of a video display terminal (VDT), and word processing.

The system supports a wide range of user environments. DNOS can support as few as one or two terminals, thus allowing the user of a smaller Business System to perform tasks efficiently and yet inexpensively. Larger configurations with a wide variety of peripherals are also supported. The maximum configuration size varies with the user's environment. Almost every minicomputer system requirement or application need can be met with DNOS.

DNOS provides the base for a variety of communications products. Standard protocols for 3780/2780 and for 3270 communications are supported. Local area network software is supported for network input/output (I/O) and log-on. In addition, sophisticated networking software is available with the Distributed Network Communications System (DNCS) and Distributed Network I/O (DNIO) packages. DNCS includes networking capabilities for X.25 and Systems Network Architecture (SNA) protocols. DNIO provides users transparent access to other TI 990s running DNOS and can be used to connect to local or wide-area networks.

DNOS is an international operating system designed to meet the commercial requirements of the United States, most European countries, and Japan. DNOS supports a complete range of international data terminals that permit users to enter, view, and process data in their own languages. Specialized data terminals are presently available for the languages used in these countries: Austria, Belgium, Denmark, Finland, France, Germany, Japan, Norway, Spain, Sweden, the United Kingdom, and the United States. The system includes error text files that can be edited so that error messages can be easily translated into languages other than English.

DNOS supports features that incorporate the computing power of the larger Business System computers, and it is upwardly compatible with other Texas Instruments operating systems. DNOS features include:

Multiple Terminals
: The number of online terminals is limited only by the available computing power and memory for system structures.

File Security
: The system can optionally include a file security system that allows system managers and other users to determine which user groups can access data files and execute specific programs.

Output Spooling
: Output spooling is the process of queueing files for printing. Files are scheduled for printing based on job priority and availability of the printing device(s). You can specify special printing forms and formats.

Accounting Function
: The system can optionally include an accounting function that allows you to maintain accounting information on the use of system resources.

Job Structure
: The system incorporates a job structure that assists program management and enables efficient use of resources. A job is a sequence of cooperating tasks.

I/O Resource Management
: Resource-specific and resource-independent I/O operations allow flexibility in the selection of devices and file types.

Program Segmentation
: Program segmentation is the process of separating a program into segments. A program can consist of up to three segments at any one time. Additional segments can be accessed as necessary during program execution. Segments can be shared by programs.

Interprocess Communication
: The system provides the capability, through interprocess communication (IPC), for two programs (tasks) to exchange information.

Power Failure Recovery
: Should a power failure occur, DNOS maintains the state of the system at the time of the failure, if the optional software and backup power supply have been added to your system. When full power resumes, the operation will continue from the point at which the power failure occurred.

Synchronization Methods

Event and semaphore synchronization methods are included to assist interaction between programs, either within a job or across job boundaries. Event synchronization allows the program to wait for one or more events to be completed before processing is continued. Semaphore synchronization uses variables to exchange signal information between programs.

Concatenated Files

The system supports file concatenation, in which two or more physical files are recognized as a logically contiguous set of data. These files can exist on one or more volumes.

Temporary Files

A temporary file is one that exists only during the life of the created job, or during the extent of a program within that job. A job temporary file can be accessed by any program in a job, and is deleted when the job terminates. Other temporary files are created for use by a single program and are deleted when the program terminates.

Diagnostic Support

The system supports online diagnostics that operate concurrently with program execution and system log analysis tasks.

Batch Jobs

A batch job is a job that executes in the background, independent of a terminal. A user at a terminal can be executing multiple batch jobs, and at the same time, be performing foreground and/or background operations in an interactive job.

Dynamic Configuration Changes

Table size, system characteristics, and device configuration changes can be enabled and take effect after the next Initial Program Load (IPL) sequence, rather than during system generation.

Compatibility

DNOS design enables compatibility with the DX10 operating system. Many of the familiar operating concepts of the DX10 operating system are integrated within the design of DNOS. DNOS includes disk and other media formats, a Supervisor Call (SVC) interface, and SCI user commands that are all upwardly compatible with DX10.

System Generation

The system generation utility allows a user to interactively specify all necessary features, available devices, and optional functions when creating an operating system. This data is used to construct a file that defines the configuration of the operating system.

Message Facilities

The system provides a comprehensive set of codes and messages describing errors and special conditions that occur when using the operating system. The system handles messages in a uniform manner to ensure that they are easy to use. Two system directories maintain message files that contain text describing errors, information, and completion messages generated by the system. The directories are expandable to include message files written by users.

System Log
The system log stores information about errors and messages generated by hardware, input/output operations, tasks, and user programs on two files and an optional specified device.

Systems Problem Analysis
If problems occur during system operation, they can be diagnosed by using a system utility that can analyze the system whether the system is operating or has experienced a failure. In a failure situation, an image of memory can be copied to a file. When the system is operating again, an analysis utility can be used with a variety of commands entered by the user.

System Configuration History
Information about all supplied software products installed on a system is maintained on a system disk file. Users can also send entries to the file for application products they develop.

DNOS International Features
Error message files can be text edited to translate into a language other than English. It is also necessary to change the collating sequence of key indexed files (KIF) according to the translation. DNOS provides methods to change the required collating sequence.

DNOS Performance Package
An optional add-on package is available for DNOS on the larger Business System series computers. This package enhances DNOS performance by using several system routines implemented in microcode in the writable control storage.

1.2 DNOS SYSTEM CONFIGURATIONS

System configuration refers to the arrangement or occurrence of various devices and functional features that meet the requirements of a particular user or application. The hardware configuration allows a wide range of peripheral devices to be used (refer to Table 1-1). The software configuration is defined during the system generation process to meet specific requirements of the installation. Most of these items can also be dynamically configured during system operation.

1.2.1 Hardware Configuration

The following hardware configuration is the minimum requirement for DNOS:

- A Business System computer with the memory mapping option and a minimum random-access memory of 512K bytes
- A Texas Instruments VDT or teleprinter device (TPD)
- A system disk
- A means of disk backup

Disk backup can be magnetic tape, a second disk drive, or a cartridge disk. DNOS supports the hardware devices that are presently available and listed in Table 1-1. In addition, many devices supported in previous releases are still supported, although they are no longer on the price list.

Table 1-1. DNOS Hardware Devices

Device Type	Model Number	Description
Disk Drives	DS80	Moving-head disk drive with five- platter removable disk pack; provides 67.2M bytes formatted storage.
	DS300	Moving-head disk drive with 10-platter removable disk pack; provides 238.2M bytes formatted storage.
	CD1400/32	Moving-head disk drive with two units, one fixed and one removable. Each unit provides 13.45M bytes formatted storage.
	CD1400/96	Moving-head disk drive with fixed and removable units. Fixed unit contains three platters that provide 67.2M bytes formatted storage. Removable unit has a single platter with 13.45M bytes formatted storage.
	WD500	Five-inch Winchester disk system with one or two fixed unit 10M byte disks and an eight-inch, double-sided, double-density diskette drive backup that can also read single-sided, single-density diskettes.
	WD500A	5 1/4-inch Winchester disk systems with FD1000 backup. Two configurations are available: one fixed disk with 5M bytes of formatted storage or two fixed disks with 17M bytes.
	WD800/18	Eight-inch Winchester disk system with 18.5M bytes of formatted storage and a cartridge tape backup system with 14.5M bytes formatted storage per cartridge.
	WD800/43	Eight-inch Winchester disk system with 43.2M bytes of formatted storage, and a cartridge tape backup system with 14.5M bytes formatted storage per cartridge.
	WD800A/38 WD800A/69 WD800A/114	5 1/4-inch Winchester disk systems with cartridge tape backup. The WD800A/38 has 38.5M bytes formatted storage, the WD800/69 has 69.5M bytes formatted storage, and the WD800A/114 has 114.6M bytes formatted storage.
	WD900/138 WD900/425	Nine-inch Winchester disk systems available with 138M or 425M bytes formated storage.

Table 1-1. DNOS Hardware Devices (Continued)

Device Type	Model Number	Description
Terminals	911 VDT	Video display terminal (VDT) that displays 80 characters per line, 24 lines at a time (1,920 characters). The 911 VDT supports uppercase and lowercase characters, displayed in high or low intensity, as well as control characters. The 911 VDT is available in international versions, supporting the local character set in both uppercase and lowercase.
Terminals	733 ASR/KSR	Hard-copy data terminals using a thermal type dot matrix printhead to print up to 30 characters per second (cps). The 733 ASR/KSR terminals are automatic send/receive and keyboard send/receive hard-copy data terminals.
	Silent 700™ Terminal	Portable keyboard send/receive hard-copy data terminals using a thermal type dot matrix printhead. These terminals support the ASCII standard keyboard. Several models include an acoustic coupler for remote capabilities.
	820 KSR	Keyboard send/receive hard-copy data terminal using a 9 x 7 dot matrix printhead to print up to 150 cps. Characters received through an EIA line interface are held in a 640-character buffer and are printed either unidirectionally or bidirectionally for optimum efficiency.
	931 VDT	VDT displays 80 characters per line, 24 lines at a time (1,920 characters). The 931 VDT supports a 128 ASCII character set with enhanced displays, and can use EIA-standard RS-232-C as well as fiber optics interfaces. The 931 VDT includes an auxiliary printer port to transmit output independently of the host CPU.
Printers	810	Printer uses a 9 x 7 dot matrix printhead to print 132 columns bidirectionally at 150 cps. Vertical forms control is supported. Optionally, the printer provides a full uppercase and lowercase ASCII character set and vertical forms control.
	840 RO	Printer uses a 9 x 7 (or optional 9 x 9) dot matrix printhead for bidirectional printing at 75 cps. The 840 features a 256-character receive buffer, device forms control, and enhanced print options. A stored-program microprocessor controls printer functions.
	850	Desktop-size printer uses a 9 x 9 dot matrix printhead for bidirectional printing at 150 cps. The 850 supports two fonts printed in standard, compressed, double-wide, compressed double-wide, and enhanced modes, and can print mosaic or raster graphics. Available with international character sets, the 850 uses either serial or parallel interfaces.

Silent 700 is a trademark of Texas Instruments Incorporated.

Table 1-1. DNOS Hardware Devices (Continued)

Device Type	Model Number	Description
	855	Desktop-size printer with two modes of operation. Fast draft mode uses a 9 x 9 dot matrix printhead for bidirectional printing at 150 cps. The correspondence mode (near letter quality) makes two passes per line at 35 cps, with a 32 x 18 dot matrix printhead. Both modes have a maximum line width of eight inches. The 855 uses interchangeable plug-in font modules.
	LP300	Medium-speed line printer prints 132-character lines at a rate of 300 lines per minute.
	LP600	High-speed line printer prints 132-character lines at up to 600 lines per minute.
	LQ45	Letter quality printer uses a daisywheel printing element to print at speeds up to 45 characters per second. The printer supports a full 96 character printwheel.
Other I/O Devices	804 Card Reader	Column-oriented serial card reader that reads 80-column cards at a rate of 400 cards per minute. The 804 uses a photoelectric sensor and can read both punched and mark-sense cards.
	MT1600 Magnetic Tape	Serial access, nine-track magnetic tape transport with a standard recording density of 1,600 bits per inch.
	MT3200 Magnetic Tape	Streaming cache, serial-access, nine-track magnetic tape transport. Standard recording density of 1600 or optional 3200 bits per inch (phase-encoded).

1.2.2 Software Configuration

The complete software configuration, other than certain optional software packages, is specified during the system generation process. System generation is the process of designing the functional capabilities of the operating system by responding to a series of questions. DNOS system generation offers a high degree of flexibility in designing system software configurations. It allows the specification of standard communications software, as well as operating system components. System generation includes linking system object modules and user-supplied modules that were created specifically for an installation.

1.2.3 Dynamic System Configurability

In addition to the flexibility of system configuration during system generation, DNOS also allows modification of most features of the system while the system is running. A system configuration utility allows you to:

- Add or delete devices
- Modify table area size
- Modify scheduler and swapping parameters

For some features, the changes take place immediately. For others, the new set of system characteristics is in effect after an IPL sequence is performed. Another system generation is not needed.

2

System Command Interpreter

2.1 GENERAL INFORMATION

You communicate interactively with the operating system by entering commands with the keyboard of a video display terminal (VDT) or a hard-copy data terminal (a terminal that prints the commands instead of displaying them). The commands are interpreted by the System Command Interpreter (SCI) and are called SCI commands. When an SCI command is entered, the system can prompt for additional information about the requested operation. The additional information, called *parameters*, is interpreted by SCI, which then initiates the requested operation.

SCI is supported on DNOS interactive devices, including VDTs and hard-copy data terminals. SCI operations on a VDT allow all parameter values to be displayed at once. When using hard-copy data terminals, you are prompted for parameter entry one line at a time.

2.2 FLEXIBLE COMMAND PROCEDURES

All SCI commands are processed by a set of command specifications. The command specifications are coded in a special-purpose language consisting of primitives and other commands. As you enter commands, SCI interprets the specification language and performs the prompting, data entry, and verification functions that are appropriate to the command. Statements within the language structure tell the system what data to collect and to which processing program to pass the data. The SCI language allows you to modify the set of available procedures.

2.3 SYNONYMS

SCI language variables, called *synonyms*, can be used as abbreviations for long text strings frequently entered by a user. When a synonym is entered as a response to an SCI prompt, the system recognizes that synonym as equal to the value of the actual text string. For example, the user can choose an easy-to-remember abbreviation, even one as short as a single character, to represent the longer and more complex pathname of a frequently-used directory. Each user can assign several synonyms, and DNOS maintains a list of active synonyms associated with each user.

2.4 INTERACTIVE AND BATCH MODE SCI

At each terminal, a user can have one foreground task and one background task active at the same time in the interactive job. A foreground task requires the use of the display screen of a VDT or the printing capability of a hard-copy data terminal. Background tasks, although they are associated with the terminal, do not tie up the interactive capability. Batch jobs can be started independently of the interactive job at the terminal, and they make use of a file of SCI commands. DNOS provides the capability of executing foreground and background activity as well as concurrent execution of batch job activity.

When operating in the interactive or batch job environment, logical name and synonym definitions are supported. A logical name can define parameters for resources, as well as identify a resource. Logical names and their use within DNOS are described in Section 6 of this book. A background task can use a logical name definition supplied by the job, or the logical name can be redefined without affecting the foreground definition. As is true with synonyms, after a background task is initiated, changes in foreground definitions do not affect background definitions.

2.5 TAILORING SCI TO AN APPLICATION ENVIRONMENT

DNOS provides extensive support for application development. One of the primary tools is the capability to modify existing SCI commands or add new SCI commands in order to tailor a particular user's SCI to a specific environment.

For those instances in which a detailed menu approach is desired, application programs can be developed that incorporate both existing SCI commands and new SCI commands. (Refer to the *DNOS Systems Programmer's Guide* and the *DNOS System Command Interpreter (SCI) Reference Manual* for detailed information on SCI capabilities.)

2.6 COMMANDS AVAILABLE

DNOS offers a comprehensive set of SCI commands that perform various utility operations. Many other commands are supplied for programmers to use as they develop applications under DNOS. (Refer to the *DNOS System Command Interpreter (SCI) Reference Manual* for complete details on specific SCI commands.) The following is a list of the functional areas addressed by one or more SCI commands.

- Log on and off
- Set and display the time and date
- Initialize, install, and unload disk volumes
- Restore, backup, and copy disk directories
- Create and delete directories, files, and IPC channels
- Support synonyms
- Support file security
- Change file names and protection
- View and list directories and files
- Copy files
- Assign, position, and release logical units
- Display I/O status

- Display job and task status
- Activate and control programs and jobs
- Activate and monitor batch status
- Maintain terminal status
- Install and delete programs
- Activate the system log
- Debug programs and include items such as:
  - — Breakpoints
  - — Memory/disk dump or display
  - — Decimal/hexadecimal arithmetic
  - — Interactively controlled program trace
- Control text editing
- Activate Assembler, BASIC, COBOL, FORTRAN, Pascal, and RPG II applications
- Activate Link Editor
- Activate DBMS
- Activate Query
- Activate DD
- Activate TIPE
- Activate CPG
- Start communications systems
- Use TIFORM
- Assign logical names and parameters
- Support spooling system
- Support operator interface
- Support teleprinter devices

3

Application Development

3.1 APPLICATION DEVELOPMENT TOOLS

Texas Instruments provides a comprehensive set of application development tools that operate in combination with DNOS. Some of these tools are standard system utilities; others are optional software packages or communications emulators. The utilities and packages that are available include the following:

- Interactive Text Editor
- Macro Assembler
- Link Editor
- Interactive Debugger
- High-Level Languages (BASIC, COBOL, FORTRAN, Pascal, and RPG II)
- Data Base Management System (DBMS)
- Data Dictionary (DD)
- Interactive DBMS Retrieval (Query)
- Interactive Screen Format Design (TIFORM)
- Sort/Merge
- TIPE
- COBOL Program Generator (CPG)
- Communications Support
- Distributed Network Communications System (DNCS)

3.1.1 Interactive Text Editor

The Text Editor operates under the DNOS System Command Interpreter (SCI). An SCI command activates the Text Editor. When the Text Editor is active, you use the edit control keys to enter, modify, delete, and display data. You can also use Text Editor commands which affect the entire file to efficiently display, move, and modify data. Commands generally require responses to field prompts for information necessary for command execution. You can easily change data at any position in the file, as the system allows positioning to a given character string or line number within the file. You can copy data from other files during your edit. All new data and any changes made become part of the output file created at the end of the edit session. You can edit files up to 240 columns wide. Edit control functions are similar for both video and hard-copy terminals.

3.1.2 Macro Assembler

The DNOS Macro Assembler accepts the standard 990 computer assembly language instructions, and includes a macro facility and conditional processing. The macro facility provides character string manipulation, access to binary values in the symbol table, and support of macro definition libraries. The relocatable object code produced by the assembler can be partitioned into segments. Common blocks, program segments, and data segments have separate location counters. The Link Editor gathers segments of the same type into contiguous memory areas. Conditional processing of a sequence of assembly language statements depends on the value of a logical expression. Directives are provided to specify the amount of information in the assembly listing.

3.1.3 Link Editor

The DNOS Link Editor is used to accept relocatable object code produced by the assembler or a compiler, and it combines the individual modules into a linked module. References between modules are resolved to the correct values. Common blocks, program segments, and data segments are collected. Then each segment type is assigned to a contiguous area of memory.

The Link Editor accepts control statements that can specify the use of shared resources and the use of overlay structures. Available options include generating a load map, searching a set of object libraries to automatically resolve unresolved values, and establishing a partial link that leaves external references for later resolution. Where functions require overlays, an automatic overlay load option is available. Link Editor output can be an object file or can be written directly as memory image format into a program file or DNOS image file.

3.1.4 Interactive Debugger

The Interactive Debugger is a program that is used for symbolic debugging of assembly language tasks. The Debugger can operate interactively from a video display terminal or a hard-copy device. It allows display and modification of the contents of central processing unit (CPU) registers, workspace registers, and memory. The Interactive Debugger also controls execution of a task.

In the run mode, a task can be halted or started, and new breakpoints can be set to halt the task when a breakpoint is encountered.

In the simulation mode, task execution is analyzed between each instruction. You can specify commands to examine the program counter or memory content.

3.2 LANGUAGE SUPPORT

DNOS provides developmental and operational support for programs written in BASIC, COBOL, FORTRAN, Pascal, and RPG II. These high-level languages are available from Texas Instruments as options with DNOS. Basic features of these languages are described in the following paragraphs.

3.2.1 BASIC

BASIC is an easily understood programming language that is well suited for scientific and business problem solving. An interactive language, BASIC is primarily designed for multiuser, multiterminal systems. Using BASIC, the computer will provide feedback to each command at the terminal of the user who entered the command, pointing out any programming errors with diagnostic messages during program execution.

The BASIC language provided by Texas Instruments is a significantly improved version of the American National Standards Institute (ANSI) Minimal BASIC. Improvements include the support of multiple file organizations, closed subroutines, character strings, and VDT screen handling. The BASIC package includes the following components:

- A compatible language, TI BASIC, executable on all minicomputers of the Business System Series
- An interactive reentrant editor/interpreter for creating and executing TI BASIC programs

This BASIC package interacts with the Business System minicomputers to make the most efficient use of program development time. The powerful editor and fast system response deliver a quick turnaround time. The 13-digit precision provides the accuracy needed for scientific calculations. When used for commercial applications, the decimal representation of noninteger variables eliminates the round-off problems of other BASICs.

Language compatibility within the Business System computer family permits the development of BASIC programs on larger Business Systems that can also be used on the smaller Business Systems. Texas Instruments version of BASIC supports key indexed files and temporary files in addition to the relative record and sequential files provided on other BASIC systems.

3.2.2 COBOL

COBOL is a high-level computer language especially designed for business data processing. COBOL consists of a set of English words and symbols that the programmer can use to define the problem and to create a program to solve that problem. The COBOL language is self-documenting because the commands are English-like statements that use familiar business terms.

The COBOL compiler includes the complete ANSI COBOL subset (as defined in the ANSI document X3.23-1974). Texas Instruments incorporates extensions to this subset that provide further capabilities, especially in the areas of I/O and debugging. Table 3-1 shows which areas have been enhanced from the ANSI 74 standard, and to what level those enhancements have advanced. The compiler package employs the following ANSI 74 standard COBOL modules at the level indicated in Table 3-1.

Table 3-1. TI COBOL Features

COBOL Module	Level
Interprogram communications	1
Library	1
Segmentation	1
Nucleus	1 + *
Relative I/O	1 + *
Indexed I/O	1 + *
Sequential I/O	1 + *
Table Handling	1 + *
Debug	Nonstandard

Note:

* Selected features from level 2.

In addition, the ACCEPT/DISPLAY statements provide standard COBOL functions and nonstandard extensions that are designed for ease of use on VDTs.

COBOL object code modules can be executed directly with SCI Execute COBOL commands, or they can be link-edited and installed on DNOS program files for execution as tasks.

3.2.3 FORTRAN-78

FORTRAN is a high-level, general-purpose programming language in which mathematical statements and English words allow the programmer to develop programs that solve problems requiring mathematical and logical computations.

The version of FORTRAN provided for the Business System minicomputers, referred to as FORTRAN-78, complies with the subset specifications described in the ANSI publication USAS X3.9-1978. FORTRAN-78 includes extensions to the ANSI standard subset that provide increased flexibility. Some of the more significant extensions are as follows:

- Dynamic subprogram recursion
- Optional array bounds checking
- Variable names of any length
- DO loop control variables of any numeric type
- ACCEPT/DISPLAY VDT handling statements
- Multirecord internal units with error exit
- Mixed-mode expressions

- Hexadecimal constants and assignments
- Extended integer data type
- Sixteen-bit, fixed-point arithmetic
- Optional direct floating-point arithmetic execution on the Business System 800 minicomputers or 990/12 computers
- Character string manipulation run-time routines
- Key indexed file handling run-time routines
- Full ANSI standard version of the OPEN and CLOSE statements that provide the following additional capabilities:
  - — Specification of file access privileges
  - — Creation of temporary scratch files
  - — Control of blank interpretation during formatted numeric input

In addition, FORTRAN-78 incorporates the extensions to the FORTRAN language recommended by the Instrument Society of America (ISA extensions S61.1-1975).

The FORTRAN-78 package also includes the set of mathematical and statistical subprograms referred to as MATHSTAT-78. The MATHSTAT-78 subprograms perform functions such as matrix operations, polynomial operations, integration, linear programming, Fourier Series analysis, statistical analysis and presentation, regression, and analysis of variance.

3.2.4 Pascal

Texas Instruments Pascal is a general-purpose language well suited for a variety of applications. Pascal is also straightforward, which makes it a fast language to learn and use. Its readability makes it especially useful when programs are maintained by users other than the original author.

One application of Pascal is the development of systems-level software. Pascal adapts well to scientific or engineering applications that are traditionally programmed with FORTRAN or ALGOL. Also, its general-purpose structure makes Pascal useful in many business areas.

Some of the more significant features of Pascal are:

- A block structured format that directly supports structured programming concepts
- Stack allocation of variables for each routine
- Recursive routines
- User-defined data structures that are adaptable to specific applications

- User-defined data types
- Bit manipulation capabilities

Pascal consists of five major components:

- Nester utility
- Configuration processor
- Pascal compiler
- Pascal run-time library
- Reverse assembler

The Nester utility generates source code indented in a standard format to improve readability. The configuration processor supports the separate compilation of nested program modules. The Pascal compiler with optimizing features produces linkable object modules. The Pascal run-time library provides an interface with the operating system. The reverse assembler can produce assembly language source files or listings from object modules created by the compiler.

3.2.5 Report Program Generator, Version II (RPG II)

RPG II is an easy-to-use, high-level language for business data processing. Based on a predetermined sequence that reads a record, processes the data, then outputs the results, RPG II is especially suited for applications requiring file maintenance or report generation. A series of six basic formats is used to specify the actions to be taken within an RPG II sequence execution.

The Texas Instruments version of the RPG II language is closely compatible with the frequently-used IBM System/3 RPG II. Extensions of many of the System/3 features have been included in RPG II to provide more flexible programming. RPG II provides a utility program to copy System/3 or System/32 source programs or files from diskette to Business System disk files.

The RPG II package also includes an RPG II-oriented VDT text editor and a trace feature that prints each major step occurring in the execution of an RPG II program.

3.3 PRODUCTIVITY TOOLS

A set of productivity tools is provided with special-purpose application options that accomplish more specific tasks. The tools available with DNOS include the following:

- DBMS
- DD
- Query

- TIFORM
- Sort/Merge
- TIPE

3.3.1 DBMS

The Data Base Management System (DBMS) is designed for minicomputer data base applications. DBMS handles data access in a logical format similar to the use of physical documents or records in daily business transactions. DBMS allows you to define and access a centralized, integrated data base without the physical data access requirements necessitated by conventional file management software. Considerations such as access method, record size, blocking, and relative field positions are resolved when the data base is initially defined. Thus, you can concentrate fully on the logical data structure needed for interface operations.

Since the data definitions are independent from the application software, the data base can be changed without affecting existing programs. DBMS also provides a single, centralized copy of the data to be used for all application subsystems. (Conventional file management results in fragmented and/or multiple copies of data, one for each application.) A centralized copy results in more efficient data storage on disk, uniform processing of data requests, and simpler data base maintenance.

Security is an optional feature of DBMS. Its purpose is to eliminate unauthorized use of the data base. Password security controls file access. Access rights define the type of access allowed to the data elements of a file for a particular password and/or user ID. Each file that requires a password also requires specified access rights.

The primary user interface to DBMS consists of the data definition language (DDL) and the data manipulation language (DML).

DDL is used to completely describe the DBMS data base and the associated data elements. The DDL logical data base definition source is compiled by the DDL compiler; the output is stored on disk with the associated data.

DML provides a means to manipulate data base information by supporting the reading and/or writing of the information. DBMS data can be accessed by embedding the appropriate DML syntax in a COBOL or Pascal application program. The application program is used to construct a call to DBMS that specifies the function to be performed on the data. The DBMS request is processed and the results are returned to the program.

3.3.2 DD

The Data Dictionary (DD) allows you to define all the data used in an organization and store these definitions in a central location. This centralization helps enforce data standards, clarifies the impact of changes to the data, and limits data redundancy.

The DD system consists of a dictionary file, a data librarian, utilities, and a data manager. The dictionary file contains the definitions of data in other files. (Note that the dictionary file does not contain the actual data from these other files.) The dictionary file controls and maintains key indexed, relative record, sequential, and data base files. You use the data librarian to enter information into the dictionary. The accuracy of all definitions in the dictionary file depends on the accuracy of the information in the data librarian. The utilities generate detail, summary, and cross-reference reports. This is an effective means of managing file definitions in the dictionary file and understanding relationships among the definitions. The data manager provides the DD interface to Query. This allows total control of and access to non-DBMS files for inquiry processing.

DD can be used in a stand-alone manner or in combination with Query and DBMS for full data file control. Query does not require the data manager to access data base files.

3.3.3 Query

Query is an English-like nonprocedural language with statements composed of several clauses. The clauses allow you to specify the content and format of each line as well as the complex conditions that a data base record or line must meet to be qualified for output. Calculations can be performed on fields, then the results can be output, tested, sorted, or used in further calculations. Output sorting, default column headings, and automatic paging are supported. String operations for test conditions (such as "contains") are also available.

The Query language makes it possible to specify a complex report in a few clauses. An application program to obtain the same report might require several hundred lines. The Query software package for DNOS provides you with a convenient and efficient means of retrieving data from a DBMS data base. You can gather and review data without writing a program.

The Query processor produces a report or data file by accepting and executing a Query language statement. You can build and execute a Query statement by either of the following two methods:

- Invoke the Query processor directly and gain access to a Query statement file (interactively or in batch mode) that was built through either the Query editor or the system text editor.

- Build a Query language statement by executing the Guided Query utility, which constructs the statement by prompting you with questions to determine the content and format of the report.

3.3.4 TIFORM

TIFORM is a software utility package for controlling the interactive interface to an application. TIFORM provides convenient control of complex screen formats for COBOL, FORTRAN, and Pascal applications. TIFORM includes an interactive screen drawing capability and a screen description language compiler. Through the use of these tools, TIFORM isolates the description of the screen format from the procedural code of the application. This allows applications to become independent of the terminal. TIFORM also features:

- All available VDT features (blink, dim, high/low intensity, no display)

- Character and field level editing

- Significant reduction in the amount of time required to develop interactive applications

3.3.5 Sort/Merge

DNOS supports a comprehensive Sort/Merge package. SCI commands provide access to Sort/Merge in batch or interactive mode. Sort/Merge supports the following features:

- Record selection
- Reformatting on input
- Summarizing on output

Ascending key order, descending key order, or an alternate collating sequence can be specified. Any number of keys can be specified as long as the total is less than 256 characters. The merge process supports up to five input files. The sort process allows the following:

- Key sort (tag-along)
- Summary sort (summary tag-along)
- Address only sort

Figure 3-1 shows an example of the Sort/Merge process with printouts of the results at each step.

3.3.6 TIPE

The TI Page Editor (TIPE) package provides word processing features for creating, filing, printing, and editing of memos, reports, and letters (including form letters). TIPE includes the primary features required to efficiently produce letters and documents without complex training or operation methods.

TIPE features provide you with the capability of performing a wide variety of page editing operations. The following descriptions introduce some of the features of the system.

Wordwrap (automatic return)
: When the right margin is reached, the cursor and any unfinished word automatically move to the next line.

Foreground or background printing
: Specify foreground printing so that you can halt the print operation before completion to insert more paper, change a printwheel, and so on. Specify background printing to allow other activity (such as creating or editing files) to be performed at the terminal during the printing process.

Shared/multiple printers
: Several operators can share the same printer; although only one print operation can occur at a time. Additional printers can be connected to TIPE, then the operator can specify which printer to use for a given print operation.

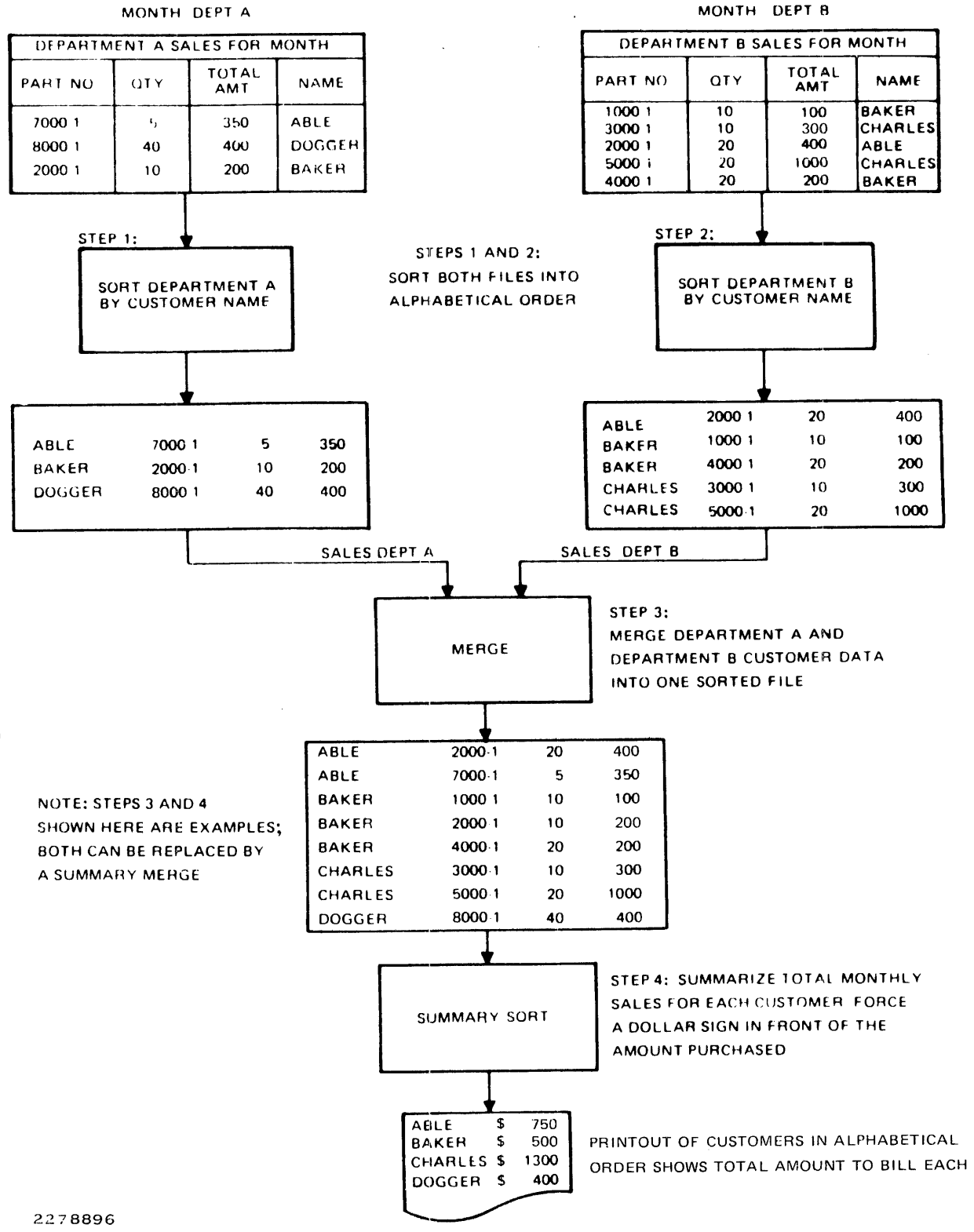


Figure 3-1. Sort/Merge Process

Form letter printing
: Allows a merge of variable data with the body of the letter to produce form letters automatically. Each letter has the appearance of an original, since it is tailored for each recipient.

Automatic pagination/repagination
: During document creation, TIPE paginates the text into the desired number of lines per page. Repagination, the restructuring of the document into pages of specified length, occurs automatically during printing.

Insert/delete functions
: Allows complete control over insertions or deletions as small as one character or as large as the entire document.

Move operations
: Permits movement of any specified block of text from one location to another. Can also store and recall text blocks to assist in document assembly and speed up the creation of letters, reports, memos, and so on.

TIPE features such as these make it simple to create, edit, and print bulky documents as well as shorter reports or form letters.

3.3.7 CPG

The COBOL Program Generator (CPG) is an easy-to-use programming system that generates error-free COBOL source code. CPG generates complete COBOL source programs that, when compiled, can be executed under DNOS or any system that supports RM/COBOL™. CPG reduces the time and effort normally associated with COBOL program development by performing routine coding. Menus, fill-in-the-blank screens, prompts, and explanation screens guide the CPG user.

CPG generates both batch and interactive programs. Batch programs can update files and prepare reports with no user interaction. Interactive programs accept data entry from formatted screens or modify data files by adding to, deleting from, or performing caculations on the current data fields in the file.

The CPG system provides two levels of usage: standard and advanced. The standard operation of CPG can be used by both experienced and inexperienced COBOL programmers to generate programs that perform routine tasks such as preparing reports and accepting data entry for file updating. The advanced operation of CPG can be used by experienced COBOL programmers to generate complex programs and define all program parameters.

3.4 COMMUNICATIONS SUPPORT

Several communications methods are available using DNOS and other Business System software packages and communications modules. Depending on the application, you can tailor a DNOS system to meet your needs. Communications software is described in the following paragraphs.

3.4.1 3780/2780 Emulator Communications Software

The 3780/2780 Emulator communications software package provides a means of remote job entry (RJE) communications with an IBM 360/370 host computer or any other computer equipped with a 3780/2780 Emulator. Communications consist of exchanging data files between master and slave stations over leased point-to-point or switched telephone lines.

RM/COBOL is a trademark of Ryan McFarland Corporation.

Using the 3780/2780 Emulator, DNOS systems can serve as satellite or central stations in a distributed processing network. Remote stations can also be operated in an unattended mode as a called station in a distributed network. An optional auto-call unit and a modem will provide manual or automatic dialing to remote stations. DNOS can also handle remote job or batch data entry for processing by the host CPU.

Texas Instruments 3780/2780 Emulator communications software emulates the operation of the IBM 3780 Data Communications Terminal, or the IBM 2780 Data Transmission Terminal. However, unlike the IBM 3780 and the IBM 2780, the source and destination of the transferred files are not restricted to the card reader and line printer. Any file, input device, or output device available to a user's system can be used for input or output.

3.4.2 3270 ICS

The 3270 Interactive Communications Software (ICS) package provides a means of connecting IBM mainframe computers to Business System computer systems. ICS allows interactive access to applications on the IBM computers that support the 3270 Information Display System. The software provides interactive access through the 911, 931, or 940 video display terminals. ICS also supports batch access through user-written COBOL, FORTRAN, Pascal, or 990 assembly language programs that control the Programmed Station Control (PSC) Emulator.

ICS operates over either leased or private communications lines. Communications require an appropriate controller along with an appropriate modem. When operating with ICS, a Business System computer can share a multipoint line with other IBM 3270 compatible terminals using binary synchronous communications software (BSC) protocol.

ICS maintains comprehensive statistics about data-link and application performance to aid in network troubleshooting.

3.4.3 Remote Terminal Subsystem

Remote Terminal Subsystem (RTS) software is a remote device control and communications software package. It enables a Texas Instruments Business System host computer to interact and communicate with 915 and Remote Terminal Controller (RTC) remote terminals. Each 915 remote terminal can connect with an optional 810 printer. Together, this terminal and printer are referred to as a *remote station*.

A host computer and its connected remote stations form a distributed network of terminals. Several remote stations can connect to the same communications line. In turn, several lines can connect to a host computer. RTS enables remote stations to enter data to the host, inquire about data being processed, and receive output from the host as if the stations were situated locally, although they can be located at great distances from the host computer.

3.5 NETWORK SOFTWARE

There are several network products available for DNOS. The Distributed Network Communications System (DNCS) provides a framework of common communication software in support of networking. Another network product, Distributed Network I/O (DNIO), allows users to transparently access remote systems and resources.

3.5.1 Distributed Network Communications System

DNCS consists of the Nucleus and add-on packages. Together, these packages allow users of Business System computers to participate in networks that conform to the Systems Network Architecture (SNA) and/or the International Standards Organization (ISO) open systems interconnection (OSI) model.

3.5.1.1 Nucleus. The DNCS Nucleus provides basic communications software that enables a DNOS system to function within a network. The DNCS Nucleus provides the following capabilities:

- A generation utility that configures DNCS according to add-on packages and user requirements
- The DNCS Command Interpreter (DNCS CI) that allows the user to monitor and control the status of resources in the configuration
- The service queue (SVQ), a work scheduling utility that permits time-delayed execution of requests, provides an alternate means of accessing an SNA host, and frees terminals for other foreground or background activity.

3.5.1.2 Remote File Transfer. The Remote File Transfer (RFT) system provides the means to transfer files between Business System computers operating under DNOS. A transfer can occur over public packet switched networks that conform to CCITT X.25 recommendations and/or leased lines. Each site included in the remote file transfer must have RFT installed.

Files can be transferred by SCI commands or by user application programs. The types of transferable files include sequential files, relative record files, program files, and image files.

RFT provides a service queue facility that allows the user to perform remote execution of SCI batch streams and batch jobs. The queue facility also allows remote execution of file printing and remote submission of noninteractive transactions to an IBM SNA host.

3.5.1.3 DNCS/SNA. Together, the DNCS Nucleus and DNCS/SNA Emulators allow the Business System computers to function as an intelligent cluster controller, physical unit type 2 (PU.T2) node, with the added functions of a distributed data base and local applications. Texas Instruments terminals and printers can also function as SNA display stations and printer stations in IBM's SNA network.

The Business System computer interfaces with the IBM 3705 communications controller in any of the following ways:

- Directly, using leased point-to-point, multipoint, or switched telecommunications lines using SDLC protocol
- Indirectly, using a packet switching network-X.25 (PSN-X.25)
- In virtual circuit (permanent or switched) configurations, over leased telecommunications lines using High Level Data Link Control balanced link access procedure frame level (HDLC-LAPB) protocol

In using the PSN-X.25 environment, DNCS emulates IBM's network interface adapter (NIA 5973-2) product functions. IBM's 3705 communications controller (local or remote) node runs the Advanced Communication Function/Network Control Program (ACF/NCP). The ACF/NCP software product used for the PSN-X.25 must have the X.25 NCP Packet Switching Interface (NPSI) program option.

DNCS can access the IBM S/370, 303X host systems having the Advanced Communication Function/Virtual Telecommunication Access Method (ACF/VTAM), and application subsystems such as:

- The Information Management System (IMS)
- The Customer Information Control System (CICS)
- The Time Sharing Option (TSO)

The DNCS/SNA Emulator package consists of the following individual emulators:

- The VDT2 Display Station Emulator enables a supported Texas Instruments VDT to emulate an IBM display station, a secondary logical unit type 2 (SLU.T2).
- The PTR3 Printer Station Emulator enables a supported Texas Instruments printer or disk file to emulate an IBM printer station using 3270 data stream commands and orders, a secondary logical unit type 3 (SLU.T3).
- The PTR1 Printer Station Emulator enables a supported Texas Instruments printer or disk file to emulate an IBM printer station using SNA character string controls, a secondary logical unit type 1 (SLU.T1).
- The KSR Station Emulator enables a supported Texas Instruments keyboard send/receive (KSR) terminal to serve as an SNA secondary unit type 1 (SLU.T1) input/output station using SNA character string controls.
- The programmed station control (PSC) routines and interface allow a user-written program to emulate an SNA station (SLU.T2 or SLU.T3) using the routines to pass data across the network.

3.5.2 Distributed Network I/O

Distributed Network I/O (DNIO) is a software package that allows a TI Business System computer using DNOS to access resources on other TI DNOS computers. With DNIO, users and programs can access remote system resources (files, devices, and IPC channels) as if these systems and resources were locally attached.

Users of the DNIO system do not need to understand complicated procedures, interfaces, or protocols. DNIO is easy to use because its access method is *transparent* to the calling program. That is, the program accesses remote systems and their resources using standard DNOS I/O operations, just as it does with local operations. This allows most standard utilities and applications to communicate between systems *without alteration*. This is a major advantage over most common communication packages.

The DNIO package allows users to easily:

- Share printers and mass storage devices among systems
- Access large files and data bases from other locations without needless duplication
- Log on to a remote system to use a communication link to another host system (such a host system may or may not be another TI computer)
- Dedicate computers to perform only certain tasks and share their operation with other users

DNIO provides support for Ethernet® local area networks. Ethernet support allows connection of a large number of Business Systems within one building.

Wide area network support is provided with the CCITT standard X.25 protocol. The X.25 communication support requires the DNCS Nucleus in addition to DNIO.

3.5.2.1 Network I/O. Network I/O operations allow access to resources on remote systems in a simple and natural manner. Terminal operators and application programmers do not need to understand complex network operations. They only need to know a remote system's site name to efficiently access a resource.

DNIO provides access to remote resources through the standard DNOS I/O system calls. Network resources are accessed by prefacing the resource access name with a site name. Network resources include:

- Files having a maximum record length of 4K bytes, including sequential, relative record, key indexed, and image or program file format
- Devices, including printers, terminals, disks, and tape drives
- IPC channels, including master/slave and symmetric

Ethernet is a registered trademark of the Xerox Corporation.

3.5.2.2 Network Log-on. With DNIO Network Log-on, a user can access network resources by logging on to DNOS SCI or any other application at a remote site. DNIO Network Log-on provides several features that can increase productivity, including the following:

- An application need only be located at one site.
- Slave systems can be established for CPU or I/O intensive applications.
- Communication gateways are provided through other TI-supported packages, such as DNOS 3270 ICS.
- Remote software maintenance is provided to look at crash dumps, install new software, and perform other maintenance operations.

Network Log-on provides other capabilities for increasing the efficiency and flexibility of data processing operations. At the same time, it provides a simple method of accessing remote applications without requiring any special network operation knowledge on the part of the terminal operator.

4

Program Management

4.1 JOBS AND TASKS

DNOS uses jobs and tasks to perform the functions of a multitasking operating system. A job is a collection of cooperating tasks (programs) that perform one or more functions. Jobs can be initiated either by a user or automatically by a program.

An example of a job is the output spooling job. Completion of a spooling job requires the cooperation of several programs. The device scheduler task dispatches print requests to other tasks that cause files to be printed. The spooler job is the "owner" of the printer receiving the spooled data. Other jobs, requiring the same or other system resources, can be executed concurrently with the spooler job. Program management is the controlling factor in this multitasking environment, and program management uses scheduling and priority factors to effectively control system operation.

Both interactive and batch jobs run under DNOS. An interactive job "owns" at least one interactive terminal. It is through the terminal(s) that tasks within a job communicate with the user. Alternatively, batch jobs do not communicate with a terminal. Both types of jobs carry the attributes of the initiating user. For example, every job is associated with the user ID of the user for whom the job is executing. Execution parameters, associated with past operations for that user, are available for the job. In this manner, an interactive or batch job can be regarded as a representative of the user within DNOS.

A job can be looked at as an environment for cooperating programs that share resources (e.g., files, devices, etc.), semaphores, and variables such as synonyms and logical names. Because programs are not necessarily associated with a physical terminal, this program isolation and job concept provides easy migration of programs between DNOS terminals and systems.

4.2 MULTIUSER SUPPORT CAPABILITY

DNOS allows the simultaneous execution of independent jobs from several terminals. Each user is given the impression that all system resources, including the CPU, memory, and any peripherals (e.g., disks), are dedicated to that user. In addition to job and task management, other DNOS capabilities help to reinforce this impression. For example, spooling allows a user program to write a file to the printer when the printer is busy. The file is stored on disk until a printer is available.

4.3 ACCOUNTING CAPABILITIES

The DNOS accounting subsystem provides a method of accumulating information related to the use of resources by various jobs in the system. This provides a method of acquiring information for those sites that require a utilization report. The accounting function is an option, selectable during system generation. DNOS keeps track of the amount of system resources used by jobs and tasks. For example, DNOS measures CPU time, memory allocation, and the number of pages printed. At certain times during the existence of the job, the accounting data is stored on disk. When a task terminates, the amount of CPU time and memory used for the task is written to the disk.

DNOS provides two accounting files for the storage of accounting information. Any subsequent processing of the accounting file(s) is accomplished by a user-written application program.

4.4 TASK SCHEDULING AND EXECUTION

DNOS allocates main memory and CPU execution time based on the installed priority and the characteristics of an executing program. The priority of an executing task (effectively the run-time priority) indicates the relative importance of that task to other active tasks in the system. DNOS ensures that the highest-priority ready task is in execution at any particular time. Ready tasks of equal priority share the CPU by time slicing. Time slicing is described in a subsequent paragraph.

4.4.1 Task Priorities

Scheduling tasks for execution in the DNOS system is accomplished by using priorities that are determined when a task is installed. Those priorities are:

- Real-time priority
- Time-sharing dynamic priority
- Time-sharing static priority

A real-time priority is used to assign a constant priority level to a task. Either a dynamic or a static priority can be assigned to a time-sharing task. The initial run-time priority of a time-sharing task is based on the installed priority of the task and the priority of the job in which the task is executing.

A task installed with a dynamic priority has a run-time priority level that can vary several levels during task execution. The degree of priority level change is based on whether the task is I/O-bound or CPU-bound. An I/O-bound task spends more execution time in I/O operations than in CPU operations. A CPU-bound task spends more time in CPU operations. I/O-bound tasks are given a higher priority than CPU-bound tasks.

During execution, a task installed with a static priority has a run-time priority level that can vary by fewer levels than a task installed with a dynamic priority. However, the priority level change is based on the same factors as those for a dynamic priority task (I/O-bound or CPU-bound).

4.4.2 System Clock and Time Slicing

The internal system clock of the computer uses the alternating current (ac) power supply to maintain the system time and date. The system clock interrupts DNOS 120 times per second (100 times per second, international). The interrupts are counted to increment the time and date and to serve as the time monitor that drives the DNOS task scheduler (to maintain the time slicing process).

Time slicing allows a task to execute for a given time period, then releases the CPU from that task to allow another task to execute. The time limit can be specified or can be disabled during the system generation process. Time slicing is accomplished through an interface with the clock interrupt processor. The clock interrupt routine maintains a count of the time that a task has been executing. When the clock reaches a predefined limit, the routine forces a return to the task scheduler rather than to the executing task. At this point the priority queue is used to determine the next task for execution.

4.5 SUPERVISOR CALLS (SVCs)

Programs request services from DNOS by issuing SVCs. Each SVC includes a block of information containing the detailed parameters associated with the services requested. Table 4-1 lists general-purpose application SVCs that are supported by DNOS. All SVCs available to user programs are described in detail in the *DNOS Supervisor Call (SVC) Reference Manual*.

Table 4-1. DNOS General-Purpose Supervisor Calls

Category	Functions
File and I/O Calls:	Perform I/O Operations Wait for I/O Wait for Multiple Initiate I/O Get Event Character Abort I/O I/O Utility Requests
Job Management Calls:	Create a Job Temporarily Halt a Job Resume a Job Modify Job Priority Map a Job Name Kill a Job
Program File Management Calls:	Install/Delete a Task Install/Delete a Procedure or Segment Install/Delete an Overlay Assign Space on a Program File
Program Control Calls:	Schedule a Task Execute a Task Delay Task Execution Resume Execution of a Delayed Task Change a Task Priority Level Unconditionally Suspend a Task Activate a Suspended Task Inhibit Task Suspension Force Abnormal Task Termination (Kill Task) Terminate a Task
Other Calls:	Convert ASCII/Binary Services Perform Job Accounting Services Expand/Contract Segments via Memory Control Service Task Synchronization Provide Status/System Information Manage Segments Encrypt/Decrypt Data

4.6 SYNCHRONIZATION

Synchronization controls the execution of one program in relation to the execution of other programs, then ensures that the execution proceeds in a dynamic, although structured, manner. DNOS provides synchronization on several functional levels. These synchronization tools include:

- Interprocess communication (IPC) — Read and write message operations for any two tasks in the system.
- Semaphores — The capability for two tasks, in one or more jobs, to exchange timing signals. A semaphore is implemented as an integer variable. The integer variable indicates the number of remaining timing signals; however, if no signals are present, the integer represents the number of tasks waiting for a timing signal. Semaphores are provided on job-local variables through SVCs that detail the completion code, operation code, semaphore number, and the initial or returned value.
- Events — The ability to initiate one or more SVC events, and wait for one or more of those to complete before continuing.

4.7 PROGRAM SWAPPING

If, during an attempt to load a task, there is insufficient memory available, the task loader enables a swapping routine. The swapping routine attempts to free up memory by temporarily writing program segments to the swap file.

There are three categories of task status that make a task eligible for swapping. These categories, listed from most eligible to least eligible, are as follows:

- Tasks suspended for more than a minimum amount of time. The longer each task has been suspended, the more eligible that task is for swapping.
- Tasks of lower priority than the task to be loaded. The lower the priority, the more eligible the task is for swapping.
- Tasks of the same priority as the task to be loaded, but these tasks have already executed for a minimum amount of time since the last loading. The longer each task has executed, the more eligible that task is for swapping.

5

DNOS Memory Organization

5.1 GENERAL INFORMATION

DNOS uses the Business System memory mapping option to dynamically allocate memory to disk-resident task (program) segments, as well as file blocking buffers. These allocated segments can be released from memory and swapped to disk as necessary. (Program swapping is described in Section 4 of this manual.)

DNOS memory organization serves to support both memory and instruction protection. Memory mapping can protect one program from being accessed by other programs. Even DNOS cannot be accessed by any nonprivileged program. Privileged instructions, such as instructions that manipulate the memory mapper, can be executed only by privileged tasks.

5.2 MEMORY ALLOCATION

The memory-resident portion of DNOS occupies lower main memory. The remaining memory space is available for user programs, initial memory-resident programs, and dynamically loaded disk-resident programs.

DNOS places tasks into memory wherever space is available. The memory mapping feature permits dynamic memory allocation; that is, a program can be dynamically segmented into separate areas of physical memory. Although an application program can be a task consisting of three segments, physically separated in memory, the mapping feature causes the program to recognize an environment in which:

- All three segments appear to be contiguous in memory with no gaps in addressability.
- The first segment appears to begin at memory address 0 (zero).
- The maximum addressable memory space for any one program is 64K bytes.
- Each program segment begins on a 32-byte boundary.

For example, a program can use one segment containing the data for the program and another segment for the procedure (the executable code). Each of these segments is mapped into the program's address space. Several programs can share the same procedure segment. The shared segment occupies one physical area of memory, but is mapped into the address space of each program that uses the segment.

5.3 TASK ADDRESS SPACE MANAGEMENT

DNOS provides overlay and segment mechanisms that allow a user to manage the logical address space of user programs. The user has the ability to change the group of current segments, and by doing so, to change the address space. Another technique to control the memory content of user programs is to load the content of segments from a backup storage device using the overlay mechanism.

5.3.1 Overlays

Overlays consist of phases of memory utilization that share common memory addresses with the restriction that only one overlay can occupy memory at one time. When using overlays, a disk or other mass storage medium must be available to store the overlay information. The Link Editor (described in Section 3 of this manual) or user-entered commands can enable the storage of overlays on program files.

5.3.2 Segmentation

DNOS programs can consist of various program sections with each section having certain attributes. The number of segments in a program is, in part, determined by the attributes assigned to the various sections of the program. Generally, if all sections of a program have the same attributes, there is only one segment. However, if the program sections have different attributes, the program requires several segments. If a division of the program can be made into sections with different attributes, then the user can benefit from using multiple segments (up to three segments per task can be in use at any one time).

Segment attributes, specified during task installation, include the following:

- Execute protect — Segment contains no executable code.
- Readable — Memory read accesses are allowed.
- Write protect — Segment cannot be modified when in memory.
- Share protect — Segment cannot be shared (concurrently) by more than one task.
- Reusable — Segment can be used consecutively without reloading.
- Replicatable — Multiple instances of a segment can exist concurrently.
- Copyable — Segment in memory can be copied to effect replication.
- Updatable — Segment can be written to its permanent file position after being modified in memory.
- System — Segment can be used by a system task only.
- Memory resident — Segment permanently resides in memory.

Disk-based program segments are installed as memory images on program files. A maximum of 256 program segments are allowed in a program file. (The system also allows up to 256 task and 256 overlay segments in each program file.)

One illustration of program segmentation is a program that consists of eight logical sections of code. Figure 5-1 shows the segment structure of this program. This program includes a task segment, several segments of equal length (S1, S2, and S3 in Figure 5-1), and several segments of various lengths (S4, S5, S6, and S7). At any point during execution, the program can use three of the following segments:

- The task segment (always present in memory)
- One of the segments: S1, S2, or S3
- One of the segments: S4, S5, S6, or S7

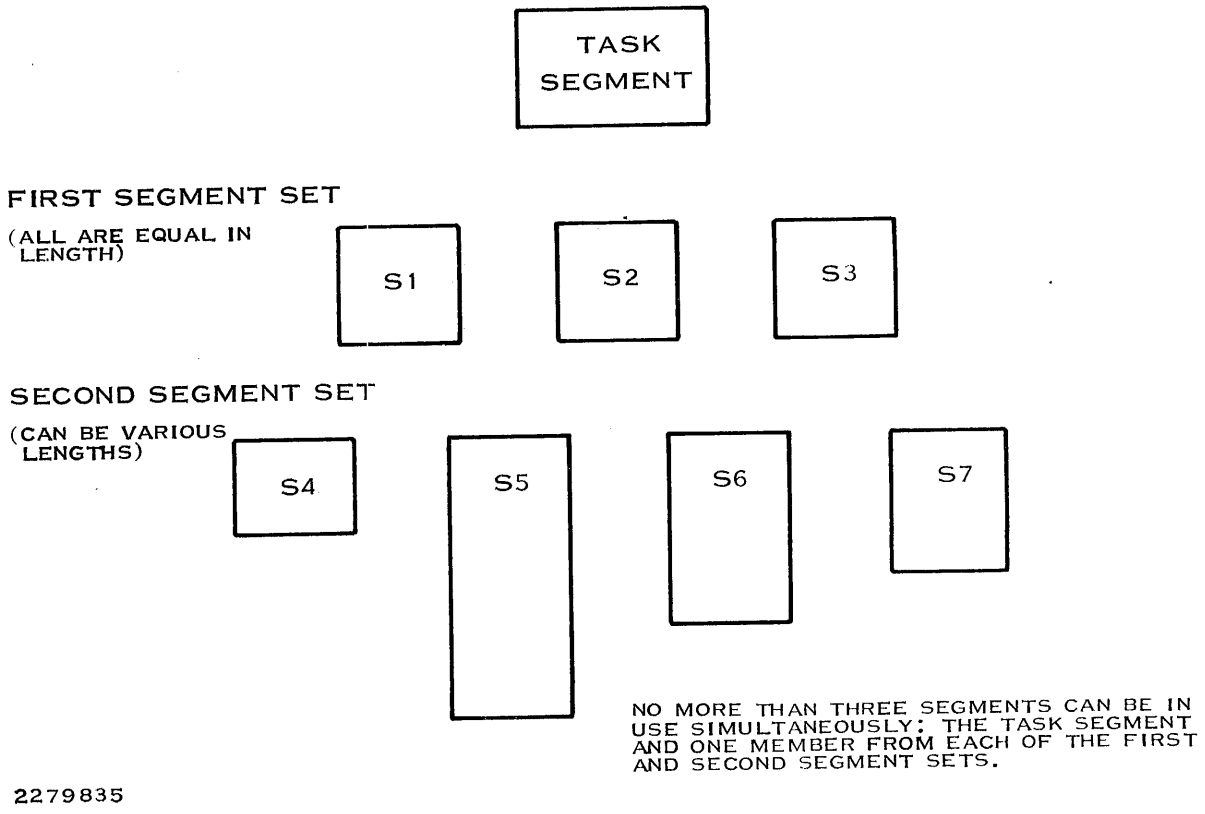


Figure 5-1. Program Segmentation

DNOS also supports memory-based segments created in memory and used during program execution. Users can also map segments that correspond to physical records of a relative record file into their programs.

6

I/O Resource Management

6.1 GENERAL INFORMATION

I/O resource management involves the orderly allocation of logical I/O resources to tasks (programs). Logical I/O resources include disk space, devices, files, and interprocess communication (IPC) channels. The DNOS system uses a method called *incremental allocation* to manage I/O resources. This type of allocation allows a program to dynamically request resources during execution.

Whenever a resource is requested, but is not available, the program or the user is notified immediately. The request for resources is not queued, and the program is not suspended. This allows the flexibility to either abort or retry the resource access function.

I/O resources are allocated to programs according to access privileges that the program requests during an open operation. If the requested privilege can coexist with previously granted requests, the open operation completes without error. Thereafter, the program is guaranteed the type of access requested (exclusive, exclusive write, shared, or read only access).

6.2 I/O METHODS

DNOS supports input and output operations to various types of devices and to several types of files. It also supports communication between programs. Devices, files, and communication channels are referred to as I/O resources. There are two methods of I/O to resources: *resource-specific* and *resource-independent*. Resource-specific I/O depends on a particular type of device or file. Resource-independent I/O is more flexible and allows a user to simply specify I/O for any of several types of devices.

Both resource-specific and resource-independent operations allow a program to interact with predefined devices, files, and channels. The interaction occurs through the use of logical unit numbers (LUNOs) that are assigned to the device, file, or channel.

6.2.1 Resource-Specific I/O

Resource-specific I/O allows the programming of specific capabilities of devices, channels, and files. Resource-specific I/O is supported for the following:

- Direct disk I/O
- Extended VDT I/O
- Create/delete files

- File specific I/O utility operations
- Random-access operations to key indexed and relative record files
- Interprocess communication (IPC) operations

6.2.2 Resource-Independent I/O

Resource-independent I/O requires use of a sequentially-oriented device, file, or channel. A program using this kind of I/O call can read and write data records in sequential order, independent of the type of device or file used. Examples of such types of operations include Read, Write, Forward Space, and Write EOF. Extensions to resource-independent I/O allow a program to take advantage of device capabilities. Such programs work only with a specified type of a device or file; that is, they are using resource-specific I/O. All devices, files (including key indexed files), and channels support a resource-independent type of access.

6.3 LOGICAL NAMES

A logical name represents the pathname of an I/O resource, such as an IPC channel or a concatenation of files. The user defines a convenient logical name with one to eight characters and can assign a LUNO to the defined name. Individual files can then be accessed by entering the complete pathname or a logical name, but only a logical name will access a concatenation of files as if they were a single file.

A logical name also provides a mechanism for specifying parameters associated with the resource. For example, a logical name that represents a spooler device has parameters such as form, type, and number of copies. The name can be further specified as global (available to all users of a system) or job-local (available only to the user who defined it).

The System Command Interpreter (SCI) initializes a user's set of logical names from a disk-based file associated with each user ID. Generally, logical names are created at the SCI level, but programs can create them directly.

6.4 SPOOLING

The spooling of data occurs during job execution as output is generated by one or more tasks. Spooling is the process of receiving data destined for a particular device (or class of devices) and writing that data to a temporary file (or files). The spooler subsystem schedules the printing of files among available printing devices.

The user has the option, by using SCI commands, to specify the following:

- Form — The printing form on an output device
- Format — ANSI or standard printing format
- Device Class — A group of related devices, accessed by a single name
- Number of Copies — Number of copies of a file or files to be printed

7

Interprocess Communication (IPC)

7.1 GENERAL INFORMATION

Interprocess communication (IPC) allows two or more tasks to exchange information. Tasks exchange messages by reading and writing over IPC channels that are created by the system and exist independently of the tasks using them. (A channel is an IPC path between two tasks.) During each message exchange, one of the tasks is designated as the owner of the channel—that task controls the channel. The requester task has less flexibility and fewer privileges. IPC channels may exist between tasks in the same computer or between tasks in different computers. Computer boundaries are transparent to the communicating tasks.

7.2 USES OF IPC

Interprocess communication is used for four primary reasons:

- Synchronizing jobs and tasks
- Building queue servers
- Intermediate processing of data
- Sending and receiving messages

Task synchronization differs from other uses of IPC since, usually, only the existence of a message is important, not its content. Tasks may require synchronization in order to share resources or in order to cause external interactions to occur in a particular order when programs are executing in parallel.

IPC provides the mechanism for building system services or user applications to process queues of requests. Associated with each provided service is a channel to receive requests and a task that accepts and handles those requests.

Some queue servers can provide intermediate processing of data between a requesting task and a final destination. These tasks receive requests from queues, interpret or modify the information, and pass the changed data to another task or device.

Another use of IPC is to send and receive messages between tasks. The tasks may use the messages in many ways, depending on their function.

8

File Organization

8.1 DISK FILE STRUCTURES

DNOS supports three major file types for user data: sequential, relative record, and key indexed. This section describes these file types. The multilevel directory structure, disk allocation, specific file features, and file security are also described.

8.1.1 Sequential Files

Sequential files are useful for recording variable length data records in the order in which they are received. Similarly, data must be read back in the same order as found in the file. To reach a given record, all preceding records must be processed. A pointer to the current file position is kept for each active assignment to the file. As each record is read or written, the pointer is advanced.

Sequential files have the property that no valid data can exist beyond the most recently written record, although records can be written to the file by appending them to the existing data. Because of this, a sequential file can contain multiple subfiles in a serial format. Since each subfile is separated by an end-of-file mark, a sequential file can also contain multiple end-of-file marks within its boundaries.

Several programs can read a sequential file concurrently at different positions in the file. However, only one program at a time can write to a sequential file. The current position is retained while the file is logically assigned, even though the file is closed and reopened.

8.1.2 Relative Record Files

Relative record files are those files in which all logical records are of a fixed record length, and each record can be randomly accessed by its unique record number. This type of file is especially useful for rapid access to data that is naturally ordered by record number. Sequential access is also permitted. One end-of-file record is maintained wherever it was specified by the last program to access the file.

8.1.3 Key Indexed Files

The most sophisticated file type supported by the system is the key indexed file. In key indexed files, variable length records are accessed by providing the operating system with any one of up to 14 keys that identify the data. A key is a string of up to 100 characters. For example, an employee file can be constructed so that the data record for any given employee is accessed by the employee's name, employee number, social security number or any other designated key. Keys can be declared to overlay one another within the record. Although keys can be structured anywhere within the record, they must appear in the same relative position in all records in the file. One of the 14 possible keys must be selected as the primary key. All other keys are known as secondary keys. The primary key must be present in all records, but secondary keys can be optionally present in any given record within the file.

8.1.3.1 Key Values. Key values for both primary and secondary keys are kept in indexes within the file. These indexes are maintained in a data structure that allows rapid random access while still allowing sequential access in the sorted order of any selected key. In a manner similar to that for sequential files, a pointer to the current position within the file is maintained for each key by DNOS. When updating any given record, the user of a key indexed file can add, delete, or change a key value.

8.1.3.2 File Stability. When a write operation is to be performed on records in a key indexed file, DNOS makes a copy of physical records necessary to perform the operation. If a failure occurs during the I/O operation, the prelogged records can replace the master copy. Thus, DNOS can guarantee the integrity of the file.

8.1.4 Concatenated Files and Multifile Sets

DNOS file management and I/O utilities support logical files that are created by concatenating several physical files. Concatenated files consist of multiple files that have been logically appended and are known collectively by a single logical name. The physical files can exist on one volume or on several volumes. Only sequential and relative record files can be concatenated. The file concatenation continues only during the existence of the job, and only the last component is expandable.

A multifile set is a group of key indexed files whose pathnames are the values of a single logical name. Like concatenated files, components of multifile sets can reside on one volume or on several volumes.

You can assign a logical name to two or more key indexed files and use the logical name as if it were the pathname of a key indexed file. In this case, the files are not logically concatenated but are physically associated with one another in a multifile set. The file structure on disk is altered in such a way that you can no longer separately access the individual files by key indexed file operations. You can access individual files by operations that examine a physical record of a file or an absolute disk address.

8.2 MULTILEVEL DIRECTORY STRUCTURE

Under DNOS, each disk volume contains system overhead and space reserved for files. DNOS utilizes one disk volume from which the operating system is loaded. That disk, designated as the system disk, is also used by DNOS to perform internal disk-based functions. Figure 8-1 illustrates the multilevel directory structure of a disk volume.

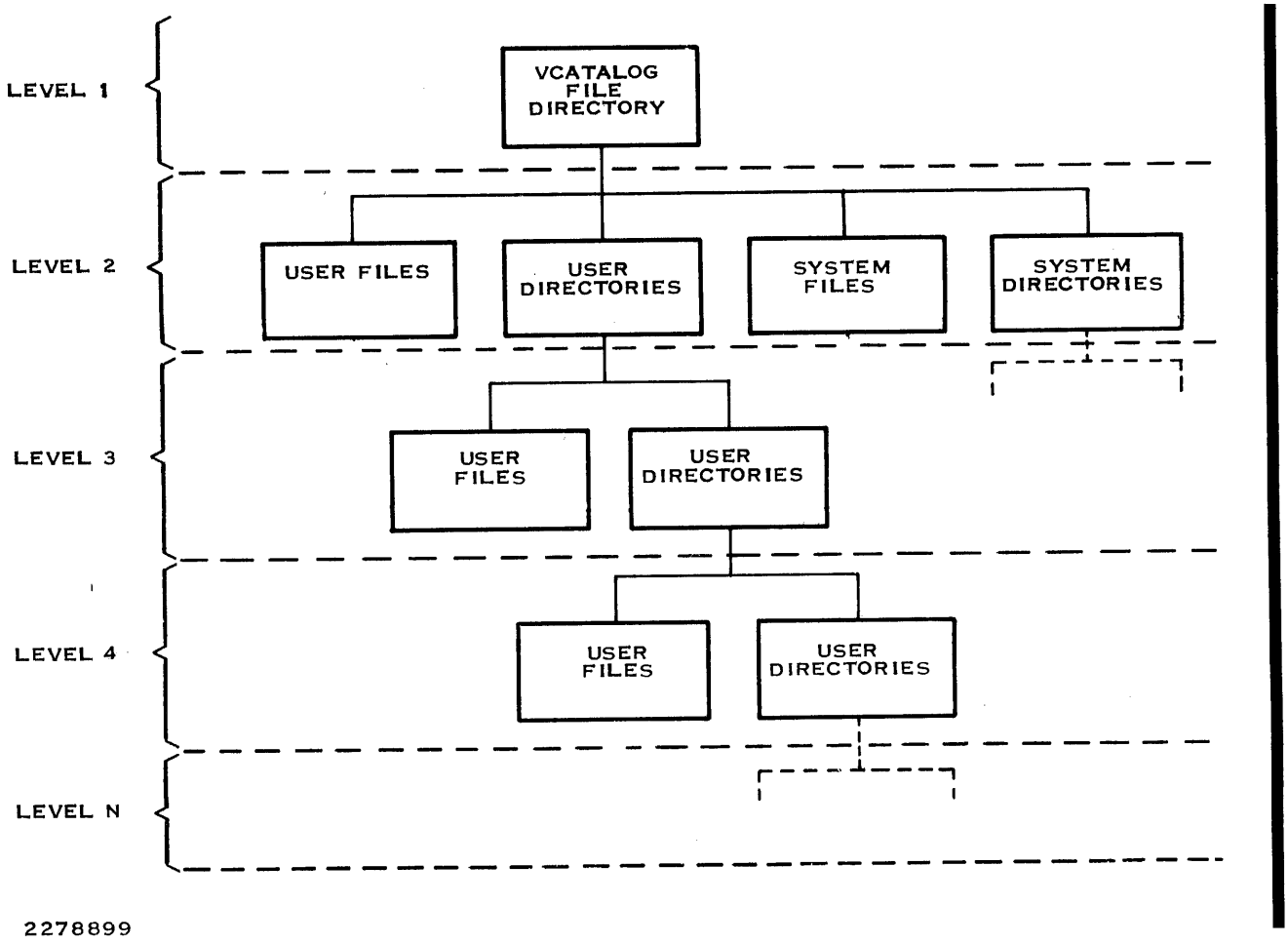


Figure 8-1. Multilevel Directory Structure

8.3 DISK ALLOCATION

Track 0 and part of track 1 are always allocated to system overhead. The innermost cylinder is always allocated for use by system diagnostics. The remainder of a disk volume is dynamically allocated to directories and files as they are created, expanded, and deleted. The primary directory is called .VCATALOG; it corresponds to the name of the volume. Within this directory, other directories and files are cataloged, many of which are used for system functions. These system disk files store the DNOS kernel, utilities (in memory image code), swapped program images, and an image of the contents of system memory (as a precaution against a possible system crash). The DNOS disk manager task automatically allocates available space to files when they are created and as they expand. When files are deleted, additional disk space is automatically available for other files.

8.4 FILE FEATURES

DNOS supports a variety of file features that add to the flexibility and expand the overall usefulness of the system. These file features include the following:

- File use from high-level languages
- Delete and write protection
- File access privileges
- Record locking
- Temporary files
- Blocked files
- Deferred or immediate write options
- Blank compression and adjustment
- Expandable files

8.4.1 File/Language Relationship

The various file features and file types are all available to the assembly language programmer. High-level languages can access many file features using built-in language functions. Assembly language programs can be written and called from high-level language programs, thus providing indirect access to features not directly supported by high-level language syntax.

8.4.2 Delete and Write Protection

Although each created file can be protected from accidental deletion (from the disk volume), in some cases it is desirable to further protect the file. DNOS permits write protection for files, meaning that the data in such a file can only be read. Write-protected files are automatically delete protected.

8.4.3 File Access Privileges

Under DNOS, a program can request specific access privileges for any use of a disk file. Use is defined as the entire file transaction from the open operation through the close operation. These privileges include exclusive access, exclusive write access, shared access, and read-only access.

8.4.4 Record Locking

Record locking is the process of locking individual records within a file. This feature allows a program exclusive access to a locked record until that record is unlocked. For example, record locking could be used to lock an inventory record while updating the quantity that is in stock. The locking prevents other programs from changing the same quantity before the first update is complete.

8.4.5 Temporary Files

DNOS allows the creation and use of temporary files. These files are subject to automatic deletion by the operating system. This feature allows trial preparation of a file; if the file is satisfactory, it can be renamed and designated as a permanent file. If the file is not satisfactory, and not renamed, it is deleted by DNOS when the LUNO is released or when the task terminates. A job temporary file can be created for use by one or more tasks, then deleted when the job terminates.

8.4.6 Blocked Files

Multiple logical records can be automatically combined by DNOS into larger physical records. These larger records conserve disk space and reduce the number of physical transfers of data between memory and the disk.

8.4.7 Deferred or Immediate Write

The physical transfer to disk of blocks of logical records is normally deferred by DNOS until the memory space held by the blocking buffer is required for some other purpose. This reduces the number of physical disk accesses. The system automatically updates all records on disk when the file is closed. In some cases it is advantageous to write data immediately upon request (for example, to maintain data integrity within highly sensitive files). The immediate write option supports this capability.

8.4.8 Blank Compression and Adjustment

Blank compression is the process of removing consecutive blank characters from each record in file types that support variable length records. The strings of consecutive blanks in a record are coded in a shortened form. Blank adjustment is the removal of trailing blanks on a write operation and the replacement of the blanks on a subsequent read operation. Blank adjustment is available for devices as well as files.

8.4.9 Expandable Files

DNOS permits the file size to be declared when the file is created. Unless otherwise specified, if the file exceeds the initial allocation, additional space is automatically allocated. This allows files to grow beyond the initial boundary. Secondary allocations to the file become increasingly larger as the file continues to expand.

8.5 FILE SECURITY

DNOS provides file security as an operating system option during system generation. The security system is based on the relation defined between groups of users and access rights to files. The users of the system are divided into access groups, according to the ways they use the system. A given user can be a member of one or more access groups. Each file available to the system can have access rights that restrict its use by various access groups. The access rights available for a file are:

- Read — read or show the file
- Write — extend or modify the file
- Execute — execute a task in the program file

- Delete — delete the file
- Control — change the access rights to a file

The creator of a file can specify which access groups have which access rights to that file.

There are two special access groups in the file security system. The group named PUBLIC is composed of all users of the system. Any user can access a file with the rights given to the PUBLIC access group. The SYSMGR access group is privileged; it can access any file in the system. SYSMGR is intended for the system security manager's use in handling problem situations and removing files that are no longer used.

9

Messages and Exception Handling

9.1 MESSAGE STRUCTURE

All DNOS error messages include the source of the error, the category of the subsystem reporting the error or condition, and a short description of the problem. The system includes error text files of messages that can be changed or translated into languages other than English by text editing the file. To aid in the error determination, utilities pass information to SCI that, when combined with other information in the error text file, creates a complete message.

Messages are formatted in the following manner:

Source Category-Message ID Message Text .....

The message source (user, system, hardware) is represented by one, two, or three characters. The subsystem that generated the message is identified by the category of the message. It is an acronym that consists of one to eight characters. The message ID contains 1 to 14 characters identifying the message (usually a four-digit decimal number). The message text is a short description of a special situation or error condition; it can be up to five lines in length. In addition to the short description, expanded explanations and recommendations for user action are available. When the short explanation appears, the user can enter a question mark to be given further details.

9.2 ONLINE DIAGNOSTICS

Online diagnostics execute without affecting normal system operations. They execute as application level tasks in a nonprivileged mode, thus assuring that the online diagnostics or tasks do not damage existing operations or data bases. You can specify the execution of only one diagnostic or the simultaneous execution of two or more diagnostics.

9.3 POWER FAILURE HANDLING

An optional power failure recovery feature is available for DNOS. When this feature is included and selected (as a system generation option), and a power failure occurs, the backup power supply (battery) is enabled to refresh and rebuild the VDT screens until full power is restored. Any disk I/O that is in progress at the time of the power failure is reissued when power is restored. Input/output operations that are in progress, to devices other than the disk or a VDT, are aborted with a suitable indication to the requesting task(s). The task can retry the operation when power is restored. The system keeps track of the last instruction that was executed, and execution can be resumed at that point.

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)!*.

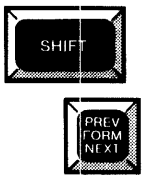
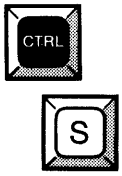
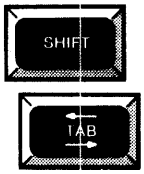
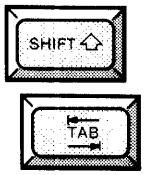
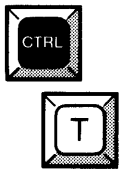
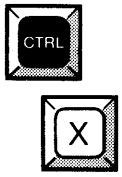
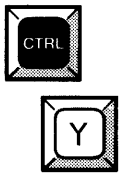
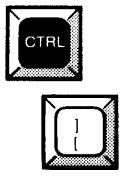
Table A-2 shows some frequently used multiple key sequences.

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					

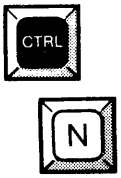
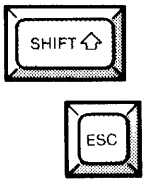
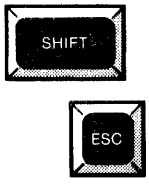
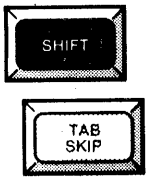
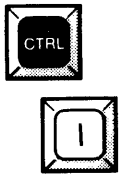
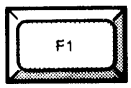
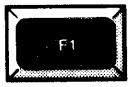
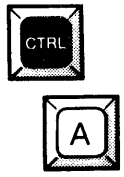
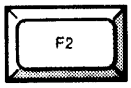
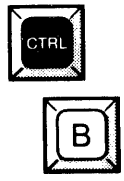
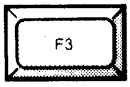
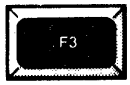
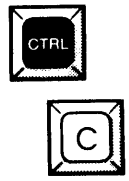
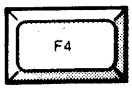
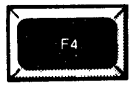
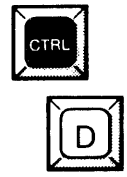
Notes:

[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.

2284734 (2/14)

Table A-1. Generic Keycap Names (Continued)

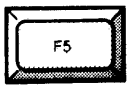
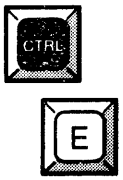
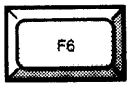
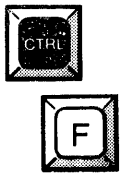
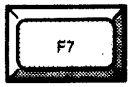
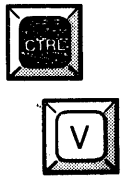
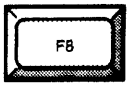
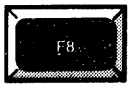
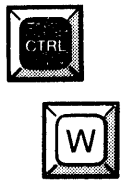
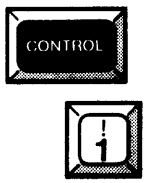
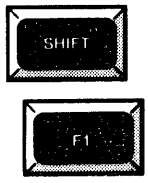
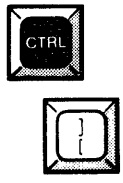
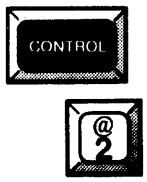
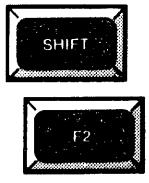
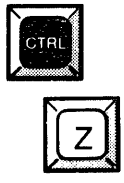
Generic Name	911 VDT	940 EVT	931 VDT	Business System Terminal	820[1] KSR
Erase Input					
Exit					
Forward Tab					
F1					
F2					
F3					
F4					

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.

2284734 (4/14)

Table A-1. Generic Keycap Names (Continued)

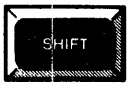
Generic Name	911 VDT	940 EVT	931 VDT	Business System Terminal	820[1] KSR
F11	CONTROL + $ 4	F11	F11	SHIFT + F3	CTRL + ! \
F12	CONTROL + % 5	F12	F12	SHIFT + F4	CTRL + { }
F13	CONTROL + ^ 6	SHIFT + F1	SHIFT ⇧ + F1	SHIFT + F5	CTRL + ± =
F14	CONTROL + & 7	SHIFT + F2	SHIFT ⇧ + F2	SHIFT + F6	CTRL + — _
Home	HOME	HOME	HOME	HOME	CTRL + L
Initialize Input		SHIFT + LINE INS CHAR			CTRL + O

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 (5/14)

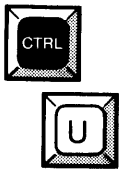
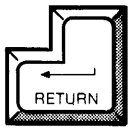
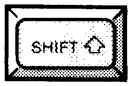
Table A-1. Generic Keycap Names (Continued)

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					  or 
Previous Character	 or 				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
Uppercase 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.

2284734 (7/14)

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)

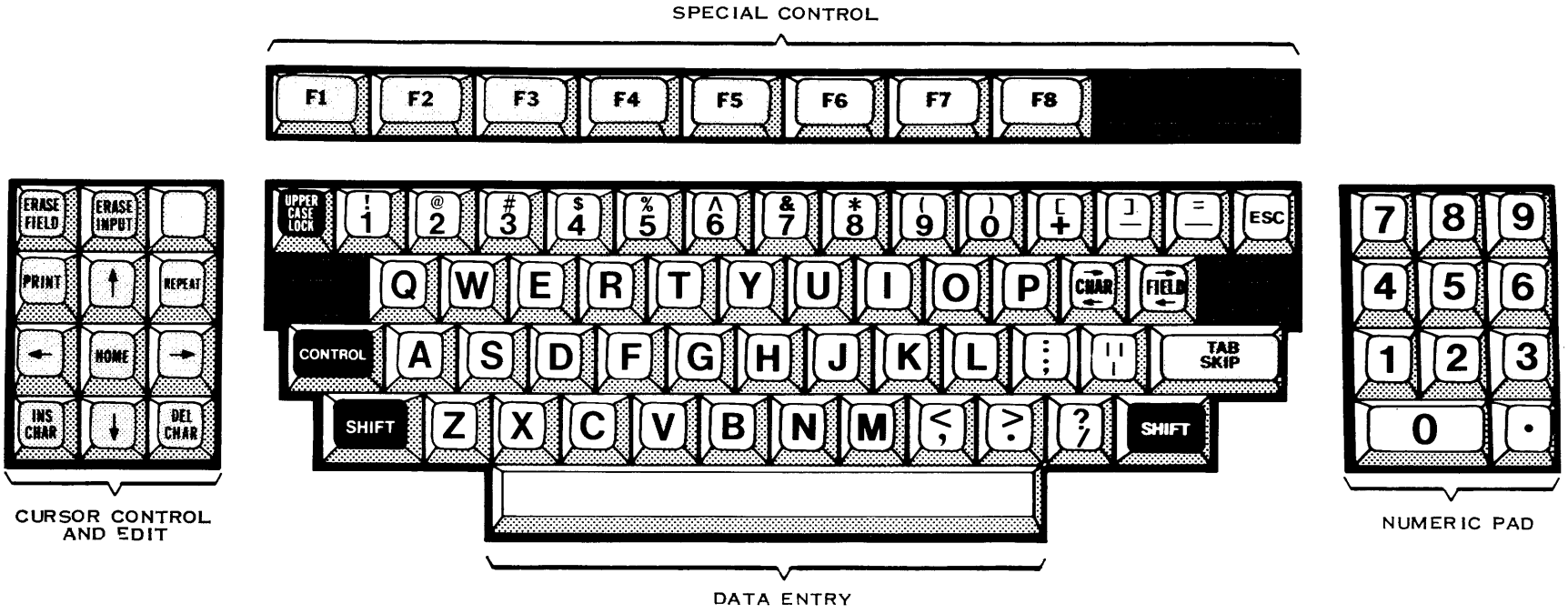


Figure A-1. 911 VDT Standard Keyboard Layout

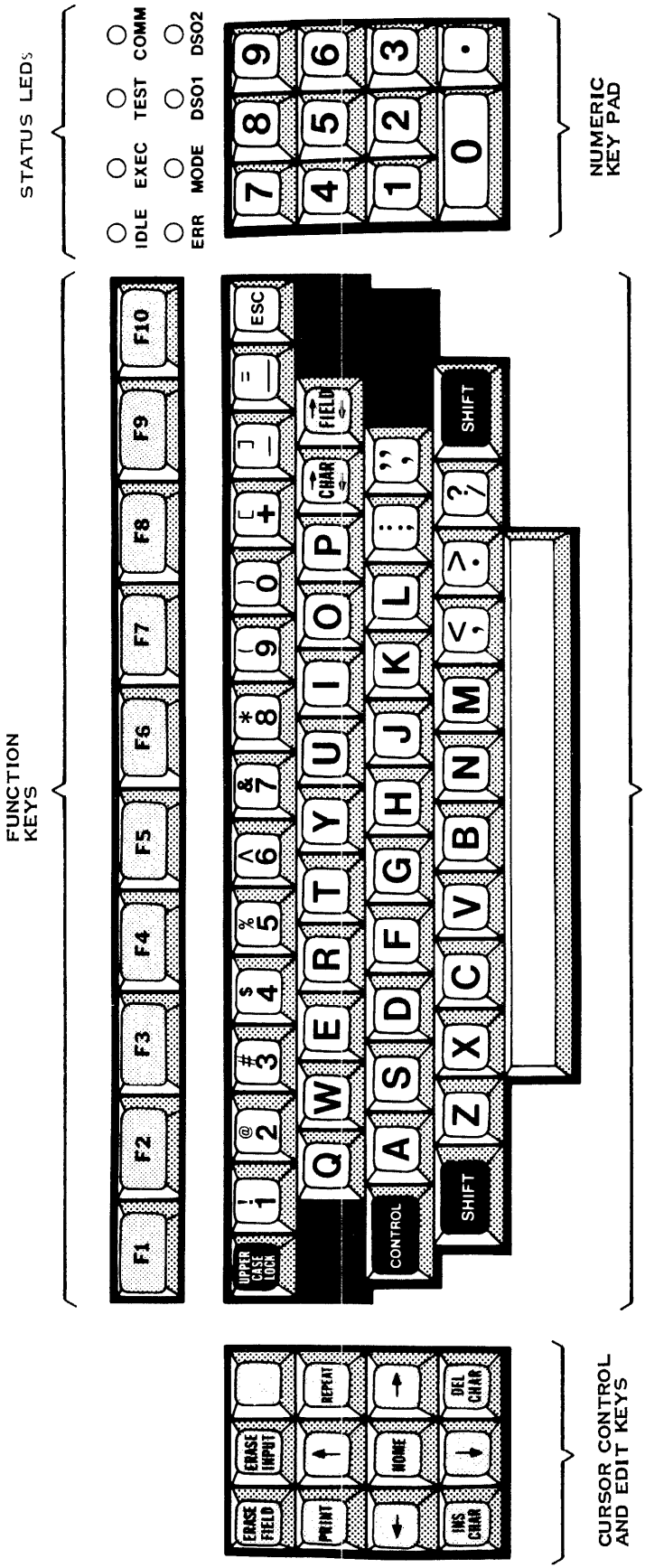
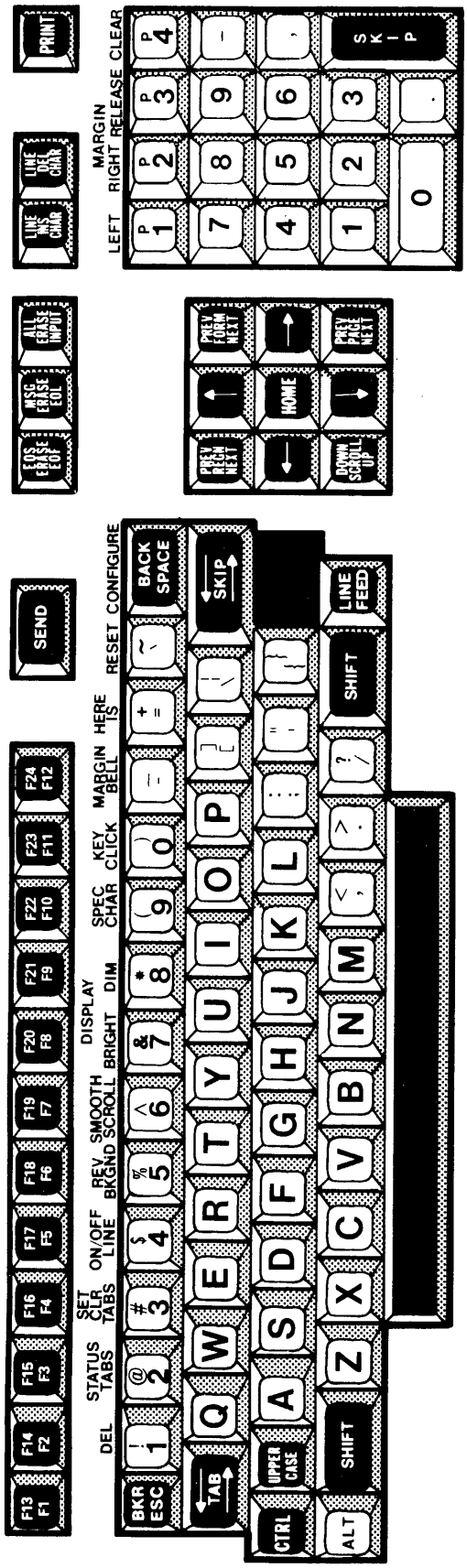


Figure A-2. 915 VDT Standard Keyboard Layout



2284734 (11/14)

Figure A-3. 940 EVT Standard Keyboard Layout

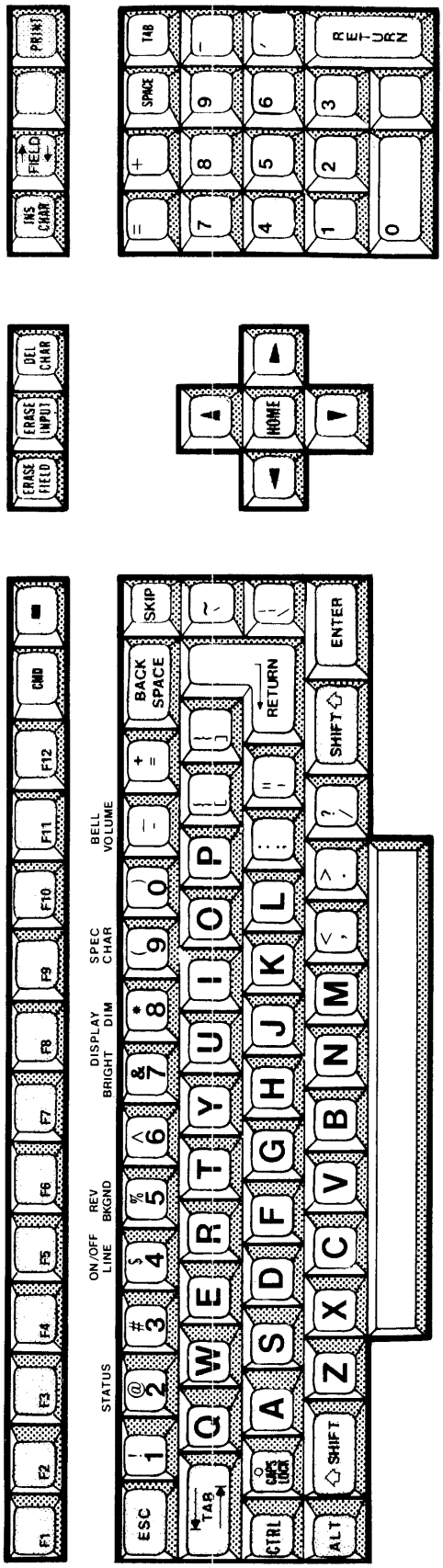


Figure A-4. 931 VDT Standard Keyboard Layout

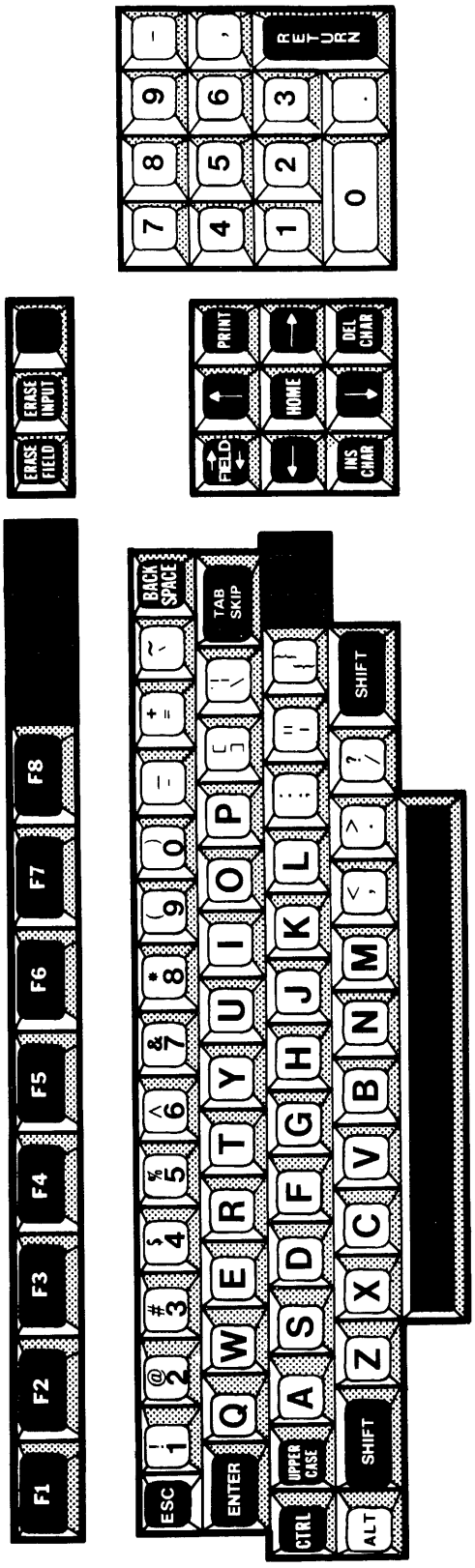
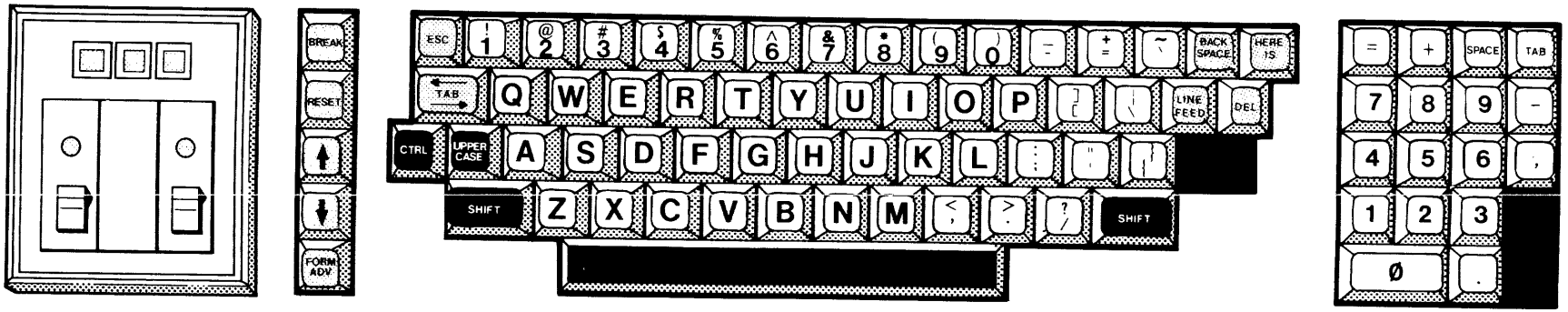


Figure A-5. Business System Terminal Standard Keyboard Layout



2284734 (14/14)

Figure A-6. 820 KSR Standard Keyboard Layout

Glossary

The terms defined in this glossary are used to describe the structure and operation of DNOS. All members of the DNOS family of documents contain some of these terms. The terms are arranged alphabetically except for symbols, which are located at the end.

Access Group — A group of users of a DNOS system, usually with a common work assignment or method of using the system. The access group name is used to define access rights to a file for the users in the group.

Access Name — A name used to access an I/O resource. An access name can be a device or volume name, a file or channel pathname, or a logical name assigned to a device, volume, file, or channel.

Access Privileges — The levels of security that ensure the desired isolation for a given resource. A task can have one of the following types of access privileges for a resource:

- Shared access (more than one task can read and write)
- Read-only access (this task will only read)
- Exclusive write access (other tasks can read, but only this task can read and write)
- Exclusive all access (only this task can read or write)

Access Rights — Certain rights allowed to the user of a file as determined by the file security system. The rights available to a user are read, write, execute, delete, and control.

Account ID — A 16-byte character string, defined by a user, that is used by the accounting subsystem to identify the account to be charged for a work session with the computer.

Active State — The state in which a task is ready to execute.

Active Task — See Ready Task.

Address — A group of characters that specify the location of an area of memory. The operating system uses addresses to correctly access data and instructions. For example, an address can be 4 hexadecimal digits that represent a word of memory.

Address Space — The memory that can be accessed by an addressing scheme. The memory that a task can address is different from the memory that the Business System computer can physically address. This difference is resolved by the mapping hardware and the operating system.

ADR — See Alias Descriptor Record.

ADU — See Allocatable Disk Unit.

Alias — An alternate for a pathname component. The alias can be used in place of the pathname component.

Alias Descriptor Record (ADR) — A disk directory record that carries an alias name and a pointer to the structure for which it is an alias.

Allocatable Disk Unit (ADU) — The smallest disk space that can be allocated for file creation or expansion. This space varies from 1 to 12 sectors, depending on the type of disk in use.

Application Program File — A program file on which users install programs that are to be memory resident. It can also include programs that are disk resident, if the user wants to include them.

ASR — See Automatic Send/Receive.

Autocreate — The process performed by the operating system when a bit is set in the supervisor call block for an Assign LUNO; it specifies that the operating system is to create the file to which the LUNO is being assigned if that file does not already exist.

Automatic Send/Receive (ASR) — A line-oriented terminal device with attached cassette tape drives or paper tape drives.

Background Mode — A type of system use in which, once the command is initiated, the task executes without interaction with the terminal. The user can do other processing in the foreground mode during the background process. The initiated background task receives as its environment a snapshot of the current System Command Interpreter (SCI) environment, and it proceeds to run simultaneously with SCI.

Base System — The initial system image sent to the customer. The base system is a minimum system capable of performing system generation and supporting a VDT, a hard-copy terminal, a tape drive, a line printer, and multiple disk drives.

Batch Job (Batch Mode) — A job that runs independently of a terminal. The user initiates the job through the System Command Interpreter (SCI). SCI will bid a sequence of tasks and pass environments from one task to the next.

Beet — A 32-byte block of memory starting at a memory address that is exactly divisible by 32.

Bid — To place a task in the active state (ready for execution), i.e., to request that the system call the task into memory (if necessary) and enter that task on the active list at the appropriate priority.

Blank Adjusted — An attribute of records in a file, denoting that trailing blanks are added when the records are read and removed when the records are written to a file. This attribute can be specified during the I/O operation.

Blank Compressed — A characteristic of a sequential file indicating that each occurrence of a string of consecutive blanks is replaced by a code that represents the number of blanks. Thus, consecutive blanks are not actually stored. This characteristic, when desired, is specified at file creation time. It is used to reduce the disk space required for a file.

Blank Suppressed — See Blank Compressed.

Blocked Disk Transfer — The movement of data through a system-supplied buffer, as an intermediary between the disk and the user buffer. This employs packing of one or more logical user records into a single physical record.

Blocked File — A file in which a physical record includes more than one logical record.

Bootstrap — A technique for loading the first few instructions of a system into memory, and then using these instructions to load the rest of the system. For DNOS, this bootstrap sequence begins in a ROM loader after the initial program load (IPL) sequence is keyed into the front panel of the Business System computer.

Boundary Error — An error in which a task tries to access memory outside its designated address space. The error causes task termination.

BRB — See Buffered Request Block.

Break Key Sequence — A sequence of key strokes used to abort a task in an interactive job. The system selects the task to be terminated from the tasks of the current job, according to the current environment. If only SCI is active, it is terminated, the job at the terminal ends, and the terminal is again available for use. If tasks other than SCI are active, the system will terminate foreground tasks first, then background tasks, then SCI. Unless the task is being debugged, a task with end action will take end action when terminated by the Break key sequence. See End Action.

Breakpoint — A place in a routine at which execution stops. A breakpoint is specified to the operating system as XOP 15,15. Except when controlled by the Debugger, breakpoints do not automatically return control to the controlling task.

Buffer — Storage used to compensate for differences in the rate of data flow, time of occurrence of events, packing of data, or transmitting data from one location to another.

Buffered Request Block (BRB) — A data structure used by SVC processors that contains a copy of the user's SVC block and several words of overhead information.

Byte — A group of binary digits operated on as a unit. The Business System computer uses bytes consisting of eight binary digits.

Cache Memory — A smaller portion of memory (2K bytes) that operates much faster than primary memory (64K bytes). The controller stores frequently used data in the cache memory for faster access than allowed by primary memory.

Call Block — See Supervisor Call Block.

CCB — See Channel Control Block.

CDR — See Channel Descriptor Record.

Central Processing Unit (CPU) — The arithmetic and logic unit of a computer.

Channel, IPC — A logical path used for interprocess communication between two tasks. One of the tasks is designated as the channel owner and can perform special operations, depending on the channel type. See Symmetric Channel and Master/Slave Channel.

Channel Control Block (CCB) — An in-memory data structure that defines a channel; it shows the current state of a channel that is in use.

Channel Descriptor Record (CDR) — A disk-resident data structure that defines the characteristics of a channel and the location of the channel owner task; used to create the channel control block and bid the owner task for the channel.

Channel Name — A pathname for a channel.

Collating Sequence — A method of specifying an order for a collection of character strings. For example, one type of specified order might be special characters followed by digits, followed by alphabetic characters in English alphabetic order. The collating sequence for DNOS is determined by the system's country code.

Command — A directive to perform an action. The System Command Interpreter (SCI) is provided to interpret commands and initiate utilities as needed to perform the desired functions.

Command Mode — A processor mode in which the processor interprets input as commands rather than as data. The Text Editor and the system generation utility have a command mode.

Command Procedure — A program in the SCI language that implements a command.

Command Processor — A task executed by an SCI procedure to perform a command.

Common Memory — A memory area that can be shared by more than one routine. Also see System Common.

Communications Register Unit (CRU) — The Business System computer general-purpose, command-driven I/O mechanism. The CRU is used to control I/O to low-speed peripherals such as video display terminals, cassette drives, EIA devices, card readers, line printers, communications devices, ASR and KSR devices.

Compose Mode — A mode of operation in the Text Editor in which a new line is introduced each time the RETURN key is pressed. It is used to enter text when creating a new file or to insert efficiently a large volume of data into an existing file.

Concatenated File — A set of two or more physical files (sequential or relative record) recognized as a logically contiguous set of data. A concatenated file is accessible by the logical name used when defining the concatenated file.

Concurrent Tasks — Tasks that are simultaneously active, that is, ready to execute.

Configuration File — A file describing a DNOS system created during system generation.

Context Switch — A transfer of control to a program or subroutine, activating a workspace associated with the program or subroutine as control passes to the new program counter contents. The context switch stores the program environment, that is, the workspace address, the address of the next instruction in sequence, and the program status. A Return with Workspace Pointer (RTWP) instruction restores the environment by returning the stored data to the workspace pointer register, the program counter, and the status register. Context switches transfer control to subroutines when an interrupt occurs, when an extended operation instruction is executed, or when a Branch and Load Workspace Pointer (BLWP) instruction is executed.

Controlled Mode — A mode of operation in which one task (the controlled task) yields control to another task (the controlling task), with the provision that the controlled task will later regain control. One example of a controlling task is the Debugger initiated by an Execute Debug command.

Copyable — An attribute of a segment specifying that an in-memory copy of the segment can be duplicated for another use.

Country Code — An indicator carried within system data structures to indicate the country in which this file or program is used.

CPU — See Central Processing Unit.

CPU-Bound — A property that describes a program when it uses more execution time for CPU operations than I/O operations. See I/O-bound.

Crash Code — A numeric code, displayed on the front panel indicator lights, that indicates the operating system has detected an uncorrectable system error and has aborted itself (crashed).

CRU — See Communications Register Unit.

Currency — A block of memory that contains information about the current record position while processing a key indexed file.

Cylinder — The set of bands on the recording surfaces of a multiplatter disk unit that are accessed when the read/write heads are in a vertically linear position. Each recording band is called a track and is divided into sectors.

Deadlock — A system state in which two or more programs are stalled—contending for resources in such a way that none can continue unless others release resources they already hold. The operating system will detect the presence of deadlock in some cases, but is unable to prevent or avoid a deadlock. It is therefore the user's responsibility to organize resource requests carefully when contention is possible.

Debug — A procedure to remove mistakes from a program. Debugging aids include extensive error detection capabilities and the interactive Debugger.

Default Value — A value used by the system if a field prompt for an SCI procedure is null. Also see Initial Value.

Deferred Write — A file creation attribute that causes logical records to be written to disk after they are buffered in memory. File I/O normally uses a deferred write operation to place logical records, being written to disk files, into an area of memory that corresponds to a physical record. Later, they are written to disk when the system requires the memory space occupied by the buffer that contains the physical record. For disk files, deferring disk writes can significantly increase system throughput. It is the system default. Also see Immediate Write.

Device — Physical equipment, such as a card reader or line printer, to which the system allows input or output.

Device Name — A four-character pathname for a physical peripheral device.

Device Service Routine (DSR) — A routine within the operating system that communicates directly with an I/O device. It services interrupts and performs the desired input and output operations.

Diagnostics — Information, messages, and routines that provide assistance in detecting hardware or software errors and clarifying special conditions. See Online Diagnostics.

Directory — A relative record file that contains the information necessary to locate other files and to describe the characteristics of those files. It does not contain user data.

Disk Components — On a disk medium, each recording band is called a track. These tracks are divided into lengths called sectors. Cylinders are those tracks on a multiplatter disk unit that are accessed when the read/write heads are in a vertically linear position.

Disk Initialization — See Initialization, Disk.

Disk-Resident Task — A task that resides on disk when it is in a terminated state or is not yet used. Also see Memory-Resident Attribute.

Disk Volume — A disk unit storage medium that has been named and initialized. Once the volume is loaded in a disk drive and installed on the system, it is known to the system by the volume name.

Download — Performing an initial program load (IPL) for a system from a communications link attached to a remote computer rather than from a local storage device. See Initial Program Load.

Drive — A peripheral device that holds and operates a disk volume, a magnetic tape reel, a cassette, or other similar medium.

DSR — See Device Service Routine.

Edit Mode — A mode in the Text Editor in which a command function or key is used to enter each new line in a text file. Also see Compose Mode.

End Action — A routine specified by a task, that is to be executed before aborting the task if a fatal error occurs or if the task is terminated. This routine is called the end action routine, and the task is described as having taken end action. Unless a Reset End Action instruction is performed, a task can perform end action only once.

End-of-File (EOF) — A mark or other identification that denotes the end of a sequential file of data.

End-of-Information (EOI) — A mark or other identification that denotes the point beyond which there is no information.

End-of-Medium (EOM) — A mark or other identification that denotes the end of available storage space for a file of data. Also referred to as end-of-volume.

End-of-Record (EOR) — A mark or other identification that locates the end of a logical record on a file or device.

Enqueue — To place a request on a list or queue.

Entity — A system component being described in the inquiry mode of the system generation utility. A device is an example of an entity.

EOF — See End-of-File.

EOI — See End-of-Information.

EOM — See End-of-Medium.

EOR — See End-of-Record.

Error Code — A numerical code returned when an error is detected. Depending on the cause of the error, this code can be returned in a supervisor call block, communicated to a terminal, or displayed on the programmer panel.

Event Characters — Characters, generated by keys on the keyboard, that have priority over data keys and have special significance to the executing task.

Executable — An attribute of a segment specifying that the segment contains executable code. (This option is not supported on small Business System computers.)

Executing Task — A task that has control of the CPU.

Expandable File — A file that can be extended beyond its initial size.

Fatal Error — An error in a task that causes the task to be terminated. Examples of fatal errors include boundary errors, memory parity errors, use of illegal instructions, and other nonrecoverable situations.

FCB — See File Control Block.

FDB — See File Directory Block.

FDR — See File Descriptor Record.

Field Prompt — A word or phrase used by an SCI command procedure that signals the user to input information relevant to further processing.

File — A named, organized collection of data records. Disk files can be organized in a sequential, relative record, or key indexed format. Special forms of relative record files include directory files, program files, and image files.

File Attribute — A declared characteristic of a file that limits the types of operations performed on a file. For example, delete protection is an attribute, and files with this attribute cannot be deleted until the attribute is removed.

File Control Block (FCB) — A memory structure that describes the name, location, and characteristics of a disk file. The FCB is built as a result of an Assign LUNO operation to a file.

File Descriptor Record (FDR) — A disk directory record that describes the name, location, and characteristics of a disk file.

File Directory Block (FDB) — A single node of an in-memory directory tree structure used to locate file control blocks for files on which an Assign LUNO operation has been performed.

Foreground Mode — The mode in which an executing task interacts with the terminal. After SCI bids a foreground task, SCI is suspended and releases access to the terminal and the foreground terminal local file. The task that is bid shares the environment currently being used by SCI. When using SCI from a batch stream, the foreground mode of operation is equivalent to background mode. See Batch Stream, Background Mode.

Front Panel — The display of system conditions and program counter locations on the front of the computer chassis.

Global LUNO — A logical unit number that is available to any job or task.

Hard Break — See Break Key Sequence.

Hashing — A technique for mathematically processing key words or file names to produce numbers, usually record numbers.

Hexadecimal (Hex) — A numerical system using a radix of 16. Hexadecimal values are indicated by a leading "> ", or by the phrase "hexadecimal" or "hex", when showing the value.

Image File — A special form of relative record file in which the logical record size equals the physical record size, which equals the disk sector size. Image files usually contain a memory image of some code. These files are designed so that a program image can be read into memory in one disk access.

Immediate Write — A file creation attribute that forces output to a disk file to occur immediately. Usually, file I/O is buffered in memory and deferred until the memory space is required.

Initial Program Load (IPL) — The operation of pressing HALT and LOAD on the front panel of the Business System computer in order to trigger the work of the ROM Loader in performing the bootstrap sequence. The term is also used to refer to the entire sequence of placing the initial program load (IPL) program in memory and loading the operating system.

Initial Value — A value shown next to a field prompt from an SCI command procedure. The user can choose to use this value as the response to the prompt or can replace it with another response. If the initial value is replaced by a null value, the system will use a default value for the field prompt, if one exists. The initial value is specified by the command procedure to be a constant value or to be the current value of some synonym. Also see Default Value.

Initialization, Disk — The process of writing the required system overhead tracks and formatting a disk prior to using the disk.

Initiate I/O — An I/O call that causes I/O to be started but does not cause the requesting task to be suspended during the time that the I/O is in progress. This mode of I/O is specified by a flag in the SVC block.

Inquiry Mode — A mode in the system generation program in which a new system configuration is described through specific prompts.

Install — When applied to programs, to install means to place a task, procedure, segment, or overlay into a program file. When applied to volumes, install means to direct the system to refer to an initialized volume on a given drive by the supplied name.

Installed ID — A unique identification number from hexadecimal 1 to FF assigned to a task, procedure, segment, or overlay that is installed in a program file. The operating system uses this identification to locate that module in the program file.

Interactive Mode — See Foreground Mode.

Interprocess Communication (IPC) — The capability for two or more tasks to exchange information. Also see Channel.

Interrupt — A coded signal that can transfer CPU control; thus enabling the CPU to service devices, power up/down procedures, and error conditions. The Business System computers support interrupts with 16 levels of priority. The interrupt that is presently executing prevents interrupts of a lower priority.

Interrupt Service Routine — A system module that directs the system to take action according to the interrupt received. Examples include: power up or down, error instructions, and clock operations.

Intertask Communication Block — An internal data structure used by the Get Data and Put Data SVCs.

I/O-Bound — A property that describes a program when it spends more execution time in I/O operations than CPU operations. See CPU-bound.

IPC — See Interprocess Communication.

IPL — See Initial Program Load.

JCA — See Job Communications Area.

Job — An entity within the system that exists to perform a user-defined function. Many times, the user will see a job as an interactive or batch execution of the System Command Interpreter. In general, however, a job has the following characteristics:

- Is uniquely identified with a job ID
- Possesses authority through an accounting and security system to own and access permanent and job-local files
- Consists of one or more cooperating tasks that can execute concurrently or serially

Job Communications Area (JCA) — A section of memory used by the system for information concerning a particular job. Each job has a unique JCA.

Job ID — A 16-bit integer that uniquely identifies a job within a given site.

Job-Local LUNO — A logical unit number that is limited for use within a particular job, but not for any particular task in that job.

Job Name — An eight-character name associated with a job. A job name does not have to be unique within the system and is supported only for user convenience.

Job Status Block (JSB) — A memory structure in the system table area that contains information describing the state of a job.

Job-Temporary File — A temporary file which, after creation, exists for the life of the creator's job. A job-temporary file is accessible by a logical name or by the pathname if it is known.

JSB — See Job Status Block.

K — An abbreviation for 1024, often used when referring to a number of bytes of memory.

Kernel — The memory-resident part of DNOS, which includes the device service routines, interrupt processors, extended operation processors, system tables, device buffers, the task scheduler, nucleus support functions, many supervisor call processors, and memory-resident tasks.

Kernel Program File — A program file containing the loadable memory image of the operating system.

Key — A character field within a record in a key indexed file, used to determine the order of the record in relation to other records in the file.

Keyboard Send/Receive (KSR) — A line-oriented terminal device, such as a teletypewriter.

Keyboard Status Block (KSB) — A block of memory used as a workspace by an interrupt service routine for a keyboard device.

Key Indexed File (KIF) — A file in which records can be accessed by the value of a character string called a key. Each record can have up to 14 unique keys. Access through each key is not dependent on the other keys.

Key Value — A particular character string being used in a key field.

KIF — See Key Indexed File.

KSB — See Keyboard Status Block.

KSR — See Keyboard Send/Receive.

LDT — See Logical Device Table.

Least Recently Used Strategy — A strategy used by the operating system to choose task segments to be swapped out of memory. The segment that has been inactive the longest is recognized as the least likely to be used in the immediate future, so it is the first selected for swapping to disk.

Link Control File — A file created by the user that contains instructions for the Link Editor.

Link Editor — A system utility that takes related object modules and links them together into a single load module. The linking process resolves symbol references between separate modules to produce an executable structure.

Linked Object File — A file created by the Link Editor that contains one or more program object modules that have been linked together to produce linked object modules.

Load — To copy a task or segment from an external storage medium into the memory of the computer in preparation for execution.

Logical Address Space — The memory accessible to a task. The maximum extent of any logical address space is 64K bytes.

Logical Device Table (LDT) — A memory structure in the system that contains the LUNO and pointers to system tables that correspond to the resource to which the LUNO is assigned.

Logical Name — An alphanumeric name of up to eight characters that can be assigned to an I/O resource. The resource can be a file or device that is to be accessed by programs using the logical name, a set of concatenated files, or a spooler device.

Logical Record — A logical division of data in a file. A logical record can be transferred with a single I/O supervisor call.

Logical Record Length — The length (in bytes) of records in a file as accessed by a program. This length does not need to correspond with any physical division of the medium.

Logical Resource Name — See Logical Name.

Logical Unit Number (LUNO) — A number specified in an I/O operation that represents a file, channel, or device.

Log-Off — The action that ends a work session. Using the supplied System Command Interpreter, log-off is accomplished by entering a Quit (Q) Command when the SCI prompt is shown.

Log-On — The action that begins a work session. It identifies a user to the system and allows the user access to it. Using the System Command Interpreter, log-on is accomplished by entering a keyboard sequence, such as the Attention key followed by an exclamation point.

LUNO — See Logical Unit Number.

M — An abbreviation for 1,048,576, when referring to a number of bytes of memory.

Machine Operator — See System Operator.

Master/Slave Channel — A type of channel in which the owner of the channel (the master) interprets and/or executes the messages and/or commands transmitted by requestors (slaves) of the channel.

Memory-Based Segment — A segment not associated with a file. It is created by a user program and assigned attributes at run time.

Memory Mapping — A hardware feature of the Business System computer controlled by the operating system software, which allows the computer to address 2048K bytes of memory (even though the standard address format of 16 bits could only address, at most, 64K bytes). Memory mapping is not available on some models of the Business System computers.

Memory-Resident Attribute — The characteristic of a task or segment that indicates it is loaded at the time of the initial program load sequence and remains in memory even when it is in a terminated state. The term can also refer to an attribute of a segment which specifies that the segment permanently resides in memory after it has been created.

Modem — (Modulator-Demodulator) A device that interfaces between the digital signals of a computer and the analog signals of data transmission facilities.

Module — A set of computer program instructions treated as a unit by an assembler, compiler, link editor, or similar processor.

Multifile Set — A set of two or more physical files (key indexed files) recognized as a logically contiguous set of data. The set of files composing a multifile set must be accessed only by the logical name assigned to the set.

Multiprogramming — A mode of operation in which more than one computer program can be in memory and queued for execution at the same time. This mode is different from multiprocessing, which enables two or more programs to execute simultaneously using two or more processors.

Multitasking — Another term for multiprogramming. Several tasks can be in memory simultaneously, each allotted execution time in turn.

Multivolume File — A logical file that spans volumes. A multivolume file is accessible only by a logical name.

Multivolume Set — A set of tape reels in a specified order that is recognized as a single, large volume.

Name Correspondence Table — A name management structure which contains the synonyms, logical names, and field prompt values presently defined for a job.

Nonblank Suppressed — The characteristic of a sequential file that indicates that consecutive blanks in a record are stored as consecutive blanks, rather than being stored in a compressed format.

Nonexpandable File — A file that cannot be extended beyond its initial size.

Object Code — Machine language code together with load control code.

Object File — A file that contains one or more program object modules, usually created by an assembler or compiler.

Online Diagnostics — Nonprivileged diagnostic routines that operate concurrently with program execution. See Diagnostics.

Operator — See System Operator.

Overlay — A part of a task that resides on disk until specifically requested. When requested, the overlay replaces part of the task previously in memory. The use of overlays can reduce the amount of memory required by a task to the amount required for the largest segment needed at one time.

Parameters — Arguments or input that specify an attribute, resource, or value to a command processor, supervisor call, or user program.

Password — A character string supplied by a user for validation of access and security privileges, primarily for log-on.

Pathname — A character string that indicates a path to a resource such as a file, channel, or device. For a file or channel, the pathname components include an optional volume name, 0 to 24 directory names, and a final component identifying the file or channel.

PC — See Program Counter.

PDT — See Physical Device Table.

Peripheral Device — Any equipment in a computer system, other than the CPU itself, that can provide information, communication, and/or capabilities to the system. Examples include disk units, VDTs, printers, and communication equipment.

Permanent Name Definition Table — A disk data structure which contains the synonyms and logical names presently defined for a specified user ID.

Physical Address Space — The addressable memory of the computer. This term also refers to the range of memory addresses available to the computer although the memory may not actually be implemented in a particular configuration.

Physical Device Table (PDT) — A data structure associated with a device and used by device service routines.

Physical Record Length — The number of bytes of data transferred as a unit to and from a disk or tape. Often, a physical record includes several logical records.

Preanswered Parameter — A parameter for the system generation program that can be explicitly changed by the user; however, no prompt is provided.

Preemptive Scheduling — The task scheduling algorithm that causes the highest priority active (ready) task to be executed even if a presently executing task must be preempted from its time slice.

Priority, Initial — The priority given to a task by the system when the task is initially loaded. Initial priorities are in the range of 0 through 255 (with 0 highest and 255 lowest). The initial priority is based on the installed priority of the task and the priority of the job under which it is executing. Priority 0 is for tasks installed at system priority, priorities 1 through 127 are for real-time tasks, and priorities 128 through 255 are for time-sharing tasks.

Priority, Installed — The priority given to a task at installation time. For time-sharing tasks it can be one of five priorities (0,1,2,3,4). Priority 0 indicates system priority. Priorities 1, 2, or 3 are assigned to all other fixed priority time-sharing tasks, with 1 having the highest priority, and 3, the lowest. Priority 4 indicates a floating or dynamic time-sharing priority. Real-time tasks can be installed at any one of 127 priorities (1 through 127).

Priority, Run-Time — The priority used to schedule the CPU. Initially, it is the same as the initial priority but can vary (for tasks with dynamic priority) from the initial priority depending on whether the task is I/O-bound or CPU-bound. I/O-bound tasks have run-time priorities higher than their initial priorities, and CPU-bound tasks have run-time priorities lower than their initial priorities.

Privilege — A set of attributes for tasks installed on a program file. "Hardware privilege" allows the task to execute privileged machine instructions. "Software privilege" allows the task to execute SVC operations that are designated as privileged SVCs.

Procedure Library — A directory of files containing SCI command procedure definitions.

Procedure Segment — A segment of a task that can be shared with any other tasks; usually consists of executable code.

Program — The instructions and data that perform a particular function. A program consists of one or more segments that define a logical address space for a set of instructions and data. Also see Task.

Program Counter (PC) — A hardware register that indicates to the computer the next instruction to be executed.

Program File — A special form of relative record file used to contain executable programs and segments in memory image form. A program file contains system generated information that enables more than one program to be stored in the file.

Programmer Panel — A peripheral device mounted on the front of the computer providing an elementary interface to the computer as well as status indicators and program counter values. Most Business System computers do not have a programmer panel.

Queue — A first-in, first-out waiting list. A queue is a data structure consisting of a list of items to be processed. The order of the list is chronological. The first item entered on the list (first in) is the first item removed for processing (first out).

Random File — A relative record file.

Ready Task — A task that is ready to execute. That is, the task is queued in memory with a given priority level, waiting its turn for execution.

Record Locking — A procedure used to restrict access to a record in a file. A locked record cannot be accessed until it is unlocked. This process is useful when controlling file access by several contending tasks.

Record Lock Table — An internal data structure, associated with a file, that identifies all records of the file which are presently "locked."

Reentrant Program — A program that can be shared among several users in a multitasking environment without conflict of data among the users.

Register — See Workspace Register.

Relative Record File — A directly addressable file in which the records are numbered sequentially from the beginning of the file. The length of records in the file is a fixed number specified during file creation. Any record can be easily accessed after calculating the record's displacement from the beginning of the file. The calculation is based only on the record number and the fixed record length attribute of the file.

Relocatable — An attribute of an overlay that allows that overlay to be loaded at an address other than its natural load address. Addresses within the overlay are resolved at load time.

Replicatable Segment — A segment that can have multiple copies in memory simultaneously.

Replicatable Task — An installation attribute for a task, indicating that multiple copies of the task can reside simultaneously in memory. This attribute is frequently present for such utility tasks as the compilers and the System Command Interpreter.

Replication — The action of making available another copy of a task or segment in memory.

Reserved Segment — A segment that will not be discarded by the system even when no task has the segment mapped into its logical address space. Segments can be reserved at the job level, thus ensuring the ability to access the segment when desired. Reserved segments are subject to swapping.

Reserved Segment Table (RST) — A data structure used to contain a list of segments that have been reserved by a job.

Resource — A logical or physical commodity (such as a peripheral device, file, or memory) that can be allocated to a program. An I/O resource is any such commodity other than computer memory.

Resource Contention — The situation in which two or more tasks attempt to use the same resource at the same time.

Resource-Independent — A type of I/O operation that allows a programmer to specify any of several devices or resources. For those devices, I/O operations occur in essentially the same manner and are specified by assigning a LUNO that represents the resource. Data is accessed sequentially and is supported for such devices as VDTs, printers, sequential files, and communication channels. See Resource-Specific.

Resource Ownership Block (ROB) — An in-memory data structure that represents a job's ownership of a resource.

Resource Privilege Block (RPB) — An in-memory data structure that is used to control access privilege requests (OPEN SVCs) from tasks within a job. RPB also carries file position information for file access requests from tasks within the job.

Resource-Specific — A type of I/O operation that allows the specific capabilities of a device to be programmed. I/O occurs through a LUNO that is assigned to the resource. Resource-specific operations are supported for VDTs, relative record files, key indexed files, and communication channels. See Resource-Independent.

Reusable — An attribute of a segment that specifies the segment can be used consecutively without reloading.

RLT — See Record Lock Table.

ROB — See Resource Ownership Block.

Rolling — See Swapping.

ROM Loader — A program in Read-Only-Memory (ROM) on the system interface board at a dedicated memory address that is triggered by the IPL sequence to initiate the loading of the operating system into main memory.

RPB — See Resource Privilege Block.

RST — See Reserved Segment Table.

Run-Time ID — A unique identification number within a job, assigned by the operating system to an executing task for the duration of its execution. Also, an identification number assigned by the operating system to a segment for the duration of its existence in memory.

SAT — See Secondary Allocation Table.

Scheduler — The part of the operating system that decides which task is to receive execution time.

SCI — See System Command Interpreter.

SCI Prompt — A signal to the user that SCI is waiting for a command to be entered. The standard SCI prompt is [ ].

Scrolling — Moving the contents of the screen of a video display terminal up or down in order to view a file that contains more data than the screen can display. This is accomplished by using the Up Arrow key, Down Arrow key, or some other special key.

Secondary Allocation — A block of disk space automatically allocated by the system to an expandable file that requires more space than its present allocation. The size of the secondary allocation is specified when the file is created, but it increases with each subsequent secondary allocation.

Secondary Allocation Table (SAT) — A structure that exists in the file descriptor record of each expandable file. It contains the size (in allocatable disk units) of the secondary allocation and the starting address of the allocation.

Sector — See Disk Components.

Segment — A piece of software occupying a single block of memory, or an in-memory image on disk. Each segment has a set of attributes that describes its characteristics.

Segment Group — A set of related segments, such as those from the same program file or those in a memory-based set of segments.

Segment Group Block (SGB) — An internal data structure that defines the access to a specific set of segments.

Segment Group LUNO — A logical unit number assigned to a file associated with a segment group.

Segment Status Block (SSB) — An internal data structure that describes the attributes, state, and location of a segment.

Segmentation — The process of separating a task into segments. A task can be composed of as many as three segments. During execution, only three segments can be addressed at any given time. However, additional segments can be accessed by making supervisor calls to select a new segment to replace one that is presently addressable. If desired, segments can be shared by tasks.

Semaphore — A job-local variable used for exchanging signal information between tasks and for program synchronization. A set of SVCs is provided for handling semaphores.

Sequential File — A file of variable length records accessed in successive order (that is, the order in which the records are written to the file).

SGB — See Segment Group Block.

Shareable — An attribute of a segment specifying that the segment can be shared concurrently by more than one task.

Site — A particular collection of computing resources identified by a name of one to eight characters.

Source File — A file containing one or more program source modules (source code or statements). It is usually created using the Text Editor.

Special Table Area — A section of memory used by system functions to contain data structures specific to the function (for example, file management table area).

Spooler Task — A task responsible for spooling a job's output data to job local files and enqueuing the files to be printed by the Spooler Job.

Spooling — The process of queuing files of data for printing. The files are scheduled for printing depending on the priority assigned to the job issuing the print request. Files can be printed on special forms or in special formats as specified by the user.

SSB — See Segment Status Block.

ST — See Status Register.

STA — See System Table Area.

Stage — The environment established to bid a task by using the System Command Interpreter. Specifically, the synonyms and logical names available to the task that is bid.

Station — An interactive terminal; for example, a video display terminal or a hard-copy device such as the 733 ASR terminal.

Status Register (ST) — A hardware register on the Business System computer that holds condition codes set by preceding operations, fault flags, mode control information, and the interrupt mask.

Subdirectory — A directory that is pointed to by another directory. Every directory on a disk, except VCATALOG, is a subdirectory. If directory A contains the name B and a pointer to directory B, then B is referred to as a subdirectory of A.

Supervisor Call (SVC) — A user task request for operating system services. A supervisor call is an instruction that performs a context switch into the operating system, which then interprets the call and performs the desired service, if that service is allowed to the user.

Supervisor Call Block — A list of necessary parameter values for a supervisor call. The size and content of the supervisor call block depend on the particular call being made.

Suspended Task — A task that was temporarily removed from the active list and from execution as a result of a supervisor call. When reactivated, suspended tasks are executed from the point of suspension.

SVC — See Supervisor Call.

Swap Directory Table — An in-memory data structure that represents each segment on the swap file.

Swap File — A file on the system disk that maintains segments that have been temporarily unloaded from main memory to be used in future operations.

Swapping — The action of copying a segment from memory to a special system file (the swap file) on the system disk so that the memory it was occupying can be allocated to a segment of a higher priority task; or, the action of copying a segment from the swap file into memory when the segment is needed again.

Symmetric Channel — A type of channel in which the channel owner task and requestor tasks issue simple Read and Write commands that must match each other. The Read command of one task is processed when the other task issues a Write command and vice versa.

Synonym — A text string that functions as an alternative for another string. Usually a synonym is shorter than the text it replaces and is more convenient to use.

Sysgen — See System Generation.

System — An attribute of a segment specifying that the segment can be used only by a system task.

System Command Interpreter (SCI) — The user interface to the operating system. It prompts the user, accepts input, interprets commands, and directs activities to satisfy those commands.

System Command Interpreter Language — The language in which command procedures for the System Command Interpreter are written.

System Common — An area of memory accessible to all tasks through the Get Common Data Address supervisor call. The system common memory area is a system parameter supplied during system generation. It is accessed as one of the memory segments of each task that uses it.

System File — A disk file reserved for use by the operating system. The name of a system file begins with S$.

System Generation — An interactive process in which a new version of the operating system is configured to match a particular hardware installation. A set of static data structures is created that represents the configuration. Also, certain parameters supplied during system generation allow the fine tuning of system performance. Also see Command Mode, Inquiry Mode.

System Job — The job within which operating system tasks are executed.

System Log Reporting — The capability of the system to report hardware and software errors to a file and to an optionally specified logging device. Any task can also write a message to the system log using the System Log SVC.

System Operator — The person who controls system start and restart, places information media into the input devices, removes the output, and performs other related functions.

System Program File — A program file containing tasks that are part of the DNOS package (including utilities). They are not necessarily system tasks.

System Table Area — Memory reserved for the operating system and used for system data structures and buffers. The size of the system table area is specified during system generation.

System Task — A task that executes in an environment that includes part of the operating system memory space.

System Time Unit — A standard operating system measurement of 50 milliseconds.

Tape File — A set of records enclosed by tape marks on magnetic tape. No End-of-File mark can appear within a tape file.

Tape Volume — A magnetic tape reel that has been initialized.

Task — A program that executes under control of the operating system. At least one task exists for each job. A task consists of an address space, a program counter, a workspace pointer, and a status register. The address space of the task can consist of one or more segments. Segments can be dynamically exchanged with other segments during task execution, with a limit of three segments comprising the address space at any one time. Also see Procedure Segment, Task Segment.

Task-Local LUNO — A logical unit number that can be used only by the task that assigns it.

Task Parameters — A set of modifiable information (four bytes), contained in an Execute Task SVC, that can be accessed through a Get Parameters SVC.

Task Segment — That part of a program that contains the initial workspace pointer, entry point address, and end action address. The task segment may also contain the workspace, data, and program code. Also see Program, Task, and Procedure Segment.

Task State — The present disposition of a task. For example, a task can be in execution (executing state), suspended for any of a number of reasons (with separate states to describe these), or in any of several other states.

Task Status Block (TSB) — An internal data structure, representing a task in the system, that contains the necessary information to describe the state of the task.

Teleprinter Device (TPD) — A type of device used for input and/or output. The TPD prints a hard copy of the output. If used for input, the device also has a keyboard. This category includes printers and hard-copy terminals.

Teletypewriter — See TTY.

Template — A diagram that includes the descriptions of various fields of a DNOS data structure, such as their locations, meanings of flags, and comments.

Temporary File — A file that DNOS deletes automatically after all LUNOs assigned to the file are released. See Job-Temporary File.

Terminal — Any interactive user device. Also see Station.

Terminal Local File (TLF) — A file associated with a background task or a foreground task when using the System Command Interpreter in an interactive job. The file is used for communicating information from a command processor to the user.

Terminated Task — A task that has been removed from execution and from the active list. Termination occurs when the task completes normally, when the system detects a fatal error on the part of the task, or when a user commands the operating system to end the task. When reactivated from the terminated state, a task is executed from its beginning.

Text Editor — A utility task that allows you to create and/or modify files of user-defined data. The Text Editor is a cursor-oriented VDT editor.

Thrashing — The state of the system in which the overhead required to resolve resource contention uses most of the system's time.

TILINE* — The Business System computer asynchronous 50 megabit per second memory bus, used by the CPU, memory, disk and tape controllers.

TILINE Peripheral Control Space — A range of TILINE addresses reserved for TILINE device controller interfacing.

Time-Out — The end of that time period during which a device must respond to remain an active device.

* TILINE is a registered trademark of Texas Instruments Incorporated.

Time Slice — The amount of execution time allocated to a task when it is scheduled for execution. A preanswered parameter of the system generation utility is used to specify the length of a time slice.

Time Unit — See System Time Unit.

TLF — See Terminal Local File.

TPD — See Teleprinter Device.

Transfer Vector — A pair of memory addresses in two consecutive words of memory. The first word contains the address of a 16 word area of memory, called a workspace. The second word contains the address of a subroutine entry point. Also see Context Switch.

TSB — See Task Status Block.

TTY — A line-oriented terminal, such as a teletypewriter.

TTY Mode — Line-oriented I/O access to a terminal. A VDT can be accessed in either TTY or VDT mode by the System Command Interpreter.

Unblocked File — A file in which each physical record contains one logical record. During data transfer to or from such a file, no internal buffering is required.

Unload — Refers to the software action of preparing a disk or tape medium for removal, as well as the act of physically removing the medium from the drive.

Updateable — An attribute of a segment that specifies the segment can be written to its permanent file position.

User — Any person who uses a DNOS-based system.

User ID — An eight-character string that identifies a user.

VCATALOG — The highest level directory on each disk volume that specifies, either directly or indirectly, all files on the volume.

VDT — A video display terminal.

VDT Mode — A screen-oriented mode, with the ability to read and write fields at any position on the screen. Although video display terminals can be used as hard-copy emulators (TTY mode), they can also be used in this mode to fully utilize the VDT's power and speed capabilities.

Volume — A logical device, such as a disk pack or magnetic tape reel, that can be uniquely identified by name and can store one or more logical files.

Volume Information — Data stored in track 0, sector 0, of every initialized disk volume, describing system information and the address of VCATALOG.

Volume Name — A character string that identifies a volume. It is used when installing and unloading the volume and can be used as part of the pathname for files contained within it. Disk volume names can have up to eight alphanumeric characters, but must begin with a letter.

WCS — See Writable Control Store.

Word — A group of binary digits that can be operated on as a single unit. The Business System computer has 16 binary digits in a word.

Workspace — A 16-word area of memory addressed as workspace registers 0 through 15. The active workspace is defined by the contents of the workspace pointer.

Workspace Pointer (WP) — The hardware register that contains the address of workspace register 0, and indicates the presently active workspace.

Workspace Register — A memory word that is accessible to an instruction of the computer as a general purpose register. It can be used as an accumulator, a data register, an index register, or an address register.

WP — See Workspace Pointer.

Writable — An attribute of a segment that specifies the segment can be modified when in memory. (This attribute is not available on some models of the Business System computers.)

Writable Control Store (WCS) — A part of computer memory that is separate from general computer memory. WCS can contain supplied or user-written microcode instructions that perform the operations of assembly language instructions.

XOP — A 990 assembly language instruction (extended operation) that generates a software interrupt.

> — A symbol used to indicate that the digits which follow are base 16 (hexadecimal) digits. For example, >11 is equal to decimal 17.

— A symbol used in Pascal to indicate that the digits which follow are base 16 (hexadecimal) digits.

Index

This index lists key topics of this manual and specifies where each topic appears, as follows:

- Sections — Section references appear as *Section n*, where *n* represents the section number.
- Appendixes — Appendix references appear as *Appendix Y*, where *Y* represents the appendix letter.
- Paragraphs — Paragraph references appear as alphanumeric characters separated by decimal points. The first character refers to the section or appendix containing the paragraph, and any other numbers indicate the sequence of the paragraph within the section or appendix. For example:
  - *3.5.2* refers to Section 3, paragraph 5.2.
  - *A.2* refers to Appendix A, paragraph 2.
- Figures — Figure references appear as *Fn-x* or *FY-x*, where *n* represents the section and *Y* represents the appendix containing the figure; *x* represents the number of the figure within the section or appendix. For example:
  - *F2-7* refers to the seventh figure in Section 2.
  - *FG-1* refers to the first figure in Appendix G.
- Tables — Table references appear as *Tn-x* or *TY-x*, where *n* represents the section and *Y* represents the appendix containing the table; *x* represents the number of the table within the section or appendix. For example:
  - *T3-10* refers to the tenth table in Section 3.
  - *TB-4* refers to the fourth table in Appendix B.
- See and See also references — *See* and *See also* direct you to other entries in the index. For example:

  Logical Unit Number ..... *See* LUNO

  Device ..... *See also* individual device names or numbers

Page numbers that correspond to these index references appear in the Table of Contents.

CUT ALONG LINE

USER'S RESPONSE SHEET

Manual Title: DNOS Concepts and Facilities (2270501-9701)

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.
FOLD ON TWO LINES (LOCATED ON REVERSE SIDE), TAPE AND MAIL

FOLD

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
DATA SYSTEMS GROUP

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

FOLD