SEQUEL™ by Microdata

Programmer's Reference Manual

SEQUEL™

PROGRAMMER'S REFERENCE MANUAL

83-1109

Series 1.0
Release 1.2
June, 1984

For Series 1.0 SEQUEL Systems

Price: 25.00

Microdata Corporation

17481 Red Hill Avenue, Irvine, California 92714
Post Office Box 19501, Irvine, California 92713
Telephone: 714/250-1000 · TWX: 910-595-1764

WARNING: This equipment generates, uses, and can radiate, radio frequency energy, and if not installed and used in accordance with the instructions manual, may cause interference to radio communications. As temporarily permitted by regulation, it has not been tested for compliance with the limits for Class A computing devices, pursuant to Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference. Operation of this equipment in a residential area is likely to cause interference, in which case the user, at his own expense, will be required to take whatever measures may be required to correct the interference.

TABLE OF CONTENTS

1 INTRODUCTION

1.1 Microdata™ SEQUEL™ Computer System

The SEQUEL computer system is a generalized system for data base management. It is a complete system providing multiple users with the capability to instantly update and/or retrieve information stored in on-line data files. Users communicate via local or remote terminals with computer files that may be private, common, or security-controlled. Each user's vocabulary can be individually tailored to meet specific applications.

SEQUEL systems are built of field-proven Microdata computers and peripherals, using microprograms to provide you with unrivaled performance and reliability in the super-mini computer market.

The SEQUEL computer system includes the powerful, yet simple to use, ENGLISH® retrieval language, DATA/BASIC™ and PROC high level languages, the EDITOR processor, complete program development facilities and file maintenance tools, and a host of other user amenities. SEQUEL systems run in an on-line, multiuser environment with all system resources and data files efficiently managed by a microprogrammed virtual-memory operating system.

SEQUEL has advantages from every angle: system capability, multiuser performance, file management languages, ease of programming, data structure, and architectural features. SEQUEL's high performance and fast response time are made possible by extensive use of high-speed microprocessors which greatly reduce program execution time and system overhead. The entire SEQUEL computer system is unique -- one of a kind.

Microprogrammed firmware contains:

- Virtual memory manager
- Multiuser operating system
- Special data management instructions
- Input/output processors

System software includes:

- ENGLISH, DATA/BASIC, PROC, EDITOR and ASSEMBLY languages
- Selectable/automatic report formatting
- Dynamic file/memory management
- RUNOFF™ text processing
- SCREENPRO™ language -- an easy way to set up terminal displays

The file structure provides:

- Variable length files/records/fields
- Multivalues (and subvalues) in a field
- Efficient storage utilization
- Fast access to data items
- Selectable degrees of data security
- File size limited only by size of disc
- Item size up to 32,267 bytes

1.2 The Flexible Family of Microdata Computer Systems

The expandable Microdata family of high-performance data base management processors ranges from an economical system for first-time users with limited data processing requirements and/or experience, to the high capacity systems used by some of the largest companies in the United States.

Besides superb performance, the entire Microdata computer line offers unmatched growth advantages. As your company grows, you can add Microdata equipment to meet its increased data processing needs without the costly replacement and conversion charges usually associated with updating computer facilities. All Microdata systems are both hardware- and software-compatible. Start with Microdata. Grow with Microdata.

A basic SEQUEL system has:

- Central processing unit (CPU and cabinet)
- Diagnostic Maintenance Processor
- Mass storage disc drive
- Input/Output Processor
- Magnetic tape drive
- Asynchronous communications line controller
- PRISM® 4 cathode ray tube (CRT) data terminal (up to 127)
- System printer

All Microdata systems operate in Microdata's easy-to-use ENGLISH retrieval language, as well as the more advanced DATA/BASIC and PROC, and are fully compatible with REALITY data processing systems.

Microdata's most advanced microprocessing technology was used in the design of the SEQUEL system to enable greater load capacity and still faster data processing to handle more applications -- all without overloading the central processing unit or degrading the speed of terminal response. This advanced system's exceptional power and adaptability provides up to 127 separate users with fingertip access to voluminous business, technical, and scientific applications that use data base management techniques.

- Complete business computer capabilities
- Microprogrammed virtual memory operating system
- Up to 127 users and one billion characters of file storage
- On-line file update/retrieval
- ENGLISH retrieval language
- Fast terminal response
- Multiple printer spooling
- Optional communications capabilities
- Special data management processors
- High-speed sort capability
- Small computer price
- Big computer performance
- Top to bottom computer/peripheral compatibility within the Microdata family

Figure A. Microdata SEQUEL System Advantages

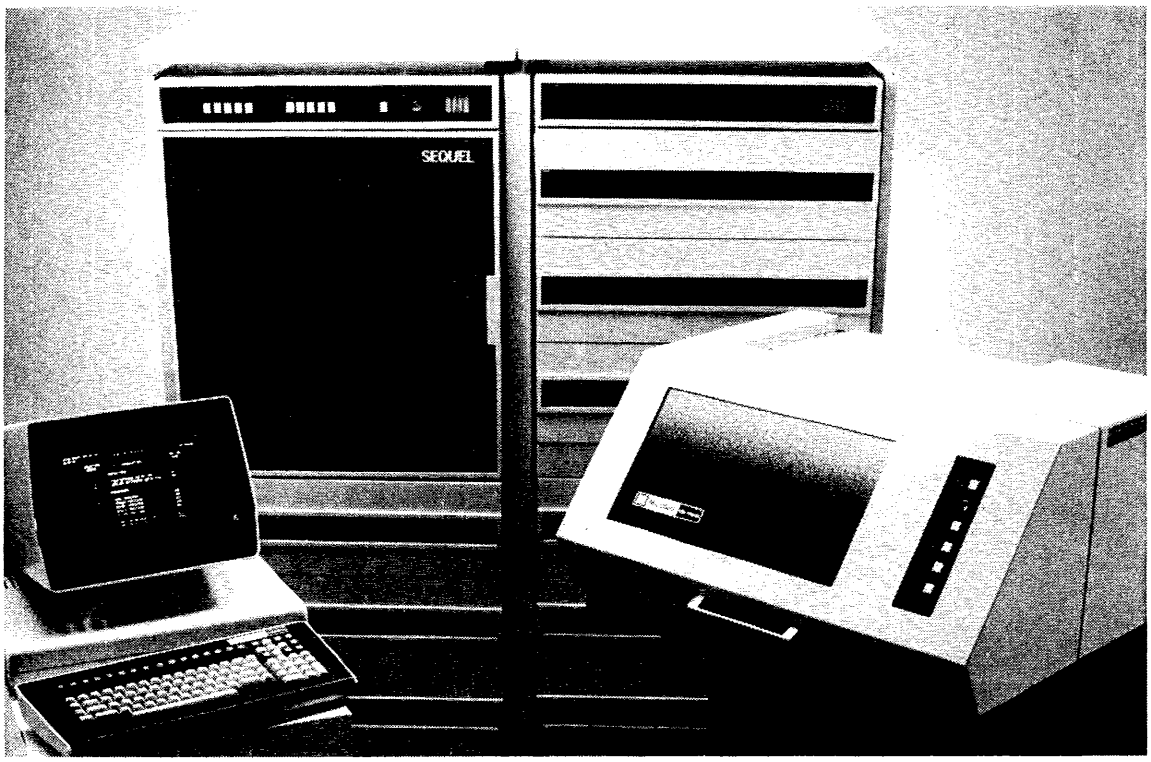


Figure B. Typical Microdata SEQUEL System

1.3 SEQUEL Software

Processors available on the SEQUEL computer system comprise the most extensive data base management software available on any minicomputer. Overviews of the processors and their typical uses follow.

ENGLISH Language

ENGLISH is a generalized data retrieval/report generator language. A typical ENGLISH inquiry consists of a relatively free-form sentence containing verbs, file-names, data selection criteria, and control modifiers. An easy-to-use, dictionary-based language that uses simplified prose statements, ENGLISH permits you to produce original reports rapidly and efficiently.

ENGLISH applications are limitless because of the ease with which output can be accessed from user files. Since nonprogrammers can master the process quickly, ENGLISH is a valuable information management tool for many people in an organization, from sales personnel to top-level executives. Its major uses are report generation and inquiry/response applications. You may also use ENGLISH to produce output after file updates with DATA/BASIC or PROC software, as well as for printing one-of-a-kind reports without writing a program.

SCREENPRO

The SCREENPRO processor was developed to minimize the software gap between the establishment of data files and the creation of reports. No longer must you develop your own method of creating and processing screens to display text, inputs, validations and updates.

Because SCREENPRO requires fewer program statements, it greatly simplifies program maintenance while increasing operator and programmer efficiency. Data throughput is accelerated. A screen can be designed, displayed, tested and changed without affecting the program.

DATA/BASIC

BASIC (Beginners All-purpose Symbolic Instruction Code) is a simple, yet versatile, programming language suitable for expressing solutions to a wide range of problems. DATA/BASIC, an extension of Dartmouth BASIC, is especially easy for the beginning programmer to learn.

DATA/BASIC is the primary method of updating user files on a SEQUEL system. Because of its flexibility, DATA/BASIC is used for a variety of business applications including accounts payable/receivable, general ledger, inventory control, payroll, sales forecasting/analysis, order processing, invoicing, claims processing, data entry, and other projects.

With the addition of SCREENPRO, DATA/BASIC programs are even easier to write -- and run faster -- since screen handling and data validation can be removed from the program.

PROC

The PROC processor enables you to prestore a complex sequence of operations which can then be evoked by a single word command. Any sequence of operations that can be executed from the terminal can be prestored in a PROC. Although PROC is similar to the Job Control Language (JCL) used in larger computer systems, it is less cryptic and has far greater capabilities including interactive (optionally formatted) terminal prompting, input validation, printer formatting, and file input/output.

PROCs are typically used to create special user-defined functions by combining execution of DATA/BASIC programs, ENGLISH data retrieval operations, and PROC argument passing.

TERMINAL INDEPENDENT PROCESS HANDLER

The Terminal Independent Process Handler (TIPH) initiates a process on a port without an associated terminal, thus freeing the terminal for user interaction. Any terminal output (such as error messages, logon/off messages) will be placed in a spooler hold file. Although terminal input is not allowed, you may "stack" input in the command stream sent to the TIPH processor. This allows execution of a program which requires operator input and such input is known in advance.

EDITOR

The EDITOR permits on-line interactive modification of any item in the data base. Primarily, the EDITOR is used to create and/or modify DATA/BASIC or PROC programs. The EDITOR enters and updates text processed by RUNOFF. Particularly useful in word processing is the EDITOR's global search and replace capabilities. Performing one-of-a-kind modifications to items in user files is another EDITOR function.

RUNOFF

RUNOFF is a text processing facility offering many special features. Your RUNOFF text is entered and modified with the EDITOR. RUNOFF numbers pages automatically and can print text headings and footnotes.

Another RUNOFF feature is chapter and section numbering. New chapters and/or sections may be added to a document, and the subsequent updated publication, with changes and additions, will be completely renumbered automatically. You may instruct RUNOFF to assemble and print a table of contents and an index based on specified words and phrases. Any changes you make to the text will automatically be reflected in both the table of contents and index.

RUNOFF also performs tabulations, centering, selective left/right justifications, underlining, and boldface printing.

1.4 How to Use the SEQUEL Manuals

This manual is written in modular format with each pair of facing pages presenting a single topic.

The approach taken in this and other SEQUEL manuals differs substantially from the typical reference manual format. Here, each pair of pages discusses an individual topic. Generally the left-hand page is devoted to text, while the right-hand page presents figures referred to by the text. A pair of titles, the first naming the chapter and the second naming the topic, are at the top of each text page. Immediately below these titles is a brief summary of the material covered in the topic.

The advantage of this format will become readily apparent as you begin to use this manual. First of all, the figures referred to in the text are always conveniently in front of you at the point where the reference is made. Secondly, there is a psychological advantage knowing that when a topic is completed and the page is turned, you are done with one idea and are ready for another.

Documentation for the SEQUEL system includes the following manuals:

- SEQUEL Programmer's Reference Manual
- SEQUEL Assembly Language Programming Manual
- SEQUEL System Operation Guide
- Microdata Operator's Quick Guide
- Microdata EDITOR Programming Manual
- Microdata ENGLISH Programming Manual
- Microdata DATA/BASIC Programming Manual
- Microdata PROC Programming Manual
- Microdata SCREENPRO Programming Manual
- Microdata 5750 Terminal Operator's Guide
- SEQUEL 2870PLUS Communications Executive Manual
- SEQUEL HASP Communications Executive Manual

In presenting general command formats and examples throughout this and other SEQUEL manuals, certain conventions apply. Conventions used in presenting general command formats are listed in Figure A, while conventions used in the examples are listed in Figure B.

Marginal change bars will be used in future manuals and supplements for the convenience of present SEQUEL users and will indicate significant additions or changes from prior SEQUEL publications.

Convention	Meaning
UPPER CASE	*Characters printed in upper case are required and must appear exactly as shown.*
lower case	*Characters or words printed in lower case are parameters to be supplied by you (i.e., file-name, item-id, data, etc.).*
{ }	*Braces surrounding a word and/or parameter indicates that the word and/or parameter is optional and may be included or omitted at your option.*
{ }...	*If an ellipses (i.e., three dots) follows the terminating brace, then the enclosed word or parameter may be omitted or repeated an arbitrary number of times.*
item-list*	*An asterisk following an item-list indicates that the list of item-ids may be omitted if supplied by a previous SELECT, SSELECT, GET-LIST, or FORM-LIST statement.*

Figure A. Conventions Used in General Command Formats

Convention	Meaning
TEXT	*Shaded text represents your input.*
TEXT	*Standard text represents output printed by the system.*
TEXT	*Italicized text is used for comments and notes which help explain or describe the example.*
<cr>	*This symbol represents a carriage return.*
<lf>	*This symbol represents a line feed.*
<c>	*This symbol specifies that the following character is a control character generated by pressing the <CTRL> key while typing the character. Also press the <SHIFT> key if the character appears on the upper half of a key top.*
_	*This is the ASCII underline character represented as a backarrow (←) on some terminals.*

Figure B. Conventions Used in Examples

1.5 SEQUEL Computer System Overview

SEQUEL is a complete system of computer hardware, software, and firmware, specifically oriented to implement cost-effective data base management. Data base management systems implemented in SEQUEL afford two major benefits: (1) accurate and timely information to significantly improve the decision-making process, and (2) substantially reduced clerical and administrative effort associated with the collection, storage, and dissemination of information.

SEQUEL is a complete computer system combining both proprietary hardware and software to create an effective tool for on-line data base management. Through the use of microprogramming, Microdata has implemented an on-line transaction processing system. Three major components of the system have been implemented directly in CPU firmware:

- Virtual memory operating system
- Software level architecture
- Terminal input/output routines (terminal I/O controller)

Virtual memory operating systems, long used in larger computer systems, have been impractical for minicomputers due to the large amount of main memory needed for the operating system itself. In SEQUEL, the virtual memory operating system has been directly implemented in firmware (i.e., high-speed read-only memory), which executes many times faster than a comparable system normally implemented in software. With the virtual memory manager directly implemented in read-only memory, only a small amount of main CPU memory is needed to run SEQUEL.

Slightly over 6,000 bytes of main memory must be allocated for the operating system monitor. Everything else (system software and data) is transferred automatically into main memory from virtual memory (i.e., disc) by the virtual memory operating system in a demand-paged environment. The SEQUEL computer system is organized into 512-byte pages (frames) stored on the disc. As a frame is needed for processing, the operating system automatically transfers that frame from the disc unit (virtual memory) to main memory; this concept is illustrated in Figure A. The virtual memory feature of SEQUEL allows you to have access to a programming area not constrained by a main memory, but rather can be as large as the entire available disc storage on the system.

The second feature implemented directly in SEQUEL firmware is the software level architecture of the machine itself. Through microprogramming, Microdata has implemented a machine architecture expressly designed and optimized for data base management. SEQUEL assembly language architecture has very powerful instructions expressly designed for character moves, searches, compares, and all supporting operations to manage variable length fields and records. In addition, this software architecture has a very large field-proven software base written for data base management.

The addition of a Terminal Input/Output Controller comprises the third microprogrammed feature. Microprogrammed onto this controller are input/output routines designed to handle communications between terminals and the CPU. In all minicomputer on-line applications, one of the main problems is managing the I/O from on-line interactive terminals. As these terminals increase in number, the load on the CPU becomes overwhelming; consequently, terminal response time can degrade dramatically. To solve these problems, Microdata developed a microprogrammed controller to directly control communications from and to all on-line terminals connected to the SEQUEL computer system. This means that process execution need not be interrupted to handle a character coming from or going to a terminal. This controller handles all these transactions and only interrupts the software at the completion of a line of data. As a result, a large number of terminals may be connected to the Microdata SEQUEL system without degradation due to terminal I/O.

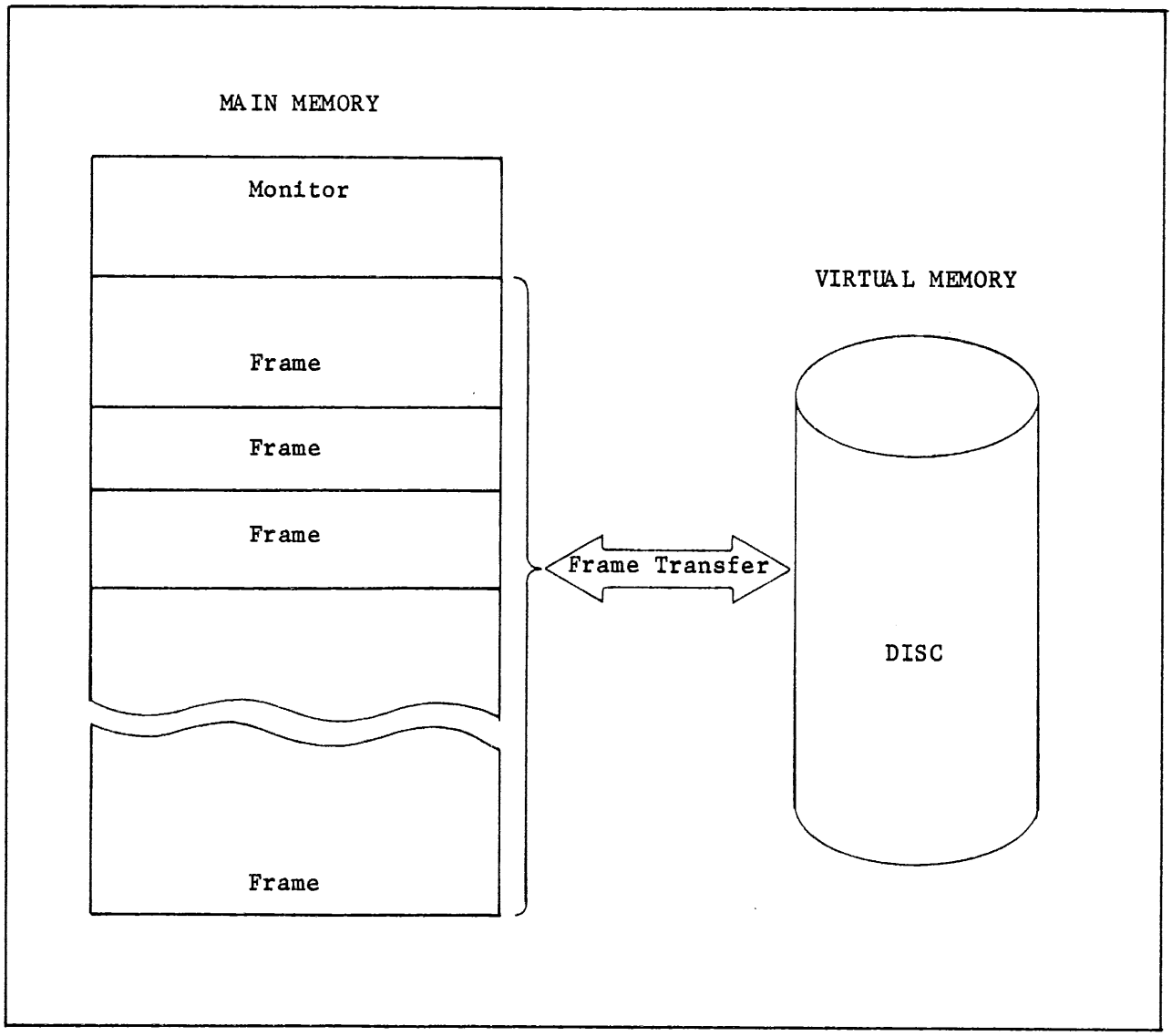


Figure A. SEQUEL Demand-Paged Environment

1.6 SEQUEL File Organization

SEQUEL files are organized in a hierarchical structure, with files at one level pointing to multiple files at a lower level. Four distinct file levels exist: System Dictionary (SYSTEM), User Master Dictionary (M/DICT), Dictionary (DICT) Level File and Data File.

The hierarchial SEQUEL file structure is illustrated in Figure A. Each level is described briefly below.

SYSTEM Dictionary (SYSTEM)

The highest level file on each system is the System Dictionary. This file is used for system control; it contains pointers to each user's Master Dictionary and to various system support files.

User Master Dictionaries

Master Dictionaries (M/DICTs) comprise the next level. Each user's account has an M/DICT associated with it; the M/DICT defines all user vocabulary, all accessible file-names and attributes describing the structure of the information in lower level dictionaries. File-name pointers can reference any file or dictionary in the system, subject to security restrictions.

Dictionary Level Files

Dictionary Level Files describe the structure of data in associated data files.

Data Files

Data Files contain the actual data stored in variable-length format. In addition to the normal record/field data structure, a field (called an attribute) can contain multiple values, and a value, in turn, can consist of multiple subvalues. Thus, data may be stored in a three-dimensional variable-length format.

Internal Dictionary/File Structure

The term "file" in the SEQUEL system refers to a mechanism for maintaining a set of like items logically together. The data in a file is normally accessed via the dictionary associated with it. Since the dictionary itself is also a file, it contains items just as a data file does. The items in a dictionary serve to define lower level dictionaries or data files. Several types of items are used in dictionaries:

- File definition items: define files
- File synonym definition items: point to files
- Attribute definition items: describe fields

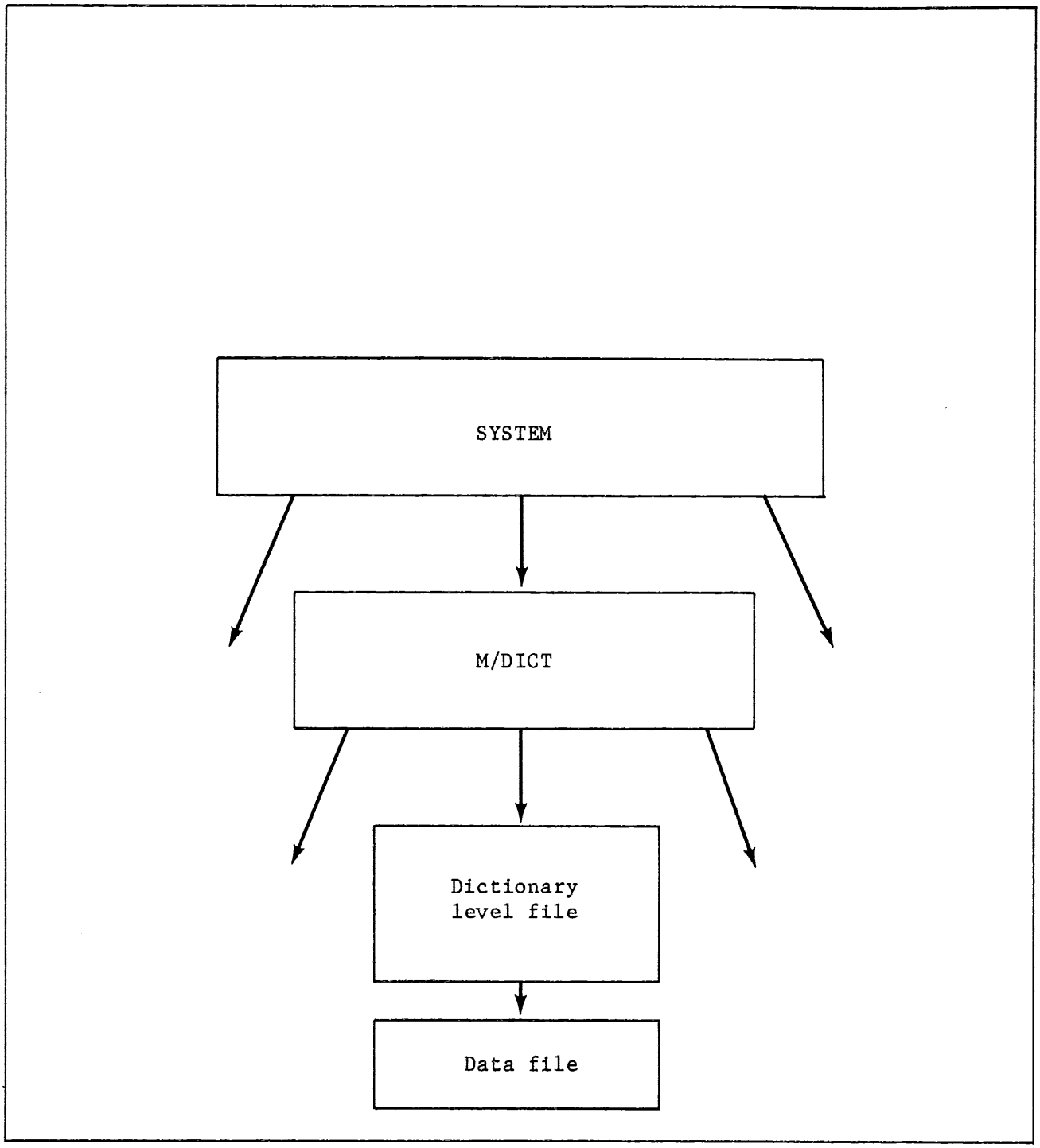


Figure A. SEQUEL Hierarchial File Structure

2.1 Virtual Memory Structure

SEQUEL's virtual memory structure allows disc space to be immediately available as main memory. The entire set of data associated with a SEQUEL system (including executable programs, process work space, and all system and user files) reside in the virtual memory. The monitor remains resident in main memory.

Prior to understanding the data structures used in the SEQUEL system, you must have a general understanding of the virtual memory structure. The following brief overview of that structure should supply sufficient background information.

The virtual memory consists of a set of 512-byte frames, each addressable by a positive integer called the Frame-ID (FID). The virtual memory contains absolute, work, and file areas, as illustrated in Figure A. Frames move between disc and main memory as required by the system.

ABS Area

Starting with the first frame (FID=1) and continuing upward is the Absolute area (called ABS). In this area, a minimum of 1215 frames, but not more than 4096 frames, must be reserved. Frames 0 through 399 and 1000-1279 of the ABS Area are reserved for current and future SEQUEL software. The remaining frames of this area (400 through 999) are available for user-written assembly language programs.

Primary Work Space

You interact with the SEQUEL system via a terminal attached to a communications port on the SEQUEL CPU. The ongoing dialogue with any port is called a process. (Additionally, certain processes not actually connected with a communications port may be specified at system generation time.) Uniquely associated with each process is a Primary Control Block (PCB) which is a one-frame block that defines the state of the process at any instant. Each PCB is followed by a 31-frame work space used by the process. Thus, 32 frames are initially reserved for each defined process.

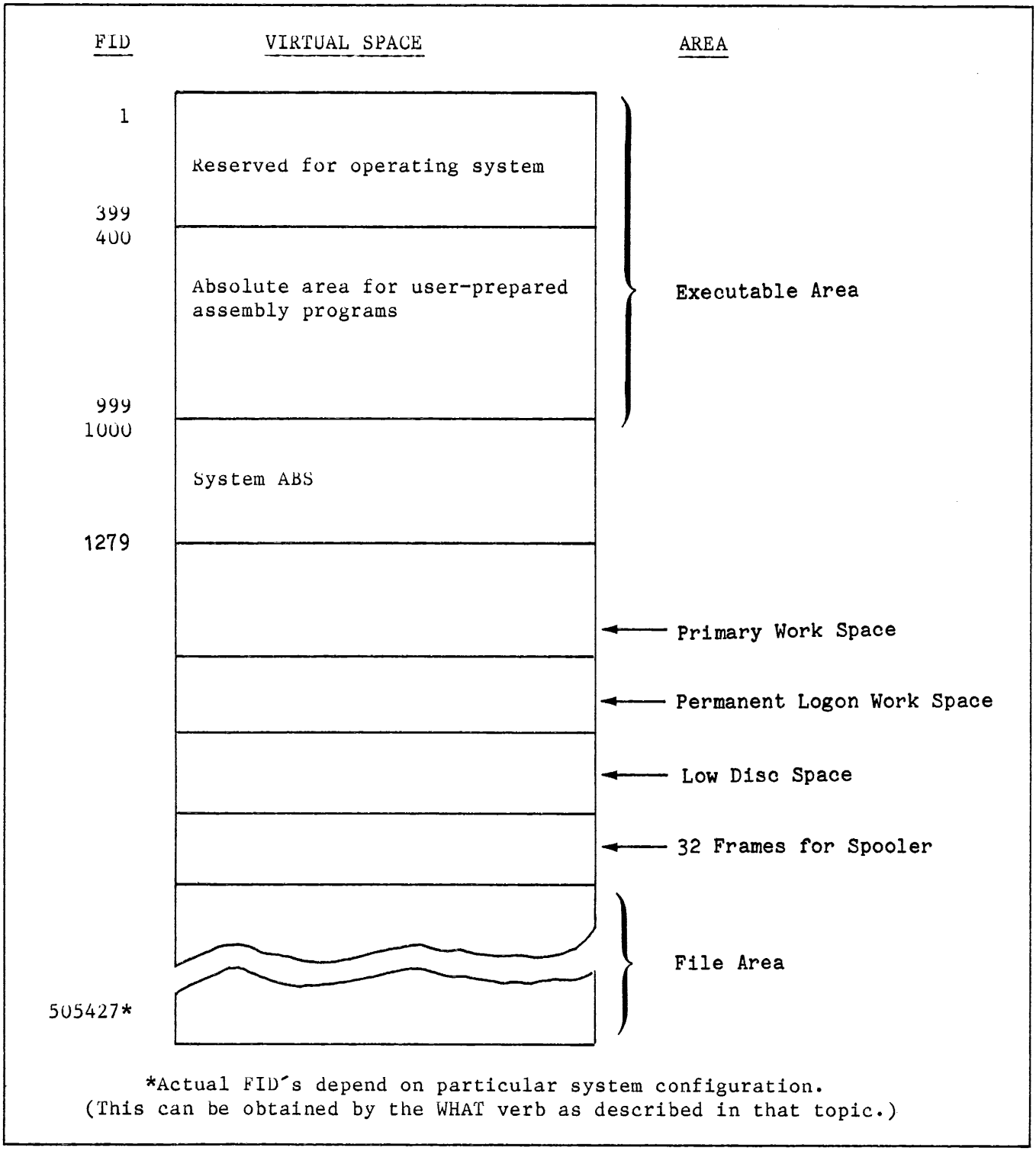


Figure A. Overview of Virtual Memory Structure

2.2 Low Disc Space Area and File Area

Beginning immediately after the primary work space is the low disc space area. This area is used primarily for additional process work space. After the low disc space, the remainder of the virtual memory (called the file area) is available for storage of data in files. If the main file area is full, the low disc space area is used for data storage. Figure A illustrates these areas.

Low Disc Space

During a file restore, a pool of frames (called low disc space area) is allocated for additional LOGON work space. Low disc space consists of 381 frames for each process (line) as specified by hardware.

The LOGON processor attempts to obtain additional work space frames from low disc space; if no low disc space is available, the LOGON processor attempts to obtain additional work space from the high disc space. Additional work space assignment is detailed in Chapter 6.

High Disc Space (File Area)

Immediately following Low Disc Space is an area called High Disc Space. Most processors, such as SORT, spooler and file processors, attempt to get space from this area; if no high space is available, an attempt is made to obtain low space.

The effect of this allocation schedule is to keep additional process work spaces close to each other and to the 32-frame primary work spaces they are linked to. Because these usually constitute the most active frames on the system, disc head movement is minimized and overall system throughput is increased.

Frames which are not in use at a given time are maintained as a pool of available space. Management of available space is discussed in the following topic.

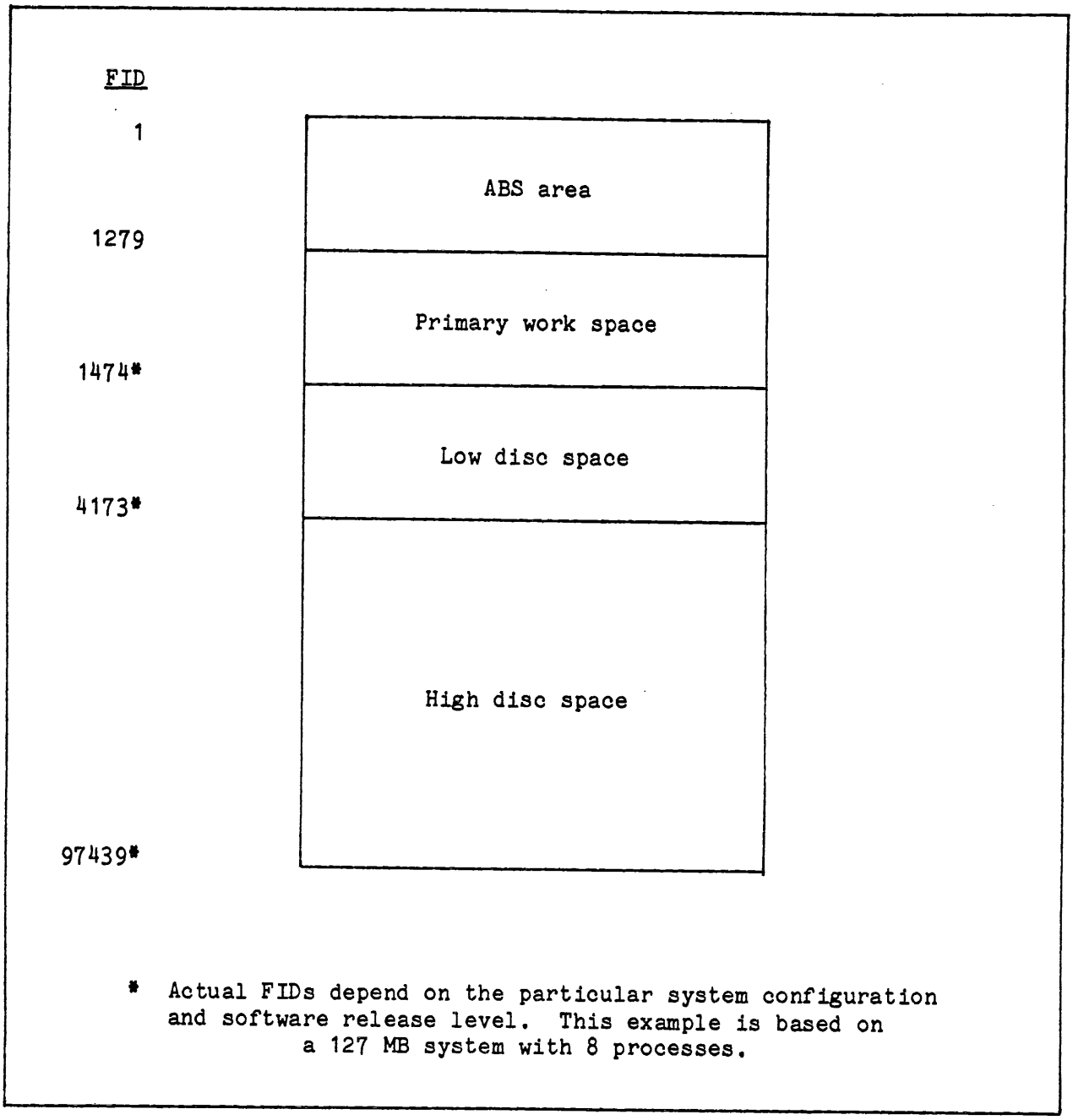


Figure A. Location of Low Disc Space and File Areas

2.3 Available Space Area

File area frames (from high and low disc space) which are not allocated to the files are maintained as a pool of available space.

The SEQUEL system uses available space for two purposes: 1) for additional file space and 2) for temporary work space used by such processors as the spooler, PROC, and ENGLISH. Available space may be either in a "linked" form or in a "contiguous" form. Contiguous available space, as the name implies, consists of blocks of contiguous frames (defined by starting and ending numbers) that can be taken out of the pool either singly or as a block. Linked available space can only be taken a frame at a time. Conversely, space may be released by processors to the linked available pool a frame at a time or to the contiguous pool as a block.

The SEQUEL system maintains four tables of pointers that define the available space. Two tables define contiguous blocks of space, one for file space and the other for temporary work space. The other two tables define linked chains, one for file space and the other for temporary space.

Additional File Space

The table defining contiguous file space may have up to 32 entries. When a block of contiguous file space is requested, this table is searched for a block of space equal to the amount requested, or the next largest block of space. The purpose is to allocate the smallest block that fits the size of the requested space.

When a linked frame is requested, the frame is taken from the linked chain of file space. If there is no linked chain, the frame is taken from a contiguous block of file space. When there is no contiguous file space, the linked chain of temporary space is searched. If still unsuccessful, the frame is taken from the contiguous block of temporary space.

File space (both linked and contiguous) is returned to the two file space tables. The system attempts to match the space up with an existing block in one of the tables. If the attempt fails, the space is returned to the linked chain of temporary space.

Temporary Work Space

Requests for temporary space are made in the following order: 1) the temporary linked chain, 2) the temporary block of contiguous space, 3) the contiguous block of file space (if successful, this space is placed in the temporary block of contiguous space), and 4) the linked chain of file space.

The system attempts to give the space back to the temporary block of contiguous space and to the contiguous block of file space. If the attempt fails, the frame is added to the linked chain of temporary space.

At the conclusion of the file-restore process on the SEQUEL system, an initial condition may be said to exist; there will be one principal block of available high disc space, extending from the end of the current data space through the last available data frame. This is illustrated in Figure A; the shaded area represents the used portion of the file area. As the system obtains and releases available space, the available space may get fragmented; at any particular time there may be several blocks of contiguous available space and a chain of linked frames. (The low disc space area is not generally subject to fragmentation because its space is usually obtained and released in large blocks as processes log on and off.) This is illustrated in Figure B; here the linked available chain starts at FID 8000 and contains 400 frames. Linked available space may be reorganized into contiguous available space via the CLAIM verb (refer to Chapter 9). The POVF verb may be used to display the available space.

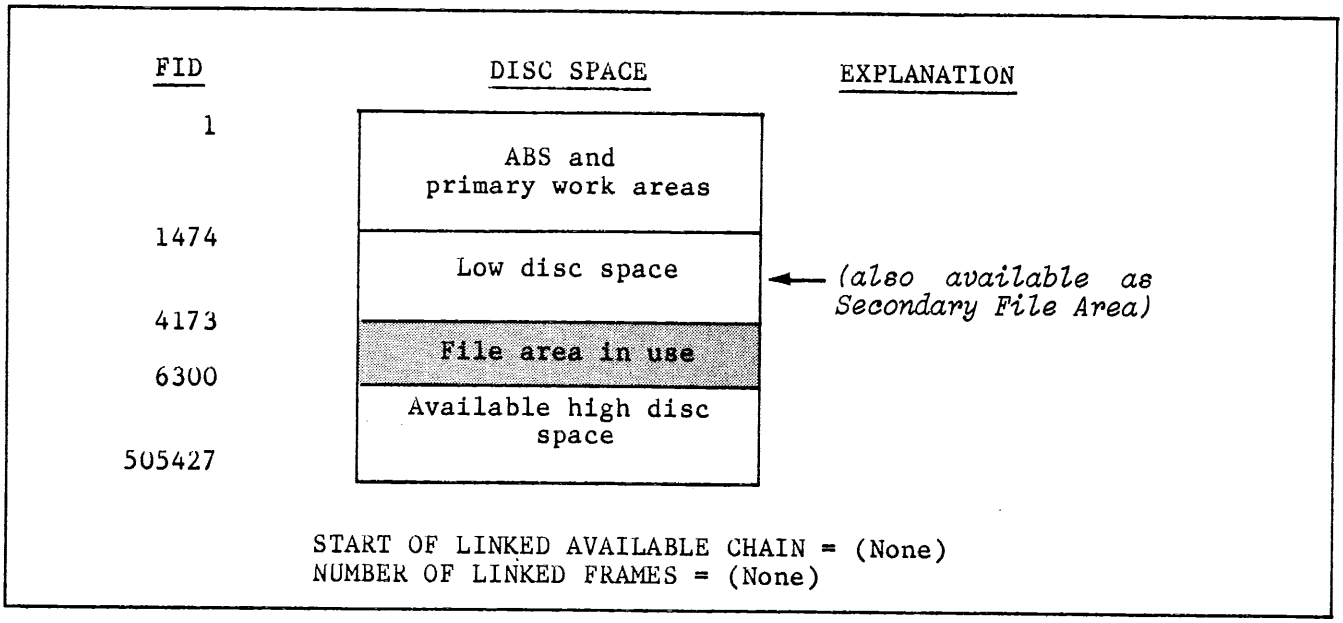


Figure A. Example of Contiguous Available Space After a File-Restore

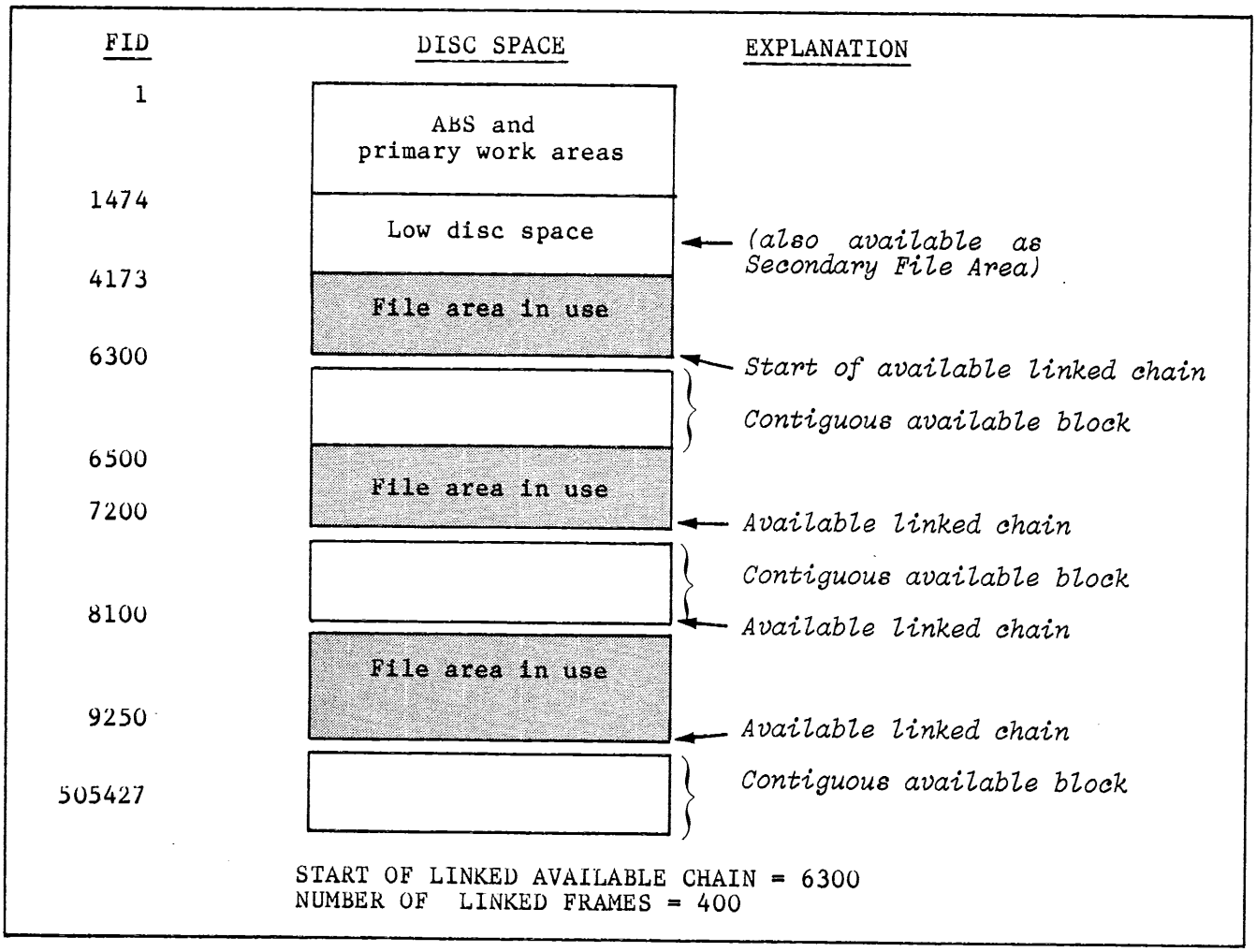


Figure B. Example of Fragmented Available Space

2.4 File Structure

A file is a related group of items which can be accessed for retrieval and update operations by referencing the item-id.

To understand the file structure, the following terms must be defined:

Bit	A digit in binary notation (either 1 or 0).
Byte	A unit of information consisting of 8 bits. Each character stored represents one byte.
Item	Synonymous to record. Usually made up of various attributes (fields) that describe an entity. An example is an inventory item with description and quantity fields. Maximum size is 32,267 bytes.
Item-id	A unique datum (key) within a file item by which all of the data in the item is identified or referenced, such as a part number. Maximum size is 50 characters. May not contain multivalue or subvalue marks.
Attribute	A field within an item.
Multivalues	One of several values in the same attribute (field).
Subvalues	One of several values within a multivalue within an attribute.
File	A set of like items brought together for a like purpose.
Group	A set of linked frames (a frame = 512 bytes) where items may be sequentially stored. A group consists of one or more linked frames and can vary in size from file to file and within files.
Base	The FID (frame identification) of the first frame of the first group in a given file.
Modulo	The number of groups allocated for a given file.
Separation	The number of frames allocated for each group in a file.

A set of items, each with a unique item-id, is stored in a file. A file is defined by a base, modulo and separation. This structure is illustrated in Figure A.

You select the file modulo and separation when the file is initially created. Items are created and modified by the EDITOR, PROC, DATA/BASIC or other processors. Data is stored in items in terms of attributes, multivalues, or subvalues.

A hashing algorithm is used which operates on the item-id (using base, modulo, and separation) to produce an index to the FID of the first frame of the group in which the item is stored. For further information regarding this hashing algorithm, refer to the topic "Item Storage and Hashing Algorithm" in this chapter.

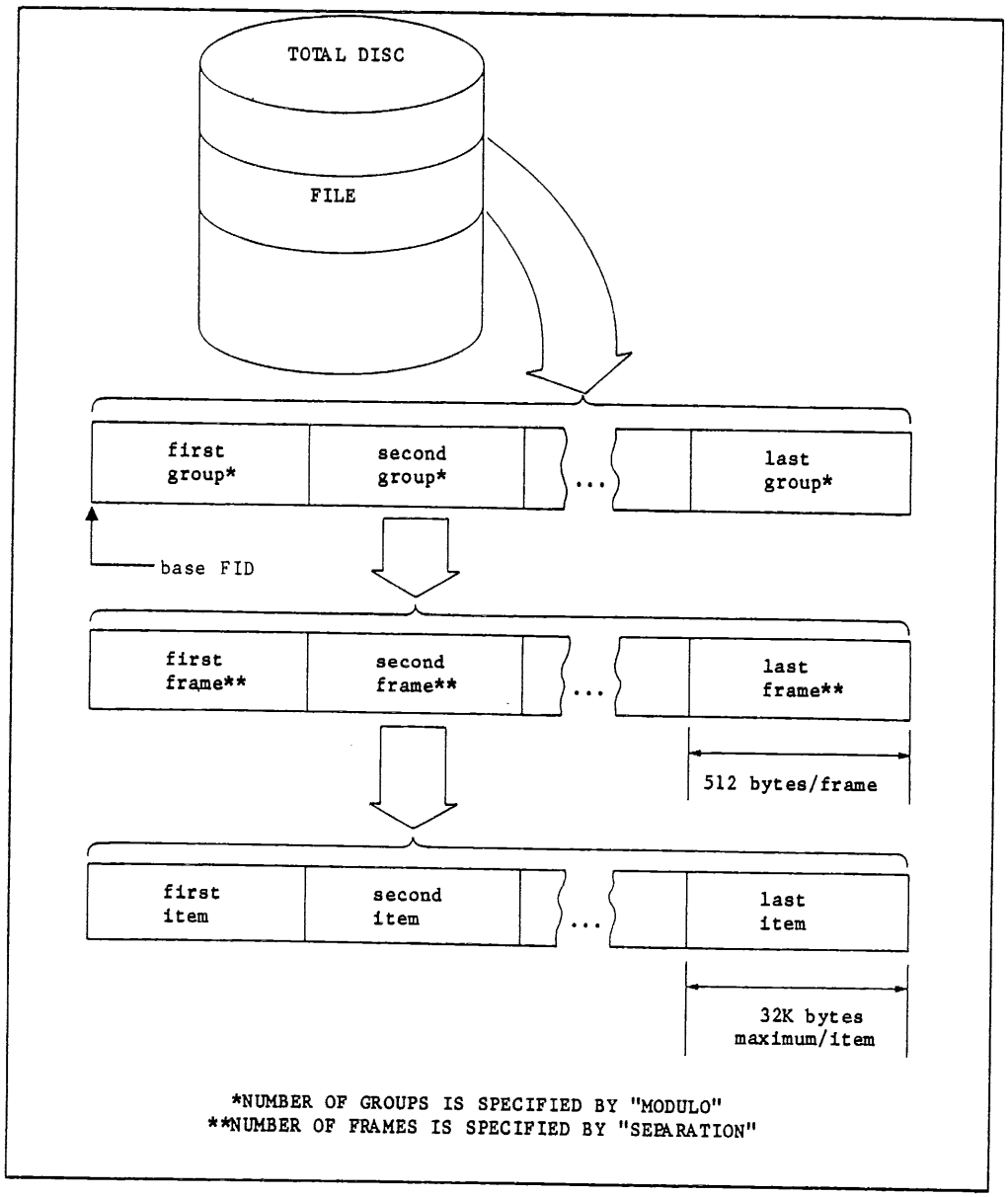


Figure A. General File Structure

2.5 Item Structure (Physical)

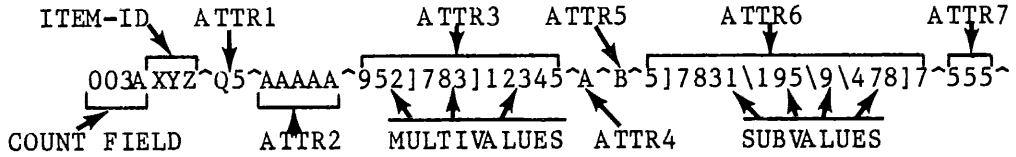
Data within an item is stored as attributes (i.e., fields), multivalues, and subvalues, all of which provide variable length storage. This topic describes the physical item format.

An item consists of one or more variable-length attributes (also known as fields) separated by attribute marks. An attribute mark is an eight-bit byte with a value of X'FE', which prints as an up-arrow or "^". An item is preceded by a four-character hexadecimal count which specifies the total number of characters in the item including the count field, the item-id, and an attribute mark to terminate the item. For example, consider the following item:

```
002EITEMX^LINE 1^SMITH, JOHN^1234 MAIN STREET^
```

In this example, the item-id is "ITEMX". It is preceded by "002E" which specifies that there are 002E (46) bytes in the item. Attribute 1 is "LINE 1". Attribute 2 is "SMITH, JOHN". The last attribute (attribute 3) is "1234 MAIN STREET".

An attribute, in turn, may consist of any number of variable-length multivalues separated by value marks. A value mark has an eight-bit value of X'FD', which usually prints as "]". Finally, a multivalue may consist of any number of variable-length subvalues (also known as secondary values) separated by subvalue marks. A subvalue mark has an eight-bit value of X'FC', which usually prints as "\". For example, consider the following item:



The absence of an attribute value is specified by an attribute mark (to maintain the proper attribute sequence) immediately following the attribute mark indicating the end of the previous attribute. The "space" between two adjacent attribute marks can be thought of as representing the absent or null attribute.

Note: An item-id must be less than 50 characters and the total size must not exceed 32,267 characters.

Figure A illustrates the general form of an item.

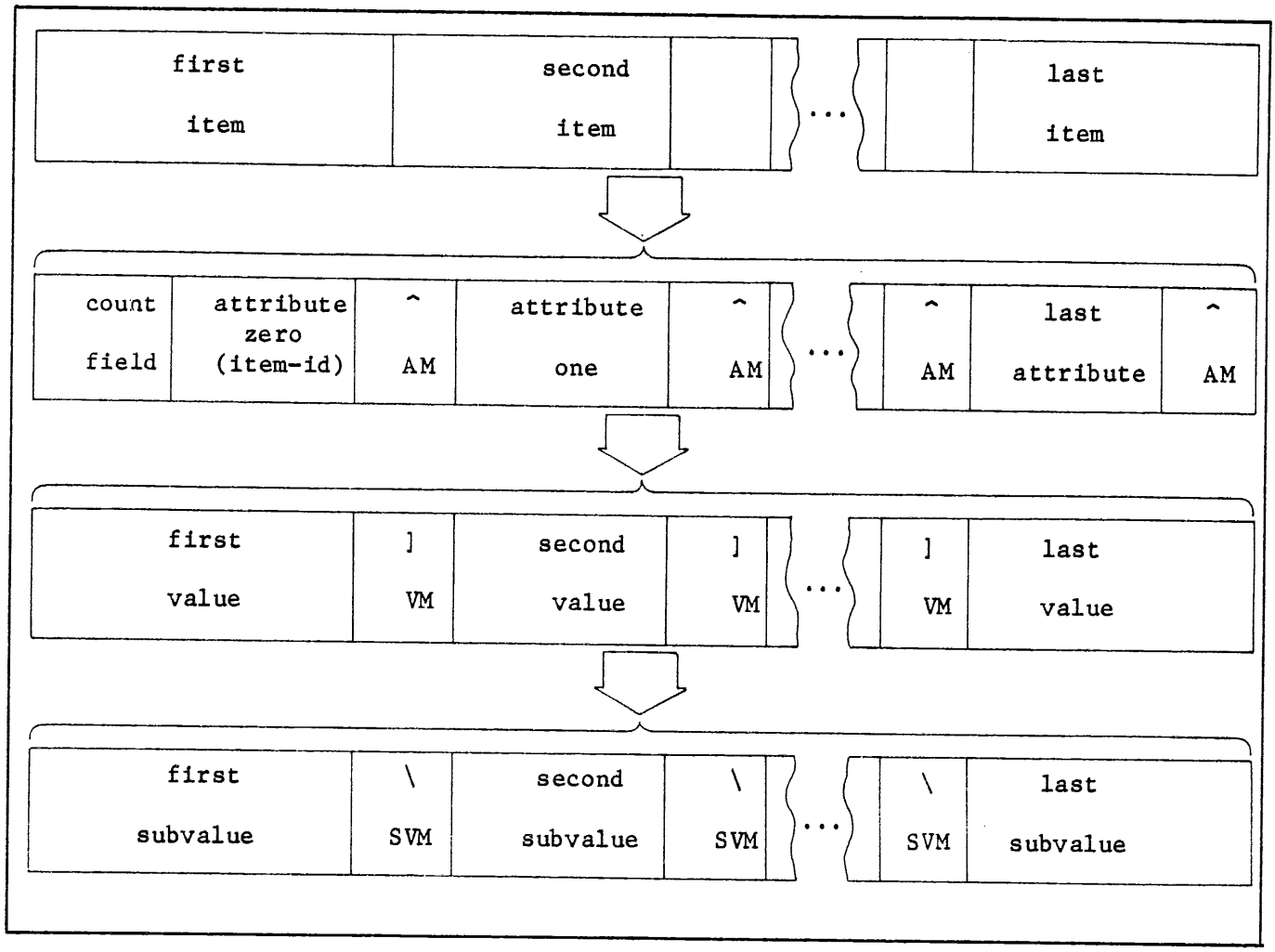


Figure A. General File Item Structure

2.6 Item Structure (Logical)

This topic describes the SEQUEL item structure at the logical level.

While it is important to understand the physical item format, in normal system usage items are always accessed at a more abstract level. Files are identified by a file-name. Within a file, items are referenced by the item-id. Attributes are referred to as lines (i.e., Attribute 1 is called line 1). Figure A shows a sample COPY operation where the item with item-id ITEMX (in the file SAMPLE-FILE) is being copied to the terminal. The item is shown to have three attributes (lines) each with sample data.

Utility processors like COPY and the EDITOR deal at the file-item-line level. They make no logical distinction in definition between various lines in an item, other than their implied line numbers.

ENGLISH processors, however, add an additional dimension through the use of the dictionary. The dictionary informs ENGLISH as to the nature of information stored for each of the attributes.

The logical item format is identical for ENGLISH and non-ENGLISH processors. It is your responsibility to ascertain further qualifications, if any, of the various attributes. For example, the item listing in Figure A, is shown in Figure B, as produced by the ENGLISH LIST processor. Here, for example, the file dictionary defines Attribute 2 (line 2) as NAME and Attribute 3 (line 3) as ADDRESS. This permits you to reference data symbolically when, in fact, the actual data stored on file is the same regardless of the processor accessing it. (For further information, refer to the MICRODATA ENGLISH Programming Manual.)

```
FID = BASE + [Remainder ((Function of item-id)/MODULO)] * SEPARATION
```

Figure A. Hashing Formula

```
                                forward link                         hex link
FID:     38186  :     0 38197        0     0   (  952A  :  0  9535    0    0 )

000 .....5.......0011T-ATT^P^80A8^001D*A3^A^03^^^^^^T10
050 ^L^10^1^0013SP-STOP^P^10A6^001DSAMC^S^04^S/AMC^^^^
100 ^^L^2^000CSUPP^CH^000BNOT^CD^002ALISTPROCS^PQ^(DIC
150 T PROCLIB LISTPROCS)^001DV/CORR^A^08^^^^^^^L^10^1^
200 0018T-RDLBL^P^6033^8024^000AEQ^CC^001E9^S^09^V/TYP
250 ^09^^^^^L^2^1^0016BLOCK-TERM^P^1041^0026ERRMSG^Q^S
300 YSPROG^ERRMSG^1^^^^^R^5^0016ITEM^P^2^30A0^^NP^0013
350 CHARGES^P^5032^0017C-READ^P^2^94^^CUN^001DV/EDIT^A
400 ^12^^^^^^^L^10^1^0011SUM^PW^35^4F^001ASSELECT^PB^3
450 5^76^3076^001E2^S^02^A/AMC^02^^^^^L^2^1^0029PE^PQ^
500 HPRINT-ERR E


FID:     38197  :    0     0 38186    0         (  9535  :  0     0    952A   0  )

000 .........*..RRMSG ^F^A(,999)^P^000EBEFORE^CB^^_LW.
050 P2..p3.A..G.Q..]p..L.....Qp...p...0..60..NO..f0.p.
100 F6..B.%..B.A....Q.p....6.r...N.>P&..G.VB.ZB.U.W.G.
150 W.G.P..&G.FB.JB.Ep............Mw.......q.P....P.^p     group-
200 .p.G.F@DN.IX.P.@DJ:J....l..p.......P..J.G.VF!AOG.t     terminating
250 D....!CAPTQMR!R.w..%GZ...+AN^N_ BNG?^K@@.FBGT..F6      attribute mark
300 BGD.3..G.FP...U........G[ >.X.#BJV6.;BGV@HFANVBmVW
350 Z.QG ?.^6.".U....B.VWZ.5XBmV..Bgb.BjF.ANV666F.._
400 ....F'..F@..FO..F...F...F..&F..'F^...+f.@.OBgVw...
450 ..N:.7NO ...............B5..B1..000000..B5..B1..00
500 00........ S
```

NOTE: The data after the group-terminating attribute mark is random garbage and is ignored by the system.

Figure B. Dump of First Group of Sample File

2.8 File Updates: Structure

A single system routine handles all additions and deletions of items to files. Frames are added to groups as they grow and released as they shrink. Updates to items occur "in place".

Figure A shows a file sparse enough to be contained in its primary space. It has a modulo of seven and a separation of one.

As items are added to the file, new items will be added to the end of some groups (the groups to which these items hash). Also, items within the groups may shrink or grow. The position of items which shrink or grow is not changed, relative to other items in the group. (This is called "update in place.")

As groups grow, frames are taken from overflow and appended to the ends of groups (Figure B). The primary space for the file remains unchanged. The appended frames will be singly linked together, as opposed to being contiguous blocks.

When items are deleted, groups shrink and remaining items maintain their relative order. To make unused secondary frames available for other needs of the system, these frames are released to the system available space pool as soon as they become available.

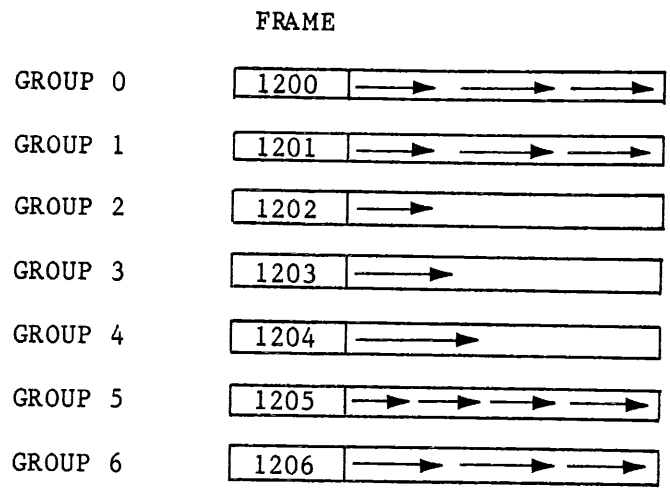


Figure A. File with Only Primary Space

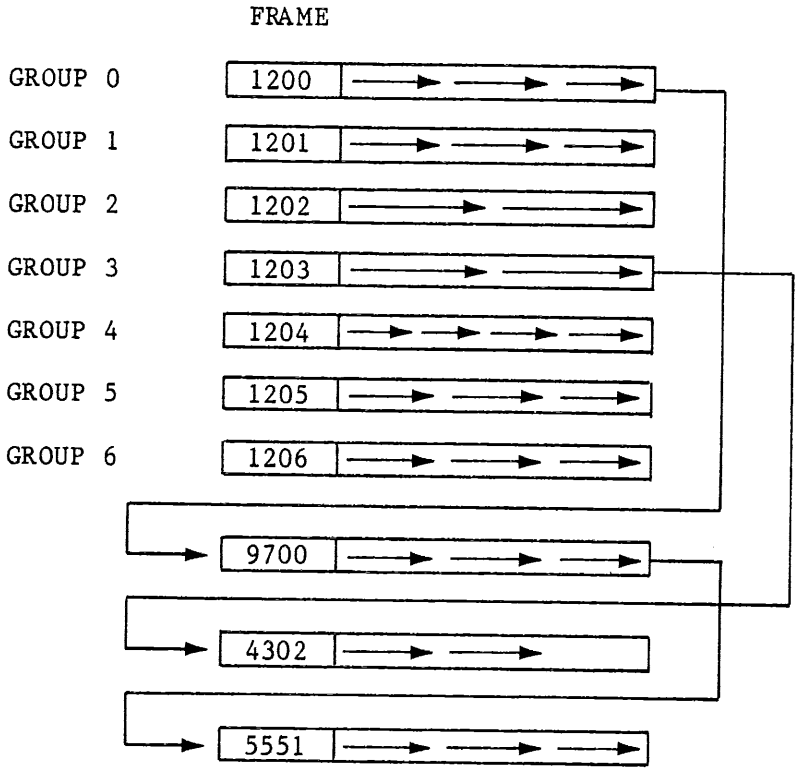


Figure B. File Growth into Secondary Space

2.9 File Updates: File Update Routine

One common routine is used for all file updates on a system.

See Figure A for file update routine.

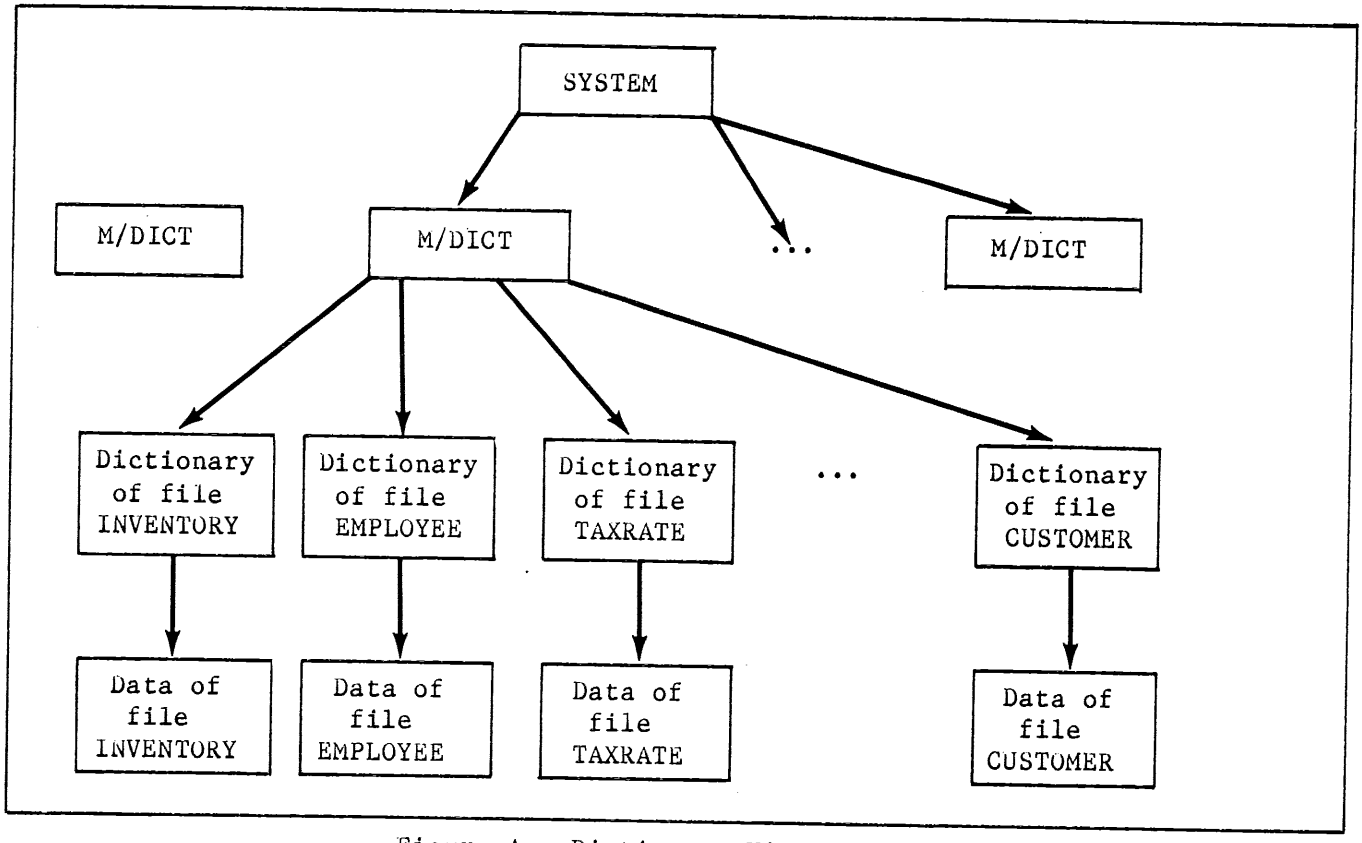


Figure A. Dictionary Hierarchy

D/CODE (Attribute 1)	TYPE OF DICTIONARY ITEM
D	File definition item.
DX	File definition item; the X indicates that the file will be skipped by the file-save processor.
DY	File definition item; the Y indicates that it is not required to save the data portion of the file on a file-save.
Q	File synonym definition item.
A, S, or X	Attribute definition item.

Figure B. Dictionary Items

3.2 File Definition Items

File definition items define lower level dictionary files or data files. File definition items are specified by a D/CODE of "D", "DX", or "DY". They are created automatically by the CREATE-FILE verb.

At the SYSTEM dictionary level, file definition items define various system files (see the topic INITIAL SYSTEM FILES/DICTIONARIES), each user's M/DICT, and an item containing the logon message. File definition items in the M/DICT define the dictionary level files which in turn may contain a file definition item which defines the associated data area. The item-id and each attribute of the file definition item contains the information used to describe the lower level dictionary or file:

Item-id - The item-id of a file definition item is the file-name of the dictionary or data file being pointed to.

Attribute 1 - This is the D/CODE attribute; it must contain a "D", "DX", or "DY". See description of each in previous topic.

Attribute 2 - This is the F/BASE attribute; it must contain the Base FID (as a decimal number) of the defined file.

Attribute 3 - This is the F/MOD attribute; it must contain the modulo (as a decimal number) of the defined file.

Attribute 4 - This is the F/SEP attribute; it must contain the separation (as a decimal number) of the defined file.

Attribute 5 - This is the L/RET attribute; it may contain an optional retrieval security code (see Chapter 6, the topic titled SECURITY).

Attribute 6 - This is the L/UPD attribute; it may contain an optional update security code (see the topic in Chapter 6).

Attributes 7 through 8 - These attributes are identical to those used in attribute definition items; refer to the topic, ATTRIBUTE DEFINITION ITEMS, in this chapter.

Attribute 9 - This is the V/TYP attribute which indicates the type of justification for the column of item-ids printed by ENGLISH. A value is mandatory and may be one of the following:

L - Left justified
R - Right justified
U - In user identification item in System Dictionary, indicates update accounting file

Attribute 10 - This is the V/MAX attribute which indicates the width of the item-id column printed by ENGLISH. A numeric value is mandatory.

Attributes 11 through 12 - Reserved.

Attribute 13 - This is the F/REALLOC attribute, which allows for the reallocation of the physical extent of a file during a system file-restore process (see Chapter 4). The format of this specification is (m,s); where "m" and "s" are decimal numerics specifying the new modulo and new separation parameters of the file. Another way the physical extent of a file can be altered is to COPY the file to a new file. The first character may be a 'B' indicating to the file save/restore processor that the file is a pointer-file to saved lists of item-ids, cataloged DATA/BASIC programs and shared data items. The 'B' is only valid at the System Dictionary level.

WARNING: Attributes 2, 3, and 4 of a file definition item should never be altered under any circumstances.

Figure A illustrates a sample file definition item which defines the dictionary level file for the INVENTORY file; this item has an item-id of INVENTORY and is stored in the user's M/DICT. The figure also shows the file definition item which defines the data area of the INVENTORY file; this item has the special item-id 'DL/ID' and is stored in the file dictionary.

Note that in a single-level file, the DL/ID item may be absent in the file dictionary, or it may point back to the dictionary itself.

Item 'INVENTORY' in M/DICT	Item 'DL/ID' in dictionary of INVENTORY
001 D	001 D — *D/CODE=D (file definition item).*
002 2537	002 15132 — *Base FID of file.*
003 1	003 13 — *Modulo of file.*
004 1	004 1 — *Separation of file.*
005 /;P123	005 — *File access retrieval lock-code.*
006 UPDATE*LOCK!	006 — *Update lock/code.*
007	007 — *Conversion (null).*
008	008 — *Correlatives (null).*
009 L	009 L — *Left justification.*
010 10	010 10 — *Column width.*
	011 — *(null).*
	012 — *(null).*
	013 (3,1) — *Reallocation parameters.*

Figure A. Sample File Definition Item in M/DICT

3.3 File Synonym Definition Items

File synonym definition items allow access to files in another user's account or provide alternate file-names. File synonym definition items are specified by a D/CODE of "Q".

The item-id and attributes of a file synonym definition item are:

Item-id - The item-id of a file synonym definition item is the synonym name by which the defined file may be referenced.

Attribute 1 - This is the D/CODE attribute; it must contain a "Q".

Attribute 2 - This attribute must contain the name of the account in which the actual file definition is to be found (the account name is an entry in the System Dictionary).

Attribute 3 - This is the S/NAME attribute; it must contain the item-id of the actual file definition item to which the synonym equates (i.e., the actual file-name). If this attribute is null, then it is implied that the synonym file is the M/DICT of the account specified in attribute two.

Attribute 4 - Must be null.

Attribute 5 Through 12 - Not used.

Note that a "Q-pointer" must point directly to a "D pointer"; one "Q-pointer" cannot point to another "Q-pointer".

Figure A illustrates a sample INVENTORY file synonym definition item which allows you access to the file in the account named USER2; you can reference this file via the synonym file-name INV. Figure B illustrates the overall relationship of this plus an additional file synonym definition item. Note that SYNUSER3 points to the M/DICT for USER3 since the value for Attribute 3 is null.

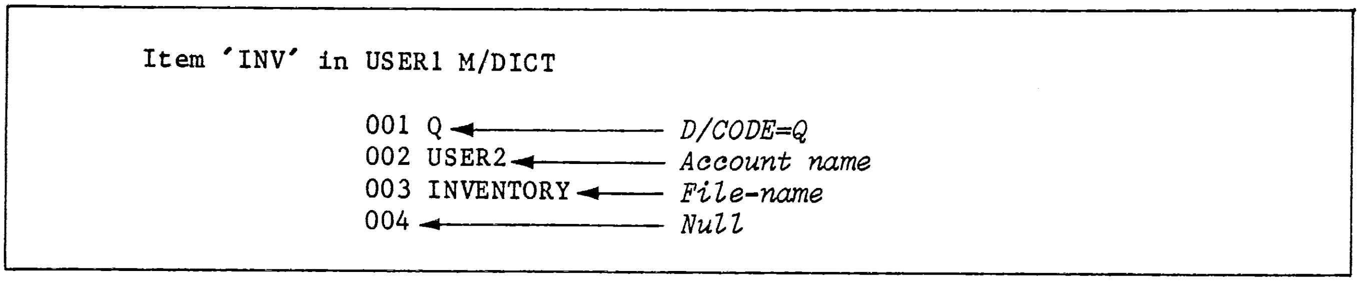


Figure A. Sample File Synonym Definition Item

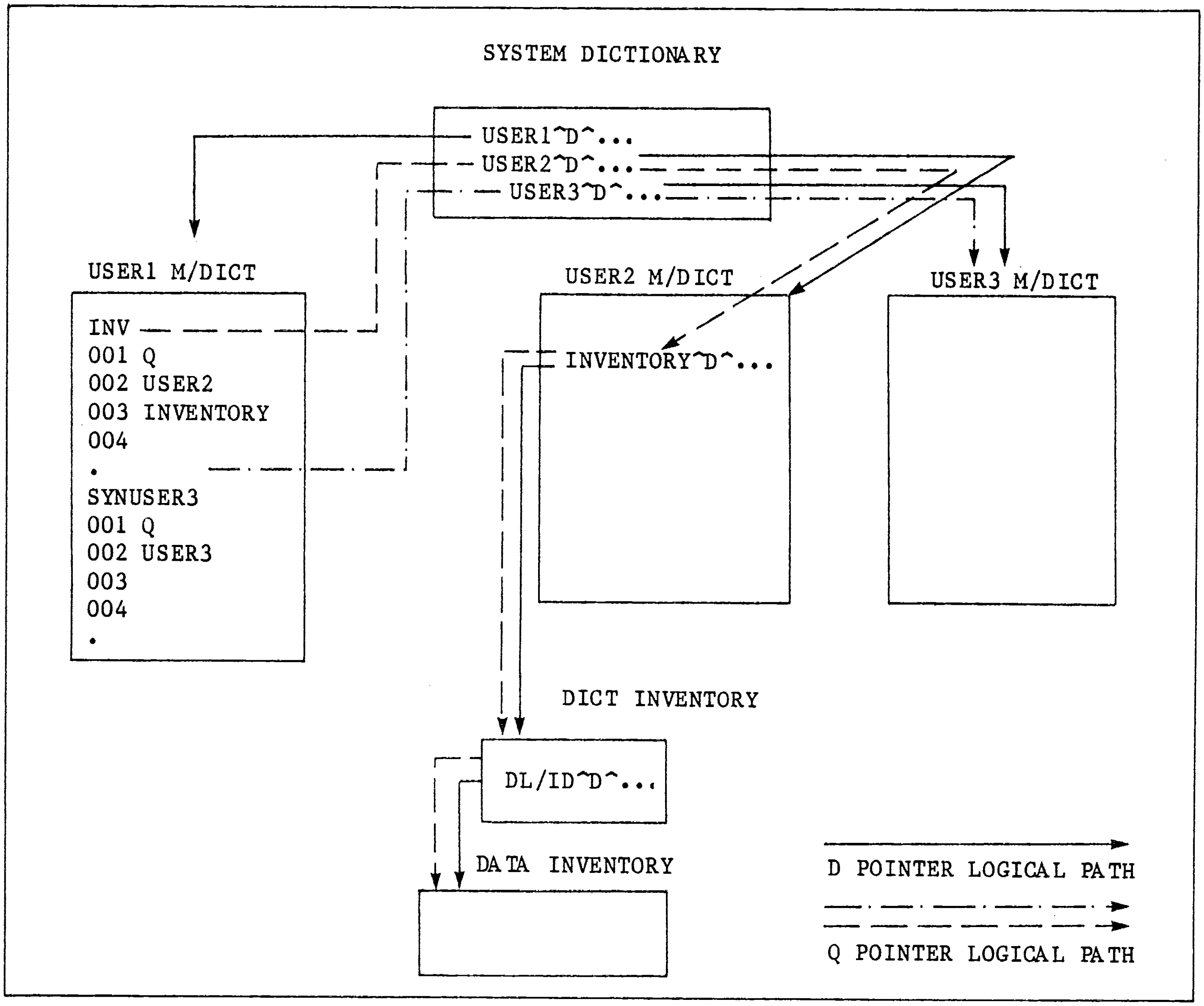


Figure B. Sample File Synonym Definition Interrelationship

3.4 Attribute Definition Items

Attribute definition items define various attributes (fields) in the data items for use by the ENGLISH processor. Attribute definition items are specified by a D/CODE of "A", "X" or "S", and are typically formed with the EDITOR.

An attribute definition item defines a specific attribute (field) for subsequent access via the ENGLISH processor. Each attribute definition item has a value, called the Attribute Mark Count (AMC), which acts as a pointer to the data field defined by it. The AMC is simply the attribute number (e.g., AMC=5 defines Attribute 5, AMC=22 defines attribute 22, etc.). An attribute definition item defines a corresponding attribute for all items in the file and provides a symbolic name for that attribute. Attribute items with numeric and sequential item-ids (1,2,...) are used for the automatic attribute inclusion feature of ENGLISH (when the output-specification is omitted). "A" or "S" codes cause columns to be listed, and act identically with one minor exception, as noted under "Attribute 3" below. "X" codes preserve the chain of numeric item-ids but are otherwise ignored. Attribute definition items are constructed as follows:

Item-id - The item-id is the name desired for the defined attribute. This name may be used in ENGLISH input statements to reference the attribute.

Attribute 1 - This is the D/CODE attribute; it must contain an "A", "S" or "X".

Attribute 2 - This is the A/AMC attribute; it contains the AMC of the defined attribute (i.e., specifies which attribute is being defined).

Attribute 3 - This entry will be used as the column heading. A multiple-line heading can be specified by separating individual lines with value marks (<c>], X'FD'). If this attribute is null and the attribute definition item is an "A" type, then the column heading will default to the item-id of the attribute definition item. If this attribute is null and the attribute definition item is an "S" type, then there will be no column heading. This is the only distinction between "A" and "S" type attribute definition items!

Attributes 4 through 6 - Reserved.

Attribute 7 - This is the V/CON attribute; it contains the conversion specification used to convert from the processing format to the output format, and in some cases, vice-versa. Refer to the ENGLISH Programming Manual for further details.

Attribute 8 - This is the V/CORR attribute; it contains the correlative specification used to convert from the internal format to the processing format (refer to the ENGLISH Programming Manual).

Attribute 9 - This is the V/TYP attribute; it defines the justification for output. A value is mandatory and may be one of the following:

I - Left justified, wrapped text is indented one space
L - Left justified, no specified type
R - Right justified, no specific type
T - Text data, left justified. Fold at blanks.
U - Left justified, print entire value on line, ignoring column boundaries.

Attribute 10 - This is the V/MAX attribute; it defines the column width in which values for the attribute will be printed. An entry is a decimal numeric, and is mandatory. Note, however, that if this attribute is less than the width of the column heading specified in Attribute 3, the width of the column heading will override the width specified here.

Attributes 11 through 20 - Used by SCREENPRO.

Figure A illustrates a sample attribute definition item which defines the third attribute of each item in the inventory file. Figure B shows a sample attribute item which defines the fifth attribute of each item in the employee file.

Item 'QUANTITY' in dictionary of file INVENTORY

```
001 A  <-------  D/CODE=A
002 3  <-------  AMC (defines 3rd attribute)
003 QTY<-------  Column heading
004
005
006
007 MD2 <------  Conversion specification
008 G*1 <------  Correlative specification
009 R   <------  Right justified
010 10  <------  Column width
```

Figure A. Sample Attribute Definition Item

Item 'RATE' in dictionary of file EMPLOYEE

```
001 A  <-------  D/CODE=A
002 5  <-------  AMC (defines 5th attribute)
003
004
005
006
007 MD2$<------  Conversion specification
008
009 L  <-------  Left justified
010 5  <-------  Column width
```

Figure B. Sample Attribute Definition Item

3.5 Summary of Dictionary Items

This topic summarizes items used in various SEQUEL dictionaries.

File and Attribute Definition Items

The file definition items, file synonym definition items and attribute definition items which may be used as dictionary entries are summarized in Figure A.

System Dictionary (SYSTEM) Items

There is one and only one System Dictionary for each SEQUEL system. The SYSTEM dictionary should contain only items with D/CODE=D, DX, DY, or Q, representing user accounts or special system files. (The exception is the LOGON message; refer to the topic SYSTEM MESSAGE FORMATS: LOGON MESSAGE.) The Logon processor uses these "D" type items to verify users attempting to logon to the system. Only one "D" type item should be present for each account; if more than one user-name is to be established for the same user-account, additional names should be file synonym definition ("Q" type) items.

Master Dictionary (M/DICT) Items

There is one M/DICT for each account. The M/DICT, like any other file or dictionary, is made up of items. Items with D/CODE=A or S define the attribute formats for all dictionaries including the M/DICT itself; their formats are identical to those for dictionary level files. The file-defining items (D/CODE=D) point to various files defined for the account.

In addition to elements in the M/DICT which define files and attributes, there are items which define verbs, PROCs, and various ENGLISH language elements. Each of these items has a coding structure which uniquely identifies it; refer to the following sources for their respective definitions:

- The chapter titled TERMINAL CONTROL LANGUAGE
- PROC Programming Manual
- ENGLISH Programming Manual

ATTRIBUTE NUMBER	NAME	FILE DEFINITION ITEM	FILE SYNONYM DEFINITION ITEM	ATTRIBUTE DEFINITION ITEM	ATTRIBUTE CONTINUATION DEFINITION ITEM
1	D/CODE	D, DX, or DY	Q	A or S	X
2	F/BASE or A/AMC	Base FID of file	Account-name	AMC	AMC
3	F/MOD or S/NAME	Modulo of file	File-name	Alternate heading	Not used and reserved
4	F/SEP or S/AMC	Separation of file	Must be null	←Not used and reserved→	
5	L/RET	Retrieval lock code(s)	←Not used and reserved→		
6	L/UPD	Update lock code(s)	←Not used and reserved→		
7	V/CONN	Conversion specifica-tion(s)	Not used and reserved	Conversion specification(s)	
8	V/CORR	Correlative specifica-tion(s)	Not used and reserved	Correlative specification(s)	
9	V/TYP	Justification type code	Not used and reserved	Justification type code	
10	V/MAX	Column width	Not used and reserved	Column width	
11	←Not used and reserved→				
12	←Not used and reserved→			Used by SCREEN PROCESSOR	
13	F/REALLOC	Reallocation specification	←Not used and reserved→		
14-20	←Not used and reserved→			Used by SCREENPRO	Not used and reserved

Figure A. Summary of File and Attribute Definition Items

3.6 Initial System Files/Dictionaries

Certain files are essential to the operation and maintenance of the SEQUEL system. This topic describes these files. Figure A illustrates the interrelationship of these files.

SYSPROG Account

The System Programmer (SYSPROG) account is the only account needed to maintain the SEQUEL system. The system message file (ERRMSG) and the prototype M/DICT (NEWAC) are defined in this account. The former is accessed by all users of SEQUEL to obtain error and informative messages, while the latter is used to create new user M/DICT's. Also contained in the SYSPROG account are the system-level PROCs which perform the file-save and file-restore functions, the initialization of the Accounting History file (ACC) on a system setup, etc.

ERRMSG File

This dictionary level file, defined from the SYSPROG account, contains the system messages. Entries in the ERRMSG file are listed in the appendix to this manual; they consist of both error messages and informative messages.

SYSPROG-PL File

This dictionary level file contains the system maintenance PROCs. These PROCs can be used from the SYSPROG account. Refer to Chapter 11, SYSTEM MAINTENANCE PROCEDURES, for a full description of the entries in this account.

NEWAC File

This dictionary is defined from the SYSPROG account, and is a prototype M/DICT used as a model from which a new user's M/DICT is created. The NEWAC file is used by the CREATE-ACCOUNT PROC.

Accounting History File

The Accounting History File (ACC) and dictionary (DICT ACC) contain the accounting history for the system, as well as entries that describe currently active users. The formats of these entries are described in Chapter 6, LOGON/LOGOFF. The Accounting History file should be cleared periodically.

PROCLIB File

The PROCLIB file contains a number of common PROCs (e.g., LISTU, CT, etc.). Each M/DICT contains a Q-pointer to PROCLIB, along with short versions of the common PROCs (using the same PROC names), which merely transfers control to the corresponding PROC in PROCLIB. For further information, refer to the PROC Programming Manual.

BLOCK-CONVERT File

This file contains items used by the BLOCK-TERM and BLOCK-PRINT verbs to convert characters to a block format.

POINTER-FILE

This file contains pointers to saved select lists, cataloged DATA/BASIC programs, etc. Do not modify this file! Attribute 13 of the file definition item in the SYSTEM dictionary must contain a 'B' as the first character.

SYSTEM-OBJECT File

This file contains assembly object code for all Microdata supplied system software. These items are loaded via the MLOAD verb into the ABS region from which the code can be executed. (Refer to the SEQUEL Assembly Language Programming Manual for further details.) Some items in this file are used to create the coldstart (bootstrap) portion of a file-save tape.

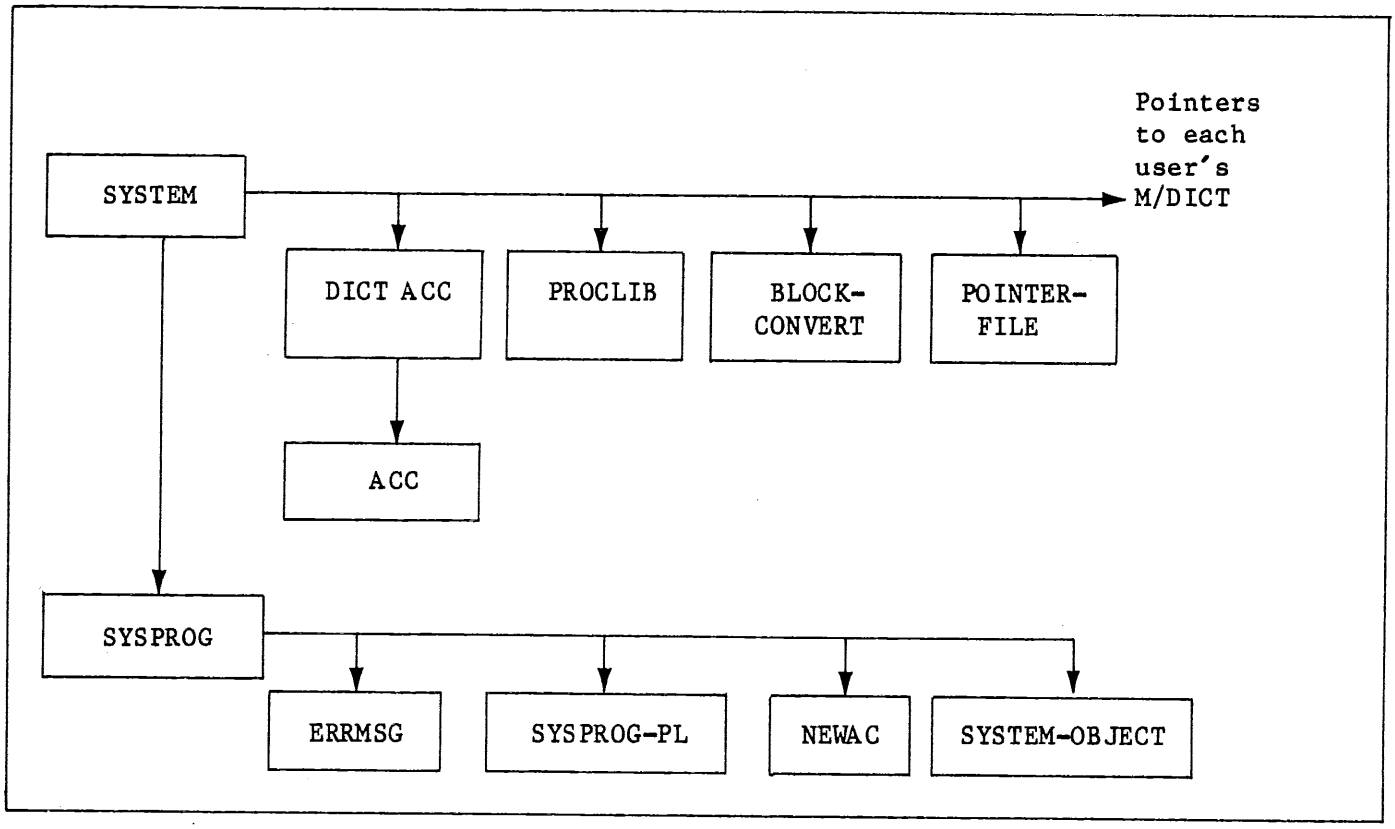


Figure A. General Organization of Initial Files/Dictionaries

4 DIAGNOSTIC MAINTENANCE PROCESSOR (DMP)

4.1 DMP Overview

All SEQUEL systems have a Diagnostic Maintenance Processor (DMP) board. With the DMP, Microdata Customer Service Engineers perform various diagnostic tasks, such as using the Firmware Debugger and executing diagnostic programs. You may use the DMP to: 1) boot the system, and 2) access the system when system software becomes inoperable.

You access the DMP from either Port Zero or a remote port (used for diagnostic testing). If the remote port is used, the REMT button must be pressed. Whatever appears on the remote terminal will also appear on Port Zero's terminal.

You can activate the DMP by powering up the system or by switching the key switch to TEST and typing <c>D on the remote terminal or Port Zero. The DMP first executes diagnostic programs. These tests may take several minutes to execute (see section titled EXECUTING SEQUEL SELF TESTS). When the tests are complete, the DMP displays the System Control Menu (see Figure A).

SEQUEL Control Panel

The SEQUEL Control Panel has switches, lights and the REMT button to select and reflect the operational mode of the DMP and main CPU. The functions of the switches, lights and the REMT button are as follows:

4-Position Key Switch

OFF	All system power is off.
HOLD	Power is supplied to memory only.
RUN	Power is supplied to the entire system. If the switch is turned from OFF to RUN, the System Control Menu is displayed. To start the operating system, you must perform a coldstart ('X' option). If the switch is turned from HOLD to RUN, the system returns to the previous RUN state.
TEST	Power is supplied to the entire system. If the switch is turned from OFF to TEST or from HOLD to TEST, the System Control Menu is displayed. To start the operating system, a coldstart ('X' option) must be performed. When in the TEST position, the DMP firmware responds to the <c>D from the keyboard. If the key is turned from RUN to TEST to RUN, no effects are generated.

REMT button — This is a button that generates an interrupt to the DMP, allowing the firmware to select either the local or the remote terminal for Port Zero.

Lights

PWR	Power is available to the entire system.
RDY	The hardware is ready to run.
BUSY	The SEQUEL bus is busy.
ERR	The DMP self-test is running, or the DMP self-test has encountered a fatal error condition.
TEST	The DMP is in the diagnostic mode.
REMT	The remote console keyboard is selected.

Figure B illustrates the SEQUEL DMP control panel.

```
                * * * SYSTEM CONTROL MENU * * *


SYSTEM FUNCTIONS
     1- RESET SYSTEM
     2- BOOTLOAD
     3- ENTER FIRMWARE DEBUGGER
     4- STOP CPU
     5- RESUME CPU PROCESSING

TERMINAL FUNCTIONS
     6- SWITCH CONTROL TO LOCAL/REMOTE TERMINAL
     7- CHANGE BAUD RATE

DIAGNOSTIC FUNCTIONS
     8- RUN CPU FIRMWARE DIAGNOSTICS
     9- RUN DMP FIRMWARE DIAGNOSTICS
    10- DISPLAY/ALTER MEMORY

ENTER OPTION (1-10) :
```

Figure A. SEQUEL System Control Menu

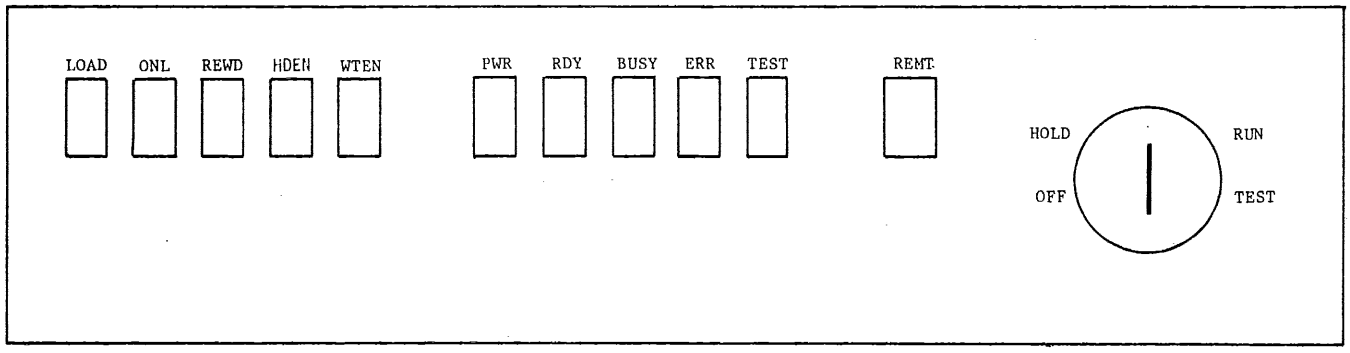


Figure B. SEQUEL DMP Control Panel

4.2 Powering Up a SEQUEL System

If your SEQUEL system is powered down via the key switch or due to a power failure, you must power up the CPU. The system begins by running diagnostic tests. During the course of the tests, the disc units are activated. Should the tests encounter no errors, main memory is still intact, and the key switch is in the RUN position, the system will continue processing where it left off. If main memory is lost or if the key switch is in the TEST position, the system displays the message "POWER FAIL, COLDSTART REQUIRED".

There are three ways to power-up your SEQUEL system. Where the key switch is positioned prior to power-up determines what method you should use. Each method is explained below.

Method #1: (Power-up, memory contents destroyed)

a. Turn 4-position key switch from OFF to RUN, or
   Turn 4-position key switch from OFF to TEST.
b. There is a six second delay to stabilize the power supply.
c. The message "POWER UP" is displayed on the terminal screen (see Figure A).
d. CPU initializes memory to remove memory parity errors caused by the memory power fail. The maximum time required to initialize memory is approximately six seconds. If the CPU does not acknowledge the completion of the initialization task within six seconds, the message "NO CPU ACKNOWLEDGEMENT (B)" is displayed.
e. If the system is started or restarted from the power fail condition, the system time is set to 00:00:00. If the system date is destroyed, you must reset it.
f. If the message "POWER FAIL" appears on the screen, mount a bootload tape on the tape unit (close tape unit door securely) and press <c>D to bring up the System Control Menu. To the message "ENTER OPTION (1-10):", enter "2" (BOOTLOAD) and perform a coldstart ('X' option).

Method #2: (Power-up, Contents of Memory Not Destroyed)

a. Turn 4-position key switch from HOLD to TEST, or
   Power-up with key switch in TEST position.
b. There is a six second delay to stabilize the power supply.
c. The message "POWER UP" is displayed on the terminal screen (see Figure B).
d. Self-tests are executed. See the following section for more detailed information.

Method #3: (Power-up, Contents of Memory Not Destroyed)

a. Turn 4-position key switch from HOLD to RUN, or
   Power-up with key switch in RUN position.
b. There is a six second delay to stabilize the power supply.
c. The message "POWER UP" is displayed on the terminal screen (see Figure B).
d. Self-tests are executed (see following section for detailed explanation).

WARNING: This option should not be used unless the CPU was previously executing a valid program.

```
POWER UP

--RUNNING CPU FIRMWARE DIAGNOSTICS
CPU STATUS 01000000 GOOD STATUS
--RUNNING IOP SELF TEST
--RUNNING DISC SELF TEST
--STARTING DISC MOTORS
--RUNNING ACLC SELF TEST
--RUNNING MAG TAPE SELF TEST
SELF TESTS COMPLETED

POWER FAIL -- COLDSTART REQUIRED

ENTER RETURN TO DISPLAY CONTROL MENU
```

Figure A. Sample Screen Display of Method #1 (Assuming No Error Messages)

```
POWER UP

--RUNNING CPU FIRMWARE DIAGNOSTICS
CPU STATUS 01000000 GOOD STATUS
--RUNNING IOP SELF TEST
--RUNNING DISC SELF TEST
--STARTING DISC MOTORS
--RUNNING ACLC SELF TEST
--RUNNING MAG TAPE SELF TEST
SELF TESTS COMPLETED

ENTER RETURN TO DISPLAY CONTROL MENU
```

Figure B. Sample Screen Display of Methods #2 & #3 (Assuming No Error Messages)

4.3 Executing SEQUEL Self Tests

The SEQUEL system automatically performs DMP Firmware and I/O Controller Self Tests. Each test is described below.

DMP Self Tests

DMP Self Tests examine the internal operation of the DMP. When the self tests begin, the ERR light is turned on and the RDY light is turned off. If the tests are completed without errors, the ERR light is turned off and the RDY light is turned on.

Execution of these self tests should take no longer than one second (exception: when powering up, there is a six second delay). Should the ERR light stay on indefinitely, the DMP has encountered an error condition that it cannot report to the terminal.

CPU Firmware Tests

The Firmware Tests follow the execution of the DMP Self Tests. Upon their execution, the message

-- RUNNING CPU FIRMWARE DIAGNOSTICS

is displayed. If the diagnostics are completed without errors, a status message such as

CPU STATUS 01000000 GOOD STATUS

is displayed. However, should errors occur, appropriate error messages are displayed (refer to the section titled DMP SYSTEM CONTROL MENU). Time required to execute these tests is two seconds per memory board.

I/O Controller Self Tests

If the Firmware Tests are executed successfully, the I/O Controller Self Tests are run. Each test is examined below:

IOP Self Test

When executed, the message

--RUNNING IOP SELF TEST

is displayed. There are two IOP controllers. Their device addresses are FB and FC, respectively. FB must always be present; if missing, an error message is displayed. If FC is missing, no error message is printed.

Time required to execute this test is less than one second.

Disc Self Test

When executed, the messages

--RUNNING DISC SELF TEST
--STARTING DISC MOTORS

are displayed.

If a disc drive is not ready, the program waits 12 seconds before starting the next disc drive. When waiting for the disc drives to come ready, the program will wait up to 60 seconds per drive for the status interrupt to occur. Error conditions (except for missing or not-ready drives) are reported.

ACLC Self Tests

When executed, the message

--RUNNING ACLC SELF TEST

is displayed.

Error conditions (except for missing ACLC boards) are reported.

Time required to execute these tests is approximately 20 seconds per ACLC board.

Magnetic Tape Self Test

When executed, the message

--RUNNING MAG TAPE SELF TEST

is displayed.

Error conditions (except for missing boards) are displayed.

Time required to execute this test is approximately two seconds.

SELF TESTS COMPLETED

After the Mag Tape Self Test is completed, the message

--SELF TESTS COMPLETED

is displayed.

There is a 0-5 second delay between the above message and the next message to appear on the screen.

4.4 DMP System Control Menu

The DMP System Control Menu enables you to 1) reset the system, 2) bootload the system, 3) enter the Firmware Debugger, 4) stop the CPU, 5) restart the CPU, 6) toggle the local/remote terminal, 7) change the baud rate for Port Zero, 8) execute CPU firmware diagnostics, 9) execute DMP firmware diagnostics, and 10) display/alter the contents of main memory.

To display the System Control Menu, turn the key switch to TEST and enter a <c>D. Options One and Two are defined below. Options Three through Ten are described briefly in the following section.

Option 1 - RESTART SYSTEM

This option resets the CPU and all controllers. This means that all current I/O operations are destroyed, and you must do a coldstart.

To execute this option, type in "1" and press the RETURN key. The message "DO YOU WANT A SYSTEM RESET? (Y/N)" will be displayed. To execute a system reset, type in "Y" and RETURN. To abort the system reset, type in "N" and RETURN. The System Control Menu will then be displayed.

WARNING

This option is not comparable to an "IRI" sequence on a REALITY system. Once this option is executed, a warmstart can not be performed; a coldstart is required.

```
            * * *  SYSTEM CONTROL MENU  * * *

      SYSTEM FUNCTIONS
           1- RESET SYSTEM
           2- BOOTLOAD
           3- ENTER FIRMWARE DEBUGGER
           4- STOP CPU
           5- RESUME CPU PROCESSING

      TERMINAL FUNCTIONS
           6- SWITCH CONTROL TO LOCAL/REMOTE TERMINAL
           7- CHANGE BAUD RATE

      DIAGNOSTIC FUNCTIONS
           8- RUN CPU FIRMWARE DIAGNOSTICS
           9- RUN DMP FIRMWARE DIAGNOSTICS
          10- DISPLAY/ALTER MEMORY

      ENTER OPTION (1-10) :1 <cr>

DO YOU WANT A SYSTEM RESET? (Y OR N) Y <cr>
```

Figure A. Sample Dialogue of RESET SYSTEM Option

Option 2 - BOOTLOAD

The bootload is the procedure for loading all or part of the software in order to bring the system to a run state. The bootload loads the software from magnetic tape.

To execute this option:

1. Mount a file-save or binary-save tape on the tape unit.
2. Close the tape unit door securely.
3. Press the LOAD button twice.
4. Press the ONL button once.
5. When the "ENTER OPTION (1-10) :" prompt is displayed, enter "2" and press the RETURN key.
6. Enter the option you have selected.

Before the bootload sequence is initiated, the automatic diagnostics are executed to verify system integrity. The "OPTIONS" message indicates that the bootload is completed and the system is running the monitor. Each of the options listed in the "OPTIONS" message is discussed later in this chapter.

```
                * * *  SYSTEM CONTROL MENU  * * *


          SYSTEM FUNCTIONS
               1- RESET SYSTEM
               2- BOOTLOAD
               3- ENTER FIRMWARE DEBUGGER
               4- STOP CPU
               5- RESUME CPU PROCESSING

          TERMINAL FUNCTIONS
               6- SWITCH CONTROL TO LOCAL/REMOTE TERMINAL
               7- CHANGE BAUD RATE

          DIAGNOSTIC FUNCTIONS
               8- RUN CPU FIRMWARE DIAGNOSTICS
               9- RUN DMP FIRMWARE DIAGNOSTICS
              10- DISPLAY/ALTER MEMORY

          ENTER OPTION (1-10) :2 <cr>
--RUNNING IOP SELF TEST
--RUNNING DISC SELF TEST
--STARTING DISC MOTORS
--RUNNING ACLC SELF TEST
--RUNNING MAG TAPE SELF TEST
SELF TESTS COMPLETED
ENTER BOOTLOAD DEVICE NUMBER (1,2,3,4):
OPTIONS (W,X,A,F,AF,D) =
```

Figure A. BOOTLOAD Dialogue

4.5 DMP System Control Menu - Options 3 - 10

Options Three through Ten of the System Control Menu are defined below.

WARNING: These options are for Customer Service use only. Do not execute without first consulting your Microdata Service Representative.

Option 3 - ENTER FIRMWARE DEBUGGER

This option activates the Firmware Debugger. You may also enter the Firmware Debugger by typing a <c>F from Port Zero's terminal when the key switch is in the TEST position. To leave the Firmware Debugger, enter an "X" at the debugger prompt (">"). Entering an "X" takes you to the system (user) program. To return to the menu, enter <c>D.

If the system entered the Firmware Debugger on its own, or you are not sure how it happened, contact your Microdata Technical Support Representative immediately.

Option 4 - STOP CPU

This option halts the CPU and disables all interrupts except the DMP handshake interrupt to the CPU.

To execute this option, enter "4" and press the RETURN key.

Option 5 - RESUME CPU PROCESSING

This option tells the CPU to continue processing. Interrupts from operations that were on-going when the STOP CPU option was executed will now be allowed to take place.

To execute this option, type in "5" and press the RETURN key.

Option 6 - SWITCH CONTROL TO LOCAL/REMOTE

This option selects either the local (Port Zero) or remote mode to control the DMP.

To execute this option, type in "6" and press the RETURN key. The message "LOCAL OR REMOTE (L OR R)?" will be displayed. If in the local mode, type in "R" and RETURN to switch input/output to the remote terminal. If in the remote mode, type in "L" and RETURN to switch input/output to Port Zero. Note: The baud rate must be set per requirements for the modem/data phone before selecting the remote mode.

Option 7 - CHANGE BAUD RATE

This option enables you to change the baud rate for both the local (Port Zero) and remote terminal.

To execute this option, type in "7" and press the RETURN key. The message:

```
ACCEPTABLE BAUD RATES FOR DMP AND ACLC ARE:
          50, 75, 110, 134.5
          150, 300, 600, 1200
          1800, 2000, 2400, 3600
          4800, 7200, 9600, 19.2K
ENTER ACCEPTABLE BAUD RATE FOR YOUR TERMINAL:
```

is displayed. Type in the desired baud rate and press the RETURN key. Garbled data will appear on the screen. Set the baud rate selector on the terminal to the desired baud rate and press the RETURN key again. The System Control Menu will then be displayed in readable characters.

Option 8 - RUN CPU FIRMWARE DIAGNOSTICS

This option executes a diagnostic program. To execute, type in "8" and RETURN. The message "CPU HALTED" is displayed. After four seconds the following option menu is displayed:

```
CPU FIRMWARE DIAGNOSTIC MENU

          1 - SINGLE RUN, ALL TESTS
          2 - SINGLE RUN, SELECT TESTS
          3 - LOOP ON TESTS, STOP ON ERROR
          4 - LOOP ON TESTS, LOOP ON ERROR

ENTER OPTION (1-4):
```

Type in the desired option and press the RETURN key. These tests destroy CPU registers. As a minimum, a warmstart is required to continue CPU execution; a coldstart is permissible.

Option 9 - RUN DMP FIRMWARE DIAGNOSTICS

To execute this option, type in "9" and RETURN. The screen is cleared and the diagnostic menu is displayed. Type in the desired option number and press the RETURN key.

WARNING: Because some of these tests are destructive to the system, do not execute this option without first consulting with your Microdata Customer Service Engineer.

Option 10 - DISPLAY/ALTER MEMORY

This option enables you to display and/or alter memory locations in the SEQUEL memory. After entering "10", the following information is displayed:

```
DISPLAY/ALTER MEMORY OPTIONS
W = DISPLAY WORD
B = DISPLAY BYTE
SPACE = DISPLAY NEXT LOCATION
CTRL A = DISPLAY CURRENT LOCATION
RETURN = TERMINATE

>
```

The ">" is a prompt for a command. To alter memory, first display the location and then type in the desired new data. To return to the System Control Menu, press the RETURN key three times.

4.6 System Restore: Overview

System restore is the process of "bringing up", or making operational, the SEQUEL system. A SEQUEL system is brought up by doing a file-restore from tape. This procedure transfers system software, dictionaries, and data from tape through main memory onto the discs. It also establishes the monitor in the low portion of main memory. Furthermore, the system configuration (number of terminals, number of discs, etc.) is established at this time.

A system restore is performed using a save tape (or a SYS-GEN tape). A save tape has three parts; each section is terminated by an end-of-file mark (see Figure A).

1. The bootstrap section contains the monitor, the configurator, and some virtual memory frames needed to bootstrap the system.

2. The ABS (absolute) section contains the majority of system software.

3. The files section restores the data base including all accounts, dictionaries, as well as DATA/BASIC object code.

The bootload process is controlled by the Diagnostic Maintenance Processor (DMP). The bootload option on the DMP's Control Menu is used to load the bootstrap section of the tape.

To bring up a system from scratch, place a file-save tape on the magnetic tape unit. Turn the key switch to RUN. The DMP will run the self-tests and then display the Control Menu. To the prompt "ENTER OPTION (1-10) :", type "2" followed by <RETURN> to start the bootload. The self-tests are run again. If there are no errors, the "OPTIONS" message is displayed. Respond with "AF" to the message "OPTIONS (W,X,A,F,AF,D)" for a complete system restore (see Figure B).

When a complete system restore (option 'AF') is performed, the system configuration is determined. When a coldstart ('X'), files only ('F'), or ABS only ('A') restore is performed, most aspects of the configuration must be maintained, although some portion of main memory may be changed within limits. The warmstart ('W') option restarts a system halted during normal operation.

The 'D' option is used by SEQUEL customer engineers for formatting disc platters and verifying check-codes. If this option is inadvertently entered, you may recover by rewinding the tape and restarting the system load.

After you respond to the "OPTIONS" message, the configuration software determines the following:

1. Number of communication lines
2. Amount of main memory
3. Number and size of all disc drives

These are compared with limits set inside the hardware, and in each case the smaller number is selected for the system.

Should you have I/O errors, such as parity errors during the bootload, one of the following messages is displayed:

```
TAPE ERROR, RECORD n;  STATUS = xxxxxxxx
     or
TAPE ERROR
```

The first message is displayed by the firmware if the error occurred in the first or second record. The status allows a Microdata Customer Service Engineer to determine exactly what is wrong. The second message is printed by the software if the error is in any other record. In this case, the status is stored in memory.

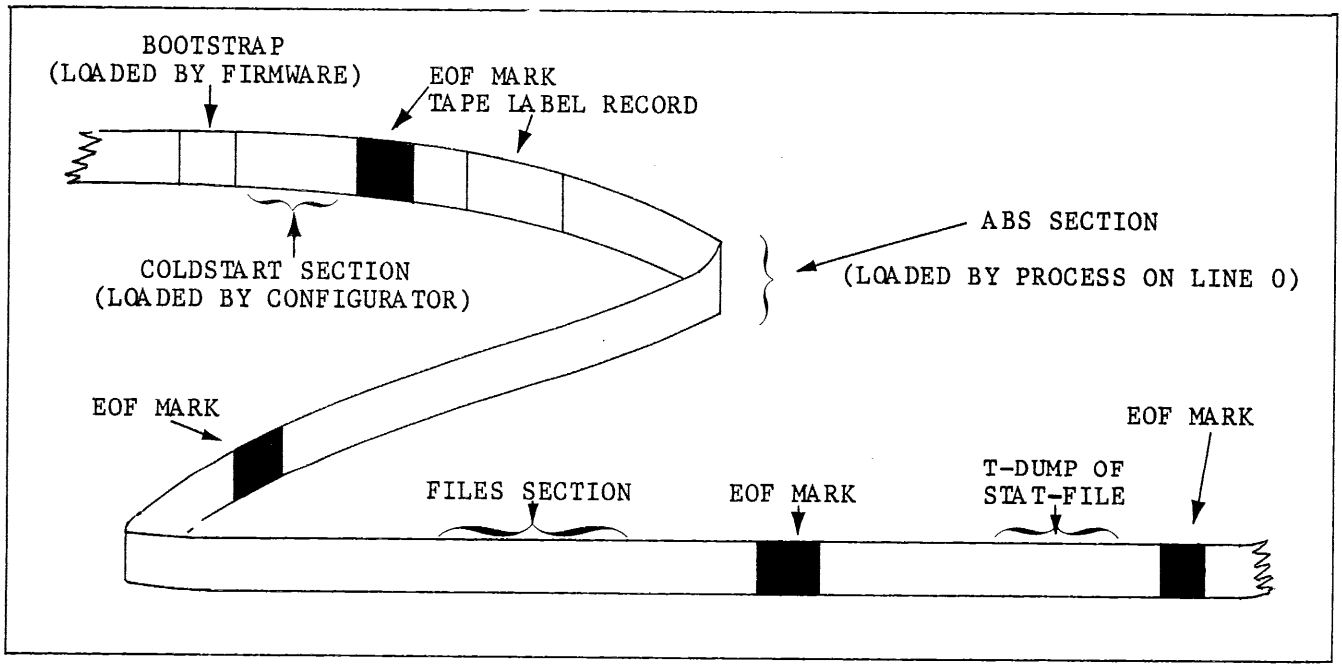


Figure A. Sections of a Formatted File-Save Tape

```
OPTIONS (W,X,A,F,AF,D) =

W   WARMSTART
X   COLDSTART
A   COLDSTART; RESTORE ABS
F   COLDSTART; RESTORE FILES; CLEAR SPOOLED JOBS
AF  COLDSTART; RESTORE ABS AND FILES; CLEAR SPOOLED JOBS
D   DISC FORMATTER
```

Figure B. Options Message

4.7 Complete System Restore: Load Dialogue

The configurator passes control to a virtual process on Line Zero which loads the system software and all the files. When you respond with 'AF' to the "OPTIONS" prompt, the configurator loads the proper disc tables, calculates the maximum FID, and the System Dictionary Base. It configures the main memory buffer tables, etc. Configuration limits are set by the firmware; the parameters are displayed on Line Zero's terminal.

The following lines may appear on Terminal Zero's screen:

1. The assigned spooler port is listed.
2. The prompt "ENTER NUMBER OF TIPH LINES:" appears if the spooler's line is not the last port (i.e., if the spooler's line is 127, this prompt will not appear). Pressing RETURN defaults to 0 TIPH ports. When performing an 'AF' restore, the system uses this number to determine the base of the SYSTEM file. Therefore, for all restores other than an 'AF' restore, you must specify the same number each time the system prompts for this response; if you are performing an 'AF' restore, you may change this number.
3. The amount of main memory is displayed. This will be the smaller of the hardware limit and the amount of memory sensed.
4. A listing of the disc drives, the device address of the disc controller and the number of heads is displayed. This list is the smaller of the sensed disc configuration and the configuration specified in the hardware.
5. You are asked "CONFIGURATION CORRECT (Y/N)?" At this time it is important to verify that the correct configuration is displayed to prevent disc mismapping. Answer "Y" if the display is correct; a response of "N" causes the "OPTIONS" message to be displayed.
6. The prompt "OTHER OPTIONS?" appears. An "N" or <cr> (normal) response causes the subsequent file load to honor reallocation parameters (Line 13 of D-pointer items), if present. A response of "R" inhibits reallocation.

Next, the ABS restore proceeds automatically. As the ABS restore continues, this message is completed: A1-1215, indicating that Frames 1 through 1215 have been loaded. This concludes the load of the system software.

Between the ABS restore and the file-restore, the remaining lines are activated and sent to LOGON. The spooler is started on the last line as indicated by the message "SPOOLER STARTED", and all system software locks are initialized. At this time you are asked to enter the correct date and time.

The file section is restored next. As the file load proceeds, a series of messages appear on the screen which identify the files being loaded. When the file-restore finishes, the system load is completed, and all ports are at LOGON. All that remains to be done is to run the SYSTEM-SETUP PROC (refer to the topic "INITIALIZING THE SYSTEM: SYSTEM-SETUP").

```
OPTIONS (W,X,A,F,AF,D) = AF
120 IS THE SPOOLER'S LINE
ENTER NUMBER OF TIPH LINES: <cr>
1024K OF MEMORY
DRIVE D0,  REFLEX-II
DRIVE D1,  REFLEX-II
CONFIGURATION CORRECT? (Y/N): Y
OTHER OPTIONS? <cr>

TAPE SIZE CHANGED TO 08192

A1 - 1215

SPOOLER STARTED

ENTER DATE.(MM/DD/YY)...11/10/82

ENTER TIME.(HH:MM:SS)...16:46:00
16:46:00   10 NOV 1982

CORRECT TIME AND DATE? (Y/N): Y

SYSTEM 6080,11,1
 BLOCK-CONVERT 6091,7,1
 DL/ID 6091,7,1 (S)
 CHANNEL 6098,1,1
  DL/ID 6099,7,1
DL/ID 6080,11,1 (S)
 PROCLIB 6106,7,1
 DL/ID 6106,7,1 (S)
 POINTER-FILE 6113,5,1
 DL/ID 6113,5,1 (S)
SYSTEM 6080,11,1 (S)
 SYSPROG 6118,69,1
 MD 6118,69,1 (S)
 M/DICT 6118,69,1 (S)
 DL/ID 6118,69,1 (S)
  STAT-FILE 6187,1,1
   DL/ID 6188,87,1


LOGON PLEASE: SYSPROG <cr>

:SYSTEM-SETUP <cr>
```

Complete restore

Configurator

ABS

The system locks are initialized at this point.

Files

Figure A. Complete System Load Dialogue

4.8 System Configuration for Complete Restore: Disc Space Assignment

The location of the SYSTEM dictionary is determined by the hardware. It follows the ABS area, process work spaces, and a region of low disc space.

Each process (as specified by the hardware) is assigned 32 frames of work space. All ABS frames are restricted to the region below Frame 4096. The region of available low disc space is 200 frames per process. This area is used as second priority by the file restore, and first priority in obtaining LOGON workspace. (See the topics, WORK SPACE ASSIGNMENT in Chapter 6 and LOW DISC SPACE AREA AND FILE AREA in Chapter 2.)

The FID of the PCB of the process on Line zero is called PCB0. This will be one more than the number of ABS frames.

The base of the System Dictionary is calculated by:

SYSTEMBASE = (FID of first PCB) + ((number of processes)*232) + 1

The maximum legal (and usable) frame number (FID) for the system is calculated by listing the number of disc platters included in the system. Then use the formula:

Maximum FID = number of REFLEX-II drives * 249536 - 1

This calculation is illustrated in Figure B. Figure A illustrates the physical arrangement of the frames.

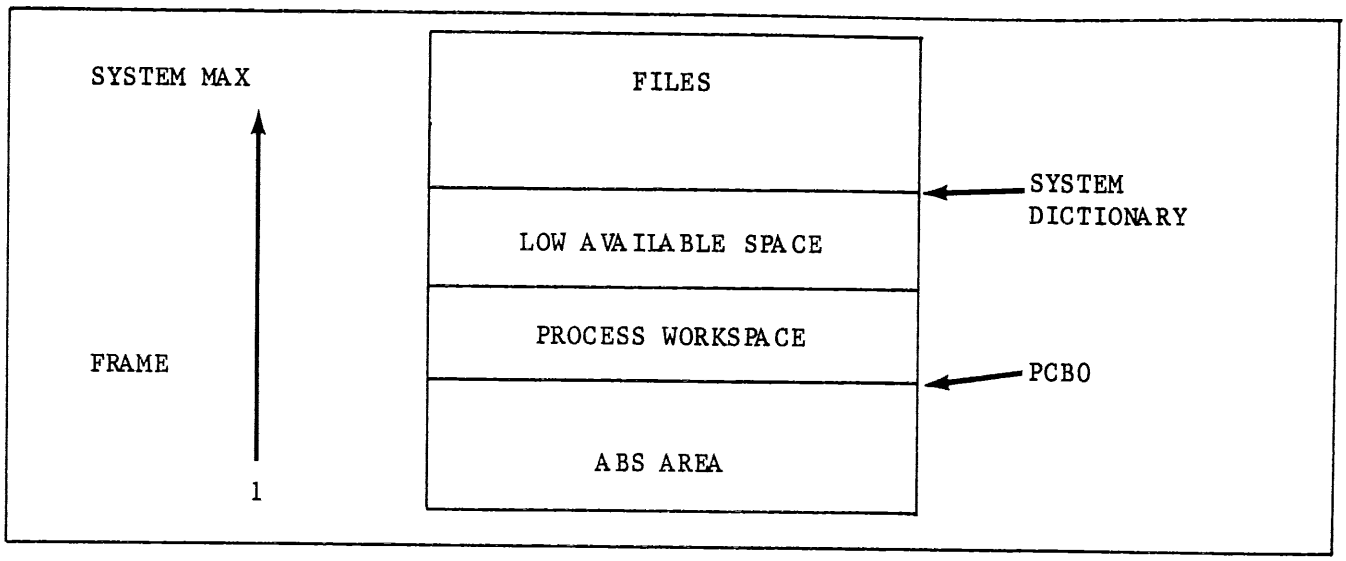


Figure A. Disc Space Assignments

```
Drive 0
Drive 1

total number of drives = 2

maximum fid = 2 * 249536 - 1 = 499071
```

Figure B. Calculation of Maximum FID for a System with Two Disc Drives

4.9 Warmstart

A warmstart allows you to restart the monitor. This section describes when and how to do a warmstart.

If any of the system's terminals hang, and it is suspected that the monitor has stopped operating correctly, it is desirable to perform a warmstart from tape. If the system failed when work was in progress, the warmstart is the only way to restart the system without loss of work in progress. Performing a warmstart reloads only the system monitor. In addition to reloading the system monitor, a warmstart also flushes memory, in case a coldstart has to be performed.

CAUTION

NEVER PERFORM A WARMSTART AFTER CHANGING OR SWAPPING THE SYSTEM'S MEMORY!

Follow these instructions to warmstart the system:

1. Mount a tape that begins with a bootstrap section on the tape unit.
2. Close the tape unit door securely.
3. Press the LOAD button twice.
4. Press the ONL button once.
5. When the READY light is on, turn the key to TEST.
6. Enter <c>D to display the System Control Menu.
7. When the prompt "ENTER OPTION (1-10) :" is displayed, enter "2" followed by the RETURN key.
8. Enter "W" in response to the "OPTIONS" message.

If the system encounters a buffer linkage problem while performing the warmstart, it notifies you with error messages and the prompt:

ENTER F FOR FIRMWARE DEBUGGER OR RETURN TO CORRECT?

Press the RETURN key to attempt to correct the problem. If the problem is not corrected, the above message is displayed again. If this occurs, you have two options: 1) contact you Microdata support representative if you have the time to research and solve the problem, or 2) perform a coldstart.

When the warmstart is complete, all terminals should be operational at the point where they previously failed. Figure A illustrates a terminal display that has been interrupted, followed by a warmstart.

```
M/DICT.............. D/CODE F/BASE..........F/MOD........... F/SEP

DISC-ERRORS            D      1031         3                  1
ERRMSG                 D      1035
OPTIONS (W,X,A,F,AF,D) = W
127 IS THE SPOOLER'S LINE
1024K OF MEMORY
DRIVE D0,  REFLEX-II
DRIVE D1,  REFLEX-II
CONFIGURATION CORRECT? (Y/N)? Y <cr>
DL/ID                  D      874         68                  1
M/DICT                 D      874         68                  1
MD                     D      874         68                  1
STAT-FILE              D      942          1                  1
ACC                    Q      ACC
ACCOUNT                Q      ACC
CHANNEL                Q      ACC
PROCLIB                Q      PROCLIB

I 7.D9
!END <cr>
```

Note that a warmstart may cause some terminal characters to be dropped.

Figure B. Example of LISTFILES Interrupted, Followed by Warmstart

4.10 Coldstart

If a warmstart fails, or if any main memory module has been moved or replaced, it is necessary to perform a coldstart. Also, every complete ('AF') system load begins with a coldstart. In a coldstart, the following steps occur (see Figure A).

1. 1024 bytes are transferred to low main memory from tape. These 1024 bytes include a program for Part 2.
2. The configurator is loaded from tape.
3. The configuration process is started.
4. An initial set of virtual frames is read from tape.
5. Control is given to Channel Zero which carries out other desired options (e.g., ABS load, file-restore).

To perform a coldstart, follow these steps:

1. Mount tape beginning with a bootstrap section on the tape unit.
2. Close tape unit door securely.
3. Press LOAD button twice.
4. Press ONL button once.
5. When RDY light is on, turn key to TEST.
6. Enter <c>D to display System Control Menu.
7. To the prompt "ENTER OPTION (1-10) :", type in "2" followed by the RETURN key.
8. Enter "X" in response to the "OPTIONS" message.
9. To the question "INITALIZE SPOOLER TABLES? (Y/N):", enter "N" to restart the spooler without losing print jobs. A "Y" reply also restarts the spooler but loses all print jobs, and the space associated with these print jobs is not returned to overflow. Also, the additional work space consumed by the processes logged on at the time of the coldstart is lost, and all lines are sent to logon.
10. To the question "CONFIGURATION CORRECT? (Y/N):", enter "Y" if the configuration is correct; an "N" response redisplays the "OPTIONS" message.
11. Enter the correct date and time.

    At this point, logon to the system and run the SYSTEM-SETUP PROC.

NOTE: A potential situation exists that may cause group format errors in files that had frames in main memory at the time of the coldstart. It is desirable to repeat an attempt at a warmstart several times before proceeding to a coldstart.

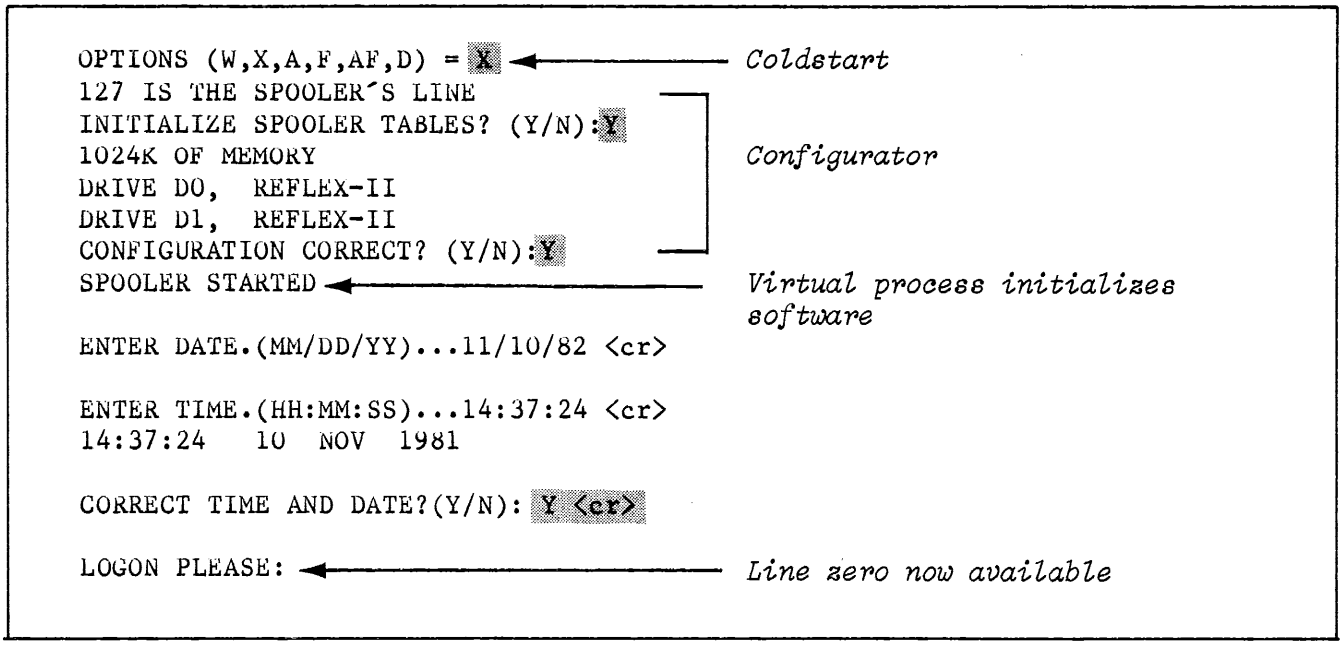


Figure A. Example of a Coldstart Dialogue

4.11 ABS Loads

This topic describes recovery procedures affecting portions of system software only.

ABS LOADS

It may sometimes be necessary to change or restore the virtual part of system software. This is done via an ABS load from tape. There are two ways to do this:

1. Following a coldstart from tape. This is normal during a complete system load. It may also be necessary if a failure prevents operation of the system via TCL. This method is accomplished by responding with "A" to the prompt:

   ```
   OPTIONS (W,X,A,F,AF,D) =
   ```

2. From TCL. (Use with CAUTION because data within frames is not initialized.) This is accomplished by mounting a tape which has the desired ABS section, attaching the tape unit spacing past the bootstrap section (T-FWD if the bootstrap section is present), and entering:

   ```
   :T-RDLBL 1 <cr>
   :ABSLOAD <cr>
   ```

In Case 1 above, all system software locks are initialized, the spooler is restarted, and all lines are sent to logon.

In Case 2 above, locks are not reset, the spooler is not restarted, and no lines are sent to logon (see the :INIT-SYSTEM and :RESTART-SPOOLER verbs).

In both cases, press RETURN in response to the prompt "OTHER OPTIONS?" (reserved for future enhancement). ABS frames are loaded from tape in the order in which they occur. The progress of the load is indicated by a sequence of one or more messages in the form:

Am-n (for a range of frames)

Am (for individual frames)

If a parity error is encountered during an ABS load, the prompt:

PARITY ERROR! TYPE A (TRY AGAIN) OR TYPE I (IGNORE)?

appears (see Figure A). Entering "I" causes the ABS load to be accepted with the error. This should be used with caution because all aspects of the system may not operate properly after this has been done. It is not possible to identify the frame which contains the parity error other than to say that it is within the range of frames indicated by the current message (Am-n).

Another possible error is a checksum error. In this case, the following message appears:

CHECKSUM ERROR FRAME = xx TYPE A (TRY AGAIN) OR TYPE I (IGNORE)?

Entering "I" causes the ABS load to be accepted with the error.

Finally, the message:

FORMAT ERROR IN ABS SECTION; TYPE A (TRY AGAIN) OR TYPE I (IGNORE)?

may occur. This means that a format error was discovered on the tape. Possible causes are:

1. Tape is not positioned at the start of an ABS section or the T-RDLBL was not used if initiated from TCL (wrong record size in effect).

2. Tape has deteriorated or was misread in such a way that the format was unrecognizable. This may happen after entering "I" in response to the prompt following a parity error.

Entering "I" in response to this prompt initiates a search for the next ABS frame segment on tape. If such a segment is found, the restore continues with the frame indicated. If no segment is found, the tape continues moving until it is stopped or an end-of-file is reached.

The best way to recover from these situations is to use another tape to perform the ABS load, or if this is not feasible, it may be possible to MLOAD the necessary frames from the SYSTEM-OBJECT file.

```
OPTIONS (W,X,A,F,AF,D) = A <cr>          <-- Coldstart, ABS restore
127 IS THE SPOOLER'S LINE
INITIALIZE SPOOLER TABLES? (Y/N):N
1024K OF MEMORY                          <-- Configurator
DRIVE D0,  REFLEX-II
DRIVE D1,  REFLEX-II
CONFIGURATION CORRECT? (Y/N): Y
OTHER OPTIONS? <cr>
TAPE SIZE CHANGED TO 08192
A1
PARITY ERROR!
TYPE A (TRY AGAIN) OR TYPE I (IGNORE) I <cr>
FORMAT ERROR IN ABS SECTION              <-- Caused by parity error
ENTER A (TRY AGAIN) OR TYPE I (IGNORE) I <cr>
-53                                      <-- This indicates that frames
A55-1215                                     1 - 53 were loaded.
SPOOLER STARTED                              Frame 54 was ignored.
                                             Frames 55 - 1215 were
ENTER DATE.(MM/DD/YY)...11/10/82 <cr>        loaded.

ENTER TIME.(HH:MM:SS)...14:37:24 <cr>
14:37:24   10  NOV  1982

CORRECT TIME AND DATE?(Y/N):Y <cr>

LOGON PLEASE:
```

Figure A. An ABS Load, Including Recovery from a Parity Error

4.12 System Configuration on System Upgrade

If a system is changed to use a different number of ports, discs, or ABS frames, an 'AF' restore is mandatory.

Because placement of the System Dictionary is set by hardware maximums for communication lines and ABS frames, a complete ('AF') restore from a file-save tape is needed when a system upgrade changes the configuration.

Similarly, a complete ('AF') restore is needed when a system is changed to run on a different combination of discs because the frame mapping onto the discs is altered.

Only an 'X' level coldstart is necessary when changing the amount of main memory a system uses.

(THIS PAGE INTENTIONALLY LEFT BLANK)

4.13 Formatted File-Restore

File-restores cause files to be created, loaded, and integrated into the system. A file-restore of a complete system builds a system containing all the files and data of the source system.

Sequence of Events in a Formatted File-Restore

The first event in a complete formatted file-restore is initialization of available overflow space to the complete range on the system from the process work spaces forward to the end of disc.

The System Dictionary is then created and cleared. The first account Master Dictionary is created and a pointer to it is placed in the System Dictionary. Then the first file dictionary is created and a pointer to it is placed in the account's Master Dictionary. Next is the data file, which will proceed in one of two ways:

1. The slow method. The file is created, a pointer is added to the dictionary, then the data is loaded. This method is necessary if reallocation is being performed, or if the file is a pointer file.

2. The fast method. The file is loaded group by group as it is created. After it is completely loaded, a pointer is placed in the dictionary. This is the normal method.

Next, the file dictionary is loaded. The next file's dictionary and data sections are created and loaded, and so forth until all of the account's files are present. Then the account dictionary is loaded. When the account is finished, a D-pointer to the account is placed in the System Dictionary. The file restore continues in the same manner for the next account (if any).

Account-restores proceed in the same sequence, except that the System Dictionary is already present, and only the pointer to the account Master Dictionary is added to it.

Console Listing Accompanying File-Restore

Figure A is an example of a formatted file-restore listing. Each line corresponds to a file pointer and is indented in accordance with the level of the file in which the pointer is placed. The file-name is first, followed by the base, modulo, and separation of the file as it is being restored. An "(S)" following the line indicates that the pointer has the same base as some other pointer already listed and that file has already been created.

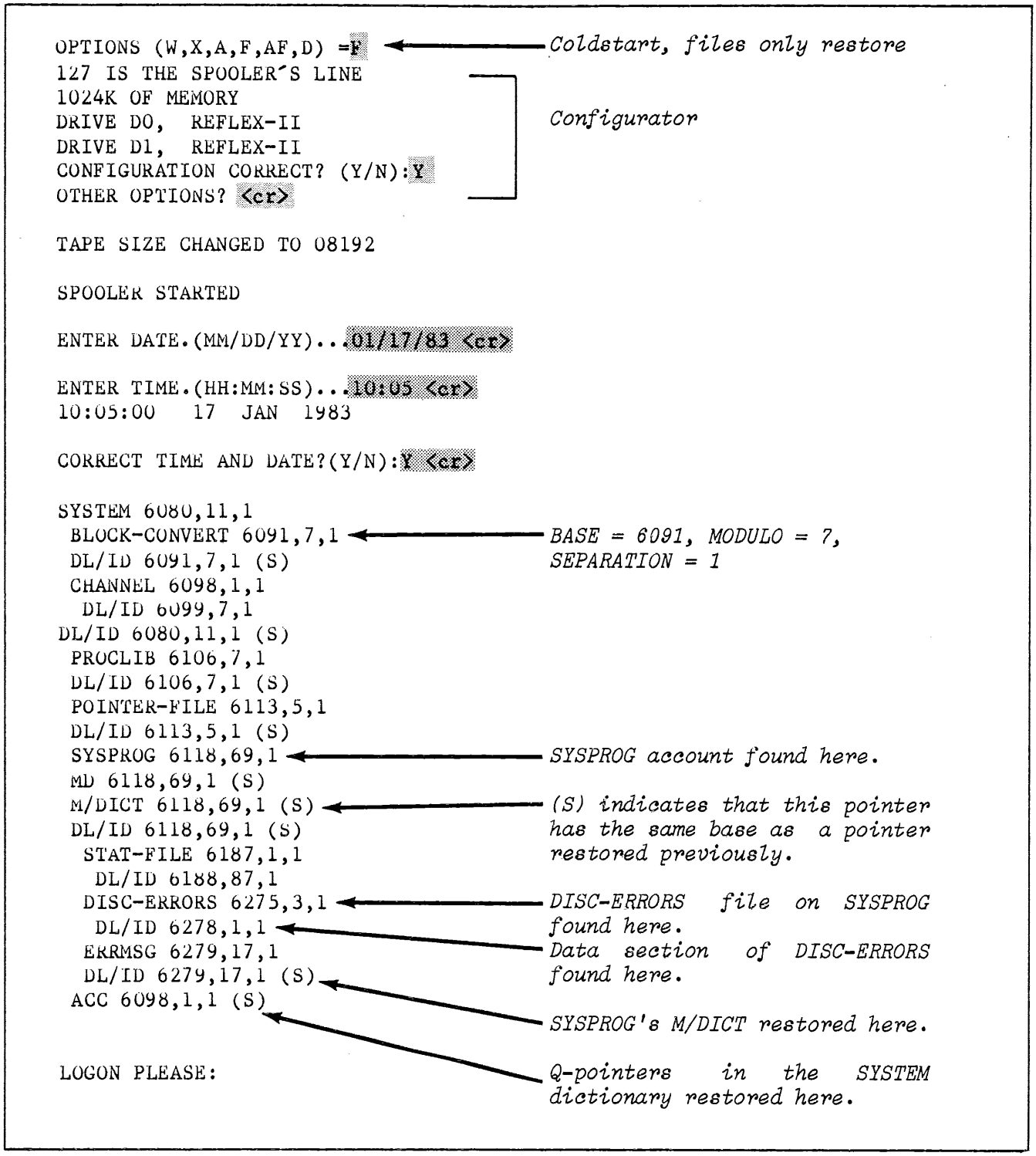


Figure A. Sample Formatted File-Restore Listing

4.14 Error Recovery During File Loads

If parity errors or other errors mar the files section of a formatted file-save tape, some data may be lost. The restore will continue, but operator assistance may be needed.

Parity Error Recovery Procedure

If a parity error is detected on a formatted file-restore, the prompt

```
PARITY ERROR!
TYPE A (TRY AGAIN) OR TYPE I (IGNORE)
```

is printed (see Figure A). Entering 'I' causes the data block to be accepted as is from tape without data correction. The specific item affected cannot be determined. The file is the one whose name was last displayed on the terminal controlling the restore.

Recovery From Destroyed Pointers

If tape information identifying a file is destroyed, it may be impossible for the restore to create that file and subsequent files in the right order. The message

```
'ERROR IN D-SEGMENT
@xxxxxxxx
LEVEL (1-3)?'
```

is printed. "xxxxxxxx" describes the software location where the error was detected. You must advise the restore processor whether to search for and continue with the next account (response = 1), the next dictionary (response = 2), or data file (response = 3). The response requires your judgment as to the positioning of files on the tape and the total situation.

Tape Format Errors

If a segment of the file-save tape cannot be recognized, a tape format error occurs and the following message is displayed:

```
TAPE FORMAT ERROR @xxxxxxxx SEGMENT SKIPPED
```

The segment is skipped and the restore continues with the next segment (see the topics FILE-SAVE TAPES: FORMAT and TAPE SEGMENTS IN FORMATTED FILE-SAVE in Chapter 11); the beginning of the segment skipped will be displayed.

A common cause of these errors is improper positioning of the tape or failure to use T-RDLBL if initiating the restore from TCL.

```
OPTIONS (W,X,A,F,AF,D) =F
127 IS THE SPOOLER'S LINE
1024K OF MEMORY
DRIVE D0,  REFLEX-II
DRIVE D1,  REFLEX-II
CONFIGURATION CORRECT? (Y/N):Y
OTHER OPTIONS?  <cr>

TAPE SIZE CHANGED TO 08192

SPOOLER STARTED

ENTER DATE.(MM/DD/YY)...01/17/83 <cr>

ENTER TIME.(HH:MM:SS)...10:10 <cr>
10:10:00   17  JAN  1983

CORRECT TIME AND DATE?(Y/N):Y <cr>

  SYSTEM 4224,11,1
   BLOCK-CONVERT 4235,7,1
   DL/ID 4235,7,1 (S)

PARITY ERROR! ENTER A TO TRY AGAIN, I TO IGNORE? I <cr>

TAPE FORMAT ERROR @00D60104 SEGMENT SKIPPED
003AP^12^C11^1
   CHANNEL 4242,1,1
    DL/ID 4243,7,1
  DL/ID 4224,11,1 (S)
   PROCLIB 4250,7,1
   DL/ID 4250,7,1 (S)
   POINTER-FILE 4257,5,1
   DL/ID 4257,5,1 (S)
  SYSTEM 4224,11,1 (S)
   SYSPROG 4262,69,1 (S)
   MD 4262,69,1 (S)
   M/DICT 4262,69,1 (S)
   DL/ID 4262,69,1 (S)
    STAT-FILE 4331,1,1
     DL/ID 4332,87,1
    DISC-ERRORS 4419,3,1
     DL/ID 4422,1,1


LOGON PLEASE:
```

Coldstart, files only restore

Configurator

The 'I' caused one or more bytes to be incorrect.

The start of the skipped item looked like this

The restore continued

Figure A. Recovery from Parity Error in Formatted File-Restore

5 TERMINAL CONTROL LANGUAGE

5.1 Overview

The Terminal Control Language (TCL) is the primary interface between you and various SEQUEL processors.

Most processors are activated directly from TCL by a single input statement, and return to TCL after completion of processing. Some processors (the EDITOR for example) retain control of the terminal until explicitly exited, at which point they return control to TCL. TCL prompts you by displaying a colon (:). This is referred to as the "TCL prompt character". Input statements are constructed by typing one character at a time on the terminal until the carriage return or line feed key is pressed, at which time the entire line is processed by TCL.

TCL expects the first word of an input statement to be a "verb". There are three types of verbs in SEQUEL:

- ENGLISH verbs
- TCL-I verbs
- TCL-II verbs

One of SEQUEL's powerful features is your ability to customize each user's vocabulary. Since verbs reside in each user's Master Dictionary (M/DICT), the vocabulary may be added to or deleted from without affecting the other users. In addition, an unlimited number of synonyms may be created for each verb.

Figure A presents an overall view of the intraprocessor linkages within the SEQUEL system. This figure illustrates the flow of control between TCL and various SEQUEL processors.

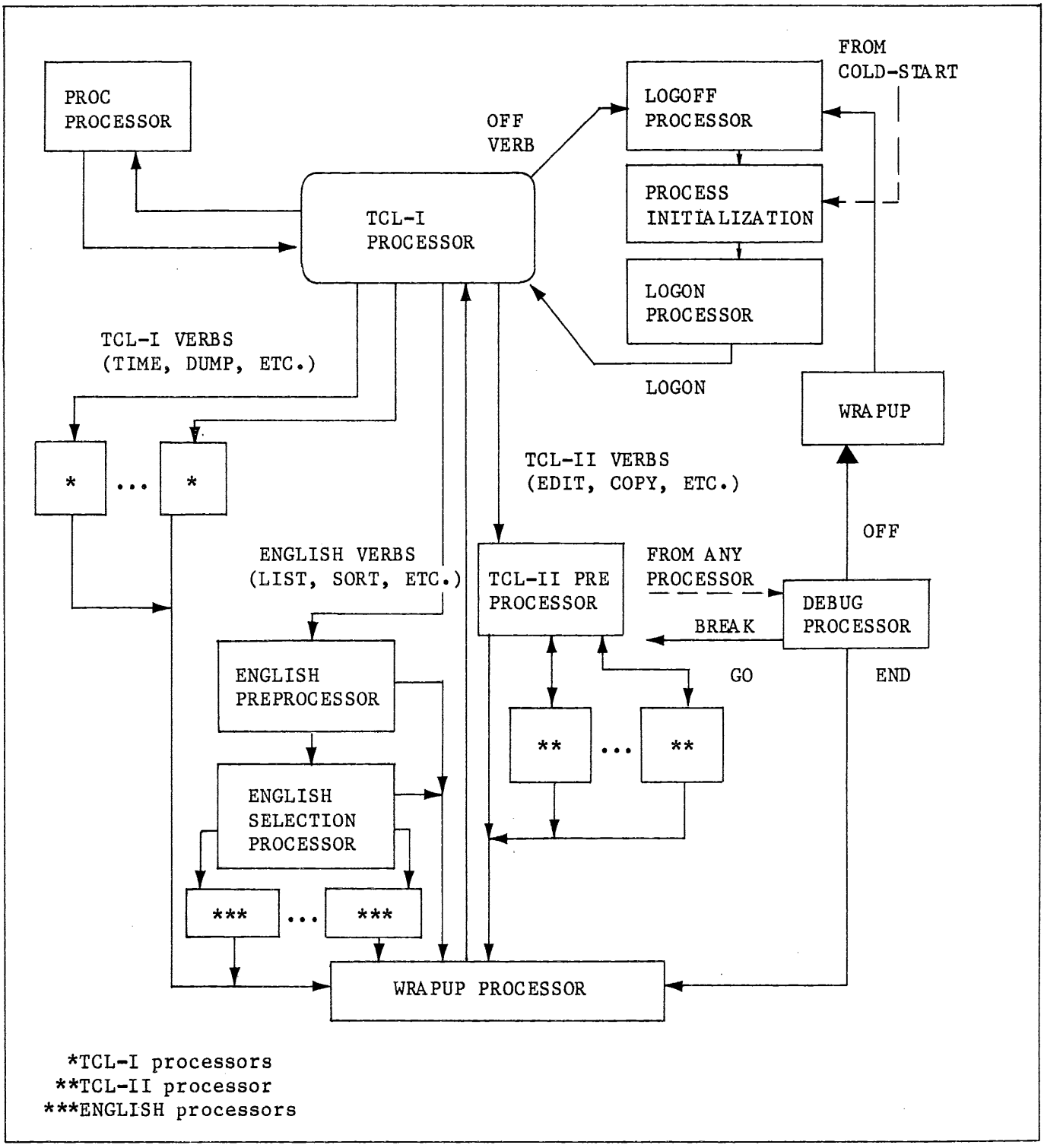


Figure A. Overall View of SEQUEL Intraprocessor Linkages

5.2 Input Statements

TCL operates on one input statement at a time. An input statement must begin with a verb, and may be comprised of multiple lines.

The system normally operates in the "full-duplex" mode of communication with each terminal. "Full-duplex" means that data is being transmitted between the terminal and the computer in both directions simultaneously. The system also operates in what is known as an "Echo-Plex" environment. This means that each data character input by the terminal is echoed back to the terminal by the computer. Thus, the character displayed on the screen is the same as the character stored in the computer. It is possible to stop terminal echo by using the HUSH verb. Terminal is restored with the NOHUSH verb.

TCL handles only complete input lines. When you compose an input statement, no action is taken until TCL detects a carriage return character. If a carriage return is not detected, data characters are assembled in your input buffer until 140 characters have been entered, and TCL automatically generates a carriage return.

In addition to the standard ASCII character set recognized by TCL (refer to the Appendix), special operations are performed when control characters are detected. The control characters listed in Figure A perform editing functions. All other control characters are deleted from the input line that is passed to lower procedures, but remain in the original input line. Control characters are represented in this manual by preceding the character with the symbol <c>. This means that the <CONTROL> or <CTRL> key must be pressed while typing the desired character. The <SHIFT> key may also have to be pressed simultaneously if the desired character appears on the upper half of a key. TCL analyzes your input statement and copies the statement into a buffer. It also performs the following tasks:

The first word of the sentence is assumed to be a verb and is looked up in your Master Dictionary (M/DICT), but is not copied.

Redundant blanks surrounding all words in the statement are deleted.

Character strings surrounded by single quotes (') or double quotes (") are identified and copied verbatim, including redundant blanks.

Figure B defines the options that may be used with TCL-I and TCL-II verbs.

Unless defeated with an "N" option (see Figure B), all listings to the terminal will pause at the end of each screen and wait for a keystroke before continuing with the next screen. The control characters <c>E and <c>S have special meanings if input at this pause:

<c>E The listing will be terminated and control returned to PROC or TCL. BASIC will respond to this control character if a heading or footing is enabled.

<c>S The <c>S is echoed and entry of another character is required before the listing will continue. This allows the screen-dump facility of certain terminals to be used to reproduce selected pages on a slave printer.

CONTROL CHARACTER	FUNCTION
<c>H	Backspace. Deletes last character typed in and allows you to re-enter character.
<c>X	Cancel. Deletes the entire line currently being typed in.
<c>R	Retype. Causes entire line currently being typed in to be retyped.
<c>_	Line Continuation. If typed as the last character in a line, will allow continuation of the line onto the next physical line. This character must be immediately followed by a carriage return or line feed.
<c>E	The listing will be terminated and control returned to PROC or TCL. BASIC will respond to this control character if a heading or footing is enabled.
<c>S	The <c>S is echoed and entry of another character is required before the listing will continue.

Figure A. Special Control Characters

N	Inhibit automatic pause for carriage-return at end of page (NOPAGE).
I	List item-ids while performing copy, dump, etc.
P	Produce output on printer (spooler).

Figure B. TCL-I Option Specifications

5.3 Verb Definition Items in M/DICT

Each TCL-I, TCL-II, or ENGLISH verb is defined as an item in your Master Dictionary (M/DICT).

Each verb definition resides as an item in your M/DICT. The item-id of a verb definition item is the verb name itself. The attributes used in a verb definition item are defined in Figure A.

As an example of a verb definition item, the following item (stored as item 'LIST' in your M/DICT) defines the ENGLISH verb LIST:

Item 'LIST' in M/DICT

```
001 PA
002 35
003 4D
```

The following verb definition item defines the TCL-II verb MLIST:

Item 'MLIST' in M/DICT

```
001 PY
002 2
003 20
004
005 CU
```

As a final example, the following verb definition item defines the TCL-I verb TIME:

Item 'TIME' in M/DICT

```
001 PZ
002 3033
```

You may create any number of synonyms for verb definition items (and may even remove the predefined verb definition items), thereby creating your own vocabulary. Synonyms may be created by copying the verb definition item into another M/DICT item with the desired synonym name as the item-id.

ATTRIBUTE NUMBER	DESCRIPTION
0	This is the item-id, which is the name of the verb
1	Must contain: Pc P identifies the M/DICT item as a verb definition item. The single character c is passed to the defined processor. (Note: If c is "Q", the item is a PROC, not a verb; in this case, the "Q" may be followed by an "N". Refer to the PROC Manual for further information.)
2	This attribute defines the processor entry point to which TCL passes control (i.e., the mode-id in hex). An ENGLISH verb will have an entry of: 35 A TCL-II verb will have an entry of: 2 A TCL-I verb will have an entry of: xxxx where xxxx is the entry point
3	Secondary transfer point. Use depends on Attributes 1 and 2
4	Tertiary transfer point. Use depends on Attributes 1 and 2
5	TCL-II parameter string. These parameters govern treatment of the items retrieved by TCL-II verbs to be passed to the processor whose entry point is defined in Attribute 3. Parameter may be any of the following: C - Copy item to a work area E - Blanks are compressed on file, expanded for processing F - Pick up file parameters only (ignore item-list) N - Okay if item is not on file P - Print item-id if multiple items are specified U - Items will be updated by processor Z - Final entry required after processing last item

Figure A. Verb Definition Item in M/DICT

5.4 TCL-II Input Statements

The format of a TCL-II input statement is more restrictive than that of an ENGLISH statement (refer to the ENGLISH Programming Manual). This enhances processing speed, as statement parsing is quicker.

Selection-criteria and output-specifications are not allowed in TCL-II statements. The file-name (or DICT file-name) must immediately follow the TCL-II verb. Item selection is more restricted, since each item-id must be explicitly named in the statement, although all items may be specified by the use of an asterisk (*).

Figure A illustrates the general form of a TCL-II statement. The file-name specifies the desired file. The DICT option specifies the dictionary portion of the file. The item-list is made up of one or more item-ids, separated by blanks. It an item-id contains embedded blanks or parentheses, it must be surrounded by single quotes. All items in a file may be specified by entering asterisk (*) as the item-list. Options, if specified, must be enclosed in parentheses at the end of the input line. Multiple options may be separated by commas. The specified options are passed to the appropriate TCL-II processor. Figure B illustrates a sample TCL-II input statement.

Some TCL-II verbs are listed in Figure C. For further information, refer to the following sections of this manual:

FILE MANAGEMENT PROCESSORS
MAGNETIC TAPE UNIT VERBS
SPECIAL SYSTEM UTILITIES
SYSTEM MAINTENANCE PROCEDURES

Also, refer to the following manuals:

ENGLISH Programming Manual
SEQUEL Assembly Language Programming Manual
DATA/BASIC Programming Manual
PROC Programming Manual
EDITOR Programming Manual

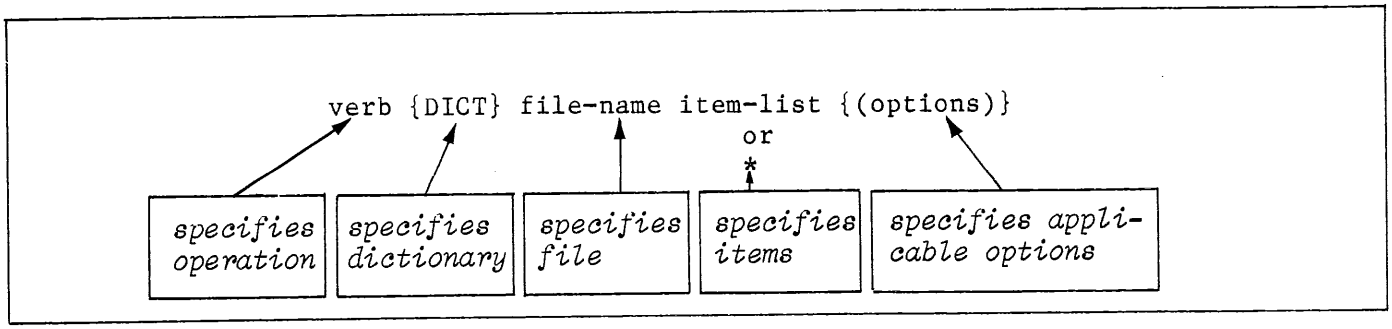


Figure A. General Form of TCL-II Input Statement

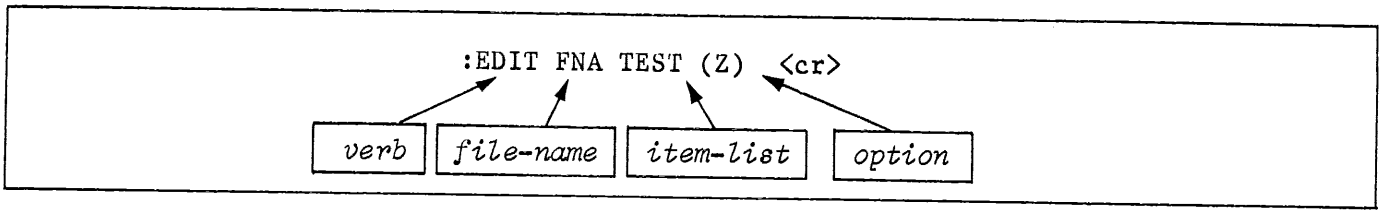


Figure B. Sample TCL-II Input Statement

VERB	DESCRIPTION
N	*Inhibit automatic pause for <cr> at end of page (NOPAGE).*
I	*List item-ids while performing COPY, DUMP, etc.*
P	*Produce output on printer (spooler).*

Figure C. Sample TCL-II Option Specifications

VERB	DESCRIPTION
COLDDUMP	*Create coldstart (bootstrap) tape.*
COPY	*Copy data files and dictionaries.*
EBASIC	*Compile compressed DATA/BASIC program.*
ECOPY	*Expand compressed items.*
EED	*Edit compressed items.*
FIX-FILE-ERRORS	*Repair group format errors.*
GROUP	*List file usage statistics for an entire file.*
ITEM	*List usage statistics for a single item.*
MEM-DIAG	*Start memory diagnostic.*
SAVE	*Backup system.*

Figure D. Some TCL-II Verbs Described in This Manual

5.5 TCL-I Input Statements

Verbs which have a code other than "2" or "35" in Attribute 2 of their respective M/DICT verb definition items are called TCL-I verbs. TCL-I verbs are used to form TCL-I input statements. They may have an option specification of the form: (options).

A TCL-I input statement must begin with a TCL-I verb and end with a carriage return. Some TCL-I verbs additionally allow for various parameter specifications. TCL-I verbs are listed in Figure A. For further information regarding these verbs, refer to the following sections in this manual:

- LOGON/LOGOFF
- FILE MANAGEMENT PROCESSORS
- SPOOLER
- MAGNETIC TAPE UNIT VERBS
- SPECIAL SYSTEM UTILITIES
- SYSTEM MAINTENANCE PROCEDURES

VERB	DESCRIPTION
:ABSLOAD	*Loads executable assembly code (ABS) from tape.*
:ABS/FILES	*Loads ABS and files from tape.*
:FILES	*Restores files from tape.*
:INIT-SYSTEM	*Initializes system locks.*
:RESTART-SPOOLER	*Restarts spooler process.*
ACCOUNT-RESTORE	*Restores user account from tape.*
ADDD, ADDX	*Adds decimal or hexadecimal numbers.*
BLOCK-PRINT	*Sends block characters to printer.*
BLOCK-TERM	*Sends block characters to terminal.*
BUFFERS	*Displays status/content of memory buffers.*
CHARGE-TO	*Keeps track of computer usage.*
CHARGES	*Prints current computer usage.*
CLAIM	*Returns linked overflow to contiguous overflow.*
CLEAR-BASIC-LOCKS	*Resets execution locks (0-63).*
CLEAR-FILE	*Removes all file items from a file or dictionary.*
CREATE-FILE	*Creates a new file.*
DE-RESET	*Resets disc error recording table.*
DE-START	*Starts disc error recording.*
DE-STOP	*Stops disc error recording.*
DELETE-FILE	*Deletes an entire file.*
DISCIO	*Produces disc activity report.*
DIVD, DIVX	*Divides decimal or hexadecimal numbers.*
DTX	*Converts decimal number to hexadecimal.*
DUMP	*Dumps virtual frames to terminal.*

Figure A. Some TCL-I Verbs Described in This Manual

VERB	DESCRIPTION
HUSH	*"Hushes" terminal echo.*
LIST-LOCKS	*Displays group locks.*
LOGTO	*Switches to another user account.*
MESSAGE	*Communicates to other users.*
MSG	*Same as MESSAGE.*
MULD, MULX	*Multiplies decimal or hexadecimal numbers.*
NOHUSH	*Restores terminal echo.*
OFF	*Terminates user's session.*
P	*Toggles printing at terminal on and off.*
P-ATT	*Attaches line printer.*
P-DET	*Detaches line printer.*
POVF	*Prints available overflow report.*
SET-DATE	*Sets system date.*
SET-TERM	*Sets terminal/printer characteristics for all lines.*
SET-TIME	*Sets system time.*
SLEEP	*Sets a process into quiescent state for specified time.*
SP-ASSIGN	*Assigns print job(s) to form queue and assigns option(s) to print job(s).*
SP-CHECKQ	*Counts number of hold files.*
SP-EDIT	*Examines closed print jobs.*
SP-EJECT	*Ejects line printer pages.*
SP-KILL	*Cancels current spooler output on designated device.*
SP-RESUME	*Recovers from spooler halts.*
SP-SKIP	*Sets number of form feeds.*
SP-STATUS	*Displays status of spooler form queues.*
SP-STOP	*Halts spooler after current print job.*
SP-TAPEOUT	*Prints tape file on line printer.*
SUBD, SUBX	*Subtracts decimal or hexadecimal numbers.*
T-ATT	*Attaches magnetic tape unit.*
T-BCK	*Backspaces tape.*
T-CHK	*Checks tape for parity errors.*
T-DET	*Detaches magnetic tape unit.*
T-FWD	*Forward-spaces tape.*
T-RDLBL	*Reads tape label.*
T-READ	*Reads one record from tape.*
T-REW	*Rewinds magnetic tape.*
T-WEOF	*Writes EOF on tape.*
TERM	*Sets terminal characteristics.*
TIME	*Prints time and date.*
USER	*Displays privilege level and workspace assignment.*
WHAT	*Displays current system parameters.*
WHERE	*Shows user points of execution.*
WHO	*Prints the line number and account name to which the terminal is logged on.*
XTD	*Converts hexadecimal number to decimal.*

Figure A. Some TCL-I Verbs Described in This Manual (Continued)

5.6 Program Interruption (Debug Facility)

Processing can be interrupted by pressing the <BREAK> key on the terminal. This causes an interrupt in current processing and an entry into the DEBUG state. However, such an interruption is inhibited during critical stages of processing.

When the <BREAK> key has been pressed and the DEBUG state has been entered, the following message is displayed:

```
I x.d
!
```

where "x" and "d" describe the software location of the interruption (refer to the DEBUG documentation in the SEQUEL Assembly Language Programming Manual). The DEBUG prompt character (!) prompts you for a DEBUG command. For users with system privilege levels zero or one, the commands listed in Figure A are the only DEBUG commands allowed. Users with system privilege level two should refer to the SEQUEL Assembly Language Programming Manual for further DEBUG facilities.

Figure B illustrates a sample terminal interaction with the DEBUG facilities.

Note that pressing the <BREAK> key while in the terminal input or output mode causes a loss of an entire line of characters. If in the input mode, the retype-line character (<c>R) should be used to check the loss of data after returning from DEBUG via the 'G' command.

Upon encountering one of the hardware abnormal conditions, the system automatically traps to the DEBUG state with a message indicating the nature and location of the abort. If you have system privileges level zero or one, you must type "END" or "OFF" to exit from the DEBUG state. Hardware abnormal conditions are described in the DEBUG section of the SEQUEL Assembly Language Programming Manual.

When either "END" or "OFF" is entered while in the DEBUG state, the locks set by that process are unlocked. These locks include execution locks (0-63), and group locks.

COMMAND	DESCRIPTION
P	*Print on/off. Each entry of a "P" command switches (toggles) from print suppression to print nonsuppression. The message OFF is displayed if output is currently suppressed. The message ON is displayed if output is resumed. This feature is useful when debugging.*
H	*Toggles terminal echo off and on.*
G or <lf>	*Causes resumption of process execution from the point of interruption. "G" cannot be used if a process ABORT condition caused the entry to DEBUG.*
END <cr>	*Terminates current process and causes an immediate return to TCL.*
END <lf>	*Terminates current processing, but if a DATA/BASIC program was executing, control will pass to next statement in a PROC, if executing within a PROC.*
OFF	*Terminates current process and causes you to be logged off the system.*

Figure A. DEBUG Commands for Users with System Privilege Levels Zero or One

:LIST STAFF CITY HDR-SUPP <cr>	*ENGLISH LIST statement*
STAFF..... CITY................	
1432 DALLAS	
2391 PHILADELPHIA	*<BREAK> key pressed.*
I 6.1A3	*Interrupts message.*
!P <cr> OFF	*Turns Print off.*
!G <cr>	*Resumes execution without printing.*
I 3.FB	*<BREAK> key pressed, interrupts message.*
!P <cr> ON	*Turns Print back on.*
!G <cr>	*Resumes execution with printing.*
2951 LOS ANGELES 3051 NEW YORK 4260 CHICAGO	*Listing output resumes.*
5913 MIAMI	*<BREAK> key pressed.*
I 6137	*Interrupts message.*
!END <cr>	*Terminates LIST execution.*
:	*TCL prompt.*

Figure B. Sample Use of DEBUG Facilities

6.1 Logging On and Off the System

The logon processor initiates your session by identifying valid users and their associated passwords. The logoff processor terminates the session and should always be evoked via the verb OFF. These processors can accumulate accounting statistics for billing purposes and also associates you with your privileges and security codes.

Logging On to the System

You may log on to the SEQUEL system when the following (or a user-created) message is displayed:

```
LOGON PLEASE:
```

You then enter the user-id of the account established for you in the System Dictionary. If a password has also been established, you may follow your identification with a comma, and then the password; otherwise, if the specified account requires a password, the system will display the message:

```
PASSWORD:      Entered password is not echoed.
```

The "hushed" area allows you to enter your password and not have it observable. SEQUEL validates your identification against the entries in the System Dictionary; if it is illegal, the following message is returned:

```
USER-ID?
LOGON PLEASE:
```

You must then reenter your identification and password. If your identification is valid, but the password is not acceptable, the following message is displayed:

```
PASSWORD?
LOGON PLEASE:
```

You must then reenter your identification and password. If you have successfully logged on to the system (i.e., both the identification and the password have been accepted), the following message (#335) is displayed:

```
***   WELCOME TO MICRODATA  S E Q U E L   ***
***   time        RELEASE x.y        date   ***
:
```

where "time" is the current time, "date" is the current date, and "x.y" is the current SEQUEL release level. Figure A illustrates a sample logon interaction where the valid identification is TEST and the valid password is XYZ.

Logging Off the System

Logoff is achieved by entering the word OFF, either at the TCL level or at the DEBUG level. A message indicating the connect time (i.e., number of minutes you were logged on) and the appropriate charge units will be displayed. The system then displays the LOGON PLEASE message and waits for the next session to be initiated. The general form of the logoff message is:

```
*****************************************************
***      CONNECT TIME AT 14:38:19 = 1533 MINUTES    ***
***   CPU MS.  = 253           DISC READS = 443     ***
***            NUMBER OF ACTIVATIONS = 785          ***
*****************************************************
```

Figures A illustrates sample logon interaction. Figure B illustrates a sample logoff interaction.

```
LOGON PLEASE:TEXT <cr>

USER-ID?

LOGON PLEASE:TEST <cr>
PASSWORD:--- <cr>

***   WELCOME  TO  MICRODATA  S E Q U E L   ***
***   14:07:00   RELEASE 1.1   15 FEB 1983  ***
***       MICROATA SYSTEM SOFTWARE          ***
***           SS20001920  REV B             ***
***             CONSISTING OF               ***
***     PM20001921  REV B  (SYS-GEN)         ***

:
```

FIGURE A. SAMPLE LOGON INTERACTION

```
:OFF  <cr>

*****************************************************
***       CONNECT TIME AT 15:50:00 = 3 MINUTES    ***
***   CPU MS.  = 253         DISC READS = 443     ***
***          NUMBER OF ACTIVATIONS = 785          ***
*****************************************************
***   LOGGED OFF AT 15:50:33   ON 15 FEB 1983     ***

LOGON PLEASE:
```

Figure B. Sample Logoff Interaction

6.2 System Message Formats: Logon Message

Messages in the ERRMSG file have a defined format. The item 'LOGON' in the SYSTEM Dictionary is a prompt message for logging on to the system.

The ERRMSG file contains items which format system messages. This file is improperly called the Error Message File. SEQUEL software uses items in this file to print most of the messages associated with the various processors (see the Appendix). The codes that may be present in one of these items are:

A{(n)}	Parameter insertion code. The next parameter passed to the system message print routine is inserted at this point in the message. If "n" is specified, the parameter is left-justified in a field of "n" blanks.
D	Adds system date in the format DD MMM YYYY.
E{char-string}	The message item-id, surrounded by brackets, is printed followed by the optional character string.
H{char-string}	The character string is printed. If the string is to be a prompt, a plus sign (+) may be added to hold the cursor on the current line.
L{(n)}	The specified number of line feeds are output (one if "n" not specified).
R{(n)}	As in "A" above, except the parameter is right-justified in a field of "n" blanks.
S{(n)}	The specified number of spaces (counting from the beginning of the line, not from current position) is skipped. If the end of the line is reached, the line is printed and a new line is started.
T	Adds system time in the format HH:MM:SS.
X	Skips the next parameter.

The item called 'LOGON' in the System Dictionary follows this format. This message is the prompt invitation to log on to the system. Figure A displays a sample logon item in the System Dictionary. Figure B lists error message 335; you see this message immediately after logging on to the system.

WARNING: Items in the System Dictionary must not be altered while other users are logged on because the position of items might change, invalidating critical pointers.

```
    LOGON
001 L(2)
002 T
003 S(21)
004 D
005 L
006 HLOGON TO THE DEVELOPMENT SYSTEM: +
```

Figure A. Sample Alternate Logon Message in System Dictionary

```
    335
001 L(2)
002 H   ***   WELCOME  TO  MICRODATA   S E Q U E L    ***
003 L
004 H   ***
005 T
006 S(20)
007 HRELEASE 1.1
008 S(35)
009 D
010 H   ***
011 L
012 H   ***       MICRODATA SYSTEM SOFTWARE          ***
013 L
014 H   ***           SS20001920  REV B               ***
015 L
016 H   ***            CONSISTING OF                 ***
017 L
018 H   ***       PM20001921  REV B  (SYS-GEN)        ***
019 L
```

Figure B. Listing of ERRMSG 335

6.3 Logon PROC and General System Messages

Upon logon, SEQUEL allows for the execution of a PROC with an item-id identical to your identification. SEQUEL also allows a general message to be sent to all users as they log on to the system.

The Logon PROC

When you have logged on to your account, SEQUEL permits the automatic execution of a PROC whose item-id is the same as your identification. That is, the Master Dictionary of the account is checked for a PROC matching the identification which was used to log onto the account; if found, it is executed. (For information regarding PROCs, refer to the MICRODATA PROC Programming Manual.)

Typically, the Logon PROC performs standard functions always associated with your particular needs. For example, assume that the PROC listed in Figure A (which includes a TERM operation) is stored as item TEST in your Master Dictionary (M/DICT). If your identification is the word TEST, then the TEST PROC is executed automatically every time you log on (i.e., your particular terminal characteristics are automatically set). This is illustrated in Figure B.

General System Message

SEQUEL allows you to send the same message to all users as they log onto the system. The item 'LOGON' in the ERRMSG file may define such a message which is typically used to transmit information pertaining to system up-time, and the like. It should be noted that the LOGON message item must be present in the ERRMSG file even if no general system message is to be sent; in this case, the item should have no attribute values (i.e., an item-id only). The format of the LOGON message item is the same as any other message item in the ERRMSG file (see the topic SYSTEM MESSAGE FORMATS in this chapter).

```
Item 'TEST' in M/DICT

001  PQN
002  HTERM 118,44,7,6
003  P
004  X***  TERMINAL CHARACTERISTICS SET  ***
```

Figure A. Sample Logon PROC

```
LOGON PLEASE:   TEST,XYZ  <cr>  <-------------  Logon sequence.

***   WELCOME TO MICRODATA  S E Q U E L   ***
***  17:09:50  RELEASE 1.1  15 FEB 1983   ***

***  TERMINAL CHARACTERISTICS SET ***  <-------  Message from TEST PROC.

:  <-------------------------------------------  TCL prompt character.
```

Figure B. Automatic Execution of Sample PROC from Figure A

6.4 Computer Usage: CHARGES and CHARGE-TO Verbs

The CHARGES verb prints the current computer usage since logon. The CHARGE-TO verb charges the current computer usage since logon to a specified project or account.

CHARGES Verb

The CHARGES verb displays the current computer usage since logon as connect time in minutes, CPU milliseconds, the number of disc reads, and the number of process activations (number of timeslices). The general form of the verb is:

```
CHARGES
```

CHARGE-TO Verb

The CHARGE-TO verb keeps track of computer usage for several projects associated with the same LOGON name. The general form of the verb is:

```
CHARGE-TO account-name
```

This verb performs the following functions:

1. Terminates the current charge session by updating the Accounting History File with the appropriate connect time in minutes, CPU milliseconds, and the number of disc reads.

2. Changes the logon name in the Accounting History File to the original name concatenated with an asterisk and the name following CHARGE-TO.

For example, if you are currently logged on to SYSPROG, and you type in:

```
CHARGE-TO PROJECT1 <cr>
```

the LOGON name in the Accounting History File will be changed to "SYSPROG*PROJECT1".

Figure A shows the general form of the CHARGES and CHARGE-TO verbs. Figure B shows examples of the verbs.

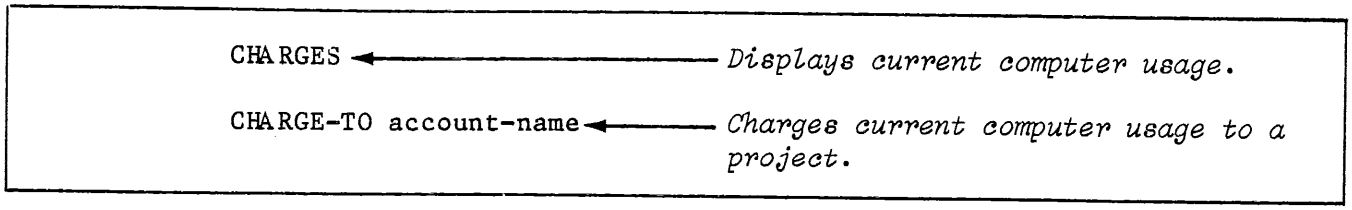


Figure A. General Form of CHARGES and CHARGE-TO Verbs

```
:CHARGES  <cr>

*******************************************************
***       CONNECT TIME AT 14:38:19 = 33 MINUTES     ***
***    CPU MS. = 253         DISC READS = 443       ***
***             NUMBER OF ACTIVATIONS = 173         ***
*******************************************************

:

:CHARGE-TO PROJECT1  <cr>

*******************************************************
***       CONNECT TIME AT 14:39:41 = 34 MINUTES     ***
***    CPU MS. = 261         DISC READS = 449       ***
***             NUMBER OF ACTIVATIONS = 197         ***
*******************************************************

:
```

Figure B. Examples of CHARGES and CHARGE-TO Verbs

6.5 LOGTO Verb

The LOGTO verb allows you to log to another account without first having to log off. The work space assigned to the original account is retained.

The general form of the verb is:

```
LOGTO account-name{,password}
```

where "account-name" is the account to log to. After entering the LOGTO command, the following occurs:

1. The system checks to see if the specified account is appropriately defined in the System Dictionary.
2. If you did not enter a password and one is required, you are then prompted for the passwword.
3. The accounting statistics are updated if being kept for the present account.
4. The security codes and system privilege level are changed to those of the new account.
5. The work space assignment of the original account is retained.
6. The spooler assignment and device attachment are retained.
7. If found, the Logon PROC is executed.

The LOGTO verb must not be executed from a PROC. Figure A shows the form of the LOGTO verb. Figure B shows an example of the LOGTO verb.

```
LOGTO account-name{,password}
```

Figure A. General Form of LOGTO Verb

```
:LOGTO SYSPROG,PWD <cr>

*********************************************************
***       CONNECT TIME AT 14:38:19 = 1533 MINUTES     ***
***    CPU MS. = 253              DISC READS = 443     ***
***            NUMBER OF ACTIVATIONS = 785             ***
*********************************************************

:
```

Figure B. Sample Usage of LOGTO Verb

6.6 User Identification Items

Each user has a user identification item stored in the System Dictionary. These items define users who can log on to the system. User identification items are either file definition items or file synonym definition items.

User identification items are initially created via the CREATE-ACCOUNT PROC (see Chapter 11, SYSTEM MAINTENANCE PROCEDURES). These items may subsequently be updated via the EDITOR (see the Microdata EDITOR Programming Manual). Entries in the System Dictionary should only be updated (from the SYSPROG account) when no other users are logged on to the system. This is because the system software maintains pointers to data in the System Dictionary when users log on, and updating the System Dictionary will invalidate the pointers. An exception to this rule is when creating a new account (or a synonym to an existing account), which can be done at any time since new items are added to the end of the existing System Dictionary data, and thus do not disturb any existing items.

User Identification Attributes

Attributes 5 through 8 of a user identification item contain data associated with a user's security (lock) codes, password, and privileges:

Attribute 5: Contains the set of retrieval lock-codes associated with a user. Multiple values (separated by value marks) are allowable. There is no restriction on the format of individual lock-codes. This attribute may be null, indicating no lock-codes. (Lock-code usage is described in the topic SECURITY in this chapter.)

Attribute 6: Contains the set of update lock-codes associated with a user (same as described for retrieval lock-codes above).

Attribute 7: Contains a user's password, which is a single value. This attribute may be null. There is no restriction as to the format of the password.

Attribute 8: Contains a code which indicates the level of "system privileges" and work space assigned to a user. Three levels of system privileges are available: they are referred to as zero (lowest), one, and two (highest). Lower levels of system privileges restrict usage of certain facilities of the system, as described in Figure B. SYS0 designates Level 0, SYS1 designates Level 1 and SYS2 designates Level 2. The system privilege code may be immediately followed by an "additional work space" parameter enclosed in parentheses (see the topic, WORK SPACE ASSIGNMENT in this chapter).

Attribute 9: May contain the code "U" to indicate that logon/logoff times should be collected in ACC file by the system.

Attributes 1 through 4 and Attribute 13 are as defined for regular file definition items (see the topic FILE DEFINITION ITEMS in Chapter 3). Figure A shows a sample user identification item (for user XYZ).

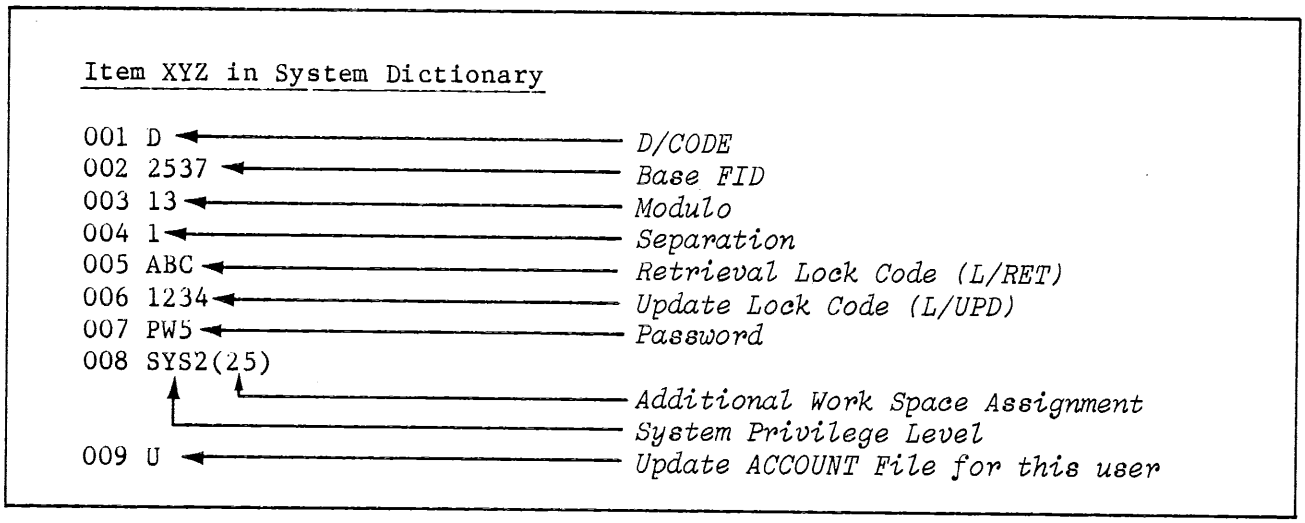


Figure A. Sample User Identification Item (For User XYZ)

FACILITY	LOWEST PRIVILEGE LEVEL REQUIRED
Updating of M/DICT	One
Use of magnetic tape	One
Use of DEBUG (other than P, OFF, END and G commands)	Two
Use of DUMP processor	Two
Use of assembler and loader	Two
Use of FILE-SAVE and FILE-RESTORE processors	Two

Figure B. Required System Privilege Levels

6.7 Work Space Assignment

Additional work space may be specified immediately following the system privilege code in Attribute 8 of the user identification item.

There are three "linked" work spaces symbolically named HS, IS, and OS. These work spaces are set to an initial size of six frames each (3000 bytes) at logon time. Some users require additional work space to run DATA/BASIC programs and manipulate larger items; this requirement may be specified in Attribute 8 of the user identification item, immediately following the system privilege code.

The format of the "additional work space" parameter is:

(n)

where "n" is the work space requirement (in number of frames) for each of the three linked work spaces (HS, IS, and OS). The "n" parameter must be a decimal number greater than six (because six frames are initially made available by the system) and no more than 133. For example, if the requirement for line 8 is 40 frames, (40-6)*3=102 frames must be available as a contiguous block.

Additional work space is obtained from the pool of contiguous overflow space. The first priority is to obtain this work space from low disc space (see topic LOW DISC SPACE AND FILE AREA). If no space is available, the following message is displayed immediately after the "WELCOME TO SEQUEL" message:

334 REQUESTED NUMBER OF ADDITIONAL WORK-SPACE FRAMES: xxxx IS NOT AVAILABLE. ADDITIONAL WORK-SPACE HAS NOT BEEN ASSIGNED.

where "xxx is the amount of work space entered in Attribute 8 of the user identification item.

If the format of the work space parameter is illegal (e.g., "n" out of range, missing parenthesis, nonnumeric "n", etc.), then the after message is returned immediately following the "WELCOME TO SEQUEL" message:

333 THE FORMAT OF THE ADDITIONAL WORK-SPACE PARAMETER: xxxx IS ILLEGAL FOR THIS ACCOUNT NAME.
ADDITIONAL WORK-SPACE HAS NOT BEEN ASSIGNED.

where "xxxx" is the amount of work space entered in Attribute 8 of the user identification item.

Consider the sample situation where the entry in the eighth attribute of a user identification item is SYS1(10). This gives the user level one system privileges and 10 frames of work space per linked set. Thus, 12 additional frames have to be obtained from contiguous overflow space. This situation is illustrated in Figure A. It is assumed that the user logs in on process one which corresponds to a PCB FID of X'220' (544). The arrows in the figure represent frame linkages set up after logon.

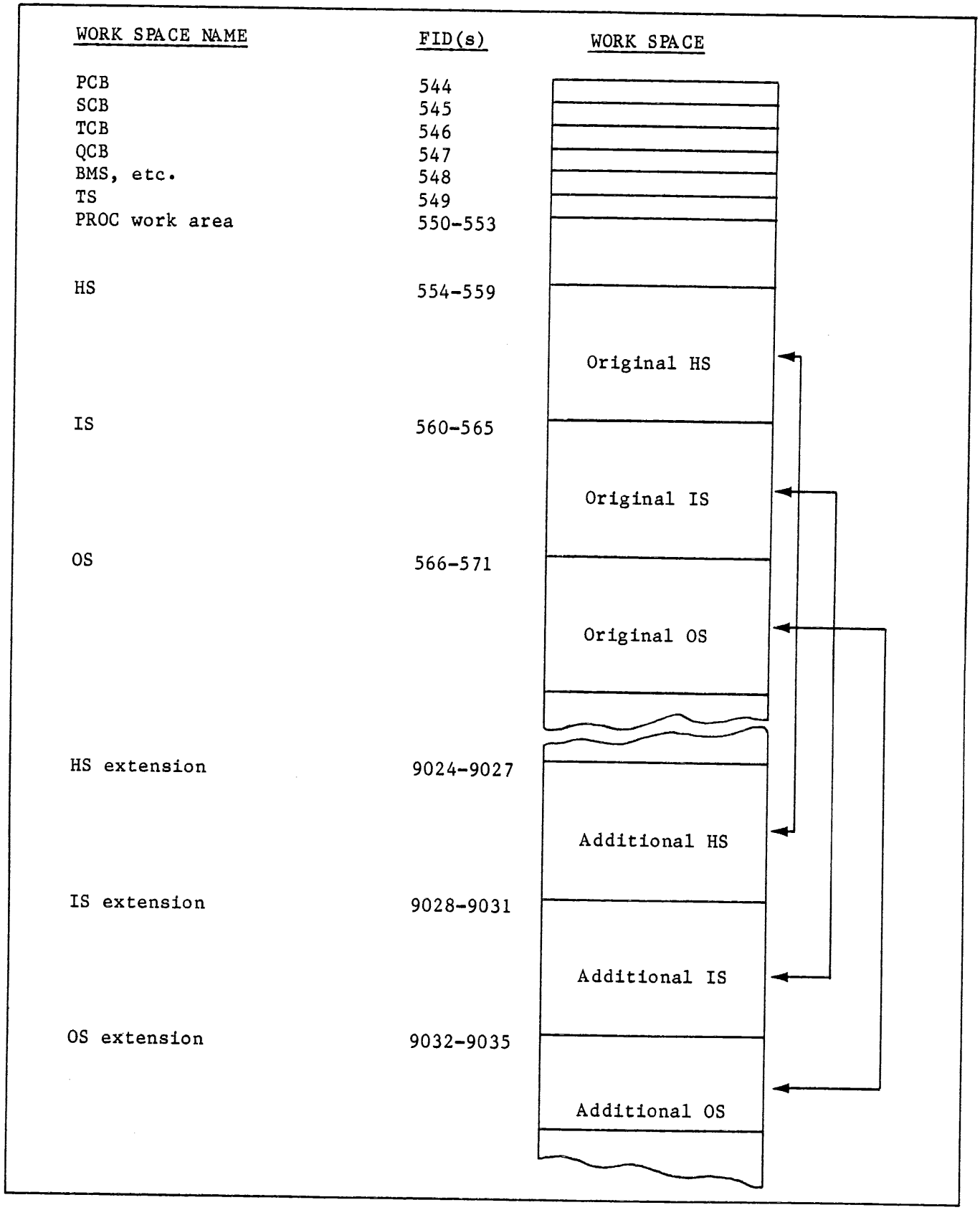


Figure A. Sample Additional Work Space Assignment

6.8 Allocating Permanent Blocks of Additional Work Space

You may allocate a permanent block of additional work space via the WORKSPACE verb. The Logon Work Space Table (LWST) keeps track of each "block" of space by listing the starting and ending frame id, the size of the block, its status, whether the block is linked, and the port currently using the block (if any).

The WORKSPACE Verb

This verb allocates and deallocates permanent blocks of additional work space. Its general format is:

```
WORKSPACE size {(D) or (quantity) or (D,quantity)}
```

where:

size	allocates one block of space three times 'size' from low disc space. This number corresponds to the number of additional frames (less six) requested on Line 8 of the account definition item in the System Dictionary.
(D)	deallocates one block of space and returns it to overflow.
(quantity)	specifies the number of ports that will be allocated (or deallocated) the specified block of space.

Figure B shows several examples of how the WORKSPACE verb is used.

The LOGON Work Space Table (LWST)

This table defines the permanently allocated blocks of work space. Logged in this table are the starting and ending frame ids of each block and the status of that block. The integrity of the block links is guaranteed because this table is written back to disc every time the table is updated (i.e., this space is not lost following a coldstart).

The LWST provides the added benefit of fast logon. Blocks of space are taken from available space and linked the first time they are used, rather than every time a terminal is logged on. Therefore, when a terminal is logged on, the additional work space frames do not have to be read and linked.

Permanent Blocks

When you log on, the system first attempts to use a block of space permanently allocated with the WORKSPACE verb. The LWST is searched for the smallest block of space not "IN USE" by another terminal that satisfies the size requested by Line Eight of the account definition item in the System Dictionary. If one is found, it is marked "IN USE" and linked to the terminal's primary work space.

Permanent blocks of space are not linked until the first time they are used. Therefore, the first time you log on after a coldstart or file restore, there is some delay while these frames are contiguously linked. Subsequently, they will already be linked and you will be immediately logged on. Permanent blocks of space may also be marked "RELINK" when you log off from the assembly or DATA/BASIC debugger or when the 'L' option is used with the OFF command. If you are concerned about work space links, press the <BREAK> key, enter "OFF (L)", and then logon to relink your work space.

Temporary Blocks

When a suitable block of permanently allocated space is not available, a temporary block of overflow space is used. Note that you will encounter a delay during logon while the temporary space is contiguously linked. When you log off, the temporary block is returned to overflow. Coldstarting the system while assigned to a temporary block causes the system to lose that block of space.

To view the current LWST table, execute the WORKSPACE verb without any parameters or options (see Figure C).

The SYSTEM-SETUP PROC has been modified to reserve a maximum size block (133) for each terminal on the system. You should modify this PROC to accommodate your system needs.

```
WORKSPACE size {(D) or (quantity) or (D,quantity)}
```

Figure A. General Format of the WORKSPACE Verb

WORKSPACE 66	*Permanently allocates one block of 192 frames (66 x 3 - 6) from Low Disc Space. The number 66 corresponds to the number found in Line Eight of the account definition item in the SYSTEM Dictionary. If a temporary block of 192 frames is found before a null LWST entry is encountered, it is converted to a permanent block.*
WORKSPACE 33 (10)	*Allocates 10 blocks, each of which is 93 frames.*
WORKSPACE (D)	*Deallocates one block of space and returns it to overflow. If the first permament block found is "IN USE", it will be converted to temporary space and returned to overflow when the line using it logs off.*
WORKSPACE (D,5)	*Deallocates 5 blocks of space.*
WORKSPACE 66 (D)	*Deallocates a block of space that contains 192 frames. This command has no effect if a block of 192 frames cannot be found.*
WORKSPACE 33 (D,5)	*Deallocates 5 blocks of space that each contain 93 frames.*

Figure B. Sample Usage of the WORKSPACE Verb

```
:WORKSPACE <cr>

        FROM        TO        SIZE/3   STATUS   LINKED   PORT
        7307 -      7396        30       T        Y       63   IN USE
        8409 -      8468        20                Y
        9176 -      9220        15                Y
        9300 -      9680       127                Y       12   IN USE
        9681 -     10061       127
```

Note: The "T" listed under STATUS means that block of space is temporary and will be returned to overflow when the user logs off.

Figure C. Sample LOGON Work Space Table (LWST)

6.9 Security

Security codes may optionally be placed in the L/RET and L/UPD attributes of a dictionary item to restrict access and update. At logon time, you are assigned the set of security codes which are in your user identification item. During the session, whenever an L/RET or L/UPD code is encountered, a search is made of the assigned codes for a match. If no match is found, you are denied access. A code may consist of any combination of ASCII characters.

L/RET and L/UPD

Both file definition ("D" code) and synonym file definition ("Q" code) items have L/RET (retrieval locks) and L/UPD (update locks) attributes. When these attributes have stored values, they are known as security codes. Although there is no prohibition against multiple values for these attributes, only the first attribute value is matched against the user assigned codes. Since each file may be individually locked for both update and retrieval, your user identification item must contain the same security codes which protect the data you wish to access. Using this feature, a complex "mask" can be constructed for each user, giving you a different subset of files which you may access.

Security at the file level is activated at the processor level. The following processors are assumed to be updating processors and, therefore, require a match on the L/UPD attribute in the file definition item: COPY, EDITOR, and the Assembler. Other processors, with the exception of DATA/BASIC, are assumed to be retrieval processors and require a match on the L/RET attribute in the file definition item. (See Microdata DATA/BASIC Programming Manual for an explanation of DATA/BASIC security features.) Failure to match one of the user security codes with either the L/RET or L/UPD attribute value generates the following message (and returns control to TCL):

```
[210] FILE file-name IS ACCESS PROTECTED
```

User Assigned Codes

Each user identification item in the System Dictionary (see the topic USER IDENTIFICATION ITEMS in this chapter) contains the list of security codes assigned for that particular user. The lock code in the file being verified must match.

Security codes may be assigned initially when an account is created via the CREATE-ACCOUNT PROC (refer to Chapter 11). Security codes may be added or deleted by updating the appropriate user identification item using the EDITOR (assuming one has the appropriate security codes); however, updates to the user identification item should only be performed when no one else is logged on to the system.

Care must be taken when securing the System dictionary and the SYSPROG account. SYSPROG (and any synonyms, such as SP) should be protected first. Then SYSTEM may be secured, usually with the same codes as SYSPROG. Remember that there are two items in SYSTEM that should be secured with the same codes; their item-ids are SYSTEM and DL/ID. If SYSTEM were to be secured before SYSPROG, then SYSPROG would not be able to access SYSTEM.

Security Code Comparison

Security codes are verified by comparing the value in the file dictionary against the corresponding string of values in the user identification item. Characters are compared from left to right. An equal (verified) compare occurs when the value in the file dictionary is exhausted and all characters match up to that point. This is illustrated in Figure A.

When referencing a file using a .Q synonym (refer to Chapter 3), a security code match is made at all levels (i.e., SYSTEM, M/DICT, and file dictionary). Therefore, a correspondence must be maintained at all levels to process the Q synonym files. Because the user identification item for the account containing the primary file is verified for security codes, the user referencing the Q synonym must have a code defined in this user identification item which will verify with the first code in the equated account's user identification item. Thus, in a user identification item, only the first code is used to protect the account from Q synonym accesses, while all the codes in the item are assigned to you when you log on.

FILE DICTIONARY CODE	USER IDENTIFICATION CODE	RESULT
123	123	MATCH
12	123	MATCH
123	12	NO MATCH
XYZ	XYZ5	Match
AQ2	AQ	No match

Figure A. Sample Security Code Comparisons

6.10 Accounting History File: Introduction

The Accounting History File is one of the mandatory files in the SEQUEL system. This file contains accounting history for the system, as well as entries that describe currently active (logged-on) users.

The System Dictionary must contain the file definition item ACC and the file synonym definition item ACCOUNT for the Accounting History File, as illustrated in Figure A. (The System Dictionary must also contain a duplicate D-pointer 'CHANNEL' to ACC.) The ACC dictionary is set up for examining and listing the data in Accounting History File via ENGLISH (see the topic titled ACCOUNTING HISTORY FILE: SUMMARY AND EXAMPLES in this chapter). There are two types of entries (items) in the Accounting History File: those that represent active (logged-on) users, and those that keep track of accounting history.

Active Users Items

The item-id of an active user item in the Accounting History File is the four-character hexadecimal FID of the PCB of the user's process. If the PCBs start at FID 512, (they proceed in steps of 32 frames from there on), a user logged on to process zero will have an entry with an item-id "0200" (512), while a user logged on to process one will have an entry with an item-id "0220" (544), and so on. Attribute 1 of an active user item contains the name of the user (i.e., the item-id of the user identification item), Attribute 2 the date logged on, and Attribute 3 the time logged on. Active user items are created when a user logs on, and deleted when he logs off. (Attributes of an active user item are summarized in the topic titled ACCOUNTING HISTORY FILE: SUMMARY AND EXAMPLES in this chapter.)

Accounting History Items

The item-id of an accounting history item is the name of the user (i.e., the item-id of the user identification item). Attributes 1, 2, and 3 are not used. The remainder of the attributes are described below:

Attribute 4: Date(s) logged on. Each unique date is stored. Value marks are tagged on to the value in this attribute if multiple logoffs occur on the same date (for LIST alignment purposes). Date is stored in SEQUEL date format.

Attribute 5: Time(s) logged on. An entry is made for each logoff, representing the time at which you logged on. Time is represented in seconds past midnight (24-hour clock).

Attribute 6: Connect time(s). This entry represents the time in seconds between the logon and logoff.

Attribute 7: Charge-units. A number representing the CPU usage (CPU milliseconds) is added for each logoff.

Attribute 8: Disc-reads. This entry reflects the number of frame faults encountered between logon and logoff.

Your System dictionary item must have a 'U' in Attribute Nine if you are to have your account file history items updated. The entries in the account file contain the history of each session (logon to logoff). If the System dictionary data has been changed since logon or the history item to be updated is too large for your work space, the following message is printed:

ACCOUNT FILE STATISTICS WERE NOT UPDATED DUE TO EITHER:

1. INSUFFICIENT WORKSPACE TO CONTAIN THE ACCOUNT FILE ITEM, OR
2. SYSTEM DICTIONARY CHANGED WHILE YOU WERE LOGGED ON

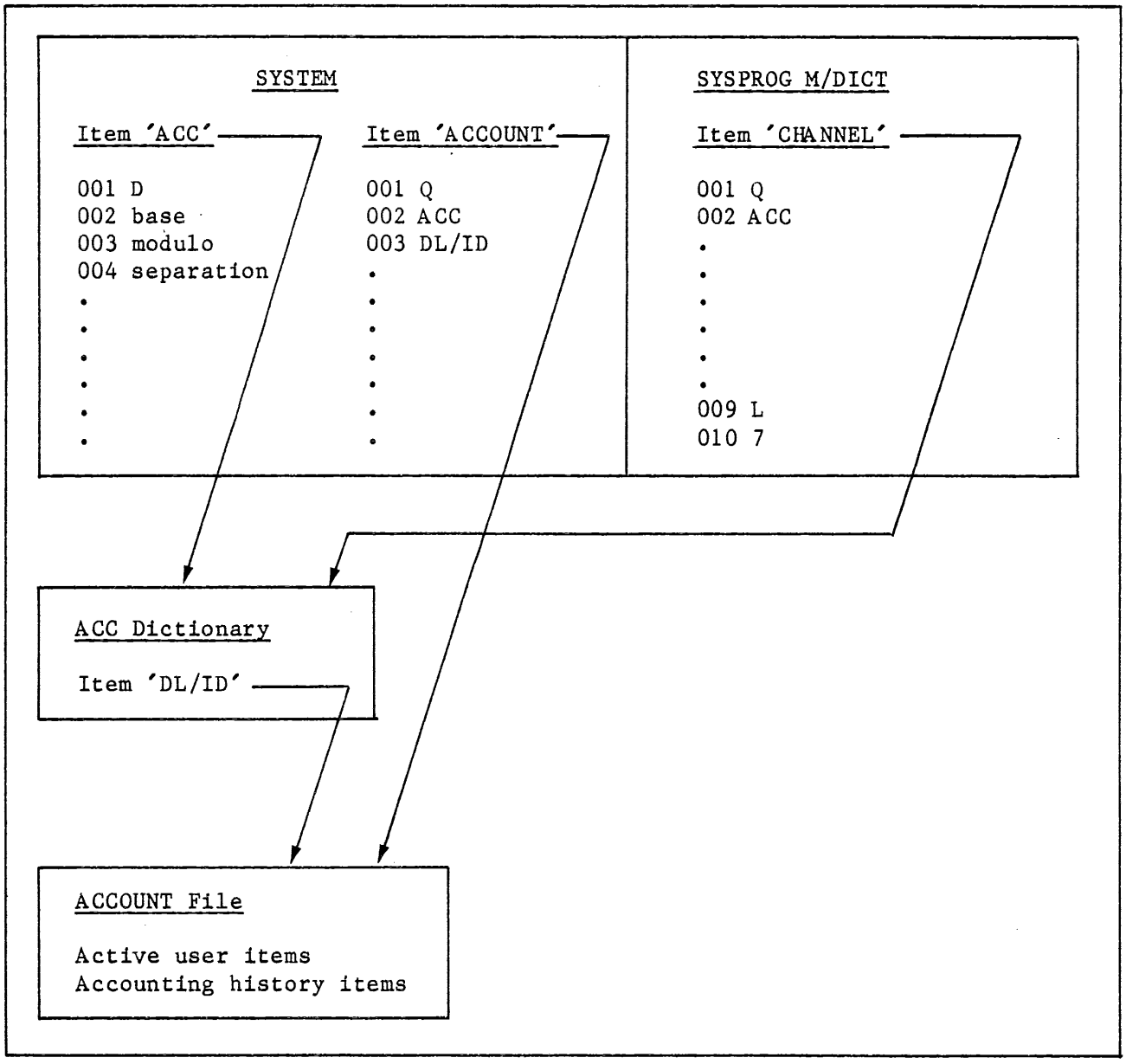


Figure A. Accounting History File Hierarchy

6.11 Accounting History File: Summary and Examples

This topic summarizes formats of the active user items and accounting history items in the Accounting History File. Also presented are sample entries for the Accounting History File.

Figure A summarizes attributes for the active user items and accounting history items. Figure B shows a sample sorted listing of the active users (users with a value for Attribute A1) via an ENGLISH SORT statement. Figure C shows a sample listing of the accounting history item for user FRANK via an ENGLISH LIST statement. This same item is "dumped" in Figure D to show its internal storage format.

ATTRIBUTE NUMBER	'ACC' DICTIONARY NAME	ACTIVE USER ITEM	ACCOUNTING HISTORY ITEM
0	(Item-id)	Four-character hexadecimal PCB-FID	User name
1	A1	User name	Not used
2	A2	Date logged on	Not used
3	A3	Time logged on	Not used
4*	DATE		Dates logged on
5*	TIME		Times logged on
6*	CONN		Connect times (seconds)
7*	UNITS		Charge units (CPU milliseconds)
8*	DISC-READS		Disc reads
	TOT-CONN		Total connect times
	TOT-UNITS		Total of CPU milliseconds
	TOT-DISC-READS		Total of disc-reads

*Multivalued attribute; one value for each LOGON/LOGOFF sequence

Figure A. Summary of Active User Items and Accounting History Items

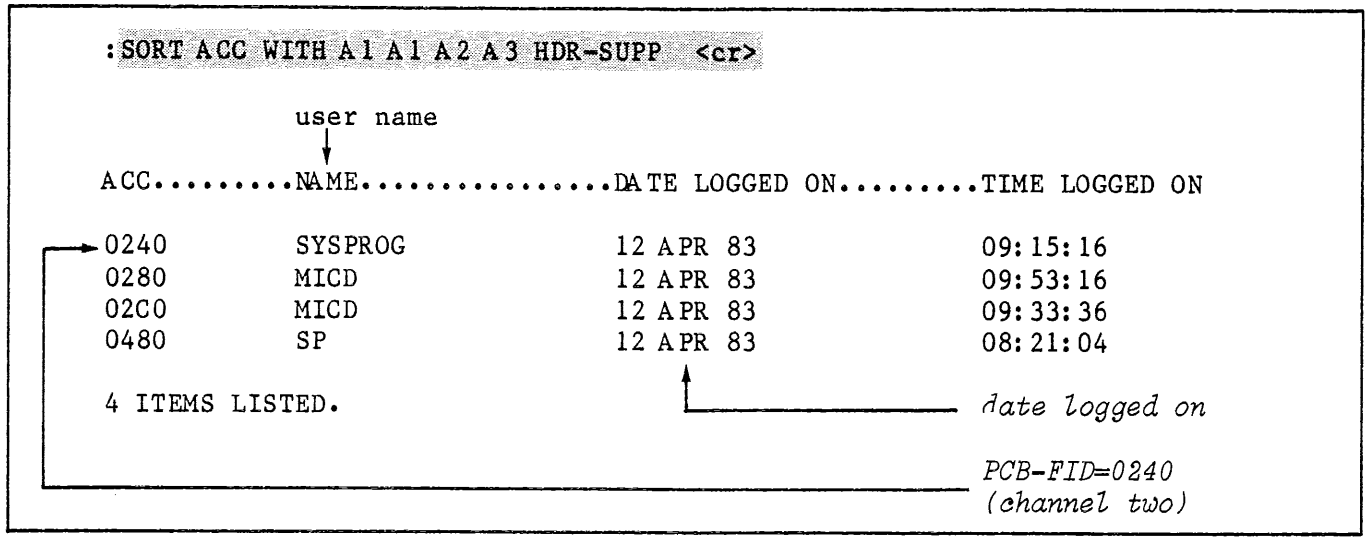


Figure B. Sorted Listing of Active Users

```
:LIST ACC 'FRANK' HDR-SUPP <cr>

ACC.......DATE.....TIME.....CONNECT..CHRGE DISC....TOTAL.....TOTAL..TOTAL
          LOGGED ON LOGGED ON TIME     UNITS READS   CONNECT     UNITS  DISC
                                                     TIME               READS

FRANK     10 APR 83 08:38:43  00:32:12   302   2107     02:01:57   1217  7189
                    09:11:51  00:05:00    58    307
                    09:17:03  00:32:33   321   2009
          11 APR 83 10:50:00  00:03:20    30    121
                    11:06:40  00:08:20    83    413
          08 JUN 83 11:08:31  00:40:32   423   2232
```

Figure C. Listing of Accounting History Item for User FRANK

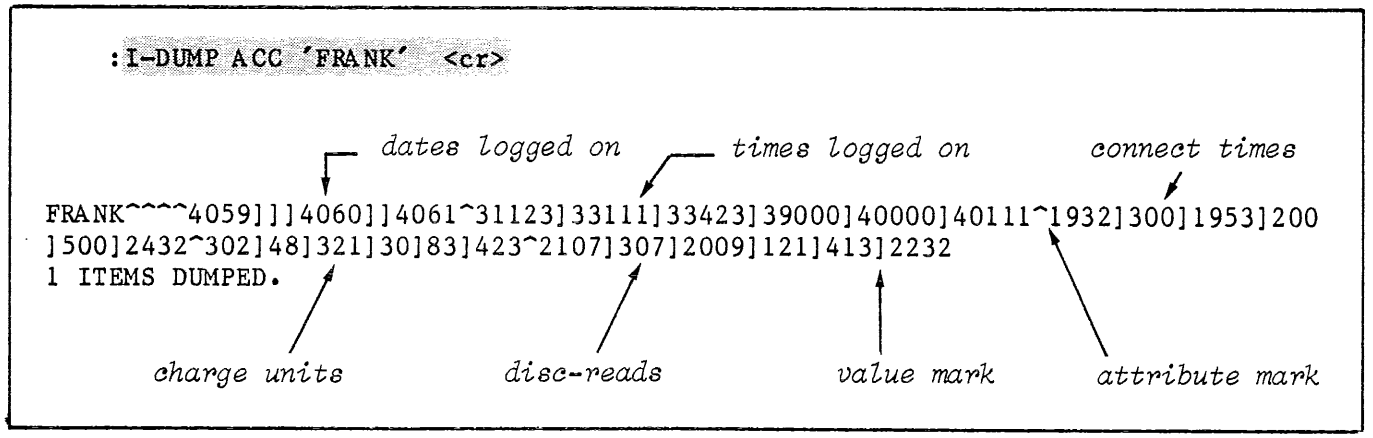


Figure D. Dump of Accounting History Item for User FRANK

6.12 Accounting History File: Periodic Clearing

To avoid overflowing the accounting history item in the Accounting History File for a specific user, the item should be periodically cleared.

The point of overflow is determined by the activity of your account together with the amount of work space assigned to you. This point can be calculated by following the procedure detailed in Figure A.

To clear your accounting history item, follow the steps detailed in Figure B.

If the accounting history item for your account exceeds the available workspace, you can log off, but the Accounting History File will not be updated. To recover from this situation, follow the procedure detailed in Figure B.

1. Type the following:

   ```
   :CT SYSTEM user-name <cr>
   ```

   This lists the user identification item for the specified user. Attribute 8 shows the system level privilege followed by (in parentheses) the amount of work space. If a value is not shown for the work space, 6 is assumed. Multiply this value by 100 to compute an approximate maximum size.

2. To determine the current size, type:

   ```
   :STAT ACC ´user-name´ <cr>
   ```

   This produces the following output:

   ```
   STATISTICS OF ACC:
   TOTAL = xxx AVERAGE = yyy COUNT=1
   ```

3. If the value displayed for TOTAL in Step 2 (i.e., "xxx") approaches the value calculated in Step 1, then the user-account is approaching the overflow point.

Figure A. Procedure to Determine Overflow Point

1. Logon to the SYSPROG account
2. Type the following:

   ```
   :LISTACC user-name (P) <cr>
   ```

3. Type the following:

   ```
   :DELETE ACC user-name <cr>
   ```

Figure B. Procedure to Clear Specific Accounting History Item

7 FILE MANAGEMENT PROCESSORS

7.1 Overview

This chapter describes the data base management processors for the SEQUEL system.

The file management processors provide capabilities for creating, managing, and manipulating files and items within the SEQUEL system. File management processors include the CREATE-FILE processor, CLEAR-FILE processor, DELETE-FILE processor, and COPY processor.

CREATE-FILE Processor

The CREATE-FILE processor generates new dictionaries and/or data files. The processor establishes file dictionaries by creating "D" entries in your Master Dictionary (M/DICT) and reserving and linking primary file space. You need only specify values for the desired modulo(s) and separation(s).

CLEAR-FILE Processor

The CLEAR-FILE processor eliminates the data from a file (i.e., it sets the file to the "empty" state by placing an attribute mark in the first data position of each group of the file). "Overflow" frames that may be linked to the primary file space will be released to the system's overflow space pool. Either the data section or the dictionary section of a file may be cleared.

DELETE-FILE Processor

The DELETE-FILE processor deletes a file. Either the data section or both the dictionary and data sections (if present) of the file may be deleted. Frames are returned to available space.

COPY Processor

The COPY processor copies an entire file or selected items from the file to the terminal, to the printer, to another file or to items with different item-ids within the same file.

Examples

As a general introduction to this chapter, Figure A presents a number of examples illustrating the use of file management processors.

Additional File Management Procedures

Additional file management procedures (such as the creation of new user-accounts, saving and restoring of files, etc.) are detailed in Chapter 11, SYSTEM MAINTENANCE PROCEDURES.

EXAMPLE	EXPLANATION
:CREATE-FILE (DICT TEST 1,1) <cr>	*Creates a file dictionary for the TEST file, with a modulo of 1 and a separation of 1.*
:CREATE-FILE (DATA TEST 7,2) <cr>	*Reserves disc space for the data area of the TEST file, with a modulo of 7 and a separation of 2.*
:CREATE-FILE (FNA 1,1 3,1) <cr>	*Creates a file dictionary for the FNA file, with a modulo of 1 and a separation of 1. Also creates the data area of the FNA file, with a modulo of 3 and a separation of 1.*
:CLEAR-FILE (DATA XYZ) <cr>	*Clears data section of XYZ files.*
:DELETE-FILE (DICT INV) <cr>	*Deletes dictionary section of INV file (assuming NO DATA SECTION).*
:DELETE-FILE (FAE) <cr>	*Deletes data and dictionary section of FAE file.*
:COPY TEST I1,I2,I3 <cr> TO:X1 X2 X3 <cr>	*Copies data item I1, I2, and I3 back into the same file (TEST) giving duplicate items with item-ids of X1, X2 and X3.*
:COPY DICT SAMPLE * <cr> TO:(DICT FLAVORS) <cr>	*Copies all dictionary items from file SAMPLE to dictionary of file FLAVORS.*
:COPY TEST * <cr> TO: <cr>	*Copies all items in the TEST file to your terminal.*

Figure A. Sample Usage of File Management Processors

7.2 Creating New Files: CREATE-FILE Processor

The CREATE-FILE processor generates new files and dictionaries in the SEQUEL system.

The CREATE-FILE processor establishes file dictionaries by creating "D" entries in your Master Dictionary (M/DICT). The CREATE-FILE processor is also used to reserve disc space for the data portion of the new file by automatically placing the DL/ID (Data-Level Identifier) entry in the file dictionary, which then points to the data area. CREATE-FILE will automatically locate and reserve a contiguous block of disc frames from the available space pool. You need only specify values for the modulo and the separation of both the file dictionary and data area (Figure A). (For a discussion of the values to use for modulo and separation, refer to the topic SELECTING MODULO AND SEPARATION in this chapter.)

The dictionary portion (DICT) of the file must always be created first. The name given to the new file must not exist in the user M/DICT. The general form for creating a file dictionary is:

```
CREATE-FILE (DICT file-name m,s)
```

where "file-name" is the name given to the file, "m" is the desired modulo, and "s" is the desired separation. An example is shown in Figure B, where the DICT for a file named SAMPLE is being created.

Once the DICT has been created, the primary file space for the data area of the file can be reserved. The general form is:

```
CREATE-FILE (DATA file-name m,s)
```

where "file-name" is the name given to the file, "m" is the desired modulo, and "s" is the desired separation. An example is shown in Figure C. (Note that a data area need <u>not</u> be reserved for a single-level file where data is to be stored in the DICT.)

The most common form of the CREATE-FILE command is shown below. This enables the creation of both the dictionary and data areas with one command. The general form is:

```
CREATE-FILE (file-name m1,s1 m2,s2)
```

where "file-name" is the name of the file, "m1" and "s1" are the modulo and separation of the DICT (respectively), and "m2" and "s2" are the modulo and separation of the data area. This usage is shown in Figure D.

The BREAK key is inhibited during a CREATE-FILE.

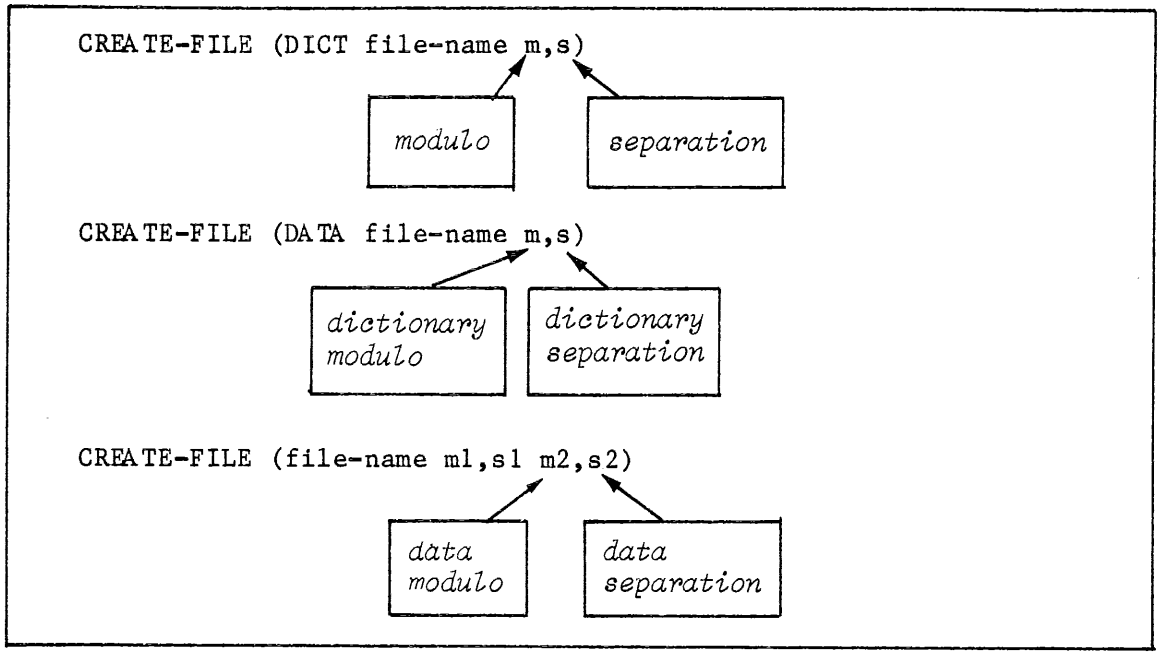


Figure A. General Forms of CREATE-FILE Command

```
:CREATE-FILE (DICT SAMPLE 1,1)  <cr>

[417] FILE 'SAMPLE' CREATED; BASE = 14946, MODULO = 1,SEPAR = 1.
```

Figure B. Sample Creation of Dictionary Portion

```
:CREATE-FILE (DATA SAMPLE 11,1)  <cr>

[417] FILE 'DL/ID' CREATED; BASE = 14947, MODULO = 11, SEPAR = 1.
```

Figure C. Sample Creation of Data Portion

```
:CREATE-FILE (SAMPLE2 1,1 11,1)  <cr>

[417] FILE 'SAMPLE2' CREATED; BASE = 14980, MODULO = 1, SEPAR = 1.
[417] FILE 'DL/ID' CREATED; BASE = 15132, MODULO = 11, SEPAR = 1.
```

Figure D. Sample Creation of Both Dictionary and Data Portions

7.3 Selecting Modulo and Separation

Efficient file accessing and disc use depends on proper selection of modulo and separation.

"Modulo" is the number of groups in a file; "separation" is the number of contiguous frames per group. A file is created by specifying its modulo and separation parameters; the frames allocated by the system (modulo*separation) are referred to as the "primary" file-space. As data is placed into the file, any group may overflow by attaching frames from the available system space pool; this space is referred to as the "overflow" file-space. To locate an item, given its item-id, the item-id is "hashed" using the modulo of the file, which results in a unique group number in which it may exist. The item-ids in that group are then linearly searched for the required item. A proper selection of the "modulo" parameter is essential to minimize this search time. The current maximum modulo and separation are 8,388,607 and 127 respectively.

In the current file structure, the SEPARATION parameter is of practically no value. The disc-head will almost definitely have moved between the moment that a process requests one frame of the group and the next; therefore, whether the next linked frame of the group is contiguous (that is, if the separation is one) or not makes a marginal difference. What this amounts to is that there is little difference in accessing a frame in "prime" file-space and in "overflow" space, particularly since the file-restore process "compacts" all files and leaves the "prime" and "overflow" frames close to each other.

Selecting a proper MODULO is extremely important, since the number of groups directly affects the search and update time for an item in the group. What is proposed here is a modulo selection process that would make the average GROUP length between one and two frames (i.e., at most two frame faults to find a particular item). Obviously, if the item-size is of the order of 250 bytes or greater, this rule of thumb must be modified.

The number of frame faults -- the factor that causes the most degradation of overall system response -- increases dramatically as the number of frames per group increases. This is because, on the average, half of the frames in a group have to be written back to the disc after an item update. Thus, to update an item in a group, we have to read every frame in the group and write half of them (see Figure A).

A good rule of thumb is to select the MODULO to be the smaller of:

(1) $\frac{\text{The number of bytes of data}}{500}$ or (2) The number of items

and use a separation of one.

The DATA/BASIC program 'SET-ALLOCATION' in SYSPROG-PL uses these formulas to set the reallocation parameters (contained in attribute 13 of the file definition item) for existing files.

Using the table, one can select an appropriate ITEMS/GROUP value; knowing the expected number of items in the file then gives the approximate modulo. Modulo must not be a multiple of 2 or 5.

Type of Activity	Frames per Group					
	1	2	3	4	5	6
READ	1.0	1.5	2.0	2.5	3.0	3.5
UPDATE	3.0	4.5	6.0	7.5	9.0	10.5

Figure A. Disc Operations Versus Frames/Groups

If Avg Item-Size Is	Then Item/Group Should Be	And Bytes/Group Will Be
20	22.0	440
35	13.0	455
50	9.0	450
75	12.0	900
100	9.0	900
125	7.5	937
150	6.0	900
175	8.0	1400
200	7.0	1400
250	5.8	1450
300	6.4	1920
350	5.5	1925
400	4.8	1920
500	3.8	1900
1000	3.0	3000
5000	0.8	4000

Figure B. Selecting Items/Group

Avg Item Size	Approximate No. of Items		Items/Group (From Figure B)		Approximate Modulo
20	800	/	22.0	=	36
40	5000	/	13.0	=	383
210	1800	/	7.0	=	257
4000	230	/	0.8	=	287

Figure C. Examples of Computing Modulo

7.4 Clearing & Deleting Files: CLEAR-FILE & DELETE-FILE Processors

The CLEAR-FILE processor clears (i.e., empties) files. The DELETE-FILE processor deletes files.

The CLEAR-FILE processor clears the data from a file (i.e., it sets the file to the "empty" state by placing an attribute mark in the first data position of each group of the file). "Overflow" frames that may be linked to the primary file space will be released to the system's additional space pool. Either the data section or dictionary (DICT) section of a file may be cleared using the CLEAR-FILE verb. If the dictionary section is cleared, and a corresponding data section exists (as implied by the presence of the DL/ID item in the dictionary), then the DL/ID will be maintained in the dictionary. The <BREAK> key is inhibited during the DELETE process, but not the CLEAR process.

To clear the data section of a file, the following verb is used:

```
CLEAR-FILE (DATA file-name)
```

To clear the dictionary section of a file, the following verb is used:

```
CLEAR-FILE (DICT file-name)
```

The DELETE-FILE processor erases a file. Either the data section, dictionary section, or both of these file sections may be deleted. The dictionary section of a file which has a data section cannot be deleted until the data section is deleted. All frames no longer required are returned to the available space pool.

To delete the data section of a file, the following verb is used:

```
DELETE-FILE (DATA file-name)
```

which will delete the data section pointed to by the DL/ID and will also delete the DL/ID from the dictionary section.

To delete the dictionary section of a file, the following verb is used:

```
DELETE-FILE (DICT file-name)
```

To delete both the data and dictionary sections, the following verb is used:

```
DELETE-FILE (file-name)
```

In the latter two cases, the file-definition item in the M/DICT (the file-name) is also deleted. Files defined by file-synonym in the user's M/DICT cannot be specified in a DELETE-FILE verb. These definitions should be deleted with the DELETE PROC or the EDITOR.

CLEAR-FILE and DELETE-FILE should not be used on files containing group format errors. If the cause of the group format errors is due to incorrect frame linkages, use of these verbs could destroy data in other files. Either repair the error (refer to the topic GFE'S: FIX-FILE-ERRORS VERB) or delete the file definition item in the M/DICT (all associated space will be lost until a restore is performed).

The CLEAR-FILE and DELETE-FILE verbs are summarized in Figures A and B, respectively. Figure C presents examples.

```
CLEAR-FILE (DATA file-name) <---------- Clears data sections.
CLEAR-FILE (DICT file-name) <---------- Clears dictionary section.
```

Figure A. General Forms of CLEAR-FILE Verb

```
DELETE-FILE (DATA file-name) <---------- Deletes data section.
DELETE-FILE (DICT file-name) <---------- Deletes dictionary section.
DELETE-FILE (file-name) <---------- Deletes data and dictionary
                                    section.
```

Figure B. General Forms of DELETE-FILE Verb

COMMAND	EXPLANATION
:CLEAR-FILE (DATA TESTFILE) <cr>	*Clears data section of file TESTFILE.*
:CLEAR-FILE (DICT EMPL) <cr>	*Clears dictionary section of file EMPL.*
:DELETE-FILE (DATA A/P) <cr>	*Deletes data section of file A/P.*
:DELETE-FILE (DICT INV) <cr>	*Deletes dictionary section of file INV.*
:DELETE-FILE (PAYROLL) <cr>	*Deletes data and dictionary section of file PAYROLL.*

Figure C. Sample Usage of CLEAR-FILE and DELETE-FILE Verbs

7.5 Copying Items: Copy Processor

The COPY processor copies items from a file to the terminal, printer, to another file (either in your account or in some other user-account), or to items with different item-ids within the same file.

The COPY processor is evoked via the COPY verb. The general form is:

```
COPY {DICT} file-name item-list* or * {(options)}
```

The "file-name" parameter specifies the source file. The "item-list" consists of one or more item-ids separated by blanks, or an asterisk (*) specifying all items. The item-list specifies items to be copied. The "item-list*" need not be included if processing a selected or saved item-list (refer to the ENGLISH Programming Manual). The "options" parameter, if used, must be enclosed in parentheses (see the topic titled ADDITIONAL COPY PROCESSOR FEATURES in this chapter). Multiple options are separated by commas. Once a COPY command has been issued, the COPY processor responds with:

```
TO:
```

You must then enter the item-id(s) for the destination of the COPY operation (if the COPY is to the same file), or you must enter the destination file-name (enclosed in parentheses) optionally followed by the destination item-ids (if the copy is to another file). The first COPY operation in Figure B illustrates the creation of a new item WORTH in DICT SAMPLE by copying the existing item COST from DICT SAMPLE. The second operation illustrates the creation of one data item (1242-99) from another (1242-01) in the same file.

When using the COPY operation, multiple items may be specified as source and destination. Each item-id must be separated by a space and may be repeated within the item list. There may be different numbers of items within the source and destination lists. If the source file list is exhausted first, the COPY terminates. If the destination file list is exhausted first, the remainder of the items are copied with no change in item-id. The third example in Figure B copies data items RED, WHITE, and BLUE back into the same file (called FLAVORS) but gives them item-ids of ALPHA, BETA, and GAMMA.

When copying items with explicit source and destination item-lists, care should be taken to ensure that all of the source items exist. If an item being copied is 'NOT ON FILE', it causes the source-destination correspondence to be out of sequence.

If copying all existing items is desired, an asterisk (*) may be used as the item-list or the list may be supplied by a previous SELECT, SSELECT, GET-LIST, or FORM-LIST. Figure C illustrates a COPY of all dictionary items from file SAMPLE to the dictionary of file FLAVORS. Note that when a destination of other than the source file is desired, the destination file must be enclosed in parentheses. (If a destination file is not specified, the items will be copied into the original file). Note also that if the dictionary is the destination, the word DICT must be included within the parentheses (see Figure C).

When copying from one dictionary to another, the COPY processor does not copy dictionary items which have D/CODE=D (such as the DL/ID entry). DL/ID entries must only be created by the CREATE-FILE processor. To recreate both dictionary an ata sections of one file in a new file, a command sequence such as the example shown in Figure D must be used. Great care should be taken when copying to dictionaries to avoid overlaying D-pointers when using the 'O' option.

Note that to COPY to another user's account, a file synonym definition (D/CODE=Q) which points to that specific account and file must already exist.

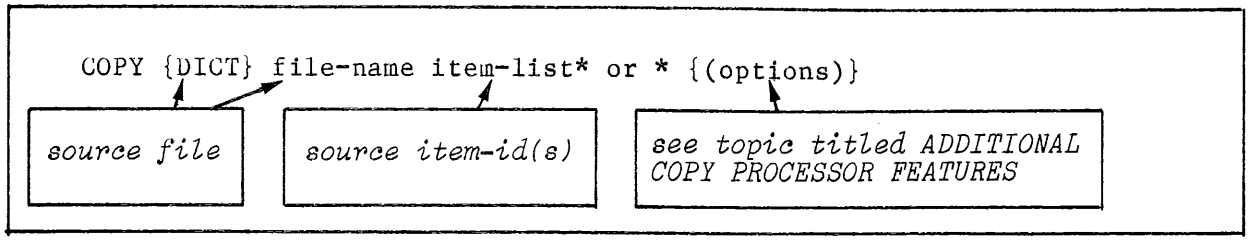


Figure A. General Form of COPY Verb

```
:COPY DICT SAMPLE COST <cr>          <-- Single dictionary item copied.
TO:WORTH <cr>
1 ITEMS COPIED

:COPY SAMPLE 1242:01 <cr>            <-- Single data item copied.
TO:1242-99 <cr>
1 ITEMS COPIED

:COPY FLAVORS RED WHITE BLUE <cr>    <-- Multiple data items copied.
TO:ALPHA BETA GAMMA <cr>
3 ITEMS COPIED
```

Figure B. Copying Items to Same File

```
:COPY DICT SAMPLE * <cr>             <-- All dictionary items copied.
TO:(DICT FLAVORS) <cr>
DL/ID NOT COPIED
2 ITEMS COPIED

:COPY INV 1007 <cr>                  <-- Copy one item to a different file
TO:(PARTS) A1007-1 <cr>                  with a new item-id.
1 ITEMS COPIED
```

Figure C. Copying Items to Different File

```
:CREATE-FILE (NEW-SAMPLE 1,1 3,1) <cr>   <-- New file created.
[417] FILE 'NEW-SAMPLE' CREATED;  BASE = 15417, MODULO = 1, SEPAR = 1.
[417] FILE 'DL/ID' CREATED:  BASE = 15418, MODULO = 3, SEPAR = 1.

:COPY DICT SAMPLE * <cr>             <-- All dictionary items (except
TO:(DICT NEW-SAMPLE) <cr>                DL/ID) copied.
DL/ID NOT COPIED
3 ITEMS COPIED

:COPY SAMPLE * <cr>                  <-- All data items copied.
TO:(NEW-SAMPLE) <cr>
2 ITEMS COPIED
```

Figure D. Creation of Entire Dictionary and Data Sections

7.6 Additional Copy Processor Features

The COPY processor may be used to copy items to the printer or to the terminal. Several options are additionally available for use with the COPY processor.

Items may be copied to the terminal by entering a null line (carriage return only) in response to the "TO:" message or by using the 'T' option (Figure B).

Several options are available for use with the COPY processor. Desired options are specified by one or more single characters separated by commas. The entire options list must be enclosed in parentheses and must follow the item-list in the COPY command (see the topic COPYING ITEMS: COPY PROCESSOR). Valid options are listed in Figure A.

Note that not all combinations of option characters are meaningful. 'D' and 'O' options only apply on a file-to-file copy. 'X' and 'L' options apply only on a copy to the terminal or printer.

The sample in Figure C copies two items from the FLAVORS file back into the FLAVORS file, changing the item-ids from ALPHA to BLUE and from BETA to YELLOW. Items ALPHA and BETA are deleted. If either item previously existed (BLUE or YELLOW), it is overwritten.

The example in Figure D shows the creation of a new file and copying the entire 'XYZ' file to 'ZZZ'. The modulo and separation are specified the same as in the CREATE-FILE.

OPTION	DESCRIPTION
D	*Deletes items from source file after they are copied to the destination file. (This option may not be used with the 'P' or 'T' option.)*
I	*Lists destination item-ids as they are copied.*
L	*Suppresses line-numbers on a copy to the terminal or printer, when used with the 'T' or 'P' option (respectively).*
N	*Suppress automatic paging of output when used with the 'T' option.*
O	*Overwrites destination file items with source file items if item-ids are the same. (This option may not be used with 'P' or 'T' option.)*
P	*Copies items to the printer (spooler).*
S	*Suppresses printing of "'item-id' NOT COPIED" messages.*
T	*Copies items to your terminal.*
X	*Specifies a hexadecimal dump to the terminal or printer, when used with the 'T' or 'P' option (respectively).*

Figure A. COPY Processor Options

```
:COPY M/DICT LISTACC  <cr>
TO: <cr>

(Item LISTACC will be displayed at the terminal.)
```

Figure B. Copying Items to Terminal

```
:COPY FLAVORS ALPHA BETA (D,O)  <cr>
TO:BLUE YELLOW  <cr>

2 ITEMS COPIED
```

Figure C. Sample Usage of COPY Options

```
:COPY XYZ *  <cr>
TO:(ZZZ 1,1 1,1)  <cr>

[417] FILE 'ZZZ' CREATED, BASE=3633, MODULO=1, SEPAR=1
[417] FILE 'DL/ID' CREATED, BASE=22391, MODULO=1, SEPAR=1

6 ITEMS COPIED

(NOTE:  Data items are copied, not dictionary items)
```

Figure D. Creating New File During Copy Process

8 SPOOLER

8.1 OVERVIEW

The SEQUEL computer system is a multiuser system which permits each user to perform processing operations with complete independence from other users. However, each system printer and magnetic tape unit can only support one user at a time. The SEQUEL spooler resolves such conflicts by providing simultaneous output to a maximum of four system printers, one tape drive, and as many ports as each system configuration will allow. The spooler enables each terminal to continue processing even if the desired peripheral is temporarily unavailable.

The spooler allows multiple users to share SEQUEL's system peripherals. Output reports are "spooled", which frees the terminal to initiate some other processing task. When the designated peripheral device and the "despooling" processor become available, the report is automatically "despooled" from the disc unit to either a magnetic tape, a system printer, or a terminal port.

Print Jobs

Individual reports to be spooled to an output device are called "print jobs". A print job may be created by a PROC, an ENGLISH statement, a DATA/BASIC program or other processor. Before a print job can be output, it must be assigned to a form queue. The spooler will accommodate up to 300 print jobs.

Form Queue

A form queue is a "list" of print jobs waiting to be despooled. Print jobs are assigned to a form queue via the SP-ASSIGN verb. Once created, the form queue may be assigned to an output device (a system printer, a tape unit, or a terminal port). If the device assignment is successful (i.e., no other form queue is assigned to that device), output may then begin.

Line-At-A-Time Mode

Using the SP-ASSIGN verb with an 'N' option, you may bypass the spooler. In this case, the process outputs text directly and immediately to the attached output device. This is called line-at-a-time output.

Figure A shows a listing of the major spooler features. Figure B illustrates a procedure that may be used to spool a print job to an output device.

The SEQUEL spooler offers the following features:

1. Simultaneous output to a maximum of four line printers, one magnetic tape unit, and as many ports as the system configuration will allow.
2. Ability to restart a print job when the printer malfunctions.
3. Creation of as many form queues as necessary.
4. Movement of one or more print jobs from one form queue to another.
5. Ability to change order of print jobs in queue.
6. Reassignment of devices to form queues by device type and number.
7. Specification of up to 100 print job copies.
8. Examination of closed jobs.
9. SP-STATUS display to show the current status of all devices and spooled reports, open and closed.
10. Reports when spooler has aborted via SP-STATUS.
11. System restart without loss of closed hold files.

Figure A. Major Features of SEQUEL Spooler

A procedure that may be used to spool a print job to an output device:

1. Create a form queue.
2. Assign the print job to a form queue.
3. Assign form queue to an output device.

Figure B. Sample Procedure for Spooling a Print Job

8.2 Overview (Continued)

Spooler Verbs

Figure A summarizes those verbs capable of communicating with the spooler. The "Screen-Oriented Verbs" (SP-STATUS and SP-JOBS) produce a screen display showing current spooler status. Immediately following each display is a "menu" of action codes you may enter to perform the desired spooling function. These action codes are listed below SP-STATUS and SP-JOBS in Figure A.

Each verb listed under "TCL Verbs" (except SP-ASSIGN, SP-PORTON, SP-PORTOFF, and SP-KILL) performs the same function as the action code displayed on the same line. However, the "TCL Verbs" do not cause a status display to appear.

If you do not remember the sequence of required parameters needed to execute a spooler verb, press the RETURN key immediately after entering the verb. The system will then display the required parameter(s) in the correct sequence.

Device Attachment

SEQUEL avoids conflicts by permitting only one process at a time to use each device. System software keeps track of device attachments. Spooler users need not be concerned with attachment/detachment since the spooler automatically attaches the appropriate unit before output begins, and automatically detaches the device when the queue is empty.

Queue Assignment

When the system is initialized, a form queue STANDARD is created and assigned to printer #1. Any print job created without an SP-ASSIGN statement will be put in the STANDARD form queue. You may modify this assignment at any time by executing the SP-ASSIGN verb. Several options are available (i.e., creation of hold files, instant output, nonspooling, and suppressed output). See the next topic, ASSIGNMENT OF PRINT JOBS AND OPTIONS: SP-ASSIGN VERB.

NOTE: Although the SEQUEL computer system allows you to change the name of the STANDARD form queue, the default queue will be referred to as STANDARD throughout this manual.

Hold Files

The creation of hold files permits print jobs to be spooled to the disc and held for an indefinite length of time for subsequent output. The spooler may accommodate up to 300 jobs including hold files. The actual number of entries your spooler can handle is dependent on the amount of overflow space available. Hold files are normally created for jobs requiring special forms or very lengthy jobs. Hold files may now be recovered if an 'X' or 'A' system restore or :RESTART-SPOOLER is performed.

Screen-Oriented Verbs	TCL Verbs	*Explanation*
SP-STATUS		*Displays status of spooler and all output devices.*
Create Form Queue	SP-CREATE	*Creates a form queue and optionally assigns it to a device.*
Change Form Queue	SP-FORM	*Changes the name of a form queue.*
Change Form Type	SP-TYPE	*Changes type of form used.*
Change Device Assignment	SP-DEVICE	*Assigns device to a form queue.*
Change Page Skip	SP-SKIP	*Changes number of pages ejected from printer after exhausting the form queue.*
List Print Jobs	SP-JOBS	*Displays SP-JOBS screen.*
Delete Form Queue	SP-FQDELETE	*Causes form queue to be released from table.*
Page Eject	SP-EJECT	*Causes attached printer to eject specified number of pages.*
99		*Exits screen display.*
SP-JOBS		*Displays status of print jobs.*
Move Form Queue	SP-MOVEQ	*Moves all print jobs of form queue to another form queue.*
Move Print Job	SP-SWITCH	*Moves a print job from current form queue to another.*
Change Options	SP-OPTS	*Changes options on a print job.*
Change #of Copies	SP-COPIES	*Specifies number of times a print job will be printed.*
Delete Job	SP-DELETE	*Deletes one or more print jobs.*
Stop Printing	SP-STOP	*Stops printer.*
Resume Printing	SP-RESUME	*Reactivates spooler.*
Edit Print Job	SP-EDIT	*Allows examination of print jobs.*
Change Priority	SP-PRIORITY	*Changes print job priority within a form queue.*
Suspend Printing	SP-SUSPEND	*Cancels printing but does not delete job.*
SP-STATUS	SP-STATUS	*Displays SP-STATUS screen.*
99		*Exits screen display.*
	SP-ASSIGN	*Assigns option codes, a print job code, and/or assigns subsequent print jobs to a queue.*
	SP-LOOK	*Displays parameters of current SP-ASSIGN statement.*
	SP-PORTON	*Starts terminal as a spooler port.*
	SP-PORTOFF	*Logs off an SP-PORTON line.*
	SP-KILL	*Cancels and deletes currently printing job.*

Figure A. Summary of Spooler Verbs

8.3 Assignment of Print Jobs and Options: SP-ASSIGN Verb

The SP-ASSIGN verb delegates print jobs to a form queue and may assign options to subsequent print jobs; SP-ASSIGN may also assign a print report number to a queue and/or specify the number of job copies to be output.

The general form of the SP-ASSIGN verb is:

```
SP-ASSIGN {{n} =form-queue-name {copies}} {options}
```

where:

n	specifies the print report number found in the PRINT ON statement of the corresponding DATA/BASIC program (refer to DATA/BASIC Programming Manual). If "n" is omitted, the assignment applies to all files except those with a specific SP-ASSIGN statement (i.e., this statement becomes the default and all previous assignments are cleared). Note: If 'n' is entered, you must also include "=form-queue-name".
=form-queue-name	specifies the name of the queue to receive print job(s). This name must be less than or equal to 20 alphabetic, numeric or special characters (asterisks may not be used).
options	may be one or more of the following: creation of hold files, instant output, no spooling and suppress output.
copies	specifies a number from 1 to 100 which represents the number of copies to be printed. Should "copies" not be specified, the default value is "1".

If no SP-ASSIGN statement is used, print jobs are put in the STANDARD form queue. Should an SP-ASSIGN verb be executed with no parameters or options, the job assignment is reset to the default queue STANDARD. This same result occurs when the operator logs off. If options and/or copies are specified in a SP-ASSIGN statement without using "=form-queue-name", the STANDARD form queue is assumed.

Options may be input in any sequence. The option(s) remain in effect until the next use of the SP-ASSIGN verb. There are four SP-ASSIGN options as shown in Figure A. Descriptions of these options follow.

H - Create Hold File on Disc

The 'H' option retains a print job on disc after printing is completed. Jobs which require special forms or special manipulations can be handled with this option. When a print job is assigned to a hold file, the message "ENTRY #n" appears on the screen. The 'n' specifies the print job number of the hold file.

The 'H' option cannot be used with the 'N' option (which is explained on the following page).

I - Instant Output

The spooler normally waits for an output report to be completed (closed) before it begins the output. By using the 'I' option, the spooler begins output as soon as the first frame of output is available. The frames of the job are then released back to the overflow table as soon as they are printed. Should a malfunction occur while outputting a print job with an 'I' option, you may suspend printing; however, you cannot recover any data already printed. SP-RESUME causes output to begin where it left off. The 'I' option is incompatible with multiple copies and the 'N' option.

N - No Spooling (Output Directly to Device)

The 'N' option specifies no spooling. Data is output a line at a time to the assigned device. If the device is not available, you are notified and the 'N' option is removed from the assignment. This option should be used only when the intent is to prohibit other users from accessing the output device. The device(s) will become attached to your line and will require you to detach the device(s) when complete so that others (including the spooler itself) may gain control. The 'N' option is incompatible with the 'I' or 'H' options, multiple copies and with output to a port.

S - Suppress Output

The 'S' option suppresses spooler output. Normally, the 'S' option is used with the 'H' option. However, if the 'S' option appears alone, an output job is built but not printed because the job is not queued for output.

Figure A displays a list of incompatible spooler option and device combinations.

Spooler Options	Incompatible Option and Device Combinations
H	N
I	N, Copies
N	I, H, Copies, PORT
Copies	I, N

NOTE: If any of the above combinations are used, the following error message appears: INCOMPATIBLE OPTIONS: NI, NH, N COPIES, I COPIES, OR N PORT.

Figure A. Display of Incompatible Spooler Option and Device Combinations

8.4 General Format and Examples of SP-ASSIGN Verb

The general format of the SP-ASSIGN verb is shown in Figure A. Note that each parameter is separated by one space. Figure B illustrates the use of the SP-ASSIGN verb.

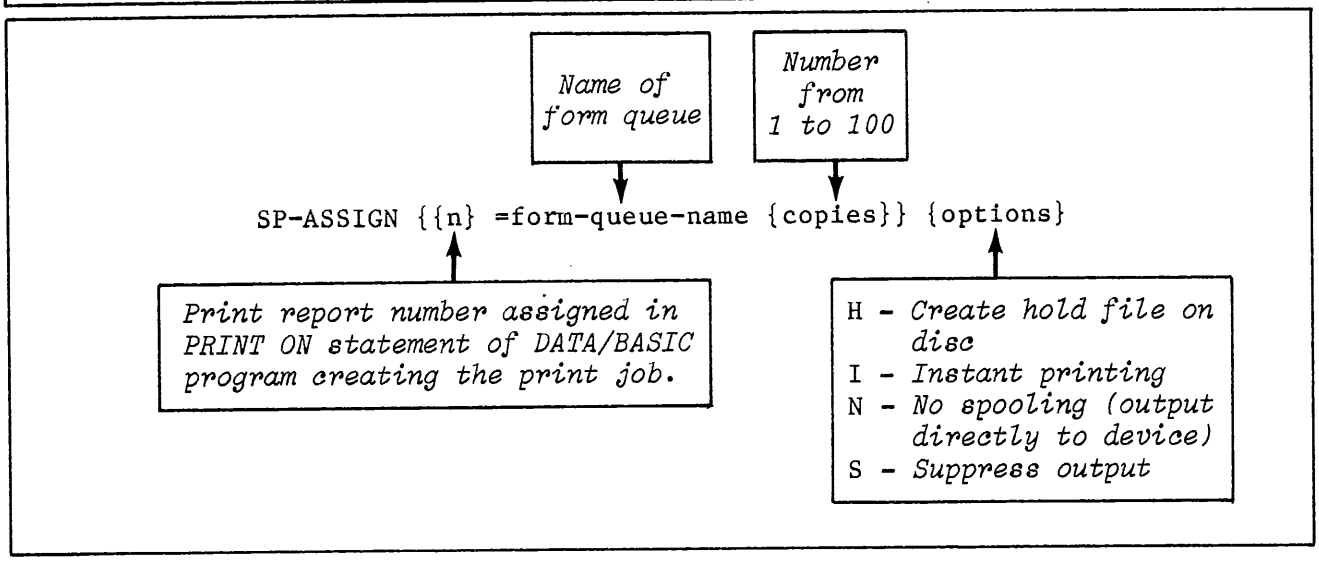


Figure A. General Form of SP-ASSIGN Verb

EXAMPLE	EXPLANATION
:SP-ASSIGN	*Spooler device assignments are cleared and all future print jobs are assigned to form queue STANDARD.*
:SP-ASSIGN =INVOICE :LIST ENG '750L' (P)	*A print job containing item "750L" is assigned to the form queue INVOICE.*
:SP-ASSIGN 1 =CHECK I	*All future print jobs with a print report number of "1" will be assigned to form queue CHECK with the option of instant printing.*
:SP-ASSIGN =STANDARD HS :RUNOFF INV RECEIPTS (P) ENTRY #6	*All future print jobs will be assigned to the STANDARD form queue as hold files with suppressed printing. Print job #6, which contains the item "RECEIPTS", is now a hold file.*
:SP-ASSIGN =CHECK N :LIST PROD 'PART#' (P)	*A print job containing the item "PART#" is assigned to form queue CHECK and is output a line at a time to the assigned device.*
:SP-ASSIGN H	*All future print jobs assigned to the default form queue will be hold files.*
:SP-ASSIGN =INVOICE :SP-ASSIGN 1 =CHECK HS	*All jobs except those with print report #1 will be assigned to INVOICE.*

Figure C. Sample Use of SP-ASSIGN Verb

8.5 Displaying SP-ASSIGN Parameters: SP-LOOK Verb

The SP-LOOK verb displays the parameters used in your current SP-ASSIGN statement.

The format of the SP-LOOK verb is

SP-LOOK {line #}

To display the parameters of your current SP-ASSIGN statement, enter:

```
:SP-LOOK <cr>
```

If you wish to display the SP-ASSIGN parameters for another terminal, enter:

:SP-LOOK line#

where 'line#' represents the port number you wish to access. If the port number you enter is greater than or equal to the spooler's line number, the message "LINE # TOO LARGE" is displayed and you are then returned to TCL.

After executing the SP-LOOK verb, you may also see the following message:

DEFAULT ASSIGNMENT IS TO QUEUE "STANDARD".

This message appears:

1. After you log on and before either an SP-ASSIGN statement is entered or before a print job is run.
2. After you enter an SP-ASSIGN statement with no parameters and before a print job is run.

The system maintains an assignment table that logs in your current SP-ASSIGN statement. The above message appears because you do not currently have an entry in the table. If you execute an SP-ASSIGN statement without parameters or options, any previous table entry assigned to your process is removed. Should you execute an output operation without an entry in the assignment table, one will be created and assigned to the STANDARD form queue.

Figure A illustrates the use of SP-LOOK when viewing the SP-ASSIGN parameters for your own terminal. Figure B shows you how to display the SP-ASSIGN parameters for another port.

```
SP-LOOK {line#}
```

Figure A. General Format of SP-LOOK Verb

```
:SP-LOOK <cr>

FORM QUEUE       =   STANDARD
DEVICE TYPE      =   LINE PRINTER
DEVICE NUMBER    =   1
SPOOLER OPTIONS  =   HS
COPIES           =   1
```

Figure B. Displaying SP-ASSIGN Parameters For Your Own Terminal

```
:SP-LOOK 18 <cr>

FORM QUEUE       =   TAPEOUT
DEVICE TYPE      =   TAPE
DEVICE NUMBER    =   1
SPOOLER OPTIONS  =
COPIES           =   1
```

Figure C. Displaying SP-ASSIGN Parameters for Port 18

8.6 Displaying Status of Spooler: SP-STATUS Verb

The SP-STATUS verb displays current spooler status.

The general form of the SP-STATUS verb is:

SP-STATUS

The SP-STATUS verb displays: 1) the name of each form queue, 2) the types of forms used, 3) the device number assigned to each form queue, 4) the type of output device, 5) the status of each device, 6) the number of print jobs waiting to be printed, and 7) the number of page skips specified.

The device status may be one of the following:

Status	Explanation
NEED DEV	The desired device is attached to another line.
CABLE OFF	The cable to the line printer is not plugged in.
NOT READY	Device is off-line or, if device is a tape unit, the mounted tape has no write ring.
NEXT REEL	Tape drive needs a new reel.
PARITY	A parity error has occurred on tape.
SUSPEND	A suspend order has been issued.
STOP	A stop order has been issued.
ACTIVE	Device is outputting a job.
ASSIGNED	The queue is assigned to a device but there are no jobs ready for output.
WAITING	Queue is waiting for a device that is currently assigned to another queue.
NO DEVICE	Queue is not assigned to a device.

At any given time, more than one of the status conditions listed above may exist (e.g., ACTIVE and ASSIGNED or SUSPEND and NOT READY); however, only one status condition is displayed. The display priority is: CABLE OFF, NEXT REEL, PARITY, SUSPEND, NEED DEV, NOT READY, STOP, ACTIVE, ASSIGNED, WAITING.

Immediately following the status display is a list of action codes. You may enter the desired action code number in response to the prompt "ENTER ACTION CODE:".

If the RETURN key is pressed in response to the "ENTER ACTION CODE:" prompt, the SP-STATUS screen is displayed again. To exit from the SP-STATUS display, enter Action Code 99.

Each action code and equivalent spooler verb is described in the following sections.

See Figure A for a sample screen layout which would follow the execution of the SP-STATUS verb.

```
:SP-STATUS <cr>


                              FORM QUEUE ASSIGNMENT

  QUEUE        FORM        DEVICE      DEVICE      DEVICE      NUMBER      PAGE
  NAME         TYPE        ASSIGNED    TYPE        STATUS      ENTRIES     SKIP

  STANDARD     1-PART         1        LPTR        ACTIVE         7         0
  CHECK        PAYROLL        1        TAPE        NOT READY      1         0
  LIST-T                      0        0           NO DEVICE      0         0
  LABEL        MAIL           2        PORT        ASSIGNED       0         0
  SAMPLE                      3        LPTR        ASSIGNED       0         0
  PAYABLES                    0        LPTR        NO DEVICE      1         0
    .            .            .          .           .            .         .
    .            .            .          .           .            .         .

                                   ACTION CODES
  1. CREATE FORM QUEUE     4. CHANGE DEVICE ASSIGNMENT     7. DELETE FORM QUEUE
  2. CHANGE FORM QUEUE     5. CHANGE PAGE SKIP             8. PAGE EJECT
  3. CHANGE FORM TYPE      6. LIST PRINT JOBS             99. EXIT

ENTER ACTION CODE:
```

Figure A. Sample Screen Display of SP-STATUS Verb

8.7 Spooler Output Devices

The spooler can output print jobs to three types of devices: a line printer, a tape unit, and a terminal port.

Line Printer

The spooler may support a maximum of four line printers. Each printer is assigned a device number; this may be a number from 1 to 4. When the system is initialized, the default form queue STANDARD is assigned to printer #1.

Tape Unit

When a job is spooled to tape, the output is written to the magnetic tape unit starting wherever the tape is currently positioned. The tape unit is rewound by the spooler only when the end of a reel is reached; thus, jobs may be spooled to tape one after another. You must mount the tape (with write ring), place the unit on-line, and rewind or position the tape prior to using the spooler.

Caution should be used when spooling to tape while some other user is using the tape drive (a status of "NEED DEV"). The spooler will wait for the tape to become available before attempting to write. If the other user rewinds the tape and detaches it, the spooler will gain control of the tape and start writing the spooled output.

Normally the spooler writes unlabeled tape jobs consisting of 500-byte records. However, if the 'N' option is in effect, the records will be of varying length and each record will terminate at the end of a print line closest to 500 bytes. In other words, the records will be split at line boundries if the 'N' option is in effect, rather than at the 500th character. This is useful when taking spooled reports to other devices that must have full lines within records.

Terminal Port

The spooler may spool output to any port if assigned to a queue. Any number of these ports may be attached to CRT terminals or serial printers. Note: You may not assign Port Zero as a spooler port.

Figure A listed the terms used to represent the three types of spooler output devices and a brief description of each.

Device Type	Brief Explanation	Legal Device# Range
LPTR	Line Printer; spooler supports a maximum of four.	1 - 4
TAPE	Magnetic Tape Unit; spooler supports a maximum of one.	1
PORT	Terminal Port; spooler supports as many ports as the system configuration allows.	1 - 63

Figure A. Terms Used to Define Spooler Output Devices

8.8 Creating a Form Queue

A form queue is created by the execution of either the SP-STATUS Action Code 1 (Create Form Queue) or the SP-CREATE verb.

The general form of the Create Form Queue Action Code is:

```
ENTER ACTION CODE:1 <cr>
FORM-QUEUE  DEVICE-TYPE(LPTR,TAPE,PORT)  DEVICE#:
```

where:

FORM-QUEUE — is prompting for a form queue name. This name must be less than or equal to 20 alphabetic, numeric, or special characters (asterisks and embedded spaces are not allowed). Note: a form queue name may not be a single numeric character.

DEVICE-TYPE — is prompting for one of the following:

Type	Explanation
LPTR	Line Printer
TAPE	Tape Drive
PORT	Terminal Port
0	No Device

DEVICE# — is prompting for the number assigned to the output device. Each system printer is assigned a device number from 1 to 4. The tape unit is device 1, 2, 3, or 4, depending on the number of drives on the system. A port device may have a device number up to the maximum number of ports on the system.

Each prompt response must be separated by one space. Figure A illustrates the use of the SP-STATUS Action Code 1.

```
:SP-STATUS <cr>

                              FORM QUEUE ASSIGNMENT

  QUEUE        FORM        DEVICE      DEVICE      DEVICE      NUMBER      PAGE
  NAME         TYPE        ASSIGNED    TYPE        STATUS      ENTRIES     SKIP

  STANDARD     1-PART         1        LPTR        ACTIVE         7         0
  CHECK        PAYROLL        1        TAPE        NOT-READY      1         0
  LIST-T                      0        0           NO DEVICE      0         0
  LABEL        MAIL           2        PORT        ASSIGNED       0         0
  SAMPLE                      3        LPTR        ASSIGNED       0         0
  PAYABLES                    0        LPTR        NO DEVICE      1         0

                                  ACTION CODES
  1. CREATE FORM QUEUE    4. CHANGE DEVICE ASSIGNMENT    7. DELETE FORM QUEUE
  2. CHANGE FORM QUEUE    5. CHANGE PAGE SKIP            8. PAGE EJECT
  3. CHANGE FORM TYPE     6. LIST PRINT JOBS            99. EXIT

ENTER ACTION CODE:1 <cr>
FORM-QUEUE  DEVICE-TYPE(LPTR,TAPE,PORT)
   DEVICE#:PART LPTR 2 <cr>
```

The Create Form Queue Action Code produces the form queue PART which is assigned to printer #2.

Figure A. Sample Usage of the Create Form Queue Action Code

The general form of the SP-CREATE verb is:

```
SP-CREATE form-queue device-type device#
```

The device number and device type may both be "0" (zero). In this case, the form queue is created but no print jobs are output. A device is assigned to the queue by either the SP-STATUS Action Code 4 (Change Device Assignment) or the SP-DEVICE verb (refer to the section CHANGING DEVICE ASSIGNMENT). If you create a queue and assign it to a device that is already assigned to a queue, the message:

```
A QUEUE IS ALREADY ASSIGNED TO THIS DEVICE
OK TO CHANGE?
```

is displayed. If you enter "N", your queue will be created without a device assignment. If you enter "Y", the print job currently being output on the selected device will be completed, and your queue will have a status of "WAITING". When the job is complete, the device is reassigned to the queue you created, and the previously assigned queue has a status of "NO DEVICE".

Figure B illustrates the use of the SP-CREATE verb. Figure C displays a sample SP-STATUS screen which would appear following the execution of the example in Figure A or B.

```
:SP-CREATE PART LPTR 2 <cr>
:SP-STATUS <cr>
```

The SP-CREATE verb performs the same function as Action code 1.

Figure B. Sample Usage of SP-CREATE Verb

FORM QUEUE ASSIGNMENT

QUEUE NAME	FORM TYPE	DEVICE ASSIGNED	DEVICE TYPE	DEVICE STATUS	NUMBER ENTRIES	PAGE SKIP
STANDARD	1-PART	1	LPTR	ACTIVE	7	0
CHECK	PAYROLL	1	TAPE	NOT READY	1	0
LIST-T		0	0	NO DEVICE	0	0
LABEL	MAIL	2	PORT	ASSIGNED	0	0
SAMPLE		3	LPTR	ASSIGNED	0	0
PAYABLES		0	LPTR	NO DEVICE	1	0
PART		2	LPTR	ASSIGNED	0	0

Figure C. Sample SP-STATUS Screen Display

8.9 Changing the Name of a Form Queue

The name of a form queue may be changed by the execution of either the SP-STATUS Action Code 2 (Change Form Queue) or the SP-FORM verb.

The format of the Change Form Queue Action Code is:

```
ENTER ACTION CODE:2 <cr>
OLD-NAME  NEW-NAME:
```

where "OLD-NAME" and "NEW-NAME" are prompts specifying the order of the required response.

The format of the SP-FORM verb is:

```
SP-FORM old-name  new-name
```

Action Code 2 and the SP-FORM verb both change the name of an already established form queue. The new queue name remains in effect until another name is entered by one of these two methods.

You may change the name of STANDARD; however, the queue itself remains as the default form queue.

Figure A and B illustrate the use of the Change Print Action Code and the SP-FORM verb. Figure C displays the SP-STATUS screen which would follow the execution of the example in Figure A or B.

```
:SP-STATUS <cr>

                              FORM QUEUE ASSIGNMENT

  QUEUE        FORM        DEVICE      DEVICE      DEVICE      NUMBER      PAGE
  NAME         TYPE        ASSIGNED    TYPE        STATUS      ENTRIES     SKIP

  STANDARD     1-PART         1        LPTR        ACTIVE          7        0
  CHECK        PAYROLL        1        TAPE        NOT READY       1        0
  LIST-T                      0        0           NO DEVICE       0        0
  LABEL        MAIL           2        PORT        ASSIGNED        0        0
  SAMPLE                      3        LPTR        ASSIGNED        0        0
  PAYABLES                    0        LPTR        NO DEVICE       1        0
  PART                        2        LPTR        ASSIGNED        0        0

                                ACTION CODES
  1. CREATE FORM QUEUE     4. CHANGE DEVICE ASSIGNMENT     7. DELETE FORM QUEUE
  2. CHANGE FORM QUEUE     5. CHANGE PAGE SKIP             8. PAGE EJECT
  3. CHANGE FORM TYPE      6. LIST PRINT JOBS             99. EXIT

ENTER ACTION CODE:2 <cr>
OLD-NAME  NEW-NAME:STANDARD INVOICES <cr>
```

The Change Form Queue Action Code alters the name of form queue STANDARD to INVOICES.

Figure A. Sample Usage of Change Form Queue Action Code

```
:SP-FORM STANDARD INVOICES <cr>

:SP-STATUS <cr>
```

The SP-FORM verb is performing the same function as Action Code 2 above. Notice that you must enter "SP-STATUS" to get screen display.

Figure B. Sample Usage of SP-FORM Verb

```
                              FORM QUEUE ASSIGNMENT

  QUEUE        FORM        DEVICE      DEVICE      DEVICE      NUMBER      PAGE
  NAME         TYPE        ASSIGNED    TYPE        STATUS      ENTRIES     SKIP

  INVOICES     1-PART         1        LPTR        ACTIVE          7        0
  CHECK        PAYROLL        1        TAPE        NOT READY       1        0
  LIST-T                      0        0           NO DEVICE       0        0
  LABEL        MAIL           2        PORT        ASSIGNED        0        0
  SAMPLE                      3        LPTR        ASSIGNED        0        0
  PAYABLES                    0        LPTR        NO DEVICE       1        0
  PART                        2        LPTR        ASSIGNED        0        0
```

Figure C. SP-STATUS Screen Display

8.10 Changing Type of Form Used on Printer

The SP-STATUS Action Code 3 (Change Form Type) and the SP-TYPE verb allow you to specify a form type for each queue. This form type will be displayed on the SP-STATUS screen and serves to remind you of the type of form required on the assigned device.

The format of the Change Form Type Action Code is:

```
ENTER ACTION CODE:3 <cr>
FORM-QUEUE  FORM-TYPE:
```

where:

FORM-QUEUE — is prompting for the name of the form queue assigned to the printer.

FORM-TYPE — is prompting for the name of the form to be placed on the printer.

The format of the SP-TYPE verb is:

```
SP-TYPE  form-queue  form-type
```

Note that the terms "FORM-QUEUE" and "FORM-TYPE" refer to two different elements of a form queue. The form queue is associated with devices, not necessarily with the type of forms. There may be a large number of uniquely defined form queues. It is possible to change the form type without changing the name of the form queue because the "form-type" is not used by the spooler. It is intended only as a reminder to you of the type of form that should be on the specified device.

Figure A and B illustrate the use of the Change Form Type Action Code and the SP-TYPE verb. Figure C displays the SP-STATUS screen which would follow the execution of the example in Figure A or B.

```
:SP-STATUS <cr>

                              FORM QUEUE ASSIGNMENT

   QUEUE        FORM        DEVICE      DEVICE      DEVICE      NUMBER      PAGE
   NAME         TYPE        ASSIGNED    TYPE        STATUS      ENTRIES     SKIP

   INVOICES     1-PART         1        LPTR        ACTIVE         7         0
   CHECK        PAYROLL        1        TAPE        NOT READY      1         0
   LIST-T                      0        0           NO DEVICE      0         0
   LABEL        MAIL           2        PORT        ASSIGNED       0         0
   SAMPLE                      3        LPTR        ASSIGNED       0         0
   PAYABLES                    0        LPTR        NO DEVICE      1         0
   PART                        2        LPTR        ASSIGNED       0         0

                                 ACTION CODES
  1. CREATE FORM QUEUE    4. CHANGE DEVICE ASSIGNMENT    7. DELETE FORM QUEUE
  2. CHANGE FORM QUEUE    5. CHANGE PAGE SKIP            8. PAGE EJECT
  3. CHANGE FORM TYPE     6. LIST PRINT JOBS            99. EXIT

ENTER ACTION CODE:3 <cr>
FORM-QUEUE  FORM-TYPE:INVOICES INVOICE <cr>
```

The Change Form Type Action Code changes the form type of form queue INVOICES to INVOICE.

Figure A. Sample Usage of Change Form Type Action Code

```
:SP-TYPE INVOICES INVOICE <cr>

:SP-STATUS <cr>
```

This SP-TYPE verb performs the same function as the action code in Figure A.

Figure B. Sample Usage of SP-TYPE Verb

```
                              FORM QUEUE ASSIGNMENT

   QUEUE        FORM        DEVICE      DEVICE      DEVICE      NUMBER      PAGE
   NAME         TYPE        ASSIGNED    TYPE        STATUS      ENTRIES     SKIP

   INVOICES     INVOICE        1        LPTR        ACTIVE         7         0
   CHECK        PAYROLL        1        TAPE        NOT READY      1         0
   LIST-T                      0        0           NO DEVICE      0         0
   LABEL        MAIL           2        PORT        ASSIGNED       0         0
   SAMPLE                      3        LPTR        ASSIGNED       0         0
   PAYABLES                    0        LPTR        NO DEVICE      1         0
   PART                        2        LPTR        ASSIGNED       0         0
```

Figure C. SP-STATUS Display Screen

8.11 Changing Device Assignments

A device assignment may be changed by either the SP-STATUS Action Code 4 (Change Device Assignment) or the SP-DEVICE verb.

The general form of the Change Device Assignment Action Code is:

```
ENTER ACTION CODE:4 <cr>
FORM-QUEUE  DEVICE-TYPE(LPTR,TAPE,PORT)  DEVICE#:
```

The format of the SP-DEVICE verb is:

```
SP-DEVICE form-queue  device-type  device#
```

Changing a device assignment is especially useful when a printer is idle but there are several form queues waiting to be printed. In this case, you may assign the idle printer to a form queue waiting to be printed.

In general, there may be only one queue assigned to each device at any time. For example, should you attempt to assign Form Queue B to LPTR 2 and this device is already assigned to Form Queue A, you are asked if such a device reassignment is permissible. If permission is granted to change the device assignment, a stop order is issued to Form Queue A and its assignment to LPTR 2 will be cancelled as soon as the current job has been printed. Form Queue B will acquire a "WAITING" status until Form Queue A is finished printing; Form Queue B will then be assigned to LPTR 2.

The range of legal device numbers is given in the topic titled "SPOOLER OUTPUT DEVICES". If the device number specified is not within range, the following message will appear:

```
INVALID DEVICE# RANGE.
```

NOTE: The spooler has no way of knowing which devices are physically present. It is your responsibility to ensure that the device number specified is actually present on the system.

Figure A and B illustrate the use of the Change Device Assignment Action Code and the SP-DEVICE verb. Figure C displays the SP-STATUS screen which would follow the execution of the example in Figure A or B.

```
:SP-STATUS <cr>

                              FORM QUEUE ASSIGNMENT

   QUEUE        FORM        DEVICE      DEVICE      DEVICE      NUMBER      PAGE
   NAME         TYPE        ASSIGNED    TYPE        STATUS      ENTRIES     SKIP

   INVOICES     INVOICE        1        LPTR        ACTIVE         7         0
   CHECK        PAYROLL        1        TAPE        NOT READY      1         0
   LIST-T                      0        0           NO DEVICE      0         0
   LABEL        MAIL           2        PORT        ASSIGNED       0         0
   SAMPLE                      3        LPTR        ASSIGNED       0         0
   PAYABLES                    0        LPTR        NO DEVICE      1         0
   PART                        2        LPTR        ASSIGNED       0         0

                                ACTION CODES
  1. CREATE FORM QUEUE    4. CHANGE DEVICE ASSIGNMENT    7. DELETE FORM QUEUE
  2. CHANGE FORM QUEUE    5. CHANGE PAGE SKIP            8. PAGE EJECT
  3. CHANGE FORM TYPE     6. LIST PRINT JOBS            99. EXIT

ENTER ACTION CODE:4 <cr>
FORM-QUEUE  DEVICE-TYPE(LPTR,TAPE,PORT)
   DEVICE#:PAYABLES LPTR 1 <cr>

A QUEUE IS ALREADY ASSIGNED TO THIS DEVICE
OK TO CHANGE?Y <cr>
```

The Change Device Assignment Action Code assigns form queue PAYABLES to printer #1.
A response of 'Y' permits the device reassignment.

Figure A. Sample Usage of Change Device Assignment Action Code

```
:SP-DEVICE PAYABLES LPTR 1 <cr>

:SP-STATUS <cr>
```

The SP-DEVICE verb performs the same function as Action Code 7 in Figure A.

Figure B. Sample Usage of SP-DEVICE Verb

```
                              FORM QUEUE ASSIGNMENT

   QUEUE        FORM        DEVICE      DEVICE      DEVICE      NUMBER      PAGE
   NAME         TYPE        ASSIGNED    TYPE        STATUS      ENTRIES     SKIP

   INVOICES     INVOICE        0        0           NO DEVICE      6         0
   CHECK        PAYROLL        1        TAPE        NOT READY      1         0
   LIST-T                      0        0           NO DEVICE      0         0
   LABEL        MAIL           2        PORT        ASSIGNED       0         0
   SAMPLE                      3        LPTR        ASSIGNED       0         0
   PAYABLES                    1        LPTR        ACTIVE         1         0
   PART                        2        LPTR        ASSIGNED       0         0
```

Figure C. SP-STATUS Screen Display

8.12 Deleting a Form Queue

A form queue may be deleted by use of either the SP-STATUS Action Code 7 (Delete Form Queue) or the SP-FQDELETE verb.

The format of the Delete Form Queue Action Code is:

```
ENTER ACTION CODE:7 <cr>
FORM-QUEUE:
```

where "FORM-QUEUE" is prompting for the name of the queue to be deleted.

The format of the SP-FQDELETE verb is:

```
SP-FQDELETE form-queue
```

A form queue may only be deleted if it has no jobs queued for output and there are no lines assigned to it. The default form queue (STANDARD) may not be deleted.

Figure A and B illustrate the usage of the Delete Form Queue Action Code and the SP-FQDELETE verb. Figure C displays the SP-STATUS screen which would follow the execution of the example in Figure A or B.

```
:SP-STATUS <cr>

                              FORM QUEUE ASSIGNMENT

  QUEUE        FORM         DEVICE       DEVICE       DEVICE       NUMBER      PAGE
  NAME         TYPE         ASSIGNED     TYPE         STATUS       ENTRIES     SKIP

  INVOICES     INVOICE         1         LPTR         ACTIVE          6         0
  CHECK        PAYROLL         1         TAPE         NOT READY       1         0
  LIST-T                       0         0            NO DEVICE       0         0
  LABEL        MAIL            2         PORT         ASSIGNED        0         0
  SAMPLE                       3         LPTR         ASSIGNED        0         0
  PAYABLES                     0         LPTR         NO DEVICE       0         0
  PART                         2         LPTR         ASSIGNED        0         0

                                  ACTION CODES
  1. CREATE FORM QUEUE    4. CHANGE DEVICE ASSIGNMENT    7. DELETE FORM QUEUE
  2. CHANGE FORM QUEUE    5. CHANGE PAGE SKIP            8. PAGE EJECT
  3. CHANGE FORM TYPE     6. LIST PRINT JOBS            99. EXIT

ENTER ACTION CODE:7 <cr>
FORM-QUEUE-NAME:LIST-T <cr>
```

The Delete Form Queue Action Code removes the LIST-T queue from the queue table.

Figure A. Sample Usage of Delete Form Queue Action Code

```
:SP-FQDELETE LIST-T <cr>
:SP-STATUS <cr>
```

The SP-FQDELETE verb performs the same function as the action code in Figure A.

Figure B. Sample Usage of SP-FQDELETE Verb

```
                              FORM QUEUE ASSIGNMENT

  QUEUE        FORM         DEVICE       DEVICE       DEVICE       NUMBER      PAGE
  NAME         TYPE         ASSIGNED     TYPE         STATUS       ENTRIES     SKIP

  INVOICES     INVOICE         1         LPTR         ACTIVE          6         0
  CHECK        PAYROLL         1         TAPE         NOT READY       1         0
  LABEL        MAIL            2         PORT         ASSIGNED        0         0
  SAMPLE                       3         LPTR         ASSIGNED        0         0
  PAYABLES                     0         LPTR         NO DEVICE       0         0
  PART                         2         LPTR         ASSIGNED        0         0
```

Figure C. Sample SP-STATUS Screen Display

8.13 Ejecting Pages from Printer

To alter the number of pages ejected from the printer after exhausting each form queue, use the SP-STATUS Action Code 5 (Change Page Skip) or the SP-SKIP verb. To tell an attached printer to eject a specified number of pages at your command, use SP-STATUS Action Code 8 (Page Eject) or the SP-EJECT verb.

The format of the Change Page Skip Action Code is:

```
ENTER ACTION CODE:5 <cr>
FORM-QUEUE  PAGES:
```

where:

FORM-QUEUE — is prompting for the name of the queue assigned to the printer.

PAGES — is prompting for the number of pages to be ejected.

The format of the SP-SKIP verb is:

```
SP-SKIP form-queue pages
```

The maximum number of pages the printer may eject after exhausting a print queue is 10. Answering the "PAGES" prompt with a number greater than 10 results in the following message: TOO MANY PAGES.

The format of the Page Eject Action Code is:

```
ENTER ACTION CODE:8 <cr>
FORM-QUEUE  PAGES:
```

The format of the SP-EJECT verb is:

```
SP-EJECT  form-queue  pages
```

Action Code 8 and the SP-EJECT verb attach the assigned printer to your line and issue a number of page ejects to that printer. (Note that page ejects are not shown on the SP-STATUS display.) The printer remains attached to the terminal until you issue a P-DET or SP-ASSIGN verb or you log off. See topic entitled PRINTER CONTROL: P-ATT and P-DET for a detailed explanation of the P-DET command.

If you attempt to eject more than 10 pages, the following message appears: TOO MANY PAGES.

Figure A and B illustrate the usage of Change Page Skip Action Code and the SP-SKIP verb. Figure C displays the SP-STATUS screen that would follow the execution of the example in Figure A or B. Sample usage of the Page Eject Action Code and the SP-EJECT verb is seen in Figure D and E.

```
:SP-STATUS <cr>

                        FORM QUEUE ASSIGNMENT

  QUEUE       FORM       DEVICE      DEVICE      DEVICE      NUMBER      PAGE
  NAME        TYPE       ASSIGNED    TYPE        STATUS      ENTRIES     SKIP

  INVOICES    INVOICE       1        LPTR        ACTIVE         6         0
  CHECK       PAYROLL       1        TAPE        NOT READY      1         0
  LABEL       MAIL          2        PORT        ASSIGNED       0         0
  SAMPLE                    3        LPTR        ASSIGNED       0         0
  PAYABLES                  0        LPTR        NO DEVICE      0         0
  PART                      2        LPTR        ASSIGNED       0         0

                            ACTION CODES
  1. CREATE FORM QUEUE    4. CHANGE DEVICE ASSIGNMENT    7. DELETE FORM QUEUE
  2. CHANGE FORM QUEUE    5. CHANGE PAGE SKIP            8. PAGE EJECT
  3. CHANGE FORM TYPE     6. LIST PRINT JOBS            99. EXIT

ENTER ACTION CODE:5 <cr>
FORM-QUEUE PAGES:INVOICES 2 <cr>
```

The Change Page Skip Action Code alters the Page Skip function from 0 to 2.

Figure A. Sample Usage of Change Page Skip Action Code

```
:SP-SKIP INVOICES 2 <cr>
:SP-STATUS <cr>
```

The SP-SKIP verb performs the same function as Action Code 5 in Figure A.

Figure B. Sample Usage of SP-SKIP Verb

```
                        FORM QUEUE ASSIGNMENT

  QUEUE       FORM       DEVICE      DEVICE      DEVICE      NUMBER      PAGE
  NAME        TYPE       ASSIGNED    TYPE        STATUS      ENTRIES     SKIP

  INVOICES    INVOICE       1        LPTR        ACTIVE         6         2
  CHECK       PAYROLL       1        TAPE        NOT READY      1         0
  LABEL       MAIL          2        PORT        ASSIGNED       0         0
  SAMPLE                    3        LPTR        ASSIGNED       0         0
  PAYABLES                  0        LPTR        NO DEVICE      0         0
  PART                      2        LPTR        ASSIGNED       0         0
```

Figure C. SP-STATUS Screen Display

```
ENTER ACTION CODE:8 <cr>
FORM-QUEUE PAGES:SAMPLE 4 <cr>
```

The Page Eject Action Code causes the printer assigned to SAMPLE to eject 4 pages.

Figure D. Sample Usage of Change Page Eject Action Code

```
:SP-EJECT SAMPLE 4 <cr>
```

The SP-EJECT verb performs the same function as Action Code 8 in Figure D.

Figure E. Sample Usage of SP-EJECT Verb

8.14 Displaying Print Job Status: SP-JOBS

The SP-STATUS Action Code 6 (List Print Jobs) and the SP-JOBS verb display the current status of print jobs within each form queue.

The format of the List Print Jobs Action Code is:

```
ENTER ACTION CODE / PAGE NUMBER (P#)? 6 <cr>
```

This action code evokes the same screen display as the SP-JOBS verb which is described below.

The format of the SP-JOBS verb is:

```
SP-JOBS
```

The SP-JOBS verb displays: 1) name of the form queue, 2) the print job number assigned to spooled jobs, 3) name of account from which print job was created, 4) the terminal line number from which the print job originated, 5) the date the print job was created, 6) status of the print job, 7) the options chosen, 8) the number of frames each print job occupies, and 9) number of copies to be printed.

The job status will be one of the following:

OPEN	A user process is writing to the job.
WAIT	The job is closed and waiting to be printed.
PRINT	The job is being printed.
FINISH	The job has been printed.

At any given time, more than one of the status conditions listed above may exist; however, only one status condition will be displayed. The display priority is: FINISH, PRINT, WAIT, OPEN.

If there are no print jobs assigned to any form queues and the SP-JOBS verb is executed, the message "THE QUEUE IS EMPTY" will be displayed on the screen.

The action codes listed below immediately follow the screen displays of the SP-JOBS verb:

1. MOVE FORM QUEUE	5. DELETE JOB	9. SUSPEND PRINTING
2. MOVE PRINT JOBS	6. STOP PRINTING	10. CHANGE PRIORITY
3. CHANGE OPTIONS	7. RESUME PRINTING	11. SP-STATUS
4. CHANGE #OF COPIES	8. EDIT PRINT JOB	99. EXIT

If you wish to execute an action code, enter its corresponding number in response to the prompt "ENTER ACTION CODE / PAGE NUMBER (P#)?". As soon as an action code is executed, an updated SP-JOBS screen is displayed. To page through the SP-JOBS screen(s), press the RETURN key. However, if you wish to skip to a particular page of the SP-JOBS screen, enter a "P", followed by a page number. For example, if you enter a "P22" to the "ENTER ACTION CODE / PAGE NUMBER (P#)?" prompt, all jobs listed on page 22 appear on the screen. Should you enter a number larger than the number of SP-JOBS pages, an error message appears and you are reprompted.

To display the SP-STATUS screen, enter Action Code 11. To exit the display screen, enter Action Code 99.

The SP-JOBS action codes are explained in the following sections. Figure A illustrates the format of the SP-JOBS verb; Figure B is a sample screen display which follows the execution of the SP-JOBS verb.

```
SP-JOBS
```

Figure A. Format of the SP-JOBS Verb

```
:SP-JOBS <cr>

                                PRINT JOBS

 QUEUE NAME     PRINT JOB   ACCOUNT   LINE   DATE CREATED   STATUS   OPT   SIZE   COPIES

 INVOICES          1          ACT       9      12 AUG       PRINT    H      12      1
 INVOICES          2          ACT       2      12 AUG       WAIT     H      15      1
 INVOICES          3          MANF      3      12 AUG       OPEN     H      7       1
 CHECK             4          PERS      5      12 AUG       WAIT             27      1
 SAMPLE            5          PROD      25     12 AUG       WAIT     HS     30      1
 INVOICES          6          MANF      3      12 AUG       WAIT     H      15      1
 INVOICES          7          ACT       2      12 AUG       WAIT     H      9       1
 INVOICES          8          MANF      10     12 AUG       WAIT     H      27      1
  .                .          .         .       .           .        .      .       .
  .                .          .         .       .           .        .      .       .
  .                .          .         .       .           .        .      .       .

     1.  MOVE FORM QUEUE        5.  DELETE JOB            9.  SUSPEND PRINTING
     2.  MOVE PRINT JOB         6.  STOP PRINTING        10.  CHANGE PRIORITY
     3.  CHANGE OPTIONS         7.  RESUME PRINT         11.  SP-STATUS
     4.  CHANGE #OF COPIES      8.  EDIT PRINT JOB       99.  EXIT

ENTER ACTION CODE / PAGE NUMBER (P#)?
```

Figure B. Sample SP-JOBS Screen Display

8.15 Moving a Form Queue

To transfer all print jobs in one queue to a different queue, enter the SP-JOBS Action Code 1 (Move Form Queue) or the SP-MOVEQ verb.

The general form of the Move Form Queue Action Code is:

```
ENTER ACTION CODE / PAGE NUMBER (P#)?1 <cr>
FROM-FORM-QUEUE  TO-FORM-QUEUE:
PASSWORD:
```

where:

FROM-FORM-QUEUE	is prompting for the name of the current form queue.
TO-FORM-QUEUE	is prompting for the name of the form queue accepting the print jobs.
PASSWORD:	is prompting for the password of the account which created the form queue. If the account has no password or the password is the same as the current account, you are not prompted for the password.

The format of the SP-MOVEQ verb is:

```
SP-MOVEQ from-form-queue to-form-queue
```

If the printing of a form queue has been "killed" by the SP-KILL verb before the execution of the Move Form Queue Action Code or the SP-MOVEQ verb, the interrupted job and all subsequent jobs in the queue will be moved.

NOTE: Print jobs having a status of "FINISH" are not moved. Use Action Code 2 (Move Print Job) or SP-SWITCH to move such a job to another form queue (see the topic MOVING PRINT JOBS" for more information).

Figure A and B illustrate the use of the Move Form Queue Action Code and SP-MOVEQ verb. Figure C displays the SP-JOBS screen which would follow the execution of the example in Figure A or Figure B.

```
:SP-JOBS <cr>

                                PRINT JOBS

QUEUE NAME    PRINT JOB   ACCOUNT   LINE   DATE CREATED   STATUS   OPT   SIZE   COPIES

INVOICES          1         ACT       9      12 AUG       PRINT    H      12      1
INVOICES          2         ACT       2      12 AUG       WAIT     H      15      1
INVOICES          3         MANF      3      12 AUG       OPEN     H       7      1
CHECK             4         PERS      5      12 AUG       WAIT             27      1
SAMPLE            5         PROD     25      12 AUG       WAIT     HS     30      1
INVOICES          6         MANF      3      12 AUG       WAIT     H      15      1
INVOICES          7         ACT       2      12 AUG       WAIT     H       9      1
INVOICES          8         MANF     10      12 AUG       WAIT     H      27      1

     1. MOVE FORM QUEUE          5. DELETE JOB            9. SUSPEND PRINTING
     2. MOVE PRINT JOB           6. STOP PRINTING        10. CHANGE PRIORITY
     3. CHANGE OPTIONS           7. RESUME PRINTING      11. SP-STATUS
     4. CHANGE #OF COPIES        8. EDIT PRINT JOB       99. EXIT

ENTER ACTION CODE / PAGE NUMBER (P#)?1<cr>
FROM-FORM-QUEUE  TO-FORM-QUEUE:
   SAMPLE CHECK <cr>
PASSWORD:TAB <cr>
```

The Move Form Queue Action Code moves all print jobs from form queue SAMPLE to CHECK.

Figure A. Sample Usage of Move Form Queue Action Code

```
:SP-MOVEQ SAMPLE CHECK <cr>
 PASSWORD:TAB <cr>
:SP-JOBS <cr>
```

This SP-MOVEQ verb performs the same function as Action Code 1 in Figure A.

Figure B. Sample Usage of SP-MOVEQ Verb

```
                                PRINT JOBS

QUEUE NAME    PRINT JOB   ACCOUNT   LINE   DATE CREATED   STATUS   OPT   SIZE   COPIES

INVOICES          1         ACT       9      12 AUG       PRINT    H      12      1
INVOICES          2         ACT       2      12 AUG       WAIT     H      15      1
INVOICES          3         MANF      3      12 AUG       OPEN     H       7      1
CHECK             4         PERS      5      12 AUG       WAIT             27      1
CHECK             5         PROD     25      12 AUG       WAIT     HS     30      1
INVOICES          6         MANF      3      12 AUG       WAIT     H      15      1
INVOICES          7         ACT       2      12 AUG       WAIT     H       9      1
INVOICES          8         MANF     10      12 AUG       WAIT     H      27      1
```

Figure C. SP-JOBS Screen Display

8.16 Moving Print Jobs

The SP-JOBS Action Code 2 (Move Print Job) and the SP-SWITCH verb transfer one print job or a list of print jobs to another queue.

The general form of the Move Print Job Action Code is:

```
ENTER ACTION CODE / PAGE NUMBER (P#)?2 <cr>
NEW-FORM-QUEUE  PRINT-JOB-LIST:
PASSWORD:
```

where:

NEW-FORM-QUEUE	is prompting for the name of the form queue accepting the print job(s).
PRINT-JOB-LIST	is prompting for the number of each print job to be moved. Each print job number is separated by one space.

The format of the SP-SWITCH verb is:

```
SP-SWITCH new-form-queue print-job-list
```

A job currently being printed cannot be moved to a new queue unless the Suspend Printing Action Code or the SP-SUSPEND verb has been issued to the current form queue. When the job is moved, it will be reprinted from the beginning of the job.

Figure A and B illustrate the use of the Move Print Job Action Code and the SP-SWITCH verb. Figure C displays the SP-JOBS screen which would follow the execution of the example in Figure A or B.

```
:SP-JOBS <cr>

                                PRINT JOBS

                PRINT                          DATE
QUEUE NAME      JOB     ACCOUNT   LINE         CREATED     STATUS   OPT   SIZE   COPIES

INVOICES          1     ACT         9          12 AUG      PRINT    H      12      1
INVOICES          2     ACT         2          12 AUG      WAIT     H      15      1
INVOICES          3     MANF        3          12 AUG      OPEN     H       7      1
CHECK             4     PERS        5          12 AUG      WAIT             27      1
CHECK             5     PROD       25          12 AUG      WAIT     HS     30      1
INVOICES          6     MANF        3          12 AUG      WAIT     H      15      1
INVOICES          7     ACT         2          12 AUG      WAIT     H       9      1
INVOICES          8     MANF       10          12 AUG      WAIT     H      27      1

     1. MOVE FORM QUEUE          5. DELETE JOB              9. SUSPEND PRINTING
     2. MOVE PRINT JOB           6. STOP PRINTING          10. CHANGE PRIORITY
     3. CHANGE OPTIONS           7. RESUME PRINTING        11. SP-STATUS
     4. CHANGE #OF COPIES        8. EDIT PRINT JOB         99. EXIT

ENTER ACTION CODE / PAGE NUMBER (P#)?2 <cr>
NEW-FORM-QUEUE  PRINT-JOB-LIST:PART 2 <cr>
```

The Move Print Job Action Code moves print job #2 in INVOICES form queue to PART form queue (previously created by SP-STATUS Action Code 1).

Figure A. Sample Usage of Move Print Job Action Code

```
:SP-SWITCH PART 2 <cr>
:SP-JOBS <cr>
```

This SP-SWITCH verb performs the same function as Action Code 2 in Figure A.

Figure B. Sample Usage of SP-SWITCH Verb

```
                                PRINT JOBS

                PRINT                          DATE
QUEUE NAME      JOB     ACCOUNT   LINE         CREATED     STATUS   OPT   SIZE   COPIES

INVOICES          1     ACT         9          12 AUG      PRINT    H      12      1
PART              2     ACT         2          12 AUG      PRINT    H      15      1
INVOICES          3     MANF        3          12 AUG      OPEN     H       7      1
CHECK             4     PERS        9          12 AUG      WAIT             27      1
CHECK             5     PROD       25          12 AUG      WAIT     HS     30      1
INVOICES          6     MANF        3          12 AUG      WAIT     H      15      1
INVOICES          7     ACT         2          12 AUG      WAIT     H       9      1
INVOICES          8     MANF       10          12 AUG      WAIT     H      27      1
```

Figure C. SP-JOBS Screen Display

8.17 Changing Print Job Options

To change print job options, use the SP-JOBS Action Code 3 (Change Options) or the SP-OPTS verb.

The general form of the Change Options Action Code is:

```
ENTER ACTION CODE / PAGE NUMBER (P#)?3 <cr>
PRINT-JOB#  OPTIONS:
```

where:

PRINT-JOB#	is prompting for the number of the print job.
OPTIONS	is prompting for one or both of the symbols used to indicate those options that can be changed.

The format of the SP-OPTS verb is:

```
SP-OPTS print-job# options
```

By using this verb, previous options are deleted and new options are assigned to the print job. The options that may be changed are 'H' (hold print job until deleted) and 'S' (suppress output).

The options of a currently outputting job cannot be changed. If the "OPTIONS" prompt is answered with a <cr> only, the previous options are cancelled and the print job is left with no options.

Figure A and B illustrate the use of the Change Options Action Code and the SP-OPTS verb. Figure C displays the SP-JOBS screen that would follow the execution of the example in Figure A or B.

```
:SP-JOBS <cr>

                                PRINT JOBS

QUEUE NAME     PRINT JOB   ACCOUNT   LINE   DATE CREATED   STATUS   OPT   SIZE   COPIES

INVOICES           1         ACT       9      12 AUG       PRINT    H      12      1
PART               2         ACT       2      12 AUG       PRINT    H      15      1
INVOICES           3         MANF      3      12 AUG       OPEN     H       7      1
CHECK              4         PERS      9      12 AUG       WAIT             27      1
CHECK              5         PROD     25      12 AUG       WAIT     HS      30      1
INVOICES           6         MANF      3      12 AUG       WAIT     H       15      1
INVOICES           7         ACT       2      12 AUG       WAIT     H        9      1
INVOICES           8         MANF     10      12 AUG       WAIT     H       27      1

     1. MOVE FORM QUEUE          5. DELETE JOB              9. SUSPEND PRINTING
     2. MOVE PRINT JOB           6. STOP PRINTING          10. CHANGE PRIORITY
     3. CHANGE OPTIONS           7. RESUME PRINTING        11. SP-STATUS
     4. CHANGE #OF COPIES        8. EDIT PRINT JOB         99. EXIT

ENTER ACTION CODE / PAGE NUMBER (P#)?3 <cr>
PRINT-JOB#  OPTIONS:4 HS <cr>
```

The Change Options Action Code adds the 'H' and 'S' options to print job #4.

Figure A. Sample Usage of Change Options Action Code

```
:SP-OPTS 4 HS <cr>
:SP-JOBS <cr>
```

This SP-OPTS verb performs the same function as Action Code 3 in Figure A.

Figure B. Sample Usage of SP-OPTS Verb

```
                                PRINT JOBS

QUEUE NAME     PRINT JOB   ACCOUNT   LINE   DATE CREATED   STATUS   OPT   SIZE   COPIES

INVOICES           1         ACT       9      12 AUG       PRINT    H      12      1
PART               2         ACT       2      12 AUG       PRINT    H      15      1
INVOICES           3         MANF      3      12 AUG       OPEN     H       7      1
CHECK              4         PERS      9      12 AUG       WAIT     HS     27      1
CHECK              5         PROD     25      12 AUG       WAIT     HS     30      1
INVOICES           6         MANF      3      12 AUG       WAIT     H      15      1
INVOICES           7         ACT       2      12 AUG       WAIT     H       9      1
INVOICES           8         MANF     10      12 AUG       WAIT     H      27      1
```

Figure C. SP-JOBS Screen Display

8.18 Changing Number of Print Job Copies

Use the SP-JOBS Action Code 4 (Change Number of Copies) or the SP-COPIES verb to alter the number of times a print job is printed.

The general form of the Change Number of Copies Action Code is:

```
ENTER ACTION CODE / PAGE NUMBER (P#)?4 <cr>
PRINT-JOB#  COPIES:
```

where:

PRINT-JOB#	is prompting for the number of the print job whose "copies" parameter is to be changed.
COPIES	is prompting for the number of copies (1 - 100) to be printed.

The format of the SP-COPIES verb is:

```
SP-COPIES print-job-# copies
```

Execution of either of these two methods replaces the current "copies" parameter of a job waiting to be printed or a hold file that has been printed. The "copies" parameter of a currently printing job cannot be changed.

Figure A and B illustrate the use of the Change Number of Copies Action Code and the SP-COPIES verb. Figure C displays the SP-JOBS screen which would follow the execution of the example in Figure A or B.

```
:SP-JOBS <cr>
                                PRINT JOBS

              PRINT                        DATE
QUEUE NAME    JOB    ACCOUNT  LINE    CREATED    STATUS   OPT   SIZE   COPIES

INVOICES       1     ACT        9     12 AUG     PRINT    H      12      1
PART           2     ACT        2     12 AUG     PRINT    H      15      1
INVOICES       3     MANF       3     12 AUG     OPEN     H       7      1
CHECK          4     PERS       9     12 AUG     WAIT     HS     27      1
CHECK          5     PROD      25     12 AUG     WAIT     HS     30      1
INVOICES       6     MANF       3     12 AUG     WAIT     H      15      1
INVOICES       7     ACT        2     12 AUG     WAIT     H       9      1
INVOICES       8     MANF      10     12 AUG     WAIT     H      27      1

    1. MOVE FORM QUEUE          5. DELETE JOB            9. SUSPEND PRINTING
    2. MOVE PRINT JOB           6. STOP PRINTING        10. CHANGE PRIORITY
    3. CHANGE OPTIONS           7. RESUME PRINTING      11. SP-STATUS
    4. CHANGE #OF COPIES        8. EDIT PRINT JOB       99. EXIT

ENTER ACTION CODE / PAGE NUMBER (P#)?4 <cr>
PRINT-JOB# COPIES:4 3 <cr>
```

This Change #of Copies Action Code deletes the "1" copy designation for print job #4 and replaces it with "3".

Figure A. Sample Usage of Change Number of Copies Action Code

```
:SP-COPIES 4 3 <cr>
:SP-JOBS <cr>
```

This SP-COPIES verb performs the same function as Action Code 4 in Figure A.

Figure B. Sample Usage of SP-COPIES Verb

PRINT JOBS

QUEUE NAME	PRINT JOB	ACCOUNT	LINE	DATE CREATED	STATUS	OPT	SIZE	COPIES
INVOICES	1	ACT	9	12 AUG	PRINT	H	12	1
PART	2	ACT	2	12 AUG	PRINT	H	15	1
INVOICES	3	MANF	3	12 AUG	OPEN	H	7	1
CHECK	4	PERS	9	12 AUG	WAIT	HS	27	3
CHECK	5	PROD	25	12 AUG	WAIT	HS	30	1
INVOICES	6	MANF	3	12 AUG	WAIT	H	15	1
INVOICES	7	ACT	2	12 AUG	WAIT	H	9	1
INVOICES	8	MANF	10	12 AUG	WAIT	H	27	1

Figure C. SP-JOBS Screen Display

8.19 Deleting Print Jobs from a Queue

The SP-JOBS Action Code 5 (Delete Job) or the SP-DELETE verb removes print jobs from a form queue.

The general form of the Delete Job Action Code is:

```
ENTER ACTION CODE / PAGE NUMBER (P#)?5 <cr>
PRINT-JOB-LIST:
PASSWORD:
```

where:

PRINT-JOB-LIST:	is prompting for the print job numbers to be deleted. Each number is separated by one space.
PASSWORD:	is prompting for the password of the account from which the print job was created.

The format of the SP-DELETE verb is:

```
SP-DELETE print-job-list
PASSWORD:
```

After deleting a job, the overflow frames are returned to the system. A currently outputting print job may be deleted only if all queue output is killed and the queue is inactive. An open print job cannot be deleted until the job is closed and the job is placed in a "WAIT" or "FINISH" state.

If all print jobs do not have the same password, you will be prompted when a new password is needed.

Figure A and B illustrate the use of the Delete Job Action Code and the SP-DELETE verb. Figure C displays the SP-JOBS screen which would follow the execution of the example in Figure A or B.

```
:SP-JOBS <cr>

                                PRINT JOBS

                PRINT                          DATE
QUEUE NAME      JOB     ACCOUNT   LINE       CREATED     STATUS   OPT   SIZE   COPIES

INVOICES          1       ACT       9        12 AUG      PRINT    H      12      1
PART              2       ACT       2        12 AUG      PRINT    H      15      1
INVOICES          3       MANF      3        12 AUG      OPEN     H       7      1
CHECK             4       PERS      9        12 AUG      WAIT     HS     27      3
CHECK             5       PROD     25        12 AUG      WAIT     HS     30      1
INVOICES          6       MANF      3        12 AUG      WAIT     H      15      1
INVOICES          7       ACT       2        12 AUG      WAIT     H       9      1
INVOICES          8       MANF     10        12 AUG      WAIT     H      27      1

    1. MOVE FORM QUEUE          5. DELETE JOB            9. SUSPEND PRINTING
    2. MOVE PRINT JOB           6. STOP PRINTING        10. CHANGE PRIORITY
    3. CHANGE OPTIONS           7. RESUME PRINT         11. SP-STATUS
    4. CHANGE #OF COPIES        8. EDIT PRINT JOB       99. EXIT

ENTER ACTION CODE / PAGE NUMBER (P#)?5 <cr>
PRINT-JOB-LIST:5 <cr>
PASSWORD:TPH <cr>
```

This Delete Job Action Code eliminates print job #5 found in the CHECK form queue and returns the 30 frames the job used to overflow.

Figure A. Sample Usage of Delete Job Action Code

```
:SP-DELETE 5 <cr>
:SP-JOBS <cr>
```

This SP-DELETE verb performs the same function as Action Code 5 in Figure A.

Figure B. Sample Usage of SP-DELETE Verb

```
                                PRINT JOBS

                PRINT                          DATE
QUEUE NAME      JOB     ACCOUNT   LINE       CREATED     STATUS   OPT   SIZE   COPIES

INVOICES          1       ACT       9        12 AUG      PRINT    H      12      1
PART              2       ACT       2        12 AUG      PRINT    H      15      1
INVOICES          3       MANF      3        12 AUG      OPEN     H       7      1
CHECK             4       PERS      9        12 AUG      WAIT     HS     27      3
INVOICES          6       MANF      3        12 AUG      WAIT     H      15      1
INVOICES          7       ACT       2        12 AUG      WAIT     H       9      1
INVOICES          8       MANF     10        12 AUG      WAIT     H      27      1
```

Figure C. SP-JOBS Screen Display

8.20 Stopping the Printer Before or After Printing a Job

The SP-JOBS Action Code 6 (Stop Printing) and the SP-STOP verb may be used to stop the printer just after the printing of the current print job or just before the printing of a specified print job.

The general form of the Stop Printing Action Code is:

```
ENTER ACTION CODE / PAGE NUMBER (P#)?6 <cr>
FORM-QUEUE OR PRINT-JOB#:
```

where:

FORM-QUEUE — is prompting for the name of the queue whose printing will be halted.

PRINT-JOB-# — is prompting for the number of the print job whose printing will be delayed.

The format of the SP-STOP verb is:

```
SP-STOP form-queue or print-job#
```

If you wish to stop the printer just after the current job is printed, enter the name of the active form queue. Should you decide to stop the printer before printing a specified job, enter the number of that print job.

When you stop the printer prior to printing a specific job, all jobs in the queue will be printed until the specified job is encountered.

Figure A and B illustrate the use of the Stop Printing Action Code and the SP-STOP verb. Figure C displays the SP-STATUS display that reflects the interruption to the INVOICES form queue.

```
:SP-JOBS <cr>

                                PRINT JOBS

QUEUE NAME    PRINT JOB   ACCOUNT   LINE   DATE CREATED   STATUS   OPT   SIZE   COPIES

INVOICES          1         ACT       9      12 AUG       PRINT    H      12      1
PART              2         ACT       2      12 AUG       PRINT    H      15      1
INVOICES          3         MANF      3      12 AUG       OPEN     H       7      1
CHECK             4         PERS      9      12 AUG       WAIT     HS     27      3
INVOICES          6         MANF      3      12 AUG       WAIT     H      15      1
INVOICES          7         ACT       2      12 AUG       WAIT     H       9      1
INVOICES          8         MANF     10      12 AUG       WAIT     H      27      1

     1. MOVE FORM QUEUE          5. DELETE JOB              9. SUSPEND PRINTING
     2. MOVE PRINT JOB           6. STOP PRINTING          10. CHANGE PRIORITY
     3. CHANGE OPTIONS           7. RESUME PRINTING        11. SP-STATUS
     4. CHANGE #OF COPIES        8. EDIT PRINT JOB         99. EXIT

ENTER ACTION CODE / PAGE NUMBER (P#)?6 <cr>
FORM-QUEUE OR PRINT-JOB#:INVOICES <cr>
```

This Stop Printing Action Code stops the printer prior to printing job #6.

Figure A. Sample Usage of Stop Printing Action Code

```
:SP-STOP INVOICES <cr>
```

This SP-STOP verb performs the same function as Action Code 6 in Figure A.

Figure B. Sample Usage of SP-STOP Verb

```
:SP-STATUS <cr>

                          FORM QUEUE ASSIGNMENT

  QUEUE NAME    FORM TYPE    DEVICE ASSIGNED   DEVICE TYPE   DEVICE STATUS   NUMBER ENTRIES   PAGE SKIP

  INVOICES      INVOICE            1            LPTR          STOP              4              2
  CHECK         PAYROLL            1            TAPE          NOT READY         0              0
  LABEL         MAIL               2            PORT          ASSIGNED          0              0
  SAMPLE                           3            LPTR          ASSIGNED          0              0
  PAYABLES                         0            LPTR          NO DEVICE         0              0
  PART                             2            LPTR          ACTIVE            1              0
```

Figure C. SP-STATUS Screen Display

8.21 Restarting a Printer After a Forced Wait

Restarting a printer after a forced wait is accomplished by the SP-JOBS Action Code 7 (Resume Printing Action Code) or the SP-RESUME verb.

The general form of the Resume Printing Action Code is:

```
ENTER ACTION CODE / PAGE NUMBER (P#)?7 <cr>
FORM-QUEUE:
```

where:

FORM-QUEUE: is prompting for the name of the interrupted form queue.

The format of the SP-RESUME verb is:

```
SP-RESUME form-queue
```

One of these two methods must be used after correcting the problem responsible for a device status of NEXT REEL, PARITY or CABLE OFF, and may be used after the execution of Suspend Printing and Stop Printing Action Codes or SP-SUSPEND and SP-STOP verbs.

When the SP-RESUME or the Resume Printing Action Code is executed, printing will begin where the spooler left off.

Figure A and B illustrate the use of the Stop Printing Action Code and the SP-RESUME verb. Figure C displays the SP-STATUS screen which reflects the "ACTIVE" status of form queue INVOICES.

```
:SP-JOBS <cr>

                              PRINT JOBS

              PRINT                    DATE
QUEUE NAME    JOB     ACCOUNT  LINE    CREATED    STATUS   OPT   SIZE   COPIES

INVOICES       1      ACT        9     12 AUG     FINISH   H      12      1
PART           2      ACT        2     12 AUG     PRINT    H      15      1
INVOICES       3      MANF       3     12 AUG     OPEN     H       7      1
CHECK          4      PERS       9     12 AUG     WAIT     HS     27      3
INVOICES       6      MANF       3     12 AUG     WAIT     H      15      1
INVOICES       7      ACT        2     12 AUG     WAIT     H       9      1
INVOICES       8      MANF      10     12 AUG     WAIT     H      27      1

    1. MOVE FORM QUEUE          5. DELETE JOB               9. SUSPEND PRINTING
    2. MOVE PRINT JOB           6. STOP PRINTING           10. CHANGE PRIORITY
    3. CHANGE OPTIONS           7. RESUME PRINTING         11. SP-STATUS
    4. CHANGE #OF COPIES        8. EDIT PRINT JOB          99. EXIT

ENTER ACTION CODE / PAGE NUMBER (P#)?7 <cr>
FORM-QUEUE:INVOICES <cr>
```

This Resume Printing Action Code starts the printing of job #6 which was interrupted by Action Code 6 in previous section.

Figure A. Sample Usage of Resume Printing Action Code

```
:SP-RESUME INVOICES <cr>
```

This SP-RESUME verb performs the same function as Action Code 7 in Figure A.

Figure B. Sample Usage of SP-RESUME Verb

```
:SP-STATUS <cr>

                        FORM QUEUE ASSIGNMENT

  QUEUE       FORM        DEVICE       DEVICE    DEVICE       NUMBER     PAGE
  NAME        TYPE        ASSIGNED     TYPE      STATUS       ENTRIES    SKIP

  INVOICES    INVOICE        1         LPTR      ACTIVE          4         2
  CHECK       PAYROLL        1         TAPE      NOT READY       0         0
  LABEL       MAIL           2         PORT      ASSIGNED        0         0
  SAMPLE                     3         LPTR      ASSIGNED        0         0
  PAYABLES                   0         LPTR      NO DEVICE       0         0
  PART                       2         LPTR      ACTIVE          1         0
```

Figure C. SP-STATUS Screen Display

8.22 Interrupting a Printing Job

The SP-JOBS Action Code 9 (Suspend Printing) and the SP-SUSPEND verb interrupt the printing of the current print job but do not delete the job from the queue. The SP-KILL verb also halts the output of the current print job and deletes that job if it is not a hold file.

The format of the Suspend Printing Action Code is:

```
ENTER ACTION CODE / PAGE NUMBER (P#)?9 <cr>
FORM-QUEUE --OR-- DEVICE# TYPE
```

FORM-QUEUE	is prompting for the name of the interrupted form queue. If you enter the form queue name, do not enter device# and type.
DEVICE# TYPE	is prompting for the device number and type of assigned device. The form queue name is not necessary if you enter the device number and type.

The format of the SP-SUSPEND verb is:

```
SP-SUSPEND form-queue --or-- device# and type
```

When either of these two methods is initiated, the interruption to the printing job is immediate. To restart printing where the spooler left off, use Action Code 7 (Resume Printing) or the SP-RESUME verb. To resume printing at any location you wish, issue the 'P' command of the Edit Print Job Action Code and the SP-EDIT verb.

The format of the SP-KILL verb is:

```
SP-KILL form-queue --or-- device# and type
```

When the SP-KILL verb is executed, the currently printing job is halted and the next job in the queue begins printing immediately. If the interrupted job is a hold file, it will remain in the job list; if the job is not a hold file, it will be deleted.

Figure A and B illustrate the use of the Suspend Printing Action Code and the SP-SUSPEND verb. Figure C displays the SP-STATUS screen which would follow the execution of the example in Figure A or B. An SP-JOBS screen is not shown here because the actual queue status can only be seen on the SP-STATUS display.

```
:SP-JOBS <cr>

                                PRINT JOBS

QUEUE NAME    PRINT JOB   ACCOUNT   LINE   DATE CREATED   STATUS   OPT   SIZE   COPIES

INVOICES          1         ACT       9      12 AUG       FINISH   H      12      1
PART              2         ACT       2      12 AUG       PRINT    H      15      1
INVOICES          3         MANF      3      12 AUG       OPEN     H       7      1
CHECK             4         PERS      9      12 AUG       WAIT     HS     27      3
INVOICES          6         MANF      3      12 AUG       PRINT    H      15      1
INVOICES          7         ACT       2      12 AUG       WAIT     H       9      1
INVOICES          8         MANF     10      12 AUG       WAIT     H      27      1

     1. MOVE FORM QUEUE          5. DELETE JOB              9. SUSPEND PRINTING
     2. MOVE PRINT JOB           6. STOP PRINTING          10. CHANGE PRIORITY
     3. CHANGE OPTIONS           7. RESUME PRINTING        11. SP-STATUS
     4. CHANGE #OF COPIES        8. EDIT PRINT JOB         99. EXIT

ENTER ACTION CODE / PAGE NUMBER (P#)?9 <cr>
FORM-QUEUE  --OR--  DEVICE#  TYPE:INVOICES <cr>
```

This Suspend Printing Action Code interrupts the printing of print job #6.

Figure A. Sample Usage of Suspend Printing Action Code

```
:SP-SUSPEND 1 LPTR <cr>
```

This SP-SUSPEND verb performs the same function as Action Code 9 in Figure A.

Figure B. Sample Usage of SP-SUSPEND Verb

```
:SP-STATUS <cr>

  QUEUE      FORM       DEVICE      DEVICE      DEVICE       NUMBER     PAGE
  NAME       TYPE       ASSIGNED    TYPE        STATUS       ENTRIES    SKIP

  INVOICES   INVOICE        1       LPTR        SUSPEND          4        2
  CHECK      PAYROLL        1       TAPE        NOT READY        0        0
  LABEL      MAIL           2       PORT        ASSIGNED         0        0
  SAMPLE                    3       LPTR        ASSIGNED         0        0
  PAYABLES                  0       LPTR        NO DEVICE        0        0
  PART                      2       LPTR        ACTIVE           1        0
```

Figure C. SP-STATUS Screen Display

8.23 Examining Jobs Waiting to be Printed

The SP-JOBS Action Code 8 (Edit Print Job) and the SP-EDIT verb manipulate a print job for display, forms alignment and/or printing a previously interrupted print job.

The general form of the Edit Print Job Action Code is:

```
ENTER ACTION CODE / PAGE NUMBER (P#)?8 <cr>
PRINT-JOB#:
PASSWORD:
*ENTER COMMAND:
```

where:

PRINT-JOB#: is prompting for the number of the print job to be edited.

PASSWORD: is prompting for the password of the account from which the job was created. If you do not know the password, press RETURN until you get back to TCL.

*ENTER COMMAND: is prompting for the command to be executed.

The format of the SP-EDIT verb is:

```
SP-EDIT print-job#
PASSWORD:
*ENTER COMMAND:
```

These two methods allow the use of the following commands:

Command	Explanation
L{n}	List "n" lines.
L{n}"string"	Locate "string" in next "n" lines.
P	Print to the end of the job.
B{m}	Backward postioning "m" pages.
F{m}	Forward positioning "m" pages.
A{n}	Alignment of "n" lines in masked form.
T	Top of report.
EX	Exit.

These commands are detailed in the following sections.

Figure A and B illustrate the use of the Edit Print Job Action Code and the SP-EDIT verb. Figure C shows illustrates sample usage of each edit command.

```
ENTER ACTION CODE / PAGE NUMBER (P#)?8 <cr>
PRINT-JOB#:4 <cr>
PASSWORD:TECS <cr>
*ENTER COMMAND:L22 <cr>
```

The Edit Print Job Action Code displays the first 22 lines of print job #4.

Figure A. Sample Usage of Edit Print Job Action Code

```
:SP-EDIT 5 <cr>
PASSWORD:RDL <cr>
*ENTER COMMAND:P <cr>
```

This SP-EDIT command causes Print Job #5 to be printed from beginning to end.

Figure B. Sample Usage of SP-EDIT Verb

`*ENTER COMMAND:L25 <cr>`	*Lists first 25 lines.*
`*ENTER COMMAND:L8"SIZE" <cr>`	*Locates "SIZE" in next 8 lines.*
`*ENTER COMMAND:P <cr>`	*Prints the entire job starting from the current line position.*
`*ENTER COMMAND:B5 <cr>`	*Places current line position back 5 pages.*
`*ENTER COMMAND:F10 <cr>`	*Places current line position forward 10 pages.*
`*ENTER COMMAND:A7 <cr>`	*Aligns next 7 lines in masked form.*
`*ENTER COMMAND:T <cr>`	*Sets current line position to top of job.*
`*ENTER COMMAND:EX <cr>`	*Exits Editor.*

Figure C. Edit Command Formats

8.24 Edit Print Job Commands: List and Locate

The following two commands list and locate specified lines within a print job.

LIST "n" LINES

The general form of this command is:

```
L{n}
```

where "n" specifies number of lines.

This command lists a specified number of lines beginning from the current line position displayed on the terminal. If "n" is not specified, only one line is displayed on the screen, and the current print position is set at the next line.

Should a listing display more than one page, each "TOP OF FORM" will be seen on the screen.

Note: You should be aware of the current print position because subsequent EDIT commands are based on this position.

LOCATE A STRING

The general form of this command is:

```
L{n}"string"
```

where:

n	specifies the number of lines to be searched.
string	specifies the string to be located.

This command changes the current print position to the line that contains the character string specified in the command. If "n" is omitted, the entire print job is searched. If "n" is entered, the search is conducted within the specified number of lines. Any non-numeric delimiter may be used (where double quotes are shown above) to specify the character string as long as the delimiter itself is not part of the string.

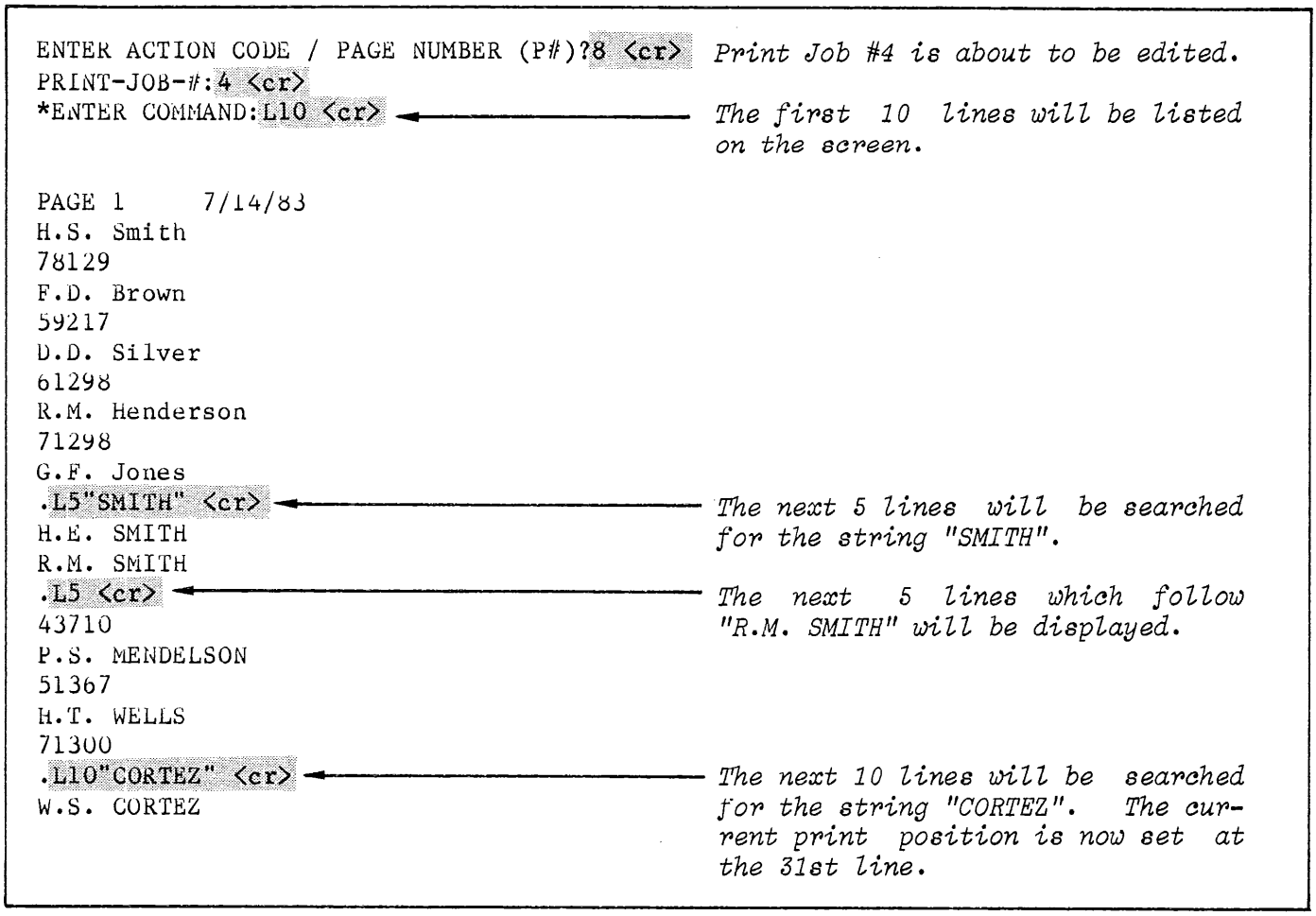


Figure A. Sample Usage of the List and Locate Commands

8.25 Edit Print Job Commands: Backward, Forward Positioning and Top

The Backward Positioning command moves the print position back a specified number of pages; the Forward Positioning command moves the print position forward a specified number of pages; the Top of Print Job command sets the print position at the top of the current job.

BACKWARD POSITIONING

The general form of this command is:

```
B{m}
```

where "m" specifies the number of pages.

This command controls the current print position. Positioning is set "m" pages backward from the current print position. If "m" is omitted, positioning is set at the top of the current print page. If "m" is greater than the number of previous pages, the position is set to the top of the print job.

FORWARD POSITIONING

The general form of this command is:

```
F{m}
```

where "m" specifies the number of pages.

This command moves the current print position forward "m" pages. If "m" is omitted, the position is moved to the top of the next print page. If "m" is greater than the number of remaining pages, the print position is set at the end of the job (EOF).

TOP OF PRINT JOB

The general form of this command is:

```
T
```

This command positions the current print position to the top of the print job. A "TOP" message is displayed following the execution of this command.

```
ENTER ACTION CODE / PAGE NUMBER (P#)?8 <cr>     Print Job #7 is about to be edited.
PRINT-JOB-#:7 <cr>
*ENTER COMMAND:L10 <cr>                         The first 10 lines of print job #7
                                                are listed.

PAGE 1      07/14/82
H.S. SMITH
78129
F.D. BROWN
59217
D.D. SILVER
61298
R.M. HENDERSON
71298
G.O. JONES
.F2 <cr>                                        Current line position is moved to
                                                top of Page 3.
.L1 <cr>                                        The first line of Page 3 is listed.
PAGE 3      07/14/82
.B2 <cr>                                        Current line position is set at the
                                                top of Page 2.
.L1 <cr>                                        First line of Page 2 is listed.
PAGE 2      07/14/82
.L"PAGE 3" <cr>                                 Current line position is set at the
PAGE 3      07/14/82                            line following the string "PAGE 3".
.B <cr>                                         Current line position is rolled
.L1 <cr>                                        back to the top of Page 3.
PAGE 3      07/14/82
.F3 <cr>                                        Current line position is rolled
                                                forward 3 pages.
.L1 <cr>                                        First line of Page 6 is listed.
PAGE 6      07/14/82
.T <cr>                                         Current line position is set to top
TOP                                             of job.
```

Figure A. Sample Use of Backward and Forward Positioning and Top Commands

8.26 Edit Print Job Commands: Alignment, Print and Exit

The Alignment command outputs to a printer the specified number of lines in masked form; the Print command begins output at the current line position and printing continues until the entire job is printed; the Exit command leaves the print job entirely.

ALIGNMENT

The general form of this command is:

A{n}

where "n" specifies the number of lines to be aligned on a printer. If "n" is not specified, the entire page is printed in masked form. Note that this command cannot be sent to a terminal port.

Masking is performed as follows:

X	represents any alphanumeric character
9	represents any numeric character
	All punctuation characters remain the same.

The current line position is set at the next line.

PRINT LINES

The format of this command is:

P

This command enables you to print a job. Printing begins at the current line position; therefore, you must first establish the current line position by using a List, Locate, Top, Forward or Backward command. Printing will then continue until the entire job is printed.

This command may also be used to restart a "suspended" job. The Editor is automatically exited by the Print command and the suspended status of the queue is cleared.

EXIT

The EXIT command format is:

EX

This command allows you to exit from the print job.

Figure A illustrates the use of Alignment, Print and Exit Commands. Figure B displays the SP-STATUS screen that would appear after restarting print job #6.

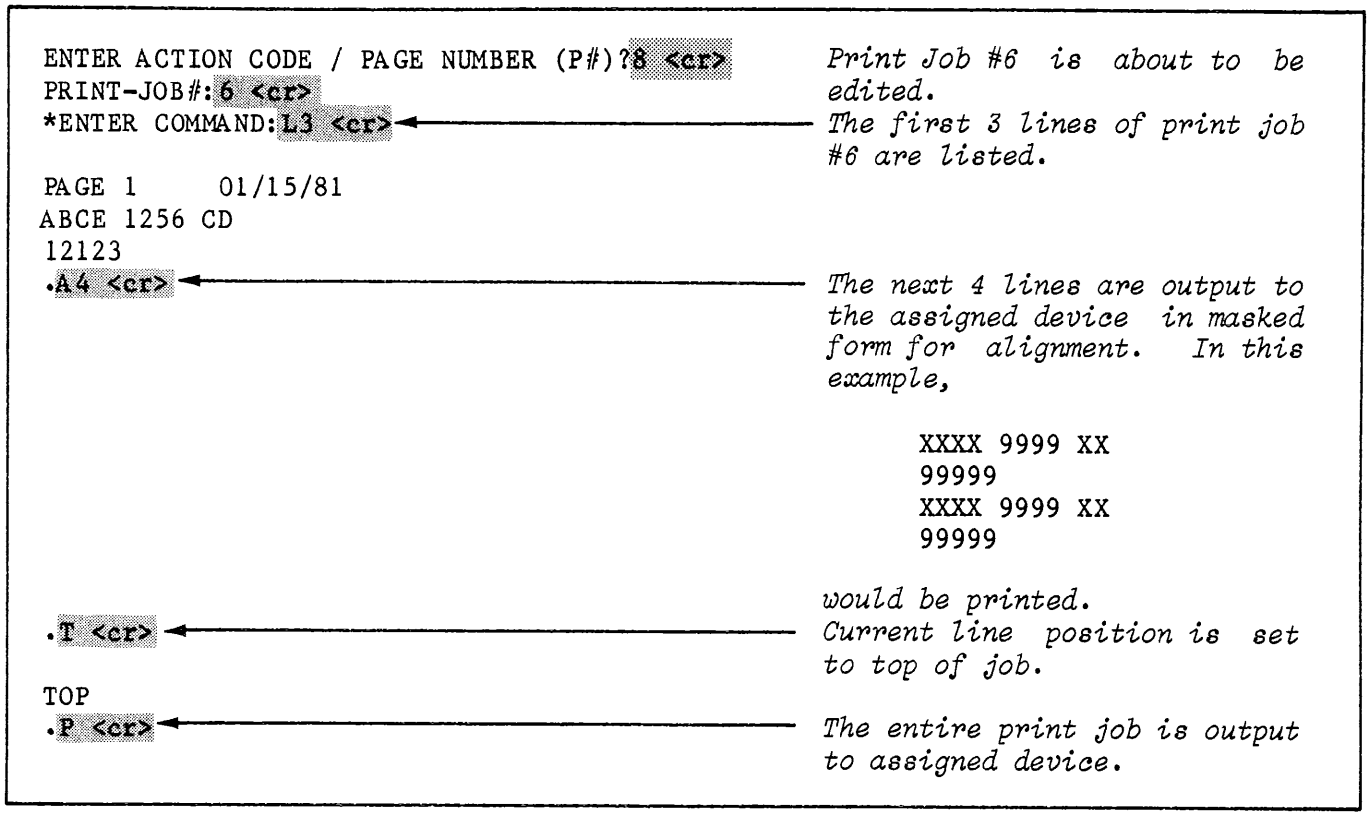


Figure A. Sample Usage of Alignment, Print and Exit Commands

```
:SP-STATUS <cr>

   QUEUE        FORM        DEVICE      DEVICE      DEVICE       NUMBER      PAGE
   NAME         TYPE        ASSIGNED    TYPE        STATUS       ENTRIES     SKIP

   INVOICES     INVOICE        1        LPTR        ACTIVE           4         2
   CHECK        PAYROLL        1        TAPE        NOT READY        0         0
   LABEL        MAIL           2        PORT        ASSIGNED         0         0
   SAMPLE                      3        LPTR        ASSIGNED         0         0
   PAYABLES                    0        LPTR        NO DEVICE        0         0
   PART                        2        LPTR        ACTIVE           1         0
```

Figure B. SP-STATUS Screen Display

8.27 Changing Priority of Print Jobs

The position of a print job within a queue may be changed by either the SP-JOBS Action Code 10 (Change Priority) or the SP-PRIORITY verb.

The format of the Change Priority Action Code is:

```
ENTER ACTION CODE / PAGE NUMBER (P#)?10 <cr>
PRINT-JOB#:
```

where:

PRINT-JOB#: is prompting for the number of the print job whose placement in the queue is to be changed.

The format of the SP-PRIORITY verb is:

```
SP-PRIORITY print-job#
```

Both of the above-mentioned methods alter the printing priority of jobs within a form queue. The job priority is changed by moving the specified print job to the top of the queue. The printing of this job commences immediately after the current job is printed.

Figure A and B illustrate the usage of the Change Priority Action Code and the SP-PRIORITY verb. Figure C displays the SP-JOBS screen which would follow the execution of the example in Figure A or B.

```
:SP-JOBS <cr>

                                FORM QUEUES

QUEUE          PRINT                          DATE
NAME           JOB      ACCT      LINE      CREATED      STATUS    OPT    SIZE    COPIES

INVOICES         1      ACT          9      12 AUG       FINISH    H        12       1
PART             2      ACT          2      12 AUG       PRINT     H        15       1
INVOICES         3      MANF         3      12 AUG       OPEN      H         7       1
CHECK            4      PERS         9      12 AUG       WAIT      HS       27       3
INVOICES         6      MANF         3      12 AUG       PRINT     H        15       1
INVOICES         7      ACT          2      12 AUG       WAIT      H         9       1
INVOICES         8      MANF        10      12 AUG       WAIT      H        27       1

   1. MOVE FORM QUEUE          5. DELETE JOB              9. SUSPEND PRINTING
   2. MOVE PRINT JOB           6. STOP PRINTING          10. CHANGE PRIORITY
   3. CHANGE OPTIONS           7. RESUME PRINTING        11. SP-STATUS
   4. CHANGE #OF COPIES        8. EDIT HOLD JOB          99. EXIT

ENTER ACTION CODE / PAGE NUMBER (P#)?10 <cr>
PRINT-JOB#:8 <cr>
```

The Change Priority Action Code moves job #8 to the top of the form queue and its printing will begin immediately after print job #6 is printed.

Figure A. Sample Usage of Change Priority Action Code

```
:SP-PRIORITY 8 <cr>

:SP-JOBS <cr>
```

This SP-PRIORITY verb performs the same function as Action Code 10 in Figure A.

Figure B. Sample Usage of SP-PRIORITY Verb

```
                                FORM QUEUES

QUEUE          PRINT                          DATE
NAME           JOB      ACCT      LINE      CREATED      STATUS    OPT    SIZE    COPIES

INVOICES         1      ACT          9      12 AUG       FINISH    H        12       1
PART             2      ACT          3      12 AUG       PRINT     H        15       1
INVOICES         3      MANF         3      12 AUG       OPEN      H         7       1
CHECK            4      PERS         9      12 AUG       WAIT      HS       27       3
INVOICES         6      MANF         3      12 AUG       FINISH    H        15       1
INVOICES         7      ACT          2      12 AUG       WAIT      H         9       1
INVOICES         8      MANF        10      12 AUG       PRINT     H        27       1
```

Figure C. SP-JOBS Screen Display

8.28 TCL Control of Ports: SP-PORTON PROC

The SP-PORTON PROC starts the process which designates a port as available for spooler output.

The general form of the SP-PORTON PROC is:

```
SP-PORTON line#  --or--  line# form-queue
SP-PORTON line# trans-code  --or--  line# form-queue trans-code
```

If you enter SP-PORTON <cr>, the system will respond with prompts for the required parameters, where:

line#	specifies the line to be used as a spooler port
form-queue	specifies the name of the queue the print job(s) are assigned to. If the form queue name is entered, you must also enter the line number.
trans-code	a translation code of 3 causes all spooler jobs to be translated to the new matrix printer commands before despooling.

The SP-PORTON PROC logs on a port as an available device for spooler output. A PORT device cannot output print jobs until the SP-PORTON PROC is executed. Once SP-PORTON is executed, the attached device (which can be a CRT terminal, serial character printer or other device) can be controlled by spooler verbs.

The line# may be any port number (the maximum number of ports is determined by the current system configuration), or it may be an asterisk (*), which specifies that print jobs will be sent to the terminal which originated the SP-PORTON PROC. Port zero cannot be used as a spooler port.

Currently, only translation code 3 is allowed. This code translates spooled output created for the old matrix printer into the proper characters used by the new matrix printer. User programs that spool data with the old matrix printer commands need not be modified for use with the new matrix printer if the port is set up for trans-code 3. A program directly utilizing the new commands in spooled output should not use a translation code.

If a port process is initiated on the line on which it is going to run, it is considered a "normal" process, and can be interrupted by the <BREAK> key. If SP-PORTON is initiated from another line with or without the optional queue name, it is a TIPH phantom process. If a queue name is specified, it will be used as the TASK ID of the TIPH process. See the chapter titled TERMINAL INDEPENDENT PROCESS HANDLER (TIPH) for a detailed explanation of TIPH.

NOTE

Do not log off an SP-PORTON line. Use the SP-PORTOFF verb to terminate an SP-PORTON assignment.

There are four conditions under which the SP-PORTON PROC may be used:

Condition #1

Form queue TEST is created and assigned to a "terminal" device type (no device number assigned) prior to executing SP-PORTON. Subsequent use to the SP-PORTON PROC with the queue name TEST causes the print jobs to be output to the specified port. The SP-STATUS display reflects a new line number in the "DEVICE ASSIGNED" column.

Condition #2

Form queue RECPTS is created without a device or device number assigned prior to executing SP-PORTON. The use of SP-PORTON with the queue name RECPTS automatically assigns that queue to the specified port, and all print jobs assigned to form queue RECPTSare sent to that port. The SP-STATUS display reflects a new line number in the "DEVICE ASSIGNED" column and a "PORT" designation in the "DEVICE TYPE" column.

Figure A illustrates Condition #1 and Condition #2.

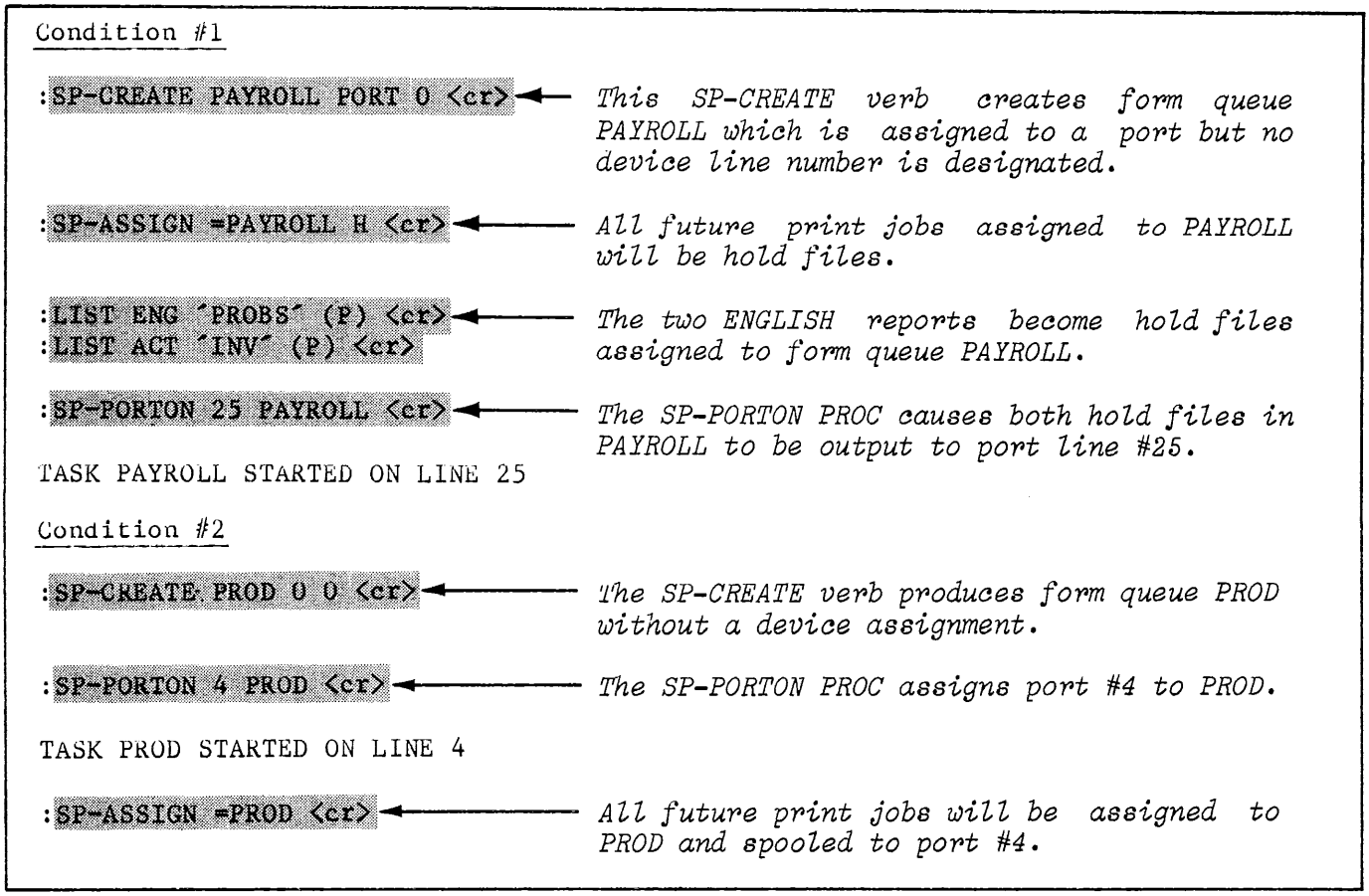


Figure A. Sample Usage of SP-PORTON PROC

Condition #3

A form queue does not exist. First, the SP-PORTON PROC is executed. Next, the form queue is created with a port device number. The use of SP-PORTON without a form queue name specified activates the port process, which waits until a form queue is created with the same device number as the port line number. The SP-STATUS display reflects the device designation only after the form queue is created.

Condition #4

Form queue INVS has been assigned to a port that has a line number assigned prior to executing SP-PORTON. The use of SP-PORTON first activates the port process, which will then find the matching device number assigned to the port. The associated print job is output to the device on the specified port. The SP-STATUS display reflects the device assignment.

Figure B illustrates Condition #3 and Condition #4.

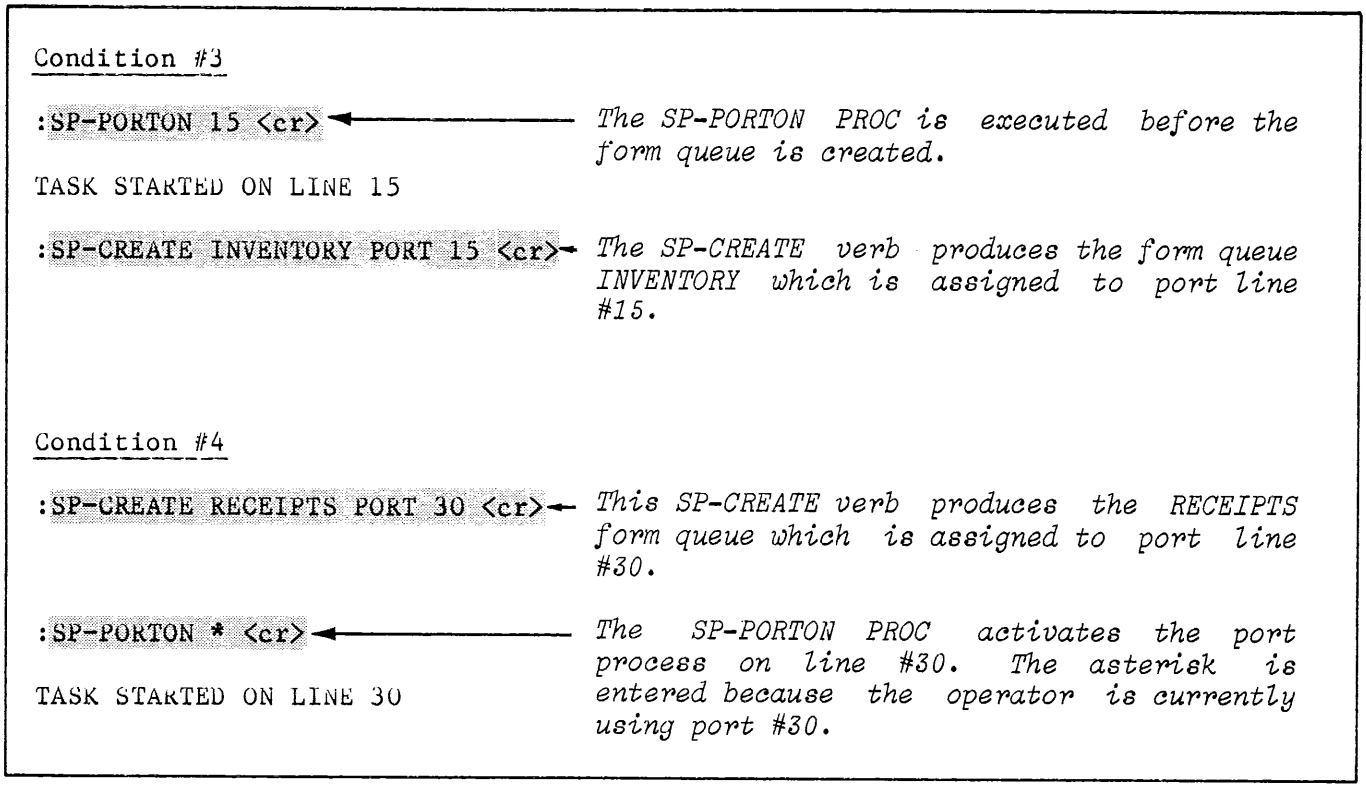


Figure B. Sample Usage of SP-PORTON PROC

8.29 TCL Control of Ports: SP-PORTOFF Verb

The SP-PORTOFF verb may log off an SP-PORTON line before or after output is finished.

The general form of the SP-PORTOFF verb is:

```
SP-PORTOFF form-queue-name  --or--  line#

     OR

SP-PORTOFF <cr>
ENTER LINE#  --OR--  FQ-NAME:
```

where:

form-queue-name	specifies the name of the form queue that was assigned to the port
line#	specifies the line used as the spooler port

An asterisk (*) cannot be specified as a "line number". All form queue names must begin with either an alphabetic or a special character (except *). If the first character is numeric, it is treated as a "line number". If SP-PORTOFF is used before output is finished, SP-PORTON may be used to restart the port process. Output will begin at the first line of the job.

Figure A illustrates sample usage of the SP-PORTOFF Verb.

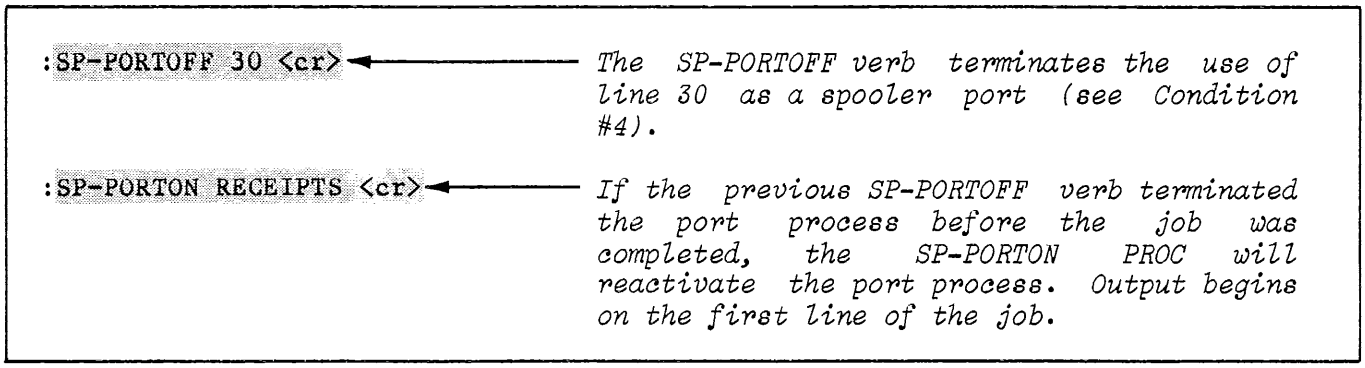


Figure A. Sample Usage of SP-PORTOFF Verb

8.30 Printer Control: P-ATT and P-DET

The P-ATT verb attaches your terminal line to the specified printer. The P-DET verb detaches your terminal line from the specified printer.

The P-ATT verb has the following general form:

P-ATT {form-queue-name or device#}

where:

form-queue-name	specifies the name of the queue assigned to the desired printer.
device#	specifies the printer number (which can be a number from 1 through 4). If no parameters are given, printer #1 is attached.

If you are already attached to the printer, only the TCL prompt is returned. If the printer is attached to another line, the following message is displayed on the screen:

LPTR ATTACHED TO LINE n.

where "n" represents the user's line number (or spooler process) the printer is attached to. If the attachment is successful, the following message is displayed:

LPTR ATTACHED.

The format of the P-DET verb is:

P-DET {form-queue-name or device#} {(U)}

The P-DET verb releases printer attachment, but only if your line is currently attached to the printer. Otherwise, the following message is displayed:

[655] NOT ATTACHED!

Users with SYS2 privileges may use the unconditional form (i.e., P-DET (U)). This will detach the printer from any line except the spooler (which detaches automatically when the queue is empty). An attempt to use this form without SYS2 privileges, or an attempt to detach the printer from the spooler, will lead to the following message:

[660] INSUFFICIENT PRIVILEGES OR DEVICE ATTACHED TO SPOOLER.

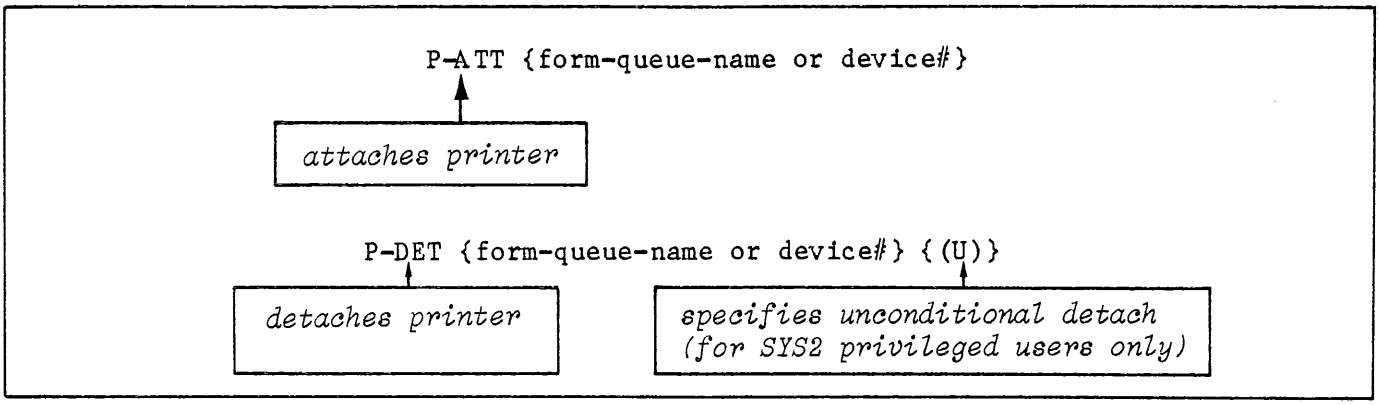


Figure A. General Form of P-ATT and P-DET Verbs

EXAMPLES	EXPLANATION
:P-ATT <cr>	*Attaches printer #1 (default printer).*
:P-ATT INVOICES <cr>	*Attaches printer assigned to form queue INVOICES.*
:P-DET <cr>	*Detaches printer assigned to user.*
:P-DET PAYROLL <cr>	*Detaches printer assigned to form queue PAYROLL.*
:P-DET 4 (U) <cr>	*Detaches printer #4, even if attached to another line (for SYS2 privileged users only).*
:SP-ASSIGN <cr>	*All user spooler assignments are cleared and all devices attached for 'N' option output are cleared. Subsequent print jobs will be directed to the STANDARD form queue.*

Figure B. Sample Usage of P-ATT and P-DET Verbs

8.31 Tape Control: T-ATT, T-DET and SP-TAPEOUT Verbs and T-PRINT PROC

The T-ATT and T-DET verbs attach and detach the tape unit. Further magnetic tape unit control is provided by the T-PRINT PROC and the SP-TAPEOUT verb.

The T-ATT verb has the following format:

```
T-ATT {n} {(options)}
```

The T-ATT verb attaches your line to the magnetic tape unit where "n" represents a tape block size (see chapter titled MAGNETIC TAPE UNIT COMMANDS). If "n" is omitted, a default block size of 512 bytes is assumed. Options are described in Chapter 9. If the tape unit is attached to another line, the message "TAPE ATTACHED TO LINE n" is displayed.

If attachment is successful, this message is displayed: [92] TAPE ATTACHED, RECORD SIZE = n.

The general form of the T-DET verb is:

```
T-DET {(U)}
```

The T-DET verb releases tape unit attachment, but only if the your line is currently attached to the tape unit. Otherwise, the message "[655] NOT ATTACHED!" is displayed.

Users with SYS2 privileges may use the unconditional form (i.e., T-DET (U)). This detaches the tape unit from any line except the spooler (which detaches automatically at the end of each job). An attempt to use this form without SYS2 privileges, or an attempt to detach the tape unit from the spooler will lead to the following message: [660] INSUFFICIENT PRIVILEGES OR DEVICE ATTACHED TO SPOOLER.

NOTE: Use the unconditional form of the T-DET verb with caution. This verb should not be executed while another user is outputting to tape.

The T-PRINT PROC has the following general form:

```
T-PRINT {"string"}
```

This is a PROC which uses as input a reel of magnetic tape previously created by spooling to tape. When T-PRINT is entered, the following steps are performed:

1. The tape unit is attached.
2. The magnetic tape is rewound to the load point.
3. The tape is searched for the "string" (if used).
4. Output of the report proceeds from the tape to the designated printer. Printing begins with the first line which contains the "string".
5. The tape is still attached to the line after T-PRINT is done.

The general form of the SP-TAPEOUT verb is:

```
SP-TAPEOUT {"string"}
```

This verb is used by the T-PRINT PROC. No device assignment is altered and no tape rewind is done. This verb can be used to build custom PROCs requiring tape jobs to be spooled. The next job on the magnetic tape is spooled according to the user's current SP-ASSIGN options.

The verbs T-ATT, T-DET, T-PRINT and SP-TAPEOUT are summarized in Figure A. Examples of these verbs are shown in Figure B.

T-ATT {n} {(options)}	*Attaches magnetic tape unit.*
T-DET {(U)}	*Detaches magnetic tape unit. 'U' specifies unconditional detach (for SYS2 privileged users only).*
T-PRINT {"string"}	*Tape and printer are attached, tape rewound, and line-at-a-time printing begins with first occurrence of optional "string".*
SP-TAPEOUT {"string"}	*Next tape job is spooled according to current SP-ASSIGN options, starting with first line containing optional "string".*

Figure A. Format of T-ATT, T-DET AND SP-TAPEOUT Verbs and T-PRINT PROC

EXAMPLE	EXPLANATION
:T-ATT <cr>	*Attaches tape unit. Default record size of 512 bytes is assigned.*
:T-DET <cr>	*Detaches tape unit.*
:T-DET (U) <cr>	*Detaches tape unit even if attached to another line (for SYS2 privileged users only).*
:T-PRINT "XYZ123" <cr>	*Tape unit and printer are attached, tape unit is rewound, and line-at-a-time printing begins with first line containing string "XYZ123".*
:SP-TAPEOUT <cr>	*Next job on tape is spooled.*

Figure B. Sample Usage of T-ATT, T-DET and SP-TAPEOUT Verbs and T-PRINT PROC

8.32 Spooler Activation: :RESTART-SPOOLER and :SP-NEWTAB Verbs

The spooler is automatically activated during job restore process. For further information regarding this process, refer to Chapter 11, SYSTEM MAINTENANCE PROCEDURES. The spooler may also be restarted with the :RESTART-SPOOLER verb; should :RESTART-SPOOLER fail, :SP-NEWTAB may be used to reinitialize the spooler.

The spooler may be activated under the following two conditions: 1) print jobs and queue data in use are retained by the spooler, and 2) print jobs and queue data are lost to the system.

The first condition occurs after a system restore with an "X" or "A" option, a Binary Restore, or after the :RESTART-SPOOLER verb is used. The spooler retains the addresses of its tables and all open print jobs are closed; therefore, no queue data or print jobs are lost.

The :RESTART-SPOOLER verb should only be used when neither the printer or tape unit is attached to a user's line and when no line is generating a spooled job. This verb may be used in an emergency situation such as a spooler abort. (The spooler has aborted if the message "SPOOLER ABORTED. RESTART BEING ATTEMPTED." appears on the SP-STATUS display.)

The second condition occurs after a system restore with an 'X', 'A', 'F' or 'AF' restore and after the verbs :FILES, :ABS/FILES, and :SP-NEWTAB are used. Under such circumstances, all spooler tables and jobs are lost, including hold jobs, and the tables are reinitialized.

THIS PAGE INTENTIONALLY LEFT BLANK.

9.1 Magnetic Tape Units

This chapter discusses the use of magnetic tape units, and the associated verbs. The system can accommodate up to four tape units.

The verbs which will be discussed in this chapter are:

T-ATT
ASSIGN
CLEAR-ASSIGN
T-DET
T-FWD
T-BCK
T-REW
T-WEOF
T-CHK
T-READ
T-DUMP
ST-DUMP
T-LOAD
MT-LOAD
MT-VERIFY
T-RDLBL
T-COPY

This chapter will also discuss tape unit sequencing for multiple tape units, tape labels, and multiple reel tape operations.

9.2 Magnetic Tape Unit Sequencing

This section discusses magnetic tape unit sequencing during tape operations.

The system will begin operations with the tape unit that is first in the unit-list of the channel being used. It will use the units associated with the channel in the order they are specified in the unit-list. When the system reaches the end of the list, it returns to the beginning. When a tape is finished, the tape is rewound and the unit is put off-line. If the drive is ready when the system returns to use it again, the system will assume that a new tape has been mounted and will continue with the tape operation. If the drive is not ready, the prompt:

```
MOUNT REEL # n ON DRIVE m
TYPE 'C' TO CONTINUE
```

will be displayed.

When the system finishes a tape operation, it will "remember" the unit it finished on, and subsequent operations on the channel will begin with this unit.

9.3 T-ATT Verb

The T-ATT verb attaches the magnetic tape unit to the terminal that issued the command and assigns a record size.

The general form of the T-ATT verb is:

```
T-ATT {n}
```

where "n" is the record size

The system will automatically attach tape unit 1. To attach other tape units, use the ASSIGN verb. The T-ATT verb must be issued before the tape unit is used. If it is not, the message:

```
NO SUCH TAPE ASSIGNMENT EXISTS
```

is displayed. When the tape unit is attached, the message:

```
DEVICE(S) TAPE 1 NOW ASSIGNED,
RECORD SIZE = n
```

is displayed, where "n" is the record size that was attached. When the tape is attached, other users are not able to access it. If you attempt to attach the tape when it is already attached to a user, the message:

```
DEVICE TAPE n IS ASSIGNED TO LINE m
```

is displayed, where "m" is the line number the tape unit is attached to.

The magnetic tape unit is automatically detached when you log off the system.

Examples of the use of the T-ATT verb are provided in Figure A.

Record Size

The record size may be from 20 to 32,256 bytes. If a record is not specified with the T-ATT verb, the default size of 8190 is used.

Tape labels generated by the system contain the record size. If the tape is positioned at a label and the record size specified with the T-ATT verb is less than the record size contained in the label, the system automatically uses the size contained in the label. The change in record size is indicated by the message:

```
TAPE SIZE CHANGED TO n
```

where "n" is the size obtained from the tape label.

If the tape is not positioned at a label, the system uses the record size specified with the T-ATT verb. If the attached size is smaller than the actual size, the record is truncated at the size specified with the T-ATT verb.

```
:T-ATT <cr>                                   No record size specified --
[1021] DEVICE(S) TAPE 1 NOW ASSIGNED,         default size used.
       RECORD SIZE = 8190


:T-ATT 2048 <cr>                              Tape was attached with a
[1021] DEVICE(S) TAPE 1 NOW ASSIGNED,         record size of 2048 bytes.
       RECORD SIZE = 2048
```

Figure A. Sample usage of the T-ATT Verb

9.4 ASSIGN Verb

The ASSIGN verb allows you to assign a unit list and a record size to a specified channel. This verb attaches the unit(s) assigned.

The general form of the ASSIGN verb is:

```
ASSIGN specification {;specification}...
```

where "specification" has the general form of:

```
{channel}=TAPE {unit-list} {SIZE=size}
```

"Channel", if present, is any alphabetic character from A to Z. If a channel is not specified, all existing assignments are destroyed, and the unit-list and record size specified are assigned to the default channel. If a channel is specified and the default channel exists, the default channel is destroyed. The ability to associate a channel name with a logical tape unit allows you to create applications and command streams which refer to channels by name.

The general form of "unit-list" is:

```
unit {,unit}...
```

where "unit" is 1, 2, 3, or 4.

"Unit-list" must be present. A unit may not be assigned to two channels at the same time, and may not appear more than once in a list. If a unit in the list has already been assigned to another channel, the message:

```
DEVICE TAPE n ASSIGNED TO THIS LINE, BUT FOR ANOTHER CHANNEL
```

is displayed, and the channel assignment is not changed.

"Size", if present, is a record size between 20 and 32,256. The default size is 8190.

```
:ASSIGN =TAPE 3
[1021] DEVICE(S) TAPE 3 NOW ASSIGNED,
       RECORD SIZE = 8190

:ASSIGN A=TAPE 1,3 SIZE=20
[1120] DEVICE(S) TAPE 1,3 NOW ASSIGNED TO CHANNEL 'A',
       RECORD SIZE = 20

:ASSIGN B=TAPE 2,4 SIZE=32256
[1120] DEVICE(S) TAPE 2,4 NOW ASSIGNED TO CHANNEL 'B',
       RECORD SIZE = 32256
```

Figure A. Sample Usage of the ASSIGN Verb

9.5 CLEAR-ASSIGN Verb

The CLEAR-ASSIGN verb allows you to destroy channel assignments.

The general form of the CLEAR-ASSIGN verb is:

```
CLEAR-ASSIGN {channel}
```

"Channel", if present, is any alphabetic character from A to Z. If a channel is not specified, all tape assignments are destroyed. If a channel is specified, only the assignment associated with the specified channel is destroyed.

CLEAR-ASSIGN operates only on channels associated with the user's own process.

```
:CLEAR-ASSIGN A                    Destroys the assignment
                                   associated with channel A.
```

Figure A. Sample Usage of the CLEAR-ASSIGN Verb

9.6 Control: T-DET, T-FWD, T-BCK, T-REW, T-WEOF, and T-CHK Verbs

This chapter describes the magnetic tape unit control verbs.

The general form of the T-DET verb is:

T-DET {(U)} {n}

where:

"(U)" allows you to detach the tape from any line except the spooler, if you have SYS2 privleges

"n" is the tape unit number specified with the "U" option

T-DET detaches tape unit 1. T-DET (U) requires that a unit (1 through 4) be specified.

The general form of the T-FWD verb is:

T-FWD {channel} {n}

where:

"channel" is any alphabetic character from A to Z

"n" is the number of records the tape will move forward

This verb moves the tape "n" records forward. The maximum value for "n" is 32,256. If "n" is not specified, the tape moves forward to the position immediately beyond the next end-of-file (EOF) mark. To perform multiple T-FWDs, use the SPACE PROC (refer to the topic MISCELLANEOUS PROCS).

The general form of the T-BCK verb is:

T-BCK {channel} {n}

where:

"channel" is any alphabetic character from A to Z

"n" is the number of records the tape will move backward

This verb moves the tape back "n" records. If "n" is not specified, the tape moves to the position immediately before the previous EOF mark, or to load point. Before reading the next record, a T-FWD must be issued to position the tape after the EOF mark.

:T-DET <cr>	Detaches tape unit 1.
:T-FWD B 10 <cr>	Moves the tape assigned to channel B forward 10 records.
:T-BCK B 2 <cr>	Moves the tape assigned to channel B back 2 records.

Figure A. Sample Usage of the T-DET, T-FWD, and T-BCK Verbs

The general form of the T-REW verb is:

T-REW {channel}

where "channel" is any alphabetic character from A to Z

This verb performs a "logical" rewind of the specified channel. Only the current drive is rewound. If there are multiple units in the channel's unit-list, the first tape unit in the list is set up as the "current" unit for the channel. Subsequent tape operations will begin with the "current" unit.

The general form of the T-WEOF verb is:

T-WEOF {channel}

where "channel" is any alphabetic character from A to Z

This verb writes an EOF mark on the tape.

The general form of the T-CHK verb is:

T-CHK {channel}

where:

"channel" is any alphabetic character from A to Z

This verb checks a tape file for parity errors. Error checking continues until an EOF mark is encountered.

The T-CHK verb attempts to reattach the tape to the maximum size of 8190 bytes and all records are then read under this buffer size. This allows all records to be completely checked for parity regardless of the current T-ATT size. At the end of the T-CHK, the tape buffer is reattached at its original size. To perform multiple T-CHKs, use the CHECK PROC (refer to the topic MISCELLANEOUS PROCS).

:T-REW B <cr>	Rewinds "logical" tape unit assigned to channel B.
:T-WEOF B <cr>	Writes an EOF mark on the tape assigned to channel B.
:T-CHK B <cr>	Checks the tape assigned to channel B for parity errors and specifies multiple reels.

Figure B. Sample Usage of the T-REW, T-WEOF, and T-CHK Verbs

9.7 Magnetic Tape I/O: T-READ Verb

The T-READ verb dumps the contents of a tape to the terminal or printer.

The general form of the T-READ verb is:

T-READ {channel} {(options)}

where:

"channel" is any alphabetic character from A to Z

"options" are:

X	Dumps in hexadecimal instead of character format
P	Dumps to the printer
N	Inhibits automatic paging of output to the terminal
n{-m}	Dumps records from "n" to "m" as specified, counting from the current position of the tape. If "m" is omitted, m=n is assumed. If the entire "n-m" option is omitted, all records up to the EOF will be dumped

This verb dumps the contents of a tape to the terminal, or optionally to the printer. The T-READ operation terminates when the specified number of records have been dumped, or when an EOF is detected.

Sample usage of the T-READ verb is illustrated in Figure A.

:T-READ (4-6) <cr>	Bypasses the first three records, dumps 4th, 5th and 6th on the terminal, and positions tape at beginning of 7th record.
:T-READ (P,4-6) <cr>	Same as above, but dumps to the printer.
:T-READ B (X) <cr>	Dumps the tape assigned to channel B in hex on the terminal until an EOF mark is detected.

Figure A. Sample Usage of the T-READ Verb

9.8 Magnetic Tape I/O: T-DUMP and ST-DUMP Verbs

The ENGLISH T-DUMP and ST-DUMP verbs cause selected file items to be dumped to the magnetic tape unit.

T-DUMP is an ENGLISH verb which allows you to dump selected file items to the magnetic tape unit. The ENGLISH verb ST-DUMP works like T-DUMP, except a SORT is performed prior to the dump. T-DUMP and ST-DUMP work only with the default channel. The general form of an ENGLISH input sentence using the T-DUMP verb is:

```
T-DUMP {DICT} file-name {item-list} {selection-criteria}
{HEADER "identification"} {(options)}
```

where:

"file-name" is the name of the file to dump

"item-list" is a list of the item-ids to dump

"selection-criteria" are specified output conditions

'"identification"' is the tape label information

"options" are:

I	Lists the item-ids as they are dumped
P	Prints the item-id list on the printer when used with the 'I' option
T	Inhibits writing the tape label

The file-name may be preceded by the DICT modifier to dump dictionary data. File definition items (D/CODE = D) are not dumped. The "item-list" and "selection-criteria" (described in the ENGLISH Programming Manual) cause a selected subset of items in the specified file to be dumped to tape. As in other ENGLISH statements, each item-id must be enclosed in single quotes. If the "item-list" and "selection-criteria" are omitted, all items are dumped. Note: The "item-list" may be supplied by a previous SELECT, SSELECT, GET-LIST or FORM-LIST verb, in which case the selection-criteria is omitted.

The "HEADER" option allows you to generate tape label identification. The "identification" material must be enclosed in double quotes. Labels are always written at the beginning of each tape file, unless specifically suppressed. For further information see the Tape Labels section in this chapter.

Figure A. illustrates the general form of T-DUMP and ST-DUMP ENGLISH sentences. Sample usage of the T-DUMP verb is illustrated in Figure B.

Files saved by a T-DUMP operation may be used with the ENGLISH tape connective. For more information, refer to the topic USING THROWAWAY CONNECTIVES in the ENGLISH Programming Manual.

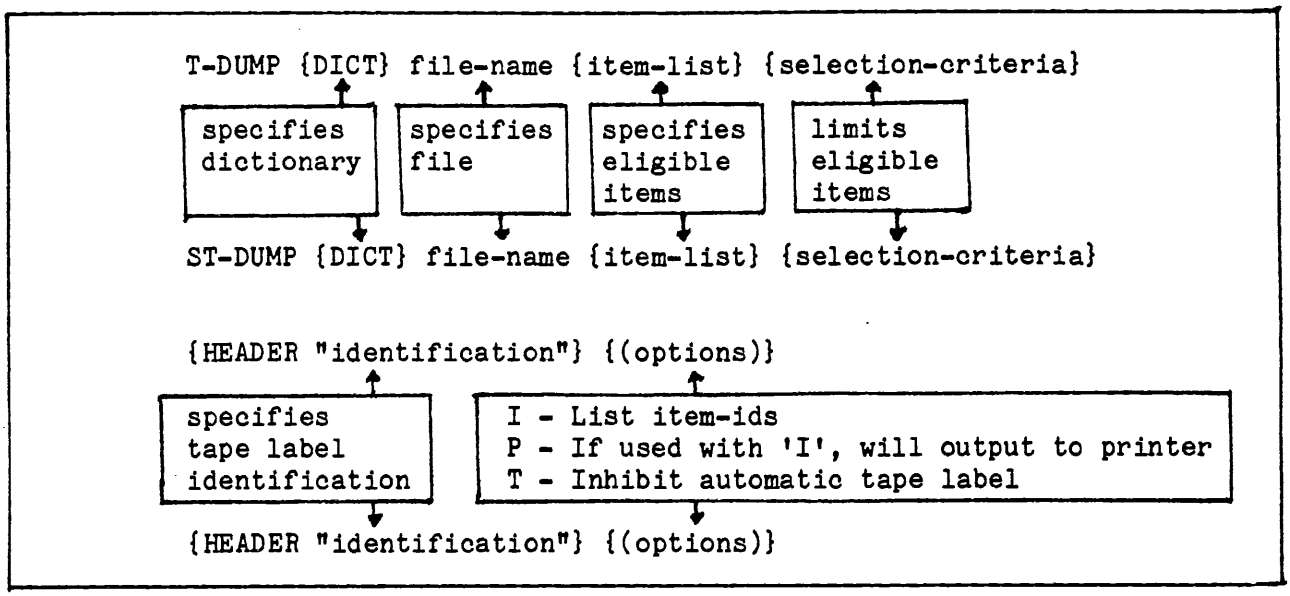


Figure A. General Form of T-DUMP and ST-DUMP ENGLISH Sentences

:T-DUMP TEST > '522' <cr>	Dumps to tape all items in the file TEST with item-ids greater than 522.
:T-DUMP DICT TEST <cr>	Dumps to tape all items in the dictionary of file TEST, except 'DL/ID'.

Figure B. Sample Usage of the T-DUMP Verb

9.9 Magnetic Tape I/O: T-LOAD Verb

The T-LOAD verb allows you to load files previously saved by a T-DUMP operation.

The T-LOAD verb works only with the default channel. The general form of an ENGLISH input sentence using the T-LOAD verb is:

T-LOAD {DICT} file-name {item-list} {selection-criteria} {(options)}

where:

"filename" is the name of the file to be loaded

"item-list" is a list of the item-ids to be loaded

"selection-criteria" are specified output conditions

"options" are:

O	Overwrites all items currently in the file that have item-ids corresponding to those on the tape
P	Lists item-ids on the printer as they are loaded
S	Suppresses the item-id listing

This verb writes to disc dictionaries or data files saved by a T-DUMP operation. Data from the tape is loaded to the file "file-name". The file-name may be preceded by the DICT modifier to load dictionary data. The "item-list" and "selection-criteria" (described in the ENGLISH Programming Manual) cause a selected subset of items in the specified file to be loaded. Note: The "item-list" may be supplied by a previous SELECT, SSELECT, GET-LIST or FORM-LIST verb, in which case the selection-criteria is omitted. Item-ids will be listed at the terminal as they are loaded, unless the 'P' option is used to specify the printer. The tape is positioned at the EOF mark at the conclusion of the operation.

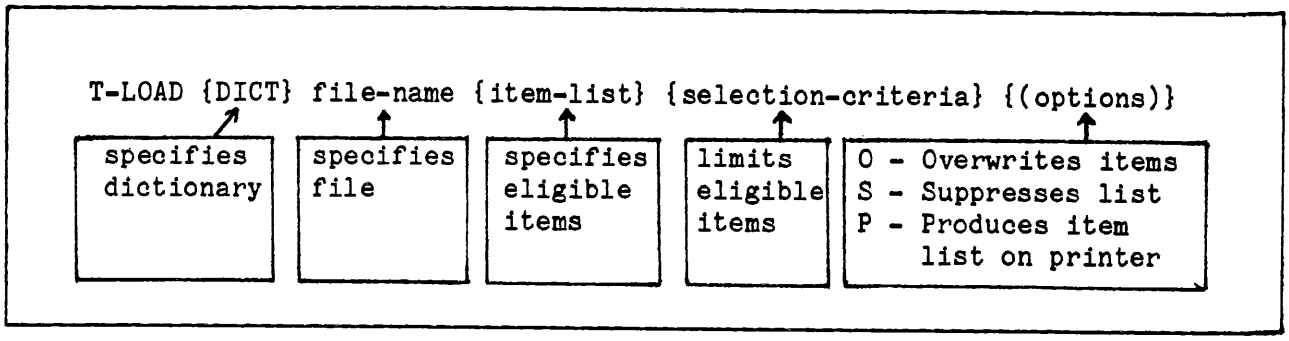


Figure A. General Form of T-LOAD ENGLISH Sentence

:T-LOAD TEST <cr>	Loads previously saved tape file to file TEST. Items will not be overwritten.
:T-LOAD DICT TEST (S,O) <cr>	Loads previously saved tape file to dictionary of file TEST. Listing of item-ids is suppressed. Items will be overwritten.

Figure B. Sample Usage of the T-LOAD Verb

9.10 Magnetic Tape I/O: MT-LOAD and MT-VERIFY Verbs

The MT-LOAD verb loads assembly code from magnetic tape. The MT-VERIFY verb verifies the assembly code. The T-DUMP verb must be used to write the assembly language routines ("modes") to tape prior to using MT-LOAD or MT-VERIFY.

The general form of the MT-LOAD verb is:

MT-LOAD {item-list}

where "item-list" specifies the item-ids to load.

This verb loads the item-ids specified in the "item-list" from tape to the ABS area. Items in the "item-list" are separated by blanks. The "item-list" may be supplied by a previous SELECT, SSELECT, GET-LIST or FORM-LIST verb. If the "item-list" is omitted, all item-ids are loaded.

The general form of the MT-VERIFY verb is:

MT-VERIFY {item-list} {(options)}

where:

"item-list" specifies the items-ids to verify

"options" are:

A	Causes a columnar listing of all mismatched bytes. Each byte in the source file which mismatches is listed, followed by the value in the ABS frame. If this option is not specified, only the first mismatched byte in each frame is printed
E	Inhibits printing of verification message lines. When this option is specified, only error messages are printed
P	Sends output to the printer

This verb checks the modes stored on tape against the ABS area. As each mode is verified, or if a mismatch occurs, a message is displayed. The modes which are to be verified are specified in the optional "item-list".

The MT-LOAD and MT-VERIFY routines do not position the tape. This must be done by the user. The routines work with items placed on the tape by the ENGLISH verb T-DUMP. Modes are retrieved from the tape until either an EOF is detected, or the "item-list" is exhausted. If processing has stopped because the "item-list" is exhausted, the tape probably will not be positioned at end-of-file.

Sample usage of the MT-LOAD and MT-VERIFY verbs is illustrated in Figure A.

:MT-LOAD <cr>	Loads all modes from the tape into executable frames.
:MT-LOAD TCL-I <cr>	Loads only the TCL-I mode from tape.
:MT-LOAD TCL-I EDIT-I DB6 <cr>	Loads the modes TCL-I, EDIT-I and DB6 from tape.
:MT-VERIFY <cr>	Verifies that the modes stored in virtual memory are identical to the modes stored on the tape.
:MT-VERIFY (E) <cr>	Verifies that the modes stored in virtual memory are identical to the modes stored on the tape, and suppresses verification message lines.
:MT-VERIFY SORT <cr>	Verifies only the mode SORT.

Figure A. Sample Usage of MT-LOAD and MT-VERIFY Verbs

9.11 Tape Labels

This section includes information on tape label size, placement and use.

Many system processors generate tape labels when writing to tape. These processors also check for a label when reading a tape. Processors that write and/or check for tape labels are DATA/BASIC, ENGLISH T-LOAD and T-DUMP, file-save, and file-restore.

A tape label is 50 characters long and is written at the beginning of the reel. When you generate a tape, you may specify identification information which will be included in the label. This information may be up to 16 characters long. If identification information is specified, the format of the label is:

(SM) L identification (VM) time date (AM) reel# (AM)
record size (AM) (SM) blanks (if necessary)

If the label is not 50 characters long, it will be padded at the end with blanks.

Label creation may be suppressed by using the 'T' option with any verb that causes tape output (T-DUMP, SAVE, RUN, etc.).

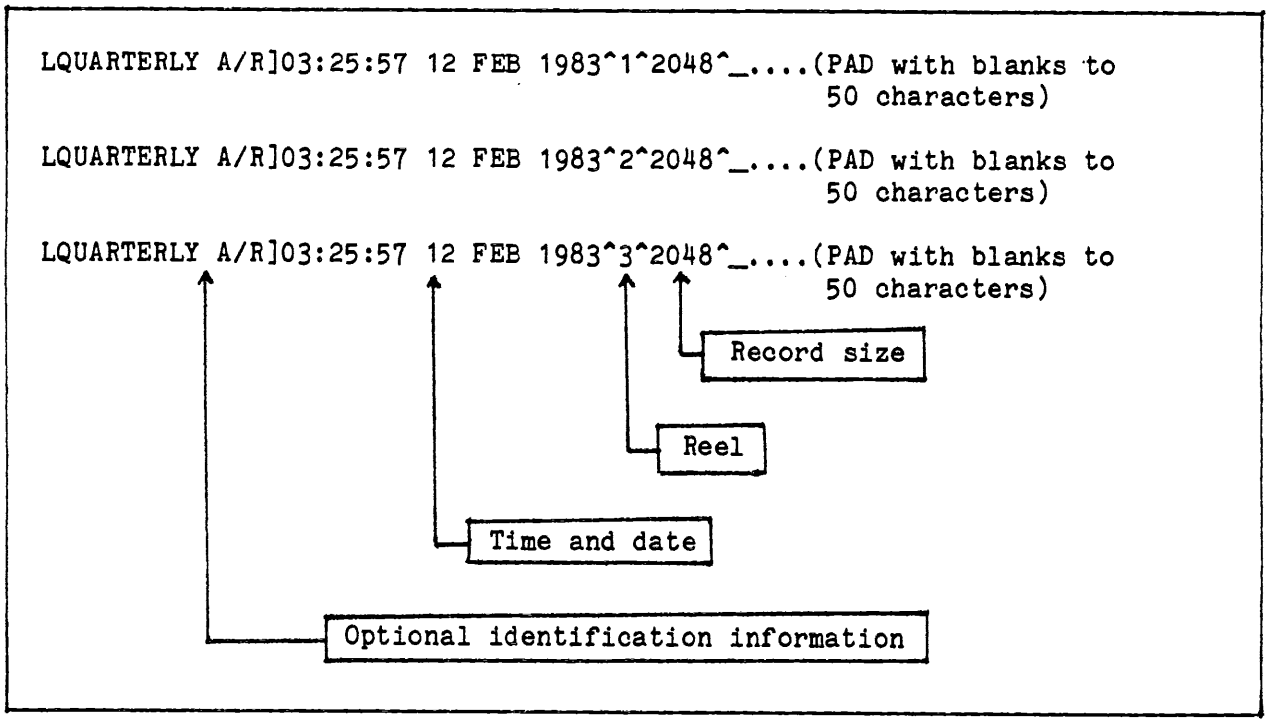


Figure A. Example of Labels for a Multireel Tape Operation

9.12 Reading Tape Labels: T-RDLBL Verb

Tape labels are generated by most tape writing operations and may be read with the T-RDLBL verb.

The general form of the T-RDLBL verb is:

T-RDLBL {channel} n

where:

"channel" is any alphabetic character from A to Z

"n" is the reel number

This verb reads, displays and stores the label from tape reel number "n" on the specified channel. Channel specification is optional. T-RDLBL automatically changes the record size if it is different from the size previously attached.

T-RDLBL must be used to initialize the internal label storage area when you are working with file-save or account-save tapes, or when beginning tape operations on a reel other than Reel #1.

You only need to use T-RDLBL once when working on a tape set, as the label data is stored during a logon session. An example is a SEL-RESTORE which begins with Reel #2 of a three reel file-save. When Reel #2 reaches end-of-tape (EOT), the system must be able to prompt for reel three. It is able to do so only if, prior to the SEL-RESTORE, Reel #2 was mounted at load point, and the verb "T-RDLBL 2" was used to initialize the label buffer.

NOTE

T-RDLBL must be used when performing an ACCOUNT-RESTORE, SEL-RESTORE or any other operation involving a save tape, even if it is a single reel operation.

:T-RDLBL 1 <cr> LABEL = DOC.ACCT]08:01:04 11 APR 1982 TAPE SIZE CHANGED TO 08190	Tape size changed to size specified in tape label.
:T-RDLBL B 1	Reads, displays and stores the label from the first reel assigned to channel B.
:T-RDLBL 3	Reads, displays and stores the label from the third reel.

Figure A. Sample Usage of the T-RDLBL Verb

9.13 Multiple Reel Tape Operations

This section describes multiple reel tape operations.

When a tape reel reaches the EOT marker, the tape system rewinds and sets the current drive offline. Tape operations continue with the next drive specified in the unit list. If this drive is offline, the prompt:

```
MOUNT REEL # n ON DRIVE m
TYPE 'C' TO CONTINUE
```

is displayed. When the next reel has been mounted, enter the character "C" to continue the operation. The system will ignore all entries other than "C".

NOTE

The prompt cannot be suppressed, and input to this prompt cannot be "stacked" by a PROC.

Figure A. lists the messages that may appear while reading labeled tapes. These messages will appear if a problem is encountered when reading the first record on any "next reel" operation.

REEL OUT OF SEQUENCE! PREVIOUS LABEL: |- -50 bytes- -| RECORD FOUND: |- -50 bytes- -| BYPASS LABEL CHECK? (Y/N) TYPE Y (YES), N (NO), Q (QUIT)	This message is displayed when the reel number on the labeled tape does not match the requested reel number, or if the first tape mounted is not reel number one.
WRONG LABEL! PREVIOUS LABEL: |- -50 bytes- -| RECORD FOUND: |- -50 bytes- -| BYPASS LABEL CHECK? (Y/N) TYPE Y (YES), N (NO), Q (QUIT)	This message is displayed if the tape label on a subsequent reel does not match the tape label on the first reel.
UNLABELED TAPE! PREVIOUS LABEL: |- -50 bytes- -| RECORD FOUND: |- -50 bytes- -| BYPASS LABEL CHECK? (Y/N) TYPE Y (YES), N (NO), Q (QUIT)	This message is displayed if an unlabeled tape is mounted when the first reel was labeled.

Figure A. Multiple Reel Tape Operation Messages

If you enter Yes (Y) to the 'BYPASS LABEL CHECK' question, the following prompt is displayed:

TYPE A (ACCEPT), S (SKIP), OR Q (QUIT)

Enter 'A' to accept the "label" as a data record and resume read operations.
Enter 'S' to skip the "label" and resume read operations with the next record.
Enter 'Q' to exit to TCL.

9.14 T-COPY Verb

The T-COPY verb allows you to perform a tape-to-tape copy.

The general form of the T-COPY verb is:

T-COPY channel1 channel2 {n} {(F)}

where:

"channel1" is the source channel name

"channel2" is the destination channel name

"n" is the number of files to copy

"(F)" causes consecutive save tape reels to be labeled

This verb allows you to perform a tape-to-tape copy. "Channel1" and "channel2" must be specified, and the channels must be assigned before using the T-COPY verb. If "n" is not specified, the system will copy all files up to two consecutive EOF's. If the tape does not contain two consecutive EOF'S, an unrecoverable parity error may appear in the first record following the actual data. You may terminate the copy at this point.

The tape copy is a logical copy, as there is no physical reel to reel correspondence between the original tape and the copy. The number of reels in the copy does not necessarily match the number of reels in the original.

If the original tape is labeled, the copy will also be labeled. Note that the time and date in the label of the copy will be the time and date the copy was made, not the time and date the original was made. If the original tape is not labeled, the copy will not be labeled.

WARNING

Binary save tapes can not be copied because the format is different than the format of a logical save.

:T-COPY A B	Copies all files from Channel A to Channel B.
:T-COPY A B 3	Copies 3 files from Channel A to Channel B.
:T-COPY A B (F)	Copies a FILESAVE or SYSGEN tape from Channel A to Channel B.

Figure A. Sample Usage of the T-COPY Verb

10.1 Arithmetic Verbs

This chapter discusses a set of special utility verbs executable at the TCL level. This initial topic describes the arithmetic verbs.

Figure A presents general forms of the arithmetic verbs. To terminate the verb format, press the RETURN key; the system will then return the arithmetic result.

Figure B shows sample usage of these verbs. Note that in the system output for the DIVD and DIVX verbs, a space separates the quotient from the remainder.

Also note that negative decimal numbers are specified by appending a minus sign immediately prior to the first digit of the number. Hexadecimal numbers are considered negative when in the range from X'FFFFFFFFFFFF' (-1) to X '800000000001' (-140,737,488,355,327).

ADDD n m	*Add decimal "n" to decimal "m"*
ADDX n m	*Add hex "n" to hex "m"*
SUBD n m	*Subtract decimal "m" from decimal "n"*
SUBX n m	*Subtract hex "m" from hex "n"*
MULD n m	*Multiply decimal "n" by decimal "m"*
MULX n m	*Multiply hex "n" by hex "m"*
DIVD n m	*Divide decimal "n" by decimal "m"*
DIVX n m	*Divide hex "n" by hex "m"*
DTX n	*Convert decimal "n" to hex*
XTD n	*Convert hex "n" to decimal*

Figure A. General Forms of Arithmetic Verbs

```
:ADDD 1 1 <cr>
    2

:ADDX FFFFFFFFFFFF 800000000005 <cr>
    800000000004

:DIVD 78645 987 <cr>
    79 672
```

Remainder (672)
Quotient (79)

```
:DIVD 1000000 25 <cr>
    40000 0

:DTX -2 <cr>
   FFFFFFFFFFFE

:ADDD -2 5000 <cr>
    4998

:MULX FFF EEF <cr>
    EEE111

:XTD ABC <cr>
   2748
```

Figure B. Sample Usage of Arithmetic Verbs

10.2 Setting Device Characteristics: TERM Verb

Terminal and/or printer characteristics may be displayed or set for the port you are logged on via the TERM verb.

TERM

This statement sets the terminal/printer characteristics for the port you are currently logged on. The general form of the TERM verb is:

TERM {a,b,c,d,e,f,g,h,i,j,k,l} {(m)}

where:

a - is terminal line length (i.e., number of characters per line). The 'a' parameter must be in the range $10 \leq a \leq 140$.
b - is the number of print lines per page on the terminal.
c - is the number of blank lines per page on the terminal (sum of 'b' and 'c' equals page length).
d - is the number of delay or idle characters following each carriage return or line feed. This is used for terminals that require a pause after a carriage return or line feed. This parameter should be set between 20 and 127 for MATRIX terminals running at 1200 baud.
e - is the number of delay characters following each top-of-form and form-feed control. If 'e' is two or greater, a form feed character is also output to the terminal before each page. If 'e' is zero, no form feed character is output to either the terminal or printer. If 'e' is one, a form-feed is output to the printer but not to the terminal.
f - is the backspace character. This parameter should be eight for all terminals. This defines the character which, upon receipt by the system, is treated as a backspace. When this character is entered, the system will delete the previous data character entered and echo the 'f' parameter followed by a space, followed by the 'f' parameter again. When this is the first character entered on a line, it is ignored.
g - is the printer line length. This parameter must be in the range $10 \leq g \leq 140$.
h - is the printer page length.
i - is the terminal type. If using DATA/BASIC cursor control and the MATRIX terminal, 'i' = 2. If using any other terminal, the 'i' parameter should be zero.
j - is the baud rate. The values supported are 19200, 9600, 7200, 4800, 3600, 2400, 2000, 1800, 1200, 600, 300, 150, 134, 110, 75 and 50.
k - is a PCI setting which defines character length, parity and number of stop bits. This parameter may be a decimal number between 0 and 255.
l - specifies the number of linefeeds to be appended to a formfeed (e.g., allows a WORDMATE header to be printed on a letter-quality printer).
m - is a numeric option to specify the port number.

If a PRISM is to run at or below 110 baud, two stop bits should be specified ('k' parameter). The byte is defined in the following hexadecimal format:

Bits (L to R)	Meaning
0 - 1	00 = invalid 01 = 1 stop bit 10 = 1.5 stop bits 11 = 2 stop bits
2	0 = odd parity 1 = even parity
3	0 = disable parity 1 = enable parity
4 - 5	00 = 5 bit character length 01 = 6 bit character length 10 = 7 bit character length 11 = 8 bit character length
6 - 7	always 10

A set of typical values are as follows:

Hex	Decimal	Meaning
4A	74	7 bit, no parity (5750)
4E	78	8 bit, no parity
5A	90	7 bit, odd parity (PRISM II)
5E	94	8 bit, odd parity
7A	122	7 bit, even parity
7E	126	8 bit, even parity

Individual parameters may be null (i.e., as specified by two adjacent commas in the TERM verb). If so, the previously defined parameter remains in force. If a parameter is specified that is out of range, an error message is output and that parameter retains its previously defined value. A TERM command without a parameter list displays the current characteristics.

If a numeric option is entered ('m' parameter), the display/changes apply to the given port. However, the verb only allows you to change another port's baud rate and PCI setting. Attempted changes to any other TERM parameters on another port are rejected. Because the DMP controls terminal I/O to Port Zero independently of the operating system, you cannot change Port Zero's baud rate or PCI setting. Instead, use Option 7 of the System Control Menu to change Port Zero's baud rate.

If a coldstart, ABS load or 'AF' restore is performed, the system reverts back to the default terminal characteristics specified by the hardware.

```
:TERM 79,24,1,5,5,8,132,60,0 <cr>      <-- Alter characteristics.

:TERM <cr>                             Display characteristics.
                TERMINAL    PRINTER
PAGE WIDTH:        79          132
PAGE DEPTH:        24           60
LINE SKIP :         1            0     <-- Number of linefeeds appended to form-
LF DELAY  :         5                      feed.
FF DELAY  :         5
BACKSPACE :         8
TERMINAL  :         0
BAUD RATE :
PCI       :

:TERM ,,,,,,85,56,,,2 <cr>             <-- Alter characteristics.

:TERM <cr>                             <-- Display characteristics.
                TERMINAL    PRINTER
PAGE WIDTH:        79           85
PAGE DEPTH:        24           56
LINE SKIP :         1            2     <-- Number of linefeeds appended to form-
LF DELAY  :         5                      feed.
FF DELAY  :         5
BACKSPACE :         8
TERMINAL  :         0
BAUD RATE :
PCI       :
```

Figure A. Sample Use of TERM Verb

10.3 Setting Device Characteristics: SET-TERM Verb

Terminal and/or printer default characteristics may be set for all ports via the SET-TERM verb.

The general format of the SET-TERM verb is:

SET-TERM {a,b,c,d,e,f,g,h,i}

where:

- a - is terminal line length (i.e., number of characters per line). The 'a' parameter must be in the range $10 \leq a \leq 140$.
- b - is the number of print lines per page on the terminal.
- c - is the number of blank lines per page on the terminal (sum of 'b' and 'c' equals page length).
- d - is the number of delay or idle characters following each carriage return or line feed. This is used for terminals that require a pause after a carriage return or line feed. This parameter should be set between 20 and 127 for MATRIX terminals running at 1200 baud.
- e - is the number of delay characters following each top-of-form and form-feed control. If 'e' is two or greater, a form feed character is also output to the terminal before each page. If 'e' is zero, no form feed character is output to either the terminal or printer. If 'e' is one, a form-feed is output to the printer but not to the terminal.
- f - is the backspace character. This parameter should be eight for all terminals. This defines the character which, upon receipt by the system, is treated as a backspace. When this character is entered, the system deletes the previous data character entered and echoes the 'f' parameter followed by a space, followed by the 'f' parameter again. When this is the first character entered on a line, it is ignored.
- g - is the printer line length. This parameter must be in the range $10 \leq g \leq 140$.
- h - is the printer page length.
- i - is the terminal type. If using DATA/BASIC cursor control and the MATRIX terminal, 'i' = 2. If using any other terminal, the 'i' parameter should be zero.

The SET-TERM verb sets the default printer and terminal characteristics except the baud rate and PCI setting for subsequent logons on all terminals. This verb is present only on the SYSPROG account.

If no parameter list is entered, the characteristics are displayed as for the TERM verb. The baud rate and the PCI setting for the port executing the verb will then be printed.

Figure A summarizes the general form of the SET-TERM verb.

```
:SET-TERM 79,24,1,2,2,21,80,60,0 <cr>      <-- Alter characteristics.

:SET-TERM <cr>                             <-- Display characteristics.
                TERMINAL    PRINTER
PAGE WIDTH:        79          80
PAGE DEPTH:        24          60
LINE SKIP :         1           0
LF DELAY  :         2
FF DELAY  :         2
BACKSPACE :        21
TERMINAL  :         0
BAUD RATE :
PCI       :


:SET-TERM ,,,,,,85,56 <cr>                 <-- Alter characteristics.

:SET-TERM <cr>                             <-- Display characteristics.

                TERMINAL    PRINTER
PAGE WIDTH:        79          85
PAGE DEPTH:        24          56
LINE SKIP :         1           0
LF DELAY  :         5
FF DELAY  :         5
BACKSPACE :         8
TERMINAL  :         0
BAUD RATE :
PCI       :
```

Figure A. Sample Use of TERM Verb

10.4 Setting Device Characteristics: TERMINAL PROC

The TERMINAL PROC allows you to build a parameter and options list for the TERM verb by printing a menu, soliciting values for each parameter, and editing the values entered.

To generate such a list, type in the following:

```
:TERMINAL
```

If a menu for another port is displayed, only changes made to the baud rate and PCI characteristics will be accepted. In addition, the PCI transmission characteristics cannot be changed without defining the baud rate first. While entering data, you may enter "HELP" to get assistance or "EX" to exit.

The following is a list of messages that may be generated when errors are detected by the verb TERM, SET-TERM or the TERMINAL PROC.

(1010) The parameters being changed are not compatible with the port option.

(1011) Baud rate and PCI transmission characteristics cannot be modified with the SET-TERM verb. Use "TERM" instead.

(1012) The PCI transmission characteristics cannot be changed without also defining the baud rate.

(1013) Port zero's baud rate or PCI transmission characteristics cannot be modified with the TERM verb. Use the DMP instead.

```
:TERMINAL <cr>

    WHICH PORT NUMBER (RETURN IF OWN PORT)? <cr>

              TERMINAL SETTING CONTROL MENU

                  1. TERMINAL PAGE WIDTH : 79 <cr>
                  2. TERMINAL PAGE DEPTH : 24 <cr>
                  3. LINE SKIP           :  1 <cr>
                  4. LINE FEED DELAY     :  5 <cr>
                  5. FORM FEED DELAY     :  5 <cr>
                  6. BACKSPACE CHARACTER :  8 <cr>
                  7. PRINTER PAGE WIDTH  : 85 <cr>
                  8. PRINTER PAGE DEPTH  : 56 <cr>
                  9. TERMINAL TYPE       :  0 <cr>
                 10. BAUD RATE           : <cr>
                 11. PCI SETTING         : <cr>
                 12. PRINTER LINE SKIP   :  2 <cr>

                 IS THIS CORRECT (FI/LINE NUMBER/EX)? FI <cr>

    :TERM <cr> <--------------------------- Display characteristics.
                   TERMINAL   PRINTER
    PAGE WIDTH:       79        85
    PAGE DEPTH:       24        56
    LINE SKIP :        1         2
    LF DELAY  :        5
    FF DELAY  :        5
    BACKSPACE :        8
    TERMINAL  :        0
    BAUD RATE :
    PCI       :
```

Figure A. Sample Use of TERMINAL PROC

10.5 Blocking Printing: BLOCK-PRINT and BLOCK-TERM Verbs

The BLOCK-PRINT or BLOCK-TERM verb prints characters in a 9-by-n block-form on the printer or your terminal, respectively.

The general form of the BLOCK-PRINT verb is:

BLOCK-PRINT character-string

This verb causes the specified character-string to be block-printed on the printer. Multiple character-strings separated by blanks are output on separate lines. Any character-string containing single quotes (') must be enclosed in double quotes ("), and vice versa. The surrounding quotes are not printed. A character-string not containing quotes as part of the string need not be surrounded by quotes, unless embedded blanks are to be preserved.

The general form of the BLOCK-TERM verb is:

BLOCK-TERM character-string

This verb is identical to the BLOCK-PRINT verb, except that the text string is output in block-print form to your terminal.

Character-strings must be such that the total number of characters does not exceed the current line length set by the most recent TERM verb.

If a BLOCK-PRINT or BLOCK-TERM verb is illegally formed, any of the error messages 520 through 525 may be displayed (refer to the list of error messages in the Appendix).

BLOCK-PRINT and BLOCK-TERM verbs use the BLOCK-CONVERT file to create the characters. This existing file contains conversion specifications for all printable ASCII characters (no lower case alphas, however). With this file, characters will be printed in 9-by-12 to 9-by-20 blocks.

If you wish to change the way any character is printed, you must change the corresponding item in the BLOCK-CONVERT file. The item-id of the item is the character to be converted. Each item in the file must consist of exactly 10 attributes. The first must specify in decimal the number of horizontal bytes in the blocked character to be output (i.e., "n" of the 9-by-n block mentioned above). The second and subsequent attributes provide the conversion specification. These attributes contain one or more values; each value is separated from the preceding by a value mark (<c>], X'FD'). The first character of the first value in each attribute must be "C" or "B"; these signal that the output matrix line of the blocked character begins with a character or a blank, respectively. Immediately following must be the number of characters or blanks (in decimal). The presence of a value mark indicates a switch from character to blank status (or vice versa) and must be followed by the number of bytes to be output. For example, an "X" might be specified as shown in Figure B.

```
:BLOCK-TERM AB12  <cr>


    AAAA          BBBBBBBBBBB          1111          2222222222
   AAAAAA         BBBBBBBBBBBB       111111         222222222222
  AAAAAAAA        BBBB     BBBB     1111111         2222    2222
 AAAA  AAAA       BBBBBBBBBBBB     1111111                 2222
AAAA    AAAA      BBBBBBBBBBB          1111          22222222222
AAAAAAAAAAAA      BBBB     BBBB        1111         222222222222
AAAAAAAAAAAA      BBBB     BBBB        1111         2222
AAAA    AAAA      BBBBBBBBBBBB    111111111111     222222222222
AAAA    AAAA      BBBBBBBBBBB     111111111111     222222222222


:BLOCK-PRINT AB12  <cr>

   Produces identical output as above, but on printer.


:BLOCK-PRINT "X'Y'Z"  <cr>

   Produces block-print of "X'Y'Z".
```

Figure A. Sample Usage of BLOCK-PRINT and BLOCK-TERM Verbs

```
Item 'X' in file BLOCK-CONVERT

001 12 <------- Blocked character is 12 bytes wide.
002 C4]4]4 <------- Output 4 characters, 4 blanks, 4 characters.
003 B1]4]2]4]1 <------- Output 1 blank, 4 characters, 2 blanks, 4 characters, 1
                        blank.
004 B2]8]2 <------- Output 2 blanks, 8 characters, 2 blanks.
005 B3]6]3 <------- Output 3 blanks, 6 characters, 3 blanks.
006 B4]4]4 <------- Output 4 blanks, 4 characters, 4 blanks.
007 B3]6]3 <------- Output 3 blanks, 6 characters, 3 blanks.
008 B2]8]2 <------- Output 2 blanks, 8 characters, 2 blanks.
009 B1]4]2]4]1 <------- Output 1 blank, 4 characters, 2 blanks, 4 characters, 1
                        blank.
010 C4]4]4 <------- Output 4 characters, 4 blanks, 4 characters.
```

Figure B. Sample Item in BLOCK-CONVERT File

10.6 Dumping to Terminal: I-DUMP and S-DUMP Verbs

I-DUMP and S-DUMP are ENGLISH verbs that allow you to selectively dump dictionaries and data files to the terminal.

I-DUMP

The I-DUMP verb (Figure A) has the following general form:

I-DUMP {DICT} file-name {item-list*} {selection-criteria} {(options)}

The "item-list" and "selection-criteria" (described in the ENGLISH Programming Manual) cause a selected subset of items in the specified file to be dumped to the terminal. If omitted, all items are dumped. The DICT modifier causes dictionary data to be dumped. File definition items (D/CODE=D) are not dumped. S-DUMP performs a sorted I-DUMP.

Valid options for these two verbs are:

N - Inhibit automatic paging of output to terminal

P - Direct output to printer

* The item-list can be supplied by a previous SELECT, SSELECT, GET-LIST or FORM-LIST command, in which case the selection-criteria is omitted.

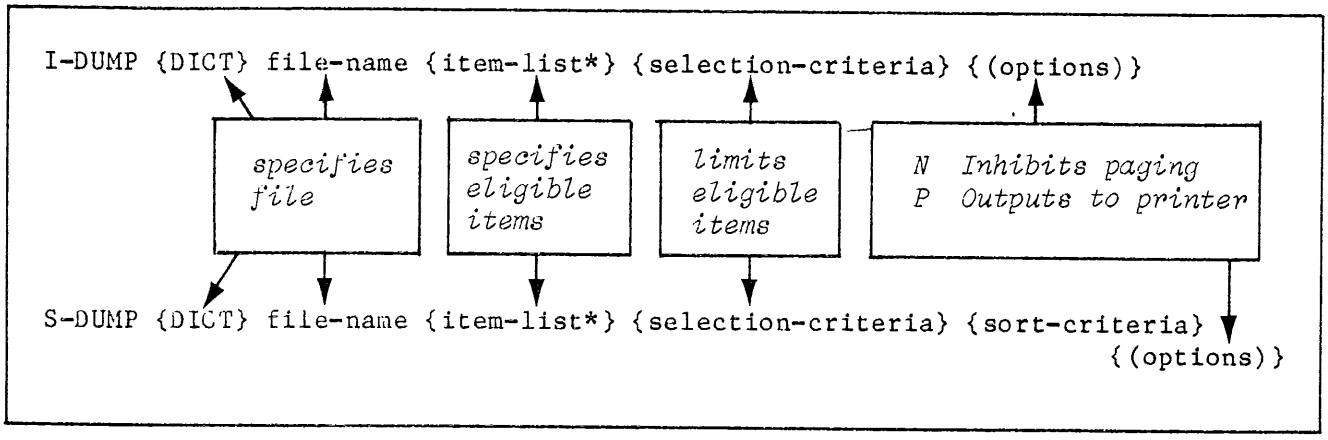


Figure A. General Form of I-DUMP and S-DUMP Verbs

```
:I-DUMP TEST '14' '15' '16' <cr>

14^THIS^IS^ITEM^14^111]222]333^AAA]BBB]CCC]DDD^1234567890^
15^THIS^IS^ITEM^15^ABCDEFGHIJK^.].].].].].]
16^THIS^IS^ITEM^16^1234]5678]9012]3456]7890^XXXXXX^
3 ITEMS DUMPED.
```

Figure B. Example of I-DUMP Verb

10.7 DUMP Verb

The DUMP verb allows you to examine the contents of virtual memory frames in character or hexadecimal format. Use of this verb requires a system privilege level two.

The general form of the DUMP verb is:

```
DUMP fid1{-fid2} {(options)}
```

where "fid1" is the beginning FID and "fid2" is the ending FID. If "fid1" is specified alone, only that frame is dumped; "fid1" and "fid2" may be specified as hexadecimal by preceding them with a period. If "fid2" is hexadecimal, it should not be preceded by the hyphen (-).

Valid options are listed below. Multiple options are separated by commas.

- G Dumps group data - all linked frames beginning with "fid1" are displayed; "fid2", if present, is ignored
- L Like 'G', but dumps linked fields of frame only (no data dump)
- N Suppresses pagination of terminal display
- P Dumps to the printer instead of the terminal
- U When used with the 'G' or 'L' option, causes dump to follow backward links instead of default forward links; if 'U' is used without 'G' or 'L' option, the 'G' option is assumed.
- X Displays data in hexadecimal as well as character format

Link information is displayed in decimal and hexadecimal. A "+" next to a frame indicates a frame found by tracking forward links. A "-" indicates a link found by tracing backward links ('U' option). Figure B shows examples of the DUMP verb.

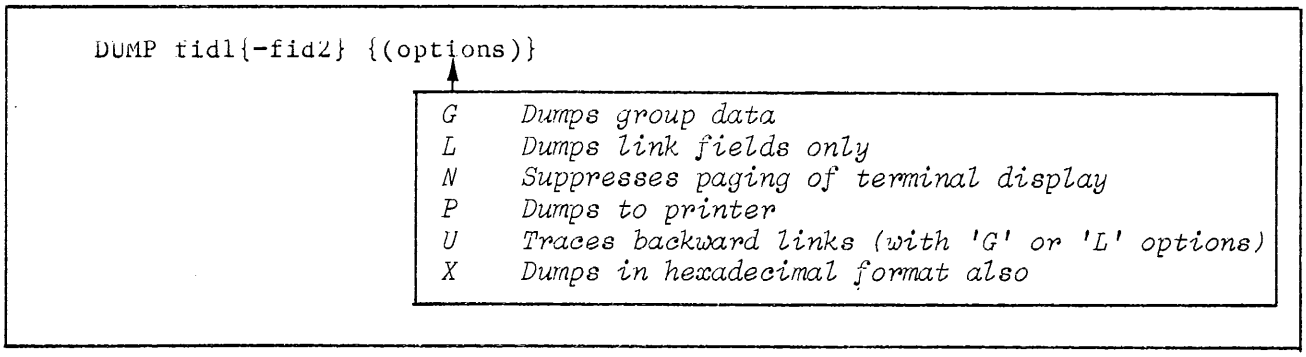


Figure A. General Form of DUMP Verb

```
:DUMP 5400 <cr>

            # of                                         # of
            Next                                       Previous
FID:     Contiguous    Forward    Backward          Contiguous           Hex
           Frame        Link        Link               Frame         Equivalents

FID:    5400 :     0    10532         0         0   (   1518 :   0    2924      0   0  )

000 ....)$......0015CHARGE-TO^P^6032^0017C-READ^P^2^94
050 ^^CUN^001AZERO-DISC-ERRS^P^20CF^001B*A11^A^11^^^^^
100 ^^L^10^1^001CCT^PQ^(DICT PROCLIB CT)^0018T-WEOF^PZ
150 ^6033^B023^002FBLOCK-CONVERT^Q^BLOCK-CONVERT^^^^^^
200 ^L^10^1^002ALISTDICTS^PQ^(DICT PROCLIB LISTDICTS)^
250 0016SAVE-LIST^PL^2064^0013SP-STOP^P^10A6^001D*A3^A
300 ^03^^^^^^T10^L^10^1^001FJIM^Q^SYSTEM^JIM^^^^^^L^10
350 ^0014MT-VERIFY^PV^D1^0027PF^Q^SYSTEM^POINTER-FILE^
400 ^^^^^R^10^0092BRANCH^ FRAME 400] FRM: 400 001 7FF0
450 0190  ORG 1 001 ^*^*09MAR77^*^*^*^*^*^*^ BSL LABX
500 001 19FD^ OR
```

```
:DUMP .2EA.2EC (L,U) <cr>
```

Dump frames X'2EA' through X'2EC'; dump links only; trace backward links.

```
 FID:   746 :   5   747      0    0    (   2EA : 5  2EB      0  0  )

 FID:   747 :   4   748    746    1    (   2EB : 4  2EC    2EA  1  )
-FID:   746 :   5   747      0    0    (   2EA : 5  2EB      0  0  )

 FID:   748 :   3   749    747    2    (   2EC : 3  2ED    2EB  2  )
-FID:   747 :   4   748    746    1    (   2EB : 4  2EC    2EA  1  )
-FID:   746 :   5   747      0    0    (   2EA : 5  2EB      0  0  )
```

```
:DUMP 5400 (X) <cr>
```

Dump frame 5400 in hex and character format.

```
FID:    5400 :   0 10532    0    0  (  1518 :   0   2924      0  0  )

000  00000000 29240000 00000000 30303135   000  :....)$......0015:
010  43484152 47452D54 4FFE50FE 36303332   015  :CHARGE-TO^P^6032:
 .       .        .        .        .       .         .
 .       .        .        .        .       .         .
 .       .        .        .        .       .         .
1E0  2AFE2AFE 2AFE2AFE 2AFE2042 534C204C   480  :*^*^*^*^*^ BSL L:
1F0  414258FC 30303120 31394644 FE204F52   496  :ABX 001 19FD^ OR:
```

Figure B. Examples of DUMP Verb

10.8 FID Information: ITEM and GROUP Verbs

ITEM and GROUP verbs provide information about the item and group structure of SEQUEL files. Output can be displayed at the terminal or optionally directed to the printer.

ITEM

The ITEM verb has the following general form:

```
ITEM file-name item-id {(options)}
```

This verb displays the base FID of the group into which the specified item-id hashes. If the item is not on file, the message "ITEM NOT FOUND" is displayed. In addition, every item-id in that group is listed in the order in which it occurs in the group along with a character count of the item (in hex). At the end of the list the following message is displayed:

```
n ITEMS m BYTES p/q FRAMES
```

where:

n is the number of items in the group
m is the total number of bytes used in the group
p is the number of full frames in the group
q is the number of bytes used in the last frame of the group

Valid options for this verb are:

N - Defeat automatic paging of terminal output
P - Direct output to printer
S - Suppress item list

GROUP

The GROUP verb has this general form:

```
GROUP file-name {(options)}
```

This verb is similar to the ITEM verb, but GROUP displays the base of FID of every group in the specified file. In addition, every item-id in each group is listed along with a character count of the item (in hex). The end of list message for each group and valid options are the same as for the ITEM verb.

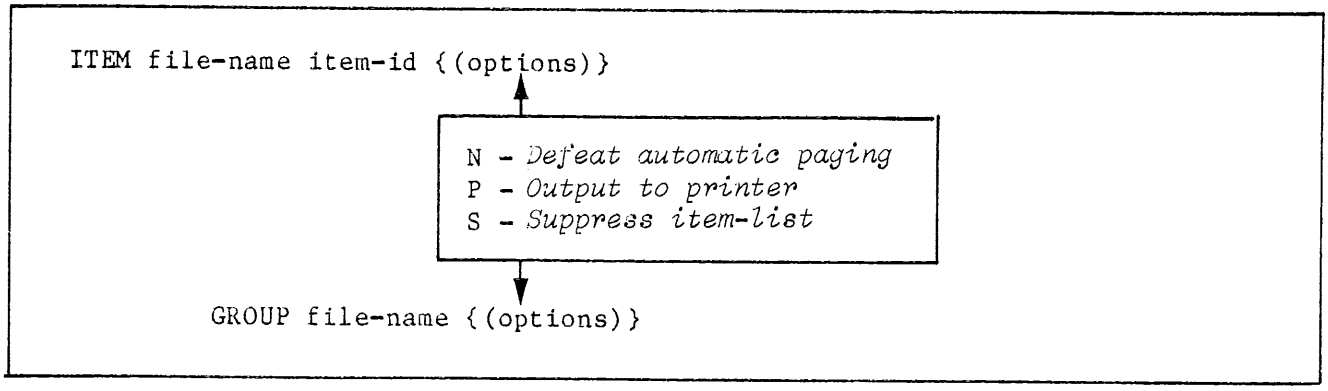


Figure A. General Form of ITEM and GROUP Verbs

```
:ITEM TEST 14  <cr>

1895 <------------------ Beginning FID of group in which item
0007 10                  '14' resides. Order of items is '10',
0007 11                  '11', and '14'; each is seven bytes.
0007 14
3 ITEMS 21 BYTES 0/21 FRAMES


:ITEM TEST 14 (S)  <cr>
1895
3 ITEMS 21 BYTES 0/21 FRAMES


:GROUP TEST  <cr>
1895 <------------------ Beginning FID of first group.
0007 10
0007 11
0007 14
3 ITEMS 21 BYTES 0/21 FRAMES
1896 <------------------ Beginning FID of second group.
0009 12
000A 33
0008 28
0010 92
4 ITEMS 43 BYTES 0/43 FRAMES
1897 <------------------ Beginning FID of third group.
001F 88
1 ITEMS 31 BYTES 0/31 FRAMES
```

Figure B. Sample Usage of ITEM and GROUP Verbs

10.9 Hashing Information: ISTAT and HASH-TEST Verbs

ISTAT and HASH-TEST are ENGLISH verbs that produce file hashing statistics and an optional histogram; ISTAT for specified file items and HASH-TEST on the basis of a user-specified test modulo.

ISTAT

An ENGLISH sentence using the ISTAT verb is illustrated in Figure A. The ISTAT verb provides file hashing statistics and, if desired, an optional histogram for selected items in the selected file, shown by examples in Figure B. For further information regarding item hashing, refer to title ITEM STORAGE AND THE HASHING ALGORITHM in Chapter 2.

HASH-TEST

HASH-TEST produces the same output as ISTAT, but uses a user-specified test modulo. The general form of this verb is shown in Figure A. Figure C illustrates a HASH-TEST dialog and printout.

Options for these two verbs are:

H - Prints histogram
N - Defeats automatic paging of output to terminal
P - Directs output to printer

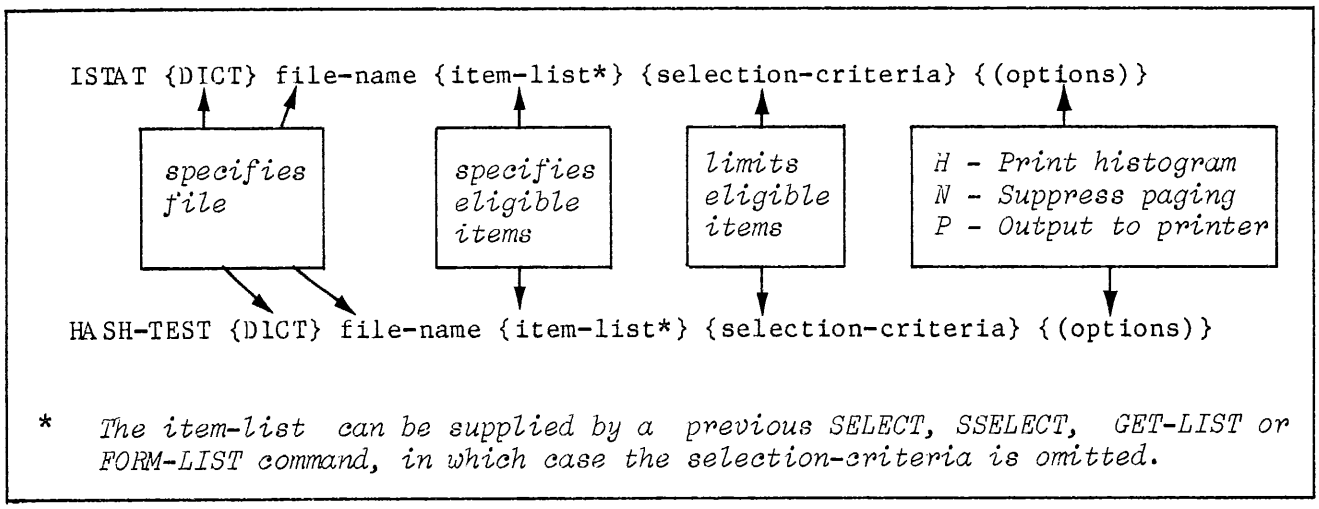


Figure A. General Form of ISTAT and HASH-TEST Verbs

```
:ISTAT ACCOUNT (H) <cr>

FILE= ACCOUNT MODULO= 3 SEPAR= 1                14:30:14  12 FEB 1983
BYTES ITMS
05379 023 *>>>>>>>>>>>>>>>>>>>>>>
05391 022 *>>>>>>>>>>>>>>>>>>>>>>
05813 023 *>>>>>>>>>>>>>>>>>>>>>>

ITEM COUNT = 68, BYTE COUNT = 16583, AVG. BYTES/ITEM = 243.8
AVG. ITEMS/GROUP = 22.6, STD DEV = .5, AVG. BYTES/GROUP = 5527.6


:ISTAT ACCOUNT > '23020' AND < '35090' (H) <cr>

FILE= ACCOUNT MODULO= 3 SEPAR= 1                14:31:15  12 FEB 1983
BYTES ITMS
03093 012*>>>>>>>>>>>>
02751 011*>>>>>>>>>>>
03001 011*>>>>>>>>>>>

ITEM COUNT = 34, BYTE COUNT = 8845, AVG. BYTES/ITEM = 260.1
AVG. ITEMS/GROUP = 11.3, STD DEV = .5, AVG. BYTES/GROUP = 2948.3
```

Figure B. Sample Usage of ISTAT Verb

```
:HASH-TEST ACCOUNT (H) <cr>

TEST MODULO: 7  <cr>
FILE= ACCOUNT MODULO=7 SEPAR= 1                 14:35:22  12 FEB 1983
BYTES ITMS
02413 010*>>>>>>>>>>
02710 011*>>>>>>>>>>>
02121 009*>>>>>>>>>
02598 011*>>>>>>>>>>>
01619 007*>>>>>>>
02462 010*>>>>>>>>>>
02660 010*>>>>>>>>>>

ITEM COUNT = 68 BYTE COUNT = 16583, AVG. BYTES/ITEM = 243.8
AVG. ITEMS/GROUP = 9.7, STD DEV = 1.3, AVG. BYTES/GROUP = 2369
```

Figure C. Sample Usage of HASH-TEST Verb

10.10 Information: MSG, WHO, and WHAT Verbs

The MSG, WHO, and WHAT verbs provide miscellaneous system information and capabilities.

MESSAGE or MSG

All system users may communicate with other users on the system. To transmit a message to another user, type the verb MESSAGE (or MSG) followed by the user's account name or '!' followed by a destination port#, then the text of the message. The general form is:

```
MESSAGE account-name text-string
        --OR--
MESSAGE !n text-string
```

where "n" is the destination port#. For example:

```
:MESSAGE USER-2 HELLO THERE <cr>
```

The maximum message is 108 characters long. Anyone currently active on the account referenced will receive the message, in the form:

```
time date FROM sender:
message
```

The above message might be received as:

```
12:41:37  22 FEB 1984  FROM USER-1:
HELLO THERE
```

If the specified user is not presently logged on, the system responds with "USER NOT LOGGED ON". If your account has system privilege level two, you may broadcast to all users by substituting an asterisk (*) for the account name. For example:

```
:MSG * GOOD MORNING <cr>
```

If you are entering data at a terminal and receive a message, part of the data in the input buffer may be destroyed. To display what is remaining, type in <c>R.

WHO

The WHO verb returns the line number and account-name to which you are logged on. The general form is:

```
WHO {n}
```

where "n" specifies a line number. If specified, line "n" and the name of the

account logged on to line "n" are displayed. The account name is obtained by looking up the PCB FID in the ACC file and returning Attribute 1. If the account file item is not found, "UNKNOWN" is returned as the account name.

WHAT

The WHAT verb causes the current system configuration parameters to be displayed. The general form is:

```
WHAT
```

Figure A illustrates the use of the MSG, WHO and WHAT verbs.

```
:MSG JOE ARE YOU FINISHED WITH THE TAPE? <cr>

:WHO <cr>                                    <-- You are on line 22 and logged
22 SYSPROG                                       on to SYSPROG account.

:WHO 3 <cr>                                  <-- User on line 3 is logged on
3 GARY                                           to account GARY.

:WHAT <cr>
2048K    OF MEMORY
1215     ABS FRAMES
17       ACTIVE LINES
16       IS THE LINE TO WHICH THE SPOOLER IS ASSIGNED
1216     IS THE FRAME ID OF THE PCB FOR LINE ZERO
7900     IS THE FRAME ID OF THE BASE OF THE SYSTEM DICTIONARY
998143   IS THE MAXIMUM FRAME ID FOR THIS SYSTEM

DISC DRIVE CONFIGURATION:
     DEVICE ADDRESSES
          D0
          D1
          D2
          D3
```

Figure A. Examples of the MSG, WHO and WHAT Verbs

10.11 Information: TIME, SET-TIME, SET-DATE, and DATE-FORMAT Verbs

The TIME, SET-TIME, SET-DATE, and DATE-FORMAT verbs provide miscellaneous system information and capabilities. Figure A shows examples of these verbs.

TIME

The system time and date can be displayed on the terminal by entering the TIME verb.

SET-TIME

The SET-TIME verb changes the current system time. The general form is:

```
SET-TIME hh:mm{:ss}
```

where "hh:mm" (seconds optional) is the time in 24-hour format.

SET-DATE

The SET-DATE verb changes the current system date. The general form is:

```
SET-DATE ddmmm{yy} or mm/dd{/yy}
```

where "dd" is the numeric day; "mmm" is the three-letter month abbreviation (or two-digit month number in the second form); "yy" is the optional year specification. If not used, the current year displayed by the system is used.

DATE-FORMAT

The DATE-FORMAT verb changes the form of dates printed with ENGLISH date conversions. The general form is:

```
DATE-FORMAT {(I)}
```

If the 'I' option is used, all dates normally displayed as mm/dd/yy are printed in European (international) format as dd/mm/yy, and the system assumes that external dates entered in formats such as mm/dd/yy are to be interpreted as dd/mm/yy. The date format may be reset by entering DATE-FORMAT (without an option).

```
:TIME <cr>
09:42:37  12 FEB 1983


:SET-TIME 14:23 <cr>
14:23:00  12 FEB 1983


:SET-DATE 13 FEB 1983 <cr>
14:23:11  13 FEB 1983


:DATE-FORMAT (I) <cr>


:DATE-FORMAT <cr>
```

Figure A. Examples of TIME, SET-TIME, SET-DATE and DATE-FORMAT Verbs

10.12 Information: WHERE and SLEEP Verbs

The WHERE and SLEEP verbs provide miscellaneous system information and capabilities. Figure A shows examples of these verbs.

WHERE

The WHERE verb causes currently logged-on user assignments to be displayed. The general form is:

```
WHERE {n} {(options)}
```

Columns in this display are line number, PCB FID, PIB Status byte one, followed by register one and the software return stack. An asterisk (*) next to an entry indicates assignments for the line which issued the verb.

If "n" is specified, user assignments for line "n" are displayed. The 'S' option displays the number of process activations, CPU milliseconds, number of disc reads, PIB Status byte one and LINKS. The 'A' option displays all lines including those not currently logged on; a question mark (?) is printed next to ports that are logged off but do not have a status '78' (an error condition). The 'N' option inhibits automatic paging of output to the terminal.

SLEEP

The SLEEP verb sets a process to "sleep" for a specified period of time. The mechanism that allows a process to "sleep" is implemented in the MONITOR so that a "sleeping" process creates practically no overhead in the system. The "sleep" mechanism is useful for processes that must remain quiescent for a period of time. The form of the SLEEP verb is:

```
SLEEP n
or
SLEEP hh:mm{:ss}
```

where "n" is the number of seconds the process should sleep, or where "hh:mm" (seconds optional) is the time (in 24-hour format) when the process is to regain control. In either case, the terminal's <BREAK> key may be used to wake the terminal.

```
:WHERE <cr>

  PORT     PS  RTN STACK..
*00 0200   FC  121.0C8 121.072
 03 0260   3C  164.070 164.066


:WHERE (A,S) <cr>

  PORTS        # ACTS     CPU MSEC  READS PS   LINKS
*00 0200       309197      320416    202 FC 00 FF 08 FF
?01 0220          234         239      0 78 00 FE FE FF
?02 0240            0        2440      0 78 00 FE FE FF
 03 0260        49030       50979     15 3C 00 00 FF FF
?04 0280            0   808728608      0 78 00 FE FE FF
?05 02A0           22          28      0 75 00 FE FE FF


:WHERE 3 <cr>

  PORT     PS  RTN STACK..
 03 0260   3C  146.070 164.066


:SLEEP 10 <cr>

:SLEEP 11:32 <cr>
```

Figure A. Examples of WHERE and SLEEP Verbs

10.13 Information: LIMITS, USER, and DISCIO Verbs

The LIMITS, USER, and DISCIO verbs provide miscellaneous system information and capabilities.

LIMITS

The LIMITS verb displays hardware configuration limits of the system. The general form is:

```
LIMITS {(P)}
```

This verb displays the system serial number, memory limit, ABS frame limit, communication ports limit, WORDMATE configuration ('Y' for yes, 'N' for no), available devices (such as printer, tape, discs), and accessible frame numbers. The 'P' option directs output to the printer. This verb may be installed on an account with a SYS2 privilege level.

USER

The USER verb displays the system privilege level and number of logon workspace frames reserved for the line which executed this verb. The general form is:

```
USER
```

Note: The actual number of additional frames is the number displayed minus six frames.

DISCIO

The DISCIO verb creates a 5-second "snapshot" disc activity report. Information displayed includes:

1. Disc I/Os per second for each unit
2. Total disc I/Os per second
3. Number of disc reads per second
4. Number of disc write/verify operations per second
5. Cumulative disc I/Os per unit
6. Cumulative disc I/Os
7. Cumulative disc reads
8. Cumulative disc write/verify operations

where "cumulative" means the total number processed since the start of DISCIO execution. The general form is:

```
DISCIO {(options)}
```

If a numeric option is used, the display is repeated the specified number of times. The 'T' option displays the current total disc I/O counters for each unit (i.e., the number of disc I/O operations since the last coldstart). The numeric option cannot be used with the 'T' option. The 'P' option produces the report on the printer.

Figure A shows sample usage of above verbs.

```
:LIMITS <cr>
SEQUEL

1005        SYSTEM SERIAL NUMBER
8192K       OF MEMORY
4194304     IS MAXIMUM FRAME-ID
128         PORTS
1           IS FIRMWARE TYPE
3           IS FIRMWARE VERSION NUMBER
Y           WORDMATE
                AVAILABLE DEVICES
                DEVICE        UNIT(S)
                TAPE(S)       1,2,3,4
                PRINTER(S)    1,2,3,4


:USER <cr>
2    SYSTEM PRIVILEGE LEVEL
50   ADDITIONAL FRAMES OF WORK SPACE


:DISCIO <cr>

              DEVICE     I/O'S PER SECOND     TOTAL I/O'S
                D0              6.17              32
                D1              7.13              37
                D2              7.22              40
                D3              5.55              28
                TOTAL :        26.1              137

                READS :        16.1              77
                WRITES :       10.0              60


:DISCIO (T) <cr>

            DEVICE    DISC UNIT I/O'S
              D0           351762
              D1           327797
              D2           342781
              D3           338765
```

Figure A. Sample Use of LIMITS, USER, and DISCIO Verbs

10.14 Setting Process Timeslices: TIMESLICE Verb

This section describes how the SEQUEL system allocates timeslices among the various terminals. The TIMESLICE verb allows each line to be assigned a different timeslice to allow adjustment of task priorities. The SET-PRIORITY verb, previously used to adjust process priorities, no longer functions and if executed, it will be ignored.

Each process (terminal) not waiting for an operator entry is placed on a queue of terminals waiting for activation. Activation means that the process is given a timeslice (or quantum) of CPU time in which to do its work.

To determine which job to activate, the system scans from the top of the queue until it finds a job which can be activated. A terminal cannot be activated if:

1. It is waiting for a frame to come in from disc
2. It is "sleeping"

A process which is activated continues to process until:

1. It uses up its timeslice
2. It references a frame which is not in main memory
3. The program voluntarily deactivates
4. The operator is prompted
5. It references a frame to which I/O is being performed by a controller

Priority is determined by a process's position on the queue or "ladder". The position of processes changes as they use various system resources such as disc accesses and CPU processing.

Normally, each process is assigned a 10 millisecond timeslice. The timeslice for each line is kept in main memory and current settings for each line stay in effect until a coldstart is performed, at which point the settings for all lines are set to 10 milliseconds.

The form of the TIMESLICE verb is:

```
TIMESLICE {n} {(options)}
```

where "n" is a number (1-1023) indicating the timeslice in milliseconds. If "n" is omitted, the timeslice is displayed but remains unchanged.

The timeslice for other lines may be changed or displayed from another terminal by specifying the line number in the option. For example, TIMESLICE 50 (3) sets the timeslice to 50 milliseconds on line 3, whereas TIMESLICE (3) displays the current timeslice setting for line 3.

All of the process timeslices may be set or displayed by using the 'A' option.

A point to remember in setting process timeslices is that setting a large timeslice on one or more lines may cause other processes to degrade because they may have to wait longer for processing time. Setting timeslices too low may also cause degradation because processes may get "bumped off" before they get to do any meaningful work.

You are encouraged to experiment with different settings for your job mix, possibly changing timeslices within PROCs that activate certain jobs.

```
TIMESLICE {n} {(options)}
```

Figure A. General Form of TIMESLICE Verb

```
:TIMESLICE <cr>  <-------- Displays current timeslice.
TIMESLICE FOR LINE 10 EQUALS 100 MILLISECONDS

:TIMESLICE 25 <cr>  <-------- Sets timeslice to 25 milliseconds.

:TIMESLICE (3) <cr>  <-------- Displays timeslice for line 3.
TIMESLICE FOR LINE 3 EQUALS 75 MILLISECONDS

:TIMESLICE 50 (3) <cr>  <-------- Sets timeslice for line 3 at 50 milliseconds.

:TIMESLICE 75 (A) <cr>  <-------- Sets timeslice for all lines at 75
                                  milliseconds.
```

Figure B. Sample Usage of TIMESLICE Verb

10.15 Available System Space: POVF and CLAIM Verbs

The POVF verb displays the number of frames in the four "available space" tables. The CLAIM verb returns linked available space to contiguous available space.

CLAIM

The CLAIM verb converts linked space to contiguous space. CLAIM can run while other processes are running. The general form (Figure A) is:

```
CLAIM
```

CLAIM sorts the FIDs of the linked frames and determines if there are any consecutive frames that could now make up a contiguous block. If so, the block is placed in the contiguous space table (attached to an existing block if possible). Then the individual frames are inspected to see if they can now be attached to existing contiguous blocks. The remaining frames (if any) remain in linked form and the POVF display is printed.

POVF

The POVF verb (Figure A) has the following general form:

```
POVF {(P)}
```

POVF displays the number of frames in the four "available space" tables. If there are any frames in the linked chain of file space, the first line of output contains the FID of the beginning frame in the chain. The "TO" field contains the word "LINKED" and the "#FRAMES" field contains the total number of frames in the linked chain. Contiguous file space is displayed with the beginning and ending FIDs of each block and the total frames in the block. Lines of output after the linked chain information are in the form:

```
m - n  p     m - n  p
```

where:

"m" is the first frame of a contiguous block
"n" is the last frame of the block
"p" is the number of frames in the block

Following the contiguous frame display is the total number of contiguous file space frames. Also listed are the number of frames in the temporary space block and the temporary space chain.

The 'P' option directs output to the printer.

Figure B illustrates usage of POVF and CLAIM verbs.

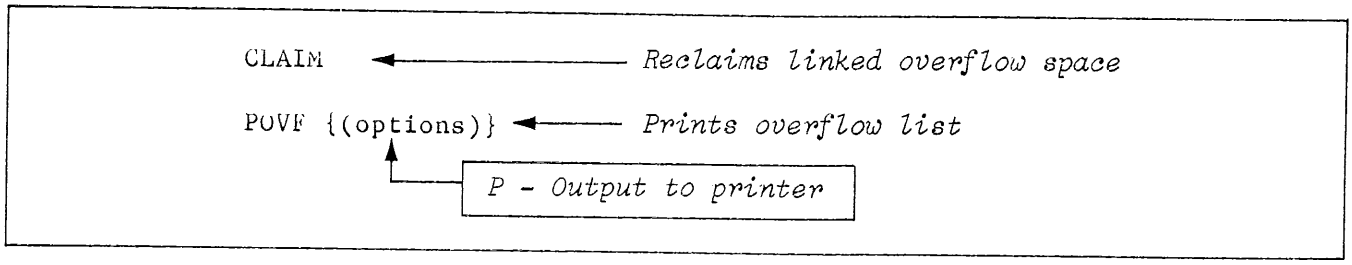


Figure A. General Form of POVF and CLAIM Verbs

```
:POVF <cr>

 FROM    TO              #FRAMES          FROM      TO          #FRAMES

54872  LINKED              5
 5622 -  5711              90           6427 -  6440             14
25805 - 25806               2          35700 - 35707              8
35712 - 35714               3          35718 - 35725              8
36566 - 36595              30          36680 - 36694             15
36899 - 36916              18          36931 - 36939              9
36959 - 36965               7          37736 - 37737              2
33740 - 37751              12          38557 - 38570             14
38584 - 38596              13          39290 - 39291              2
53936 - 53993              58          54000 - 54007              8
54016 - 54026              11          54033 - 54042             10
54049 - 54228             180          54373 - 54384             12
54413 - 54793             381          54841 - 54853             13
54914 - 54922               9          54924 - 54927              4
54937 - 54937               1          54960 - 54960              1
54969 - 54978              10          55038 - 55045              8
55399 - 55578             180          55957 - 58463           2507

TOTAL NUMBER OF CONTIGUOUS FRAMES = 3640
NUMBER OF FRAMES IN THE TEMPORARY SPACE BLOCK =  0
NUMBER OF FRAMES IN THE TEMPORARY SPACE CHAIN = 31

:CLAIM <cr>  <-- Produces same output as POVF
```

Figure B. Sample Usage of POVF and CLAIM Verbs

10.16 Generating Checksums: CHECK-SUM Command

The CHECK-SUM command generates a checksum for file items; from this you can determine if data in a file has been changed.

CHECK-SUM is an ENGLISH verb (Figure A) which has the following general form:

```
CHECK-SUM {DICT} file-name {item-list*} {attribute} {selection-criteria}
                                                          {(options)}
```

A checksum is an arithmetic total, disregarding overflow, of all bytes in the selected item(s) and is generated for items in the specified file, or subset of items if the optional "item-list" and/or "selection-criteria" appear. Furthermore, the checksum may be calculated for one specified attribute. If no attribute is specified, the first default attribute is used. If there is no default attribute, or if the AMC is 9999, the entire item is included. The checksum includes the binary value of each character times a positional value. This yields a checksum which has a high probability of being unique for a given character string. The dictionary portion is checksummed if the "DICT" option appears. The 'P' option may be used to direct output to the printer.

A message is output, giving checksum statistics, in the following form:

```
BYTE STATISTICS FOR file-name (or attribute name):
TOTAL = t AVERAGE = a ITEMS = i CKSUM = c BITS = b
```

where:

"t" is the total number of bytes in the attribute (or item) included
"a" is the average number of bytes per item
"i" is the number of items
"c" is the checksum
"b" is a bit count

The attribute mark trailing the specified attribute (or item) is included in the statistics.

Sample output statistics are shown in Figure B.

To use checksums, you should issue CHECK-SUM commands for all files, or portions of files, to be verified and record the output statistics. Subsequently, the CHECK-SUM commands can be reissued to verify that the checksum statistics have not changed. The checksum for a given file must be recalculated whenever you update that file.

* The item-list can be supplied by a previous SELECT, SSELECT, GET-LIST or FORM-LIST command, in which case the selection-criteria is omitted.

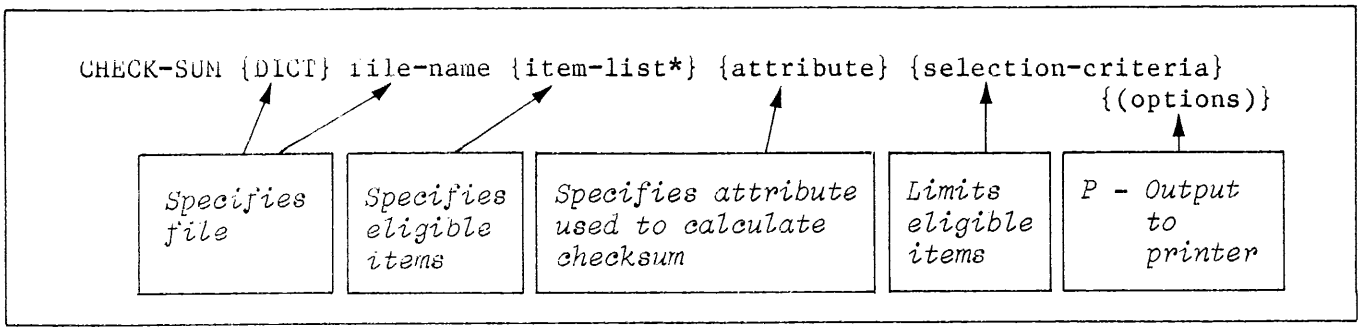


Figure A. General Form of CHECK-SUM Verb

```
:CHECK-SUM TEST <cr>

BYTE STATISTICS OF TEST :
TOTAL = 70 AVERAGE = 7.00 ITEMS = 10 CKSUM =38444 BITS =245

:DELETE TEST 13 <cr>          <-- Update file
'13' DELETED

:CHECK-SUM TEST <cr>          Note changed statistics

BYTE STATISTICS OF TEST :
TOTAL = 63 AVERAGE = 7.00 ITEMS = 9 CKSUM =35607 BITS =220

:CHECK-SUM ACCOUNT NEXT-ACCOUNT <cr>

BYTE STATISTICS OF NEXT-ACCOUNT :
TOTAL = 392 AVERAGE = 5.76 ITEMS = 68 CKSUM =148837 BITS =1484

:CHECK-SUM DICT ACCOUNT <cr>      (D/Code will be used since it is a
                                   default attribute -- item '1')
BYTE STATISTICS OF D/CODE :
TOTAL = 183 AVERAGE = 1.96 ITEMS = 93 CKSUM =53070 BITS =888
```

Figure B. Examples of CHECK-SUM Verb

10.17 Group Format Errors: FIX-FILE-ERRORS Verb

The FIX-FILE-ERRORS verb is designed to assist SEQUEL users in determining the nature of group format errors they have encountered. In many instances the program will recover data which would previously have been lost.

The FIX-FILE-ERRORS verb can be misused! It should be used only by experienced users, and then only after a cautious examination of the situation.

The general form of this verb (Figure A) is:

FIX-FILE-ERRORS {DICT} file-name

You should be logged on to an account with 66 frames of additional work-space. The TSYM file must be defined on this account.

The FIX-FILE-ERRORS verb operates on either the data or dictionary portion of a file. This operation is performed on each group until all groups have been processed. When possible, the operation will remove data in error and place it in the dictionary of the TSYM file.

Two passes are made through each group. Pass one verifies the frame linkages of the group, and pass two examines the data.

Pass one first checks the links of the primary file space. Errors are reported in the following format:

LINK FIELD ERROR - GROUP n FRAME n

LINKS xx xxxx xxxx xx

If the primary space crosses the maximum FID, the program stops with an appropriate message. After processing the primary space, the operation scans links of secondary frames searching for an incorrect backward link and for the end of group. If an incorrect backward link is found, the program scans (at most) another 66 frames in that group.

Pass two scans the data one item at a time. Items are examined for format errors, described in Figure B. Corrective actions for these errors also are listed in this figure.

If segment marks are encountered, they are relaced with backarrows (_). A message of the form "SM @ xxxx.yy" is displayed, where xxxx.yy is a frame and displacement. Error items built in the TSYM dictionary are given special item-ids. The item-id is composed of:

Error type, FID of first frame in group, sequential integers

Note that subsequent use of this program or the assembler causes the error items to be lost because TSYM is cleared!

```
FIX-FILE-ERRORS {DICT} file-name
```

Figure A. General Form of FIX-FILE-ERRORS Verb

ERROR TYPE	DESCRIPTION	CORRECTIVE ACTION TAKEN
O	Premature end of data	*
N	Nonhex character in count field	*
C	Count field out of range (was <5 or >32267)	*
A	No attribute mark at end of item	*
I	An item-id exceeds 50 characters	**
H	An item-id does not hash to current group	**
(none)	A segment mark was found in the item	

*A new item is created in the TSYM dictionary and a group-terminating attribute mark is placed after the last good item. Any secondary frames beyond the last item are disconnected from the group.

**Offending items are removed from the file and placed in the TSYM dictionary. The scan continues through the group.

Figure B. Errors Detected and Corrective Action Taken by Pass Two of FIX-FILE-ERRORS

10.18 Source Compaction: ECOPY, EEDIT and EBASIC Verbs

This topic describes the SEQUEL system capability to maintain source programs in a compacted form.

You may compact source code (for DATA/BASIC programs) so that the source programs take less file space. The compaction is achieved with only a slight decrease in processing speed during editing, compiling, etc. Compaction is achieved by replacing any sequence of three or more blanks or asterisks with a delineator, a count of the number of characters compacted, and the character. The delineator used is the data link escape character (DLE, X'10').

The compaction takes place when an item is written into a file using the 'FI' command of the EDITOR (see the EDITOR Programming Manual). This processor is the only one which is able to compact data. The EDITOR can also expand the data during the time it is being edited (i.e., while in the IS and OS work areas). In addition to the EDITOR, the DATA/BASIC compiler and COPY process evoke the expansion feature. Note: The assembler does not recognize this feature.

To use this feature, you must specify one of these new verbs in your TCL statement.

Note that the ECOPY verb will expand items as they are copied. Since this is usually not what is desired, you should, in general, use the normal COPY verb when copying compressed items.

(THIS PAGE LEFT BLANK INTENTIONALLY)

10.19 File Management PROCs: DELETE and SET-FILE

DELETE and SET-FILE are two PROCs available for file management.

DELETE PROC

This PROC is evoked by typing:

```
DELETE file-name {item-id}
```

This PROC deletes the specified item. If an asterisk (*) is used in place of an item-id, the item-ids will be supplied by a SELECT, SSELECT, GET-LIST, or FORM-LIST. If no item is specified, the message "NEXT ITEM" prompts you for an item-id. A carriage return ends the input prompting.

SET-FILE PROC

The SET-FILE PROC provides a quick way to set up a file synonym definition item (Q-pointer) to any file in the system. The general form is:

```
SET-FILE {account-name {file-name}}
```

where "account-name" is the name of the logon account the file is defined in, and "file-name" is the name of the file that a Q-pointer is to be set up for. SET-FILE always returns the message:

```
QFILE UPDATED
```

indicating that an item called "QFILE" has been set up in the M/DICT as a Q-pointer to the required file. For example:

```
:SET-FILE SYSPROG XSYM <cr>
'QFILE UPDATED'
```

This example sets up QFILE to the XSYM file in SYSPROG.

Typing SET-FILE with one or both parameters missing will cause a request for that parameter:

```
:SET-FILE <cr>
ACCOUNT-NAME?SYSPROG <cr>
FILE-NAME?XSYM <cr>
'QFILE UPDATED'
```

This example causes the same action as the previous example. Typing a null line to either the request for "ACCOUNT-NAME?" or "FILE-NAME?" leaves the corresponding name in the 'QFILE' item unchanged from its previous value.

Figure A shows the general form of DELETE and SET-FILE PROCs.

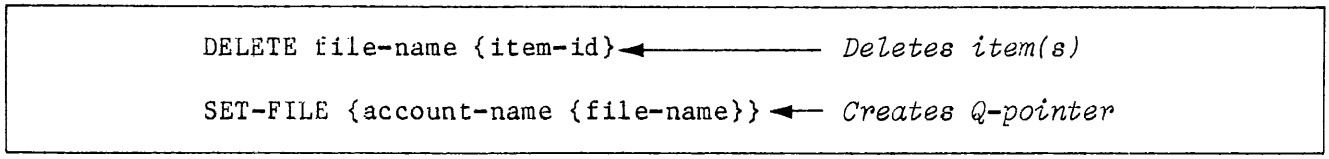


Figure A. General Form of DELETE and SET-FILE PROCs

10.20 Printronix Printer Adjustments: PRINTRONIX Utility

A cataloged utility program called "PRINTRONIX" enables you to change the form length and number of lines per inch on the Printronix printer.

This utility is executed from the SYSPROG account by typing "PRINTRONIX" at TCL. Upon its execution, you are prompted for the form length, number of lines per inch, and channel specifications. After you enter your responses to the prompts, the utility transmits the necessary control characters to the printer.

NOTE: The data is sent to the printer as a "normal" print job (i.e., job is placed in the job queue). Therefore, any changes caused by the PRINTRONIX utility will not take effect until the other print jobs ahead of it are printed. Remember, you have the option of changing the priority of print jobs via SP-JOBS Action Code 10 and SP-PRIORITY verb.

The PRINTRONIX utility also allows you to equate IBM "channel" numbers with a particular line number. For example, if Channel Three is equated to Line Six, the printer moves to the sixth line when a "Skip to Channel 3" command is sent to the printer. This feature is used only when receiving print format data from an IBM system via the 5750 Communications Terminal.

On the adjacent page you see two examples of PRINTRONIX usage. In Figure A, the user is specifying a form length of 11 inches and 88 lines per page (11 x 8). Note that the "8 LPI" button must still be pressed if eight lines per inches are desired.

In Figure B, a user is specifying a form length of 11 with 66 lines per page. In addition, two channel specifications are made. When this utility is executed, a "Skip to Channel 3" command moves the paper to the sixth line; a "Skip to Channel 4" command moves the paper to Line 33. Note that Channels 1 and 2 are reserved.

```
:PRINTRONIX <cr>
ENTER FORM LENGTH IN INCHES?11 <cr>
IS THIS 6 OR 8 LINES PER INCH?8 <cr>
VFU CHANNEL SKIPS (Y/N)?N <cr>
NOW SET PRINTER TO 6 OR 8 LPI AND ADJUST FORMS IN PRINTER.
```

Figure A. Sample Usage of PRINTRONIX Utility (Without Channel Specification)

```
:PRINTRONIX <cr>
ENTER FORM LENGTH IN INCHES?11 <cr>
IS THIS 6 OR 8 LINES PER INCH?6 <cr>
VFU CHANNEL SKIPS (Y/N)?Y <cr>
CHANNEL 1 USED AS TOP-OF-FORM
CHANNEL 2 USED AS DUMMY CHANNEL
CHANNEL, LINE#
?3,6
?4,33
?  <cr>
NOW SET PRINTER TO 6 OR 8 LPI AND ADJUST FORMS IN PRINTER.
```

Figure B. Sample Usage of PRINTRONIX Utility (With Channel Specification)

10.21 Miscellaneous PROCs

This topic describes various utility PROCs.

CT PROC

This PROC is evoked by typing:

CT file-name item-list* or * {(options)}

The item(s) specified are copied to the terminal. Options recognized by the COPY verb may be added.

CP PROC

This PROC is evoked by typing:

CP {file-name} {item-list* or *} {(options)}

The item(s) specified are copied to the printer. If the file-name and/or items are not specified, they are prompted for. Options recognized by the COPY verb may be added.

CHOO-CHOO PROC

The CHOO-CHOO PROC is evoked by typing:

CHOO-CHOO

This PROC prints a picture of a choo-choo train on your terminal (just for fun).

LISTACC PROC

The LISTACC PROC is evoked by typing:

LISTACC {account-names} {LPTR} {NOPAGE} {(options)}

This PROC lists accounting data for the account-names specified. If no account-names are specified, accounting data for all users are listed. Any valid ENGLISH options (described on facing page) may be included. Output is paged to terminal unless defeated.

LISTCONN PROC

The LISTCONN PROC is evoked by typing:

LISTCONN {file-name} {LPTR} {NOPAGE} {(options)}

This PROC sorts all connectives in the M/DICT or optionally specified file.

LISTDICTS PROC

The LISTDICTS PROC sorts all attribute or attribute synonym definition items in the M/DICT or the optionally specified file. The general form is:

LISTDICTS {file-name} {LPTR} {NOPAGE} {(options)}

LISTFILES PROC

The LISTFILES PROC sorts all file and file synonym definition items in the M/DICT or optionally specified file. The PROC displays the D/CODE of the file entry, the MODULO, SEPARATION, and BASE FID of the dictionary portion of the file, the number of ACCESS and RETRIEVAL codes, any correlatives, the output type (l, r, t, etc.), and the output length. The last three operate on the item-ids of the data items during ENGLISH output. The general form is:

LISTFILES {file-name} {LPTR} {NOPAGE} {(options)}

LISTPF PROC

The LISTPF PROC sorts the POINTER-FILE and produces a report displaying the account-names, types of cataloged items (C=cataloged DATA/BASIC program; L=save-list of item-ids/data), the names of the programs or lists, the FID of the first frame of the information, the number of frames occupied, the number of item-ids (or attributes if cataloged data) for save-lists, and the catalog date. The general form is:

LISTPF {LPTR} {NOPAGE} {(options)}

LISTPROCS PROC

LISTPROCS sorts all PROCs in the PROCLIB file or optionally specified file. The general form is:

LISTPROCS {file-name} {LPTR} {NOPAGE} {(options)}

LISTU PROC

The LISTU PROC lists the account names of all users currently active on the system, along with their PCB-FID, location, logon time and channel number.

The location field is stored as Attribute 1 in items with item-ids equivalent to the two digit channel number in the dictionary of the ACC file.

LISTVERBS PROC

The LISTVERBS PROC sorts all verbs (not PROCs) in the M/DICT or optionally specified file. The general form is:

LISTVERBS {file-name} {LPTR} {NOPAGE} {(options)}

LP80, LP106, LP136 PROCs

The LP-type PROCs set the terminal characteristics so printer listings will be formatted for paper of the following sizes:

LP80	8-1/2" wide by 11" long
LP106	11" wide by 8-1/2" long
LP136	14" wide by 8-1/2" long

N - *Inhibit paging of terminal output (NOPAGE)*
P - *Produce listing on printer (LPTR)*

Figure A. Valid ENGLISH Options

10.22 Miscellaneous PROCs

This topic describes additional utility PROCs.

FLIP PROC

The FLIP PROC presents a general feature listing and overview of SEQUEL concepts in a flip-chart format on the terminal. Subjects outlined on the terminal describe ENGLISH, PROC, SEQUEL file structure, etc. The demo outline is included for illustrative purposes only. The actual programs to execute a demo are not included and are left up to you.

SPACE PROC

The SPACE PROC may be used to perform multiple T-FWD commands to skip over a specified number of tape files. You must first attach the tape with the T-ATT verb. To execute this PROC, enter the word SPACE and the system will prompt for the number of files to T-FWD over.

COMPARE PROC

The COMPARE PROC compares items in two different files. Output includes the number of items in each file, a list of items that do not compare, and a list of items that are in one file but not the other. To execute the COMPARE PROC, enter the command COMPARE. The PROC prompts for the two file-names and asks if output is to be to the terminal or printer.

CHECK PROC

The CHECK PROC is similar to the SPACE PROC except that multiple T-CHKs are performed, checking a number of sequential tape files for parity errors. You must first attach the tape with the T-ATT verb.

GAMES

The Microdata SYS-GEN (initial system loading tape) contains a BP (DATA/BASIC program) file containing a number of sample games. Refer to the appropriate release installation instructions for the location of the file on the tape, and the loading procedure.

Figure A illustrates the sample usage of SPACE, COMPARE and CHECK PROCs.

```
:T-ATT <cr>
[1021] UNIT(#1) ATTACHED, RECORD SIZE = 512

:SPACE <cr>
SPECIFY NUMBER OF TAPE FILES TO T-FWD
MUST BE TWO DIGITS=03 <cr>
03
[94] END OF FILE
02
[94] END OF FILE
01
[94] END OF FILE
...YOU ARE SPACED OUT...


:COMPARE <cr>
ENTER FIRST FILE TO COMPARE ( [DICT] FILE-NAME ):AA <cr>
ENTER SECOND FILE TO COMPARE:BB <cr>
DO YOU WANT OUTPUT TO THE PRINTER (Y/N):N <cr>

FILE COMPARISON BETWEEN 'AA' AND 'BB'
PAGE 1     13:15:32  12 FEB 1983

-------------------------------------------------------------------------

ITEM COUNTS:  'AA': 36     'BB': 35

-------------------------------------------------------------------------

123-AA DOES NOT MATCH

1453-AC IS MISSING FROM BB

1632-ES IS MISSING FROM BB

8821-GG IS MISSING FROM AA

--- FINISHED ---


:CHECK <cr>
SPECIFY NUMBER OF TAPE FILES TO T-CHK
MUST BE TWO DIGITS=02 <cr>
02
[94] END OF FILE
01
[94] END OF FILE
...YOU ARE CHECKED OUT...
```

Figure A. Sample usage of SPACE, COMPARE, and CHECK PROCs

11 SYSTEM MAINTENANCE PROCEDURES: SYSPROG ACCOUNT

11.1 System Programming (SYSPROG) Account

Several special facilities are normally used from the System Programmer (SYSPROG) account. Procedures on this account are normally performed by persons more familiar with the overall operation of the system.

To log on to the SYSPROG account, when the message (or similar prompt):

```
LOGON PLEASE:
```

is displayed, type:

```
SYSPROG[,password] <cr>
```

where "password" is the appropriate password set up for SYSPROG. Alternate logon names (such as SP, which has no additional work space frames) may be used.

Figures A and B show some of the common verbs and PROCs available on the SYSPROG account.

:ABSLOAD	DE-STOP
:ABS/FILES	DSKAD
ACCOUNT-RESTORE	:FILES
BUFFERS	LOCK-FRAME
CHARGE-TO	MEM-DIAG
DE-COPY	SAVE
DE-RESET	SEL-RESTORE
DE-START	STRIP-SOURCE
	UNLOCK-FRAME

Figure A. Some SYSPROG Verbs

ACCOUNT-SAVE	SYS-GEN
ALL-FILE-STATS	SYS-UPDATE
CREATE-ACCOUNT	SYSTEM-SETUP
FILE-SAVE	UPDATE-ACCOUNT
SETUP-ASSY	VERIFY-SYSTEM

Figure B. Some SYSPROG PROCs

11.2 Minimizing Data Loss During Restores: SET-WRITES Verb

The SET-WRITES verb flushes write-required memory buffers. Use of this verb minimizes data loss during a coldstart and ABS restore.

Because of SEQUEL's large main memory size, it is possible that the system is waiting an inordinate amount of time before copying write-required buffers to disc. This may result in a loss of data in the event a coldstart or ABS restore is performed. To avoid such losses, the SET-WRITES verb is designed to enforce the writing of write-required buffers to disc at a user-defined rate: write-required memory buffers are written to disc after a specified number of port activations, regardless of buffer usage. These buffers are flushed in sequential order (i.e., after the first "set" of activations, the first write-required buffer is flushed; after the second "set" of activations, the second write-required buffer is flushed, etc.).

The format of the SET-WRITES verb is:

```
SET-WRITES k
```

where 'k' is a decimal number between 1 and 2,147,483,646 that specifies the number of activations between each disc write. This number is reset to a default value of 2,147,483,646 when a coldstart is performed.

Listed below are some guidelines you should consider before using the SET-WRITES verb:

1. If your system rarely needs to be coldstarted, the SET-WRITES verb is probably not needed.

2. If your system is coldstarted regularly and

   a. if you run large applications (large in terms of main memory usage) and key in small amounts of data, set 'k' to a high number

   -or-

   b. if you run "tight" programs (those that use a minimum of main memory) and key in large amounts of data, set 'k' to a low number.

It is not possible to apply a fixed "correct" value to the SET-WRITES verb because you begin to see degradation of response time as 'k' is lowered. The number of users on the system at any given time, the type of processing, and the frequency of coldstarts are all considerations. Thus, a value judgement must be made by the person responsible for the system.

A basic setting used at many system sites is a value of 20,000. At the very least, this can be a starting point. The number can be varied at different times of day, or varied on a daily basis. Use of the BUFFERS verb with the 'T' option gives you the total number of write-required buffers.

The following restrictions apply to the SET-WRITES verb:

1. The SET-WRITES verb must be executed from SYSPROG.
2. You must have SYS(2) privileges.

To better understand the way that a given SET-WRITES setting may affect the operation of your system, we suggest you use one of the following formulae: 1) calculate how long it will take to cycle through main memory with a given SET-WRITES setting or 2) calculate the correct SET-WRITES setting to achieve one complete cycle through main memory in a given amount of time. Both formulae are described and illustrated in the following two topics.

Figure A illustrates the format of the SET-WRITES verb.

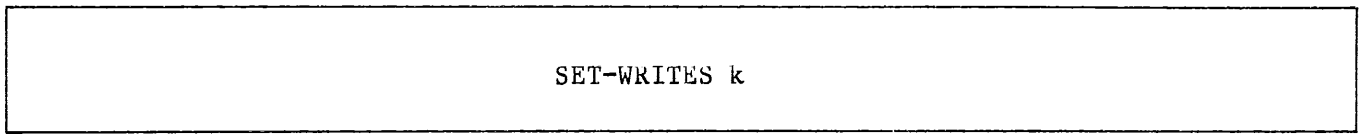


Figure A. Format of SET-WRITES Verb

11.3 Using SET-WRITES Formula #1

Formula #1 calculates a SET-WRITES setting ('k'). This topic describes the equation to calculate 'k' and illustrates the use of the formula in Figure A.

Before you can use either Formula #1 or #2, a series of system-dependent variables must be defined:

ATS = Average of timeslice values of system processes.
KBUF = 2 * total megabytes of main memory.
NDISC = Total number of disc drive units.
AVXRT = Average disc I/O transfer rate (i.e., AVXRT represents an approximation of the maximum throughput, in frames/buffers per second, that the disc sub-system can handle), to be calculated by the following formula: AVXRT = (50-(NDISC*5))*NDISC.
MINT = Minimum cycle time, in seconds, in which KBUF can be flushed to disc; use this formula: MINT = (KBUF/AVXRT)*1000.
TM = Actual Cyle Time, in seconds. This will either be the time it takes for one complete cycle through main memory, or the amount of time that a complete cycle through main memory will take with a given SET-WRITES value. Note that TM is simply a time value in seconds that cannot be less than MINT.
k = The value of SET-WRITES.

FORMULA #1: DETERMINING A SET-WRITES SETTING ('k')

This formula calculates the value for the SET-WRITES setting ('k') when it is desired that SET-WRITES make a full cyle through memory in a given time. To ensure that the time given (TM) does not exceed the saturation point of the disc sub-system transfer rate (AVXRT), TM cannot be less than MINT.

To calculate 'k', follow these steps:

1. Solve for AVXRT, then MINT.
2. Convert the desired cycle time to seconds. This is TM. If TM is less than MINT, then round TM upward to MINT (TM cannot be less than MINT).
3. Now solve for 'k': k = TM/(ATS*KBUF).

Parameters used in this example:

1 Megabyte of main memory:	KBUF = 2*1 = 2
10 Milliseconds average timeslice:	ATS = 10
2 REFLEX-II disc drives:	NDISC = 2

STEP 1. Calculate AVXRT and MINT:

```
AVXRT = (50-(NDISC*5))*NDISC
      = (50-(2*5))*2
      = (50-10)*2
      = 80 (transfers per second)

MINT  = (KBUF/AVXRT)*1000
      = (2/80)*1000
      = .025*1000
      = 25 (seconds)
```

STEP 2. Calculate the SET-WRITES setting ('k') from a given time of TM.

For this example, let's say that we want to cycle through memory every 30 minutes.

```
TM    = 30 minutes * 60 seconds
      = 1800 seconds
        NOTE:  If TM < MINT then TM = MINT.

k     = TM/(ATS*KBUF)
      = 1800/(10*2)
      = 1800/20
      = 90
```

Thus, with the given parameters and the desire to cycle through memory every 30 minutes, the SET-WRITES setting would have to be 90.

Figure A. Sample Use of SET-WRITES Formula #1

11.4 Using SET-WRITES Formula #2

Formula #2 calculates the amount of time it takes for a full cycle-through of main memory given a SET-WRITES value of 'k'. This topic describes the equation to calculate TM and illustrates the use of this formula in Figure A.

To calculate TM, follow these steps:

1. Solve for TM as follows:

   TM = k*ATS*KBUF

   NOTE: If TM < MINT then TM = MINT.

2. Use TM as the number of seconds it will take to perform a full cycle-through of main memory with a SET-WRITES setting of 'k'.

NOTE: If a SEQUEL system goes into an idle state, where the condition is such that virtual processes are not being activated for some period of time, all buffers in main memory that are in a write-required state will be written out to disc by the system Monitor. The elapsed time over which this process will be completed is equal to or less than the value calculated for the MINT variable. There is a high probability that this process will take less time than MINT depicts because the value calculated for AVXRT is based on an I/O transfer rate for the disc sub-system that tends to be on the modest side.

Parameters used in this example:

1 Megabyte of main memory:	KBUF = 2*1 = 2
10 Milliseconds average timeslice:	ATS = 10
2 REFLEX-II disc drives:	NDISC = 2

STEP 1. Calculate AVXRT and MINT:

```
AVXRT = (50-(NDISC*5))*NDISC
      = (50-(2*5))*2
      = (50-10)*2
      = 40*2
      = 80 (transfers per second)

MINT  = (KBUF/AVXRT)*1000
      = (2/80)*1000
      = .025*1000
      = 25 (seconds)
```

STEP 2. Calculate the approximate time it would take SET-WRITES to enforce one complete cycle-through of main memory given the SET-WRITES setting ('k').

Let's say that the SET-WRITES setting 'k' is 10000.

```
TM = k*ATS*KBUF
   = 10000*10*2
   = 100000*2
   = 200000 seconds
      NOTE:  If TM < MINT then TM = MINT.
```

STEP 3. Convert TM (the actual time in seconds) to minutes and/or hours:

```
CYCLE TIME IN MINUTES = TM/60
                      = 200000/60
                      = 3333.3333 minutes
```

Thus, with the given system parameters and the SET-WRITES setting of 10,000, it would take 3333 minutes, or a little over two days (55.5555 hours), for SET-WRITES to cycle-through memory one time.

Figure A. Sample Use of SET-WRITES Formula #2

11.5 ACCOUNT-SAVE PROC

The ACCOUNT SAVE PROC on SYSPROG allows you to save one account on tape.

This PROC functions similarly to the FILE-SAVE PROC, except the 'I' option of the SAVE verb is used. No coldstart or ABS section is created. The general form of the input line is:

```
ACCOUNT-SAVE {(T)}
```

After entering the ACCOUNT-SAVE PROC, you see the prompt:

```
ACCOUNT NAME=?
```

Respond to this prompt with an account name represented by a D-pointer in the SYSTEM Dictionary. No synonym Q-pointers in the SYSTEM Dictionary are saved. The STAT-FILE is cleared and loaded with statistics for only the specified account. A T-DUMP of the STAT-FILE is appended and the file statistics report are printed. Use the 'T' option to inhibit the writing of a tape label at the beginning of the files section.

Figure A illustrates the ACCOUNT-SAVE PROC.

```
:ACCOUNT-SAVE <cr>
09:33:00  07 APR 1983

TAPE RECORD BLOCK LENGTH IF OTHER THAN 512?  <cr>
[1021] UNIT(#1) ATTACHED, RECORD SIZE = 512
[94] END OF FILE

START PASS1 -- WRITE THEN READ 20 RECORDS OF X'55AA55AA'
START PASS2 -- WRITE THEN READ 20 RECORDS OF X'00F00FF0'
TAPE READ AND WRITE COMPLETE WITH NO ERRORS

TAPE LABEL IF DESIRED = ACCOUNTS PAYABLE <cr>

. . . START ACCOUNT SAVE . . .

ACCOUNT NAME = ?ACCP <cr>
ACCP
.
.
.
[99] 85 TAPE RECORDS WRITTEN.
. . . END OF ACCOUNT SAVE . . .
. . . DUMPING STAT-FILE . . .

67 ITEMS DUMPED.

. . . PREPARING FILE STATISTICS REPORT . . .

:
```

<cr> ← Produces default of 512 bytes

ACCOUNTS PAYABLE <cr> ← Specifies tape label identification

ACCP <cr> ← Account name

Figure A. Dialogue for ACCOUNT-SAVE PROC

11.6 Account-Restores

The ACCOUNT-RESTORE verb adds a new account (from tape) to an existing system.

ACCOUNT-RESTORE

An account-restore can be performed from a formatted file-save of a whole system or from a save of a single account (Figures A and B). In either case, the entire account is restored and a pointer to the account is created in the SYSTEM Dictionary. To use ACCOUNT-RESTORE, mount the tape at load point and select one of these methods, depending on the type of tape being used.

A. For full file-save tapes, follow these steps:

1. Attacn the tape with the proper record size.
2. If a bootstrap is present, enter:
   ```
   T-FWD <cr>
   ```
3. Now enter:
   ```
   T-RDLBL 1 <cr>
   ```
4. If an ABS section is present, enter:
   ```
   T-FWD <cr>
   ```
5. Now that the tape is positioned at the files section, type:
   ```
   ACCOUNT-RESTORE <cr>
   ```
6. Answer the prompt "ACCOUNT-NAME?" with the name of the account. The account name must be the same name used on the file-save tape.

B. For single account-save tapes, follow these steps:

1. Attach the tape with the proper record size.
2. Enter:
   ```
   T-RDLBL 1 <cr>
   ACCOUNT-RESTORE <cr>
   ```

The restore will procede automatically.

Do not restore an account whose name matches any item in the System dictionary. This means that you must delete an old account before restoring another account with the same name. If desired, a Q-pointer in SYSPROG's M/DICT to the new account may be added.

A 'synonym' segment may be encountered with a base which has not been found on the tape. This happens if a D-pointer on the saved account points to a file on another account (illegal use of multiple D-pointers), or if a 'D' segment on the tape is unrecognizable because of a parity error. In this case, the message

'SYNONYM NOT FOUND'

appears, and no D-pointer is created.

```
:T-ATT 8192 <cr>
[1021] UNIT(#1) ATTACHED, RECORD SIZE = 8192

:T-RDLBL 1 <cr>          <-- Reads label preceding account
LABEL = DOC.ACCT]08:01:33  01 JUL 1982

:ACCOUNT-RESTORE <cr>    <-- Initiates account-restore
 STATUS 11137,17,1
  ERRORS 11154,1,1
   DL/ID 11155,3,1
 M/DICT 11137,17,1 (S)
 DL/ID 11137,17,1 (S)
  TP 11159,1,1
   DL/ID 11160,7,1
  PARCEL 11306,3,1
   DL/ID 11309,503,1

:                        <-- Returns to TCL when finished
```

Figure A. Account-Restore from Account-Save Tape

```
:T-ATT 8192 <cr>
[1021] UNIT(*1) ATTACHED, RECORD SIZE = 8192

:T-FWD  <cr>             <-- Bypass coldstart section
[94] END OF FILE

:T-RDLBL 1 <cr>          <-- Reads label preceding ABS section
LABEL = DOC.ACCT]08:01:33  01 JUL 1982

:T-FWD  <cr>             <-- Bypass ABS section
[94] END OF FILE

:ACCOUNT-RESTORE  <cr>   <-- Initiates account-restore

ACCOUNT NAME?STATUS  <cr>  <-- Prompts for desired account
 STATUS 11948,17,1
  ERRORS 11965,1,1
   DL/ID 11966,3,1
 M/DICT 11948,17,1 (S)
 DL/ID 11948,17,1 (S)
  TP 11970,1,1
   DL/ID 11971,7,1
  PARCEL 12117,3,1
   DL/ID 12120,503,1

:                        <-- Returns to TCL when finished
```

Figure B. Account-Restore from Complete File-Save Tape

11.7 Selective-Restores

The selective-restore capability allows individual files or items to be loaded onto the system from a formatted file-save tape.

SEL-RESTORE

This verb restores a file from a formatted file-save or account-save tape (Figure A). Before using SEL-RESTORE, the tape must be positioned correctly. Mount the tape at load point and select one of these methods, depending on the type of tape being used.

A. For full file-save tapes, follow these steps:

1. Attach the tape with the proper record size
2. If a bootstrap is present, enter:
   T-FWD <cr>
3. Now enter:
   T-RDLBL 1 <cr>
4. If an ABS section is present, enter:
   T-FWD <cr>

B. For single account-save tapes, follow these steps:

1. Attach the tape with the proper record size
2. Enter:
   T-RDLBL 1 <cr>

Now that the tape is positioned at the beginning of the files section, enter:

SEL-RESTORE file-name item-list or * {(options)}

where "file-name" is the file on the system in which items will be placed. This file must be defined on the account from which the restore is run. "Item-list" is the list of items to be restored. An asterisk (*) may be used to indicate all items on the tape file. You are then prompted for:

ACCOUNT NAME? a-name
FILE NAME? f-name

where "a-name" is the name of the account under which the file was saved on tape, and "f-name" is the name of the file as it appears on the tape. Entering <cr> to FILE NAME? causes the account Master Dictionary to be restored. "F-name" may be of the format 'file', 'DICT file' or 'file DL/ID' or 'file zzzz' where "zzzz" is a D-pointer in the dictionary of 'file'.

Applicable options are:

O Overlay items already on the file if they have the same item-ids.

A The tape is already positioned in the desired account.

N File is to be identified on tape by its file number, in which case the prompt will be "FILE#?". The required file number is the one which accompanies the file on the file statistics printout for the appropriate file-save (under the column labeled "ORDER").

C This option must be used with the 'N' option. It causes every item before the next end of file to be a candidate for restore. This ensures data can be restored even if a D-pointer on tape is damaged.

When using SEL-RESTORE, remember that account dictionaries follow all other files for the account. To restore Q-pointers for the SYSTEM Dictionary, use the 'N' option with File# = 1. This will be the last file on the tape.

```
:T-ATT 8192 <cr>                          <-- Attach tape.
[1021] UNIT(#1) ATTACHED, RECORD SIZE = 8192

:T-FWD <cr>                               <-- Bypass coldstart section.
[94] END OF FILE

:T-RDLBL 1 <cr>                           <-- Read label to assure proper size.
LABEL = DOC.ACCT]08:01:33  01 JUL 82

:T-FWD <cr>                               <-- Bypass ABS section.
[94] END OF FILE

:SEL-RESTORE DICT TEMP * (O) <cr>         <-- Restore all items into DICT TEMP.

ACCOUNT NAME?SYSPROG <cr>                 <-- File on tape is called DICT STAT-
FILE NAME? DICT STAT-FILE <cr>            <-- FILE under SYSPROG account.
 SYSTEM 1
  BLOCK-CONVERT 2
   DL/ID (S)
  DL/ID (S)
  PROCLIB 5
   DL/ID (S)
  SYSPROG 7
   MD (S)
   M/DICT (S)
   DL/ID (S)
   STAT-FILE  11
[992] 33 ITEM(S) HAVE BEEN RESTORED
```

Figure A. Example of SEL-RESTORE

11.8 TCL Initiation of File-Restores: :FILES and :ABS/FILES Verbs

A complete formatted file-restore is part of a complete system ('AF') load. One may also be initiated as part of a coldstart (use the 'F' option) if it is desired to keep the current system software. Finally, a file-restore may be initiated from TCL by using the :FILES or :ABS/FILES verb.

Complete File-Restore

To use :FILES (Figure A), mount the first reel of the file-save and perform these steps:

1. Attach the tape with the proper record size.
2. If a bootstrap is present, enter:
   ```
   T-FWD <cr>
   ```
3. Now enter:
   ```
   T-RDLBL 1 <cr>
   ```
4. If an ABS section is present, enter:
   ```
   T-FWD <cr>
   ```
5. Now that the tape is positioned at the files section, type:
   ```
   :FILES <cr>
   ```

Note that all files (including spooler files) on the current system are overwritten, the spooler is restarted, and all lines are sent to logon.

The verb :ABS/FILES functions similarly to the :FILES verb, but may be used to restore both the ABS and files portions of a system. To use this verb, put the first reel of the save tape on the drive and perform these steps:

1. Attach the tape with the proper record size.
2. If a bootstrap is present, enter:
   ```
   T-FWD <cr>
   ```
3. Now enter:
   ```
   T-RDLBL 1 <cr>
   :ABS/FILES <cr>
   ```

```
:T-ATT 2048 <cr>
[1021] UNIT(#1) ATTACHED, RECORD SIZE = 2048

:T-FWD <cr>
[94] END OF FILE

:T-RDLBL 1 <cr>
LABEL = DOC.ACCT]08:01:04  11 APR 1982
TAPE SIZE CHANGED TO 08192

:T-FWD <cr>
[94] END OF FILE

::FILES <cr>

SPOOLER STARTED
SYSTEM 3136,11,1
 BLOCK-CONVERT 3147,7,1
 DL/ID 3147,7,1 (S)
 CHANNEL 3154,1,1
  DL/ID 3155,7,1
DL/ID 3136,11,1 (S)
 PROCLIB 3162,7,1
 DL/ID 3162,7,1 (S)
 POINTER-FILE 3169,5,1
 DL/ID 3169,5,1 (S)
 SYSPROG 3174,69,1
 MD 3174,69,1 (S)
 M/DICT 3174,69,1 (S)
 DL/ID 3174,69,1 (S)
  STAT-FILE 3243,1,1
   DL/ID 3244,87,1
  DISC-ERRORS 3331,3,1
   DL/ID 3334,1,1
  ERRMSG 3335,17,1
  DL/ID 3335,17,1 (S)
 ACC 3154,1,1 (S)


LOGON PLEASE:
```

Attach tape unit; a block length of 2048 bytes for the ABS and FILES sections is anticipated.

To bypass bootstrap section.

Reads tape label.

Label indicates record size of 8192 bytes.

Note: All lines go to Logon.

Figure A. Example of :FILES Verb from TCL Using Formatted File-Save Tape

11.9 Initializing the System: SYSTEM-SETUP PROC

After you complete a full restore, system upgrade, ABS load, coldstart or warmstart, initialize the system with the SYSTEM-SETUP PROC.

SYSTEM-SETUP

Execute the SYSTEM-SETUP PROC after a restore, coldstart, or warmstart to reset disc error counts, start disc error recording, initialize system locks, reset the time and date, and to perform other functions associated with setup.

Figure A presents a sample dialogue resulting from the use of this PROC.

```
:SYSTEM-SETUP <cr>

 THIS PROC IS USED TO INITIALIZE CERTAIN SYSTEM
 PARAMETERS DEPENDING ON WHAT TYPE OF
 LOAD WAS JUST PERFORMED:

 1. (AF) FULL RESTORE OR SYSTEM UPGRADE
 2. (A)  ABS LOAD
 3. (X)  COLDSTART
 4. (W)  WARMSTART

 ENTER OPTION NUMBER: 1 <cr>

ALLOCATE PERMANENT LOGON WORKSPACE

HOW MANY PAGES ARE TO BE EJECTED ON THE STANDARD QUEUE (0,1,2,3,4,5)? :1 <cr>

DO YOU WANT MEMORY ERROR RECORDING?  (Y,N)?Y <cr>

NOW RESETTING DISK ERROR COUNT

NOW STARTING DISK ERROR RECORDING

NOW RESETTING MEMORY ERROR COUNT

NOW STARTING MEMORY ERROR RECORDING

NOW CLEARING THE ACCOUNTING FILE OF ACTIVE USERS

THE CURRENT SYSTEM TIME AND DATE IS 10:32:26  31 JAN 1983

DO YOU WISH TO CHANGE THE TIME OR DATE (Y/N)? :N <cr>

 YOU ARE NOW BEING LOGGED OFF
```

Figure A. Sample Dialogue Using SYSTEM-SETUP PROC

11.10 Formatted File-Saves and FILE-SAVE PROC

File-saves should be performed on a daily basis to protect the data base. Formatted file-save tapes are also a means of transporting software and data to other machines.

A formatted (logical) file-save dumps files in hierarchical order. Because each item is treated individually, file-save tapes may be used for selective restores. It also allows statistics to be gathered by the save. This approach allows users on the system while the save is running, though updates to files or file groups already saved will not be saved on the tape.

FILE-SAVE PROC

The general form of the input line is:

```
FILE-SAVE {(T)}
```

The FILE-SAVE PROC counts the number of hold files and prompts for the tape record size (a carriage return produces a default of 8192 bytes). The record size must be a multiple of 512. Any reply that does not satisfy this requirement is rounded up to the next 512 multiple. The process then attaches the magnetic tape unit, rewinds the tape, writes an EOF mark, and then rewinds the tape. Next a tape checkout program is executed and the tape is rewound. The next prompt is:

```
TAPE LABEL IF DESIRED=
```

Any string up to 16 characters may be included in the tape label. A tape label is always written on the tape preceding the ABS section. It contains the time and date, and reel number, and the tape block record size. If the 'T' option is used, the tape label is not written.

The file-save may now proceed. A coldstart section, an ABS section, and files section are dumped. Upon completion, a T-DUMP of the STAT-FILE is appended. This T-DUMP could later be loaded into the STAT-FILE and the PROC 'LIST-FILE-STATS' could be executed to show the contents of that tape. After the T-DUMP, the PROC 'LIST-FILE-STATS' is entered to produce a file statistics report.

Figure A illustrates a file-save process.

```
:FILE-SAVE <cr>                                    Default is 8192 bytes.
09:33:00  07 APR 1983
THERE ARE 2 HOLD FILES
TAPE RECORD BLOCK LENGTH (IF OTHER THAN 8192)? <cr>
[92] TAPE ATTACHED, RECORD SIZE = 8192
[94] END OF FILE

START PASS1 -- WRITE THEN READ 20 RECORDS OF X'55AA55AA'
START PASS2 -- WRITE THEN READ 20 RECORDS OF X'00F00FF0'
TAPE READ AND WRITE COMPLETE WITH NO ERRORS

TAPE LABEL IF DESIRED = PRODUCTION SYS <cr>  <-- Up to 16 characters of
                                                 tape identification.

... START FILE SAVE ... 09:34:02  07 APR 1983

77 ITEMS SELECTED.
[94] END OF FILE

40 ITEMS SELECTED.
[94] END OF FILE

33 ITEMS SELECTED.
ENTER FILE NAME FORMATTER
NOW MAKING FRAME LIST AND LABEL FOR --->FORMATTER
[94] END OF FILE

A1-1215  <-------------------------------------- Indicates ABS area dumped.

SYSTEM   <-------------------------------------- Listing of system files.

   .
   .
   .
[99]  699 TAPE RECORDS WRITTEN
... END OF FILE SAVE ... 09:53:57  07 APR 1983
T-DUMP STAT-FILE

130 ITEMS DUMPED
[94] END OF FILE
NOW PREPARING FILE-STATISITCS REPORT

FILE SAVE COMPLETE . . .
```

Figure A. Starting a Formatted File-Save

11.11 File-Save Tapes: Format

A formatted file-save tape has many types of data. The format of a file-save tape is discussed below.

A formatted file-save tape created by the FILE-SAVE PROC contains the following:

The coldstart section consisting of several 512-byte frames of system modes in a binary-copy format

An end-of-file tape mark

A tape label

An ABS section consisting of a series of tape segments formatted as follows:

```
_A^ffff^...(512 bytes binary)...^cccc
```

where:
- _ segment mark
- ffff 4 digit hex ASCII frame number
- ^ attribute mark
- cccc 4 digit hex ASCII checkcode = sum of the 512 bytes

A final section consisting of an 'X' followed by enough blanks to complete the last tape record

An end-of-file tape mark

A file section containing various segments each separated by a segment mark

An end-of-file tape mark

A T-DUMP of the STAT-FILE

An end-of-file tape mark

Several types of segments are in the files section (Figure A). The first byte of each is a segment mark. This was designed so that restores would not be stopped if some error destroyed tape segment(s). Segments are referred to by the second character or "segment type". Currently, there are D, Q, B, I, O and X segments. The following topic describes these segments in more detail.

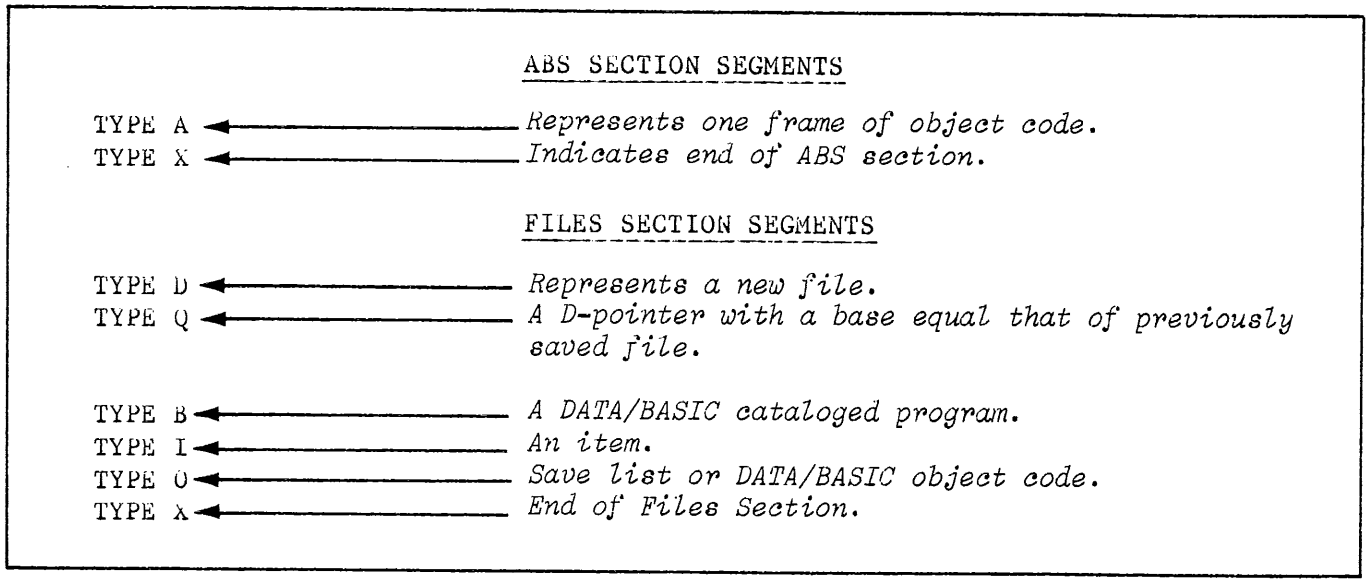


Figure A. Formatted File-Save Segment Types

11.12 Tape Segments in Formatted File-Save

Files section segment types D, Q, I and B are described below.

D-segments identify a new file. Their format is:

(SM)D(AM)level(AM)filecount(AM)size...pointer...(AM)

where "level" is 0, 1, 2, or 3, and equals one plus the level of the dictionary in which the D-pointer is placed. There is only one level zero D-segment, which identifies the SYSTEM Dictionary. This segment is derived from the item 'SYSTEM' in the SYSTEM Dictionary. A level one D-segment defines a pointer to an account Master Dictionary, a level two D-segment defines a file dictionary, and a level three D-segment defines a pointer to a data file. "Filecount" is a sequential file number in hex assigned to D-segments. The "size" field and the rest of the pointer item is saved as it is in the dictionary.

A Q-segment is generated whenever a D-pointer is found with a base which is already in the table of dumped files. Its format is:

(SM)Q(AM)level(AM)size....pointer...(AM)

where "level" is as defined above.

I-segments are generated one per item dumped. Their format is:

(SM)Ifilecount(AM)group(AM)size....item...(AM)

The "filecount" (HEX) is identical to the filecount of the D-pointer which points to the file in which the item is found; "group" is the hex FID of the first frame of the group from which the item was dumped. This is saved to eliminate hashing, and to ensure that the scheme of the "fast method" of restoring files can proceed by sequential groups.

Items in the POINTER-FILE are saved as B-segments of the following form:

(AM)size...item...(AM)

In addition, each frame (the first of which is pointed to by a POINTER-FILE item containing the cataloged object or SAVE-LIST) is saved with a segment of the form:

(SM)OB(AM)filecount(AM)1000 bytes of frame in ASCII hexadecimal format (AM)

Each binary byte in the cataloged frame is converted to two ASCII hexadecimal bytes in the tape segment. Typical formatted tape segments are shown in Figure A.

```
RECORD = 5 ; 512 BYTES.

000: ME^A^6^^^^^^^R^5^5^_I6^5519^001C5^S^5^BEGIN^5^^^^^
050: R^8^1^_I6^5519^00192^S^2^RS^2^^^^^L^2^1^_I6^5519^0
100: 01BRESULT^A^2^^^^^^^L^2^1^_I6^5519^001B8^S^8^VERS^
150: 8^^^^^L^2^1^_Q^2^0022DL/ID^D^18546^47^1^^^^^L^20^1
200: ^_D^2^9^0026TEMP^D^20626^1^1^^^^^L^10^^^(1,1)^_D^3
250: ^A^0027DL/ID^D^20627^1^1^^^^^L^10^^^(1,1)^_D^2^B^0
300: 029NEWAC^D^21787^11^1^^^^^L^10^^^(11,1)^_Q^3^0020D
350: L/ID^D^21787^11^1^^^^^L^10^_IB^551B^0014SP-EJECT^P
400: ^10A8^_IB^551B^0015BLOCK-PRINT^P^41^_IB^551B^0011M
450: ULD^PM^40A0^_IB^551B^001E*A9998^A^9998^^^^^^^R^4^1
500: ^_IB^551B^00


RECORD = 6 , 512 BYTES.

000: 1D7^S^7^V/CONV^^^^^^L^15^1^_IB^551B^000BAND^C1^_IB
050: ^551B^000FHEADING^CL^_IB^551B^0018T-WEOF^PZ^6033^B
100: 023^_IB^551B^001BV/MIN^A^11^^^^^^^L^3^1^_IB^551B^0
150: 017FORM-LIST^P^2^303A^_IB^551B^001B*A10^A^10^^^^^^
200: ^L^10^1^_IB^551B^0019*A8^A^8^^^^^^^L^10^1^_IB^551B
250: ^0010USER^P^40BB^_IB^551B^001F11^S^11^V/MIN^11^^^^
300: ^L^3^1^_IB^551B^0016T-LOAD^PT^35^207C^_IB^551B^000
350: CDICT^CY^_IB^551B^002FBLOCK-CONVERT^Q^BLOCK-CONVER
400: T^^^^^^^L^10^1^_IB^551B^0015SP-ASSIGN^P^10A7^_IB^5
450: 51B^0022LP132^PQ^(DICT PROCLIB LP132)^_IB^551B^001
500: 7DATE-FORMAT
```

(Segment marks (X'FF') print as '_' here).

Item SP-EJECT (size: 20) in group X'551B' in the 11th file.

Q-segment (level 3: data file) defines DL/ID which has same base, modulo and separation as file NEWAC.

D-segment (level 2: dictionary file) defines NEWAC file as the 11th file.

Block of I-segments, each defining an item in the NEWAC file.

Figure A. Typical Formatted Tape Segments

11.13 Operation of Formatted File-Save

This topic describes the operation of the formatted file-save process.

The file-save dumps all pointers and all subfiles at each level before dumping the data for that level. This is done so that D-pointers can be placed at the front of their group when the file is restored; e.g., for a simple system, the order might be:

```
     SYSTEM POINTER

     ACCOUNT1 POINTER
     FILE-DICTIONARY1 POINTER
  *  DATA1 POINTER
  *  DATA1 ITEMS
  *  DICTIONARY1 ITEMS
     FILE-DICTIONARY2 POINTER
     DATA2 POINTER
  *  DATA2 ITEMS
  *  DICTIONARY2 ITEMS
  *  ACCOUNT1 ITEMS

     ACCOUNT2 POINTER
     .
  *  .
     .

     SYSTEM DICTIONARY ITEMS
         * Blocks of I-segments
```

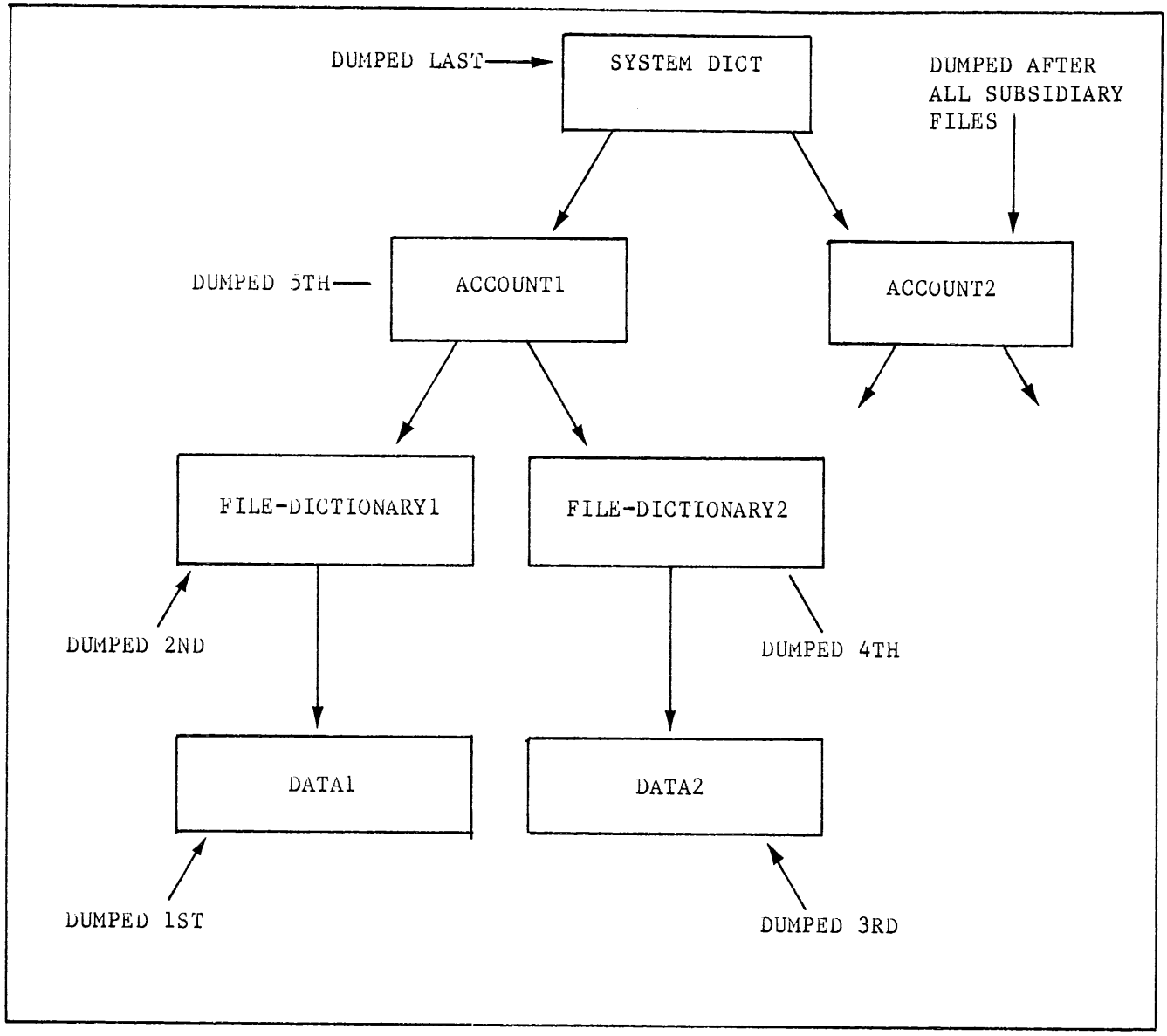


Figure A. Typical Layout of Formatted File-Save Process

11.14 Binary Save/Restore: Overview

The Binary Save/Restore (BS/R) utility performs a binary dump of the disc to magnetic tape, providing a fast method of data backup. It is loaded into memory from tape, replacing the normal SEQUEL monitor. The BS/R utility is not intended to totally replace the formatted file-save utility.

The Binary Save/Restore utility performs a disc-to-tape transfer, duplicating the disc byte for byte onto tape. Therefore, the BS/R tape should only be used to restore systems with the same disc and terminal configuration as the source system. Individual files and items cannot be restored with the BS/R process. Conditions such as Group Format Errors are not detected by this utility. In addition, since individual bytes are transferred, the integrity of the data is related to the integrity of the tape media itself. For these reasons, the Binary Save/Restore should not be used to totally replace the formatted file-save utility. The BS/R utility may be used for daily backups, while using the formatted file-save for weekly or monthly backup operations.

Saves

The Save portion of the Binary Save/Restore program transfers the data stored on each disc into memory, one track at a time, and writes each track of data to tape as one tape record. Disc space is left untouched during the saving process and is rewritten during the restore process. After either process, a SEQUEL system is brought to life by a system coldstart.

The BS/R program modules are included in the item "COLD-LIST" which is standardly used to generate a "COLDSTART" section on tape (refer to the section titled COLDDUMP AND SAVE VERBS). The "COLDSTART" section is written at the beginning of a binary-save tape as well as file-save tapes. The boot procedure using the binary-save tape displays the message "OPTIONS(X,B,D) = ". The "B" option places the BS/R program in memory and proceeds to question your intention of saving or restoring binary disc data. The binary data is saved immediately after the "COLDSTART" section on tape, separated by an end-of-file mark (EOF). Thus, after a save or a restore, the first reel of a binary-save tape may be used to coldstart ('X' option) a SEQUEL system.

As the binary-save operation proceeds, the process automatically assigns the most efficient record size according to the type of disc unit(s) on the system. During the process of transferring disc content to tape, progress is displayed in terms of tape record number, current disc being saved, and disc sector number.

You have the option of verifying the tape generated by the save. If this option is chosen, each reel is verified prior to the mounting of the next reel (for multi-reel saves). This verification checks the ability to read the written tape, validates the sequential integrity, and compares the tape data to that reread from the disc.

RESTORES

When a system is restored from a BS/R tape, the utility first reads and displays the tape label to ensure the correct tape has been mounted. The process then reads a subsequent record which has preserved the disc configuration table of the source system. If the disc configuration of the object system is not the same as the source system, the process displays an error message and aborts. Otherwise, tape records are written to disc as would be compatible with the source system.

After each record is written to disc, the system verifies that the disc frames written may be read correctly. The system automatically rereads the disc frames written and compares the data with that read from the tape; however, if the restore is performed at 100 ips, no reread is done.

Figure A lists both the features and considerations of the Binary Save/Restore utility.

FEATURES

- Copies the disc(s) byte for byte to tape.
- Provides for faster system backup than the formatted file-save.
- Provides for data verification both during the disc-to-tape transfer and the tape-to-disc transfer.

CONSIDERATIONS

- All users must be logged off during save and restore process.
- May be restored only onto systems with like disc and terminal configurations.
- Does not totally replace the formatted file-save.

Figure A. Features and Considerations of the Binary Save/Restore Utility

11.15 Binary Save/Restore: Tape Format

The Binary Save/Restore tape format is different than that resulting from a formatted file-save. The BS/R tape consists of two sections: a coldstart and a disc data section which includes a label (consisting of a reel identifier, the source system's disc configuration table, the last saved sector on each disc, a user's comment, time and date), and actual disc data. The two sections are delimited by an End-Of-File (EOF) mark.

The coldstart section consists of numerous program modules (512 byte records) which are called upon to establish the controlling program (i.e., monitor and BS/R) requested in response to the coldstart "OPTIONS" message.

At the end of the binary save, the tape generated contains the following:

A coldstart section as mentioned above.

An End-Of-File (EOF) mark.

A tape label consisting of 16 bytes of user information followed by the time, date, reel number, and information concerning the system's disc configuration. During a restore, the system's disc configuration is compared with that recorded here. If they are different, the restore is halted.

An Inter-Record Gap (IRG).

The system data in binary format.

An End-of-File (EOF) mark at the end of each reel and after the last data record of the last reel.

In the case of multi-reel saves, the subsequent reels consist of a tape label which identifies the reel sequence followed by the system data in binary format.

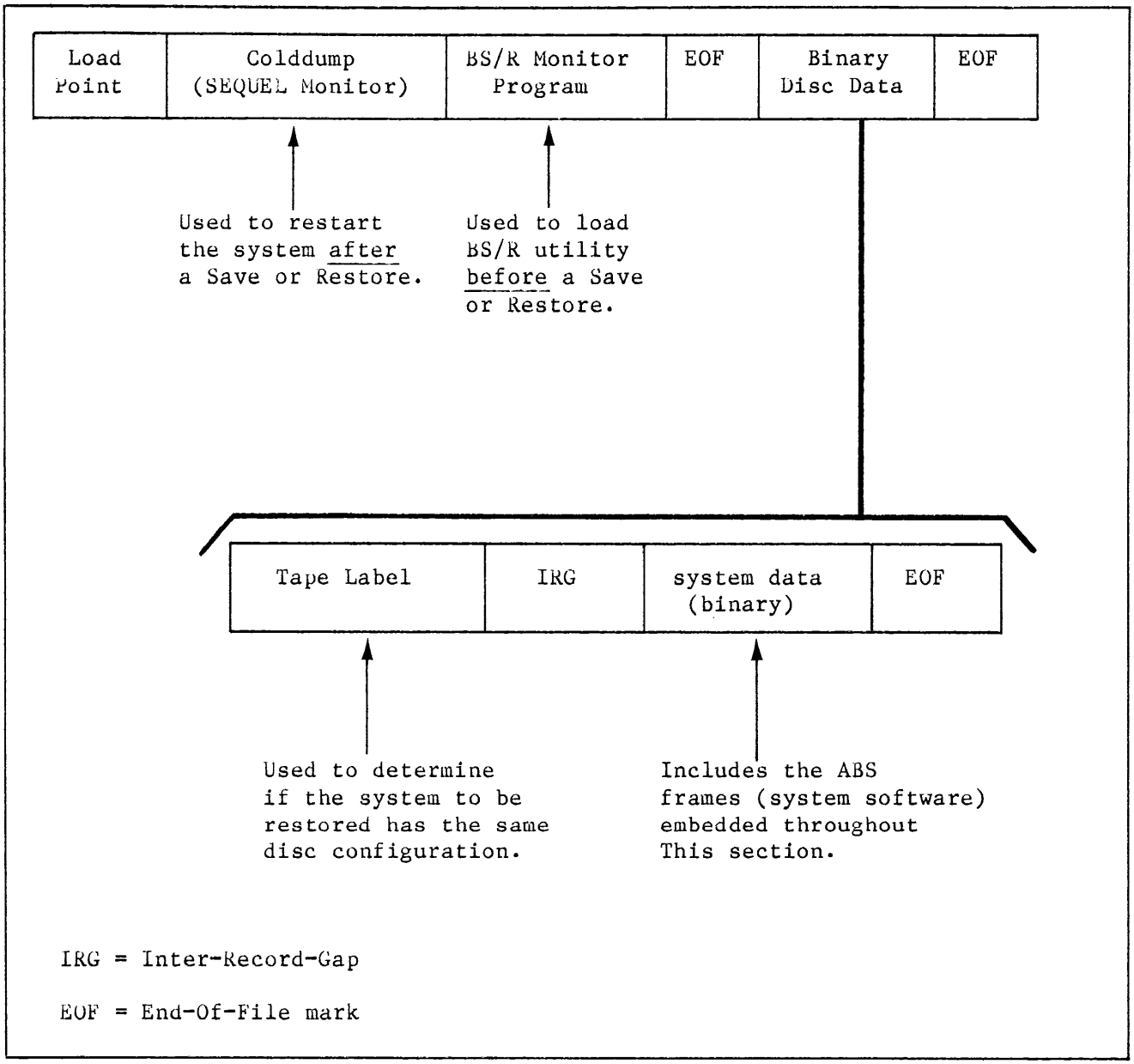


Figure A. Binary Save/Restore Tape Format

11.16 Binary Save/Restore: Disc Save (Part 1)

The Binary Save/Restore program modules must be loaded into memory from tape. As a stand-alone process, these BS/R modules may be first dumped to tape using the BINARY-SAVE PROC, and then loaded into memory by "bootstrapping" the system via the System Control Menu. This process overwrites the SEQUEL memory-resident monitor and loads the BS/R modules in its place.

To create a Binary Save Tape, you must: 1) make sure all users are logged off, 2) execute the SHUTDOWN PROC to "flush" residual updates to the disc (this PROC is described in Chapter 11), 3) load a scratch tape on the tape drive, and 4) log to SYSPROG from Terminal Zero.

The format of the BINARY-SAVE PROC is:

```
:BINARY-SAVE
```

Once you enter "BINARY-SAVE", the system COLDDUMPs a SEQUEL bootstrap section (MONITOR) to the tape which includes the BS/R monitor program. You are then asked the question:

```
START NEXT TAPE(Y/N)
?
```

Enter 'Y' if you wish to colddump a bootstrap section on another tape; enter 'N' to return control to TCL.

To load the program into memory from the BINARY-SAVE tape, turn the key switch to TEST, enter <c>D to display the System Control Menu, and enter Option Two (BOOTLOAD). When the "OPTIONS" message appears on Terminal Zero, enter 'B'. At this point the configurator display appears as for an 'AF' restore. Following this display, the message:

```
DO YOU REQUIRE A SAVE OR RESTORE?(S/R)
```

appears. To perform a disc-to-tape transfer, enter 'S'. The system returns with the prompt:

```
ENTER TAPE DRIVES TO BE USED WITH DRIVE x. <CR> IF NONE.
```

This prompt will be used for future enhancements. For this release, simply press the <RETURN> key. The next prompt you see is:

```
SKIP UNUSED CYLINDERS? (Y/N)
```

Enter 'Y' to skip cylinders associated with entries in the system's overflow table; enter 'N' to save every cylinder on all discs.

The next screen display is a report of the system's disc status. Each disc is individually started, tested and reported on. As each is tested, the message "TESTING DISC Dx" is displayed. If a disc fails the test, the system displays an error message and aborts the program. When a disc passes the test, "OK" is displayed and the program continues.

As soon as all discs pass the testing phase, the system displays the message:

STOP ON DISC ERRORS? (Y/N):

A 'Y' stops the program when a disc error is found, allowing you to then decide what to do (see the section BINARY SAVE/RESTORE: ERROR RECOVERY). An 'N' response allows the program to continue without your intervention; however, each error is logged and reported at the end of the save.

```
:BINARY-SAVE <cr>                          <-- Dumps a coldstart section and
[1021] UNIT(#1) ATTACHED, RECORD SIZE = 8192    the BS/R program to tape.
[94] END OF FILE

COLDDUMP COLDSTART

77 ITEMS SELECTED.
[94] END OF FILE

40 ITEMS SELECTED.
[94] END OF FILE

33 ITEMS SELECTED.
ENTER FILE NAME FORMATTER
NOW MAKING FRAME LIST & LABEL FOR --->FORMATTER
[94] END OF FILE
BINARY-SAVE TAPE GENERATED.
START NEXT TAPE(Y/N)
?N <cr>

:<c>D                                      <-- The <c>D brings up the System
                                               Control Menu; you then enter
                                               Option 2.

OPTIONS (X,B,D) = B
16 IS THE SPOOLER'S LINE
INITIALIZE SPOOLER TABLES? (Y/N):Y
ENTER NUMBER OF TIPH LINES: <cr>
2048K OF MEMORY
DRIVE D0,  REFLEX-II
DRIVE D1,  REFLEX-II
DRIVE D2,  REFLEX-II
CONFIGURAION CORRECT? (Y/N):Y
DO YOU REQUIRE A SAVE OR RESTORE (S/R):S
ENTER TAPE DRIVES TO BE USED WITH DRIVE 1. <CR> IF NONE. <cr>
SKIP UNUSED CYLINDERS? (Y/N)Y

TESTING DISK D0
 OK
TESTING DISK D1
 OK
TESTING DISK D2
 OK
STOP ON DISC ERRORS? (Y/N):Y
```

Figure A. Sample Dialogue of a Disc-to-Tape Transfer

11.17 Binary Save/Restore: Disc Save (Part 2)

The screen that follows the "STOP ON DISC ERRORS? (Y/N):" prompt asks you the following four questions:

1) VERIFY TAPES BY RE-READING: NO

'NO' is the default answer specifying no validation. Changing the response to 'YES' causes the system to read the saved data and compare it to that on disc.

2) REEL 1 CONTAINS BOOTSTRAP FILE: YES

Again, the default value is provided. A 'NO' response commands the program to write on the beginning of the currently mounted tape. Therefore, if you select this option, immediately mount a blank tape before the program is overwritten with data.

3) COMMENT FOR TAPE LABEL

Here you may enter a 30-character label.

4) DATE FOR TAPE LABEL

Enter whatever date you want to appear on the tape label.

At the bottom of the screen is the instruction on how to change a default entry or enter a label:

```
TO ENTER OR CHANGE DATA, ENTER PARAMETER NUMBER
TO PROCEED WITH SAVE, ENTER CARRIAGE RETURN:
```

For example, if you wanted to verify each save tape, enter the parameter "1". You would then be prompted for the proper information (i.e., a response of "YES"). After responding to any of the above prompts, an updated menu is repainted on the screen. If you did not wish to change a default value and/or enter a label, you would press <RETURN> to begin the transfer of data from disc to tape. As soon as the <RETURN> key is pressed, the screen is erased and, in the upper right-hand corner of the screen, a continually updating display of the save's progression appears in this format:

```
RECORD #     DRIVE #     SECTOR #
rrrrrrrr       dd        ssssssss
```

where 'rrrrrrrr' represents the current tape sequence number; 'dd' represents the disc being saved; and 'ssssssss' specifies the starting sector address of the track being saved.

When each reel is completed and rewound, these messages are displayed:

```
RECORD xxxxxxxx TERMINATES REEL
DISMOUNT REEL x
MOUNT REEL y ON DRIVE z
TYPE 'C' WHEN TAPE READY
```

The last two messages prompt you to mount the next tape and enter 'C'; these prompts only appear when performing a multiple-reel save. If the save concludes and you did not choose the verify option, the program terminates. If you did select the verify option, the reel is rewound to BOT and the verify is begun. As the verify proceeds, the sector number is updated.

When the entire save process is completed, the following messages appear:

```
DISMOUNT REEL z
SAVE COMPLETE
NO ERRORS
COLDSTART REQUIRED TO RESTART SYSTEM.
>
```

To perform a coldstart, mount reel one on the tape unit and enter <c>D at the ">" prompt.

```
                    SEQUEL BINARY SAVE/RESTORE

                          -SAVE OPTIONS-

                DISKS WHICH WILL BE SAVED :  D0 D1 D2

1) VERIFY TAPES BY RE-READING : NO

2) REEL 1 CONTAINS BOOTSTRAP FILE : YES

3) COMMENT FOR TAPE LABEL


4) DATE FOR TAPE LABEL


TO ENTER OR CHANGE DATA, ENTER PARAMETER NUMBER
TO PROCEED WITH SAVE, ENTER CARRIAGE RETURN: <cr>

                                          RECORD #  DRIVE #  SECTOR #
                                          00000096    D0     00001923


RECORD 000005C3 TERMINATES REEL

DISMOUNT REEL 1
SAVE COMPLETE

NO ERRORS
 COLDSTART REQUIRED TO RESTART SYSTEM.
E    0073F 00
>^D
```

Enter <c>D to display System Control Menu; select Option 2 (BOOTLOAD). When the "OPTIONS" message appears, enter 'X'.

Figure A. Sample Dialogue of a Disc-to-Tape Transfer

11.18 Binary Save/Restore: Restore

The Binary Save/Restore modules are loaded into memory from the BS/R tape by "booting" the system via the System Control Menu. A system may be restored from the BS/R tape(s) provided that its disc and terminal configuration is identical to the source system. The data is transferred byte for byte from the tape to the disc unit(s), the disc maps will appear to be the same. It is not intended that the Binary Save be used to transfer data from one system to another (the formatted file-save should be used for this purpose).

To restore a system using the BS/R tape(s), mount Reel #1 on the tape unit. Boot the system as described in the topic BINARY SAVE/RESTORE: DISC SAVE (PART 1). To the message:

```
DO YOU REQUIRE A SAVE OR A RESTORE?(S/R)
```

enter 'R'.

To the prompt:

```
ENTER TAPE DRIVES TO BE USED WITH DRIVE n.  <CR> IF NONE.
```

press the <RETURN> key.

The system then displays these messages:

```
MOUNT REEL 1 ON DRIVE 1
WHEN TAPE READY ENTER "C"
```

After you mount the first reel of the BS/R save and enter 'C', the label information from the tape is displayed on the upper half of the screen.

This display is followed by the message:

```
RESTORE PASS
REQUIRED DISK DRIVES ARE Dy Dz ...
```

The 'Dy' and 'Dz' represent the disc drives saved on the tape; these drives must also be present on the system being restored. If the identical configuration is not present, the restore is aborted. Should an abort occur, you may either mount another BS/R save tape, or perform a coldstart to reload the SEQUEL monitor (provided the system was already running). If no abort occurs, the disc drives are then tested; the results of the testing are seen as:

```
TESTING DISK Dx
 OK
```

After all discs are tested, you are asked:

```
STOP ON DISK ERRORS?  (Y/N)
```

Enter 'Y' to stop the program and display the disc errors; enter 'N' to continue the restore without interruption. After answering the above question, there is a five-second pause; at this point, the screen is erased. In the upper right-hand corner of the screen is a continual update of the restore's progress.

The restore transfers data from tape to disc until the current reel is exhausted or the discs are completely restored. Errors which may occur during the restore are detailed in the section BINARY SAVE/RESTORE: TAPE ERROR RECOVERY DURING RESTORE.

When the current reel finishes, the reel is rewound and an end-of-reel report is displayed:

```
TAPE INPUT,  NO ERRORS
DATA TO DISK;  NO ERRORS
```

These messages are displayed when there are no errors detected. Had there been errors, the "NO ERRORS" statement(s) would be replaced with the number of unrecoverable tape and disc errors.

Following this report, you are then instructed to mount the next reel (if you have a multiple-reel restore). The system reads the label of the newly mounted reel to ensure that the correct reel is mounted. If you mounted the wrong reel, you are notified by these messages:

```
REEL NUMBER OUT OF SEQUENCE
MOUNT REEL n
```

When the restore program concludes, an end-of-reel report is displayed, followed by:

```
RESTORE COMPLETE
```

Following "RESTORE COMPLETE" is an end-of-program report. This report is identical to the end-of-reel report except that accumulative totals of all reels are displayed. This concludes the Restore program. You may restart the system by performing a coldstart from either a BS/R bootstrap, or a sys-gen or file-save bootstrap.

```
DO YOU REQUIRE A SAVE OR A RESTORE?(S/R):R
ENTER TAPE DRIVES TO BE USED WITH DRIVE 1. <CR> IF NONE. <cr>

MOUNT REEL 1 ON DRIVE 1
WHEN TAPE READY ENTER "C"C

  COMMENT:
TIME/DATE:
  RESTORE PASS
REQUIRED DISK DRIVES ARE: D0 D1 D2

TESTING DISK D0
 OK
TESTING DISK D1
 OK
TESTING DISK D2
 OK
STOP ON DISC ERRORS? (Y/N):Y

                                          RECORD #   DRIVE#   SECTOR #
                                          00000096     D0     00001923

TAPE INPUT: NO ERRORS
DATA TO DISK: NO ERRORS
RESTORE COMPLETE

TAPE INPUT: NO ERRORS
DATA TO DISK: NO ERRORS
 COLDSTART REQUIRED TO RESTART SYSTEM.
E    00073F 00
><c>D
```

Figure A. Sample Dialogue of a Tape-to-Disc Transfer

11.19 Binary Save/Restore: Tape Error Recovery During Save

Generally, BS/R tape errors fall in two categories: recoverable (soft) and unrecoverable. All soft errors are reported to Terminal Zero at the end of each reel and at the end of the program. Depending upon the circumstances, some soft errors are reported at the time of their occurrance. Unrecoverable tape errors are reported to you as soon as they are detected.

To assist in BS/R tape error recovery, the system now writes a "sequence number" in the first four bytes of every tape record. As records are written to tape, this sequence number is incremented by one, beginning with the first record written on the first reel and ending with the last record written on the last reel. Should the system detect a discrepancy in sequence number order, the error is first logged and automatic error recovery is attempted. Should the recovery attempt fail, you may be given recovery options (system permitting).

Listed below are the different types of write errors that may occur while performing a binary save, how they are reported, and the options you have to overcome these errors.

Write Errors

When a write error is sensed, a WRITE-RETRY procedure is activated. This procedure moves the tape back several records, reads forward to synchronize the sequence numbers and repositions the tape at the beginning of the faulty record. The program then attempts to write that record again. If this procedure is successful, the save continues; if unsuccessful, the retry is repeated several times. When all retries fail, the message:

```
TAPE WRITE ERROR, RECORD n, DRIVE m
TYPE R(RETRY), T(TOPOFF), OR X(ABORT)
```

is displayed. Enter 'T' to write an end-of-file (EOF) marker after the last "good" record, report end-of-reel status, and prompt for another tape. Enter 'R' to rewrite the errant record. Enter 'X' to terminate the save program.

Verification Errors

When the optional verify portion of the save program encounters an error, the message:

```
THIS REEL DOES NOT VERIFY
ANY KEY WILL CONTINUE WITH NEXT REEL
```

is displayed. The tape is then rewound and you are asked to mount the next Save reel. Once you mount the next reel and press a key, the Save continues.

End-of-Save Status Report

When a Save is completed, the following information is displayed on Terminal Zero:

1) Last record number saved on reel
2) Number of disc errors

11.20 Binary Save/Restore: Tape Error Recovery During Restore

Described below are the different types of tape errors that may occur during a binary restore, how they are reported and the options you have for recovery.

Soft Read Errors

This error occurs when the tape hardware detects a problem and is able to correct it. Such errors are reported to you at the end of each reel and at the end of the Restore program.

DMA Overrun on Read

Should this error occur, a READ-RETRY routine is executed. This routine moves the tape back several records, reads forward to synchronize the sequence numbers, and repositions the tape at the beginning of the faulty record. The program then attempts to read the errant record again. If the retry is successful, the error is logged for subsequent reporting; if unsuccessful, the error is logged as unrecoverable and the following information is displayed:

1) Tape sequence number at the beginning of the reel
2) This information about the current record, the two previous records and, if available, the next two records:
   a) Tape sequence number
   b) Number of bytes in the tape record
   c) Whether the error was caused by a hardware problem
3) Status of the tape drive

You are then prompted for a decision:

```
TYPE R(RETRY), A(ACCEPT), OR K(SKIP)
```

where 'R' executes the READ-RETRY routine; 'K' skips the error and continues with the next program step; 'A' accepts the error, the error is logged in the end-of-program routine as an unrecoverable error, and the Restore continues where it left off.

Sequence Errors

When the Restore program encounters an unrecoverable sequence error, the following information is displayed:

1) Tape sequence number at the beginning of the reel
2) The following information about the current record, the two previous records and, if available, the next two records:
   a) Tape sequence number
   b) Number of bytes in the record
   c) Whether the error was caused by a hardware problem
3) Status of the tape drive

End-of-Reel Status Report

The following messages are reported to Terminal Zero when a reel is completed:

1) Number of unrecovered tape errors
2) Number of hard disc errors

End-of-Program Status Report

The following messages are reported to Terminal Zero when the last reel of the Restore is completed:

1) End-of-reel status for final reel (see above)
2) Total number of unrecovered tape errors on all reels
3) Total number of hard disc errors
4) An end-of-program message

11.21 Binary Save/Restore: Disc Error Recovery

When a disc error is detected, the Save or Restore program finds out if the error is recoverable. If the error is not recoverable, it is considered a hard disc error; if recoverable, the command executed at the time of the error is re-executed several times. As soon as the error is recovered, the error information is stored in the end-of-reel and end-of-program routine.

When a disc error is detected, the message:

```
DISC ERROR
SECTOR # nnnnnn
DISC STATUS mmmmm
STOP ON DISC ERRORS?  (Y/N)
```

is displayed. Enter 'Y' to abort the program; enter 'N' to accept the bad data and continue with the program.

THIS PAGE INTENTIONALLY LEFT BLANK.

11.22 Planned Shutdown Program

This program allows you to perform a secure and complete halt of the system (i.e., no data will be altered in a manner that cannot be accurately recovered when power is restored). You must have a SYS(2) privilege level to execute this PROC.

It is suggested that you use this program when you:

1) boot the system
2) perform a binary save or restore
3) change the system hardware
4) leave the office for the evening or weekend.

The following text includes a description of the prompts initiated by the program, responses expected from and options available to you, and an explanation of the system's status report.

NOTE: Before you run the Shutdown program, you must make sure that all active users are logged off.

Shutdown Program

To execute the Shutdown program, enter the following command:

SHUTDOWN <cr>

Once "SHUTDOWN" is entered, you are asked:

DO YOU WANT TO MESSAGE ALL USERS OF THE PENDING SHUTDOWN?

Enter 'N' if you do not wish to send the message; entering 'Y' causes this question to be displayed:

'HOW MANY MINUTES WARNING SHOULD BE GIVEN?'

Here you may enter a number from 1 to 5. Once a number is entered, the message "THE SYSTEM WILL SHUTDOWN IN x MINUTES. PLEASE LOGOFF." is repeated each minute along with the number of minutes remaining. Following these messages is a System Status Report.

System Status Report

The initial task of the Shutdown program is to evaluate the state of the system. By scanning through all configured PCBs, the system finds those users still logged on. Then, for each user logged on, their PIB and PCB is checked for type of process (i.e., phantom, spooler, or normal) and its status. This data is then used to generate the following status report covering all active processes:

PORT	PIB STATUS	ACTIVITY	ACCOUNT NAME	LOCATION
##	status	NORMAL	name	NO
		TIPH		OK
		SPOOLER		

Following this report are the following messages:

```
WARNING! THE SHUTDOWN SHOULD NOT PROCEED IF USERS ARE LOGGED ON,
OR IF THE SPOOLER IS ACTIVE!

DO YOU WANT TO PROCEED WITH THE SHUTDOWN? (ENTER 'Y' OR 'N').
```

Enter 'N' to terminate the program and return to TCL; enter 'Y' to begin the Shutdown. As soon as the Shutdown is completed, these messages are displayed:

```
SYSTEM SHUTDOWN
TYPE "R" TO RESUME
>
```

At this point, the system is in a secure state, and power to the system may be turned off. To an active terminal, the system now appears to have stopped; no one can log on to the system.

Restarting the System

There are three methods you can use to restart the system: 1) enter 'R' in response to the terminal prompt, 2) perform a warmstart or, 3) perform a coldstart.

You may enter 'R' or perform a warmstart if the system's memory devices were maintained after the Shutdown (i.e., a MOS system had battery backup). This procedure is especially useful if all active useres were not logged off. Either of these two methods should reactivate each process, thus appearing as if a Shutdown had never occurred.

You must perform a coldstart if: 1) memory boards were changed, 2) battery backup was not provided (for a system with MOS memory), or 3) a Binary-Save was performed after the Shutdown.

11.23 COLDDUMP and SAVE Verbs

The COLDDUMP verb creates a coldstart section on tape from items in the SYSTEM-OBJECT file. The SAVE verb allows you to dump ABS frames and files.

The COLDDUMP verb creates a series of binary data 512-byte tape records necessary to perform a bootstrap on any system. An item called COLD-LIST in the SYSTEM-OBJECT file contains the list of item-ids necessary for the coldstart. The FORM-LIST verb is used to select these items (see Figure A). Modes included in the monitor and boot loader are not located in the ABS area. The monitor is always resident in main memory. The general form of the COLDDUMP verb is:

```
COLDDUMP file-name {item-list} {(E)}
```

The 'E' option causes error messages to be printed.

The SAVE verb initiates dumping of the ABS and/or the files sections. It also generates file statistics. The general form of the SAVE verb is:

```
SAVE SYSTEM * {(options)}
```

where "options" are:

A Dumps ABS frames. ABS limits will be prompted for. Enter a <cr> to dump all ABS sp specified by the hardware, otherwise enter decimal numbers or ranges separated by commas. For example; 1-10,15,20-500.

D Dumps data files.

F Lists the name of every file as it is encountered. Otherwise, only account names are listed.

I Saves individual account.

K Kills output to tape. This option is used to generate statistics only.

S Creates file statistics in STAT-FILE.

T Inhibits writing tape label.

Z Used instead of the 'S' option to create file statistics. When the 'Z' option is used, a final prompt is made for the name of the file in which the statistics will be recorded.

Note that the 'S', 'Z', 'I', and 'F' options are meaningful only if the 'D' option is also specified.

Any prompts displayed (a prompt is indicated by a "?") will be in the following order:

1. An optional description to be included in the tape label. The description may be up to 16 characters.

2. ABS limits. See 'A' option above.

3. The name of the file in which the file statistics will be recorded if the 'Z' option is used.

Group Locks

The file-save locks each group before accessing data in the group, to prevent data loss due to "soft" group format errors. Up to four groups may be locked at one time; one in the SYSTEM Dictionary, one in the account Master Dictionary, one in a file dictionary, and one in a data file.

Treatment of Group Format Errors

If the file-save encounters a group format error, the secondary space is copied into new frames in overflow, the frames previously in the secondary space are disconnected, and the new frames are connected to the group. This is done to alleviate conflict between files that are using common frames for their secondary space.

Non-Saved Items and Skeleton Files

An item in a dictionary whose D/CODE is 'DX' will not be saved. A file pointer whose D/CODE is 'DY' will be saved, but items in the file will not be saved, such as the data portion of the STAT-FILE, which is not valid after a restore.

Use of Overflow

If the 'S' or 'Z' option is not used, the file-save can be run when there is no available disc space on the system. No additional space is needed unless the total number of files on the system exceeds 1500. One frame of overflow is needed for every 125 files on the system in excess of 1500.

```
:ASSIGN =TAPE 2
[1021] DEVICE(S) TAPE 2 NOW ASSIGNED,
       RECORD SIZE = 8190

:FORM-LIST SYSTEM-OBJECT COLD-LIST <cr>   Gets item-ids for items necessary
 xx ITEMS SELECTED.                       for coldstart.

 >COLDDUMP SYSTEM-OBJECT (E) <cr>         Creates coldstart tape suppressing
                                          messages except errors.
```

Figure A. Creating a Coldstart Tape

11.24 Creating and Enabling User-Accounts

The CREATE-ACCOUNT PROC creates new user-accounts. The SETUP-ASSY PROC enables the user-account to assemble SEQUEL assembly language programs. All accounts are set up to run DATA/BASIC. Both of these PROC's are present in the SYSPROG account.

CREATE-ACCOUNT PROC

The CREATE-ACCOUNT PROC performs the following:

- Creates a new file (to be the user's M/DICT) with the new user-name as the file definition item-id; the file definition item is placed in the SYSTEM Dictionary
- Copies the contents of the NEWAC file (the prototype M/DICT) to the newly created user M/DICT
- Adds a file synonym definition item in the SYSPROG account equated to the new M/DICT (to allow access from SYSPROG)

The CREATE-ACCOUNT PROC is evoked by typing in the PROC name:

```
CREATE-ACCOUNT
```

This PROC then prompts you for the required information which is summarized in Figure A. A null input (i.e., carriage return only) for any of the prompts (except ACCOUNT NAME, PRIVILEGES, and MODULO, SEPARATION) generates a null value for that attribute in the user identification item (see Chapter 6). A null input is not allowed for ACCOUNT NAME. A null input for "PRIVILEGES" generates a SYS0 level and zero additional work space. The modulo and separation default values are 29 and 1, respectively.

After all prompts have been answered, the system further prompts:

```
ENTER NUMBER TO CHANGE, * TO CANCEL, RETURN TO CONTINUE:
```

A carriage return in reply to the above prompt causes the PROC to create the new account. A number causes the corresponding entry-screen prompt to be reopened. An asterisk cancels the PROC. Once a carriage return is entered, the PROC should not be interrupted with the <BREAK> key.

The CREATE-ACCOUNT PROC should not be used to create a new synonym to an existent account; this should be done by using the EDITOR to create the file synonym definition item in the System Dictionary.

SETUP-ASSY PROC

The SETUP-ASSY PROC sets up a user-account so that it is able to assemble assembly language programs. To evoke this PROC, enter "SETUP-ASSY" followed by the account name. For example:

```
SETUP-ASSY USER3 <cr>
```

Once activated, the PROC asks a series of questions requiring input from you. These questions are self-explanatory. However, it should be noted that a response of "Y" to the prompt:

ENTER 'Y' TO CREATE CSYM AND XSYM FILES FOR THE ACCOUNT (Y/N)?

causes file %XSYM to be temporarily created on the SYSPROG account.

PROMPTS	EXPLANATION
ACCOUNT NAME ←	*The new account name may be up to 50 characters long and contain any characters except commas.*
L/RET-CODE(S) ←	*This is the set of retrieval lock codes to be associated with the user; multiple values can be entered separated by value marks (<c>], X'FD'). (The retrieval lock codes are described in the chapter LOGON/LOGOFF.)*
L/UPD-CODE(S) ←	*This is the set of update lock codes associated with the user (same as retrieval lock codes above).*
PASSWORD ←	*This is the password associated with the user's LOGON item in the SYSTEM Dictionary.*
PRIVILEGES ADDITIONAL WORK SPACE ←	*This is the code describing the user's privileges and work space assignment (described in the chapter LOGON/LOGOFF).*
ACCOUNT FILE UPDATE Y/N ←	*If 'Y' is entered, a 'U' is placed in the 9th attribute and logon history data will be accumulated in the ACCOUNT file.*
MODULO,SEPARATION ←	*This is modulo and separation for the M/DICT. Default value for modulo is 29; default value for separation is 1.*

Figure A. Information Required by CREATE-ACCOUNT PROC

11.25 Deleting User-Accounts: DELETE-ACCOUNT PROC

The DELETE-ACCOUNT PROC deletes all files belonging to a specified account, thus removing the account from the system.

DELETE-ACCOUNT is a PROC in SYSPROG-PL which prompts for the name of an account to delete. The dialogue is shown in Figure A.

After the account name is verified, the PROC displays all duplicate D/CODE pointers and Q-pointers to D/CODE pointers that point to the account. You are then given the option of cancelling the deletion or continuing with it. If you continue, all duplicate D/CODE and Q-pointers are deleted from the system.

DELETE-ACCOUNT also checks for users logged on to the master, duplicate D/CODE, and Q-pointer accounts. If any user is logged on, the delete is cancelled.

This PROC sets the group lock, as does the FILE-SAVE PROC, to prevent the account from being deleted during a FILE-SAVE operation.

Requirements to run DELETE-ACCOUNT are:

1. All other users should be logged off before running this PROC because an item in the System Dictionary will be deleted.

2. You must be logged on to SYSPROG.

3. DELETE-ACCOUNT uses the item-ids '%DAaccount-name%' and '%DAaccount-name%%' in SYSPROG's M/DICT; if the process is interrupted and terminated before completion for any reason, DELETE THESE ITEMS FROM SYSPROG'S M/DICT!

```
:DELETE-ACCOUNT <cr>

MICRODATA DELETE-ACCOUNT    !!! IF A FILE-SAVE IS SAVING THE ACCOUNT TO BE   !!!
                            !!! DELETED, THEN DELETE-ACCOUNT WILL WAIT UNTIL !!!
                            !!! THE ACCOUNT IS SAVED. . . . . . . . . . . .  !!!

ENTER ACCOUNT NAME? DUMMY <cr>
IS 'DUMMY' THE RIGHT ACCOUNT ? Y <cr>
CHECKING FOR GROUP LOCKS ON FILE-SAVE

NOW PREPARING LIST OF FILES TO DELETE

NOW DELETING ACCOUNT LEVEL FILES

**** DUMMY'S DL/ID DELETED
**** DUMMY'S M/DICT DELETED
**** DUMMY'S MD DELETED
**** DUMMY'S PROC DELETED

**** DELETE ACCOUNT 'DUMMY' COMPLETED
```

Figure A. Sample DELETE-ACCOUNT Dialogue

11.26 System Generation and Verification

This topic discusses SYSPROG PROCs that relate to the system generation process. For further information regarding these PROC's, refer to the installation procedures supplied with the applicable software release.

SYS-GEN PROC

The SYS-GEN PROC creates a SYS-GEN tape from which either an initial system load or a system update may be performed. The SYS-GEN tape consists of a coldstart section, an ABS section, a files section, and T-DUMPs of system files. The system places two EOF marks after the last tape file. Because the SYS-GEN PROC changes with each release of the system, it is not described here in detail.

SYS-UPDATE PROC

The SYS-UPDATE PROC updates system files after a new release of software is loaded. Because its use and operation are peculiar to each release of the system, detailed instructions are not given here.

UPDATE-ACCOUNT PROC

The UPDATE-ACCOUNT PROC updates accounts after a SYS-UPDATE to reflect changes to verbs, PROCs, and other elements of a M/DICT. It should never be used to update the SYSPROG account, which is always done separately as part of the system update procedure. Because the use and operation of this PROC is peculiar to each release of the system, detailed instructions are not given here.

VERIFY-SYSTEM PROC

The VERIFY-SYSTEM PROC verifies the object code in the SYSTEM-OBJECT file against the corresponding code loaded in the executable ABS region and lists modes that have mismatches.

If no options are specified, the MVERIFY 'E' option (list only errors) is assumed. Other options for the MVERIFY verb may be specified, such as, 'P' (output to printer) and 'A' (list all mismatch bytes). The MVERIFY verb is described in the SEQUEL Assembly Language Programming Manual.

This PROC should be executed whenever there is a suspicion that the system software is not operating correctly (e.g., abnormal aborts, etc.). If mismatches occur, the code loaded in the ABS area is bad and the corresponding item from the SYSTEM-OBJECT file should be MLOADed. If a checksum error occurs, then most likely the item in the SYSTEM-OBJECT file has changed (or was patched incorrectly). In this case, it is best to T-LOAD the item from a SYS-GEN tape, apply applicable patches, and then MLOAD the item.

(THIS PAGE INTENTIONALLY LEFT BLANK)

11.27 On-line Testing of Main Memory: MEM-DIAG Verb

The MEM-DIAG (Memory Diagnostic) verb on the SYSPROG account allows on-line testing of main memory and of data transfers to and from disc.

The MEM-DIAG operation may run concurrently with other processes. The diagnostic requires a block of contiguous frames which must be present (otherwise, the diagnostic ends without printing a message). The diagnostic runs by writing (and reading) data into (and from) these frames one at a time. The memory block each frame is read into is tested by propagating (via an MIIT instruction) the patterns of X'FF', X'00', X'55', and X'AA'. Each is then verified with an SCD instruction. This is repeated the number of times specified in the loop count parameter (x). As different frames are accessed, different main memory buffers are used. Note that the incidence of checking different buffers is random because different buffers are available at different times. Also, only buffers which are virtual (i.e., not memory locked and not used by the monitor) are checked.

Running this diagnostic not only checks main memory, but also checks data transfers to and from disc. On the second and subsequent loops through the contiguous block of frames (selected large enough to flush memory) the data pattern is checked in each frame as it was written to and read from the disc. An incrementing pattern of X'00' through X'FF' is used for this test.

The diagnostic puts items in the specified error file to tell the operator what it found. The diagnostic runs as long as these items fit in the primary space of the error file. To limit the diagnostic in the event of many errors, the diagnostic quits as soon as a secondary frame is needed.

The data portion of the error file should be cleared prior to execution of MEM-DIAG. The diagnostic must be terminated by pressing the <BREAK> key and typing "END".

The general form of the MEM-DIAG verb is illustrated in Figure A. The errors recorded in the specified file have the general format shown in Figure B. Hard errors are recorded only 10 times.

Errors found while testing a memory resident frame are given a code of "M" indicating memory failure. Errors found while checking a frame that has previously been written with the incrementing pattern are given code "C" indicating a transfer error. No attempt is made to determine a read versus write error.

The incidence of buffers tested by the diagnostic is recorded in PCB+28. Each of the first 128 bytes in this frame is a count of times the memory test was run in the corresponding buffer starting with buffer 0. The next 128 bytes contain counts of times the DMA check was run. After the counts reach 255, they are reset to 0. The only way to display these counts is with the debugger.

```
MEM-DIAG file-name {(x,yyyy-zzzz,c)}
```

where: file-name is the file which will be used to store errors

x is the loop count for the memory portion of the test; default is 3

yyyy is the lowest memory address to be tested (in hex); default = X'0000'

zzzz is the highest memory address to be tested (in hex); default = limit of current configuration (maximum = X'7FFFFF')

c is the size of the block of frames to be used. The default is 200 unless "yyyy" and "zzzz" are specified, in which case the default is:

$$\frac{zzzz-yyyy}{256}$$

Figure A. General Form of MEM-DIAG Verb

Item-id:	date*time	
Attr 1:	code	M-memory failure, C-disc read/write failures
Attr 2:	error addr	Memory location of first bad data byte
Attr 3:	correct-byte	What the data should have been
Attr 4:	error-byte	Actual data read from memory
Attr 5:	FID	Frame identification under test
Attr 6:	count	Number of errors in one frame

Figure B. General Form of Errors Recorded in Specified File

11.28 Disc-Error Recording

The system records disc errors for use by customer engineers.

When enabled, the system records disc errors as they occur. Up to 36 errors can be recorded, after which the oldest errors are overwritten. All types of disc errors except DMA channel overruns are recorded.

When a disc error occurs, an ampersand (&) is printed on the terminal that caused the disc operation. The rest of the system will not hang up on the one disc error.

The data recorded includes: FID, major and minor status, action code, time and date, error count, device address, unit number, and cylinder and sector.

Disc error recording is started automatically by an 'X', 'A', or 'AF' restore. The disc error table remains in its current state after an 'X' restore. However, after an 'A' or 'AF' restore, the table contains the disc errors recorded from the tape used to execute the restore. Therefore, the SYSTEM-SETUP PROC should be executed after an 'A' or 'AF' restore to initialize the disc error table. This table is not reinitialized by a warmstart (option 'W'); however, disc error recording is stopped. The SYSTEM-SETUP PROC should be used following a warmstart to begin error recording.

To stop disc error recording, execute the DE-STOP verb. To record the errors in a file, execute the DE-COPY verb. At system generation time a file DISC-ERRORS is created for this purpose.

The general form of the DE-COPY verb is:

```
DE-COPY file-name
```

The DE-COPY verb creates items in the file specified with item-ids in the form:

```
date*time*count
```

where "count" makes a unique item-id in case more than one error occurred within one time interval.

The item-body is in the form:

```
E^FID^major-status^minor-status^action-code
^count-of-errors^disc-device#^unit#^head#^cylinder^sector
```

This form is compatible with the memory diagnostic and the ATP (Acceptance Test Procedure) diagnostics. The DE-COPY verb stops recording and initializes the table. To continue recording errors, the DE-START verb must be issued.

The PROC LIST-DISC-ERRORS produces a report of the disc errors contained in the DISC-ERRORS file, while the PROC RESET-DISC-ERRORS clears the DISC-ERRORS file and executes DE-RESET and DE-START.

Figure A presents examples of the use of some of these commands.

```
:DE-COPY DISC-ERRORS <cr>

:LIST-DISC-ERRORS <cr>


PAGE   1                                              13:31:07   02 FEB 1983


DATE.... TIME. FID..... MAJ MNR DRV CMND CNT DEV HEAD CYL. SCTR..
                        STS STS STS          ADD

02/02/83 13:32      52878  20  80  31   80 838  D0           8813

END OF LIST
```

Figure A. Sample Usage of DE-COPY and LIST-DISC-ERRORS Commands

11.29 Disc Mapping: DSKAD Verb

The DSKAD verb computes and displays a disc address for a frame on the current system.

The general form of the DSKAD verb is:

```
DSKAD frame-number
```

Figure A shows an example of the DSKAD verb.

```
:DSKAD 1024 <cr>


DISK ADDRESS FOR FRAME NO.  1024
DISK ADDRESS   :   D2
HEAD NO. :          3
CYLINDER :          0
SECTOR :          4
```

Figure A. Sample Usage of DSKAD Verb

11.30 File Statistics Report: LIST-, ALL-, and ACCOUNT-FILE-STATS

The File Statistics Report is a valuable tool for data base management. This report is automatically created by the LIST-FILE-STATS, ALL-FILE-STATS, and ACCOUNT-FILE-STATS PROCs.

The Statistics Report adds data security by providing a list of file base, modulo and separation parameters, and by recording the order of files on a file-save tape.

The report is broken down by account. The following information is printed for each file in the account:

Minimum and maximum item size
Minimum and maximum number of items in a group
Breakdown of groups by percent full

A total line is generated for each account, which lists the total:

Items
Bytes (characters)
Frames (includes linked)
Group format errors

Figure A shows a portion of a File Statistics Report.

The LIST-FILE-STATS PROC prints the File Statistics Report using statistics generated by the last save.

The ALL-FILE-STATS PROC prints the File Statistics Report after creating a new set of statistics.

The ACCOUNT-FILE-STATS PROC is similar to ALL-FILE-STATS except that it prompts for one account name and generates statistics for that account only.

Statistics are recorded in a temporary file when the 'Z' option is used with the SAVE verb, or in the STAT-FILE when the 'S' option is used. When a temporary file is needed by one of the above PROCs, it is automatically created. Statistics are written to the file in use and then the File Statistics Report is generated.

06 DEC 1983 ACCOUNT NAME - SYSPROG PAGE 1

FILE	BASE.	MOD.	SEP.	ORDER.	ITEMS.	SIZE.....	MX/GP	MN/GP	PRAMES...	25%	50%	75%	100%	125%	150%	200%	>200%	GFE
%CHANGED%	17093	1	1	256	2	1,095	2	2	3	0	0	0	0	0	0	0	1	0
%CHANGED%*DL/ID	17093			257														
%OBSOLETE%	17092	1	1	254	2	475	2	2	1	0	0	0	1	0	0	0	0	0
%OBSOLETE%*DL/ID	17092			255														
BP	16605	43	1	247	42	255,442	5	0	541	18	1	0	1	0	1	1	21	0
BP*DL/ID	16605			248														
CSYM	16541	1	1	266	1	32	1	1	1	1	0	0	0	0	0	0	0	0
CSYM*DL/ID	16542	63	1	267	0	0	0	0	63	63	0	0	0	0	0	0	0	0
DISC-ERRORS	14611	1	1	261	19	639	19	19	2	0	0	0	0	0	1	0	0	0
DISC-ERRORS*DL/ID	14612	13	1	262	7	338	2	0	13	13	0	0	0	0	0	0	0	0
DL/ID	14211			251														
ERRMSG	14240	31	1	231	557	47,820	25	6	115	0	0	0	0	2	2	3	24	0
ERRMSG*DL/ID	14240			232														
JUNK	34001	1	1	239	1	31	1	1	1	1	0	0	0	0	0	0	0	0
JUNK*DL/ID	34002	1	1	240	0	0	0	0	1	1	0	0	0	0	0	0	0	0
M/DICT	14211			258														
MD	14211			265														
MEMORY-ERRORS	14314	3	1	235	9	284	4	2	3	2	1	0	0	0	0	0	0	0
MEMORY-ERRORS*DL/ID	14317	1	1	236	0	0	0	0	1	1	0	0	0	0	0	0	0	0
NEWAC	16376	11	1	252	304	7,086	33	22	22	0	0	0	1	3	6	1	0	0
NEWAC*DL/ID	16376			253														
OSYM	16397	1	1	259	1	32	1	1	1	1	0	0	0	0	0	0	0	0
OSYM*DL/ID	16398	23	1	260	510	20,169	33	14	56	0	0	0	0	3	4	10	6	0
PROC	14211			268														
PSYM	14318	1	1	237	1	32	1	1	1	1	0	0	0	0	0	0	0	0
PSYM*DL/ID	14319	23	1	238	691	12,342	39	25	40	0	0	0	9	11	3	0	0	0
RELEASE-LETTER	17657	37	1	241	11	58,533	2	0	151	32	0	0	1	0	0	0	4	0
RELEASE-LETTER*DL/ID	17657			242														
STAT-FILE	14586	1	1	245	36	988	36	36	2	0	0	0	0	0	0	1	0	0
STAT-FILE*DL/ID	14587	23	1	246	242	17,697	12	9	46	0	0	0	0	0	11	12	0	0
SYSPROG	14211	29	1	228	497	18,860	27	10	55	0	0	6	9	9	4	0	1	0
SYSPROG-PL	14356	13	1	243	60	110,877	8	2	228	0	0	0	0	0	0	0	13	0
SYSPROG-PL*DL/ID	14356			244														
SYSTEM-OBJECT	15037	1	1	269	10	307	10	10	1	0	0	1	0	0	0	0	0	0
SYSTEM-OBJECT*DL/ID	15038	137	3	270	485	681,826	6	1	1,435	0	0	0	2	11	2	10	112	0
TEMP	45739	1	1	263	1	32	1	1	1	1	0	0	0	0	0	0	0	0
TEMP*DL/ID	45740	11	1	264	2	1,665	1	0	13	9	0	0	0	0	0	2	0	0
TSYM	16450	11	1	233	1	32	1	0	11	11	0	0	0	0	0	0	0	0
TSYM*DL/ID	16450			234														
USER-MODES	45787	1	1	249	1	32	1	1	1	1	0	0	0	0	0	0	0	0
USER-MODES*DL/ID	55626	23	1	250	24	32,232	3	0	79	8	1	0	0	1	0	1	12	0
XSYM	16461	1	1	229	7	212	7	7	1	0	1	0	0	0	0	0	0	0
XSYM*DL/ID	16462	79	1	230	0	0	0	0	79	79	0	0	0	0	0	0	0	0
***					3,524	1,269,110			2,968									0

Figure A. Sample Printout of the File Statistics Report

11.31 File Statistics Report: STAT-FILE Dictionary

The STAT-FILE file is created as part of a system update. The SYS-GEN tape contains the standard dictionary for this file.

Creation of the STAT-FILE dictionary and data areas is part of a system update. STAT-FILE is contained on the System Programmer (SYSPROG) Account. As it is normally updated from this account, there is no need for STAT-FILE on any other account.

Dictionary items as on a SYS-GEN tape are shown in Figure A. A more complete description of these items is given in the Appendix.

When a file-save is started, the STAT-FILE data area is cleared and the current file statistics information is written into the data area.

Running the ALL-FILE-STATS PROC also clears the data file and builds the current file statistics.

The STAT-FILE data area will also be empty after a file-restore is done, because Attribute 1 of the DL/ID is a "DY". This is desirable as the statistics would no longer be applicable.

It is helpful to make synonym accounts in the System Dictionary as Q-pointers, leaving only the unique and principal accounts as D-pointers. The Statistics Report will then summarize accounts using the same unique account names.

Figure A is a listing of the STAT-FILE dictionary.

See the Appendix for further descriptions of these dictionary elements.

STAT-FILE.	D/CODE	A/AMC	V/CONV.....	V/CORR...	V/TYPE	V/MAX
ORDER.	A	0			R	4
ACCNAME.	A	1		G*1	L	0
ACCOUNTNAMES		1		G*1	L	0
FILE.	A	1		G1*2	L	20
FILENAME.	A	1		G1*1	L	10
LEVEL.	A	2			R	3
BASE.	A	3			R	5
MOD.	A	4			R	4
SEP.	A	5			R	2
ITEMS.	A	6	MDO,		R	5
SIZE.	A	7	MDO,		R	10
MAX-I	A	8	MDO,		R	6
MIN-I	A	9	MDO,		R	4
MX/GP	A	10			R	3
MN/GP	A	11			R	3
FRAMES	A	12	MDO,		R	9
25%	A	13			R	3
50%	A	14			R	3
75%	A	15			R	3
100%	A	16			R	3
125%	A	17			R	3
150%	A	18			R	3
200%	A	19			R	3
>200%	A	20			R	3
GFE	A	21			R	3
ROTOTAL	A	22			L	3
RITEMS	A	23			L	3
POINTERR	A	24			L	3
OERR	A	25			L	3
ROMAX	A	26			L	3
ROMIN	A	27			L	3
AVE.SIZE	S	99		F;6;7;/	R	7
SECONDARY.	A	30		F;5;4;*;12 ;-	R	8

Figure A. STAT-FILE Dictionary

11.32 Determining Memory Buffer Status: BUFFERS Verb

The BUFFERS verb may be used to obtain statistics regarding memory buffers.

To generate a buffer report, enter the BUFFERS verb in the form:

BUFFERS {(options)}

The status of each memory buffer used as virtual memory is displayed on the screen (or on the printer with the 'P' option). The status taken is a 'snapshot' and is self-consistent. The column labeled "LOC" is the location of the buffer in the memory. The number of buffers displayed is (k of memory)*2-4. Buffers are sorted by buffer number, unless the 'S' option is used, in which case the listing is sorted by frame.

The 'N' option inhibits paging of output to the terminal. The 'P' option directs output to the printer. The 'T' option produces a memory contents report. Information displayed includes the number of buffers occupied by the monitor, ABS frames, work space frames, user program data frames, I/O busy buffers, memory locked frames, and write required frames. The 'Z' option prints the status of buffers with a FID of zero (i.e., monitor code).

FIDs which print as X'FFFF10' through X'FFFF1F' are fake FIDs used to reserve memory buffers for tape I/O and vary according to the tape record size attached. Fake FIDs which print as X'FFFFFB' through X'FFFFDA' are the PIBs.

Figure A presents the general form of the BUFFERS verb. Sample usage of BUFFERS is illustrated in Figure B.

BUFFERS {(options)}

N -	*inhibit paging of terminal output*
P -	*output to printer*
S -	*sort by frame*
T -	*produce memory contents report*
Z -	*show monitor code buffers*

Figure A. General Form of BUFFERS Verb

```
:BUFFERS <cr>

   LOC:     FID:       I/O BUSY     MEMLOCK1     MEMLOCK     WRITE FLAG
  01400    FFFFFC                       X                        X
  01600    FFFFFB                       X                        X
  01800    FFFFFA                       X                        X
  01A00    FFFFF9                       X                        X
  01C00    FFFFF8                       X                        X
  01E00    FFFFF7                       X                        X
  02000    FFFFF6                       X                        X


:BUFFERS (T) <cr>

MEMORY CONTENTS:
3453  MONITOR BUFFERS
70    ABS FRAMES
312   WORK SPACE FRAMES
261   USER PROGRAM DATA FRAMES

0     I/O BUSY FRAMES
205   MEMLOCKED BUFFERS
9     WRITE REQUIRED BUFFERS
```

Figure B. Sample Usage of BUFFERS Verb

11.33 SYSPROG Utility Verbs: LOCK-FRAME and UNLOCK-FRAME

This topic describes two special utility verbs in the SYSPROG account.

LOCK-FRAME Verb

The LOCK-FRAME verb locks a frame in main memory. The general form of this verb is:

```
LOCK-FRAME number
```

where "number" is a decimal frame number. The LOCK-FRAME verb responds with the absolute hexadecimal word address of the memory buffer in which the frame is locked. The frame remains locked in memory until it is released by the UNLOCK-FRAME verb.

UNLOCK-FRAME Verb

The UNLOCK-FRAME verb clears the memory locked buffer status of the frame indicated. The general form of this verb is:

```
UNLOCK-FRAME number
```

where "number" is a decimal frame number.

Figure A presents the format of LOCK-FRAME and UNLOCK-FRAME verbs.

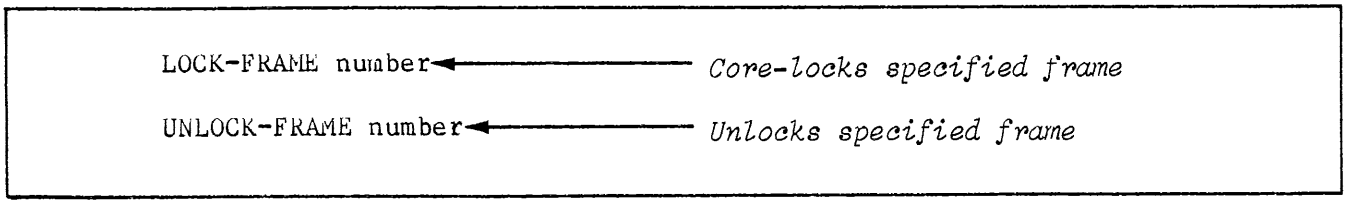


Figure A. General Forms of SYSPROG Utility Verbs

11.34 Group Locks and Execution Locks

The LIST-LOCKS verb displays group locks. The CLEAR-BASIC-LOCKS verb clears (resets) all execution locks. The :INIT-SYSTEM verb clears all group locks and execution locks.

LIST-LOCKS

The LIST-LOCKS verb lists the beginning FID of each group that is locked along with the line number which has the group locked. There are up to 128 group locks. Groups become locked by the file-save processor, or with DATA/BASIC READU, READVU, and MATREADU statements or PROC F-UREAD, and FBU commands, for example. Groups become unlocked by write operations into the group or by specific commands issued by a process. If an attempt is made to lock a group when all 128 locks are being used, the process will wait for a lock to become available.

Figure A presents an example of the LIST-LOCKS command.

CLEAR-BASIC-LOCKS

The CLEAR-BASIC-LOCKS verb resets the 64 execution locks (0-63) that may be set by DATA/BASIC and PROC.

:INIT-SYSTEM

The :INIT-SYSTEM unlocks all locked groups and resets all execution locks.

```
:LIST-LOCKS <cr>  <-------- Display lines that have groups locked.  Show
                           FID of first frame in group.

09 17319
15 31712
07 15334
```

Figure A. Sample Usage of LIST-LOCKS Verb

11.35 Verification of Assembly Language Routines from Tape

A copy of SEQUEL system software may be kept in T-DUMP format on tape for periodic system verification. This frees space which would otherwise be needed for the SYSTEM-OBJECT file.

To copy assembly language routines to tape, the following procedure is suggested:

A. Upon receipt of a full system update tape:

1. Load the assembly language verification file as directed by the updating instructions.

2. Dump (ST-DUMP) SYSTEM-OBJECT by frame.

3. Copy the modes necessary for coldstart tape sections to a temporary file by entering:

```
:FORM-LIST SYSTEM-OBJECT COLD-LIST <cr>

xx ITEMS SELECTED.
:COPY SYSTEM-OBJECT <cr>
TO:(DICT %TEMP-SO% 13,1) <cr>

:COPY SYSTEM-OBJECT COLD-LIST <cr>
TO:(DICT %TEMP-SO%) <cr>
```

4. Delete the data section of the SYSTEM-OBJECT file.

5. Recreate the data section of the SYSTEM-OBJECT file with a smaller modulo and copy the coldstart modes by entering:

```
:COPY DICT %TEMP-SO% * <cr>
TO:(DATA SYSTEM-OBJECT 13,1) <cr>
```

6. Delete the temporary %TEMP-SO% file.

7. Put the tape away and ensure that it will never be overwritten.

B. If the system includes additional assembly language routines:

1. Dump and then delete from the disc the additional modes.

2. Be sure to indicate the contents of the tape.

C. As additional updates are received:

1. T-LOAD the latest SYSTEM-OBJECT tape (it may be necessary to temporarily copy some files to tape to allow enough space).

2. Follow the update instructions accompanying the update.

3. Create a new system tape as outlined in step A above.

D. To verify the system:

1. Mount the tape(s) created above.

2. Enter MT-VERIFY.

To verify a system from a SYS-GEN tape:

1. Use the SPACE PROC on SYSPROG to position the tape at the start of the SYSTEM-OBJECT file.

2. Enter MT-VERIFY. Note that if patches are made, mismatches in effected modes will occur.

12 TERMINAL INDEPENDENT PROCESS HANDLER (TIPH)

12.1 Overview

A "process" on a SEQUEL system is defined as the session which occurs between logging on and logging off. Each process is associated with a line (or port), and (normally) each line is associated with a terminal. The Terminal Independent Process Handler (TIPH) initiates a process on a port without an associated terminal. Any terminal output (such as error messages, logon/logoff messages, etc.) is placed in a spooler hold file, the hold file number being assigned by the spooler processor. TIPH may be used for those tasks which do not require operator responses, such as file-saves, massive updates, etc.

TIPH provides a kind of foreground/background capability that allows processes to run without the need of a terminal. "Resident" programs, such as a task scheduler or a transaction logger, may use TIPH. TIPH increases the efficiency of the system by freeing the terminals for their intended purpose, which is user interaction.

A TIPH process is like any other process, except that it is not associated with a terminal. It has its own work space, and has available to it all of the resources of the SEQUEL system, with the exception of terminal input. You cannot access the system through a terminal after the line associated with it has been logged on by a TIPH process.

Although terminal input is not allowed, you may "stack" input in the command stream sent to the TIPH processor. This allows programs to be run which require operator input, where the input responses are known in advance. The processor passes one command string to each input required by the program (this is similar to the secondary input buffer used by the PROC processor).

Once activated, a TIPH process takes its place in the process schedule queue. Because interactive processes are given priority, TIPH processes have low priority by default.

Because a TIPH process uses its own PCB, work space, etc., its port number must be defined on the system, i.e., the port allocated to the TIPH process must be within the configuration of the system.

Certain lines may be designated as TIPH-only lines by the system hardware. These lines may be displayed by entering the WHAT verb from TCL. Normal lines (those normally associated with a terminal) may be designated as being available for use by TIPH by using the PH-ALLOCATE verb.

Note: Your account must have at least 27 additional frames of work space (specified in the System Dictionary) to initiate a TIPH process. You may log the TIPH port onto an account other than the current user account. In this case you must have System 2 (SYS2) privileges.

A PH-HISTORY file exists in the System Dictionary, which contains all of the information concerning each TIPH process. This information is dynamic as the process continues, displaying the status of the TIPH process(es).

The LOGOFF verb will not deactivate (logoff) a TIPH process.

TIPH Verbs

The TIPH verbs and PROCs are summarized in Figure A. The remaining topics in this chapter describe each in detail.

PH-ALLOCATE	*Allocates a port to be available for use by TIPH.*
PH-DELETE	*Designates an allocated port as no longer being available for use by TIPH.*
PH-KILL	*Aborts the designated TIPH process.*
PH-LINES	*Lists the lines currently available on which to activate a TIPH process.*
PH-RESUME	*Reactivates a temporarily halted TIPH process.*
PH-START	*Initiates a TIPH process.*
PH-STATUS	*Displays the current status of all TIPH processes listed in the PH-HISTORY file.*
PH-SUSPEND	*Temporarily halts a TIPH process.*

Figure A. Summary of TIPH Verbs and PROCs

12.2 Port Assignment: PH-ALLOCATE, PH-DELETE, PH-LINES

Certain ports may be designated as TIPH-only ports by the system hardware. These are always allocated to TIPH. Lines which are normally associated with a terminal may also be used by TIPH. Before a normal line may be used by a TIPH process, it must be listed in a table of available TIPH ports. The PH-ALLOCATE verb enters the designated line into the table. The PH-DELETE verb removes it from the table. The PH-LINES verb lists the lines available for use by a TIPH process.

PH-ALLOCATE

The format of the PH-ALLOCATE verb is:

```
PH-ALLOCATE {line#}
```

where "line#" is the number of the line to be allocated. If this number is not entered, it will be prompted for (Figure B). The entered port number must be within the range set by the particular system's hardware configuration (this may be displayed by entering the WHAT verb from TCL). The line remains allocated until either the PH-DELETE verb is used, or the system undergoes a coldstart ('X' option).

TIPH-only lines (as designated by the hardware) need not be allocated via the PH-ALLOCATE verb.

Note: A normal line designated for use by TIPH via PH-ALLOCATE may still be used as a normal terminal line when a TIPH process is not running on it. A TIPH process may not be initiated on a line already logged on or in use by another TIPH process.

PH-DELETE

The format of the PH-DELETE verb is:

```
PH-DELETE {line#}
```

If "line#" is not entered, the system prompts for the line to be deleted from the TIPH list. After the line is deleted, the line is no longer available for use by TIPH, unless it is reallocated with the PH-ALLOCATE verb.

PH-LINES

The format of the PH-LINES verb is:

```
PH-LINES
```

This verb lists all the lines currently available for use by TIPH. If an allocated line is already in use by TIPH (or any other process), it is not displayed here.

```
PH-ALLOCATE {line#}

PH-DELETE {line#}

PH-LINES
```

Figure A. General Form of PH-ALLOCATE, PH-DELETE, and PH-LINES Verbs

```
:PH-ALLOCATE <cr>                    <-- Evoke the PH-ALLOCATE verb.

INPUT LINE TO ALLOCATE? 1 <cr>       <-- Designate that Line 1 may be used by a
                                         TIPH process.

:PH-DELETE <cr>                      <-- Evoke the PH-DELETE verb.

INPUT LINE TO DELETE? 22 <cr>        <-- Designate Port 22 as no longer being
                                         available for use by TIPH.
LINE DELETED FROM PHANTOM SERVICE.

:PH-LINES <cr>                       <-- Display the lines currently available
LINES AVAILABLE FOR TIPH USE             for use by a TIPH process. In this
1                                        example, Ports 1, 11, and 16 are avail-
11                                       able.
16

3 ITEMS LISTED.
```

Figure B. Example of Correct Usage of PH-ALLOCATE, PH-DELETE, and PH-LINES Verbs

12.3 Initiating a TIPH Process: The PH-START Verb

A TIPH process is initiated by evoking the PH-START verb from TCL. The TIPH process begins by logging on the specified line. After executing all the commands in the command stream, the line is logged off. All terminal output, including the logon and logoff messages, is placed in a hold file. The command stream may contain any TCL verb except LOGTO.

The format of the PH-START verb is:

PH-START

The TIPH processor prompts for a task-id, the desired line number, any desired options, an account name, an optional password, and the command(s) to be executed (Figures A and B).

The task-id is a user-selected alphanumeric character string which identifies the TIPH process in the PH-STATUS display (see the topic DISPLAYING THE TIPH PROCESSES: THE PH-STATUS VERB). It is not the item-id of the item in the PH-HISTORY file. It may be up to 140 alphanumeric characters; however, only the first 10 are displayed.

The line number must be either a TIPH-only line or have been allocated (via PH-ALLOCATE) for use by a TIPH process prior to evoking the PH-START verb. If the line cannot be used by TIPH, the message "NO LINE AVAILABLE" is displayed. If another user (or anotner TIPH process) is already logged on to the specified line, the message "LINE ALREADY LOGGED ON" appears. Note that if another TIPH process is logged to the line, TIPH will not queue the current user's TIPH process to wait for it to finish (logoff).

The available options are explained in the topic PH-START OPTIONS.

The account name you enter is the account the TIPH process will logon to execute the command stream. The password is that specified in the System Dictionary for the particular account. If there is no password, then just press <RETURN>.

The command stream may consist of TCL statements, a PROC name, or inputs necessary to perform the desired tasks. Note that the verb LOGTO cannot be used. Each command (or input value) is entered one at a time, separated by pressing <RETURN>. The verb "OFF" should be the last entry in the command stream.

If an operation is to be performed requiring operator input (such as a DATA/BASIC program which contains INPUT statements), the input value(s) must be included in the command stream. Each is entered individually, separated by <RETURN>. Note: To decrease the number of commands entered through TIPH, it is recommended that you create a PROC prior to using TIPH, and then evoke this PROC in the command stream.

```
PH-START
```

Figure A. Format of the PH-START Verb

```
:PH-START <cr>
```
Evoke the TIPH processor.

```
ENTER TASKID FOR PH TASK? TEST <cr>
```
The task-id identifies the TIPH process in the PH-HISTORY file. For this example, the task-id is 'TEST'.

```
ENTER LINE# FOR PH TASK? 14 <cr>
```
This TIPH process will logon to Line 14 to execute the command string. If this line has not been allocated to be used by TIPH, or if the associated terminal is already in use, an error message appears, and the process is aborted.

```
ENTER OPTIONS? <cr>
```
The options are listed in the topic PH-START OPTIONS. For this example no options are specified (i.e., all terminal output goes to a hold file).

```
ENTER ACCOUNT FOR PH JOB? INV <cr>
```
The TIPH process logs on to the INV account to execute the command string.

```
ENTER PASSWORD? KRIS <cr>
```
The INV account's password is KRIS.

```
ENTER COMMAND STREAM <CR TO END>?
>SP-ASSIGN HS <cr>
>SORT INVENTORY (P) <cr>
>OFF <cr>
><cr>
```
This TIPH process performs a SORT on the INVENTORY file. The output is printed to a hold file (separate from the hold file containing the TIPH logon and logoff messages). The TIPH process ends by logging off the line.

```
TASK TEST STARTED ON LINE 14 FOR
ACCOUNT INV
```

Figure B. Example of Correct Usage of TIPH

12.4 PH-START Options

Normally, all terminal output for a TIPH processs is passed to a hold file, so that you may print the output when the process is completed. Certain options are available to either prevent the creation of the hold file or to print the hold file automatically when the TIPH process is complete. You may also hush the TIPH processor message which appears on the "master" terminal.

The following options are available:

H Hush the message displayed by the PH-START verb when the TIPH process has been started. This message is normally displayed on the terminal which started the TIPH process. The option may be useful if the TIPH process itself starts another TIPH process, to prevent the message from being passed into the hold file. The 'H' option is not shown in the TIPH status display (see the topic DISPLAYING THE TIPH PROCESSES: THE PH-STATUS VERB), because it pertains to the "master" process, rather than to the TIPH process.

P Print the hold file on the printer after the TIPH process has ended. The result is the same as if you had printed the hold file using SP-EDIT (see the topic EXAMINING JOBS WAITING TO BE PRINTED in Chapter 8). The hold file contains all terminal output generated by the TIPH process (such as the logon and logoff messages, the responses from the system after a SELECT, etc.). Note that the hold file is not deleted by this option.

N Do not maintain a hold file for terminal output. This option results in all terminal output being "lost". This may be useful when executing update programs, file-saves, etc., when the hold file would only contain messages not deemed important for a particular application.

T Display all terminal output on the terminal associated with the line. This may be used, for example, to print a report on a hardcopy terminal. If there is no associated terminal, or if the process is executing on a TIPH-only line, the TIPH process will "hang" until the process is aborted. No hold file is created because the output is to the terminal.

Because the logon and logoff messages are also sent to the terminal, you are cautioned that, if the line is associated with a hardcopy terminal, the messages may appear on the same page as the report.

The options 'HN', 'HP', and 'HT' are also available; they have the same effect as the 'N', 'P' and 'T' options (respectively), but also cause the PH-START message to be hushed. Even though no comma is placed between the 'H' and the other options, the 'H' will not appear in the PH-STATUS display.

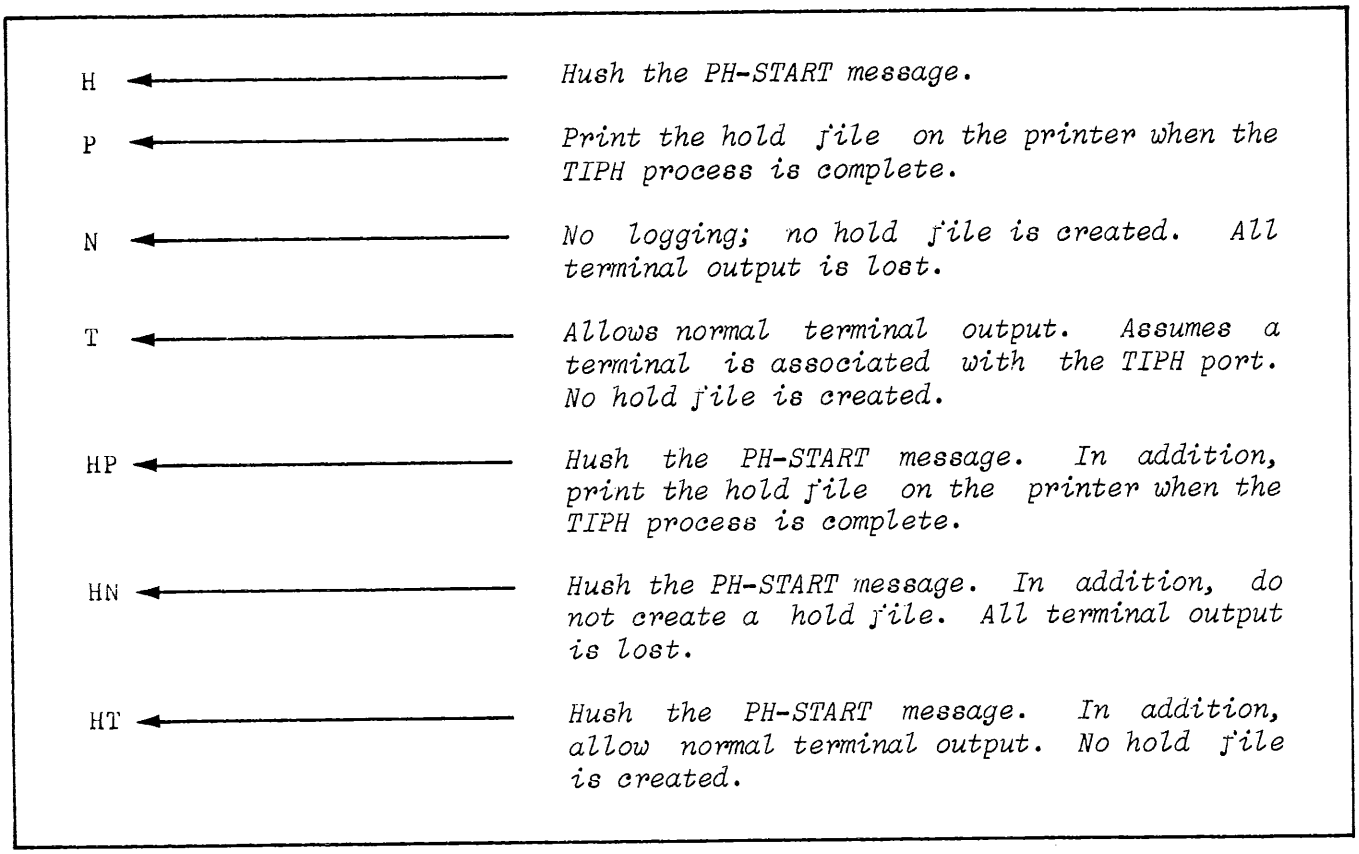


Figure A. PH-START Verb Options

12.5 Halting and Reactivating a TIPH Process

A TIPH process may be aborted or temporarily halted using the PH-KILL and the PH-SUSPEND verbs, respectively. A temporarily halted process may be reactivated using the PH-RESUME verb.

The general form of the three verbs is:

PH-KILL task-id line#

PH-SUSPEND task-id line#

PH-RESUME task-id line#

where "task-id" is your specified identification of the TIPH process, and "line#" is the line on which the TIPH process is running. An asterisk (*) in place of the task-id implies the process running on the specified line. An asterisk in place of the line# implies all processes with the specified task name. Any of the three verbs followed by a single asterisk implies all TIPH processes (the asterisk following PH-RESUME reactivates all temporarily halted processes).

PH-KILL causes the specified process to abort (logoff). Any updating being performed will not be undone. If the TIPH process (or processes) is currently running on an account other than your own, System 2 privileges are required.

PH-SUSPEND temporarily halts a TIPH process. This may be useful if additional processing resources are required for another line (i.e., to speed up its process). By suspending the TIPH process(es), more time is available for the other jobs. System 2 privileges are required if the TIPH process (or processes) is executing on an account other than your own.

Note: The TIPH process must already be active (beyond logging on) before the PH-SUSPEND verb may be used. Otherwise, it may "hang" the line. You must coldstart ('X' option) the system to release the line (see the topic COLDSTART). Because the remainder of the system is unaffected, it is recommended that you wait until all other users are logged off of the system to avoid inconvenience and possible group format errors.

PH-RESUME reactivates a TIPH process which had been temporarily halted with the PH-SUSPEND verb. System 2 privileges are required, as noted above. Note that a TIPH process may be reactivated from any terminal, not just the terminal from which it was halted.

```
PH-KILL task-id line#

PH-SUSPEND task-id line#

PH-RESUME task-id line#
```

Figure A. General form of the PH-KILL, PH-SUSPEND, and PH-RESUME Verbs

12.6 Displaying the TIPH Processes: The PH-STATUS Verb

PH-STATUS displays the status of a TIPH process as it is recorded in the PH-HISTORY file.

The general form of PH-STATUS is:

PH-STATUS {task-id line#} {(P)}

where "task-id" is the user-specified name of the TIPH process, and "line#" is the line to which the process was assigned. If an asterisk (*) is entered in place of "task-id", then all processes in the PH-HISTORY file which ran on the specified line are displayed (see note below). If an asterisk is entered in place of the line number, then all processes with the specified name are displayed. PH-STATUS with no task-id or line# specified displays the status of all processes in the PH-HISTORY file. The 'P' option specifies printing the status on the printer.

PH-STATUS displays the following information: the task-id, the line number, the account to which the TIPH process is logged, the name of the account which started the process, the date and time started, the current status of the process, the PH-START options (if any), and the number of the hold file which contains the terminal output (Figure B).

The status of each TIPH process is indicated by displaying an alphabetic status code. 'A' indicates that the process is still active; 'S' indicates that the process has been suspended; 'T' indicates that the process has terminated normally; 'X' indicates that the process was aborted by the process itself (abnormal abort), 'K' indicates that the process was aborted by the PH-KILL verb.

'OUTFL' on the PH-STATUS display is the output file, i.e., the spooler hold file to which the terminal output has been sent. Note: If the process printed something to a hold file (using SP-ASSIGN HS), this hold file number is not displayed. The displayed hold file number contains the output which normally would have appeared on the terminal. There is no terminal output file if the 'N' option was specified with the PH-START verb.

Note: Items in the PH-HISTORY file are not automatically deleted. Therefore, when displaying all TIPH processes, many processes may be shown with the same line number and name. The data portion of the file should be cleared occasionally, using the CLEAR-FILE verb, as shown in Figure C. Do not clear the file while a TIPH process is still active!

```
PH-STATUS {task-id line#} {(P)}
```

Figure A. General Form of the PH-STATUS Verb

```
TASK-ID... LINE ACCOUNT... STARTER-ACC DATE..... TIME.... STATUS OP OUTFL#

LGRADE      4   REG         ADM         14 JUN 1980  08:05:37      K  T
FS          7   SYSPROG     DOC         21 JUN 1980  17:30:17      A  P    023
CMP.BP      12  CHRIS       CHRIS       07 JUN 1980  12:05:11      S  P    002
S.INV       17  INV         MASTER      05 JUN 1980  17:00:57      T       003
T3          18  JS          SYSPROG     11 JUN 1980  09:13:45      X  N

    where the status code is one of the following:

    A = Active
    S = Suspended
    T = Normal termination
    X = Abort by process
    K = Abort by PH-KILL

    Note that the 'H' option is not displayed.
```

Figure B. PH-STATUS Display

```
:CLEAR-FILE (DATA PH-HISTORY) <cr>
```

Figure C. Example of Clearing the Data Portion of the PH-HISTORY File

12.7 TIPH Initialization Messages

Initialization messages are mainly used to indicate errors in the specification of a TIPH process. They may appear while using the PH-START verb.

The following messages exist:

NO LINE AVAILABLE - If you did not specify a line, this message indicates that all the TIPH lines (both TIPH-only and those normally associated with a terminal) are in use. If a line was specified, it indicates that the specified line was not allocated for use by TIPH (using the PH-ALLOCATE verb). You should allocate a line, if necessary, and reenter the PH-START verb.

PH-PWD ERROR - The incorrect password was entered for the specified account. You should reenter the PH-START verb.

PH-ID ERROR - No user account was specified. You should reenter the PH-START verb.

HISTORY FILE PROBLEM - No PH-HISTORY file exists, or the current account does not have a Q-pointer to it. If the file does not exist, you may create it using the CREATE-FILE verb and the EDITOR. The procedure is described in the Appendix at the back of this manual. If no Q-pointer exists, then you should create it using the EDITOR (Figure B).

LINE ALREADY LOGGED ON - The specified line is being used either by an actual user or another TIPH process. You may enter the PH-LINES verb to display the currently available lines.

HISTORY FILE ACCESS ERROR - For some reason, TIPH cannot properly access the PH-HISTORY file.

MASTER PRIVILEGE ERROR - You attempted to use TIPH on an account other than your own, and did not have SYS2 privileges. If you are able, you should log to the SYSPROG account and increase your privilege level.

ACC FILE ACCESS ERROR - For some reason TIPH cannot access the ACC file to indicate which lines have been logged on.

[82] YOUR SYSTEM PRIVILEGE LEVEL IS NOT SUFFICIENT FOR THIS STATEMENT - You attempted to initiate a TIPH process and your account only had System 0 (SYS0) privileges.

NOT ENOUGH WORKSPACE - You attempted to initiate a TIPH process and your account did not have at least 27 additional frames of work space assigned. If you are able, you should log to the SYSPROG account and increase the account's work space assignment.

NO LINE AVAILABLE ←	*Either a line number was not entered, or the specified line was not allocated to be used by a TIPH process (using PH-ALLOCATE).*
PH-PWD ERROR ←	*The entered password was incorrect for the specified account.*
PH-ID ERROR ←	*No user account was specified.*
HISTORY FILE PROBLEM ←	*Either the PH-HISTORY file is missing from the SYSTEM dictionary, or a Q-pointer to it does not exist in the current account's M/DICT.*
LINE ALREADY LOGGED ON ←	*The specified line is in use by either an actual user or another TIPH process.*
HISTORY FILE ACCESS ERROR ←	*TIPH cannot access the PH-HISTORY file.*
MASTER PRIVILEGE ERROR ←	*You attempted to log the TIPH process onto an account other than your own, and your account did not have System 2 (SYS2) privileges.*
ACC FILE ACCESS ERROR ←	*TIPH cannot access the ACC file.*
[82] YOUR SYSTEM PRIVILEGE LEVEL IS NOT SUFFICIENT FOR THIS STATEMENT ←	*Your account has only System 0 privileges.*
NOT ENOUGH WORKSPACE ←	*Your account does not have at least 27 additional frames of workspace assigned.*

Figure A. Initialization Error Messages

```
PH-HISTORY        ---in user account's M/DICT
001 Q
002 SYSTEM
003 PH-HISTORY
004               ---null
005               ---null
006               ---null
007               ---null
008               ---null
009 L
010 10
```

Figure B. Q-Pointer to PH-HISTORY File

12.8 TIPH Termination Messages

The TIPH Termination Messages are placed in the hold file along with the other terminal output. These messages indicate the reason the process halted.

The following are the Termination Messages:

T PROCESS STOP - Indicates that the TIPH process has completed normally and has logged off the line.

T PROCESS HAS BEEN LOGGED OFF BY PH-KILL - Indicates that the TIPH process has been aborted by the PH-KILL verb.

T PROCESS ABORT - Indicates that the TIPH process has aborted due to some reason related to the process itself (i.e., not caused by TIPH). For example, a TIPH process executing a DATA/BASIC program which aborted due to a programming error. This message may be preceded by one of the messages below.

E INPUT ERROR - Indicates that a process prompted for terminal input, and there was no TIPH stacked input available. This may occur if: 1) a TCL verb which requires user input (such as SSELECT, or COPY) was inadvertently used without specifying the required subsequent responses in the command string, or 2) the verb "OFF" was not the last command in the command stream.

E PROCESS IN DEBUGGER - Indicates that the process has trapped to the system debugger. The TIPH process will abort with the *T* PROCESS ABORT message.

[910] *T* PROCESS STOP ←	*TIPH process terminated normally.*
[911] *T* PROCESS HAS BEEN LOGGED OFF BY PH-KILL ←	*TIPH process terminated by PH-KILL.*
[912] *T* PROCESS ABORT ←	*TIPH process aborted due to an error by the process being executed (such as a DATA/BASIC program).*
[913] *E* INPUT ERROR ←	*The TIPH process prompted for terminal input, and there were no responses in the command string.*
[914] *E* PROCESS IN DEBUGGER ←	*The TIPH process has trapped to the system debugger.*

Figure A. TIPH Termination Messages

APPENDIX A. ASCII CHARACTER SET

Decimal	Hex	EBCDIC Equiv.	ASCII Char.	PRISM II Display	PRISM II Key	Special Use
0	00	00	NUL	None	<c>@	Delay character
1	01	01	SOH	None	<cs>A	PRISM home command
2	02	02	STX	None	<cs>B	
3	03	03	ETX	None	<cs>C	End of text
4	04	37	EOT	None	<cs>D	
5	05	2D	ENQ	None	<cs>E	
6	06	2E	ACK	None	<cs>F, <s>6	Cursor forward on PRISM
7	07	2F	BEL	None	<cs>G	Bell of PRISM
8	08	16	BS	None	<cs>H	Backspace on PRISM
9	09	05	HT	None	<cs>I	Tab
10	0A	25	LF	None	<cs>J, <s>2	Cursor down on PRISM
11	0B	0B	VT	None	<cs>K	Vertical address on PRISM
12	0C	0C	FF	None	<cs>L	Screen erase on PRISM
13	0D	0D	CR	None	<cs>M	Carriage return
14	0E	0E	SO	None	<cs>N	
15	0F	0F	SI	None	<cs>O	Sort key delimiter
16	10	10	DLE	None	<cs>P	Horizontal address on PRISM blank compression character
17	11	11	DC1	None	<cs>Q	
18	12	12	DC2	None	<cs>R	Retype entire line; enable slave printer
19	13	3A	DC3	None	<cs>S	Dump PRISM screen to slave printer (option)
20	14	3C	DC4	None	<cs>T	Disable slave printer
21	15	3D	NAK	None	<cs>U, <s>4	Cursor back on PRISM
22	16	32	SYN	None	<cs>V	
23	17	26	ETB	None	<cs>W	
24	18	18	CAN	None	<cs>X	Cancel line
25	19	19	EM	None	<cs>Y	
26	1A	3F	SUB	None	<cs>Z, <s>8	Cursor up on PRISM
27	1B	27	ESC	[	ESC, <c>[	EDITOR command delimiter. Envokes SCREENPRO command-mode
28	1C	1C	FS	None		
29	1D	1D	GS	None		
30	1E	1E	RS	None		
31	1F	1F	US	None		
32	20	40	blank		space	
33	21	5A	!	!	!	
34	22	7F	"	"	"	String delimiter in ENGLISH and BASIC
35	23	7B	#	#	#	
36	24	5B	$	$	$	
37	25	6C	%	%	%	
38	26	50	&	&	&	
39	27	7D	'	'	', <c>"	String delimiter in ENGLISH and BASIC

Decimal	Hex	EBCDIC Equiv.	ASCII Char.	PRISM II Display	PRISM II Key	Special Use
40	28	4D	(	(	(	
41	29	5D	)	)	)	
42	2A	5C	*	*	*	
43	2B	4E	+	+	+	
44	2C	6B	,	,	, or <c><	
45	2D	60	-	-	-	
46	2E	4B	.	.	., <c>>	
47	2F	61	/	/	/, <c>?	
48	30	F0	0	0	0, <c>)	
49	31	F1	1	1	1, <c>!	
50	32	F2	2	2	2	
51	33	F3	3	3	3, <c>#	
52	34	F4	4	4	4, <c>$	
53	35	F5	5	5	5, <c>%	
54	36	F6	6	6	6	
55	37	F7	7	7	7, <c>8	
56	38	F8	8	8	8, <c>*	
57	39	F9	9	9	9, <c>(	
58	3A	7A	:	:	:	
59	3B	5E	;	;	;, <c>:	
60	3C	4C	<	<	<	
61	3D	7E	=	=	=, <c>+	
62	3E	6E	>	>	>	
63	3F	6F	?	?	?	
64	40	7C	@	@	@	
65	41	C1	A	A	<s>A	
66	42	C2	B	B	<s>B	
67	43	C3	C	C	<s>C	
68	44	C4	D	D	<s>D	
69	45	C5	E	E	<s>E	
70	46	C6	F	F	<s>F	
71	47	C7	G	G	<s>G	
72	48	C8	H	H	<s>H	
73	49	C9	I	I	<s>I	
74	4A	D1	J	J	<s>J	
75	4B	D2	K	K	<s>K	
76	4C	D3	L	L	<s>L	
77	4D	D4	M	M	<s>M	
78	4E	D5	N	N	<s>N	
79	4F	D6	O	O	<s>O	
80	50	D7	P	P	<s>P	
81	51	D8	Q	Q	<s>Q	
82	52	D9	R	R	<s>R	
83	53	E2	S	S	<s>S	
84	54	E3	T	T	<s>T	
85	55	E4	U	U	<s>U	
86	56	E5	V	V	<s>V	
87	57	E6	W	W	<s>W	

Decimal	Hex	EBCDIC Equiv.	ASCII Char.	PRISM II Display	PRISM II Key	Special Use
88	58	E7	X	X	<s>X	
89	59	E8	Y	Y	<s>Y	
90	5A	E9	Z	Z	Z	
91	5B	80	[	[	[	String search delimiter
92	5C	E0	\	\	\	RUNOFF control character
93	5D	90	]	]	]	ENGLISH string search delimiter
94	5E	5F	^	^	^	ENGLISH string search delimiter
95	5F	6D				RUNOFF control character
96	60	79	`	@	<cs>0	
97	61	81	a	A	A	
98	62	82	b	B	B	
99	63	83	c	C	C	
100	64	84	d	D	D	
101	65	85	e	E	E	
102	66	86	f	F	F	
103	67	87	g	G	G	
104	68	88	h	H	H	
105	69	89	I	I	I	
106	6A	91	j	J	J	
107	6B	92	k	K	K	
108	6C	93	l	L	L	
109	6D	94	m	M	M	
110	6E	95	n	N	N	
111	6F	96	o	O	O	
112	70	97	p	P	P	
113	71	98	q	Q	Q	
114	72	99	r	R	R	
115	73	A2	s	S	S	
116	74	A3	t	T	T	
117	75	A4	u	U	U	
118	76	A5	v	V	V	
119	77	A6	W	W	W	
120	78	A7	x	X	X	
121	79	A8	y	Y	Y	
122	7A	A9	z	Z	Z	
123	7B	C0	{	[	<c>;	
124	7C	6A	|	\	|	
125	7D	D0	}	]	<c>=	
126	7E	A1	~	^	<c>>	
127	7F	07	DEL	None		
128	80	04		None		
129	81	06		None		
130	82	08		None		
131	83	09		None		
132	84	0A		None		

Decimal	Hex	EBCDIC Equiv.	ASCII Char.	PRISM II Display	PRISM II Key	Special Use
133	85	13		None		
134	86	14		None		
135	87	15		None		
136	88	17		None		
137	89	1A		None		
138	8A	1B		None		
139	8B	20		None		
140	8C	21		None		
141	8D	22		None		
142	8E	23		None		
143	8F	24		None		
144	90	28		None		
145	91	29		None		
146	92	2A		None		
147	93	2B		None		
148	94	2C		None		
149	95	30		None		
150	96	31		None		
151	97	33		None		
152	98	34		None		
153	99	35		None		
154	9A	36		None		
155	9B	38		None		
156	9C	39		None		
157	9D	3B		None		
158	9E	3E		None		
159	9F	41		None		
160	A0	42		None		
161	A1	43		None		
162	A2	44		None		
163	A3	45		None		
164	A4	46		None		
165	A5	47		None		
166	A6	48		None		
167	A7	49		None		
168	A8	4A		None		
169	A9	4F		None		
170	AA	51		None		
171	AB	52		None		
172	AC	53		None		
173	AD	54		None		
174	AE	55		None		
175	AF	56		None		

Decimal	Hex	EBCDIC Equiv.	ASCII Char.	PRISM II Display	PRISM II Key	Special Use
176	B0	57		None		
177	B1	58		None		
178	B2	59		None		
179	B3	62		None		
180	B4	63		None		
181	B5	64		None		
182	B6	65		None		
183	B7	66		None		
184	B8	67		None		
185	B9	68		None		
186	BA	69		None		
187	BB	70		None		
188	BC	71		None		
189	BD	72		None		
190	BE	73		None		
191	BF	74	blank	None		
192	C0	75	@	@		
193	C1	76	A	A		
194	C2	77	B	B		
195	C3	78	C	C		
196	C4	8A	D	D		
197	C5	8B	E	E		
198	C6	8C	F	F		
199	C7	8D	G	G		
200	C8	8E	H	H		
201	C9	8F	I	I		
202	CA	9A	J	J		
203	CB	9B	K	K		
204	CC	9C	L	L		
205	CD	9D	M	M		
206	CE	9E	N	N		
207	CF	9F	O	O		
208	D0	A0	P	P		
209	D1	AA	Q	Q		
210	D2	AB	R	R		
211	D3	AC	S	S		
212	D4	AD	T	T		
213	D5	AE	U	U		
214	D6	AF	V	V		
215	D7	B0	W	W		
216	D8	B1	X	X		
217	D9	B2	Y	Y		
218	DA	B3	Z	Z		

Decimal	Hex	EBCDIC Equiv.	ASCII Char.	PRISM II Display	PRISM II Key	Special Use
219	DB	B4	[	[		
220	DC	B5	\	\		
221	DD	B6	]	]		
222	DE	B7	^	^		
223	DF	B8				
224	E0	B9	@	@		
225	E1	BA	A	A		
226	E2	BB	B	B		
227	E3	BC	C	C		
228	E4	BD	D	D		
229	E5	BE	E	E		
230	E6	BF	F	F		
231	E7	CA	G	G		
232	E8	CB	H	H		
233	E9	CC	I	I		
234	EA	CD	J	J		
235	EB	CE	K	K		
236	EC	CF	L	L		
237	ED	DA	M	M		
238	EE	DB	N	N		
239	EF	DC	O	O		
240	F0	DD	P	P		
241	F1	DE	Q	Q		
242	F2	DF	R	R		
243	F3	E1	S	S		
244	F4	EA	T	T		
245	F5	EB	U	U		
246	F6	EC	V	V		
247	F7	ED	W	W		
248	F8	EE	X	X		
249	F9	EF	Y	Y		
250	FA	FA	Z	Z		System Delimiters:
251	FB	FB	[	[		Start Buffer (SB)
252	FC	FC	\	\	<c>\	Subvalue Mark (SVM)
253	FD	FD	]	]	<c>]	Value Mark (VM)
254	FE	FE	^	^	<c>^	Attribute Mark (AM)
255	FF	FF	_	_	<c>_	Segment Mark (SM)

NOTE: The <SHIFT> key may have to be pressed on some terminals if the desired character appears on the top half of a key top. For example, to generate an attribute mark (X'FE') on some terminals, it is necessary to press and hold the <SHIFT> key to generate the "^" character, while pressing the <CTRL> key to cause the character to be a control character.

APPENDIX B. SYSTEM ERROR MESSAGES

Message Number	Message
1	ILLEGAL USE OF DOUBLE-QUOTE IN ITEM-ID
2	UNEVEN NUMBER OF SINGLE OR DOUBLE QUOTE-SIGNS ('")
3	VERB?
4	MODE'mode'CHECKSUM ERROR; FRAME = n MODE = cccc ABD = aaaa
5	THE WORD 'word' IS ILLEGAL
6	FILE NAME 'file name' IS ILLEGAL.
7	A VALUE MUST FOLLOW THE HEADING, FOOTING, TAG OR GRAND-TOTAL CONNECTIVE.
8	A WINDOW SPECIFICATION STRING MUST FOLLOW THE "WINDOW" CONNECTIVE.
9	SYSTEM DL/ID MISSING.
10	FILE NAME MISSING.
11	FRAME LOCKED AT LOCATION X 'nnnn'
12	POWER FAILURES HAVE OCCURRED SINCE COLD-START
13	DATA LEVEL DESCRIPTOR MISSING
14	TAPE INPUT ILLEGAL
15	THE FILE-NAME IS PRECEDED BY AN ILLEGAL CONNECTIVE.
17	"WITHIN" VALID ONLY IN COUNT/LIST STATEMENTS.
18	LAST WORD MAY NOT BE CONNECTIVE
19	VALUE WITHOUT AN ATTRIBUTE NAME IS ILLEGAL.
20	n MOS MEMORY ERRORS
21	CONFLICT BETWEEN USER & SYSTEM NAMES
22	MEMORY CONTENTS: n MONITOR BUFFERS n ABS FRAMES n WORK SPACE FRAMES n USER PROGRAM DATA n I/O BUSY FRAMES n MEMLOCKED BUFFERS n WRITE REQUIRED BUFFERS
24	THE WORD 'word' CANNOT BE IDENTIFIED
25	"WITH" MAY NOT IMMEDIATELY PRECEDE A VALUE
26	ATTRIBUTE VALUES MAY NOT BOTH PRECEDE AND FOLLOW AN ATTRIBUTE NAME.
27	SEQUEL n SYSTEM SERIAL NUMBER nK OF MEMORY n ABS FRAMES n PORTS n WORDMATE n AVAILABLE DEVICES DEVICE UNIT(S)
28	NO CONFIGURATION CHIP.
29	AT LEAST ONE ITEM-ID MUST BE SPECIFIED FOR A "WITHIN"-TYPE STATEMENT.
30	FORMAT ERROR IN M/DICT ENTRY DEFINING VERB.
32	**A REQUIRED ITEM-ID IS MISSING.
33	n SYSTEM PRIVILEGE LEVEL n ADDITIONAL FRAMES OF WORK SPACE

42 -ILLEGAL MULTIPLE CONNECTIVES EXIST-
43 ERROR - RECEIVE OR TRANSMIT FUNCTION NOT SPECIFIED
44 ERROR - FORMAT SPECIFICATION INCORRECT
46 ERROR - INCORRECT DEVICE TYPE
47 ERROR - RECEIVE ID 'n' SPECIFIED
48 ID 'n' ENTERED IN TABLE
49 RECEIVE ENTRY 'n' COMPLETED
50 OPERATOR REQUESTED ABORT ON TRANSMIT ENTRY 'n'
51 TRANSMIT ENTRY 'n' COMPLETED
53 ERROR - TRANSMIT ID SPECIFIED
54 WAIT TIME FOR RECEIVE ENTRY 'n' MATURED - FUNCTION TERMINATED
55 LOCAL LINE-PRINTER SET OPERABLE
57 ID 'n' DELETED FROM TABLE
58 ID 'n' IS NOT IN TABLE
59 TRANSMIT ENTRY 'N' ABORTED
60 LOCAL TASK REJECTED - NO LOCAL PRINTER DEFINED
61 ERROR - INCORRECT OPTION SPECIFICATION
62 ID ENTRY IN TABLE FILE MISSING
63 ERROR - INVALID RECEIVE OPERATION SPECIFIED
64 ONLY ONE ITEM ID IS PERMITTED IN A "WITHIN" TYPE STATEMENT
65 SEQUEL - DATA SET NOT READY
67 TOO MANY IDS SPECIFIED - (MAXIMUM OF 10)
71 AN ILLEGAL CONNECTIVE MODIFIES THE WORD 'word'
72 THE VALUE 'value' IS MEANINGLESS.
74 ABORT ON RECEIVE ENTRY 'n' - FUNCTION TERMINATED
75 SEQUEL - 2780PLUS
DATA SET NOT READY
76 SEQUEL - 2780PLUS
79 THE NUMBER OF SEPARATE AND CLAUSE SETS CANNOT EXCEED 9.
80 A SYSTEM ERROR HAS OCCURRED IN MODE: mode THIS MAY BE DUE TO SORT KEY(S) PRECEDING SELECTION CRITERIA.
81 SECURITY CODE VIOLATION
82 YOUR SYSTEM PRIVILEGE LEVEL IS NOT SUFFICIENT FOR THIS STATEMENT.
86 FILE REFERENCE ATTEMPTED ON FILE NOT PREVIOUSLY OPENED
90 TAPE IS NOT T-DUMP FORMAT
94 END OF FILE
96 BOT
97 END OF TAPE
98 TAPE BUFFER SIZE OF n EXCEEDED.
99 TAPE RECORDS WRITTEN.
100 ASYNC DEVICE DETACHED
101 ASYNC DEVICE ATTACHED
111 ITEM 'item-id' IS NOT ON FILE
112 NO ENTRIES IN TRACE BUFFER
113 FORMAT ERROR IN TRACE BUFFER
117 A DELETE STATEMENT MUST CONTAIN EITHER ITEM-IDS OR SELECTION CRITERIA
120 'value' NEGATIVE BALACE NOT PERMITTED
136 'value' DOES NOT MATCH THE G-CORRELATIVE SPECS.
151 LINE FID PS RETURN STACK
158 AN ILLEGAL CONNECTIVE OF THE FORM 'x' MODIFIES 'value'
163 ATTRIBUTE FOR SORT-KEY MISSING

164 A D1 OR D2 ATTRIBUTE HAS CORRELATIVES THAT ILLEGALLY PRECEDE THE D-CORRELATIVE.
166 ILLEGAL ATTRIBUTE NAME-- 'name'
167 NON-NUMERIC AMC
168 ILLEGAL CHARACTER IN A-CORRELATIVE.
169 ILLEGAL A-CORRELATIVE
170 MISSING LEFT PAREN IN A-CORRELATIVE
171 MISSING RIGHT PAREN IN A-CORRELATIVE.
172 MISSING RIGHT BRACKET IN A-CORRELATIVE
173 MISSING SEMI-COLON IN A-CORRELATIVE
175 END OF TAPE....
180 MESSAGE NOT IN QUEUE
181 MESSAGE 'nnnn' ENTERED IN QUEUE
182 DATA LINK CONTROL CHARACTERS DELETED FROM TEXT
183 END OF QUEUE
199 INSUFFICIENT WORK SPACE FOR ITEM 'item-id'.
200 FILE NAME?
201 'file name' IS NOT A FILE NAME
202 'item-id' NOT ON FILE.
203 ITEM NAME?
204 FILE DEFINITION 'file-name' IS MISSING.
205 NO STATEMENTS TO BE ASSEMBLED.
206 'item-id' ASSEMBLED.
207 UNDEFINED SYMBOLS.
208 ERROR IN ITEM-ID LIST
209 'POINTER-FILE' MISSING A 'B' IN THE 13TH ATTR. CONTACT THE SYSTEM MANAGER OR PROGRAMMER.
210 FILE 'file-name' IS ACCESS PROTECTED.
211 MODE 'item-id' NO ASSEMBLED CODE CAN BE FOUND.
212 MODE 'item-id' HAS MISSING OR ILLEGAL FRAME STATEMENT
213 MODE 'item-id' LOCATION COUNTER ERROR AT LINE NO. n
214 MODE 'item-id' EXCEEDS MAX. OF 512 BYTES AT LINE NO. n
215 MODE 'item-id' HEX ERROR AT LINE NO. n
216 MODE 'item-id' LOADED; FRAME = n SIZE = n CKSUM =n
217 MODE 'iten-id' VERIFIED; FRAME = n SIZE = n CKSUM = n
218 MODE 'item-id' FRAME = n HAS n MISMATCHES
219 ILLEGAL CAMMAND: command
220 -EXITED-
221 'item-id' FILED.
222 'item-id' DELETED.
223 'item' EXISTS ON FILE
224 VERIFIED
225 CATALOGED PROGRAM HAS n MISMATCHES
226 TAPE FORMAT ERROR
229 'account-name' IS NOT A VALID ACCOUNT NAME.
230 CARD READER NOT READY
231 CARD READER MECHANICAL ERROR
232 CARD READER EBCDIC ERROR
233 CARD READER HOPPER EMPTY
234 ITEM SIZE EXCEEDS 32,000 BYTES
240 A "SELECT" OR "SSELECT" MUST BE USED PRECEDING A "SAVE-LIST" STATEMENT!
241 'item-id' CATALOGED; n FRAMES USED.

242 'item-id' DELETED.

260 BATCH-STRING LOCK CODE ELEMENT ERROR AT: n

267 PROC XFER TO 'item-id ' CANNOT BE COMPLETED

268 THE DESTINATION OF THE PROC "GO" STATEMENT: statement, CANNOT BE FOUND

270 FORMAT ERROR IN THE PROC STATEMENT : STATEMENT

272 A VALUE EXISTS FOR THE ATTRIBUTE REFERENCED BY THE ELEMENT: element

273 ERROR IN COLUMN-NUMBER/FIELD-WIDTH OR FORMAT SPECIFICATION AT: statement

274 UNRECOGNIZABLE BATCH-STRING ELEMENT : element

275 Y OR F SUB-ELEMENT ERROR AT BATCH-STRING ELEMENT : element

276 D-2 UPDATE WITHOUT D-1 BEING SPECIFIED, AT BATCH-STRING ELEMENT : element

277 J ELEMENT MISSING AT BATCH-STRING ELEMENT : element

278 ERROR IN PROCESSING SECONDARY BATCH-STRING ELEMENT : element

279 INCORRECT SCALING FACTOR IN F* BATCH-STRING ELEMENT : element

280 FILE-DEFINITION BATCH ELEMENT ERROR AT : element

281 D1 MUST HAVE Y11 STORAGE CORRELATIVE... ERROR AT : element

282 DATA INPUT LINE TO BATCH AFTER A SELECT MUST CONTAIN AT LEAST ONE ITEM-ID SUBSTITUTION CODE (ASTERISK *)

287 TIMESLICE FOR LINE n EQUALS m MILLISECONDS

288 CURSOR CONTROL IN PROC STATEMENT : statement SPECIFIES COL#/ROW# GREATER THAN TERM SETTING.

289

	TERMINAL	PRINTER
PAGE WIDTH :	n	n
PAGE DEPTH:	n	n
LINE SKIP:	n	
LF DELAY:	n	
FF DELAY:	n	
BACKSPACE:	n	
TERMINAL:	n	
BAUD RATE:	n	
PCI:	n	

290 THE RANGE OF THE PARAMETER 'parameter' IS NOT ACCEPTABLE

291 'file name' FILE-DEFINITION IS MISSING.

293 TOTAL NUMBER OF CONTIGUOUS FRAMES = n
NUMBER OF FRAMES IN THE TEMPORARY SPACE BLOCK = n
NUMBER OF FRAMES IN THE TEMPORARY SPACE CHAIN = n

294 START OF LINKED OVERFLOW :
TOTAL NUMBER OF LINKED FRAMES :
TOTAL NUMBER OF CONTIGUOUS FRAMES :

295 OVERFLOW TABLE

298 FORMAT ERROR IN SPECIFICATIONS.

316 WHICH LINE?

318 COUNT TOO HIGH

330 ILLEGAL LINE NUMBER.

331 THE ACCOUNT FILE IS MISSING.

332 THE LINK FROM PRIMARY HS WORKSPACE TO ADDITIONAL WORKSPACE WAS INVALID

```
333     THE FORMAT OF THE ADDITIONAL WORK-SPACE PARAMETER: xxxx IS
        ILLEGAL FOR THIS ACCOUNT NAME.
        ADDITIONAL WORK-SPACE HAS NOT BEEN ASSIGNED.
334     FRAMES OF ADDITIONAL LOGON WORKSPACE ARE NOT AVAILABLE
335     "Logon Proprietary Message"
336     ***LOGGED OFF AT time ON date ***
337     USER IS NOT LOGGED ON.
338     ACCOUNT FILE STATISTICS WERE NOT UPDATED DUE TO EITHER:
            1. INSUFFICIENT WORK-SPACE TO CONTAIN THE ACCOUNT FILE ITEM,
        OR
            2. SYSTEM DICTIONARY CHANGED WHILE YOU WERE LOGGED ON.

339     IMPROPER OR UNDEFINED FORMAT FOR DATE CONVERSION
340     ************************************************************************
        ***           CONNECT TIME AT hh:mm:ss = n MINUTES                   ***
        ***       CPU MS. = n                  DISC READS = n                ***
        ***               NUMBER OF ACTIVATIONS = n                         ***
        ************************************************************************

341     A VALUE MUST FOLLOW THE CHARGE-TO VERB.
350     INDENTATION PAST END OF LINE.
398     THE FILE OR  DL/ID  REQUIRES A (VERTICAL) CORRELATIVE FOR THIS
        statement.
399     THE MAXIMUM OF 20 LEVELS FOR A "WITHIN"-TYPE STATEMENT HAS BEEN
        EXCEEDED.
401     NO ITEMS PRESENT
402     FILE DEFINITION 'item-id' CANNOT BE DELETED OR OVERWRITTEN
403     END OF LIST
404     n ITEMS SELECTED.
405     n ITEMS LISTED.
406     ITEM COUNT = n, BYTE COUNT = n,  AVG. BYTES/ITEM = n, AVG.
        ITEMS/GROUP = N, STD.  DEVIATION = n,  AVG. BYTES/GROUP = n
407     n ITEMS COUNTED.
408     ONE ITEM COUNTED.
410     A SYNONYM (Q-TYPE) FILE CANNOT BE SPECIFIED IN THIS STATEMENT.
411     'DICT' OR 'DATA' MUST BE SPECIFIED IN A CLEAR-FILE STATEMENT.
412     INSUFFICIENT DISK SPACE AVAILABLE
413     THE FILE NAME ALREADY EXISTS IN THE MASTER DICTIONARY.
414     ILLEGAL OR MISSING MODIFIER USED IN DEFINING THE FILE AREA(S).
415     'item-id' EXISTS ON FILE.
416     RANGE ERROR IN MODULO OR SEPARATION PARAMETER.
417     FILE 'file-name' CREATED;  BASE = n, MODULO = n' SEPAR = n
418     FILE-DEFINITION ITEM 'item-id' WAS NOT COPIED.
419     THE SPECIFIED FILE CANNOT BE CLEARED OR DELETED!
420     DICTIONARY FILE SELETION CANNOT BE DONE WITHOUT DELETION OF DATA
        FIRST.
421     STATISTICS OF attribute-name:  TOTAL = n AVERAGE = n COUNT = n
422     BYTE STATISTICS OF attribute-name:
        TOTAL = n AVERAGE = n ITEMS = n CKSUM = n BITS = n
423     TOTAL OF attribute-name IS:  n
```

424 DISK ADDRESS FOR FRAME NO. n
DEVICE ADDRESS: n
UNIT NO.: n
HEAD NO.: n
CYLINDER: n
SECTOR: n

425 INVALID FRAME-ID REQUEST

426 DATA FILE ALREADY EXISTS!

427 THERE IS NO DATA SECTION FOR THIS FILE.

470 YOUR REQUEST FOR FRAMES IS ILLEGAL! ONLY NUMBERS FROM 1 TO 127 ARE VALID

469 ILLEGAL FORM QUEUE NAME

489 OPERATOR INTERVENTION REQUIRED

520 NO DATA FOR BLOCK OUTPUT

521 TOO MANY CHARACTERS IN WORD TO BLOCK

522 BLOCK-CONVERT FILE MISSING OR IMPROPERLY DEFINED

523 BLOCK OUTPUT WOULD EXCEED PAGE WIDTH

524 INPUT CHARACTER 'x' IS NOT IN BLOCK-CONVERT FILE.

525 INPUT CHARACTER 'x' IS IMPROPERLY FORMATTED IN BLOCK-CONVERT FILE

530 ALREADY LOGGED ON

531 PROCESS ROADBLOCKED

532 ILLEGAL USER ID

533 LOGON SUCCESSFUL

534 LOGOFF SUCCESSFUL

535 ILLEGAL LINE NUMBER.

550 A REQUIRED NUMERIC PARAMETER IS MISSING OR INVALID

551 SPECIFY NUMBER OF LINES TO SKIP

552 ITEM 'item-id' HAS INVALID FORMAT

599 OBJECT CODE IN 'item-id' IS IN INCORRECT FORMAT.

603 DEFAULT ASSIGNMENT IS TO QUEUE "STANDARD".

604 'x' IS AN UNDEFINED LABEL REFERENCE.

605 FORM QUEUE =
DEVICE TYPE =
DEVICE NUMBER =
SPOOLER OPTIONS =
COPIES =

607 UNACCEPTABLE VALUE

609 THE PAGE YOU REQUESTED IS NOT AVAILBLE+

610 LINE# 'n' IS ASSIGNED TO THIS QUEUE. FORM QUEUE CANNOT BE DELETED!

611 TYPE Y (YES) OR N (NO):

612 THE DEFAULT FORM QUEUE MAY NOT BE DELETED

613 THE FORM QUEUE IS NOT EMPTY

614 NOT A PORT PROCESS

615 DEVICE TYPE:

616 CANNOT MOVE A FORMQ TO ITSELF

617 DEVICE DOES NOT EXIST

618 N OPTION DISREGARDED

619 LINE# --OR-- LINE# FORM-QUEUE:

620 LPTR ATTACHED!

621 CHECK SPOOLER!

622 SPOOLER ABORTED. RESTART IS BEING ATTEMPTED.

623 ILLEGAL ACTION CODE

624 ENTER ACTION CODE / PAGE NUMBER (P#)+
625 FORM-QUEUE PAGES:
626 NO OVERFLOW FRAMES
627 ENTER ACTION CODE:
628 PRINT-JOB# OPTIONS:
629 FORM-QUEUE FORMTYPE:
630 'STOP' IN EFFECT. USE 'RESUME'!
631 OLD-NAME NEW-NAME:
632 PRINT JOB # COPIES:
633 PRINTFILE #:
634 FORM-QUEUE:
635 FORM-QUEUE OR PRINT JOB #:
636 FORM-QUEUE--OR--DEVICE# TYPE:
637 FROM-FORM-QUEUE TO-FORM-QUEUE:
638 NEW-FORM-QUEUE PRINT JOB LIST:
639 PRINT JOB LIST:
640 FORM-QUEUE DEVICE-TYPE(LPTR,TAPE,PORT,SPOOL) DEVICE#:
641 THERE ARE n HOLD FILES
642 EJECT AND SKIP VALID ONLY FOR PRINTERS
643 THE FORM QUEUE TABLE IS FULL
644 PRINTING SUSPENDED ON JOB# 'n'
645 THERE IS NO ACTIVE JOB TO RESTART
646 ********** TOP OF PAGE **********
647 THE DEVICE IS ALREADY ASSIGNED AND CANNOT BE CHANGED.
648 DEVICE TYPE DEVICE#:
649 NEW-NAME:
650 FORMTYPE
651 CANNOT ALIGN - CANNOT ATTACH DEVICE
652 TOO MANY PAGES!
653 OPTIONS:
654 ILLEGAL OUTPUT STRING LENGTH!
655 NOT ATTACHED!
656 JOB ALREADY BEING EDITED BY LINE # 'n'
657 FILE NOT CLOSED
658 AN INSTANT PRINT JOB WITHOUT AN H OPTION CANNOT BE EDITED
659 MUST SUSPEND PRINTING TO EDIT THIS JOB
660 INSUFFICIENT PRIVILEGES OR DEVICE ATTACHED TO SPOOLER
661 TAPEOUT TERMINATED BECAUSE OF ASSIGN TAPE.
662
```
                    PRINT JOBS
PAGE     OF     PAGES
PRINT        DATE
QUEUE  NAME        JOB  ACCOUNT       LINE  CREATED  STATUS  OPT  SIZE
COPIES
```

663
```
1. MOVE FROM GUEUE    5. DELETE JOB        9. SUSPEND PRINTING
2. MOVE PRINT JOB     6. STOP PRINTING    10. CHANGE PRIORITY
3. CHANGE OPTIONS     7. RESUME PRINTING  11. SP-STATUS
4. CHANGE #OF COPIES  8. EDIT PRINT JOB   99. EXIT
```

664 PAGE x OF y PAGES
665 THE QUEUE IS EMPTY
666 ILLEGAL NUMBER: n
667 WRONG PASSWORD FOR PRINT JOB # n

668 JOB NOT QUEUED FOR OUTPUT
669 PRINT JOB # n DOES NOT EXIST
671 PRINT JOB # n ALREADY PRINTING
672 COPY RANGE IS 1-10
673 INVALID OPTION: option
674 THE DEVICE YOU WANTED IS CURRENTLY ASSIGNED TO AN ACTIVE FORM QUEUE. PLEASE SUSPEND THAT FORM QUEUE AND THEN CHANGE THE DEVICE ASSIGNMENT.
675

FORM QUEUE ASSIGNMENT

QUEUE NAME	FORM TYPE	DEVICE ASSIGNED	DEVICE TYPE	DEVICE STATUS	NUMBER ENTRIES	PAGE SKIP

676 PRINT JOB SELECTED IS NOT A HOLD FILE
677 STRINGS?
678 OCCURANCE MUST BE NUMERIC
680 SYNTAX ERROR
681 INVALID SPOOL TAPE FORMAT
682 THE FORM-QUEUE CANNOT BE FOUND
683 PRINT JOB CANNOT BE OPENED ACCOUNT NAME AND PASSWORD TOO LONG
685 PRINT JOB TABLE FULL
686 PRINT JOB FULL
687 A DEVICE HAS NOT BEEN ASSIGNED TO THIS QUEUE
688 INCOMPATIBLE OPTIONS: NI, NH, N COPIES, I COPIES, OR N PORT
689 ACCOUNT NAME CANNOT BE FOUND
690 QUEUE IS ALREADY ASSIGNED TO THIS DEVICE. OK TO CHANGE?
691 THE LIST OF PORTS WITH NO DEVICES ASSIGNED IS FULL
692 INVALID DEVICE TYPE
693 INVALID DEVICE# RANGE
694 FORM QUEUE ALREADY EXISTS
695 FORM QUEUE ACTIVE. ITS ASSIGNMENT CANNOT BE CHANGED
696 NAME TOO LONG
697 PRINT JOB# n REQUIRES A PASSWORD
698 PRINT JOB# n IS NOT CLOSED
699

ACTION CODES

1. CREATE FORM QUEUE 4. CHANGE DEVICE ASSIGNMENT 7. DELETE FORM QUEUE
2. CHANGE FORM QUEUE 5. CHANGE PAGE SKIP 8. PAGE EJECT
3. CHANGE FORM TYPE 6. LIST PRINT JOBS 99. EXIT

700 INVALID G-CORRELATIVE DEFINITION: definition
701 INVALID FUNCTION CORRELATIVE DEFINITION: function-definition
702 OF MEMORY
ABS FRAMES
ACTIVE LINES
IS THE LINE TO WHICH THE SPOOLER IS ASSIGNED
IS THE FRAME ID OF THE PCB FOR LINE ZERO
IS THE FRAME ID OF THE BASE OF THE SYSTEM DICTIONARY
IS THE MAXIMUM FRAME ID FOR THIS SYSTEM
DISC DRIVE CONFIGURATION:
DEVICE UNIT DENSITY PLATTERS

703 INVALID C-CONVERSION/CORRELATIVE DEFINITION: definition
704 INVALID P-CONVERSION DEFINITION: definition

705	ILLEGAL CONVERSION CODE: conversion-code
706	THE TRANSLATE CONVERSION CODE: 'code' IS ILLEGAL.
707	DL/ID ENTRY FOR T-CONVERSION: conversion-code NOT FOUND.
708	'item-id' CANNOT BE CONVERTED.
709	CONFLICT IN T-CONV DEFINITION; REQUESTS VERIFY BUT THERE IS A NULL AMC
710	INVALID L- OR R- CONVERSION DEFINITION: definiton
712	VALUE TEST IMPOSSIBLE ON ATTRIBUTE WITH F-CONVERSION: definiton
715	NON-NUMERIC DIGIT WITHIN PARENTHESIS : 'n'
716	MISSING LEFT PARENTHESIS: 'n'
717	'P' IS MISSING IN PICTURE CLAUSE :'n'
718	ILLEGAL CHARACTER FOUND ON PICTURE CLAUSE: 'n'
719	LITERAL EXCEEDS MAX. OF 20: 'n'
720	MISSING RIGHT PARENTHESIS: 'N'
721	MISSING ' IN CURRENCY SYMBOL: 'n'
722	MISSING " IN LITERAL SYMBOL: 'n'
723	MAX. LENGTH DEFINED IN DICTIONARY IS TOO SMALL FOR MF CONVERSION
736	NO SCALING WITH NON-NUMERIC DATA: PAGE DATA PAGE DATA PAGE DATA
737	TRUNCATION ERROR FOUND IN STORED DATA: PAGE DATA PAGE DATA PAGE DATA
738	NON-ALPHABETIC CHARACTER FOUND: PAGE DATA PAGE DATA PAGE DATA
739	NON-NUMERIC CHARACTER FOUND: PAGE DATA PAGE DATA PAGE
740	SPECIAL CHARACTER FOUND IN DATA: PAGE DATA PAGE DATA PAGE DATA
741	INSUFFICIENT DATA TO MATCH PICTURE CLAUSE: PAGE DATA PAGE DATA PAGE DATA
742	NON-KATAKANA CHARACTERS FOUND: PAGE DATA PAGE DATA PAGE DATA
780	'item-id' NOT ON FILE
781	'item-id' ADDED
782	'item-id' UPDATED
783	'item-id' DELETED
800	n ITEMS DUMPED
801	+
802	n ITEMS DUMPED.
803	n ITEMS LOADED
805	n ITEMS COPIED
860	CHECK CODE ERROR
888	INVALID DATE
910	*T* PROCESS STOP.
911	*T* PROCESS HAS BEEN LOGGED OFF BY PH-KILL
912	*T* PROCESS ABORT DUE TO TIPH PROCESSING ERROR
913	*E* INPUT ERROR
914	*E* PROCESS IN DEBUGGER.
915	ERROR FOUND IN SUSPENDING LINE
916	ERROR FOUND IN RESUMING LINE
917	ERROR FOUND IN KILLING LINE
918	WRONG-INPUT
919	SYS2 ERROR
920	LINE # MISSING
949	ILLEGAL FRAME-ID IN PRIMARY SPACE! PROGRAM TERMINATED.
950	LINKFIELD ERROR - GROUP AT n FRAME n LINKS n n n n
951	ILLEGAL FORWARD LINK. GROUP AT n FRAME n LINKS n n n n

952 ILLEGAL BACKWARD LINK. GROUP AT n FRAME n
LINKS n n n n

953 SHOULD LINK BACK TO: GROUP AT n FRAME n
LINKS n n n n

954 n NEW ERROR ITEMS CREATED IN DICT TSYM

987 x x x x x x x

990 INVALID ABSDUMP SPECIFICATION

991 MM/DICT ENTRY MISSING, OR REQUIRED SPECIFICATION(S) MISSING.

992 n ITEM(S) HAVE BEEN RESTORED

999 INSUFFICIENT CONTIGUOUS BLOCKS

1004 ITEM 'item-id' IS NOT ON FILE

1006 ITEM 'item-id' EXISTS ON FILE

1010 THE PARAMETERS BEING CHANGED ARE NOT COMPATIBLE WITH THE PORT OPTION.

1011 BAUD RATE AND PCI TRANCMISSION CHARACTERISTICS CANNOT BE MODIFIED WITH THE SET-TERM VERB. USE "TERM" INSTEAD.

1012 THE PCI TRANSMISSION CHARACTERISTICS CANNOT BE CHANGED WITHOUT ALSO DEFINING THE BAUD RATE.

1013 PORT ZERO'S BAUD RATE OR PCI TRANSMISSION CHARACTERISTICS CANNOT BE MODIFIED WITH THE TERM VERB. USE THE DMP INSTEAD.

1014 ENFORCED WRITE FREQUENCY IS 1 WRITE PER n ACTIVATIONS

1021 DEVICE(S) TAPE 'unit list' NOW ASSIGNED,
RECORD SIZE = 'n'

1025 DEVICE TAPE 'unit #' IS ASSIGNED TO LINE 'line #'

1100 xxxxxxK OF MEMORY
xxxx ABS FRAMES
xxx ACTIVE LINES
xxx IS THE LINE TO WHICH THE SPOOLER IS ASSIGNED
xxx IS THE FRAME ID OF THE PCB FOR LINE ZERO
xxx IS THE FRAME ID OF THE BASE OF THE SYSTEM DICTIONARY
xxx IS THE MAXIMUM FRAME ID FOR THIS SYSTEM
DISC DRIVE CONFIGURATION:
DEVICE ADDRESSES

1101 DISK ADDRESS FOR FRAME NO.
DEVICE ADDRESS :
HEAD NO :
CYLINDER :
SECTOR :

1102 ERROR IN WRITE INSTRUCTION TEST NUMBER

1111 MISSING OR ILLEGAL CHANNEL NAME

1112 ILLEGAL ASSIGN STATEMENT FORMAT

1113 ILLEGAL UNIT SPECIFICATION

1114 MISSING OR ILLEGAL RECORD SIZE

1115 EXTRANEOUS INPUT AT END OF STRING

```
 1116      DUPLICATE ENTRIES IN UNITLIST
 1119      DEVICE TAPE 'unit #' IS ASSIGNED TO THIS LINE, BUT FOR ANOTHER
           CHANNEL.
 1120      DEVICE(S) TAPE 'unit list'  NOW ASSIGNED TO CHANNEL 'channel name'
 1121      ILLEGAL CHANNEL SPECIFICATION.
 1122      TAPE UNIT NUMBER MUST BE SPECIFIED.
 1123      ILLEGAL RECORD SIZE.
 1124      DEVICE TAPE 'unit #' NOT ON THIS SYSTEM
 1125      UNIT NOT ASSIGNED
 1127      SEQUEL
           n     SYSTEM SERIAL NUMBER
           nK    OF MEMORY
           n     IS MAXIMUM FRAME-ID
           n     PORTS
           n     IS FIRMWARE TYPE
           n     IS FIRMWARE VERSION NUMBER
                 WORDMATE
                     AVAILABLE DEVICES
                     DEVICE    UNIT(S)

 1129      NO SUCH TAPE ASSIGNMENT EXISTS.
 1193      UNRESOLVED TAPE SYSTEM ERROR
 2001      MISMATCHES IN INTERMEDIATE OBJECT CODE.
 2002      ABORT, RE-EXECUTE TCL-II VERB 'STOREPF'
 2321      ITEM 'item-id' ERROR
 9009      PQ-SELECT REGISTER OUT OF RANGE 1-5
 9012      UPDATE ATTEMPTED ON AN INPUT FILE AT STATEMENT: statement
 9020      COMPILATION ABORTED.
 9021      PROC FORMAT ERROR IN LABEL
 9022      DUPLICATE LABEL FOUND.
 9025      FORMAT ERROR IN 'GO' STATEMENT
LOGON      L
```

APPENDIX C. STAT-FILE DICTIONARY

A/AMC	Name	Definition
0	ORDER.	Sequence in which file was encountered during the file-save
1	ACCOUNTNAME.	First segment of Attribute 1 (G*1). Specifies the account the file resides in
1	FILENAME.	Second segment of Attribute 1 (G1*1). The name of the file
1	FILE.	Second and third segments of Attribute 1 (G1*2). If the third segment of Attribute 1 is present, this specifies the data section of the file named in second segment.
2	LEVEL.	System level of the specified file 0 = System Dictionary 1 = Master Dictionary 2 = Data dictionary 3 = Data file
3	BASE.	Beginning frame number of the file
4	MOD.	Modulo of the file
5	SEP.	Separation (number of frames per group) of the file
6	ITEMS.	Total number of items in the file
7	SIZE.	Size of the file in bytes (characters)
8	MAX-I	Size (in bytes) of the largest item in the file
9	MIN-I	Size (in bytes) of the smallest item in the file
10	AM/GP	Maximum number of items found in any group in the file
11	MN/GP	Minimum number of items found in any group in the file
12	FRAMES.	Total frames, including linked, used by the file
13	25%	Number of groups that are 25% full or less
14	50%	Number of groups that are 25% to 50% full
15	75%	Number of groups that are 50% to 75% full
16	100%	Number of groups that are 75% to 100% full

A/AMC	Name	Definition
17	125%	Primary space of groups in this category is full and 25% additional linked space is required
18	150%	Primary space of groups in this category is full and 50% additional linked space is required
19	200%	Primary space of groups in this category is full and additional linked space equals the primary space
20	>200%	Primary space of groups in this category is full and additional linked space is greater than the primary space.
21	GFE	Group format errors, if any, in the file
99	SECONDARY.	Computes the number of linked frames required for the file by multiplying modulo times separation and subtacting the product from the total frames (F;5;4;*;12;-)
99	AVG.SIZE	Computes the average item size by dividing the size by the total items in the file (F;6;7;/)
	DL/ID	DL/ID of STAT-FILE. Note that Attribute 1 is DY, not D.

APPENDIX D. CREATING THE PH-HISTORY FILE

The Terminal Independent Process Handler (TIPH) uses the PH-HISTORY file to indicate the status of the TIPH processes. If the file does not exist, you may create it by using the CREATE-FILE verb, adding the necessary attribute definition items using the EDITOR.

To create the PH-HISTORY file, perform the following:

1. Create the file on the SYSPROG account using the CREATE-FILE verb. The modulo and separation of the dictionary section should both be "1". The modulo of the data section should be "37", the separation should be "1".

2. Use the EDITOR to enter the attribute values shown in Figure A to create the attribute definition items in the dictionary portion of the PH-HISTORY file.

3. Use the EDITOR to change the file's D-pointer in SYSPROG's M/DICT to an 'E'.

4. Copy the E-pointer to the SYSTEM file.

5. Use the EDITOR to change the E-pointer back into a D-pointer.

6. Place a Q-pointer to PH-HISTORY in the M/DICT of each account which requires access to it.

Figure B shows an example of the creation procedure.

Attribute		ACCOUNT	DATE	LINE	OP	OUTFL#	STARTER-ACC	STATUS	TASK-ID	TIME
1.	D/CODE	A	A	A	A	A	A	A	A	A
2.	A/AMC	1	2	6	7	8	4	9	5	3
3.		*	*	*	*	*	*	*	*	*
4.		*	*	*	*	*	*	*	*	*
5.		*	*	*	*	*	*	*	*	*
6.		*	*	*	*	*	*	*	*	*
7.		*	D	*	*	*	*	*	*	*
8.		*	*	*	*	G 1	*	*	*	MTS
9.	V/TYPE	L	L	L	R	R	L	R	L	L
10.	V/MAX	10	12	2	2	3	11	6	10	8

* null

Figure A. Attribute Definition Item Information for PH-HISTORY

1. :CREATE-FILE (PH-HISTORY 1,1 37,1) <cr>

```
[417] FILE 'PH-HISTORY' CREATED;  BASE = xxxxx, MODULO = 1, SEPAR = 1
[417] FILE 'DL/ID' CREATED;  BASE = xxxxx, MODULO = 37, SEPAR = 1
```

2. :ED DICT PH-HISTORY ACCOUNT <cr>

```
NEW ITEM
TOP
.I <cr>
001 A <cr>
  .
  .
  .
'ACCOUNT' FILED.
```

Use the EDITOR to enter the other attribute definition items in a similar fashion.

3. :ED MD PH-HISTORY <cr>

```
TOP
.R/D/E/ <cr>
001 E
.FI <cr>
'PH-HISTORY' FILED.
```

4. :COPY MD PH-HISTORY (D) <cr>
TO:(SYSTEM) <cr>

1 ITEM COPIED.

5. :ED SYSTEM PH-HISTORY <cr>

```
TOP
.R/E/D/ <cr>
001 D
.FI <cr>
'PH-HISTORY' FILED.
```

6. :ED MD PH-HISTORY <cr> ← *(while logged onto the user's account)*

```
NEW ITEM
TOP
.I <cr>
001 Q <cr>
  .
  .
  .
'PH-HISTORY' FILED.
```

The format of the Q-pointer to the PH-HISTORY file is shown in Figure B of the topic TIPH INITIALIZATION MESSAGES.

Figure B. Creating the PH-HISTORY File

APPENDIX E. SAFEGUARDING SAVE TAPES

Tape errors may result from the effects of numerous sources. Tape transport and controller hardware defects and misalignments are the cause of many tape errors. A reliable save should not be expected from a faulty medium. On the tape transport, the tension bars must be properly adjusted to prevent tape cinching and edge damage. Head misalignment, signal level improperly set, and excessive wear from overuse of the same tape can all contribute to unrecoverable tape errors.

However, the more common source of tape errors is not mechanically induced by faulty equipment; instead, it is the result of careless handling of the tape during its use or its improper care while in storage. If the data to be saved is properly written to tape, the data cannot be expected to remain intact if the tape is left: 1) lying around the computer room, out of its case, near electromagnetic field and dust-generating equipment, 2) around eating and smoking people, 3) underneath stacks of listings, or 4) next to an open window on a windy, hot day.

The tape should be stored inside its case in a vertical position, in an environment whose temperature is within 5 degrees C of 24 degrees C, whose humidity is within 10% of 10 RH, and will be exposed to an electromagnetic field of less than 50 oersteds.

When the tape is used during a save or restore, care should be taken to ensure that the tape reel is handled in a manner which will not cause it to get dirty. The tape's case should be closed when the reel is not inside so that the tape is not exposed to dust. The reel should be held with the thumb inside the hub, not by the flanges, to prevent excessive pressure on the sensitive edges of the tape. When mounting the reel onto the tape transport, exert equal pressure all around the hub to ensure the reel is mounted flush to prevent edge damage.

It is necessary to emphasize that correct handling and storage of the magnetic tape is an integral part of any Save/Restore program.

For more information on care and maintenance of magnetic tapes, refer to the following manuals:

1) Computer Tape Care and Handling, Memorex Corp., 1979.

2) Magnetic Tape Management, Computer-Link Corp., 1981.

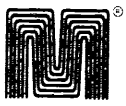
INDEX

E

M

N

T



Microdata Corporation

17481 Red Hill Avenue, Irvine, California 92714
Post Office Box 19501, Irvine, California 92713
Telephone: 714/250-1000 · TWX: 910-595-1764