# Xerox 9700 Electronic Printing System
# Reference Manual

XEROX

9700

Xerox Corporation
701 South Aviation Boulevard
El Segundo, California 90245
213 679-4511

XEROX

# Xerox 9700 Electronic Printing System

## Reference Manual

91 00 02C

April 1979



Printed in U.S.A.

## NOTICE

This publication documents the A00 version of the Operating System Software for the Xerox 9700 Electronic Printing System and supersedes the Reference Manual dated September 1978.

## RELATED PUBLICATIONS

| Title | Publication No. |
|---|---|
| Xerox 9700 Electronic Printing System Forms Creation Guide | 90 00 01 |
| Xerox 9700 Electronic Printing System Font Users Guide | 91 00 03 |
| Xerox 9700 Electronic Printing System Tape Formats | 91 00 04 |
| Xerox 9700 Electronic Printing System Operator's Guide | 600P81096 |
| Xerox 9700 Electronic Printing System Forms Description Language Self-Study Module | 500485 |
| Xerox 9700 Electronic Printing System Print Description Language Self-Study Module | 500484 |

# CONTENTS

## APPENDICES



## TABLES

# 1. INTRODUCTION

## General

The Xerox 9700 Electronic Printing System is a versatile, high-performance, general purpose, programmable printing system which may be used either as a stand-alone off-line printing system to process a variety of computer-generated magnetic tapes or as a sophisticated on-line printing subsystem.

High-performance operating characteristics of the Xerox 9700 are achieved by the effective combination and application of xerographic, laser, and computer technologies to control the flow of digital data, as well as the use of pre-cut standard size (8.5 x 11 inch) sheets of paper.

## Features

- Standard 8.5 x 11 cut sheet output, using plain or pre-printed paper.
- Executive quality output (300 x 300 bits per inch resolution).
- Large variety of paper weights (16-pound bond to 110-pound index) can be intermixed.
- Wide range of paper types - multicolored, 3-hole drilled, perforated.
- 2 sheets per second - up to 18,000 LPM.
- Continuous operation - dual input paper trays, dual output stackers, disk buffering of variable data.
- Electronic forms - 9700 generated capability for forms, logos, shading. (A forms library is stored on the system disk.)
- Highlighting of data - copy modification entries (CME).
- Landscape or portrait orientation.
- Multiple character sets (fonts) per page, including OCR, typewriter, italics, script.
- Fonts are selectable on a character-to-character basis.
- Multiple forms per report.
- Varying page densities by use of different font styles and sizes.
- Variable character size (4-24 point) and spacing (4-30 CPI) and proportional character spacing based on font style.
- Variable line spacing (3-18 LPI).
- Overprint capability
- Sample print tray (25 page capacity when using 20-pound bond paper).
- Copy set integrity maintained.
- Stacked reports.
- Interspersed reports.
- Selective processing of portions of reports.
- User-initiated offsetting of portions of reports.
- Extended job accounting.
- Job routing capability.
- Containerized output.
- Multiple copies, collate option.
- On-line to IBM 360/30 and up, and IBM 370/135 and up.
- Selectable input mode (on-line or off-line).
- Extensive tape formats.
- System status and operator prompting via text sequences.

- Keyboard display, for use as operator control and editing functions.
- Disk, for use as secondary memory.
- 2 bin output 1500 page/bin capacity (using 20-pound bond paper).
- 2500 page main input feeder, 400 page auxiliary input feeder (using 20-pound bond paper).

## Off-line Host System Support

The Xerox 9700 supports tapes from the following host systems, as defined in the 9700 Tape Formats manual:

- Burroughs medium systems (B2500, B2700, B3500, B3700, and B4700) printer backup and standard tapes.
- Burroughs large systems (B6700) ANSI, printer backup, and standard tapes.
- Honeywell 200/2000 series SPR, standard and COBOL print tape formats.
- Honeywell 6000 series SSF print tapes (packed format with embedded normal edit mode carriage control).
- OS Writer support, including recognition of banner pages for report offsetting.
- IBM DOS/360 and DOS/VS/370 POWER
- ANSI
- IBM OS/360 and DOS/360 Standard Tape Formats
- IBM DOS/360 GRASP
- UNIVAC SERIES 70
- UNIVAC 1100 Standard Data Files

## Upward Compatability from Xerox 1200 Computer Printing System

Xerox 9700 provides an upward compatible path for Xerox 1200 (EPCP B01) users. The following compatibility requirements are met:

- Standard 9-track, 1600 bpi tape formats accepted by Xerox 1200 (as listed above) are accepted by Xerox 9700.
- Format control specifications supported on Xerox 1200 are available on Xerox 9700.
- Options for stacked reports, interspersed reports, multiple copies, logical file separators, etc., are specifiable for Xerox 9700.
- Xerox created user interface conventions established for the Xerox 1200 are supported by Xerox 9700.
- On-line 1200 facilities are available on Xerox 9700.

# 2. SYSTEM OVERVIEW

## Introduction

This chapter provides a functional overview of both the hardware and software aspects of the Xerox 9700 Electronic Printing System.

## Hardware Functional Description

Major hardware units of this Xerox 9700 Electronic Printing System are: a xerographic printer, an imaging unit, an output stacker, a system disk, a controller (mini-computer), and a keyboard/display unit. Additionally, for on-line operations a channel interface controller is required, for off-line operations a magnetic tape unit and controller are required, and for selectable on-line/off-line operations both controllers and the magnetic tape unit are required. As shown in Figure 2-1, the printer, imaging unit, and output stacker are physically joined. The controller, system disk, magnetic tape unit, and the keyboard/display unit are connected via cables to the system.

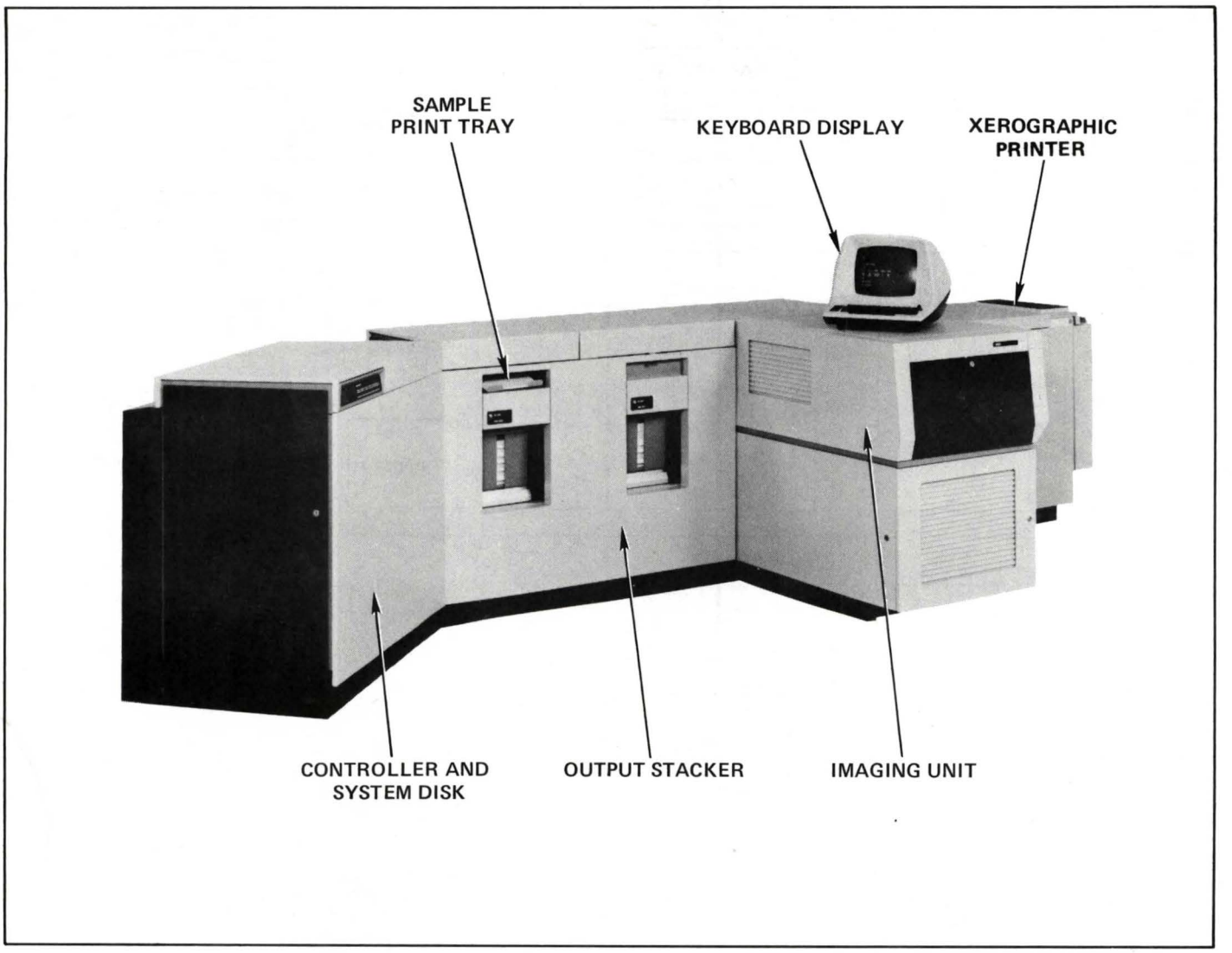



Figure 2-1. Typical Xerox 9700 Electronic Printing System

Functionally, the Electronic Printing System may be considered as five subsystems which perform the tasks necessary to meet the system's requirements. These subsystems (see Figure 2-2) are: Input, Control, Imaging, Xerographic, and Output.

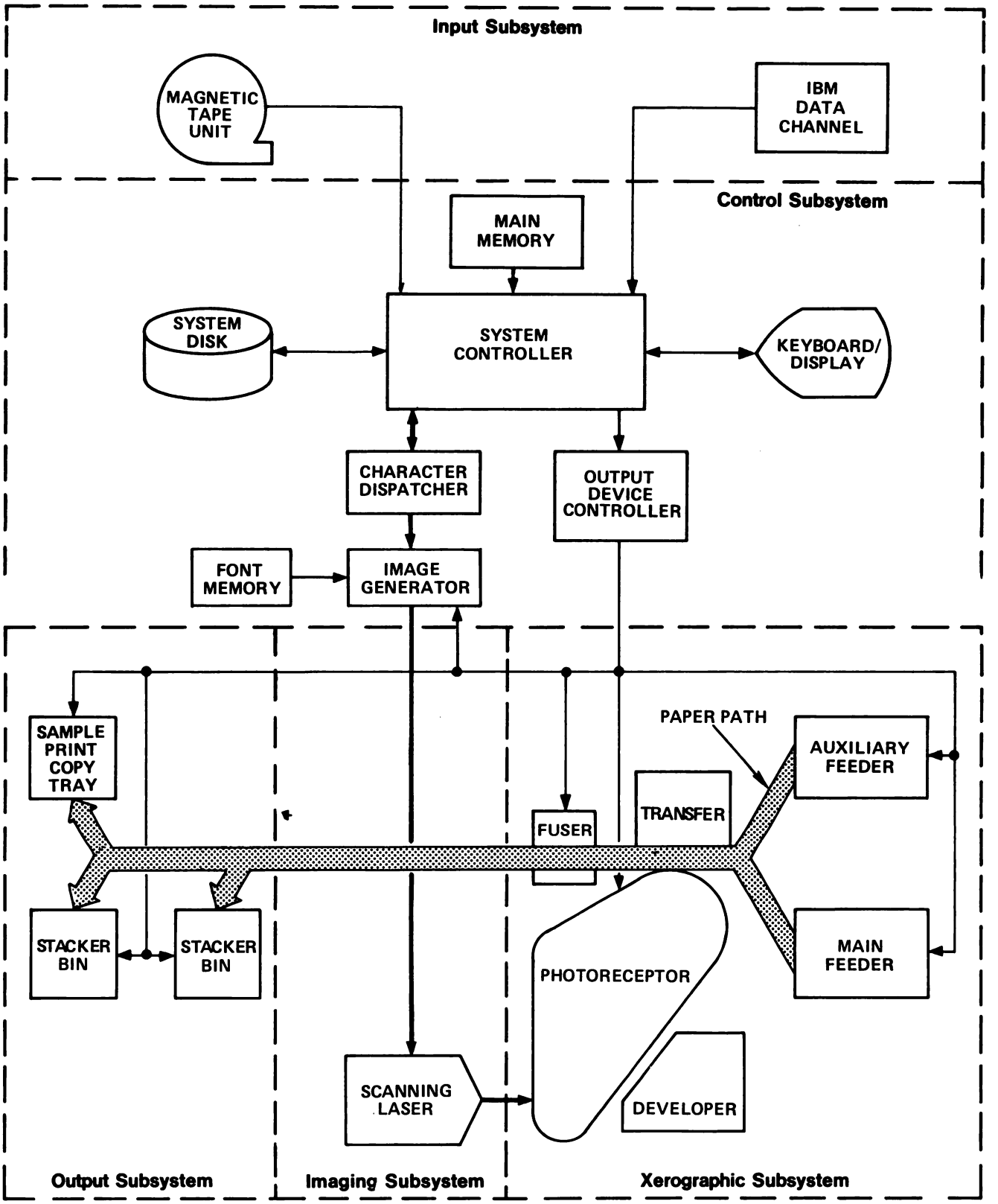



Figure 2-2. Functional Diagram of Xerox 9700 Electronic Printing System

### Input Subsystem

The input subsystem provides print data to the system. In off-line operation, the print data is obtained from a 9-track 1600 or 6250 bpi magnetic tape unit. In on-line operations, the print data is obtained via a data channel connected to either am IBM 360/30 and up or an IBM 370/135 and up. A selectable system allows a choice between either mode of operation.

### Control Subsystem

The control subsystem performs all of the data handling, formatting, buffering and operator control of the system It consists of a system controller (a mini-computer), main memory, a keyboard/display unit, a disk unit, and interface and control electronics. Called into the system controller main memory are software tasks to direct page formatting, system control, forms compilation, and system diagnostics.

The interface and control electronics handle controller-to-device communications and control of specific device functions. They provide the necessary interfacing between the I/O circuits and the device circuits. They also accept control orders from the system controller and exercise the indicated functional control over the devices. Device status is communicated to the computer software via the control and interface electronics.

The keyboard/display unit permits the operator to exercise control over the system. The control software communicates with the operator via easy-to-use English language instructions, status, and informational displays. Operator commands are also in English and easy to learn.

### Imaging Subsystem

The imaging subsystem accepts the formatted page of data, which has already been merged with the forms data, and utilizes a scanning laser beam to generate lines of modulated light for the continuously moving photoreceptor of the xerographic system, creating a latent image of the output page. The raster scanning laser beam gives wide flexibility in formatting the output page, since composition, size, style, and orientation of character sets, electronic forms, and logos can be programmably modified. Electronic forms, logos, and character sets which have been previously stored on the disk may be called as needed and merged electronically resulting in perfect registration between the variable data and the forms data. The orientation of the printed data may be either landscape (parallel to the long dimension of the paper) or portrait (parallel to the short dimension).

### Xerographic Subsystem

The xerographic printing subsystem incorporates all the typical functions of the xerographic printer including paper handling and development of the latent image of the output page.

All information to be printed, including forms and logos, as well as text, is received by the xerographic processor as a series of binary light signals from the scanning laser beam. Xerographic properties of an endless belt permit its surface to be uniformily charged electrically and then selectively discharged (or "exposed") by a modulated laser beam to create a latent image of the desired output. The latent image is subsequently "developed" by using dry ink (or toner) and then transferred to a sheet of plain paper. The final image is "fixed" when the paper passes through a fusing station on its way to the output

stacker. The endless belt is cleaned and recharged in preparation for the next cycle of operation. This imaging and printing process is performed with a resolution of 300 bits (or dots) per inch in the horizontal and vertical directions and at the rate of two sheets per second, where each sheet is 8.5 x 11 inches.

The main paper feeder may hold up to 2500 sheets and the auxiliary paper feeder may hold up to 400 sheets. When both paper feeders are filled with the same type of paper the printer may be operated continuously by utilizing the auxiliary paper feeder while the main paper feeder is being filled. Alternatively, the auxiliary paper feeder may be filled with paper (as heavy as 110-pound index) to be used as covers or separators. Transfer between main and auxiliary paper feeders is programmably controlled. Note that paper weights may be intermixed in either feeder.

Under program control, individual sheets of paper are obtained either from the main or auxiliary paper hopper, passed over a transfer station (where the image is received on the underside of the sheet, i.e., the sheet is in a face down position), passed through the fuser station (where the image is fixed), and routed to the appropriate output bin or sample print tray (still face down). Thus, successive sheets of a multi-page report are in the proper sequence in the output bins or sample print tray. Since the individual sheets are moved in a straight, direct, and short path, without being inverted, the probability of a paper handling error is very low.

### Output Subsystem

The output subsystem provides the capabilities for paper stacking, report collating and sample prints. The two-bin output stacker is a modular unit with its own paper transport mechanism. It is physically joined to the imaging unit and operates as an extension to the printer. Pages delivered by the printer are carried across the top of the stacker, routed to the appropriate bin, and stacked face down. Therefore, the first page of a job or run appears on the bottom of the bin and the last page appears at the top. The pages are in the correct order because of the face-down stacking. As a convenience to the operator in distinguishing between output sets, separation is provided by offsetting each copy set approximately 0.5 inches from the previous one. The output is containerized to help retain offset integrity of the stack while being moved to an offline finishing station or work area.

Each stacker bin may hold up to 1500 pages. Automatic bin switching to the inactive bin occurs when the active bin has been filled. If the inactive bin is not ready when the active bin reaches its capacity, the system will stop printing. Normal processing may be continued after a bin has been made ready. Bins can be unloaded without stopping the printer, as long as one bin is in the ready mode.

The operator may select either bin for loading. If both bins are empty and initialized, and neither is manually selected, bin 1 is selected automatically. Bin selection may be switched by the operator at any time, provided the alternate bin is in the ready mode.

The sample print tray permits delivery of sample-print pages directly to the user. Sample print pages may include printouts of information stored in the system (i.e., specified forms, font sets, and logos) or a duplicate copy of the current output page or a sample page, if no report is being processed. Logs and Dumps, generated at operator request or automatically by the system, are also delivered to the sample print tray. This sample print feature allows the user to examine any job page.

Unless the sheet is intended for delivery to the sample print tray only, each sample print sheet is reprinted and delivered to the active stacker bin. Thus, the integrity of the output copy set is maintained.

## Software Functional Description

Functionally, the Electronic Printing System software, called the Xerox 9700 Operating System Software (OSS) consists of several identifiable components. These components are:

- The EXECUTIVE which schedules all processing for the Operating System and the Operating System Diagnostic Software.
- An INPUT PROCESSOR which decodes input data from an off-line magnetic tape or from an on-line interface. Further details about the INPUT PROCESSOR functions are described below.
- An OUTPUT PROCESSOR which controls the operation of the xerographic printer, automatically merges forms data with the computer-generated data, and outputs this data in a coded format to be printed. Further details about the OUTPUT PROCESSOR functions are described below.
- A PRINT DESCRIPTION LANGUAGE (PDL) processor which permits the specification of general printing environment characteristics.
- A FORMS DESCRIPTION LANGUAGE (FDL) processor which allows for the description of forms on the system using simple, English language commands. Refer to Forms Creation Guide Xerox Publication No. 91 00 01 for further details on FDL.
- An EDITOR which facilitates editing source files and manipulating disk files.
- A FILE MANAGEMENT SUBSYSTEM which creates and maintains disk files.
- OPERATOR COMMUNICATION SUBSYSTEM which provides the interface between the 9700 operator and other subsystems.
- Various UTILITIES which do such tasks as produce accounting summaries and control font loading.

Figure 2-3 illustrates these components as they relate to main memory usage. The EXECUTIVE, FILE MANAGEMENT SUBSYSTEM, and OPERATOR COMMUNICATION SUBSYSTEM are always resident in main memory, the other tasks are disk resident and are called into main memory when needed.

### Input Processor Functions

Input data is processed and written to disk for subsequent printing. During processing, the following functions are performed:

- Block selection/deletion
- Extraction of records within blocks
- Record selection/deletion
- Detection of user-defined offset points
- Detection of report delimiter records
- Detection of parts of reports to print
- Detection of Dynamic Job Descriptor Entry (DJDE) records
- Detection of user-specified font usage requests
- Character translation
- Printer carriage control processing
- Replacement of input text with pre-specified static text
- Font change information

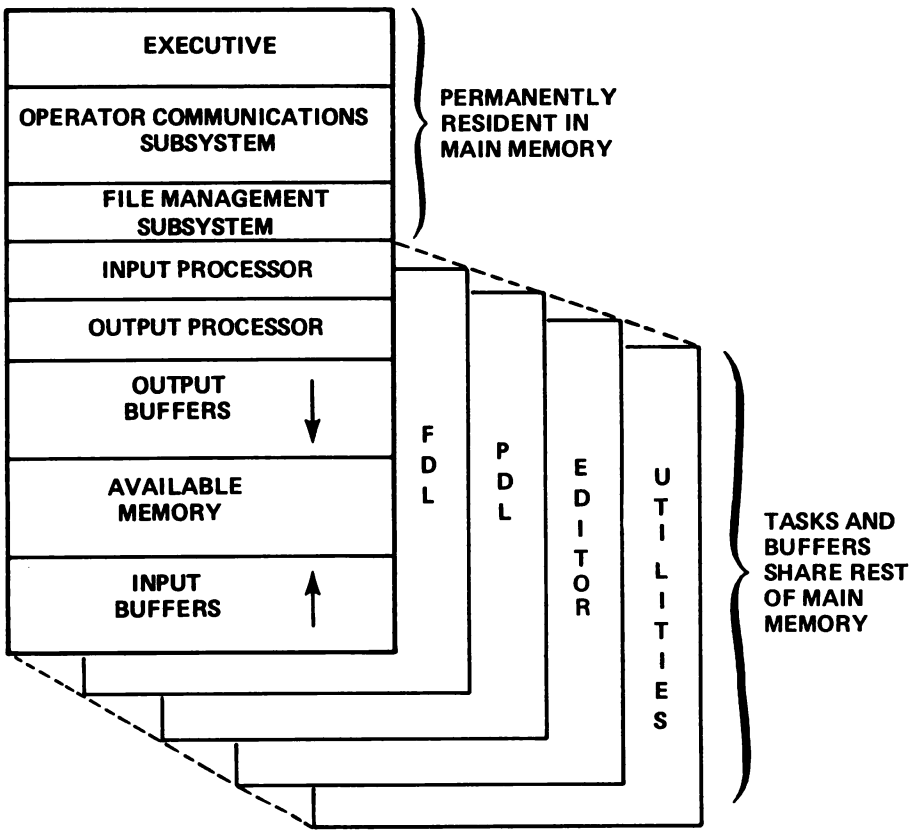



Figure 2-3. 9700 OSS Components

Block Selection/Deletion. The user may specify that certain data blocks are to be deleted for processing purposes. These options are specified through the use of the BSELECT or BDELETE commands (see Logical Processing Commands). If no option is specified all data blocks are selected for processing.

Record Extraction. Record extraction is performed according to the structure of the input data. In the case of data input from tape, blocks may be variable or fixed in length, and records within blocks may be of fixed, variable or undefined length (see "RECORD" Command). Records may be separated by constant delimiters, in which case a search process is required to locate records within blocks.

Record Selection/Deletion. The user may specify that only certain data records are to be processed for printing. Alternatively, the user may specify that certain data records are to be deleted from the printed report. These options are specified through the use of the RSELECT or RDELETE command (see Logical Processing Commands). If no option is specified all data records are selected for printing. All selected data records are moved from the input buffers to disk blocking buffers in preparation for writing to disk.

Detection of Report Delimiter Records. In general, end-of-report is synonymous with end-of-file. However, the user may specify that the end-of-report is to occur upon detection of certain data within an input record. This is done through the use of the RSTACK command. In this case, detection of the specified data will cause end-of-report processing to be performed.

Detection of Dynamic Job Descriptor Entry (DJDE) Records. If an identifier has been specified for DJDE Commands (see IDEN command), then each record is interrogated to determine if it contains DJDE information. DJDE information is processed as encountered and modifies current job characteristics. The effect of this is to permit the user to tailor the printing environment according to the particular needs of a report. DJDE information is applied to control the printing of data occurring after the DJDE information. DJDE records may occur within report delimiter records and in this case the DJDE information will apply to the next report following.

Detection of User-Specified Font Usage Requests. The user may specify that selected data records contain a byte index indicating the font to be used in printing data contained in the record. The index byte is specified as a number in the range 1 to n, where n is the number of fonts specified in the PDL FONTS option of the PDE command. The index specified is an index into the ordered list of fonts specified in the FONTS option. The user may intermix the use of fonts within a print line by specifying different fonts within different input data records, each of which contains a printer carriage control that causes normal upspacing to be inhibited.

Character Translation. As selected data records are moved from the input buffers to the disk blocking buffers, character translation is performed as necessary. The user specifies the input format through the use of the CODE option of the PDL VOLUME command.

Printer Carriage Control Processing. A printer carriage control (PCC) type is specified by the user according to the type of host system (see PCC command). Each PCC value specified corresponds to a particular action to be taken (i.e., space or skip before printing, print or not, and space or skip after printing). Channels are associated with page print line positions by the PDL VFU command. Spacing and skipping actions are processed by associating the appropriate laser scan line or dot position with the beginning of each data line to be printed. The starting print line position so generated is part of the control information associated with each print line and directs the character dispatcher in the printing of the line.

Copy Modification Entries. The user may specify that certain text is to be used to replace selected portions of printed page data. This is done through the use of Copy Modification Entries (CMEs). A CME permits replacement of selected portions of page data with pre-specified static data. Different CMEs may be associated with different copy plies of a report. CMEs may also be used to create font changes in variable data.

### Output Processor Functions

Output data is read from the disk and dispatched to the printer via the image generator a page at a time. During processing the following functions are performed:

- Page log read
- Fonts loaded into font memory
- Font specifications loaded into main memory
- Form loaded
- Paper feeding control
- Report integrity control

Page Log Read. The page log is read into memory to provide the output task with the font and form information required for imaging the page. It also provides a look ahead capability for determining what new fonts or forms will be required for the next page.

Fonts Loaded into Font Memory. Fonts are loaded into font memory as required by the page log for the next page to be imaged (i.e., if different from the fonts currently being used and not currently loaded for previous pages).

Font Specifications Loaded into Main Memory. The font specifications are loaded into main memory to provide the necessary line and character spacing information for imaging the page using that font.

Form Loaded. A form is loaded into memory for the next page to be printed, if different from the form currently being printed and not currently loaded for previous pages.

Paper Feeding Control. The paper feeding from the main and auxiliary paper trays is controlled by the output task. When the same stock is loaded into both trays, an automatic switch can occur to the full tray when the tray currently in use becomes empty. When cover stock is loaded into the auxiliary tray, it will be picked at the beginning and/or end of the report if specified by the job descriptor information.

Report Integrity. The integrity of the report is maintained by the system through character parity checking, automatic page tracking along the paper path, aborting any page which may be affected by hardware error, and automatic page recovery of all affected pages after a paper jam is cleared.

# 3. PRINT DESCRIPTION LANGUAGE

## Introduction

The Print Description Language (PDL) is a keyword oriented language used to specify the characteristics of the printing environment on the Xerox 9700. PDL source statements are coded by the user to define job tape formats, processing features, output formats to be used in the printing of tapes, and the control of on-line host processing. These PDL coded statements describing a particular set of jobs to be run are called a Job Descriptor Library (JDL). Within the JDL there may be one or more unique definitions for different processing features, output formats, and tape formats. These definitions represent a job, and are individually called Job Descriptor Entries (JDE).

A source statement Job Descriptor Library is compiled by the PDL processor and a JDL control file is written to system disk. To print a particular job tape, or start printing from an on-line host, the operator will request via a START command the JDL control file and the desired JDE within it.

This chapter defines the syntax of the PDL statements and illustrates how Job Descriptor Libraries are created, compiled and initiated. A summary and index of all PDL commands is contained in Appendix A. Detailed definitions of the PDL commands for off-line are contained in Chapters 4 and 5. Commands for on-line are discussed in Chapter 12.

For those unfamilar with PDL concepts and coding techniques, the PDL tutorial manual "XEROX 9700 PRINT DESCRIPTION LANGUAGE SELF STUDY MODULE" (Publication Number 500484) is recommended reading.

## PDL Statement Structure

### Record Format

Figure 3-1 illustrates the composition of a typical PDL statement. PDL statements consist of records of up to 133 characters, of which only characters 1-72 may be used for command information. Records of less than 72 characters are acceptable. Columns 73 through 133 may be used for identification or sequence information; however, this type of information is not processed by PDL or shown on PDL listings. Each PDL statement consists of a command keyword, and one or more left/right parts. Within a statement's left/right parts commas (or spaces) are used for separators.

Statements may be continued on successive records. Cross over is performed from one record to the next when column 73 is reached or at the end-of-record for records of less than 72 characters. Multiple statements may appear on one record if separated by semicolons.

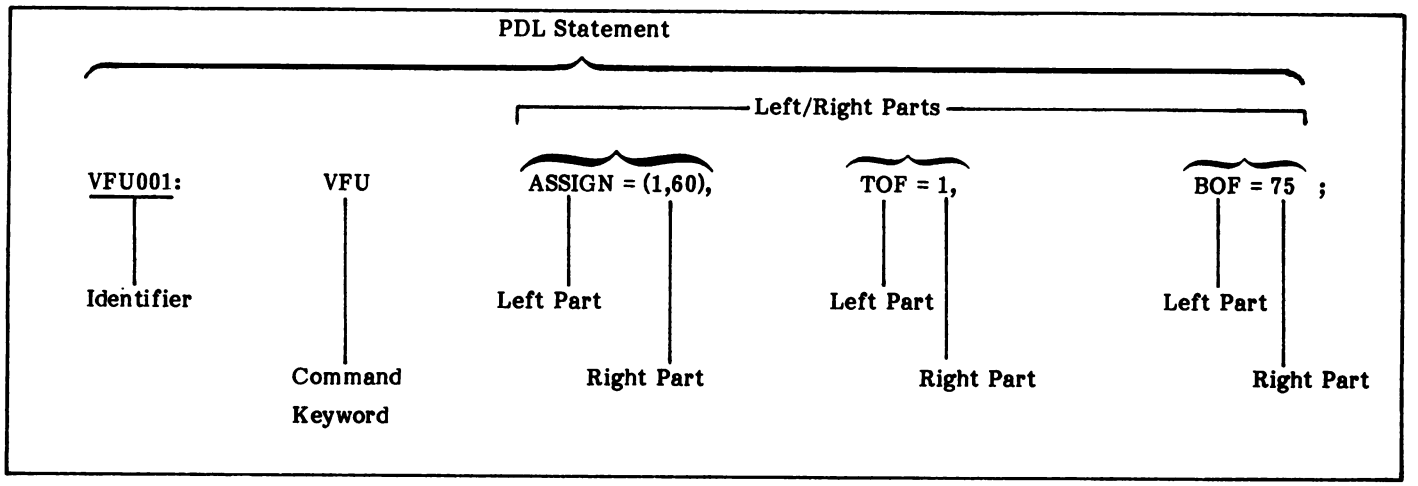



Figure 3-1. Print Description Language Statement Format

### Statement Components

A statement identifier (VFU001 in Figure 3-1), which permits referencing by another command, must be followed by the colon symbol (:). Identifiers have one of two formats depending upon the command with which the identifier is associated. All identifiers will consist of 1 to 6 alphanumeric characters (A-Z and 0-9) followed by a colon (:). At least one of the characters must be an alphabetic character except in the case of the SYSTEM (or JDL) and JOB (or JDE) commands which may have all numeric identifiers.

Keywords (commands and left/right parts) must be written as defined in Appendix A. The command VFU and the keywords ASSIGN, TOF, and BOF in Figure 3-1 are examples. They must be completely contained within one record and not continued from one record to the next. However, if a constant or identifier does not end at column 72 of one record, it must continue in column 1 of the next record.

The equals symbol (=) is used to join left parts to right parts (ASSIGN = (1, 60) in Figure 3-1). Commas are used to separate right parts as well as sets of left/right parts, blanks may be used to separate left/right parts. Parentheses are used to enclose multiple right parts.

Further details on left/right parts are contained in the following section.

Annotational comments are allowed, and they may appear anywhere within the job descriptor library. They must be preceded by the character sequence slash and asterisk (/*) and terminated by the character sequence asterisk and slash (*/).

Blanks may appear anywhere except in keywords and constants. Blanks terminate identifiers. For example:

```
V1     :VFU BOF = 65   ;
LINE PCC = (10,NOTRAN), VFU = V1   ;
```

A semicolon (;) must be used to explicitly terminate a statement. Multiple statements may be contained within one record, but each statement must be terminated by a semicolon.

## LEFT/RIGHT PARTS

Left/right parts are permitted in most statements, as shown in the PDL Statement Summary of Appendix A. With the exception of logical processing statements, left/right parts are optional; they may be written in any order within the statement. A set of left/right-parts consists of a left part, an equals (=) sign, and one or more right parts.

Left parts are always keywords, which must follow the syntax as shown in Appendix A. Right parts may consist of keywords, identifiers, value constants, string constants, or combinations of these various parts. If more than one right part is associated with a left part, the right parts must be separated by commas and enclosed in parentheses. Permissible combinations of right and left parts for a given command are defined in Appendix A. Keywords used as right parts must be written as shown.

## IDENTIFIERS

Identifiers used as right parts refer to statement identifiers defined in preceding statements. For example:

```
Statement Identifier
        |
VFU001: VFU BOF = 66;

        LINE VFU =  VFU001;
                    |
                    Right Part
```

## VALUE CONSTANTS

Value constants are constants which have an arithmetic value. They should be expressed as decimal numbers. They may be expressed as hexadecimal values, octal values, or even character values, but these expressions are not recommended. Decimal constants may be signed and in some cases may have fractional digits. For example:

```
OUTPUT COPIES = 1,  }
OUTPUT COPIES = 01, }   (equivalent decimal constant)

RECORD OFFSET = +60,   ADJUST = -50;
                |          |
                Decimal Constants
```

## STRING CONSTANTS

String constants are constants which are normally used to specify strings of characters. The width of string constants is important. String constants may be expressed as hexadecimal characters, ASCII, EBCDIC, or octal values, but not as decimal numbers.

Hexadecimal. Hexadecimal constants are normally used as string constants, but they may also be used as value constants. A hexadecimal constant must be immediately preceded by the characters X apostrophe (X') and immediately followed by the apostrophe (') character. For example:

```
                        Hexadecimal Constant
                        ~~~~~~~~~~~~~~~~~~~~
IDEN PREFIX =    X'C1C2C3C4';
```

Character. Character constants are normally used as string constants, but they may also be used as value constants. A character constant must be immediately preceded and immediately followed by the apostrophe (') character. For example:

Character Constant

IDEN PREFIX = 'THIS IS A CHARACTER CONSTANT' ;

If the apostrophe character (') is required in a character constant, it must be defined in some other fashion, such as the hexadecimal constant X '7D'.

When creating character strings in upper case (as with the 9700 EDITOR) the "#" character can be used as an upper/lower case toggle switch. When a "#" character is encountered, the lower case mode is invoked and all letters thereafter will be considered lower case until another "#" character is encountered. Two successive "#" characters (##) are necessary to represent one actual "#" character.

ASCII. ASCII constants are used as string constants. They must be preceded by the characters A apostrophe (A') and followed by an apostrophe character. For example:

IDEN PREFIX = A'ABC';

The ASCII string type allows hexadecimal representation of characters to be embedded in a string. This is done by preceding the hexadecimal representation of the character with an "!" character. For example:

IDEN PREFIX = A'ABC!44EF' is equivalent to IDEN PREFIX = X'414243444546'

Two successive "!" characters (!!) are necessary to represent one actual "!" character.

EBCDIC. EBCDIC constants are used as string contants. They must be preceded by the characters E apostrophe (E') and followed by an apostrophe character. The EBCDIC string type allows hexadecimal representation of characters to be embedded in a character string . This is done by preceding the hexadecimal representation of the character with an "!" character. For example:

IDEN PREFIX = E'ABC!C4EFG' is equivalent to IDEN PREFIX = X'C1C2C3C4C5C6C7'

Octal. Octal constants should be used only as string constants, because of the control program conversion process. Their use as value constants is not prohibited, however. Each 6-bit octal character is converted to an 8-bit octal character, internally, by prefixing two binary zeros. Thus, the arithmetic value of a multiple-character octal constant may be difficult to determine because each digit in the constant has been altered. An octal constant must be immediately preceded by the characters O apostrophe (O'), and immediately followed by the apostrophe (') character.

String constants may be preceded by an optional repeat count. A repeat count is enclosed in parentheses and must be in the range of 1 to 255. The repeat count itself may be any type constant, but a decimal constant is recommended. For example, the statement:

T1: TABLE CONSTANT = (3)'*'; is equivalent to: T1: TABLE CONSTANT = ('***');

Other examples of the use of a repeat count are:

T1: TABLE CONSTANT = (3)O'27';
T2: TABLE CONSTANT = (4)X'C1';

## Command Sets

To simplify the coding of a Job Descriptor Library, PDL statements may be grouped into command sets.

| Command Set | General Purpose |
|---|---|
| SYSTEM | Establishes installation defaults |
| CATALOG | Group sets of PDL statements |
| JOB | Defines how specific print tapes are processed |

The use of these command sets is defined below and illustrated in the coding of a JDL in Figure 3-2 and 3-3. Appendix E illustrates other examples of JDLs.

### SYSTEM Command Set

A SYSTEM Command Set establishes installation dependent requirements and default values for Job Descriptor Entries. This command set is initiated by the "SYSTEM" command[t], which is always the first statement of a JDL. When a PDL statement appears in the SYSTEM command set, it establishes a default value which may be overridden by other usage of the statement within the JDL. This system of overriding parameters is discussed in the section of this chapter entitled "Hierarchy of Replacement". Each SYSTEM command results in the creation of a Job Descriptor Library. No other PDL statements may be placed on the SYSTEM statement.

A SYSTEM command set is terminated by the detection of another SYSTEM (or JDL) command, a CATALOG command, a JOB (or JDE) command, an END command, or physical end-of-input. Detection of an END command, another SYSTEM command, or physical end-of-input causes the PDL task to stop building the current JDL. Detection of two consecutive END commands is interpreted as end-of-input.

The command has the form:

jdl-name: {SYSTEM / JDL};

where

jdl-name is a 1-6 character identifier specifying the name of the JDL to be created. It may be numeric or alphanumeric.

For example:

```
IBMDOS: SYSTEM;
```

identifies the start of a SYSTEM command set and the beginning of a Job Descriptor Library. The identifier (in this case IBMDOS) corresponds to the name used by the operator (via the START command on the keyboard/display) to indicate the Job Descriptor Library to be used when printing a tape. If the jdl-name "DFAULT" is coded, then the specification of a "jdl" parameter on the START command is not necessary. See "Starting a Job" in Chapter 7 for further details.

---

[t] JDL may be used interchangeably with SYSTEM.

### CATALOG Command Set

A CATALOG command allows the user to group PDL statements which may be subsequently referenced by one or more JDEs within a Job Descriptor Library. A CATALOG command set is identified by the CATALOG statement and ends with the appearance of another CATALOG statement or a JOB statement. CATALOG command sets may contain the same statements as appear in JOB command sets.

The CATALOG command has the form

cat-name: CATALOG;

where

cat-name is a 1-6 character alphanumeric identifier of which one character must be alpha. The name will be referenced by JDEs after the CATALOG set has been defined.

For example, the first statement in a CATALOG command set might be:

POWER: CATALOG;

where POWER is the CATALOG identifier used in the INCLUDE left/right pair of a JOB (or JDE) statement.

### JOB Command Set

The JOB[t] command set allows the user to define how the input data is to be processed. In addition to the PDL statements coded within the JOB command set, PDL statements from a CATALOG command set can also be incorporated. Job descriptor values not specified within the JOB or referenced in a CATALOG are indirectly included from the SYSTEM command set. For each JOB, values not specified in any of the command sets are taken from the PDL defaults as defined in Appendix A. These command sets are identified by the JOB (or JDE) command and have the form:

jde-name: $\left\{\begin{matrix}\text{JOB}\\\text{JDE}\end{matrix}\right\}$ [INCLUDE = (cat-name-1, ... ,cat-name-n)] ;

where

jde-name is a 1-6 character identifier specifying the name of the JDE being defined. It may be numeric or alphanumeric.

cat-name-i is a 1-6 character identifier of a previously defined Catalog command set. It must contain at least one alpha character.

---

[t] JDE may be used interchangeable with JOB.

Examples of valid JOB commands are:

```
JOB3: JOB;
JOB3: JDE;
JOB3: JOB INCLUDE = (POWER);
2001: JOB INCLUDE = (POWER, POWERA);
2001: JDE INCLUDE = (POWER, POWERA);
DFLT: JDE;
```

A JOB command set continues until another JOB (or JDE) statement or an END statement is encountered. The identifier in a JOB command set (JOB3 or 2001 in above examples) is used by the operator (along with the identifier on the SYSTEM command set) to initiate a print job from the keyboard/display (see "Starting a Job" in Chapter 7).

A JOB command set with the identifier "DFLT" has special meaning. See section "DEFAULT JDE" in this chapter for further details.

### END Statement

A JDL terminates with the following statement:

```
END;
```

If one Job Descriptor Library is to follow another, then the next statement should be a SYSTEM command. The end of all JDLs to be processed is indicated by two consecutive END statements.

Figure 3-2 provides an overview of a Job Descriptor Library illustrating the use of the SYSTEM, CATALOG and JOB command sets. Figure 3-3 is an example of an actual coded Job Descriptor Library. Appendix E contains examples of JDLs for processing tapes from several different host systems.

```
dd:  SYSTEM;/*      dd IS A 1 TO 6 CHARACTER ALPHANUMERIC THAT IDENTIFIES THIS PARTICULAR JOB
                    DESCRIPTOR LIBRARY.*/

STATEMENT;
STATEMENT;          /* A 'STATEMENT' HERE MEANS ANY PDL COMMAND  */
STATEMENT;

/*                  THE ABOVE STATEMENTS ARE USED TO SETUP DEFAULTS FOR THIS JDL*/

CATA: CATALOG;
     STATEMENT;     /*THESE THREE STATEMENTS WILL BE CATALOGED UNDER THE NAME 'CATA'.  THEY MAY
     STATEMENT;     BE USED IN ANY JDE IN THIS SYSTEM BY USE OF THE NAME CATA */
     STATEMENT;

CATB: CATALOG;
     STATEMENT;
     STATEMENT;     /*A CATALOG IDENTIFIER MUST BE A 1 TO 6 CHARACTER ALPHANUMERIC. */

JOB1:  JOB INCLUDE = (CATA, CATB);

STATEMENT;          /*JOB1 IS THE NAME OF THIS JDE.  IT IS MADE UP OF CATALOG SETS CATA, CATB,
STATEMENT;          AND THE TWO STATEMENTS THAT FOLLOW */

2001:  JDE;         /*A JDE IDENTIFIER MAY BE ALL NUMERIC.  JDE OR JOB MAY BE USED
                    INTERCHANGEABLY*/

STATEMENT;
STATEMENT;          /*JOB 2001 DOES NOT USE CATALOGED COMMAND SETS*/
STATEMENT;

JOB10: JOB;         /*JOB10 IS MADE UP OF PDL DEFAULTS AND THE ABOVE DEFINED SYSTEM COMMAND
                    SET STATEMENTS*/

END;                /*THIS TERMINATES THE JDL*/
```

**Figure 3-2. Overview of PDL Coding**

```
IBMPDL: SYSTEM;

/*                  THIS JOB DESCRIPTOR LIBRARY CONTAINS JOB DESCRIPTOR ENTRIES FOR PROCESSING
                    GRASP, POWER AND POWERVS JOB TAPES

                    THE SYSTEM COMMAND SET DEFINES CONSTANTS AND PROCESSING PROCEDURES THAT
                    WILL APPLY TO ALL JOBS PROCESSED USING THIS LIBRARY UNLESS OVERRIDDEN BY
                    THE CATALOG OR JOB COMMAND SETS*/

VFU001:  VFU        ASSIGN = (1,5), ASSIGN = (2,10), ASSIGN = (3,15),
                    ASSIGN = (4,20), ASSIGN = (5,25), ASSIGN = (6,30),
                    ASSIGN = (7,35), ASSIGN = (8,40), ASSIGN = (9,45),
                    ASSIGN = (10,50), ASSIGN = (11,55), ASSIGN = (12,60),
                    TOF = 5, BOF = 66;
         VOLUME     HOST = POWERVS, PLABEL = YES;
         BLOCK      LENGTH = 2048;
         RECORD     LENGTH = 136, STRUCTURE = VB, LTHFLD = 2, OFFSET = 0,
                    ADJUST = 0, FORMAT = BIN, PREAMBLE = 3;
         LINE       DATA = (1, 132), PCCTYPE = IBM1403, PCC = (0, NOTRAN),
                    OVERPRINT = (PRINT, NODISP), VFU = VFU001;
         ACCT       USER = (BIN, TRAY);                                    } SYSTEM Command Set

/*                  CATALOG COMMAND SET FOR POWER VERSIONS          */

CATPOW:CATALOG;
         VOLUME     HOST = POWER;
         BLOCK      LENGTH = 2048, PREAMBLE = 6, LTHFLD = 2, FORMAT = BIN, OFFSET = 4;
         RECORD     LENGTH = 135, STRUCTURE = VB, PREAMBLE = 2, LTHFLD = 2,
                    FORMAT = BIN, OFFSET = 0, ADJUST = 3;
         /*         CATALOG COMMAND SET FOR GRASP     */                   } CATALOG Command Sets
CATGRP:CATALOG;
         VOLUME     HOST = GRASP;
         BLOCK      LENGTH = 4096, PREAMBLE = 0, ZERO = YES;
         RECORD     LENGTH = 135, STRUCTURE = VB, PREAMBLE = 1,
                    LTHFLD = 1, FORMAT = BIN, OFFSET = 0, ADJUST = 2;

/*                  THE FOLLOWING JDES ARE FOR IBM POWER VS TAPES, POWER VERSION 4.0
                    TAPES, AND POWER VERSIONS 4.1/4.2 TAPES

                    CHARACTERISTICS                 JDE #
                    POWER VS TAPES                    1
                    POWER VERSION 4.0 TAPES           2
                    POWER VERSION 4.1/4.2 TAPES       3
                    GRASP                             4        */

1:JOB;
                    VOLUME    HOST =POWERVS;
2:JOB               INCLUDE = (CATPOW);
                    VOLUME    HOST = POWER;
                    RECORD    LTHFLD = 1, PREAMBLE = 1, ADJUST = 2;
3:JOB               INCLUDE = (CATPOW);
4:JOB               INCLUDE = (CATGRP);
                    VOLUME    HOST = GRASP;                                 } JOB Command Sets
END;
```

Figure 3-3. Job Descriptor Library, Programming Example

## Hierarchy of Replacement

### Introduction

The system job descriptor entry default values shown in Appendix A are the more commonly used values in job tape processing; they can be thought of as a "basic" job descriptor entry. The user needs to code PDL statements only for those parameters which must be changed to process the installation's job tapes. This coding process may be further simplified by placing parameters common to more than one job in a CATALOG command set. When these coding features are properly implemented, it is possible for the same statement to be used in more than one job command set within a library. The PDL processor evaluates these multiple statements and applies the "highest order", error-free definition to the job tape being printed. This process, which is termed the hierarchy of replacement, is demonstrated in the subsequent paragraphs and illustrated in Figure 3–5.

### Hierarchy within a JDL

Figure 3-4 shows a coded job descriptor library which contains four JOB command sets (job descriptor entries). Note that a statement to specify the recording code of the input tape appears in three places:

1. According to the SYSTEM command set, the recording code of the input tape is ASCII

   VOLUME CODE = ASCII;

2. According to the CATALOG command set, the recording code of the input tape is EBCDIC

   VOLUME CODE = EBCDIC;

3. According to the JOB command sets for jobs one and three, the recording code of the input tape is Printable EBCDIC (PEBCDIC). The PDL statement

   VOLUME CODE = PEBCDIC;

   overrides both the CATALOG and SYSTEM command set definitions.

   For job two, the recording code of the input tape is EBCDIC, as specified in the CATALOG command. For job four, the recording code of the input tape is ASCII, since neither the CATALOG nor JOB command set overrides have been coded. The last error-free PDL statement that is encountered ("highest order" definition) will be applied to job tape processed under a given JDE in a Job Descriptor Library.

```
EXAMP2: SYSTEM;
            VOLUME CODE = ASCII;
CATAA:  CATALOG;
            VOLUME CODE = EBCDIC;
            OUTPUT COPIES = 100;
            LINE DATA = (2, 130);
JOB1:   JOB INCLUDE = (CATAA);
            VOLUME CODE = PEBCDIC;
JOB2:   JOB INCLUDE = (CATAA);
            OUTPUT COPIES = 50
JOB3:   JOB INCLUDE = (CATAA);
            VOLUME CODE = PEBCDIC;
JOB4:   JOB;
            OUTPUT COLLATE = NO;
END;
```

Figure 3-4. Hierarchy of Replacement Example

### Non-JDL Hierarchy

The next level of parameter replacement above the JOB command (see Figure 3-5) is the START command (see "Starting a Job" in Chapter 7). Values specified on the START command will override those in the JOB command set. If the "copies" parameter is specified on the START command it will override any other specification. That is, it will override a "copies" DJDE value on tape and a value from a DJDE command file.

If a Dynamic Job Descriptor Entry command file (see "Command File DJDEs" in Chapter 6) is specified on the START command, then its PDL statement values will override parameters on the START command (except for "copies").

If the job tapes to be processed are labeled tapes, then the appropriate tape label information may override programmed values in the Job Descriptor Library. To determine which specific tape label fields are applied, refer to the appropriate vendor format in Xerox Tape Formats Manual No. 910004.

Final modification can be made to job descriptor parameters via DJDE commands contained on the job input tape. These DJDE commands are discussed in Chapter 6.

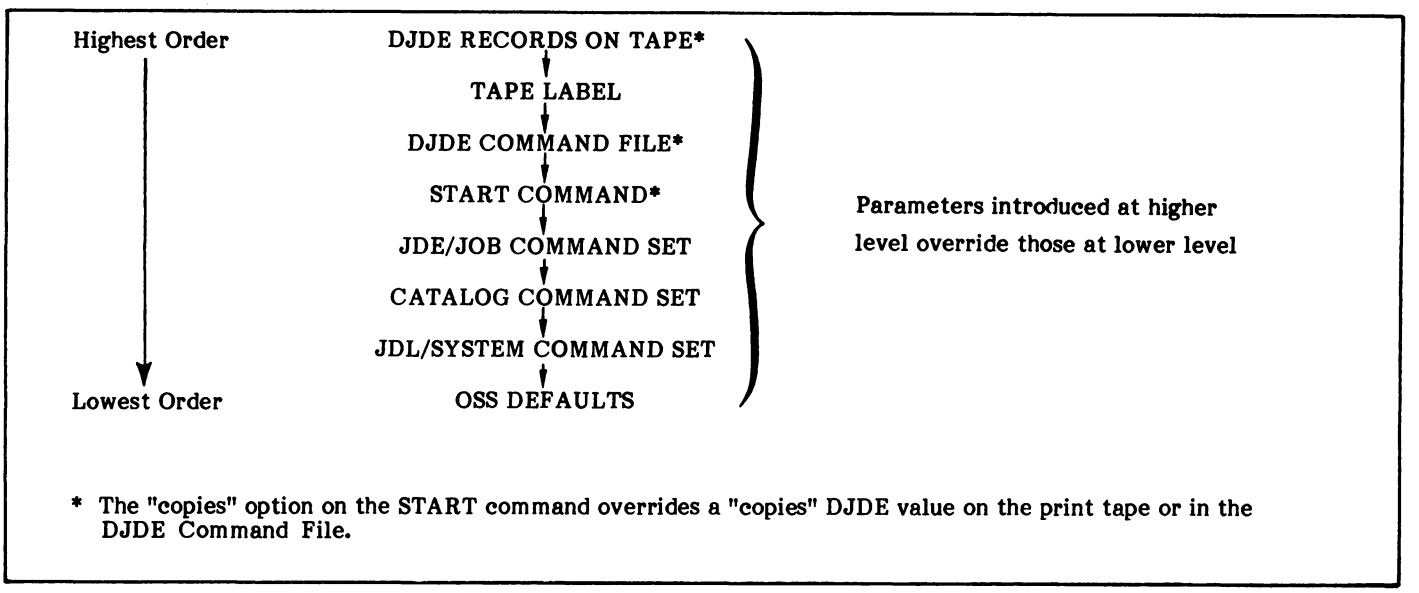



Figure 3-5. Hierarchy of Replacement

In those instances where PDL encounters statements with errors, the hierarchy of replacement is altered somewhat. "Error Processing" is discussed in the subsequent paragraphs.

## Error Processing

### Introduction

The PDL processor checks each statement in a job descriptor library to ensure that is is syntactically correct. It further checks each job descriptor entry to ensure that it is internally consistent. In performing these error-processing functions, PDL determines if an error is "fatal", or "non-fatal". When a fatal error is encountered, the job descriptor library processing will be aborted.

When non-fatal errors are encountered, PDL will validate the job descriptor entry and the entry will be included as part of the job descriptor library. Since those statements which contain errors are ignored, the highest order, error-free statement is applied. If the job descriptor entry adequately describes the job tapes that are to be processed, the errors need not be corrected. Otherwise, it will be necessary to correct the erroneous statements and recompile the library.

### PDL Statement Errors

The PDL processor evaluates a statement from left to right. When a syntax error is encountered in a statement's "left part", PDL stops evaluating the statement and outputs an appropriate error message. Thus, additional errors which may be present in the statement will not be reported. On the other hand, if a "right part" option is out of range, or otherwise invalid, an appropriate error message is output, but evaluation of the statement continues.

### Hierarchy of Replacement in an Errored JDL

The following discussion illustrates the effect of errors when the PDL processor is evaluating a job descriptor library.

Example 1. The VOLUME CODE statement in the SYSTEM command set contains a syntax error (CODE = ASCIII). The system default value, EBCDIC, will be applied to the JOB command set (job descriptor entry JOB1).

```
01:SYSTEM;
     VOLUME CODE = ASCIII;
JOB1:JOB;
     OUTPUT COLLATE = NO;
END;
```

Example 2. The VOLUME CODE statement in the CATALOG command set contains a syntax error (VOLUME CODE = EBDIC). The code default, ASCII, specified in the SYSTEM command set will be applied to the JOB command set (job descriptor entry JOB1).

```
01:SYSTEM;
     VOLUME CODE = ASCII;
AA:CATALOG;
     VOLUME CODE = EBDIC;
     OUTPUT COPIES = 100;
JOB1:JOB INCLUDE = (AA);
     OUTPUT COPIES = 50;
END;
```

Example 3. The VOLUME CODE statement in the JOB command set contains a syntax error (VOLUME CODE = PEBDDIC). The volume code, EBCDIC, specified in the CATALOG command set (catalog AA) will be applied to the JOB command set (job descriptor entry JOB1).

```
01:SYSTEM;
     VOLUME CODE = ASCII;
AA:CATALOG;
     VOLUME CODE = EBCDIC,
     OUTPUT COPIES = 100;
     LINE DATA = (2, 130);
JOB1:JOB INCLUDE = (AA);
     VOLUME CODE = PEBDDIC;
END;
```

## JDL Creation and Compilation

### JDL Creation

Once a JDL has been defined and coded it can be written to tape on a host system and then processed by the PDL processor. It may also be entered directly into the 9700 via the EDITOR (as described in Chapter 8).

JDL files on tape must be fixed, unblocked format with no carriage control and a maximum block/record length of 133 characters. The JDL (or JDLs) must be followed by two tape marks, except where the host system cannot produce tape marks (Honeywell 2000); then two END; cards must be at the end of the JDLs.

A JDL on tape may be copied (see COPY command in Chapter 7) into a 9700 disk file and then modified (if necessary) with the EDITOR as with EDITOR created files.

### JDL Compilation

A Job Descriptor Library can be compiled by the PDL processor from a disk file or from tape as illustrated in Figure 3-6. If the JDL is on tape, PDL will first copy the source JDL file to disk and then compile it. Source JDL files from tape are cataloged on disk in the Job Source Library (file type JSL). The outputs of the compilation are 1) a listing of the source statements with error diagnostics and 2) a Job Descriptor Library control file that can be selected by the operator (on the START command) to print a particular job tape.

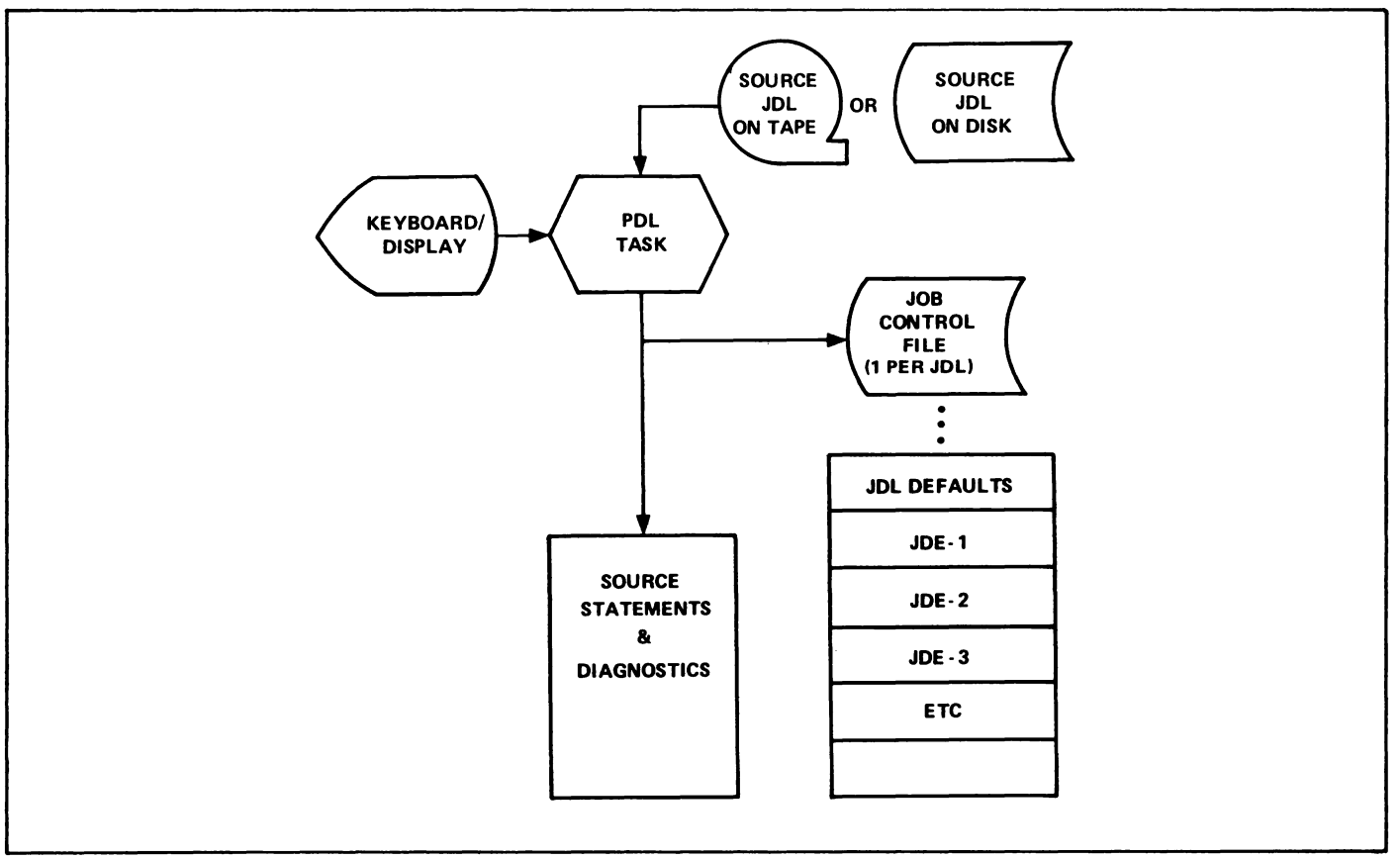



Figure 3-6. PDL Creates JDL Control File

For the PDL processor to compile the JDL from tape (or disk file), the following command is entered on the keyboard/display:

PDL [file-name] [,NOPRINT] [,NOSOURCE] [,REPLACE / ,NOREPLACE] [,TRAY] [,NRWD] [,DISPLAY]

where

file-name — specifies the 1 to 6 character name of a source Job Descriptor Library file which is to be input to the PDL task. If no filename is specified, source input for PDL will be read from tape and a JDL disk file will be created for each SYSTEM command encountered.

NOPRINT — specifies that only source records which contain errors, the diagnostics which apply to those lines, and the PDL summary reports are to be printed during compilation. If there are no errors there will be no print out. The default is to print all of the above plus the PDL source records.

NOSOURCE — specifies that no JDL source files are to be created when the input is from tape. This option has no effect when input is not from tape.

REPLACE — specifies that the PDL task may replace an existing job control file with a new output file of the same name. This is the default.

NOREPLACE — specifies that the PDL task is to abort the writing of a file if it has the same name as an existing file. REPLACE is the default.

TRAY — specifies that the listing and diagnostics from the PDL compilation are to go to the sample print tray. Bin is the default.

NRWD — specifies that no rewind is to occur after a tape file is processed. The default is to rewind.

DISPLAY — All PDL messages are displayed on the keyboard/display. The default is to print the messages on the PDL listing.

For example, the command:

PDL ,NOSOURCE,NOPRINT

will compile JDL file(s) from tape into JDL control file(s) or replace previously existing files with the same names if those already exist on disk with same SYSTEM name(s). The diagnostic listings (not the source) to the [illegible] will be printed after the JDL file(s) are processed.

The command:

PDL H2SYS,TAY

will be initiated from the keyboard/display. It will compile the disk file H2SYS (file type JSL), create a JDL file in the JDL file directory (or replace a previous JDL control file) and print the source listing and diagnostics on the sample print tray.

### Default JDE

A default Job Descriptor Entry can be defined by the user in a Job Descriptor Library. This default JDE is useful when initiating a print job with the START command in that the "jde" parameter (see "Starting a Job" in Chapter 7) need not be specified (as well as other options on the START command for other defaults). The identifier of a user created default JDE must be "DFLT". This default JDE will be handled in the JDL as any other user defined JDE (as defined previously in "JOB Command Set" of this chapter).

### PDE and CME Compilation

The PDL processor also compiles PDE and CME source files. PDE and CME source statements are created by the user, cataloged in the JSL directory and processed by PDL just as with previously discussed JDL files. After PDL processing, a control file is created and cataloged on disk in the CME or PDE file directory. These control files are referenced in a job descriptor library the same as if the CME or PDE statements were coded into it. When the system finds the CMEs or PDEs are not part of the job descriptor library it will go to the control files on disk for the information. CMEs are referenced by the MODIFY left part of the OUTPUT command. PDEs are referenced by the FORMAT left part of the OUTPUT command. Further details on PDEs and CMEs are contained in Chapter 5.

### Running a Job Tape

The START command is used to initiate the printing of a job tape. This command is entered by the operator on the keyboard/display console along with other optional parameters. An example of the START command is as follows:

```
START   JDE10,H2SYS
```

where

H2SYS is a job control file created by PDL as a result of compiling the source job descriptor library file H2SYS.

JDE10 is the name of Job Descriptor Entry within the JDL file H2SYS.

The syntactical details of the START command (see "Starting a Job") are defined in Chapter 7 along with other commands which enable the operator to control the flow of print jobs through the system.

# 4. INPUT PROCESSING FUNCTIONS

## Introduction

For off-line operations the input medium to the 9700 is magnetic tape which may be recorded in one of a variety of standard vendor formats. The PDL programmer defines the tape input deblocking and record format parameters which reduce physical tape blocks first to logical records, then to print lines. Further, special processing options can be selected which facilitate report processing or enhance the appearance of the report output.

Prior to selecting the tape options which are to be applied to a job descriptor entry to describe a job tape, certain introductory tape structure concepts must be understood. These concepts, discussed in the first section of this chapter, will enable the PDL programmer to readily define the following job tape characteristics:

- Tape Code (the CODE and LCODE left/right parts).
- Host Computer Formats (the HOST left/right parts).
- Record Structure (includes STRUCTURE, BLOCK, and RECORD options in a sample job tape).
- Packed Data Formats (interpretation and use).

A discussion of the PDL statements used to actually select these options follows the introductory material.

## Tape Structure Concepts

### Tape Codes

The tape codes recognized by the printing system are EBCDIC, ASCII, several versions of BCD, and USER-defined translation. Tables showing the correspondence between standard recorded codes and printed characters are contained in Appendix C.

### Host Computer Formats

The printing system processes the various types of standard host input tapes listed below. The format of each of these types of tapes is described within the 9700 Tape Formats Manual (Publication No. 91 00 04).

- American National Standards Institute (ANSI)
- IBM OS/360 and DOS/360 Standard Labels
- IBM DOS/360 GRASP.
- IBM DOS/360 POWER
- IBM POWER VS
- Xerox Tapes with Xerox Carriage Control
- Univac Series 70 Standard Labels
- Honeywell 200/2000 Labeled Tapes
- Honeywell 600/6000 Standard System Format Labels
- Burroughs Medium System Labeled Tapes
- Burroughs Large System ANSI Labeled Tapes
- Univac SDF Tape Format
- IBM/OS Writer Stacked Report Output

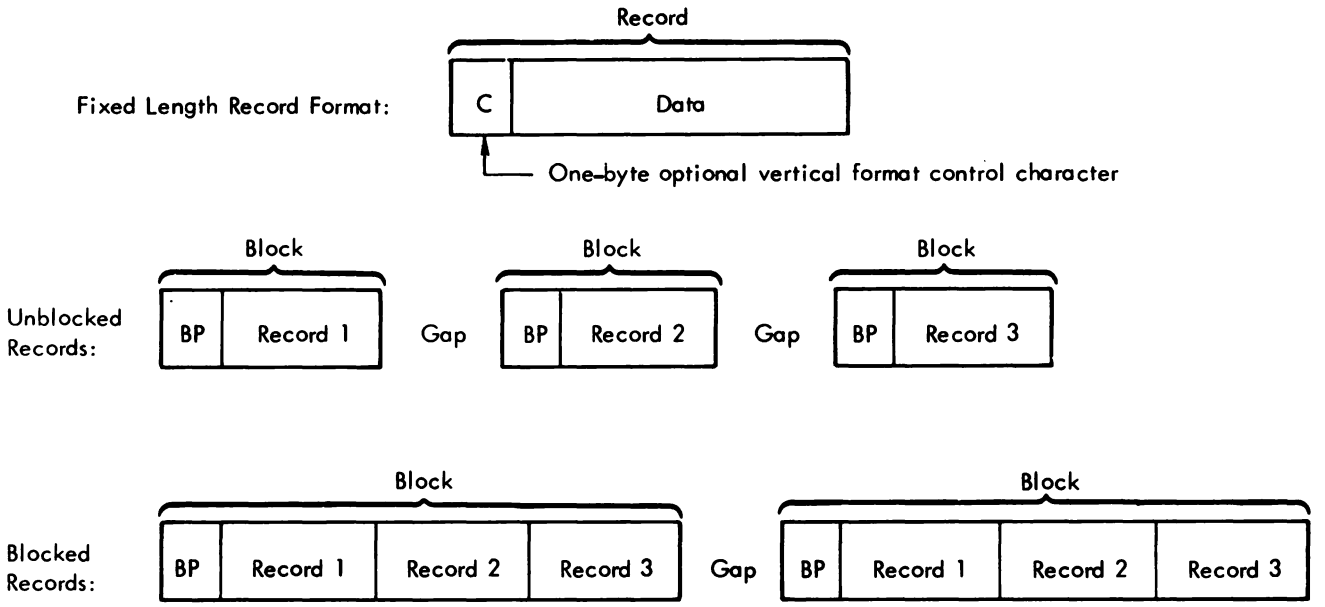
## Record Structure

All records input to the printing system are either blocked or unblocked with the fixed length, variable length, or undefined format. These record formats are illustrated in Figures 4-1, 4-2, and 4-3, respectively. In some cases, variations of the record formats can be processed on the printing system, providing the variation is rigorously observed. The GRASP tape format is an example of such a variation. Tape label contents may also describe blocking and record structure. In some cases, tape label contents override BLOCK and RECORD parameters specified in the JDL. These labels are described in the 9700 Tape Formats Manual.

A record is arbitrarily divided into two portions. The operating system's portion of the record contains information supplied by the host operating or spooling system. The user's portion of the record contains information provided by the applications or user's program running on the host system. The boundary between the two portions of the record is traditionally between the record length and printer carriage control field, such that, if there is no record length field, there is no operating system portion of the record.

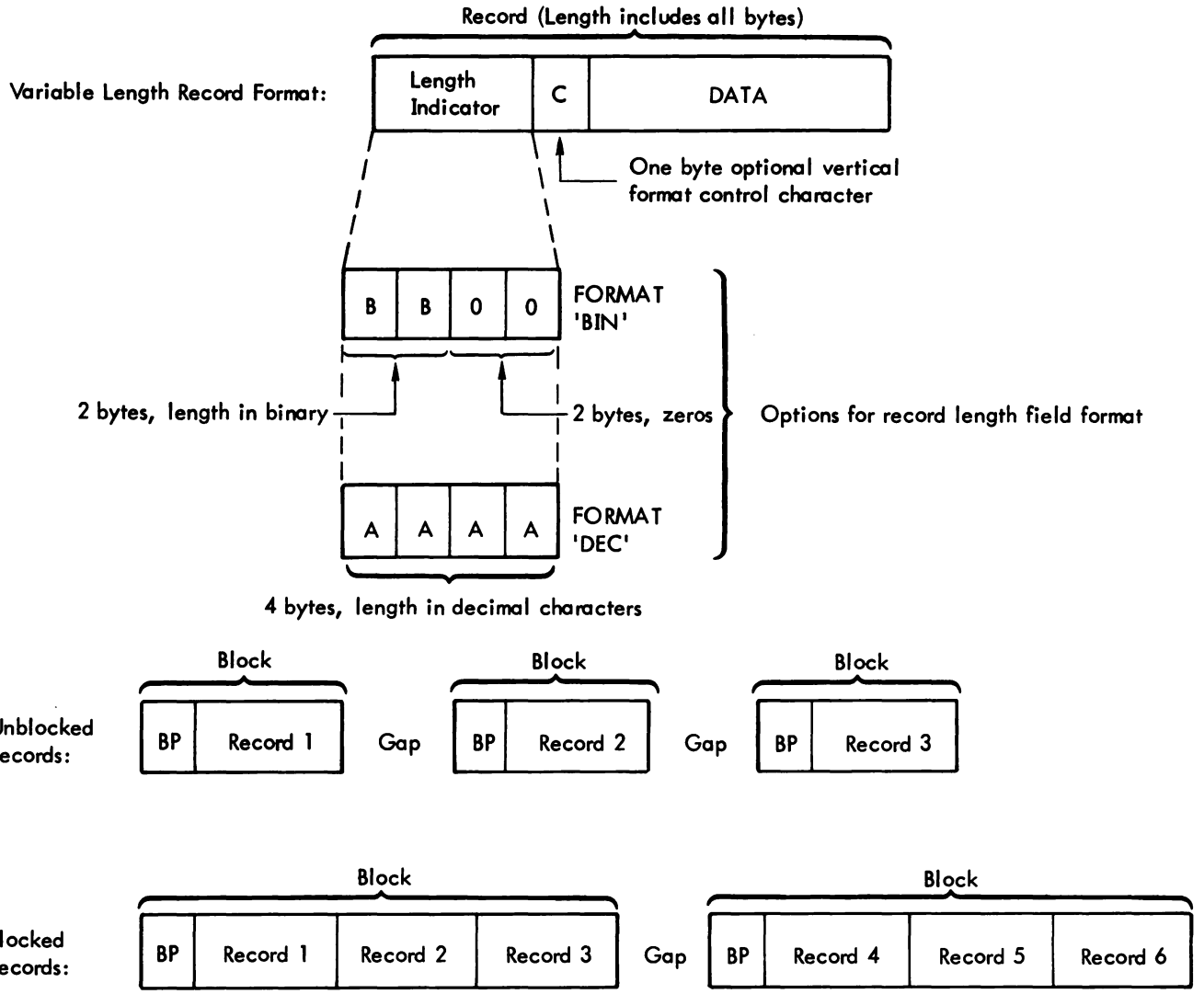
A record is arbitrarily divided to simplify job descriptor entry definition of relatively constant information such as offsets to carriage control, print data, and/or stacked report fields from a boundary which will not change as a result of a small change to the program job control language (or as a result of an operator command to start-up a writer or spooler). In other words, operating system portion of a record changes a great deal in the fixed blocked to variable blocked format transition, but the user portion of the record does not change at all.

A sample input block is broken down into its component parts in Figure 4-4. PDL statements used to define each of the components shown in this record are described in the syntax and interpretation tables for the BLOCK, RECORD, and logical processing statements in this chapter, and the LINE statement in Chapter 5, "Output Processing".



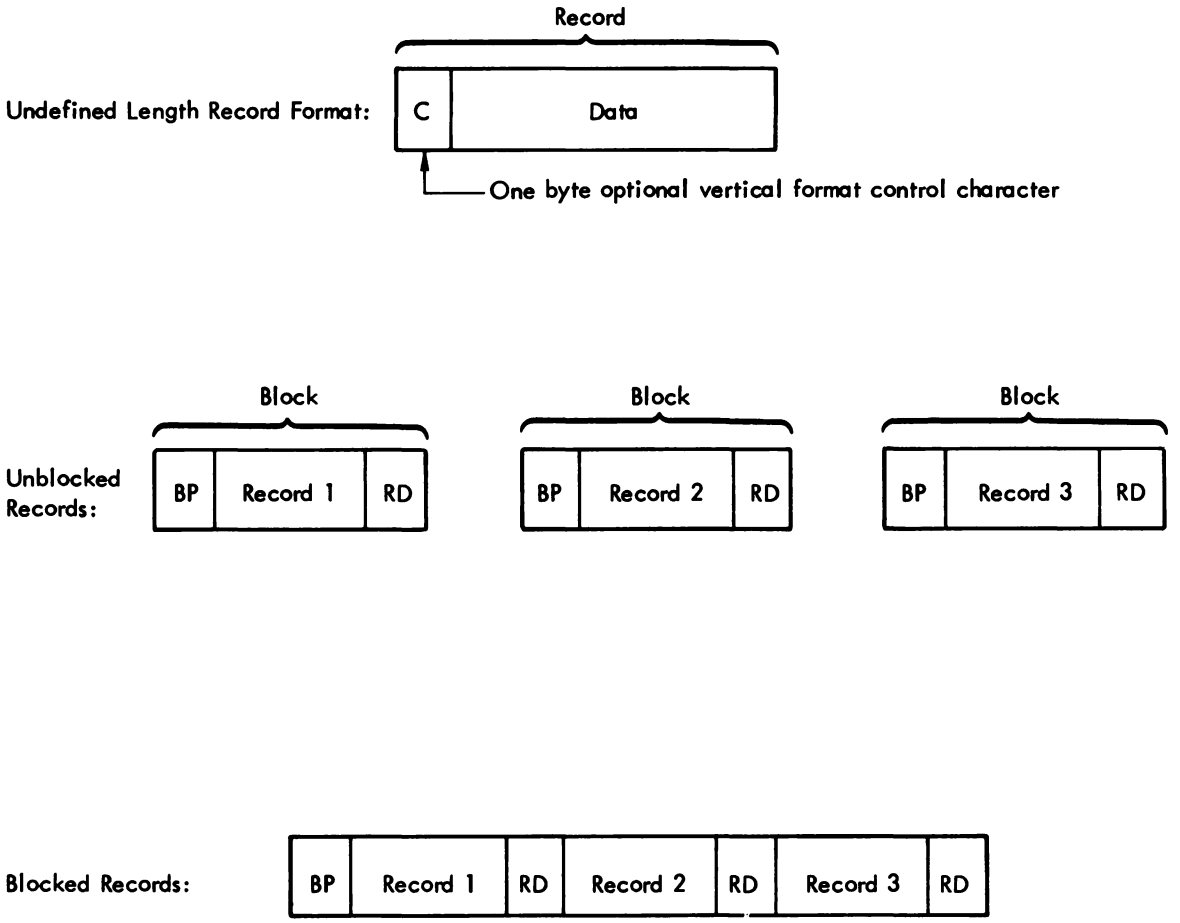

Notes:
1. BP=Optional Block Preamble. (Note: The block postamble is also optional.)
2. All records within a file are of the same length.
3. Blocks may be truncated, but only at the end of a record. Truncated blocks must be an even multiple of the record length plus the block preamble and postamble (if any).
4. Gap=Interblock Tape Gap.

Figure 4-1. Fixed Length Records




Notes:
1. BP = Block Preamble. The block preamble may contain a block length field of the same format as the record length field.
2. Blocks may be truncated, but only at the end of a record.
3. The block may optionally contain a postamble.

Figure 4-2. Variable Length Records

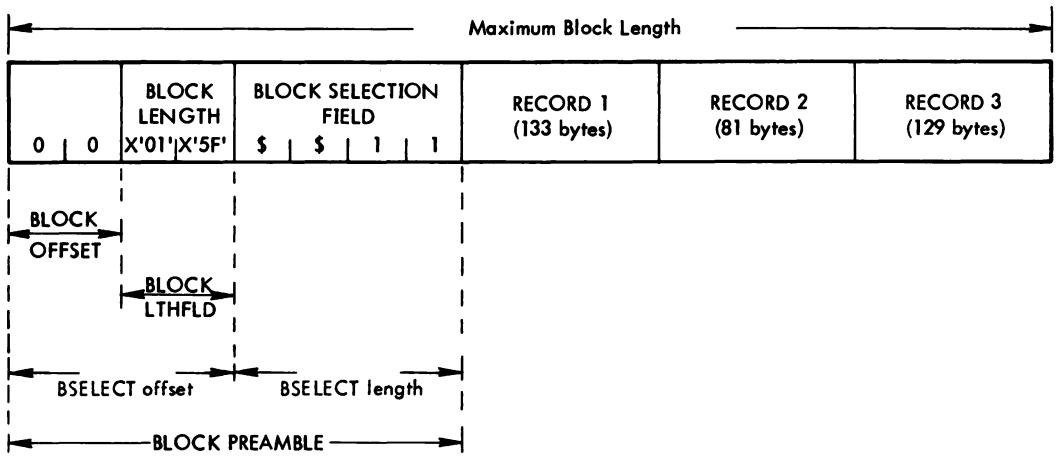



Notes: BP = Optional Block Preamble. (Note: Block postamble is also optional.)

RD = Record Delimiter. The record delimiter is a string of from one to four characters (optional for unblocked format).

Figure 4-3. Undefined Length Records

BLOCK




Interpretation

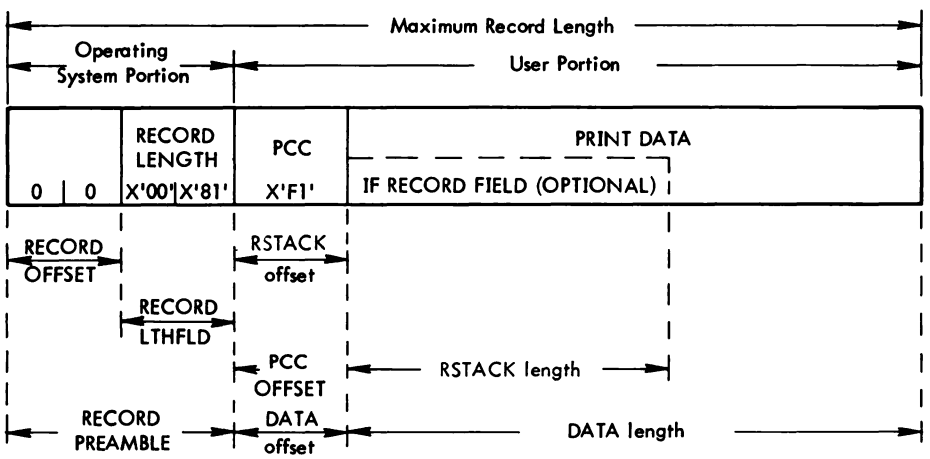
1. The BLOCK statement would be coded as follows:

```
BLOCK LENGTH = 351,
      OFFSET = 2, ADJUST = 0,
      LTHFLD = 2,
      PREAMBLE = 8,
      FORMAT = BIN;
```

2. The logical processing statements for block selection would be coded as follows:

```
T1: TABLE     CONSTANT = ('$$11');
C1: CRITERIA  CONSTANT = (4, 4, EQ, T1);
    BSELECT   TEST = C1;
```

RECORD




Interpretation

1. The RECORD statement would be coded as follows:

```
RECORD LENGTH = 133, OFFSET = 2,
LTHFLD = 2, PREAMBLE = 4, ADJUST = 4,
FORMAT = BIN;
```

2. The logical processing statements for stacked reports would be coded as follows:

```
T1: TABLE     CONSTANT = ('IF RECORD FIELD');
C1: CRITERIA  CONSTANT = (1, 15, EQ, T1);
    RSTACK    TEST = C1, DELIMITER = YES, PRINT = BOTH;
```

3. The LINE statement (discussed in Chapter 5, "Output Processing") would be coded as follows:

```
LINE PCC = (0, NOTRAN), PCCTYPE = ANSI,
DATA = (1, 132),
VFU = V1;
```

Figure 4-4. Components of a Block and a Record

### Packed Data Formats

Six-bit characters may be written onto a 9-track tape in a 4X3 packed (or compressed) format. That is, four 6-bit data bytes are compressed into three 8-bit data bytes. Two methods of packing these bits together exist. One method is used by Honeywell 6000 users and is specified by the PDL left/right parts UNPACK = T4X3. Honeywell 2000 users employ a slightly different method of packing which is specified by UNPACK = T4X3H2

Whenever an unpacking method is included in the job descriptor entry, the system unpacks the characters prior to data processing. Each 6-bit character is extracted and 2 high order zeros are appended. This unpacked character may then be translated if the CODE = "right part" option is also coded.

Figure 4-5 shows an example of how 6-bit characters packed in the T4X3 method are unpacked and then translated to ASCII by the system. Figure 4-6 shows an example of unpacking and translating the T4X3H2 method.

Normally after data is unpacked it must be translated. The character code set is defined in the CODE = "right part" option. For a 4X3 unpacking method, the data is generally BCD and either of the 3 standard BCD sets (H2BCD, H6BCD. and IBM-BCD) can be used.

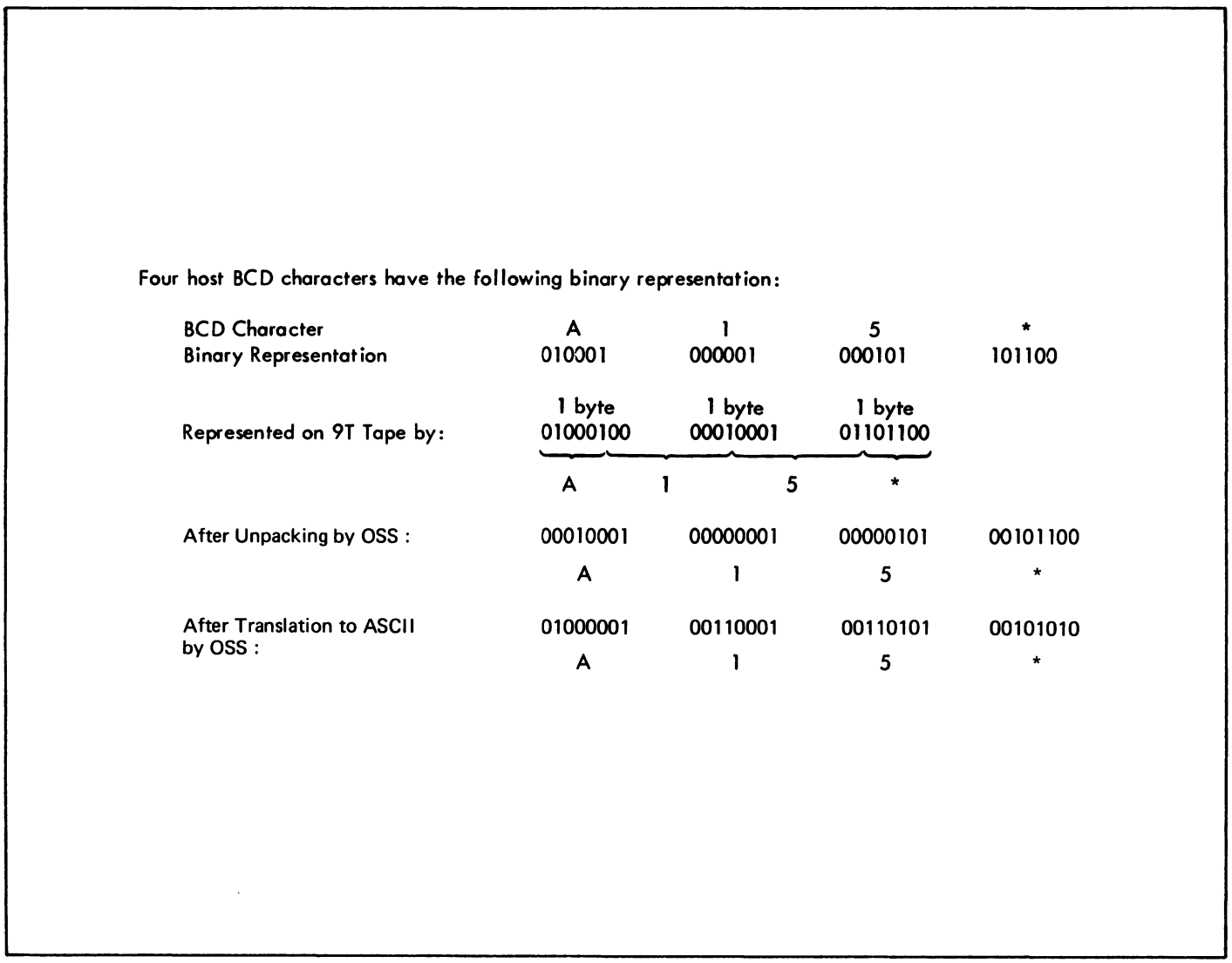



Figure 4-5. Pictorial Representation of T4X3 Packing

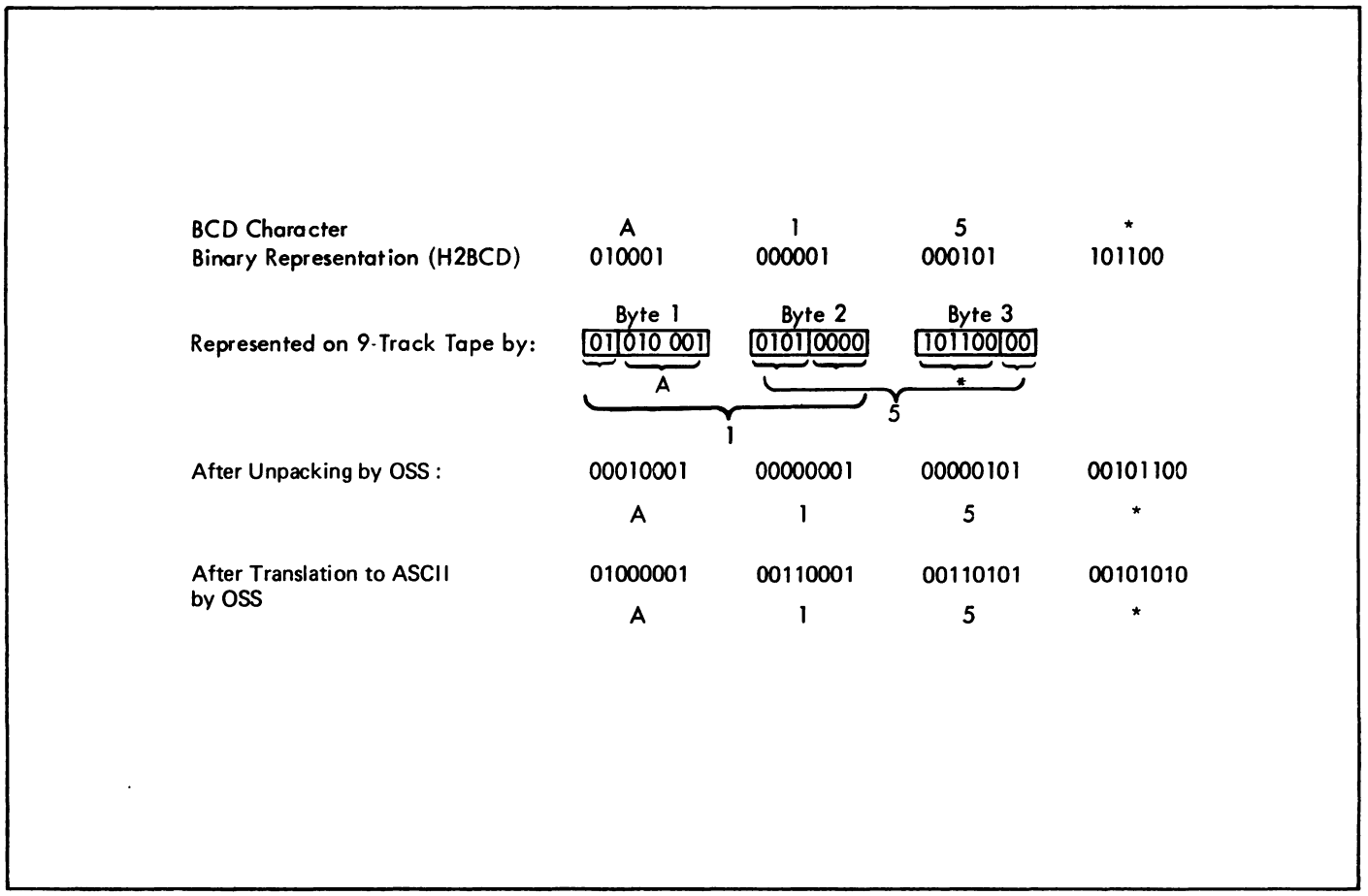



Figure 4-6. Pictorial Representation of T4X3H2 Packing

## Data Tape Characteristics

The data tape characteristics are defined by the VOLUME, CODE, BLOCK, and RECORD statements. These statements, in effect, enable the system to extract output print lines from the input data stream. They are defined on the following pages.

**VOLUME Statement**

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| VOLUME | HOST = | ANSI<br>IBMOS<br>IBMDOS<br>GRASP<br>POWER<br>POWERVS<br>OSWTR<br>US70<br>B2500<br>B2700<br>B3500<br>B3700<br>B4700<br>B6700<br>H2000<br>H6000<br>DUMP<br>OCTDUMP<br>UNIVAC<br>XEROX | IBMOS | The right part can be any one of the host options. |
| | UNPACK = | T4X3<br>T4X3H2<br>UNIVAC<br>NONE | NONE | The right parts "T4X3" and "T4X3H2", specify different unpacking routines, whereby 6-bit characters are extracted and then restored as 8-bit bytes with the new left-most bits set to zero. "T4X3" is used primarily for Honeywell 6000 tapes, "T4X3H2" is used for Honeywell 2000 tapes, and UNIVAC is used for UNIVAC tapes, although any of these routines can be specified independently of the HOST type. These routines are described in a previous section of this chapter.<br><br>The right part "NONE" indicates that no unpacking operation is to be performed. |
| | LABEL = | label | STANDARD | The right part "label" can be any one of the following tape label options:<br><br>NONE (the input tape is unlabeled)<br>ANSI<br>STANDARD<br>SPR (Honeywell 2000 System Print Tape)<br>COBOL (Honeywell 2000 COBOL Tape with 120-byte labels) |

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| VOLUME (cont.) | CODE = | code | EBCDIC | The left part "CODE =" specifies that the input tape data is to be translated prior to output.<br><br>The right part "code" can be any one of the following options:<br><br>ASCII<br>BCD<br>BCL (Burroughs BCD)<br>CBCD (CDC BCD)<br>EBCDIC<br>PEBCDIC (wrap-around, printable EBCDIC)<br>H2BCD<br>H6BCD (same as BCD)<br>IBMBCD<br>USER (single user-defined code translation)[t]<br>id (user-defined code translation)[t] |
| | LCODE = | code | EBCDIC | The left part "LCODE = " specifies that the input tape labels are to be translated.<br><br>The right part "code" can be any one of the following options:<br><br>ASCII<br>BCD<br>BCL (Burroughs BCD)<br>CBCD (CDC BCD)<br>EBCDIC<br>PEBCDIC (wrap-around, printable EBCDIC)<br>H2BCD<br>H6BCD (same as BCD)<br>IBMBCD<br>USER (single user-defined code translation)[t]<br>id (user-defined code translation)[t] |
| | EOV = | ( {PAUSE / NOPAUSE} , {EOF / NOEOF} ) | (NOPAUSE, NOEOF) | The left part "EOV =" specifies action to be taken by the control program when the end-of-volume point is encountered on the input tape. |

t The standard codes are shown in Appendix C. Creation of a USER translation table referenced by either the keyword USER or an identifier is defined using the CODE statement. See the subsequent section of this chapter "CODE Statement".

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| VOLUME (cont.) | EOV = (cont.) | | | If "PAUSE" is specified, a message will be displayed when EOV is encountered. A CONTINUE response by the operator will cause the tape to be rewound and the normal volume change sequence to be performed. If NOPAUSE is specified, the rewind will be issued as soon as the EOV label is processed.<br><br>If "EOF" is specified, the end-of-volume label will be treated as an end-of-file label. When this occurs, the first part of the volume spanning page is output as the last page of the job. (Note: The second part of the spanned page is printed as the first job page when the next volume is started, with possible page format irregularity.)<br><br>If "NOEOF" is specified, normal end-of-volume processing results. |
| | PLABEL = | {YES / NO} | NO | The left part "PLABEL =" is used to specify whether or not the tape labels on a labeled input tape are to be printed.<br><br>If "YES" is specified, all tape labels (except those encountered during a volume change) are printed on an output page as they are encountered. The output page is delivered to the sample print tray.<br><br>If "NO" is specified, no tape label printing results. |
| | OSCHN = | value | 9 | The left part "OSCHN =" is meaningful only if the statement HOST = OSWTR has been previously coded.<br><br>The right part "value" specifies the vertical format unit (VFU) channel which is used to signal the end of a report generated by the IBM OS Writer.<br><br>When a skip to the specified channel occurs (as determined by the printer carriage control field within a logical record), AND an overprint operation immediately follows the skip-to-channel operation, the IBM OS Writer banner page is considered found. |

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| VOLUME (cont.) | OSTLP = | value | 0 | The left part "OSTLP" is meaningful only if the statment HOST = OSWTR has been coded.<br><br>The right part "value" specifies the number of trailer banner pages which are to follow the end of a report created by the IBM OS Writer. |
| | OSHDP = | value | 0 | The left part "OSHDP =" is meaningful only if the statement HOST = OSWTR has been coded.<br><br>The right part "value" is the number of header pages which precede the report. |
| | BMULT =<br>RMULT = | value<br>value | 1<br>1 | The "BMULT =" and "RMULT =" statements provide for the specification of multiplication factors which are applied to the maximum block (BMULT) and record (RMULT) length values extracted from the header tape label.<br><br>The right part "value" is an integer in the range $1 \leq \text{value} \leq 15$. |
| | RMODE = | $\left\{ \begin{matrix} S \\ M \end{matrix} \right\}$ | M | The left part "RMODE =" is used to control processing of multi- or single-report processing.<br><br>The right part "S" sets up the print job to be run in single-report mode.<br><br>The right part "M" sets up the print job to be run in multi-report mode.<br><br>The operator can override the "RMODE" value on the START command. See "Starting a Job" in Chapter 7. |

Points to Note

1. The input statement LABEL = "label" should not be coded in cases where it is not appropriate. For example, if the statement HOST = GRASP is coded, no LABEL statement is required, and, if one is coded, it is ignored.
2. If the HOST left part and its associated right parts are inconsistent with the LABEL left part and its associated right parts, an error message is output during job descriptor library processing, and a valid label right part is automatically substituted. The HOST right part which is specified is assumed to be correct; PDL substitutes a valid label. Table 4-1 summarizes the label specifications that are valid for each host type. The shaded areas of the table show the label that is substituted by PDL when an invalid host/label pair is specified.
3. A tape to be dumped is treated as unlabeled (the statement HOST = DUMP or HOST = OCTDUMP is coded).

Table 4-1. Valid Host Computer/Label Specification

| HOST TYPE | LABEL SPECIFICATION | | | | |
|---|---|---|---|---|---|
| | Unlabeled | ANSI | STANDARD | SYSTEM PRINT | COBOL |
| IBMOS | X | X | X | | |
| IBMDOS | X | X | X | | |
| GRASP, POWER, POWERVS | LABEL SPECIFICATION IS IGNORED | | | | |
| OSWTR | X | X | X | | |
| US70 | X | X | X | | |
| XEROX | X | X | X | | |
| B2500 Burroughs Medium Systems | X | X | X | | |
| B2700 | X | X | X | | |
| B3500 | X | X | X | | |
| B3700 | X | X | X | | |
| B4700 | X | X | X | | |
| B6700 Burroughs Large System | X | X | X | | |
| H2000 Honeywell 200/2000 Series | | | X | X | X |
| H6000 Honeywell 600/6000 Series | | | X | | |
| DUMP | LABEL SPECIFICATION IS IGNORED | | | | |
| OCTDUMP | LABEL SPECIFICATION IS IGNORED | | | | |
| UNIVAC | X | X | X | | |
| ANSI | | X | | | |

The shaded areas of this table show the label that is substituted by PDL when an invalid host/label pair is specified.

**CODE Statement**

This statement must be coded whenever the CODE/LCODE = USER or CODE/LCODE = id left/right part(s) appear in the VOLUME statement. The CODE statement by itself causes no action to occur except to define a user code translation table.

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| [ac:] CODE | DEFAULT = | { ASCII<br>BCD<br>BCL<br>CBCD<br>EBCDIC<br>PEBCDIC<br>H2BCD<br>H6BCD<br>IBMBCD<br>value } | EBCDIC | "ac" is a 1 to 6 character statement identifier that is referenced by the CODE/LCODE = id portion of the VOLUME command. One character must be alphabetic; the other characters alphabetic or numeric.<br><br>The "DEFAULT" left part enables the user to specify a base code from which code assignment exceptions can be readily made. The base code is specified from the right part selections of the statement; the exceptions are specified via the "ASSIGN =" statement, described below.<br><br>The right part "value" is a string of characters which is a one byte hexadecimal, octal, or character constant. |
| | ASSIGN = | { (input, output)<br>(input, (output,....,output)) } | none | The right part "input" defines the input code; the right part "output" defines the corresponding output code. |

Points to Note

1. Multiple user-defined code translation tables are allowed; only one may be without a statement identifier. The corresponding CODE/LCODE left part on the VOLUME command references each user table via a statement identifier. The right part USER is used to reference the user-defined code translation table in which no statement identifier is coded.

2. The "DEFAULT =" statement must be coded prior to any "ASSIGN =" statements. A "DEFAULT =" statement following any defined corresponding input/output codes causes this correspondence to be replaced by the DEFAULT assignment.

3. An example of statement coding used to modify a base code set is shown below.

   A user's job input tape is recorded in EBCDIC. On output, however, codes 5B, 5C and 5D (characters $ * ) respectively) are to be assigned to the character blank (X'40'). The coded statements to effect this input/output modification would be as follows:

   CODE DEFAULT = EBCDIC, ASSIGN = (X'5B', X'40'), ASSIGN = (X'5C', X'40'), ASSIGN = (X'5D', X'40');

   or, alternatively,

   CODE DEFAULT = EBCDIC, ASSIGN = (X'5B', (X'40', X'40', X'40'));

   Thus, in the latter case, consecutive input codes need not be specified to accomplish code modification.

### BLOCK Statement

The BLOCK statement is used to describe the physical structure of tape blocks.

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| BLOCK | LENGTH = | value. | 1330 | The right part "value" specifies the length, in bytes, of the longest physical block in the file. "value" is an integer in the range 12 ≤ value ≤ 12,288.<br><br>The tape label contents may override a coded "LENGTH =" command. |
| | LTHFLD = | size | 0 | The right part "size" specifies the length, in bytes, of the field containing the block length right part "value" specified above. "size" is an integer in the range 0 ≤ size ≤ 5.<br><br>If "size" is set to zero, the block length field is not considered to be part of the block, and the length of a block, on tape, is the actual block length. |
| | OFFSET = | value | 0 | The right part "value" specifies the length, in bytes, of the block length field offset. This offset is the number of bytes from the first byte of a block to the block length field.<br><br>"value" is an integer in the range 0 ≤ value ≤ (LENGTH-LTHFLD) -1. |
| | FORMAT = | type | BIN | The right part "type" specifies the recorded mode of the block length field.<br><br>"type" must be one of the following:<br><br>BIN (Binary)<br>DEC (Decimal)<br>PACK (Packed with no sign)<br>PKSG (Packed with sign) |
| | ADJUST = | value | 0 | The right part "value" specifies the block adjustment length. This length is a constant integer which is to be added to or subtracted from the value in the block length field of every tape block. The resulting value is the true block length.<br><br>The range for "value" is -127 ≤ value ≤ 127 and must be ≤ LENGTH.<br><br>The character plus (+) or minus (-) may be used to specify a positive or negative adjustment. |

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| BLOCK (cont.) | PREAMBLE = | length | 0 | The right part "length" specifies the block preamble length. This length is the operating system portion of the record in the block. It is the byte offset from the first byte of a tape block to the first byte of the first logical record.<br>"length" is an integer in the range 0 ≤ length ≤ LENGTH. |
| | POSTAMBLE = | length | 0 | The right part "length" specifies the length, in bytes, of any extraneous data at the end of all tape blocks.<br>In effect, "length" is an offset from the end of a block backwards to the end of the last logical record.<br>"length" is an integer in the range 0 ≤ length ≤ LENGTH. |
| | CONSTANT = | sc | none | The "CONSTANT" statement is used when tape blocks are to be delimited by block delimiter constants. This indicates that the constant and the data following the constant are ignored until the end of the block is reached.<br>The right part "sc" specifies a string, hexadecimal, or octal character constant as described in Chapter 3. The length of the constant must be from one to four bytes. |
| | ZERO = | {YES / NO} | NO | The right part "YES" specifies that the end of a tape block is indicated by a value of zero in the record length field (before applying the record length adjustment; see the "ADJUST =" statement described previously). Data which follows the record will be ignored up through the end of the block.<br>The right part "NO" indicates that the end of a tape block is not indicated by a value of zero in the record length field. |
| | LMULT = | value | 1 | The right part "value" specifies a multiplication factor to be applied to the block length. The value specified will be multiplied by the value in the length field (see "LENGTH =" statement previously described) to compute the number of bytes in the block.<br>"value" is an integer in the range 1 ≤ value ≤ 15. |

Points to Note

1. The BLOCK LENGTH left/right parts may be overridden by ANSI, IBM OS/Standard, or Honeywell 2000 COBOL labels which specify block length.

2. The values for BLOCK LTHFLD, OFFSET, FORMAT, and PREAMBLE left/right parts may be overridden if RECORD STRUCTURE is changed as the result of ANSI, IBM OS/Standard, or Honeywell 2000 COBOL label processing.

3. The LENGTH on a 4X3 packed format tape is the number of 6-bit bytes or characters in the tape block. For Honeywell 6000 ASCII tapes, the LENGTH is still the number of 6-bit characters in the block.

4. The length of the block delimiter constant should not be coded as the block postamble. Both lengths will be subtracted from the end of the block.

5. The search for the block delimiter constant will start after the block preamble and proceed forward to the first appearance of the constant.

### RECORD Statement

The RECORD statement describes the characteristics of the logical tape records.

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| RECORD | STRUCTURE = | type | FB | The "STRUCTURE" statement describes the general record structure for a file.<br><br>The right part "type" may be any of the following structures:<br><br>F Fixed length.<br>FB Fixed length blocked.<br>V Variable length.<br>VB Variable length blocked.<br>U Undefined length.<br>UB Undefined length blocked.<br><br>The tape label contents may override the STRUCTURE left/right parts. |
| | LENGTH = | value | 133 | The right part "value" specifies the length, in bytes, of the longest logical record in the file. "value" is an integer in the range $1 \leq$ value $\leq$ BLOCK LENGTH.<br><br>The tape label contents may override the LENGTH statement. |
| | LTHFLD = | size | 0 | The right part "size" specifies, in bytes, the record length field length.<br><br>"size" is an integer in the range $0 \leq$ size $\leq 5$. If size is set equal to zero, then record lengths are not contained in the records, and the record length is the maximum length ("LENGTH =") for each record. |
| | OFFSET = | value | 0 | The right part "value" specifies the record length field offset. This offset is the byte offset from the first byte of the record to the record length field.<br><br>"value" is an integer in the range $0 \leq$ value $\leq$ (LENGTH-LTHFLD) -1. |
| | FORMAT = | type | BIN | The right part "type" specifies the format of the record length field to be one of the following:<br><br>BIN (Binary)<br>DEC (Decimal)<br>PACK (Packed with no sign)<br>PKSG (Packed with sign) |

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| RECORD (cont.) | ADJUST = | value | 0 | The right part "value" specifies the record adjustment length.<br><br>"value" is a constant integer which is to be added to or subtracted from the value in the length field of every record (see "LENGTH =" statement described previously). The range "value" is $-127 \leq$ value $\leq 127$ and must be $\leq$ LENGTH.<br><br>The first character of the "value" right part may be plus (+) or minus (-). |
| | PREAMBLE = | length | 0 | The right part "length" specifies the record preamble length. This length is the byte offset from the first byte of the record to the first byte of the user's portion of the record.<br><br>"length" is an integer in the range $0 \leq$ length $\leq$ LENGTH. |
| | POSTAMBLE = | length | 0 | The right part "length" specifies the length, in bytes, of any extraneous data in a postamble at the end of a user's record. "length" is an integer in the range $0 \leq$ length $\leq$ LENGTH. |
| | CONSTANT = | sc | none | The "CONSTANT" statement is used when tape records are to be delimited by record delimiter constants. The record delimiter constant string signals the end of the record, but it is not included in the print line.<br><br>The right part "sc" specifies a string, hexadecimal, or octal character constant as described in Chapter 3. The length of the constant must be from one to four bytes. |
| | LMULT = | value | 1 | The right part "value" specifies a multiplication factor to be applied to the record length. The value specified will be multiplied by the value in the length field (see "LENGTH =" statement previously described) to compute the number of bytes in the record. "Value" is an integer in the range $1 \leq$ value $\leq 15$. |

Points to Note

1. The RECORD LENGTH left/right parts may be overridden by ANSI, IBM OS/Standard, or Honeywell 2000 COBOL labels which specify record length.

2. The values for RECORD LTHFLD, OFFSET, FORMAT, and PREAMBLE left/right parts may be overridden if RECORD STRUCTURE is changed as the result of ANSI, IBM OS/Standard, or Honeywell 2000 COBOL label processing.

3. RECORD CONSTANT may be enabled as the result of RECORD STRUCTURE being changed to "U" in label processing; however, no definition is assumed for the constant string. The default must be zero, or it must be defined in the job descriptor entry.

4. The LENGTH on a 4X3 packed format tape is the number of 6-bit bytes or characters in the record.

# Special Processing Statements

## Introduction

The special processing statements enable the user to specify logical functions which are performed on either a record or block basis. The following functions can be specified:

1. Define an end-of-report condition.
2. Select or delete the processing of blocks.
3. Select or delete the processing of records.
4. Define a page which will be offset in the bin.
5. Control the suppression of printing for sets of records within a report.

In order to fully define a special processing statement, the user must specify one or two fields in the record or block to be tested. In general, a logical processing statement has the following format:

Function TEST = (criteria);

The logical processing statement tests the value of the specified "expression" and directs the flow of processing based on the result of the test.

The "expression" portion of the statement defines a test to be performed on either one or two specified fields and their associated constants for a true or false value. The fields in the record or block are compared with their associated set of constants using either an equal (EQ) or not equal (NE) operator.

The basic element used to describe a test for a logical function is the CRITERIA statement. Each CRITERIA statement describes a field in either a record or block and the specific test to be performed. For example, the following CRITERIA statement describes a test for a subfield equal to a specific constant table.

$id_1$: CRITERIA CONSTANT = (offset, length, EQ, $id_2$);

The left part CONSTANT specifies that the content of a tape record or block, located "offset" bytes from the start of the record or block, with length "length" in bytes, is to be compared to the table constant value where $id_2$ is the identifier of a table containing constants. (See the final section of this chapter, "Table Statement".) When the subfield matches the constant set, the CRITERIA statement is true.

To complete the description of the entire test for a logical function, the TEST left part requires a right part that specifies either one or two CRITERIA statements. If only one test is to be performed to determine the value of a particular logical processing function, the form of the TEST left/right parts is as follows:

TEST = ($id_1$);

where "$id_1$" is the identifier for the particular CRITERIA statement. The parentheses in this format are optional.

If two CRITERIA statements are needed to determine the true/false value for a logical processing function, they may be either ANDed together or ORed together. The formats of the TEST left/right parts may be as follows:

TEST = ($id_1$, AND, $id_2$); or TEST = ($id_1$, OR, $id_2$);

where "$id_1$" and "$id_2$" are the identifiers for two CRITERIA statements. The parentheses in this format are required.

The particular logical processing function to be performed when the test expression is true is specified by the command. For example, to specify that a record should be selected if a particular field is set, the user should code the following three statements:

```
T1:   TABLE      CONSTANT = (sc);
C1:   CRITERIA   CONSTANT = (offset, length, EQ, T1);
      RSELECT    TEST = (C1);
```

### CRITERIA Statement

There are two formats for the CRITERIA statements: constant mode and change mode. In the constant mode, the user must specify the location, length, and contents of a fixed field within a user's record or block. Every user's record or block is examined at the specified location (expression function "offset") to determine if the constant is present or not present (the identifier "id" defines the table containing the constant). If so, the CRITERIA statement is true; if not, the statement is false. A constant mode CRITERIA statement can be coded as follows:

$id_1$: CRITERIA CONSTANT = (offset, length, $\left\{\begin{matrix}EQ\\NE\end{matrix}\right\}$ , $id_2$);

where $id_2$ is the name (or identifier) of a value TABLE.

In the change mode, the user must specify the length and location of a control field in each record or block. When the content of the control field of one record (or block) differs from the content of the control field of the previous record (or block), the CRITERIA statement is true.

Change-mode CRITERIA statement can be coded as follows:

$id_1$: CRITERIA CHANGE = (offset, length, NE, LAST);

Each CRITERIA statement may be either constant mode or change mode but not both.

| Command | Left/Right Parts | Interpretation |
|---|---|---|
| ac: CRITERIA | CONSTANT = (offset, length, $\left\{\begin{matrix}EQ\\NE\end{matrix}\right\}$ , id) | In block processing, the expression "offset" is the offset, in bytes (relative to zero), from the start of the physical tape block to a field within the tape block which is to be compared to a table or string constant. In the case of record processing, it is from the start of the user's portion of the record to the field in the record which is to be compared. Its range is $0 \leq$ offset $\leq 4096$.<br><br>The expression "length" is the length, in bytes, of the test field. Its range is $1 \leq$ length $\leq 255$. |

| Command | Left/Right Parts | Interpretation |
|---|---|---|
| ac: CRITERIA (cont.) | | EQ and NE are keywords which specify the operation of "equal to" or "not equal to".<br><br>The expression "id" is the identifier of a TABLE statement. |
| | CHANGE = (offset, length, NE, LAST). | The expression "offset" is the offset, in bytes (relative to zero), from the start of the user's portion of the record to the control field within the record, or from the start of the physical tape block to the control field within the block. Its range is 0 ≤ offset ≤ 4096.<br><br>The expression "length" is the length, in bytes, of the control field. Its range is 1 ≤ length ≤ 255.<br><br>The keyword NE indicates "not equal to".<br><br>The keyword "LAST" indicates that the control field of the current record (or block) is being compared to the control field of the previous (last encountered) record (or block). |

**Test Expressions**

The activation of testing is done by coding the keyword TEST as the left part of any logical processing command described below. The corresponding right part specifies the criteria for the test. Either one or two CRITERIA statements can be specified. If one CRITERIA statement is specified, the form of the left/right part is as follows:

$$\text{TEST} = \text{id}_1;$$

If the CRITERIA $\text{id}_1$ is true, the test is true; if the CRITERIA $\text{id}_1$ is false, the test is false. The CRITERIA $\text{id}_1$ may be in either change-mode or constant-mode format.

If two CRITERIA statements are specified for a test, the testing of the two is linked by either the AND logical operator or the OR logical operator. If AND is coded, then both CRITERIA statements must be true in order for the TEST to be successful. If OR is coded, then if either CRITERIA statement is satisfied, the TEST is successful. The format of the left/right part is as follows:

$$\text{TEST} = (\text{id}_1, \left\{ \begin{matrix} \text{AND} \\ \text{OR} \end{matrix} \right\}, \text{id}_2)$$

The second and third parameters are optional, but if either is specified, the other must also be present. The CRITERIA tables may specify either change-mode or constant-mode functions; there are no restrictions on their usage or combination.

There is one special case when the record or block is too short to include the field being tested. If the test specifies a constant-mode function, the CRITERIA will fail. If the test specifies a change-mode function, the CRITERIA will fail as no change has occurred, but the value for LAST will be unchanged for comparison with the next record.

## Logical Processing Commands

There are eight separate logical processing commands. The user can specify all eight but each individual command can be specified only once per job descriptor entry. If any one command is specified more than once, the last occurrence will be used without notification of any error. The available commands are listed below and will be described separately in the following sections:

| | |
|---|---|
| BDELETE | Block Deletion |
| BSELECT | Block Selection |
| RDELETE | Record Deletion |
| RSELECT | Record Selection |
| RSTACK | Stacked Reports - Report Separation on a Record |
| ROFFSET | Report Offsetting on a Record |
| RSUSPEND | Suspend Printing on a Record |
| RRESUME | Resume Printing on a Record |

### Block Deletion and Selection:

Interspersed blocks within one report or file may be either selected for, or deleted from, printing by use of the BDELETE and BSELECT statements. These statements can also be used to selectively delete from printing specialized blocks that are not originally placed on the data tape for printing, i.e., control blocks, unsupported labels, etc.

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| BSELECT | TEST = | $\left\{ \begin{array}{l} id_1 \\ (id_1, \left\{ \begin{array}{l} AND \\ OR \end{array} \right\}, id_2) \end{array} \right\}$ | none | Defines test expression for selecting blocks for printing. $id_1$ and $id_2$ are identifiers for either the change-mode or constant-mode CRITERIA statements. If only $id_1$ is present, then the test is satisfied if the block satisfies the criteria in $id_1$. If $id_1$ and $id_2$ are both present and the keyword AND is coded, the test is satisfied only if the block satisfies the criteria in both $id_1$ and $id_2$. If the keyword OR is coded, the test is satisfied if the block satisfies the criteria in either $id_1$ or $id_2$.<br><br>If the test is satisfied, the block is selected for printing. |
| BDELETE | TEST = | $\left\{ \begin{array}{l} id_1 \\ (id_1, \left\{ \begin{array}{l} AND \\ OR \end{array} \right\}, id_2) \end{array} \right\}$ | none | $id_1$, $id_2$, AND, and OR as above. If the test is satisfied, the block is deleted from the printed output. |

Points to Note

1. Offsets to the subfields of the blocks are the offsets, in bytes, relative to zero from the start of the block to the beginning of the subfield.

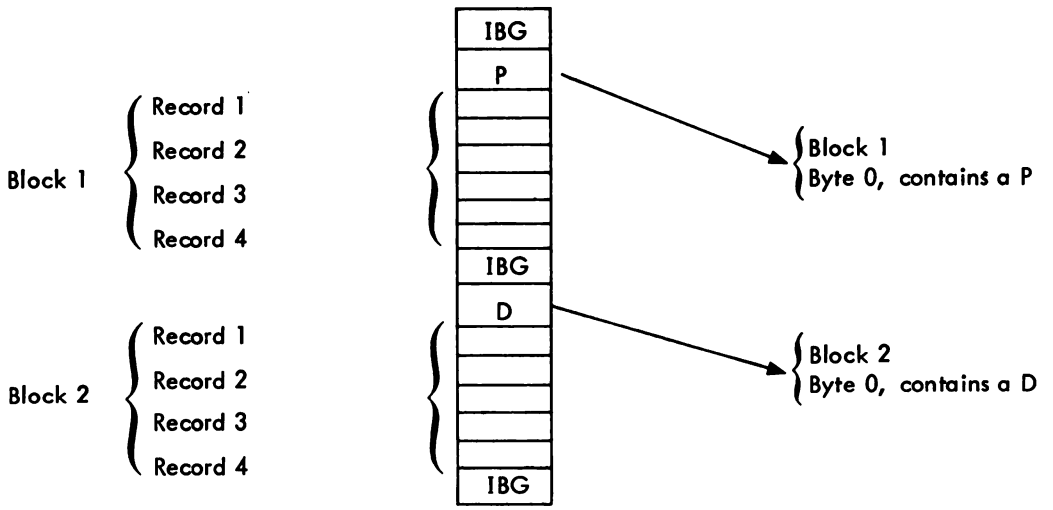
2. Block selection and deletion is performed prior to the extraction of the records from the block. If a block does not match the same format as the normal blocks, it can be deleted and will not cause a processing error. For example. a control block in a fixed blocked file may cause a processing error unless it is deleted first.

3. If a block is deleted from, or not selected for, printing, none of the records contained within the block will be processed at all, and will not be available for any other logical processing functions.

4. Figure 4-7 contains an example of the use of the BSELECT statement to process interspersed reports on a block basis.

Coded Statements

```
T1: TABLE     CONSTANT = ('P');
C1: CRITERIA  CONSTANT = (0,1,EQ,T1);
    BSELECT   TEST = (C1);
```




Key

IBG Interblock Gap

The contents of the first byte of each tape block (offset = 0, length = 1) is examined for the character constant 'P'. When a 'P' is detected, the entire block is selected (BSELECT) for printing. When the first byte of the block does not contain a 'P', the block is bypassed and not printed. In the above example, only block 1 is printed.

Figure 4-7. Processing Interspersed Reports on a Block Basis, Example

## Record Deletion and Selection

Interspersed records within one report or file may be either deleted from, or selected for, printing by use of the RDELETE and RSELECT statements. These statements can also be used to selectively delete from printing specialized records that are not originally placed on the data tape for printing, i.e., control records, offset records, etc.

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| RSELECT | TEST = | $\left\{ \begin{array}{l} id_1 \\ (id_1, \left\{ \begin{array}{l} AND \\ OR \end{array} \right\}, id_2) \end{array} \right\}$ | none | Defines test expression for selection of records for printing. $id_1$ and $id_2$ are identifiers for either the change-mode or constant-mode CRITERIA statements. If only $id_1$ is present, then the test is satisfied if the record satisfies the criteria in $id_1$. If $id_1$ and $id_2$ are both present and the keyword AND is coded, the test is satisfied only if the record satisfies the criteria in both $id_1$ and $id_2$. If the keyword OR is coded, the test is satisfied if the record satisfies the criteria in either $id_1$ or $id_2$.<br><br>If the test is satisfied, the record is selected for printing. |
| RDELETE | TEST = | $\left\{ \begin{array}{l} id_1 \\ (id_1, \left\{ \begin{array}{l} AND \\ OR \end{array} \right\}, id_2) \end{array} \right\}$ | none | $id_1$, $id_2$, AND, and OR as above. If the test is satisfied, the record is deleted from the printed output. |

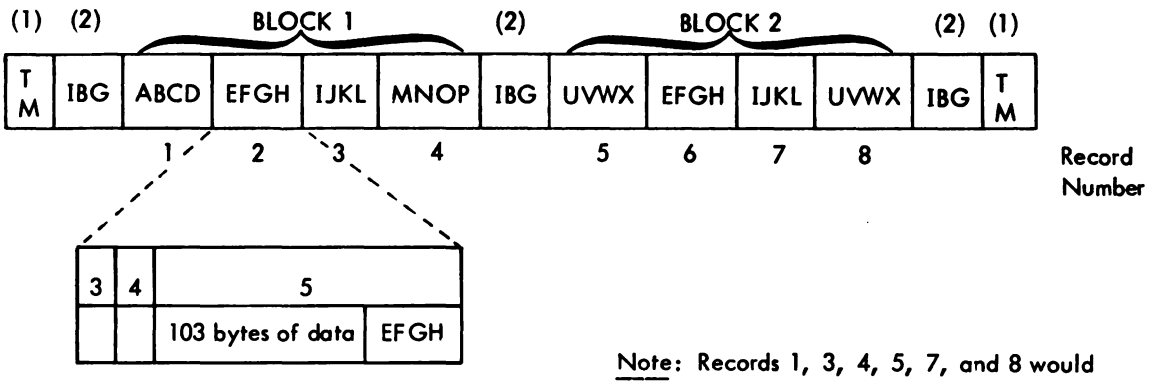
Points to Note

1. Offsets to the subfields of the records are the offsets, in bytes, relative to zero from the start of the user's portion of the record to the beginning of the subfield.

2. Figure 4-8 contains an example of the use of the RDELETE statement to process interspersed reports on a record basis.

**PDL Statements**

```
T1:  TABLE     CONSTANT = ('EFGH');
C1:  CRITERIA  CONSTANT = (104,4,EQ,T1);
     RDELETE   TEST = C1;
```




Note: Records 1, 3, 4, 5, 7, and 8 would be printed in the report.

Key

(1) TM = Tape Mark
(2) IBG = Interblock Gap
(3) Record Length Field
(4) Carriage Control Byte
(5) Print Line

If the contents of a deletion field located 104 bytes from the start of the user portion of the record are equal to the constant 'EFGH', then the record is not printed.

Figure 4-8. Processing Interspersed Reports on a Record Basis, Example

### Report Offsetting on a Record

Use of the logical processing function ROFFSET provides the capability to initiate a page offset in the stacker bin under control of the data tape. These special user-controlled offsets can be used to simplify job distribution by creating separate stacks for each separate distribution entity. No other special processing occurs as a result of the ROFFSET test being satisfied; for instance, the report is not terminated nor are multiple copies produced at the offset juncture. An accounting log will not be produced either but will occur if requested at the end of job as usual.

The page which will be offset in the bin can be determined by the following criteria:

1. If the record which satisfies the ROFFSET test is printed on a page of the output, then that page will be the offset sheet.

2. If the record which satisfies the ROFFSET test is not printed on a page of the output (i.e., it is deleted), the ROFFSET function will be performed for the next printable record. If the next printable record causes a transition to the next page, then the next page will be offset.

ROFFSET also provides the control to force an offset on either all copies of the report or only the first copy. In conjunction with job offset control OFFSET = as specified on the OUTPUT command (see Chapter 5), the user can exert extensive control over the offsetting function so as to build tailored, job-controlled stacks of output in the stacker bin of the 9700.

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| ROFFSET | TEST = | $\{ id_1 \}$ $\{ (id_1, \{AND\} , id_2) \}$ | none | Defines test expression for offsetting pages to the stacker bin. $id_1$ and $id_2$ are identifiers for either the change-mode or constant-mode CRITERIA statements. If only $id_1$ is present, then the test is satisfied if the record satisfies the criteria in $id_1$. If $id_1$ and $id_2$ are both present and the keyword AND is coded, the test is satisfied only if the record satisfies the criteria in both $id_1$ and $id_2$. If the keyword OR is coded, the test is satisfied if the record satisfies the criteria in either $id_1$ or $id_2$.<br><br>If the test is satisfied, the record will cause an offset in the stacker bin. |
| | PASSES = | $\{FIRST\}$ $\{ALL\}$ | ALL | If FIRST is coded, the record will cause an offset only on the first pass of a collated print run. If ALL is coded, the record will cause an offset on all passes of a collated print run. |

Points to Note

1. The ROFFSET feature prints the record satisfying the ROFFSET TEST according to the normal job parameters.

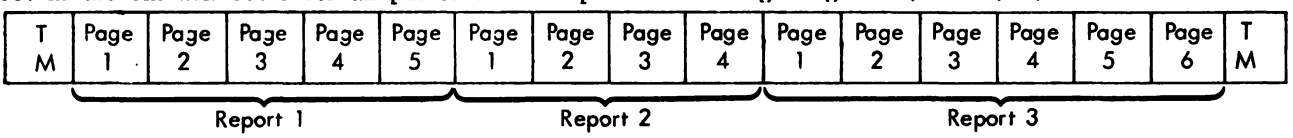
2. The record satisfying the ROFFSET TEST can be part of the normal print job, such as a record containing "Page 1" as part of its header.

3. If ROFFSET is specified for an uncollated job, an offset will occur on only the first copy of the offset page.

4. If a record satisfying the ROFFSET test criteria is then not selected for, or deleted from, printing, the offset indication will be maintained and used to cause the offset on the next record which is printed.

5. Figure 4-9 contains an example of the use of the ROFFSET feature.

A file has multiple reports without any delimiters separating the reports. Each page of the report has a page number as part of the heading. Each report causes renumbering, of the pages starting with "Page ... 1". An offset in the bin will occur for all passes of the reports at the beginning of reports 1, 2, and 3.




PDL Statements

```
T1:  TABLE     CONSTANT = ('PAGE...1');
C1:  CRITERIA  CONSTANT = (105,8,EQ,T1);
     ROFFSET   TEST = C1, PASSES = ALL;
```

Figure 4-9. Offsetting on Stacked Reports

### Print Suppression

The print suppression logical processing function permits the user to delete from printing groups of records that are distinguishable at the start and end, but whose intermediate records may not be unique or distinguishable. Print suppression is invoked by the use of two separate commands - RSUSPEND and RRESUME. The tests for each command are independent and must be separately described. Each of the commands can specify the full range of tests as described previously for the other logical processing commands.

When specifying either the RSUSPEND or RRESUME commands, the user can also specify whether the printing will begin the suspension or resumption of printing on the current or next record. This is controlled by the left/right parts BEGIN = $\left\{\begin{matrix}\text{CURRENT}\\\text{NEXT}\end{matrix}\right\}$ This additional control provides the necessary flexibility to cope with the variability of requirements for print suppression. The BEGIN left/right parts are independently specifiable in both RSUSPEND and RRESUME commands.

Upon encountering a record which satisfies the test criteria specified on the RSUSPEND command, printing will be suspended. If BEGIN = CURRENT is coded on the RSUSPEND command, then this record will not be printed. If BEGIN = NEXT is coded, then the record satisfying the TEST is printed and records will be discarded beginning with the following record.

Printing will be resumed when a record satisfying the TEST in the RRESUME command is encountered. If BEGIN = CURRENT is coded in the RRESUME command, the record satisfying the TEST will be printed. If BEGIN = NEXT is coded, printing will resume with the next record.

| Command | Left Part | Right Part | Default | Interpretation |
|---|---|---|---|---|
| RSUSPEND | TEST = | $\left\{\begin{matrix}id_1\\(id_1, \left\{\begin{matrix}\text{AND}\\\text{OR}\end{matrix}\right\}, id_2)\end{matrix}\right\}$ | none | Defines test expression for the beginning of print suppression. $id_1$ and $id_2$ are identifiers for either the change-mode or constant-mode CRITERIA statements. If only $id_1$ is present, then the test is satisfied if the record satisfies the criteria in $id_1$. If $id_1$ and $id_2$ are both present and the keyword AND is coded, the test is satisfied only if the record satisfies the criteria in both $id_1$ and $id_2$. If the keyword OR is coded, the test is satisfied if the record satisfies the criteria in either $id_1$ or $id_2$.<br>If the test is satisfied, the record will be used to suspend printing. |
| | BEGIN = | $\left\{\begin{matrix}\text{CURRENT}\\\text{NEXT}\end{matrix}\right\}$ | NEXT | If BEGIN = CURRENT is coded, the record satisfying the test expression will not be printed.<br>If BEGIN = NEXT is coded, the record satisfying the test expression will be printed and printing will be suppressed beginning with the next record. |

| Command | Left Part | Right Part | Default | Interpretation |
|---|---|---|---|---|
| RRESUME | TEST = | $\left\{ \begin{matrix} id_1 \\ (id_1, \left\{ \begin{matrix} AND \\ OR \end{matrix} \right\}, id_2) \end{matrix} \right\}$ | none | Defines test expression for the resumption of printing following print suppression. $id_1$ and $id_2$ are identifiers for either the change-mode or constant-mode CRITERIA statements. If only $id_1$ is present, then the test is satisfied if the record satisfies the criteria in $id_1$. If $id_1$ and $id_2$ are both present and the keyword AND is coded, the test is satisfied only if the record satisfies the criteria in both $id_1$ and $id_2$. If the keyword OR is coded, the test is satisfied if the record satisfies the criteria in either $id_1$ or $id_2$.<br><br>If the test is satisfied, the record will be used to resume printing following print suppression. |
| | BEGIN = | $\left\{ \begin{matrix} CURRENT \\ NEXT \end{matrix} \right\}$ | NEXT | If BEGIN = CURRENT, the record satisfying the test expression will be printed. If BEGIN = NEXT is coded, the record satisfying the test will not be printed and printing will begin with the next record. |

Points to Note

1. The user should ensure that if an RSUSPEND command is coded, then an RRESUME command is also present for the job. A warning will be issued if one but not both commands are invoked for a job. However, the JDE will be compiled as programmed.

2. If a data record satisfying the test expression in the RSUSPEND is encountered, printing will be suspended. If no record satisfying the test expression in the RRESUME command is encountered (or no RRESUME command is present for the job), there will be no output generated for records that occur after the point of suspension.

3. A record satisfying the RSTACK test will still be found and will terminate the report, even if the printing of records is suspended at the time.

4. Record selection/deletion is performed prior to suspend/resume processing. If a record satisfying either the suspend or resume test criteria was previously not selected for, or deleted from, printing, it will not cause either the suspend or resume function.

5. The records just prior to print suspension and after resumption should have compatible carriage control characters. No additional carriage control characters will be inserted by the system during the print suppression.

6. Figure 4-10 contains an example of the use of the RSUSPEND and RRESUME statements to delete the printing of Job Control statements from a job.

Sample Input

Records to be deleted from printing

```
//JOB .FOR001A0 C 219                                    DATE 09/23/78
//OPTION CAYAL                                             00000020
  PHASE FOR001D0,                                          09/23/78
//ASSGN SYSRLB,3340,TEMP,VOL=PVTLIB,SHR                    06/16/78
  INCLUDE IMPROOT2                                         06/16/78
  INCLUDE IMPCBBM                                          06/16/78
  INCLUDE UTL063A0                                         09/02/78
//EXEC    FCOBOL,SIZE=64K                                  09/01/78
```

Printed output

```
CBL SUPMAP
CBL SXREF
CBL LIB
CBL OPT
CBL STATE                                                  06/24/78
CBL NOOPT                                                  06/24/78
CBL FLOW                                                   06/24/78
00001  000120 ID DIVISION.
00002  000130 PROGRAM-ID. FOR001B0.
00003  000140 AUTHOR. D. DYLAN
00004  000150 DATE-WRITTEN. MAY 13TH 1978.
00005  000160 REMARKS:
00006  000170     THIS PROGRAM WILL UPDATE ITEM AND FORECAST
00007  000180     RECORDS OFF OF THE FORECAST MASTER AND OUT-
00008  000190     PUT A NEW TAPE. THE USER MAY REQUEST MAIN-
00009  000200     TENANCE ON ALL FIELDS, OR MAY JUST WANT TO
00010  000210     CHANGE ONE FIELD.
```

PDL statements

```
T1:TABLE          CONSTANT = ('//JOB ','//EXEC');
T2:TABLE          CONSTANT = ('EOJ ');
T3:TABLE          CONSTANT = ('//EXEC');

C1:CRITERIA       CONSTANT = (1,6,EQ,T1);
C2:CRITERIA       CONSTANT = (1,4,EQ,T2);
C3:CRITERIA       CONSTANT = (1,6,EQ,T3);

RSUSPEND          TEST = (C1,OR,C2),
                  BEGIN = CURRENT;
RRESUME           TEST = (C3,OR,C2),
                  BEGIN = NEXT;
```

Figure 4-10. Print Suppression Used to Delete Job Control Card Statements from the Printed Output

## Stacked Reports

### INTRODUCTION

The stacked reports feature enables the printing system user to define a series of reports in a single file. This is accomplished by specifying an "end-of-report" condition in the coded logical processing statement RSTACK. "End-of-report" is that point in processing a report when all of the pages of a copy of a report have been formatted to disk and processing has not begun on the next report.

Reports are "stacked" in a file if more than one report is included in a single file and separated from each other logically and not physically with tapemarks and/or operating system labels. In processing stacked reports, the system checks each record for the logical end-of-report specification in the TEST expression of the RSTACK statement.

The RSTACK statement uses a TEST = left/right part as described in the previous section "Test Expressions". One or two subfields of each record can be tested using either the constant-mode or change-mode criteria tables.

### DELIMITER MODES

There are two modes of stacked reports which are supported by the system. In the delimiter mode, the record satisfying the TEST criteria is not part of the report following the delimiter but simply serves to separate or "delimit" one report from another. In the non-delimiter mode, reports are stacked one on top of each other without any special records separating the reports. The two modes are specified by the user by coding the left/right part DELIMITER = YES/NO.

If DELIMITER = YES is coded, the user may actually separate each report with multiple successive records, each of which satisfies the TEST expression on the RSTACK statement. In this case, all consecutive delimiters are treated as a delimiter packet. In delimiter mode, the user has the option to print the delimiter or the delimiter packet and to select the output destination of this delimiter page - BIN, TRAY, or BOTH. The delimiter page, when printed, is output as part of the subsequent report. All of the printing options selected in the JDL remain in effect during report delimiter printing except carriage control, which is ignored; the carriage control is replaced by a 'Print and Space 1 Line' control. If DJDE records exist within the delimiter "packet" or immediately following, they are included on the delimiter page.

In non-delimiter mode (DELIMITER = NO), a single record separates one report from the next. This record, after satisfying the TEST criteria, is considered part of the subsequent report. In this mode, the delimiter record cannot be printed to the bin or tray (i.e., PRINT left part not available when DELIMITER = NO). It will also not be displayed on the operator's console.

### DELIMITER DISPLAY

Display of the delimiter to the keyboard/display (performed only for DELIMITER = YES) is dependent on the mode (Single/Multi) in which the job is run. Single or Multi-report mode is user selectable within the user's PDL (RMODE left part of the VOLUME command) or with the S/M option on the START command (see "Starting a Job" in Chapter 7).

If the print job is run in Single-report mode, the first delimiter of the next report will be displayed on the keyboard/ display. This will occur after an end-of-report is encountered. To begin the next single-report job, the operator must key-in another START command.

If the report mode is "Multi" the delimiter will not be displayed on the keyboard/display unit.

### Delimiter on Accounting Log

The user can select the option of including part of the first record of a report on the Accounting Log which can be printed for that report. This option is selected by coding ACCTINFO = (offset, length) on the RSTACK statement. This option will normally be used to print part of the first delimiter.

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| RSTACK | TEST = | $\left\{ \begin{matrix} id_1 \\ (id_1, \left\{ \begin{matrix} AND \\ OR \end{matrix} \right\} , id_2) \end{matrix} \right\}$ | none | Defines test expression for "end-of-report" conditions. $id_1$ and $id_2$ are identifiers for either the change-mode or constant-mode CRITERIA statements. If only $id_1$ is present, then the test is satisfied if the record satisfies the criteria in $id_1$. If $id_1$ and $id_2$ are both present and the keyword AND is coded, the test is satisfied only if the record satisfies the criteria in both $id_1$ and $id_2$. If the keyword OR is coded, the test is satisfied if the record satisfies the criteria in either $id_1$ or $id_2$.<br>If the test is satisfied, the record will specify an "end-of-report" condition. |
| | DELIMITER = | $\left\{ \begin{matrix} YES \\ NO \end{matrix} \right\}$ | NO | If DELIMITER = YES is coded, all consecutive records satisfying the TEST criteria separate one report from another but are not part of either report.<br>If DELIMITER = NO is coded, this single record separates one report from another and is actually part of the subsequent report. |
| | PRINT = | $\left\{ \begin{matrix} TRAY \\ BIN \\ BOTH \\ NONE \end{matrix} \right\}$ | NONE | If DELIMITER = YES is coded, the user may specify if the report delimiters are to be printed, and if so, the location to where the printed delimiters are to be output:<br>NONE indicates that report delimiters are not to be printed.<br>BIN indicates that report delimiters are to be printed, and the output delivered to the output stacker bin.<br>TRAY indicates that report delimiters are to be printed, and the output delivered to the sample print tray. |

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| RSTACK (cont.) | | | | BOTH indicates that report delimiters are to be printed, and the output delivered to both the sample print tray and output stacker bin.<br>If DELIMITER = NO is coded, no page will be printed. |
| | ACCTINFO = | (offset, length) | none | Specifies that a subfield of the first record is to be printed on the accounting log at the end of the report. If DELIMITER = YES is also coded then the subfield will be from the first delimiter record of the report. For DELIMITER = NO the subfield will be from the first data record. "offset" is the offset in bytes (relative to zero) from the start of the user's portion of the record to the subfield within the record.<br>"length" is the length (in bytes) of the subfield ($0 \leq$ length $\leq$ 64). |

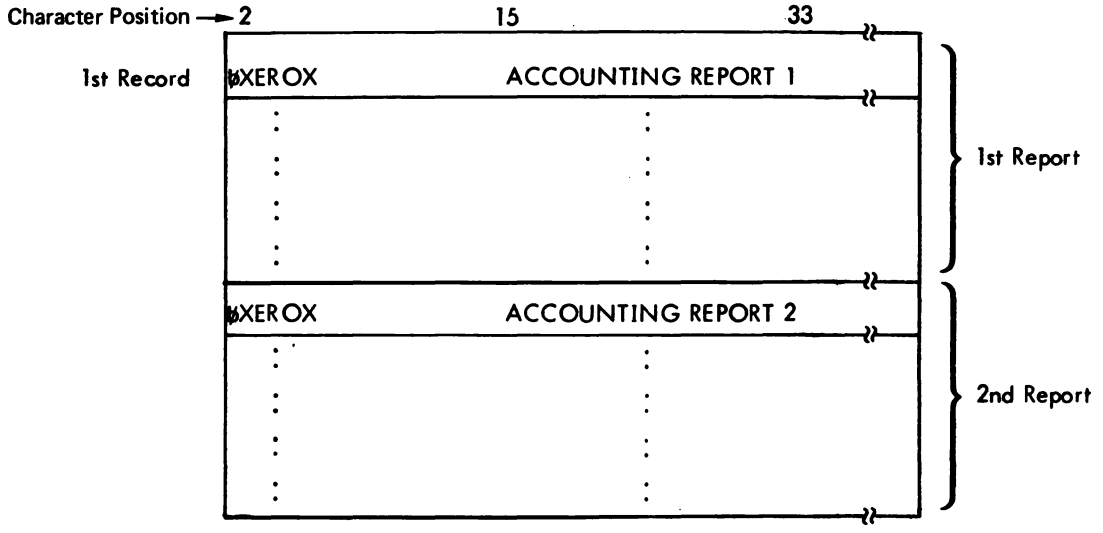
Points to Note

1. If the TEST expression on the RSTACK statement consists solely of a change-mode CRITERIA statement, the DELIMITER = NO must be coded.

2. An RSTACK statement containing a TEST expression specifying a constant-mode CRITERIA statement and DELIMITER = NO can be used to detect a heading of a report as a report boundary.

3. A record which is an RSTACK delimiter cannot be deleted from, or not selected for, printing by the RSELECT/RDELETE logical processing. If the record satisfies the RSTACK test criteria but is not a delimiter, it can be deleted from, or not selected for, printing but will still cause report separation.

4. Figure 4-11 includes an example of a stacked report.




PDL Statements:

```
T1: TABLE      CONSTANT = ('XEROX');

C1: CRITERIA   CONSTANT = (1, 5, EQ, T1);

    RSTACK     TEST = C1,
               DELIMITER = YES,
               ACCTINFO = (14, 19);
```

**Figure 4-11. Example of Stacked Report Processing**

### TABLE Statement

The TABLE statement is used to build a table of constants, for use by the special logical processing statements. Each constant included in a TABLE statement will be examined by the system to see if it is equal in value to the constant specified in the "expression constant" portion of the CRITERIA statement. THE TABLE statement must precede its reference in a CRITERIA statement.

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| ac:TABLE | CONSTANT = | (sc,...sc) | none | The table identifier (referenced in the CRITERIA statement as "id") is "ac".<br><br>The right part "(sc,...sc)" is one or more hexadecimal, octal, ASCII, EBCDIC or character string constants as described in Chapter 3. Each constant in a TABLE statement must be the same length in bytes. The number of bytes for all constants in the table (after conversion if any constants are hexadecimal or octal) is limited to 255 bytes. |

# 5. OUTPUT PROCESSING FUNCTIONS

## Introduction

The PDL programmer defines the output report format for a job or series of jobs in the job descriptor library. The OUTPUT, CME, LINE, PDE, VFU, and PCC statements enable the user to specify the following report characteristics:

1. Number of copies and collate mode.
2. Copy modification (or spot carbon) of output.
3. Forms and Fonts to be used.
4. Control of report offsetting.
5. Print line length and carriage control conventions.
6. Vertical format channel (output channel to output line correspondence).
7. Top and bottom-of-form values.
8. Automatic page numbering.
9. Provisions for report covers.
10. Non-standard assignment of printer carriage control operators.
11. Selection of page orientation.

The job accounting command ACCT, enables the installation to specify that an accounting summary is to be output at the end of processing for each report. The ABNORMAL command allows the user to select optional features that enhance output integerity. Two commands are provided to allow the user to inform the operator of special conditions. The MESSAGE command causes a message to be displayed to the operator during job processing or job printing. The ROUTE command is used to print a message on a separate sheet of paper preceding a report. The remaining sections of this chapter discuss the output statements and the syntax to be used in coding these statements.

## Output Control Features

The OUTPUT statement enables the programmer to specify:

1. The number of output copies which are to be printed.
2. Whether they are to be collated or uncollated.
3. If the copy modification (or spot carbon) feature is to be utilized.
4. Control of report offsetting.
5. Whether cover pages are to be included.
6. If automatic page numbering is to be performed.
7. Which forms are to be utilized in the report.

| Command | Left Part | Right Parts | Defaults | Interpretation |
|---|---|---|---|---|
| OUTPUT | COPIES = | number | 1 | This right part "number" specifies the number of copies which are to be output. The "copies" option parameters on the START command allows this right part to be overridden by the operator when initiating a print job. The range for "number" is 0 to 32,767. |
| | COLLATE = | { YES / NO } | YES | If YES is specified, the output pages are collated. If NO is specified, the output is not collated. |
| | OFFSET = | { ALL / FIRST / NONE } | ALL | The default (ALL) results in an offset of each copy of each file.<br>The right part "NONE" specifies that there is to be no offset at any time.<br>The right part "FIRST" specifies that an offset is to occur only on the first copy of a file.<br>The OFFSET control of "FIRST" or "ALL" copies may be modified by the ROFFSET command (see "Report Offsetting on a Record" in Chapter 4). An example of job offset control is shown in Figure 5-1. |
| | MODIFY = | { cme-id / (cme-id, init [,copies]) / NONE } | NONE | The right part "cme-id" is the statement identifier of the CME statement. The CME statement must precede a reference to its identifier by the MODIFY left/right pair (or a CME control file may be on disk cataloged in the CME directory - see "Cataloged CMEs" in this chapter).<br>The right part "init" specifies the initial ply (pass) to which the associated CME is to be applied.<br>The right part "copies" specifies the number of plies (passes) on which to apply the CME. If "copies" is not specified then the CME will apply to all copies beginning with the copy number specified by the right part "init".<br>The right part "NONE" specifies that no CME is to be used. Data will be processed without modification. |

| Command | Left Part | Right Parts | Defaults | Interpretation |
|---|---|---|---|---|
| OUTPUT (cont.) | MODIFY (cont.) | | | Different CMEs may be associated with different copies of a report by the use of multiple MODIFY left parts on the same OUTPUT command. CME copy ranges may not overlap.<br><br>See the MODIFY DJDE command described in Chapter 6 for a discussion of how CMEs may be changed on a page-by-page basis within a report ply. |
| | COVER = | { FRONT<br>(FRONT,SEP)<br>BACK<br>BOTH<br>(BOTH,SEP)<br>NONE } | NONE | The left part "COVER" allows specification of the type of cover pages to be picked from the auxiliary feed tray.<br><br>The right parts are defined as follows:<br><br>"FRONT" specifies that a cover page is to be picked at the front of each report and will be the first page of the report.<br><br>"SEP" specifies that each front cover will not have report data printed on it.<br><br>"BACK" specifies that a cover page is to be picked at the end of each report. No report data is printed on back covers.<br><br>"BOTH" specifies that both front and back cover pages are to be picked and the front of each report will be the first page of the report.<br><br>"NONE" specifies that no cover pages are to be picked. |

| Command | Left Part | Right Parts | Defaults | Interpretation |
|---|---|---|---|---|
| OUTPUT (cont.) | FORMAT = | pde-id | FMT1 | Page Descriptor Entries (PDEs) which are used in the formatting of the printed output are referenced via the left part "FORMAT".<br><br>The right part "pde-id" references a PDE which must be defined previously in a Job Descriptor Library or may be a reference to a PDE file separately cataloged in the PDE library on disk. A set of standard "pde-id"s are defined in Table 5-1 (i.e., FMT1, FMT2, etc.).<br><br>Details on the PDE command are discussed in the "Page Descriptors" section of this chapter. Creating and compiling PDE files with the PDL processor is discussed in "PDE and CME Compilation" in Chapter 3. |
| | FORMS = | { form-id<br>(form-id, init [,copies])<br>NONE } | NONE | The left part "FORMS" allows for the specification of forms to be associated with the report copies.<br><br>Different forms may be associated with different copies of a report by the use of multiple FORMS left parts on the same OUTPUT command.<br><br>The right part "form-id" specifies a 1-6 character file name (may be numeric, alpha, or alphanumeric) which exists on disk. This file is created by compiling Forms Description Language files with the FDL system task (see Chapter 10 for further details on FDL).<br><br>The right part "init" specifies the beginning ply (pass) number to which a specified form applies. This defaults to the first or next copy.<br><br>The right part "copies" specifies the number of plies (passes) to which a specified form applies. If "copies" is not defined the last (or only) form specified will apply to all copies beginning with copy number "init". If the form is not the last one specified, then "copies" defaults to 1. |

| Command | Left Part | Right Parts | Defaults | Interpretation |
|---|---|---|---|---|
| OUTPUT (cont.) | FORMS = (cont.) | | | If both "init' and "copies" are not specified then the form will apply to all copies of the report. |
| | NUMBER = | { (pnum,lnum,cnum) / NO } | NO | The left part "NUMBER" specifies whether or not page numbering is to be performed for a report.<br><br>The first font specified in the "FONTS =" left/right part (of the PDE command) is used to print the page numbers (see "Page Descriptors" in this chapter).<br><br>The right parts are defined as follows:<br><br>"pnum" specifies the starting number for page numbering. The beginning page number may be non-positive. The number will be incremented at page transitions, but not printed until it goes positive.<br><br>"lnum" specifies the line number on which the page number is to be placed.<br><br>"cnum" specifies the ending column number for the page number sequence.<br><br>"NO" specifies that no page numbering is to be performed. |

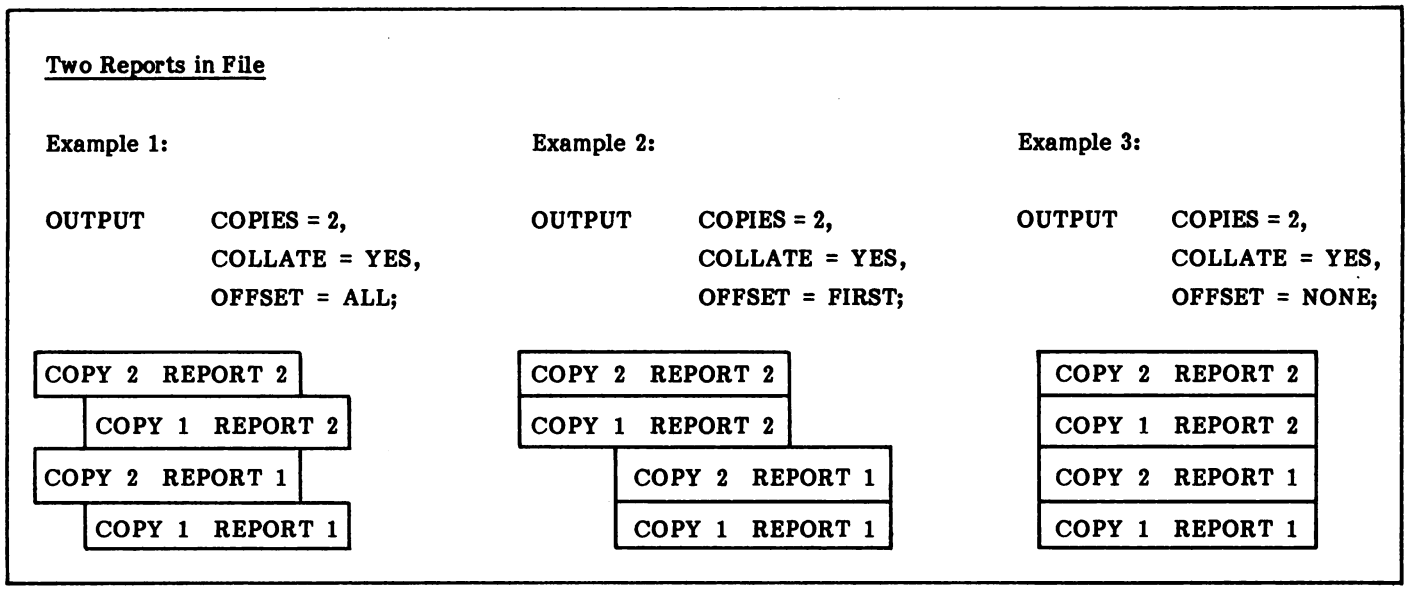



Figure 5-1. Results of Job Offset Control Options

Table 5-1. OSS Standard Formats (PDEs) Available on System Tape

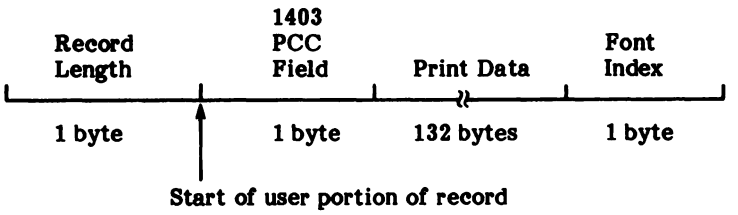
| PDE ID | Number of Lines | Number of Columns | LPI | CPI | Approximate Point Size | Mode | BEGIN Values | Default FONT ID |
|---|---|---|---|---|---|---|---|---|
| FMT1 | 66 | 132 | 8.1 | 13.6 | 9 | L | (.18,.66) | L0112B |
| FMT2 | 66 | 150 | 8.1 | 15 | 9 | L | (.18,.50) | L0212A |
| FMT3 | 88 | 132 | 10.7 | 13.6 | 7 | L | (.14,.66) | L0312A |
| FMT4 | 88 | 150 | 10.7 | 15 | 7 | L | (.14,.50) | L0412A |
| FMT5 | 49 | 100 | 6 | 10 | 12 | L | (.17,.50) | L0512A |
| FMT6 | 80 | 100 | 8.1 | 13.6 | 9 | P | (.57,.58) | P0612A |
| FMT7 | 60 | 90 | 6 | 12 | 12 | P | (.50,.50) | P07TYA |
| FMT8 | 60 | 75 | 6 | 10 | 12 | P | (.50,.50) | P0812A |
| FMT9 | 80 | 200 | 10.0 | 20.0 | 7 | L | (.25,.25) | L0912A |
| FMT10 | 132 | 132 | 12.5 | 17.6 | 6 | P | (.22,.51) | P1012A |
| FMT11 | 132 | 150 | 12.5 | 20.0 | 6 | P | (.22,.50) | P1112A |

The names in the "PDE ID" column above are standard print formats which may be refered to in the FORMAT = pde-id left/right part of the OUTPUT command. If a print format is used which has "Number of Lines" greater than 66 then the VFU command must be used to set the TOF and BOF parameters.

## Print Line Structure

The LINE statement describes the location and format of the print line data on the input tape. For example, the print data offset specifies the number of bytes between the start of the user portion of the record and the first character of that record which is to be printed. Likewise, the print length specifies the number of characters in the longest print line in the file.

Example:




Data record structure on tape

LINE DATA = (1, 132)

The LINE statement also defines the width of the left-hand margin of the print page, and the carriage control convention (ANSI, XEROX, etc.) that is to be applied to the print data file. The installation may also define its own carriage control convention if desired. In the preceding example, the following statements

```
LINE  PCC = (0, NOTRAN), PCCTYPE = IBM1403;
```

would define the carriage control type (IBM1403), and its byte position within the user portion of the record.

The LINE statement allows the user to specify in a user data record on tape a specific font to be used. The user specifies a font by including an index value (which indicates which font is to be selected from the list in the FONTS left/right part of a PDE command) into the data record. The byte position of this font index is defined by the FONTINDEX left/right part of the LINE command. In the above example, the statement would be coded as:

```
LINE  FONTINDEX = 133;
```

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| LINE | MARGIN = | { value<br>(value {b} {IN<br>CM<br>POS} )} | (1, POS) | The left part "MARGIN" provides for specification of the left margin on a physical page.<br>"value" is a value of the form "nnn.mm" (a decimal number with up to 2 digits to the right of the decimal point) which is the distance for the left margin and can be specified in inches (IN), or centimeters (CM). "value" must be specified as an integer ("nnn") for character positions (POS). |

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| LINE (cont.) | DATA = | (pdo, length) | (1, 132) | The right part "pdo" (print data offset) is the number of bytes between the start of the user portion of the logical record and the first character of the record which is to be printed.<br>The right part "length" specifies the maximum length of printable data within each logical record. |
| | PCC = | (offset, {NOTRAN / TRAN}) | (0,NOTRAN) | The right part "offset" specifies the byte offset within each logical record to the printer carriage control (PCC) field.<br>The right part {NOTRAN / TRAN} specifies whether or not the printer carriage control byte is to undergo code translation. TRAN indicates that the byte is to be translated into 9700 standard EBCDIC (see Appendix C) before being applied; NOTRAN prohibits translation. |
| | PCCTYPE = | {ANSI, B2500, B2700, B3500, B3700, B4700, B6700, H6000, H2000, IBM1401, IBM1403, UNIVAC, US70, XEROX, id, USER, NONE} | ANSI | The right part specifies the carriage control convention which is to be utilized in printing a job.<br>Creation of a user defined PCC table referenced by either an identifier "id" or the keyword USER is defined using the PCC statement. See the subsequent section of this chapter "PCC Statement".<br>The INITIAL parameter (see PCC command in this chapter) for any selected PCCTYPE (except ANSI, USER and id) is TOF. For PCCTYPE = ANSI then INITIAL is BOF. For PCCTYPE = USER or id the INITIAL parameter is set by the user in the PCC command.<br>The ADVTAPE parameter (see PCC command in this chapter) for any selected PCCTYPE (except IBM1403, USER and id) is YES. For PCCTYPE = IBM1403, ADVTAPE is set to NO. For PCCTYPE = USER or id the ADVTAPE is set by the user in the PCC command. |

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| | VFU = | {vfu-id / NONE} | NONE | The right part "vfu-id" is the statement identifier of the VFU table which must precede this reference to it. (See "Defining Print Line Positions", below). The VFU table defines print line positions corresponding to skip to channel commands for the job or jobs which are to be processed. If NONE is used then a channel is replaced by a carriage control of print and space 1. |
| | FONTINDEX = | {offset / NONE} | NONE | The left part "FONTINDEX" allows the user to specify in a user data record a specific font to be used for that line.<br>"offset" specifies the byte position (relative to zero) in the user data record which contains an index into the set of fonts as defined in the PDE command (see "Page Descriptors" in this chapter).<br>The font index in the data record (the 4 low order bits) is a number in the range 1 to n ($n \leq 15$), where n is the number of fonts specified in the PDE command. If the FONTINDEX in the data record is greater than the number of fonts specified, then the first font in the list is used.<br>The right part "NONE" specifies that there is no font index.<br>See "Points to Note" following this table for discussion of FONTINDEX usage with overprint lines. |
| | OVERPRINT = | ({PRINT / IGNORE / MERGE / PRINT2} , {DISP / NODISP}) | (PRINT, NODISP) | The right part {PRINT / IGNORE / MERGE / PRINT2} specifies the manner in which overprint lines will be handled.<br>If IGNORE is coded, all overprint lines are ignored.<br>PRINT specifies that all overprint lines are to be printed as they would be on an impact printer. |

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| LINE (cont.) | OVERPRINT = (cont.) | | | PRINT2 specifies that only one overprint will occur per line. Other overprints for the line will be ignored.<br><br>MERGE is the same as PRINT except when used with FONTINDEX (See "Points to Note" following this table).<br><br>DISP and NODISP are Xerox 1200 Computer Printing System options which are preserved here for compatibility purposes. Neither is functional. The number of overprint lines will always be printed on the accounting page. |

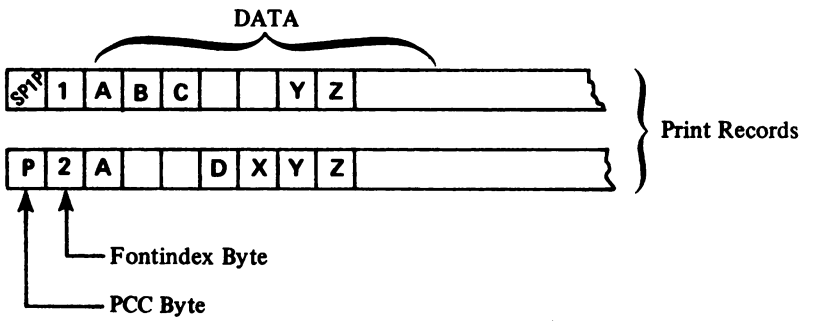
Points to Note

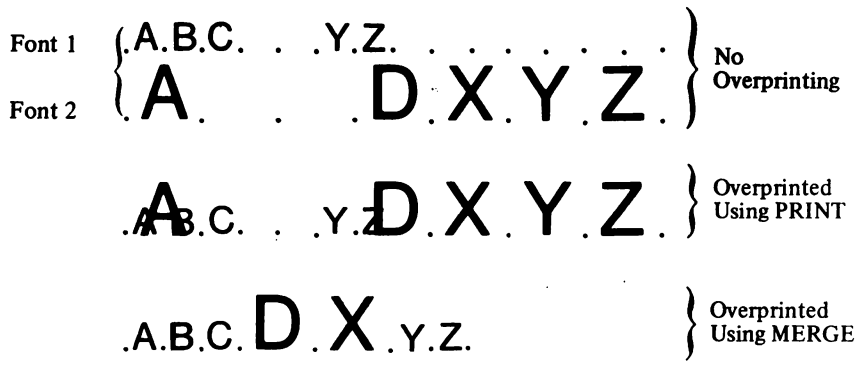
FONTINDEX, when used in conjunction with overprinting is handled as follows:

1. For OVERPRINT = PRINT the 9700 will overprint records analagous to an impact printer if the fonts are the same. If the fonts differ, records will be overprinted without regard to character spacing (see example below).

2. For OVERPRINT = MERGE the 9700 will replace characters in the previous record which are blank. Character spacing values will be adjusted, thus proportionally spaced or different size fonts may be used and the 9700 will perform the character placement. Line spacing is determined by the last font in use with the previous line. The current line is adjusted downward by the difference in height between the last font in the previous line and the largest font in the current line.

   Example:



## Defining Print Line Positions

The VFU statement is used to assign output line numbers to printer carriage control channels. These line to channel assignments perform the same function as the printer carriage control tape on a conventional line printer. The statement is also used to assign line numbers to the "top-of-form" and the "bottom-of-form". Top-of-form indicates the number of lines from the top of an output page to the first print line on the page. Likewise, bottom-of-form indicates the number of lines from the top of an output page to the last print line on the page.

Top and bottom of form are used for pre-job page alignment and for page overflow processing. For all PCCTYPEs except ANSI and user defined PCCs, the pre-job page alignment is to top-of-form in the expectation that the first carriage control command of the job will be print and space one line, or something similar. Selection of ANSI causes alignment to bottom-of-form to handle the skip to channel 1 and print command, which usually begins a job of that carriage control type. User defined PCCs may set alignment at either TOF or BOF.

Page overflow occurs when spacing to the next line causes the bottom-of-form line number to be exceeded. Page transition occurs, and line spacing is continued from the top-of-form line number. Honeywell 2000 carriage control and Xerox carriage control are exceptions to this processing (see 9700 Tape Formats manual).

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| ac: VFU | | | | "ac" is a 1 to 6 character statement identifier that is referenced by the VFU = id portion of the LINE command. One character must be alphabetic; the others may be alphabetic or numeric. |
| | ASSIGN = | {(channo, lineno),<br>(channo, (lineno,...,lineno))} | none | The right part "channo" is the number of the channel which is being assigned. It is an integer in the range 0 ≤ channo ≤ 15. The user may end the VFU statement with a semicolon and start another VFU statement without an id field to continue specification for the same channel or a different channel.<br>Example:<br>V1: VFU ASSIGN = (1,(7,14,28,35));<br>and<br>V1: VFU ASSIGN = (1,(7,14)),<br>ASSIGN = (1,(28,35));<br>are equivalent in application.<br>The right part "lineno" is the number of the output print line being assigned to a particular channel. |

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| VFU (cont.) | TOF = | value | 1 | The right part "value" specifies the number of lines from the top of the output page to the first print line on the page. |
| | BOF = | value | 66 | The right part "value" specifies the number of lines from the top of the output page to the last print line on the page. |

Points to Note

1. The VFU statement must precede the print line structure statements "LINE VFU =" described in the previous section of this chapter.

2. Top-of-form and bottom-of-form specifications are independent of channel assignments.

3. Top-of-form should be less than or equal to the smallest channel value assignments.

4. Bottom-of-form should be greater than or equal to the largest channel value assignments.

5. There are no default assignments for any channel, including channels 1 and 12.

6. Any unspecified channel assignment causes a print and space 1 line operation. Under some vendor formats, the default may be space 1 line and print, according to what is consistent with the carriage control set.

7. Multiple line numbers may be assigned to the same channel number. This simulates the vertical tabbing feature of an impact line printer where a skip to channel command causes transition to the next punched hole in the specified channel of the paper tape. This tape, which controls the printer, facilitates spacing a fixed number of lines down the print page. There may be multiple punches in any vertical format channel on the impact printer's tape. As shown in the example in Figure 5-2, a skip to channel command in the 9700 causes selection of the next line number in the ASSIGN list (for that channel) larger than the current line number, with page transition and alignment to the first line number in the list if none is larger.

8. An example of a coded VFU statement is shown in Figure 5-2.

```
V1: VFU     ASSIGN = (1,5),
            ASSIGN = (2,(10,15,20,25,30,35,40,45,50)),
            ASSIGN = (12,55),
            TOF = 5, BOF = 55;
```

Interpretation

1. Top-of-form is assigned to line number 5; bottom-of-form is assigned to line number 55.

2. Three channels (1, 2, and 12) have been assigned. Channel 2 has been assigned to nine line numbers.

3. Assume the printing system is printing a report, and the current line number is eleven:

   a. If a "skip to channel one and print"[t] command is issued, a page transition will occur.

      Printing begins on line 5 (assigned to channel one), which is top-of-form on the new page.

   b. If a "skip to channel two and print" command is issued when the current line number is 11, the next line to be printed is line 15 of the current page. Lines 10, 15, 20, and so on, are also assigned to channel two, but since the current line number is 11, the next consecutive line number assigned to channel two greater than 11 is line 15.

   c. If a "skip to channel twelve and print" command is issued, the next line to be printed is line 55 of the current page. Line 55 is assigned to channel twelve.

Figure 5-2. VFU Statement, Coding Example

[t] Printer carriage control operations are defined by the user via the printer carriage control statement (PCC) described in the subsequent section. The standard carriage control tables are described under the corresponding vendor format description in the 9700 Tape Format manual.

## Printer Carriage Control

The PCC statement enables the user to create a table of one-byte printer carriage control (PCC) codes and define their action. Line spacing, skip to channel, and printing actions are all defined via this command. The user specifies an identifier (of the type "ac") when defining the PCC table and references it in the LINE command.

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| [ac:] PCC | INITIAL = | {BOF / TOF} | TOF | "ac" is a 1 to 6 character statement identifier that is referenced by the PCCTYPE = id portion of the LINE command. One character must be alphabetic; the others may be alphanumeric.<br>The right part {BOF / TOF} identifies the initial reference point for printing a page.<br>If "TOF" is specified, the control program will perform the first spacing, skipping, or printing action from top-of-form. If "BOF" is specified, the control program will perform the first spacing, skipping, or printing action from bottom-of-form. |
| | DEFAULT = | ANSI<br>B2500<br>B2700<br>B3500<br>B3700<br>B4700<br>B6700<br>H2000<br>H6000<br>IBM1401<br>IBM1403<br>US70<br>UNIVAC<br>XEROX<br>ccln<br>NONE | PSP1 (print and space one line) | The left part "DEFAULT" allows the installation to select a set of printer carriage control codes. A specific table may be selected or a "ccln" can be defined.<br>The right part "ccln" specifies the action to be performed when a code has not been specifically assigned. The assignment codes for various actions are described under "ASSIGN = " (right part "ccln" below). The system default PCC tables are defined in 9700 standard EBCDIC (see Appendix C). |

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| PCC (cont.) | ASSIGN = | { (byte, ccln) / (byte, (ccln,...,ccln)) } | | The right part "byte" is the printer carriage control byte being defined. Its value is in the range 0 ≤ byte ≤ 225 (X'00' to X'FF').<br><br>The right part "ccln" specifies the action that should be taken when the printer carriage control byte defined in "byte" above is encountered.<br><br>The right part "ccln" may be any of the following definitions:<br>([{TOF, / OVR, / IGN,}] [{SPm / SKn}] [{P / N}] [{SPm / SKn}])<br><br>If TOF is specified, and "byte" causes bottom-of-form (BOF) to occur, the printing system is to go to top-of-form (TOF) on the next page and stop spacing.<br><br>If OVR is specified, and "byte" causes bottom-of-form (BOF) to occur, the printing system is to go to top-of-form (TOF) on the next page and continue spacing.<br><br>If IGN is specified, and "byte" causes bottom-of-form (BOF) to be encountered, bottom-of-form is to be ignored and spacing is to continue through the end of the physical page when page transition to top-of-form occurs and spacing is continued.<br><br>OVR is the default and need not be specified.<br><br>The fields [{SPm / SKn}] [{P / N}] [{SPm / SKn}]<br><br>Field 1 Field 2 Field 3<br><br>specify the action that is to be taken when "byte" is encountered. Each of the three fields is optional; however, at least one field must be specified.) |

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| PCC (cont.) | ASSIGN = (cont.) | | | Field 1:<br>SPm Space m lines before printing ($0 \leq m \leq 15$).<br>SKn Skip to channel n before printing ($0 \leq n \leq 15$).<br>Field 2:<br>P Print the output data at the line number computed after Field 1 is processed.<br>N No printing is to occur for this record.<br>If Field 2 is not specified, no printing is to take place for this record.<br>Field 3:<br>SPm Space m lines after printing ($0 \leq m \leq 15$).<br>SKn Skip to channel n after printing ($0 \leq n \leq 15$). |
| | MASK = | value | X'FF' | The right part "value" specifies an eight-bit value to be ANDed with the printer carriage control byte being defined after translation, if any, to mask off bits from the code which are not relevant to the operation being specified by the code. |
| | ADVTAPE = | {YES / NO} | YES | The right part is used to specify whether the printing system is to advance to a new page when two successive channel skip commands are issued with no intervening print.<br>For example, on most printers, the following actions:<br>PSK1 Print and skip to channel 1<br>SK1N Skip to channel 1, do not print<br>would cause a blank page to be output. However, on a 1403 printer, these actions would not cause a blank page to be output. |

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| PCC (cont.) | ADVTAPE (cont.) | | | "YES" specifies that multiple skips will be honored.<br>"NO" specifies that multiple skips will result in only one skip action being taken. |

Points to Note

1. Multiple user-defined PCC tables are allowed; only one may be without a statement identifier. The corresponding PCCTYPE left part on the LINE command references each table via a statement identifier. The right part USER can be used to reference the user-defined PCC table in which no statement identifier coded.

2. The "DEFAULT =" left-right parts must precede any "ASSIGN =" left-right parts. Any preceding "ASSIGN =" left-right parts will not be incorporated into the PCC table.

3. The user may end a PCC statement with a semicolon and start another PCC statement to continue specification of the carriage control codes. Multiple PCC statements may be used within a single PCC table definition as long as there are no intervening non-PCC statements.

4. Consecutive "byte" right parts need not be specified. Thus, the statements

   ASSIGN = (X'60', SP1), ASSIGN = (X'61', SP2), ASSIGN = (X'62', SP3);

   can be coded in the single statement ASSIGN = (X'60', (SP1, SP2, SP3));

5. If under the LINE command, byte translation is specified (i.e., TRAN is used for right part of PCC left part), then the PCC control byte is translated into 9700 standard EBCDIC (see Appendix C) before being applied. This means that the right part "byte" on the ASSIGN keyword must be specified as text (e.g., '1') or as an EBCDIC translation of the PCC byte (e.g., X'F1').

6. Figure 5-3 contains an example of a coded PCC statement. This user defined table of printer carriage control codes would be referenced by the statement PCCTYPE = PCCEX on the LINE command.

```
PCCEX: PCC      INITIAL = TOF,  DEFAULT = PSP1, MASK = X'FF', ADVTAPE = YES,
                ASSIGN = (X'00'  ,(P, PSP1, PSP2, PSP3,PSP4,PSP5, PSP6,PSP7,
                                   PSP8,PSP9,PSP10,PSP11,PSP12,PSP13 PSP14,
                                   PSP15));
          PCC   ASSIGN = (X'40'  ,(P,SP1P, SP2P,SP3P,SP4P,SP5P,SP6P,SP7P,
                                   SP8P,SP9P,SP10P,SP11P,SP12P,SP13P,SP14P,
                                   SP15P));
          PCC   ASSIGN = (X'80'  ,(PSK0,PSK1,PSK2,PSK3,PSK4,PSK5,PSK6,
                                   PSK7,PSK8,PSK9,PSK10,PSK11,PSK12,PSK13,
                                   PSK14,PSK15));
          PCC   ASSIGN = (X'C0'  ,(SK0P,SK1P,SK2P,SK3P,SK4P,SK5P,SK6P,
                                   SK7P SK8P,SK9P,SK10P,SK11P,SK12P));
```

Figure 5-3. Coded PCC Statement, Example

## Copy Modification Feature

INTRODUCTION

The Copy Modification feature (also referred to as spot carbon) offers the ability to modify 9700 output on a per copy basis. It allows the user to replace certain parts of report output on selected copies with specified static data, or to specify the placement of font change requests within variable data.

This feature is controlled through the use of the Copy Modification Entry (CME) command. CME defines a rectangular space upon the printed page within which printed data will be replaced with a substitution string. More than one CME may be applied to a job. CMEs may be coded as part of the job descriptor library or created as a separate file such that it may be referenced by one or more job descriptor libraries. This is described more fully in "Cataloged CMEs" of this chapter.

The MODIFY left part on the OUTPUT command (and the DJDE command MODIFY) relates the CME to the particular copies to be changed. The MODIFY left/right parts are described in a previous section of this chapter.

An example of CME is shown in Figure 5-4.

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| ac: CME | | | | "ac" is a 1 to 6 character statement identifier that is referenced by the MODIFY = cme-id portion of the OUTPUT command (or the MODIFY DJDE command). |
| | LINE = | { n / (n,m) / (n,-) } | none | The right part "n" is the initial line of the Copy Modification rectangle.<br>The right part "m" is the number of lines within the Copy Modification rectangle. If not specified, the information will apply only to the line indicated by "n". To indicate that the information is to apply to all lines on a page beginning with the line indicated by "n", a dash character ("-") may be used in place of "m". |
| | POS = | p | 1 | The right part "p" specifies the initial character position of the rectangle in the print line. It must not exceed the position number of the last print position on a print line. |

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| ac: CME (cont.) | FONT = | value | none | The right part "value" specifies the index into the font list on the "FONTS =" right part of the PDE command. (Note: a PDE command is selected by the "FORMAT =" right part of the OUTPUT command or the DJDE FORMAT command).<br><br>"value" may range from 1 to n, where n is the number of different fonts specified by the FONTS option. A value of 1 specifies the first font in the FONTS option, 2 the second, etc.<br><br>A font specification applies to input variable data as well as static CME data. If a line number (LINE =) and character position (POS =) but no insertion text (CONSTANT =) are specified, the font change specified will apply to input variable data at the position specified. |
| | CONSTANT = | sc | none | The right part "sc" identifies the characters that will be printed within the rectangle. "sc" is any string constant (see Chapter 3 for definition of string constants). The width of the Copy Modification rectangle is determined by the number of characters specified by "sc". More than one "sc" is allowed. |

Points to Note

1. The CME statement must precede the output processing statement which references it, or it must be represented in a control file cataloged in the CME directory.

2. There is no default for the CONSTANT = sc portion of the CME command. It must be specified unless the font specification is to apply to the input variable data.

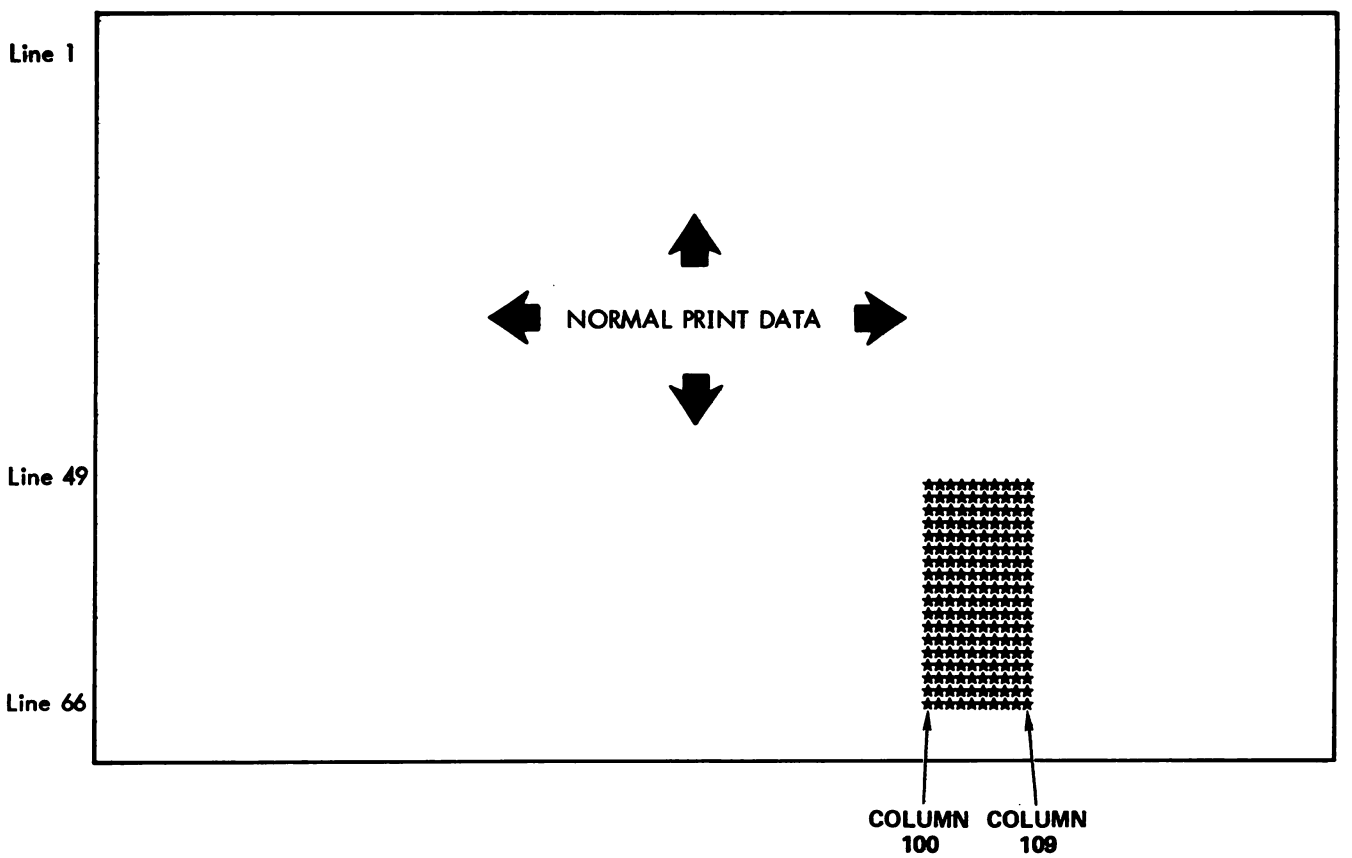
3. In order for the character string specified to be printed on a line, the normal data must appear on that line; that is, if a line is skipped, either by line spacing or channel skipping, then no character substitution will occur.

4. Copy Modification (spot carbon) is not implemented if a report is printed in the uncollated mode.

5. Within a text string, as specified with the CONSTANT = sc, the character "#" may be used as a lower-case toggle. When a text string is encountered, it is assumed that characters are to be inserted into the print line as they appear in the text string (in upper case normally). If a "#" is encountered the lower-case mode is invoked and all letters after the "#" will be considered lower-case until another "#" is encountered. The sequence "##" is used to indicate that the character "#" is to be inserted and is not to be treated as toggling lower-case-restrict mode.

6. Multiple lines may be specified and multiple columns may be specified for each line. Multiple line specifications must be given in ascending (top to bottom of page) order. Multiple column specifications for a line range must be given in ascending (left to right ) order. There also may be multiple text specifications following a column specification. These are combined to form a single text string. Font specifications may be specified at any point. The last font specified remains in effect until another font is specified. An example of a CME with these characteristics is as follows:

```
XYZ: CME    LINE = (1,10), POS = 40, FONT = 2, POS = 80, FONT = 3,
            LINE = (11,20), POS = 1, FONT = 3, POS = 40, FONT = 1, POS = 80, FONT = 2,
            LINE = (31,50), POS = 1, FONT = 2, POS = 40, FONT = 3, POS = 80, FONT = 1;
```




The CME statement C1: CME LINE = (49,18), POS = 100, CONSTANT = (10) '*' which when referenced by an OUTPUT MODIFY = C1 command would result in the above page as output.

Figure 5-4. Example of a CME

### Short Form CME Specifications

CME specifications may be given in short form to minimize the specification length. In short form, only the first character of a keyword need be given, equals signs are omitted, and commas are not inserted except where necessary to avoid ambiguity. The following is an example of CME specifications shown in standard and abbreviated form.

Standard Form:

```
CME12: CME    LINE = 47, POSITION = 1, FONT = 5, CONSTANT = (5)'*',
              LINE = 48, POSITION = 1, FONT = 1,
              LINE = 49, POSITION = 10, CONSTANT = 'ABCD';
```

Short Form:

```
CME12: CME    L47, P1, F5, (5) '*', L48, P1, F1, L49, P10, C'ABCD';
or
CME12: CME    L47P1F5(5)'*'L48P1F1L49P10'ABCD';
```

### Cataloged CMEs

CME commands need not be part of a user's job descriptor library. They may be created as separate disk files and used as if they were part of the job descriptor library that references them. This is done by creating a CME command within a file using the PDL processor to compile it. Only one CME command per file is allowed. PDL will create a control file on disk, cataloged in the CME directory.

When the CME is referenced (by OUTPUT MODIFY) in a job descriptor library, and the CME does not exist within the JDL, the system will search the CME directory for the named CME. It will be loaded into memory for use in processing the report.

The DJDE command MODIFY can also be used to dynamically associate a cataloged CME file with report processing. Further details are contained in Chapter 6. For the "MODIFY = cme-id" DJDE command the CME statement must be cataloged on disk.

## Page Descriptors

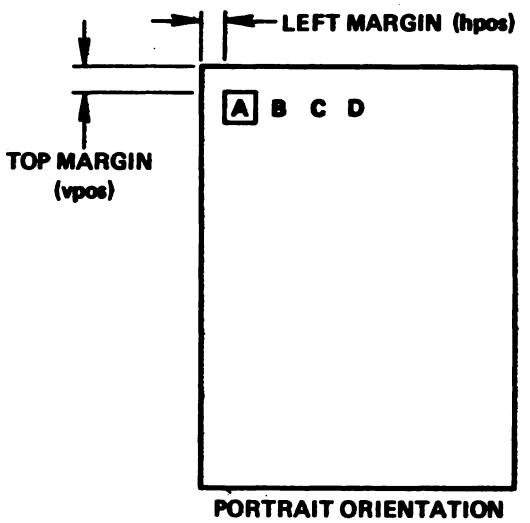
The PDE command specifies a Page Descriptor Entry (PDE) which describes formatting information for each page of a report. This formatting information includes page orientation (landscape or portrait), location of the beginning print line for each logical page and the fonts to be used.

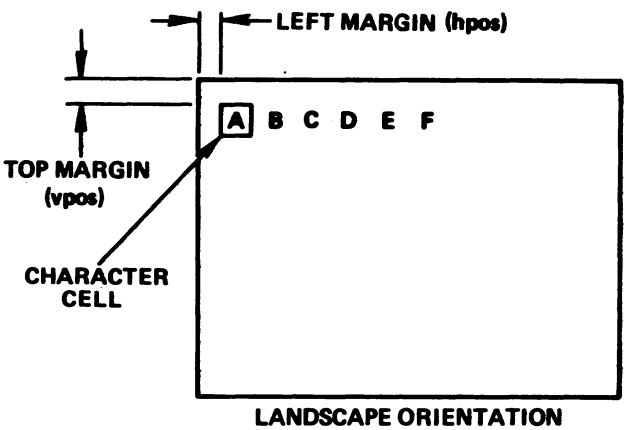
PDEs may be coded as part of the job descriptor library or created as a separate file such that it may be referenced by one or more job descriptor libraries. PDEs are called out on the "FORMAT =" left/right part of the OUTPUT command (or a DJDE FORMAT command). Typical PDE specifications are provided on the OSS system tape as described in Table 5-1. Figures 5-5, 5-6 and 5-7 illustrate the considerations in the formatting of a page with the PDE command.

PDEs created as a separate disk file may be used as if they were part of the job descriptor library that references them. This is done by creating a file of PDE commands (in the JSL directory) and using the PDL processor to compile them. PDL will create a control file on disk, cataloged in the PDE directory. When the PDE is referenced, the system will search the PDE directory for the named PDE, and if found, will load it into memory for use in processing a report. See "PDE and CME Compilation" in Chapter 3 for further details.

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| ac: PDE | PMODE = | {PORTRAIT / LANDSCAPE} | LANDSCAPE | The left part "PMODE" specifies the printing mode for each physical sheet.<br>"LANDSCAPE" indicates that printing is to be parallel to the long edge of paper.<br>"PORTRAIT" indicates that printing is to be parallel to the narrow edge of paper.<br>Figure 5-5 illustrates these two print modes. |
| | FONTS = | {(f1[[, f2]...fn]) / ((f1, s1)[[,(f2, s2)],...])} | none | FONTS specifies the fonts to be used in printing variable input and CME data. Each "fi" (i = 1, n) specifies a 1-6 character identifier corresponding to a font cataloged on the system disk. Each "si" value specifies an optional override line-spacing value to be associated with the font. Each "si" spacing value is a decimal value specifying lines per inch.<br>If an override line spacing value is specified, then lines printed using the font will cause the indicated line spacing to be performed after the line using the font. If different fonts are specified on the same print line, then the line spacing value specified for the font of the last character in the line will be used to determine the position of the next print line. |

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| ac: PDE (cont.) | FONTS = (cont.) | | | At least one font must be specified in a PDE command.<br><br>See the section "Copy Modification Feature" in this chapter for further information regarding change of fonts within a print line. |
| | BEGIN = | (vpos [CM / IN], hpos [CM / IN]) | (0. IN,0. IN) | The "BEGIN" left part specifies the location of the starting print line for each logical page on a physical page.<br><br>"vpos" specifies the vertical position of the first character of the first print line on a logical page. IN specifies inches; CM specifies centimeters. The default type measurement is inches. "vpos" may be specified as a decimal number with up to three digits to the right of decimal point (e.g., 0.563 IN, 2.35 CM, and 4.3 are all legal specifications).<br><br>"hpos" specifies the horizontal position of the first character of the first print line on a logical page. IN and CM have the same meaning as for "vpos".<br><br>In specifying the location of the beginning of a print line on a logical page, measurement is performed by viewing the physical page in the mode in which it is to be printed, e.g., landscape or portrait. |




Note: Measurements refer to the location of the character cell that contains the first character to be printed.

Figure 5-5. Vertical and Horizontal Positions in Landscape and Portrait Modes

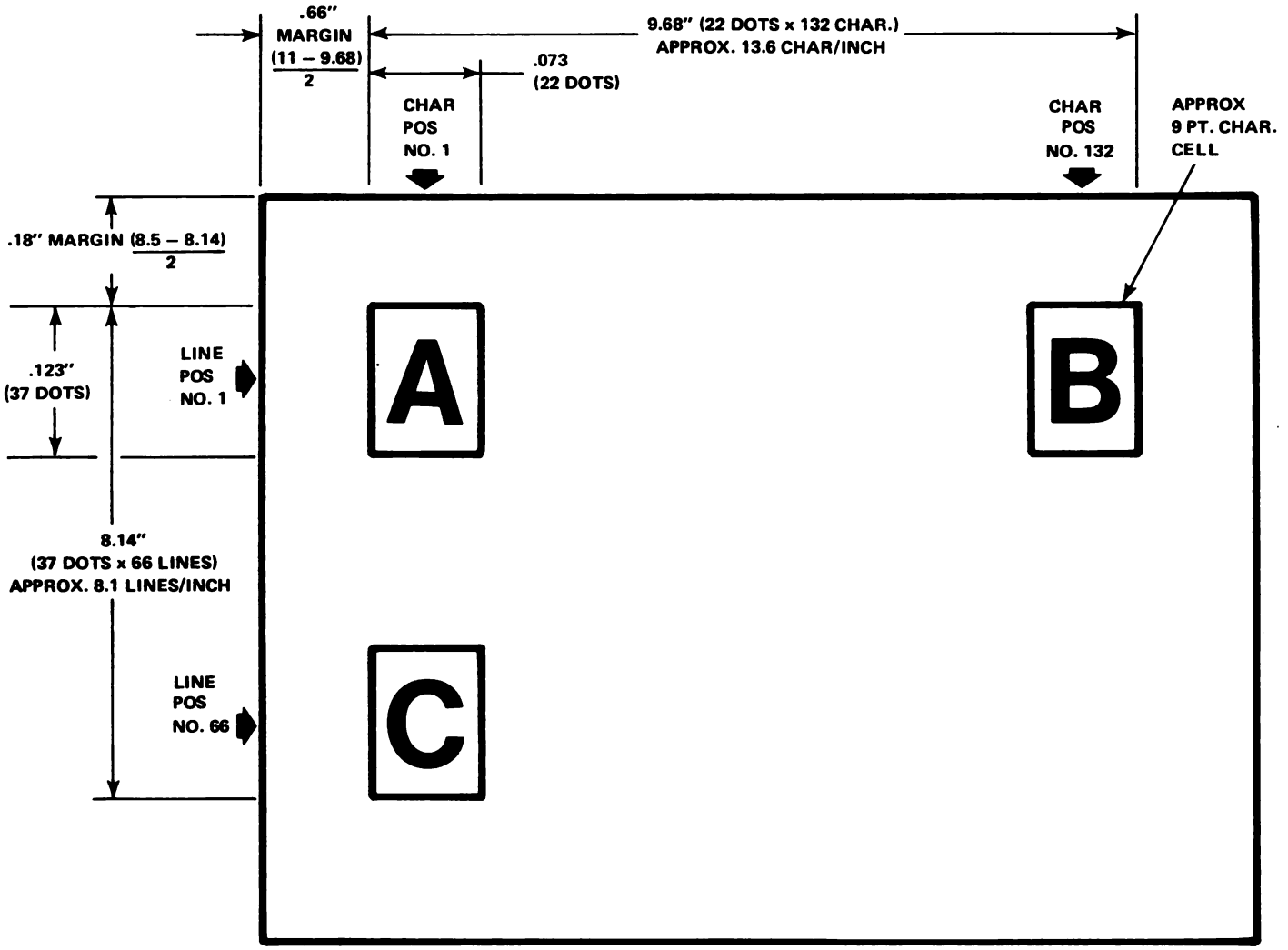



Figure 5-6. FMT1: Equavalent Impact Printer Format 6 Lines/Inches: Landscape Orientation

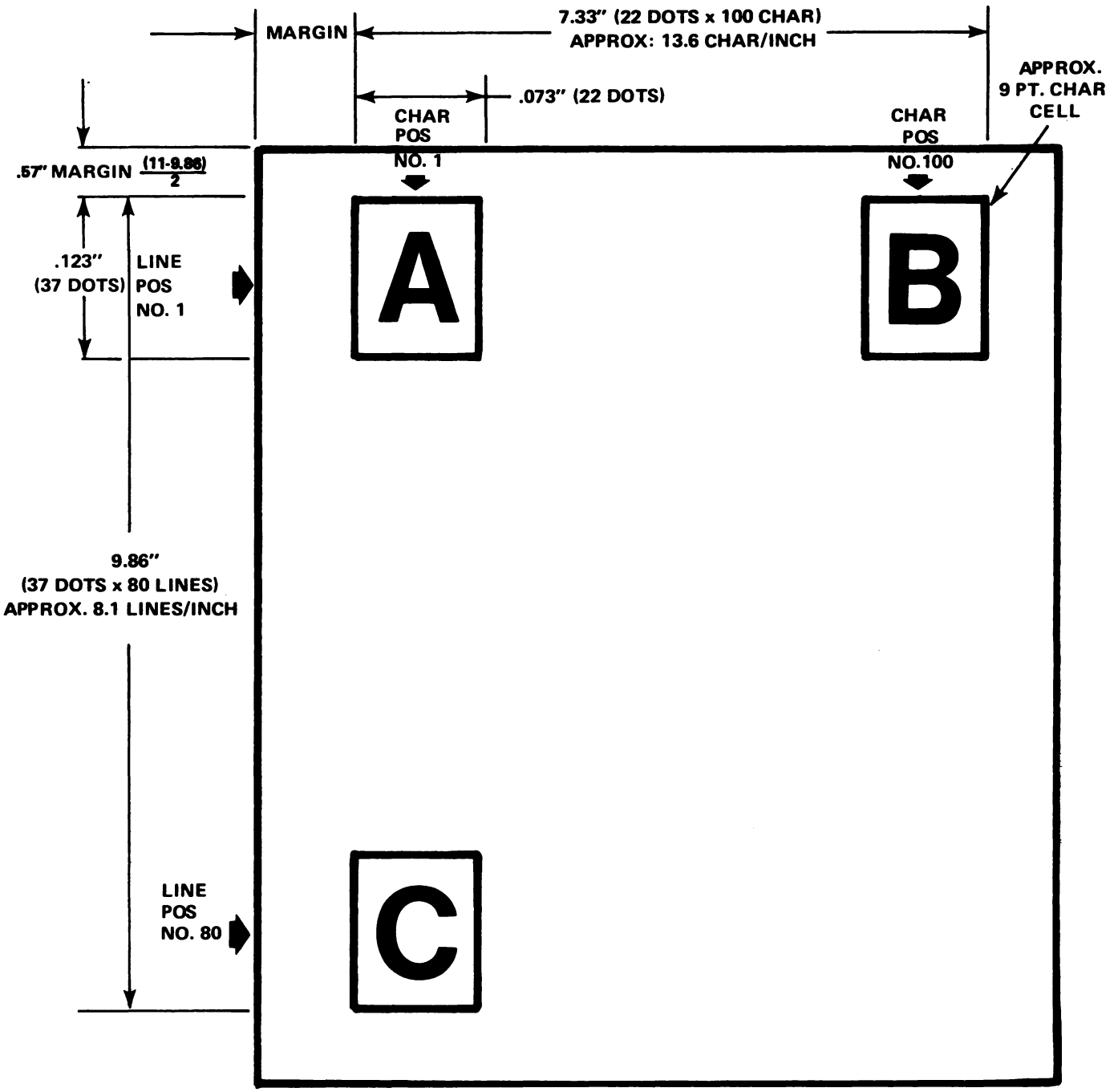



Figure 5-7. FMT6: Equivalent Impact Printer Format, 6 Lines/Inch: Portrait Orientation

## Operator Message Commands

Two commands are provided to permit the user to inform the operator of special conditions. The MESSAGE command displays user defined text to the operator and the ROUTE command allows the user to print identifying information on a page preceding a report. Examples of these commands are contained in Figures 5-8 and 5-9.

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| MESSAGE | | | | The MESSAGE command causes a message to be displayed to the operator on the keyboard/display during input processing or job printing. |
| | ITEXT = | {sc / (sc ,passnum) / NONE} | NONE | "sc" specifies the text message to be output (1-31 characters) to the operator during input processing.<br>"passnum" specifies the pass (copy ply) to which the message text applies. The message will be output to the operator just before processing of the indicated copy ply (pass) is commenced. If no pass number is specified, the indicated message will be output at the beginning of the first pass. |
| | OTEXT = | {sc / (sc, [, {passnum / END}] [, WAIT]) / NONE} | NONE | "sc" specifies the text message to be output (1-31 characters) to the operator during job printing.<br>"passnum" specifies the pass (copy ply) to which the text applies. The message will be output to the operator prior to beginning printing of the specified report ply. If no pass number is specified, the text will be output once at the beginning of printing the entire report.<br>END specifies that the text will be displayed after the last copy of the report is printed.<br>WAIT specifies that after the text is displayed, printing is to be suspended until the operator has responded with a CONTINUE command. |

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| ROUTE | | | | The ROUTE command is used to print a message or form on a separate sheet of paper preceding a report. |
| | RTEXT = | {sc<br>(sc, passnum [,line [,col]])<br>NONE} | NONE | The RTEXT left part is useful for printing text on a separate page preceding a report (or copy ply).<br>"sc" specifies the message to be printed (1-132 characters). It will be printed with the first font specified in the "FONTS =" left/right part of the PDE command.<br>"passnum" optionally specifies the pass (copy ply) to which the text applies. If not specified, the message will be printed at the beginning of the entire report.<br>"line" optionally specifies the line number on which the first line of a block of RTEXT message is to be printed. The default is line 1 for the first text string of the pass. Otherwise the default is the next line of the page.<br>"col" optionally specifies the column number at which the first character of a block of RTEXT message is to be printed. The default is column 1 for the first text string of the pass. |
| | RFORM = | {form-id<br>NONE} | NONE | The left part "RFORM" specifies a form to be printed on a separate sheet preceding the report ply or entire report.<br>"form-id" is the name of a file cataloged in the FDL directory. It is created by compiling FDL source statements with the FDL processor. Further information on forms and the FDL processor are contained in Chapter 10. |

The text: $\left\{\begin{matrix}\text{USER 1} & \text{USER 2} \\ \text{BLDG 1} & \text{BLDG 2}\end{matrix}\right\}$ is to be printed (in center of page) preceding the respective copies of a two copy landscape report. The page is 132 columns by 66 lines. The following ROUTE command will accomplish this:

```
ROUTE   RTEXT = ('USER 1', 1, 33, 64),
        RTEXT = ('BLDG 1' 1, 34, 64),
        RTEXT = ('USER 2', 2, 33, 64),
        RTEXT = ('BLDG 2', 2, 34, 64);
```

Figure 5-8. ROUTE Command Example

The following MESSAGE command will inform the operator that blue paper is required for copy 2 of a 4 copy report. Printing will be suspended at the appropriate point so that the operator can load the paper.

```
OUTPUT    COPIES = 4;
MESSAGE   ITEXT = ('COPY 2 WILL NEED BLUE PAPER'),
          OTEXT = ('LOAD BLUE PAPER', 2, WAIT),
          OTEXT = ('LOAD WHITE PAPER', 3, WAIT);
```

Figure 5-9. MESSAGE Command Example

## Accounting Information

Accounting information is automatically accumulated by the system on a report basis and also for overall system usage. The PDL command ACCT enables the user to request a printout of the accounting summary on a report basis and the system level REPORT USER command provides an installation with a printout of overall system usage statistics. The following sections define how accounting information is accessed and interpreted by the user.

### Accounting Data On Report Basis

The "USER =" left part of the ACCT command enables the user to request that an accounting summary be output at the end of the printing for each report. This summary consists of a single page of information containing job set-up information and counts of processing events. For example, the PDL statement ACCT USER = (BIN, TRAY); requests the system print an accounting summary page to the bin and sample print tray after each report is completed. An example of an accounting page is illustrated in Figure 5-10. Explanations of the various entries are provided in the next section.

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| ACCT | | | | The ACCT command permits an accounting summary to be printed at the end of each printed report. It also allows an installation to accumulate accounting statistics for print jobs under installation defined names. |
| | USER = | { NONE<br>BIN<br>TRAY<br>(BIN, TRAY) } | BIN | The right part "NONE" specifies that no accounting summary is to be output.<br>The other right parts specify where the one-page accounting summary is to be output for each report.:<br>BIN Summary is output to selected output bin only<br>TRAY Summary is output to sample print tray only<br>(BIN,TRAY) Summary is output to both output bin and the sample print tray |
| | DEPT = | sc | jdl name | The left part "DEPT =" allows an installation to accumulate accounting statistics for jobs on a "name" basis.<br>The right part "sc" is a string constant of up to 31 characters representing a "department code" or "name" under which accounting information will be maintained. Only alphanumerics (A thru Z, 0 thru 9) and the character ":" are allowed for the department code.<br>See section "Overall Usage Accounting" for further details. |

```
DATE:                    25 JAN 79  AT  12:16:20
DEPARTMENT:              H2SYS:JDL
JOB ID: 15               REPORT NO. 4
FILE ID:                 XEROX.001
INPUT PROCESSING TIME:   00:00:14
OUTPUT PROCESSING TIME:  00:00:24
REPORT COMPLETION CODE:  0
     DATA PAGES PRINTED:  90
     PAGES TO TRAY:       1
     PAPER PATH HOLES:    1
     DATA LINES PRINTED:  450
     TAPE MOUNTS:         1
     BLOCKS READ:         16
     BLOCKS SKIPPED:      0
     RECORDS READ:        450
     DJDE RECORDS READ:   0
     NUMBER OF COPIES:    1
     OVERPRINTS:          54
     COLLATE:             YES
     SF/MF:               MULTI
JDE,JDL USED:  01,H2SYS
ACCTINFO:                ACCOUNTING REPORT 1
INITIAL FONT LIST:       L0112B
INITIAL FORM LIST:       -NONE
INITIAL CME LIST:        -NONE
```

Figure 5-10. Sample Accounting Page

## Accounting Page Entry Definitions

Figure 5-10 contains an example of a report's accounting page. The entries are defined as follows:

DATE

Current date as entered by the user when booting the system from disk. The time (hh:mm:ss) represents the point at which Input Processing begins processing the input data for a report.

DEPARTMENT

This is either the character string entered as part of the ACCT command ("DEPT =") or a system default name. The system default name will be the name of the JDL under which the job was run with the characters ":JDL" appended to it.

JOB ID

This entry is the system generated identifier of the print job. It is automatically incremented by OSS each time a job is run. REPORT NO. is the number of the report within the current job.

FILE ID

This is the "File Identifier" field of the HDR1 label if it exists (see Tape Format Manual).

INPUT PROCESSING TIME

Elasped time (hh:mm:ss) for Input Processing to process the report from tape and pass the results to Output Processing. The functions of the Input Processor are discussed in section "Input Processor Functions" of Chapter 2.

OUTPUT PROCESSING TIME

Elasped time (hh:mm:ss) for Output Processing to print the report as sent to it from Input Processing. This time includes processing time of the report plus any time Output is waiting for paper jams to be cleared, bins raised, etc. The functions of the Output Processor are discussed in section "Output Processor Functions" of Chapter 2.

REPORT COMPLETION CODE

A code to indicate the completion status of the report. Completion codes are:

| CODE | MEANING |
|---|---|
| 0 | Normal completion. |
| 25 | Page spacing was invoked by operator (SPACE n PAGEs command). |
| 50 | A page was skipped because it exceeded the capability of the hardware or the data on one or more pages exceeded 8-1/2 x 11 inches. |
| 99 | Report was aborted. |

DATA PAGES PRINTED

A count of the data pages printed and delivered to the bin. This count does not include accounting, RTEXT, DJDE or delimiter pages.

PAGES TO TRAY

A count of the pages delivered to the tray. This includes SAMPLE pages and the printing of the tape label (if VOLUME PLABEL = YES is coded in JDE).

PAPER PATH HOLES

A count of paper path "holes" caused by user job characteristics. For example, a "hole" may be caused by a disk access required to load a previously unloaded form or font. Another way to cause a "hole" in the paper path is for a job or operator to change the output page destination (i.e., bin to tray or bin to bin). Holes caused by error recovery (disk read or CD/IG errors, for example) and those caused by an non-recoverable error are not included in the amount printed on the accounting summary.

DATA LINES PRINTED

A count of the lines printed the data pages delivered to the bin (DATA PAGES PRINTED). This does not include the number of lines on the accounting, RTEXT, DJDE, and delimiter pages. The count also does not include the lines printed for a user requested page number.

TAPE MOUNTS

A count of the number of tape volumes required in the processing of a report.

BLOCKS READ

Number of data blocks read from the input tape.

BLOCKS SKIPPED

Blocks skipped as result of operator initiated command "MOVE n BLOCKS" (see "BLOCK/FILE Positioning" in Chapter 7).

RECORDS READ

Records read from input tape according to format defined in JDE.

DJDE RECORDS READ

Number of DJDE records read from input tape.

NUMBER OF COPIES

The number of printed copies requested of the report (may not be the number actually printed, if for example the job was aborted). The number of copies to be printed is defined by the user in the JDE (COPIES left part of OUTPUT command) and may be overridden by DJDEs or by the "copies" option on the START command.

OVERPRINTS

The number of overprint lines in the report. If the IGNORE option of the LINE statement (OVERPRINT left part) is coded in the JDL, then the accounting entry will be 0.

COLLATE

Entry will be YES or NO depending on COLLATE left part of OUTPUT command. The default is to collate output pages. COLLATE may also be modified by a DJDE.

SF/MF

Entry will be SINGLE (single-report mode) or MULTI (multi-report mode). The control of single or multi-report mode is set-up in the users JDE (RMODE left part of VOLUME command) and may be overridden on the START command (see "Starting a Job" in Chapter 7).

JDE, JDL USED

Names of the Job Descriptor Entry and Job Descriptor Library as specified on the START command.

ACCTINFO

Text from the delimiter record when left/right part ACCTINFO (RSTACK command) is coded in JDE.

INITIAL FONT LIST

Names of the fonts referenced in the JDE. Fonts are defined in the PDE command (FONTS left part) and referenced for use in a JDE on the OUTPUT command (FORMAT left part). If the font list is modified by DJDEs, then the message "DJDE MODIFIED" will be printed along with the original list of font names.

INITIAL FORM LIST

Names of the forms specified in the JDE on the OUTPUT command (FORMS left part). If other FORMS are used via DJDEs then the message "DJDE MODIFIED" will be printed along with original list of form names.

INITIAL CME LIST

The names printed are the identifiers of CMEs referenced in a JDE. CMEs are defined with the CME command (along with an identifier) and referenced in a JDE with the OUTPUT command (MODIFY left part). If CMEs are used in DJDE processing then the message "DJDE MODIFIED" will be printed along with the original list of names.

### Overall Usage Accounting

The system automatically accumulates accounting data for each report that is printed. An installation has the option of having the system accumulate this information under a specified "name" (i.e., department code, account, etc.) or accumulate it under a system default name.

The "DEPT =" left part of the ACCT command allows an installation to define a "name" under which the data will be accumulated. If this left part is not specified, then data will be accumulated under a system created name. This name is the identifier on the Job Descriptor Library's SYSTEM (or JDL) command plus the characters :JDL appended to it. Thus, if a JDL identifier is H2SYS, then the default name will be H2SYS:JDL.

The system level command ACCOUNT must also be used in conjunction with the "DEPT =" left part. Its function is to create the name in the accounting file which corresponds with the name specified in the right part of "DEPT =". It is necessary to enter this name only once with the ACCOUNT command, but must be done before a print job is run using the "DEPT" name. If the name is not entered with the ACCOUNT command then the accounting data for a print job will be accumulated in the "NODEPT" entry in the accounting log. Usage of this command is described in section "Accounting File Maintenance" of Chapter 7.

The following example illustrates the use of the PDL command ACCT and the system level ACCOUNT command.

When the system level command

```
ACCOUNT    ADD, OPERATIONS
```

is entered on the keyboard/display the name "OPERATIONS" is added to the current list of names in the accounting file.

The PDL statement

```
ACCT USER = TRAY, DEPT = 'OPERATIONS'
```

requests OSS to deliver one accounting page to the sample print tray after a report is printed. Each time a print job is run with the above statement in the JDE, then this job's accounting statistics will be updated in the accounting file under the name "OPERATIONS".

### Installation Accounting Report

The REPORT USER system level command is used to print a copy of the accumulated accounting statistics. It prints out accounting data for each name entered into the accounting file by the ACCOUNT command. The REPORT USER command is discussed in section "Accounting and System Activity" in Chapter 7. A sample of the output is shown in Figure 5-11.

```
DEPARTMENT/JDL          PRINTING SUMMARY          TIME SUMMARY -HR, MIN-
                        PAGES =  21               INPUT  =  0       ,1
NODEPT                  LINES =  2200             OUTPUT =  0       ,2
                        PAGES TO TRAY = 0

                        PROCESSING SUMMARY        TAPE SUMMARY
                        JOBS    =  2              MOUNTS          = 0
                        FILES   =  2              BLOCKS READ     = 220
                        REPORTS =  2              BLOCKS SKIPPED  = 0

DEPARTMENT/JDL          PRINTING SUMMARY          TIME SUMMARY -HR, MIN-
                        PAGES =  105              INPUT  =  0       ,5
OPERATIONS              LINES =  1700             OUTPUT =  0       ,7
                        PAGES TO TRAY = 0

                        PROCESSING SUMMARY        TAPE SUMMARY
                        JOBS    =  1              MOUNTS          = 1
                        FILES   =  7              BLOCKS READ     = 1100
                        REPORTS =  7              BLOCKS SKIPPED  = 0

DEPARTMENT/JDL          PRINTING SUMMARY          TIME SUMMARY -HR, MIN-
                        PAGES =  3                INPUT  =  0       ,0
TASKS$:JDL              LINES =  100              OUTPUT =  0       ,0
                        PAGES TO TRAY = 0

                        PROCESSING SUMMARY        TAPE SUMMARY
                        JOBS    =  3              MOUNTS          = 0
                        FILES   =  3              BLOCKS READ     = 0
                        REPORTS =  3              BLOCKS SKIPPED  = 0
```

Figure 5-11. System Usage Accounting Report

DEPARTMENT/JDL entries (as in Figure 5-11) are those supplied by OSS (NODEPT and TASKS$:JDL) and those entered by the user with the ACCOUNT command (OPERATIONS).

NODEPT is the name under which accounting statistics are accumulated for print jobs without name entries in the accounting file.

TASKS$:JDL is a system supplied name under which accounting is accumulated for printing done by OSS tasks such as FDL, PDL and the EDITOR.

## Abnormal Command

The ABNORMAL statement of the Xerox 1200 Computer Printing System is not implemented on the 9700. The ABNORMAL command and its left/right parts are acceptable by PDL for compatibility purposes only. Appendix A contains the syntax of the command.

# 6. DYNAMIC JOB DESCRIPTOR ENTRIES

## Introduction

Dynamic Job Descriptor Entries (DJDEs) are used to dynamically modify the printing environment established by a Job Descriptor Entry (JDE). DJDE commands modify previously established JDEs via an operator initiated command file or as part of the input data tape. Dynamic job descriptor entry processing enables certain JDE parameters to be changed on a report-to-report, page-to-page and record-to-record basis. The DJDE commands which accomplish this processing are summarized in Table 6-1. Table 6-3 contains a detailed description and syntax of each command.

Some of the benefits derived from changing these job parameters with DJDEs are as follows:

- The printing system does not stop between reports nor is operator intervention required. The operator starts up a print tape on the 9700 and typically returns to it only when minor operational activity is required.
- Forms may be changed on a page-to-page basis.
- Many variations on VFU channel, margin, and top- and bottom-of-form assignments may be applied to reports as they are created instead of being stored in the 9700 via job descriptor entries. Thus, the number of JDEs required for job processing is reduced.
- Unusual processing requirements may be satisfied through DJDE adjustments to the print line length and starting print position.
- The required number of copies is automatically generated, with routing or distribution notification sent to the operator.

## Tape and Disk DJDEs

There are two ways to use DJDEs to modify JDE parameters. First, via a user created DJDE file on disk which will modify the JDE when a print job is initiated. The other way is to create DJDE records on the print tape which will modify the JDE values as the tape is being processed. DJDEs on a disk file are used only at job initiation, whereas DJDEs on the user's tape can modify JDE values dynamically on a report, page or record basis.

The following discussion will be concerned with the DJDEs that are part of the input data tape. DJDEs initiated from disk files are discussed in section "Command File DJDEs" of this chapter.

## Report, Page and Record Oriented DJDEs

As listed in Table 6-1 there are three types of DJDE commands: report, page, and record oriented. Report oriented commands are associated with the report (or report-ply) as a whole and are placed at the beginning of a report prior to the first data record.

Page oriented commands effect changes to specific pages within a report and can change these pages differently in different copies. Such commands may be placed within the report itself and would take effect at the next page boundary. They may also appear at report boundaries to effect changes to all pages in a report (or report-ply).

Record oriented commands will take effect at the next record following the completion of the set of DJDE commands (i.e., after an "END;" statement). These commands also may appear at report boundaries to effect changes to all pages in a report (or report-ply).

Table 6-1. DJDE Command Summary

| DJDE Command | Orientation | | | Description |
|---|---|---|---|---|
| | Report | Page | Record | |
| JDL or SYSTEM | X | | | Invokes the named JDL at the next report boundary. |
| JDE or JOB | X | | | Invokes a JDE within the specified JDL at the next report boundary. |
| COPIES | | X | | Specifies the number of copies to be printed starting at the next page boundary. |
| FORMS | | X | | Specifies the form to be merged on printed pages. |
| COLLATE | X | | | Invokes collated or uncollated mode at the next report boundary. |
| MODIFY | | X | | Specifies the Copy Modification Entries (CME) to be used for variable data replacement and/or font switching operations on input data. |
| FORMAT | | X | | Specifies the Page Descriptor Entry (PDE) to be used to set up formatting control. |
| NUMBER | | X | | Specifies page numbering control starting with the next page. |
| MARGIN | | | X | Specifies left printing margin within a logical page. |
| DATA | | | X | Specifies location and length of printable data within a user's input record. |
| OVERPRINT | | | X | Specifies overprint control for input data. |
| ASSIGN | | | X | Specifies an assignment of channel to a page line position. |
| TOF | | | X | Specifies the line number corresponding to top-of-form. |
| BOF | | | X | Specifies the line number corresponding to bottom-of-form. |
| ITEXT | X | | | Specifies a message to be displayed to the operator during input processing. This becomes effective at the next report boundary. |
| OTEXT | X | | | Specifies a message to be displayed to the operator during job printing and, optionally, suspends printing. This becomes effective at the next report boundary. |
| RTEXT | X | | | Specifies routing information to be printed on a separate printed sheet prior to a report copy or an entire report. This becomes effective at the next report boundary. |
| RFORM | X | | | Specifies a form to be printed on a separate sheet prior to a report copy or an entire report. Becomes effective on the next report boundary. |
| FONTINDEX | X | | | Specifies whether or not data records contain a font index. |
| C | | | | Permits inclusion of commentary in DJDE records. |
| END | | | | Signifies the end of information within a DJDE. |

## DJDE Commands on Tape

DJDEs to be used as part of the input data tape consist of two parts. The first part is an IDEN statement which is specified within the Job Descriptor Library. This statement notifies the system that DJDEs may be contained on a job input tape. It also describes the search criteria for identifying the DJDE on the input tape.

The second part is the actual DJDE record (or records) which will be enbedded in the data records. Each DJDE record contains an identification field (which matches the search criteria specified in the Job Descriptor Library) and a series of left/right parts which describe the actual job descriptor entry changes to be applied to the report.

The subsequent two sections of this chapter describe the two parts of the DJDE in detail.

### JDL Specification: IDEN Statement

The IDEN statement (coded in a user's JDL) is used to notify the system that a DJDE record (or records) may be contained on the job input tape. The statement also describes the characteristics of the DJDE record prefix so the system can identify and locate the record. The statement syntax and interpretation are described in Table 6-2. An example of a coded IDEN statement is shown in Figure 6-1.

Table 6-2. IDEN Syntax and Interpretation

| Command | Left Part | Right Parts | Default | Interpretation |
|---|---|---|---|---|
| IDEN | PREFIX = | sc | none | The right part "sc" is the DJDE prefix. It is a byte string of up to 255 characters represented as a hexadecimal, octal, or character constant (see "PDL CONSTANTS in Chapter 3). "sc" must be defined in all DJDE report records on the input tape. |
| | SKIP = | value | 1 | The right part "value" specifies the number of bytes (beginning at 0) from the beginning of the user portion of the record to the beginning of the DJDE commands (i.e., it is the offset to the starting column of the DJDE left/right parts). |
| | OFFSET = | value | 0 | The right part "value" specifies the number of bytes (beginning at 0) from the beginning of the user portion of the record to the beginning of the prefix string constant of the DJDE record. |
| | OPRINFO = | {YES / NO} | NO | The right part "YES" specifies that a sheet which contains the contents of the DJDE is to be printed and delivered to the bin.<br>The right part "NO" specifies that no sheet will be printed and delivered to the bin. |

The IDEN statement coded below describes the DJDE report record shown in Figure 6-2.

```
IDEN PREFIX = 'RTEST', SKIP = 9, OFFSET = 3, OPRINFO = YES;
```

Figure 6-1. Sample IDEN Statement Coding

## DJDE Report Record Specification

The actual DJDE records are created by the user as part of the report. They are looked for by the system only if a DJDE identifier has been specified by an IDEN command within the JDE used to process the job. The DJDE data consists of one or more fixed format records each of which may be up to the record length specified for the job input data. For each record, the prefix (the identification field) and the DJDE left/right parts must begin in the same location. The DJDE records are terminated by an "END;" statement. All specified DJDE information will be applied at the next record, page or report boundary after the "END;" is encountered. There may be multiple DJDE command sequences in a job. Each set modifies only the specific parameters mentioned within the DJDE. The DJDE left/right part options are defined in Table 6-3.

The following points should be considered when preparing to introduce DJDE records into job tape reports. Examples of DJDE records are illustrated in Figures 6-2 and 6-3.

- DJDE records need not be consecutive, as none of the parameters are applied until "END"; is encountered. However, it is highly recommended that the DJDE records be consecutive, especially when used with delimiter records. Records should contain as many DJDE commands as possible.

- For delimiter mode stacked report processing, DJDE commands may be placed in the report body. They may also be placed within, before, or after the delimiter records.

- For change mode stacked reports the DJDE commands may be placed within the body of the stacked report, where the DJDE records contain the same change field contents as the report to which the DJDE applies.

- Report oriented DJDE commands must appear within the report to which they apply but prior to the first data record; otherwise the commands will not be executed when expected.

- The DJDE record may contain more than one left/right part except for the case of a comment, which must be the only DJDE information in that record. Each left/right part within a record is separated from the next left/right part by a comma.

- The end of all DJDE left/right parts in a record is coded by either of the following:

  ,; (comma, semi-colon)
  ;b (semi-colon, blank)

- The end of the right part of a DJDE record which is split and continued on the next DJDE record is shown by:

  ,; (comma, semi-colon)

- The prefix in the DJDE record may appear after the DJDE left/right part(s) as long as it is consistent in location for all DJDE records.

- If the file containing the DJDE is variable-blocked, the program which blocks the file may strip off trailing blanks. Thus, if the comment record contains no actual comments, the blank following the "C" may be eliminated. If the "C" is the last character of the record, the command record will be accepted. However, any character other than a blank following the "C" will cause the record to be processed as a legitimate DJDE record, and not as a comment. See "C" command in Table 6-3.

- If a DJDE is always created because of coding procedures, but in fact there is no need to set any of the DJDE left/right parts for a particular report, a null DJDE may be created with only an "END;" statement and no other commands specified.

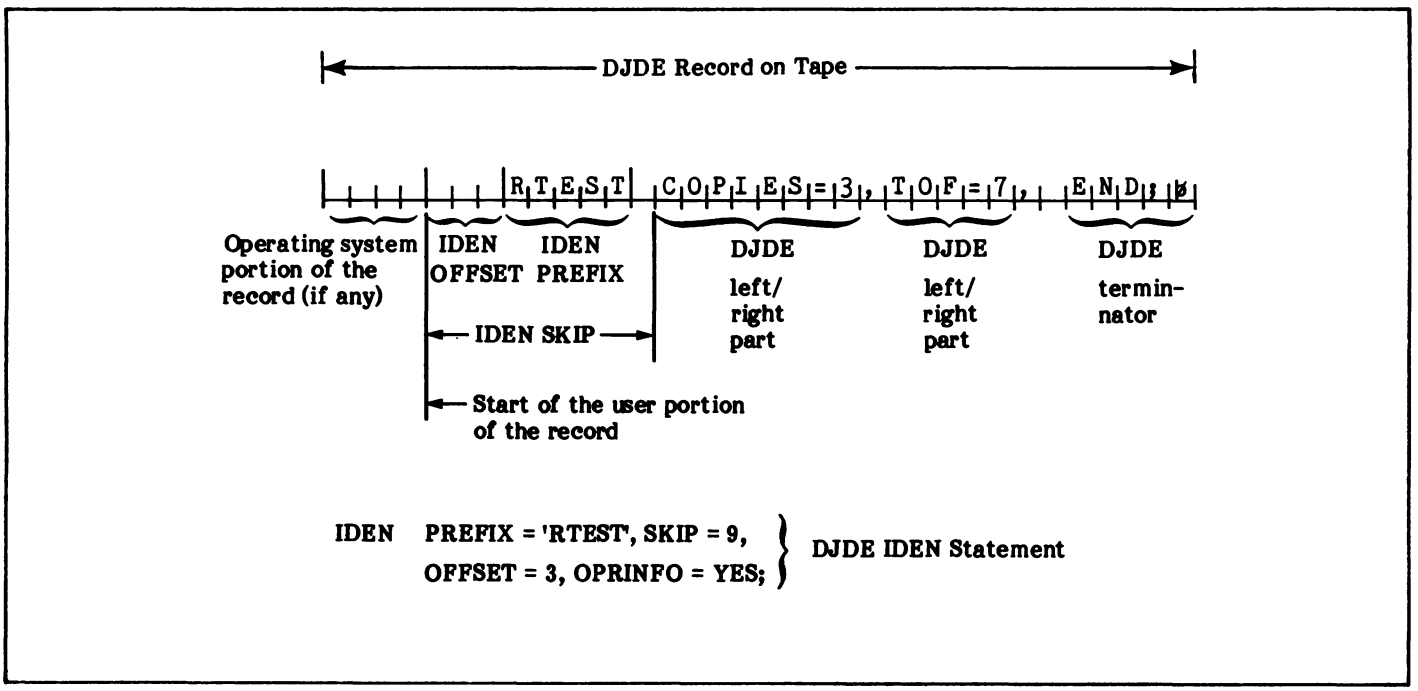



Figure 6-2. Single DJDE Record and IDEN Statement

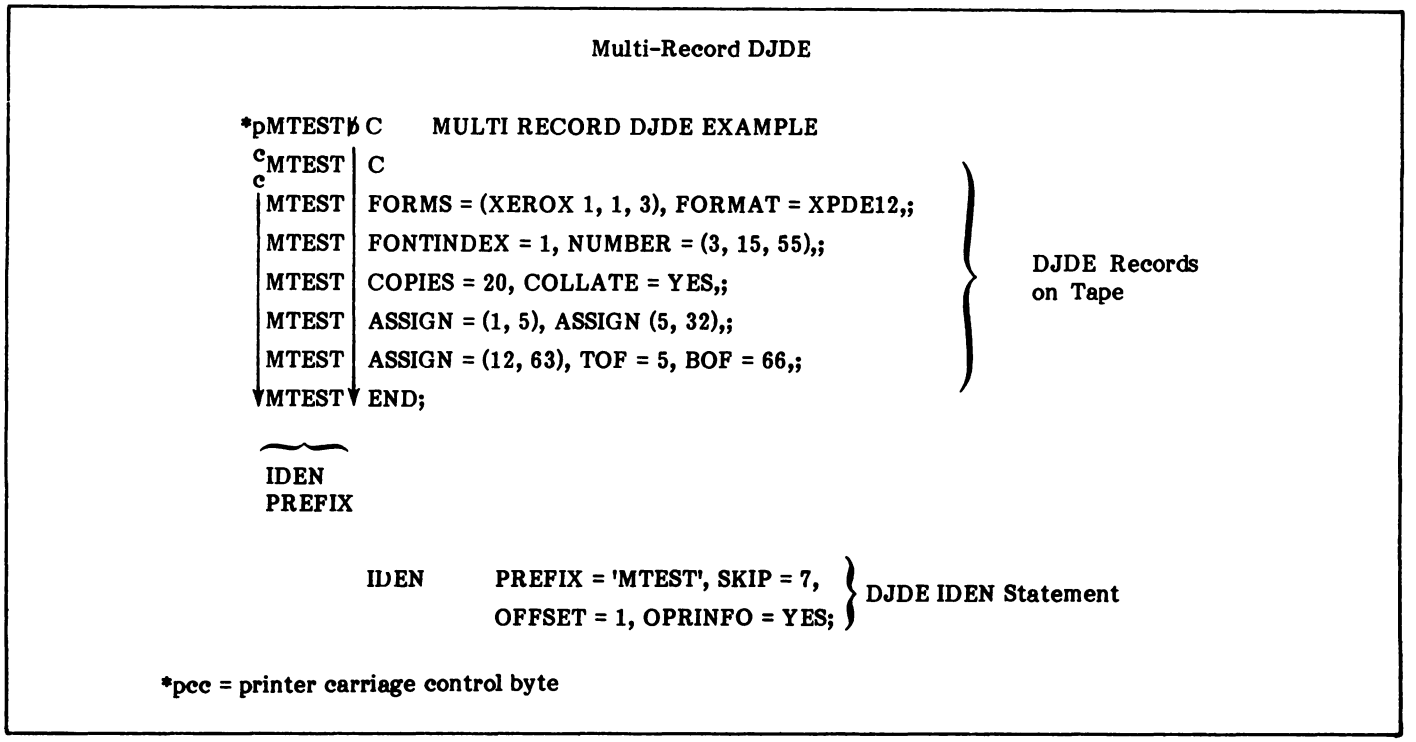



Figure 6-3. Multiple Record DJDE

### DJDE Output Report Changes

The changes to the Job Descriptor Entry, specified in the DJDE, are incorporated when "END;" is encountered. The changes begin on the next record or page following the last DJDE record. Report related parameters take effect before the first data record in the report.

The specification OPRINFO=YES in the job descriptor entry ensures that the DJDE records are printed and sent to the bin at the next page transition after an "END;" DJDE command has been processed. The DJDE records are printed on a separate page from the report data. Comments in the DJDE may be used for operator notification or output routing instructions in conjunction with the OPRINFO option.

DJDE records will be printed and delivered to the bin regardless of the OPRINFO option if the DJDE contains an error. DJDE records may also be printed along with delimiter records as described in section "Stacked Reports" of Chapter 4.

## Command File DJDEs

A DJDE command file is a disk file of DJDE commands (described in Table 6-3) which are incorporated into a user selected JDE when a print job is initiated. The command file is initiated by the operator as a parameter on the START command (START *filename) and will only modify printing envionment parameters at job startup time.

### DJDE Command File Creation

The DJDE file can be created via the EDITOR (described in Chapter 8) on the 9700 or written to tape on a host system and then entered in the 9700 file system. If the DJDE records are to be created on tape they must be in a fixed, unblocked format with no carriage control and a maximum record length of 133 characters. Only the first 72 characters of a record will be used for command information.

A DJDE command file must be cataloged in the CMD file directory before it can be initiated. A file created using the EDITOR can be put into the CMD file directory by specifying the CMD "file-type" on the SAVE command. A file on tape can be copied into the CMD directory by specifying CMD as the outfile "type" (see "Copying a File" in Chapter 7).

Each record of a DJDE command file will contain a series of left/right parts which describe the Job Descriptor Entry changes to be applied to the print job. Each left/right part within a record is separated by commas and each record is terminated by ",;". These left/right parts are defined in Table 6-3, and are the same as those available for DJDEs on the input tape. A command file DJDE does not require an IDEN statement in the JDE (nor an IDEN field on the command record) as does an input tape DJDE.

Examples of DJDE command files are illustrated in Figures 6-4 and 6-5. Figure 6-4 illustrates a simple DJDE command file. Each record of the file begins in the first byte and records are terminated by a ",;". To use this file it must be cataloged in the CMD file directory under a user specified name.

Figure 6-5 illustrates a feature of the 9700 DJDE commands that allows the JDL and JDE to be referenced within the file along with the specification of other DJDE commands. It is possible with this type of command file to have all the information within it necessary to start a job. When the operator initiates the print job he will only have to name this file on the START command. No other parameters need be entered. In this example the Job Descriptor Library "HON26" would exist on disk cataloged in the JDL file directory.

```
C
C     SAMPLE DJDE COMMAND FILE
C
COPIES = 5,COLLATE = YES,;
FORMS = FRMX,FORMAT = PDEX,;
FONTINDEX = 1,;
DATA = (2,132),;
END;
```

Figure 6-4. Sample DJDE Command File

```
C     DJDE COMMANDS WITH REFERENCE
C     TO JDL FILE ON DISK
C
JDL = HON26, JDE = H2,;
FORMS = CHART1,;
COPIES = 2,;
END;
```

Figure 6-5. JDL/JDE Specified In DJDE Command File

### Initiating a Command File DJDE

When a print job is to be run, the DJDE file is activated by the operator via the START command on the keyboard display. The syntax and explanation of the START command parameters are discussed in Chapter 7 under "Starting A Job". The command file is referenced on the START command by the "*filename" option which is the first parameter that can be specified. The asterisk ("*") preceding the file name denotes it as a DJDE command file.

Example 1:

```
START *MODSYS,JDE10,JDLDOS          (Operator Key-in)
```

where

| | |
|---|---|
| MODSYS | is a file of DJDE commands in the CMD file directory. |
| JDE10 | is the identifier of the Job Descriptor Entry which is part of the Job Descriptor Library "JDLDOS". |
| JDLDOS | is the file name of the Job Descriptor Library in the JDL file directory. |

Example 2:

```
START *HONWEL       (Operator Key-in)
```

where

| | |
|---|---|
| HONWEL | is a file name (in the CMD directory) which consists of DJDE commands. |

No other parameters are necessary because the file "HONWEL" contains DJDE commands which request a specific JDL and JDE as in Figure 6-5.

Table 6-3. DJDE Left-Right Parts

| Left Part | Right Parts | Description |
|---|---|---|
| {JDL / SYSTEM} = | jdl-id | The right part "jdl-id" specifies the name of the Job Description Library to be invoked at the next report boundary. It must exist on disk in the JDL directory. |
| {JDE / JOB} = | jde-id | The right part "jde-id" specifies the Job Descriptor Entry to be used within the selected Job Descriptor Library at the next report boundary. |
| COPIES = | value | Specifies the number of copies of a report to produce at the next page boundary.<br><br>The right part "value" is the same as defined in OUTPUT COPIES command of Chapter 5. |
| FORMS = | {form-id / (form-id,init [,copies]) / NONE} | Specifies the form to be merged on the printed pages. It takes effect on the next page boundary. The form specified will be invoked beginning with the first page of the specified starting copy number. Multiple FORMS commands may be used to associate different forms with different copy plies.<br><br>FORMS right parts are the same as defined in OUTPUT FORMS command of Chapter 5. |
| COLLATE = | {YES / NO} | Specifies whether collated or uncollated mode is to be used. It takes effect at the next report boundary.<br><br>The right part "YES" specifies that report copies are to be printed in collated mode.<br><br>"NO" specifies that report copies are to be printed in uncollated mode. |
| MODIFY = | {cme-id / (cme-id,init [,copies]) / NONE} | Specifies the Copy Modification (CME) to be used in report processing. It takes effect at the next page boundary.<br><br>"cme-id" refers to a file which is separately cataloged in the CME library. See section "Copy Modification Features" in Chapter 5 for further details.<br><br>The right parts of the DJDE MODIFY command are defined in the OUTPUT MODIFY description of Chapter 5. |
| FORMAT = | pde-id | Specifies the Page Descriptor Entry (PDE) to be used for formatting control. It takes effect on the next page boundary.<br><br>"pde-id" refers to a separately cataloged file in the PDE library on disk. Details of the PDE command and PDE files are discussed in Chapter 5. |

Table 6-3. DJDE Left-Right Parts (cont.)

| Left Part | Right Parts | Description |
|---|---|---|
| NUMBER = | { (pnum,lnum,cnum) / NO } | This command specifies page numbering control. It takes effect on the next page boundary.<br>The right parts are the same as those for OUTPUT NUMBER command as defined in Chapter 5. |
| MARGIN = | { value / (value, { IN / CM / POS }) } | This command specifies the left printing margin within each logical page. It takes effect at the next record following a DJDE "END" command. The right part "value" is the same as defined in the LINE MARGIN command of Chapter 5. |
| DATA = | (pdo, length) | Specifies the location and length of printable data within an input record. It takes effect at the next record following a DJDE "END" command. The right part parameters are defined in the LINE DATA command in Chapter 5. |
| OVERPRINT = | ( { PRINT / MERGE / IGNORE } { DISP / NODISP } ) | Command for overprint control. It takes effect an the next record following a DJDE "END" command. The right part parameters are the same as for the LINE OVERPRINT command in Chapter 5. |
| ASSIGN = | { (channo,lineno) / (channo,(lineno,...,lineno)) } | This command specifies an assignment of a VFU channel number to a page line number. It is possible to have multiple ASSIGN commands within a DJDE. They take effect at the next record following a DJDE "END" command. The right part parameters are defined the same as for the VFU ASSIGN command in Chapter 5. |
| TOF = | value | This command specifies the "top-of-form" line number. It takes effect at the next record following a DJDE "END" command. Command is defined same as VFU TOF in Chapter 5. |
| BOF = | value | This command specifies the "bottom-of-form" line number. It takes effect at the next record following a DJDE "END" command. Command is defined same as VFU BOF in Chapter 5. |
| FONTINDEX = | { offset / NONE } | This command allows the user to specify in a user record a specific font to be used. It takes effect on the next report boundary. The right part parameters are same as LINE FONTINDEX command in Chapter 5. |

Table 6-3. DJDE Left-Right Parts (cont.)

| Left Part | Right Parts | Description |
|---|---|---|
| ITEXT =<br><br>OTEXT =<br><br>RTEXT =<br><br>RFORM = | {sc<br>(sc, passnum)<br>NONE}<br><br>{sc<br>(sc [,{passnum<br>END}][,WAIT])<br>NONE}<br><br>{sc<br>(sc ,passnum [,line [,col]])<br>NONE}<br><br>{form-id<br>NONE} | The ITEXT, OTEXT, RTEXT, and RFORM commands have the same syntax and parameter definitions as defined in Chapter 5. These commands become effective at the next report boundary. |
| C$\phi$ | sc | The "C" command allows for comment text to be included in the DJDE record. The "C" must be followed by a space and not by an equal sign. All text up to a semicolon or end-of-record will be treated as commentary. |
| END;$\phi$ | | This command is used to indicate the end of DJDE information. When END command is encountered, the system will apply all DJDE information specified to the current printing environment at the next report, page or record boundary.<br><br>"END" must terminate with a semicolon/blank character pair. |

# 7. SYSTEM OPERATIONS

## Introduction

Control is exercised over the 9700 system through the use of the keyboard/display unit. It is the primary means of communication between the operator and the Operating System Software. Through the keyboard/display, the operator can control the loading of the operating system; switch between on-line and off-line processing modes; initiate jobs; build forms; modify source files; request sample prints of data, forms, fonts, and logos; catalog forms, fonts, logos, JDLs, PDEs and CMEs; and obtain accounting summaries and other system outputs. While print jobs are running, messages are provided to the operator concerning the status of the jobs, and if necessary, of unexpected conditions.

A summary of commands available to the operator is shown in Table 7-1. Commands common to both on-line and off-line processing are discussed in sections of this chapter, while commands applicable only to on-line processing are presented in Chapter 12. Further information is available in the 9700 Operators Guide (Xerox Publication No. 600P81096).

Table 7-1. System Command Summary

| Command | ON | OFF | Function | Res* | Reference Page |
|---|---|---|---|---|---|
| ABORT [R] | X | X | Abort current job (report) or discontinue system activities | * | 7-14 |
| ACCOUNT | X | X | Accounting file maintenance | 1 | 7-18 |
| ALIGN | X | X | Align laser image with paper | 1 | 7-6 |
| CONTINUE | X | X | Resume input/output activities | * | 7-13 |
| COPY | X | X | Copy files from disk/tape to disk/tape | 1 | 7-24 |
| DELETE | X | X | Delete files from system disk | 1 | 7-23 |
| ENDJOB | X | | Print all remaining unimaged pages | 1 | 12-8 |
| FEED | X | X | Select active paper tray | 1 | 7-21 |
| FILE | X | X | List file directories on keyboard/display | 1 | 7-23 |
| FONTS | X | X | Set maximum number of active fonts per print job | 1 | 7-10 |
| FORMS | X | X | Set maximum number of active forms per print job | 1 | 7-10 |
| JOBS | X | X | Display job status information | * | 7-15 |
| LIST | X | X | Hard copy listing of file directories | 1 | 7-23 |
| LOGON | X | X | Create access privileges | 1 | 7-7 |
| MOVE | | X | Position tape by blocks or files | 1 | 7-20 |
| OFFLINE | X | | Set 9700 to off-line | 1 | 12-6 |
| ONLINE | X | | Set 9700 on-line and NOT READY | 1 | 12-8 |
| RCU | X | X | Restrict command usage | * | 7-7 |
| REA | X | X | Reallocate print file | 2 | 7-9 |
| REPORT | X | X | Report of system activity and accounting | 1 | 7-17 |
| RESET | X | X | Forces all system activities to cease | 1 | 7-14 |
| REWIND | | X | Rewind tape | 2 | 7-20 |
| SAMPLE | X | X | Sample and test pattern print | 1 | 7-16 |
| SELECT | X | X | Select active output bin | 1 | 7-22 |
| SPACE | | X | Position tape by logical reports or pages | 1 | 7-19 |
| START | X | X | Initiate print job | * | 7-11 |
| STOP | X | X | Suspend input/output activities | * | 7-13 |
| TIME | X | X | Display time of day, date on console | 1 | 7-15 |
| UNLOAD | X | X | Lower inactive paper bin | * | 7-22 |
| task-name initiation | X | X | Initiate a task | 1 | 7-18 |

* The "Res" column defines the system default operator restriction level associated with each command. See RCU command for further details. An "X" in column ON (on-line) or OFF (off-line) means the command is available in that mode.

## Key-in Conventions

The following conventions are to be observed when communicating with the 9700 via the keyboard/display unit.

- The system is always ready to accept input
- All commands may be abbreviated to the first three characters
- Commands are terminated by striking RETURN key except those entered via the function keys (described below)
- Error and information messages from the system will be preceded by "OS" and followed by a four digit code (the severity level). The complete set of system messages is contained in Appendix B.
- A message acknowledging an operator's request will be displayed as positive feedback to the operator. Within this manual these messages are underlined in examples to differentiate them from user keyins
- The error message "OS2710 INVALID COMMAND RE-ENTER" is displayed when an erroneous command or keyword within a command is entered
- There are six special keys on the keyboard/display unit called function keys. They need only be pressed to invoke a specified action (i.e., RETURN key is not pressed). These function keys are:

| | |
|---|---|
| (CONTINUE) | Using this function key is like entering the CONTINUE command. (See "Continuation After Stopping Printing") |
| (DELETE) | Pushing this key while holding down the control key (CTRL) will erase the entire current line from the screen. |
| (RUBOUT) | This key is used to erase characters. Each time it is pushed, it erases the previous character on the current line and backspaces one position. |
| (SAMPLE COPY) | Pushing this function key produces a sample page in the Sample Print Tray during printing, just like the SAMPLE command does. (See "Sample Print") |
| (STOP) | Pushing this key is the same as entering the STOP command. (See "Stopping the System") |
| (STATUS) | This key can be used instead of the JOBS command to get jobs status. (See "Status Information") |

## System Start-Up

The operator must turn the main POWER ON switch on the right side of the Control Module to power up the 9700. The POWER and WARM lights will light and the system will begin its warm up cycle (10-15 minutes). When the warm up cycle is completed the WARM light will go out and the READY light will come on. On the keyboard/display the following message will be displayed:

READY
$

Once the prompt symbol $ has been displayed, the operator can boot the system from disk by keying in the appropriate command on the keyboard. The boot command has the following form:

B (RETURN)

After the boot command has been entered, diagnostics will be run to test main memory. Any errors discovered will be reported. If they are beyond the capability of the operator to repair, a Xerox field engineer must be called. If there are no errors, the system is booted and it responds with the following messages:

XEROX 9700
ELECTRONIC PRINTING SYSTEM
VERSION x REVISION y
(The REVISION level indicates the patch level of the system.
It will only be displayed after patches have been applied.)

ENTER DATE (MM/DD/YY): (Operator enters date)
ENTER TIME (HH:MM/SS): (Operator enters time)
(System will print time and date)
ARE DATE AND TIME CORRECT AS DISPLAYED (Y/N)?

If N is entered the above time and date requests will be made again. After a Y is entered, the system will perform checks to determine the state of the system. The main and font memories are checked, and tests are made to verify that the required system devices are present. If problems exist, one or more of the following messages will be displayed on the operators console. See "System Verification Messages" in the following section for an explanation of these messages.

OPERATING IN DEGRADED MODE:
DEVICES NOT ACCESSIBLE
MAIN MEMORY MISSING: BANK(S) xx,xy,xz, . . .
FONT MEMORY MISSING: BANK(S) xx, xy, xz, . . .
NO FONT MEMORY AVAILABLE.
FONT MEMORY NOT CHECKED.
MISSING MEMORY PARITY CONTROL BOARD(S): x, y, . . .
SYSTEM NOT OPERATIONAL.

After the verification checks are completed OSS is ready to accept operator commands.

### System Verification Messages

Messages will appear on the operator's console if problems occur while OSS is performing its system verification tests. The message and their explanations are as follows:

OPERATING IN DEGRADED MODE Displayed if any problem was detected. Other messages will follow.

DEVICES NOT ACCESSIBLE Displayed if any of the system devices appears not to be in the system. The device(s) which are not accessible will be displayed on the operator's console after this message.

MAIN MEMORY MISSING: BANK(S) xx, xy, xz, . . . Displayed if any memory specified during the system configuration phase is either not present or is suffering data loss. The banks are displayed in 4K sections.

FONT MEMORY MISSING: BANK(S) xx, xy, xz, . . . Displayed if font memory is missing or suffering data loss. The banks are displayed in 4K sections.

NO FONT MEMORY AVAILABLE Displayed in lieu of the previous message if all banks of font memory appear to be failing the tests.

FONT MEMORY NOT CHECKED Displayed if there was insufficient main memory to run the font memory tests. The MAIN MEMORY MISSING message indicates which banks are missing.

MISSING MEMORY PARITY CONTROL BOARD(S): x, y Displayed if either of the parity controllers is missing or non-operational.

SYSTEM NOT OPERATIONAL Displayed if devices were missing, main memory below 32K is missing, or there is no font memory available. The system will probably be unable to function in even a degraded mode if this message is displayed.

## Setting Physical Page Alignment

ALIGN

The ALIGN command is used to align the laser image with the paper. Every 9700 machine will require slightly different alignment values, but once determined, these values remain relatively constant for a given machine. Alignment values entered will be preserved by the system until changed. The command form is:

ALIGN [lines][,dots]

where

lines changes line positioning.

dots changes dot positioning.

If no operands are specified, then the current alignment values are displayed.

The premissible changes for scan line position are such that the line position must be greater than 150 and less than 260. The dot position may be greater than or equal to 0 and less than 796. If the machine setup is such that the correction would be greater than the premissible value, the machine must be mechanically adjusted to bring the ALI command into functional range.

An alignment form is provided with OSS systems to insure proper alignment of output. The file name of this form is "TEST" and it resides in the FSL directory (see Appendix D). It is recommended that sites compile and use this form prior to any other runs. The compiled form should also be sampled prior to testing any forms. When using preprinted forms, prior adjustment of the machine by use of the alignment form should be considered mandatory. It should also be used whenever any mechanical adjustments are made to the printer.

A simple way to use the alignment form is to sample the form, and then fold it in half - once on the dot orientation and once on the scan line orientation. This provides you with an easy measurement of how many dots and/or lines to add/subtract. After correcting the alignment with the ALIGN command, resample the form and check for any further variation that may be needed. For all but the most critical applications, it should be sufficient to get the alignment to the point where the center line is visible on each side of the fold.

Examples:

```
OS1000 READY FOR COMMANDS
ALIGN  151,252
ALIGNMENT IS 151 SCAN LINES AND 252 DOTS
ALIGN
ALIGNMENT IS 151 SCAN LINES AND 252 DOTS
ALIGN  ,160
ALIGNMENT IS 151 SCAN LINES AND 160 DOTS
```

## File Access and Command Protection

### Access Protection

LOGON

The LOGON command provides an installation with a way to authorize/restrict access to files of a given type (i.e., those cataloged in a certain file directory) for a particular action (record editing, deleting and copying of files). It also provides an installation with a way to authorize/restrict certain system level commands (see RCU command). Three user "classes" are provided each with varying degrees of access to files (as illustrated in Figure 7-2) and operator command usage. Passwords are required with the use of the two "higher" levels of user's classification.

The command form is:

LOGON [{1 | 2,password | 3,password}]

where

1,2, or 3 specifies the user's classification (1 = lowest, 3 = highest). Table 7-2 illustrates how this classification is related to the file access functions that can be performed.

password is a 1-15 character string that may consist of any characters (including leading and trailing blanks) on the keyboard. Passwords are provided with a 9700 system which may be modified at each installation.

When on class level 2 or 3, the password may be displayed by typing LOGON with no parameters. After displaying the password, the user also has the opportunity to change it. The initial password for level 2 after building a new system is null (i.e., logon the system with LOGON 2, (RETURN)).

### Restrict Command Usage

RCU

The RCU command provides an installation with a way to authorize/restrict the usage of system commands. The system commands (listed in Table 7-1) can be authorized/restricted as to who is able to use them by relating them to the "classes" previously defined in the LOGON command. In order to use a specific system command, the operator must be logged on to a "class" level equal to or greater than the user defined (or default) restirction level. The default restriction level for each command is defined in Table 7-1 (see "Res" column). The command form is:

RCU command [,class]

where

command is the system command to be authorized or restricted. There are certain commands that cannot be restricted as indicated in Table 7-1 (those with an * in the "Res" column).

class is the lowest class (LOGON class) allowed to use the command. If class is not specified the current class level for that command is displayed.

A user cannot restrict the use of a command to a level higher than the current LOGON class. A user also cannot change the class level of a command unless authorized to use that command.

Table 7-2. File Access Classes

| FUNCTION | RECORD EDITING | | | FILE DELETE | | | FILE COPY | | |
|---|---|---|---|---|---|---|---|---|---|
| TYPE ↓ CLASS → | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 |
| CMD | N | Y | Y | N | Y | Y | Y | Y | Y |
| CME | N | N | N | N | Y | Y | Y | Y | Y |
| FNT | N | N | N | N | Y | Y | Y | Y | Y |
| FRM | N | N | N | N | Y | Y | Y | Y | Y |
| FSL | N | Y | Y | N | Y | Y | Y | Y | Y |
| JDL | N | N | N | N | Y | Y | Y | Y | Y |
| JSL | N | Y | Y | N | Y | Y | Y | Y | Y |
| LGO | N | N | N | N | Y | Y | Y | Y | Y |
| PDE | N | N | N | N | Y | Y | Y | Y | Y |
| SYS | N | N | Y | N | N | Y | N | N | Y |
| TMP | N | Y | Y | N | Y | Y | N | Y | Y |
| TSK | N | N | N | N | N | N | N | N | Y |

TYPE — is the name of the file directory in which files are cataloged. Each file directory will contain a specific type of file (i.e., type CME contains files with DJDE commands. The directory names are defined in section "File Manipulation" of this chapter.

CLASS — is the required LOGON class to perform a particular function.

{N / Y} — means NO or YES as to whether a particular function (record editing, deleting, copying) may be performed.

RECORD EDITING — means that records in a file may (or may not) be edited. If access mode is N then a GET (EDITOR command) will not be accepted nor any record editing commands.

FILE DELETE (or COPY) — means a file may (or may not) be deleted (or copied) either with an EDITOR command or with the OSS level DELETE (COPY) command.

## Reallocate Print File

REA

The REA command allows an installation to set the size of the system print file on disk. A default print file area is created during sysgen but it is generally necessary to increase its size based on an installation's requirements. The print file size should be reallocated after a new OSS system has been created, but also may be increased/decreased in size at any time.

The REA command has no parameters. After it is invoked several messages appear on the operator's console as follows:

REA
(This invokes the Reallocation processor, which will determine current space availability for the print file.)

REA VERSION A00

DO NOT ABORT OR OTHERWISE DISTURB THIS PROCESS
(The warning means exactly what it says. Interrupting the reallocation process may make sections of the disk be marked as busy and thus unusable. The processor will be busy examining the disk directory and bit map.)

NUMBER OF CYLINDERS IN OLD PRINT FILE IS aaa (bbbb sectors)

TOTAL NUMBER OF SECTORS AVAILABLE IS xxxxx
(The value shown in the total free space available on the disk and is given in sectors.)

NUMBER OF CYLINDERS AVAILABLE FOR PRINT FILE IS cccc (dddd sectors)
(The value shown is the largest contiguous free space on the disk presented in cylinders (and sectors).

HOW MANY CYLINDERS SHOULD THE PRINT FILE CONTAIN? zzz (RETURN)
(At this point, a value of at least 250 should be entered. The value must be equal to or less than the available cylinder value. A value of more than 250 would be worthwhile in the event that the system frequently has input catching up with output. Otherwise, the 250 value sho ld be sufficient.)

RESUMING NORMAL OPERATIONS
(At this point the Reallocation processor has completed its function and the system is ready for normal operation.)

OS1000 READY FOR COMMANDS

## Establishing Print Job Characteristics

To optimize the allocation of controller memory, an installation may specify (with the FORMS and FONTS commands) parameters which define the maximum number of fonts and forms the system is to attempt to keep resident in memory during a print job.

### Fonts

FONTS

The maximum number of active fonts is specified by the FONTS command. The command form is:

FONTS number

where

number is a positive non-zero integer which specifies the maximum number of active fonts per print job. No single page to be printed may invoke more fonts than specified. This number must include all fonts used on the forms as well as variable data. The forms font, if a form is used, must be included in the derivation of "number". A larger "number" may be specified than will ever be used on a single page, thus allowing font changes on a page-to-page basis without throughput degradation if all the fonts can be loaded into memory. If "number" is omitted, the system displays the value currently in effect.

The FONTS command remains in effect until the FONTS command is reissued or a new OSS system is sysgened.

During the course of a print job, it is possible to use more than "number" of fonts as long as no more than "number" of fonts is called for on a single page. If, during a print job, the current FONTS value is exceeded while processing a page (i.e., the total of fonts required for form printing plus those required for variable data printing), that job is aborted and the following message "OS2880 MAXIMUM NUMBER OF FONTS ALLOWED EXCEEDED. ENTER FONTS COMMAND" is displayed on the operator's console.

### Forms

FORMS

The maximum number of active forms is specified by the FORMS command. The command form is:

FORMS number

where

number is a positive non-zero integer that specifies the maximum number of active forms per print job. Specifying a "number" larger than 1 allows form changes on a page-to-page basis without throughput degradation once the forms have been loaded into memory. If "number" is omitted, the system responds with the value currently in effect.

The FORMS command remains in effect until the FORMS command is reissued or a new OSS system is sysgened.

## Job Control

The operating system provides commands which enable the operator to control the flow of print jobs (starting, stopping, continuing, aborting) through the printing system.

### Starting a Job

START

The off-line START command is used to initiate the printing of a user's print tape. To initiate an on-line job see section "Starting an On-Line Job" in Chapter 12. The command form is:

START [*filename,] [[jde] [,[jdl] [,{S / M} [,copies [,REPORTS:r1,r2 ,...]]]]]

Parameters of the START command are positional and must be separated by commas. With the exception of the *filename option, a comma must be entered to replace a parameter that is not specified. Options specified on the START command will override those specified in the job descriptor library (as will options specified in the DJDE command file). Further details on override parameters are contained in section "Hierarchy of Replacement" in Chapter 3. Examples of START command usage are provided after the parameters are defined. Options on the START command are as follows:

filename — is a 1-6 character identifier for a file on disk consisting of DJDE commands. It must be cataloged in the CMD directory. These DJDE files are discussed in Chapter 6 under "DJDE Command Files". Note that filename is preceded by an "*".

jde — is a 1-6 character identifier for the Job Descriptor Entry to be used in processing the job. If the "jde" option is not specified on the START command, then the user specified default jde (DFLT) within the jdl will be used (see "Default JDE" in Chapter 3).

jdl — is a 1-6 character identifier of a disk file which contains the Job Descriptor Library for the print job. It must be catalogued in the JDL file directory. If the "jdl" option is not specified then the default is used. The default "jdl" is user created with an identifier of DFAULT on the SYSTEM command (see SYSTEM Command Set in Chapter 3).

{S / M} — are parameters used to control the processing mode of a job. The two modes are: Multi-report (M) and Single-report (S). The default mode is multi-report. Single-report mode halts the system after the processing of each report to allow the operator to select the job set-up parameters for the next report. Multi-report mode allows all reports in all files to be processed continuously. Processing will automatically sequence from report to report, file to file, volume to volume until all reports have been processed.

copies — specifies the number of copies of a report to be printed. It is an override of the "copies" value specified in the job descriptor entry, overrides a value for "copies" (if specified) in a DJDE command file, and also will override DJDE values on the job print tape.

REPORTS:r1,r2...

The keyword REPORTS allows the user to specify the sequence and set of reports to be processed. For "r1,r2..." the user specifies numeric values or ranges of values, representing the order of reports to be printed. A range is specified as n -m , where n and m are the first and last reports in the range to be processed, respectively. For example, entering "REPORTS:6,1-3,5,4" would cause the sixth report to be printed first, followed by the first through third, followed by the fifth and then the fourth. The REPORTS option is also useful if only one report is to be printed of a multiple report tape. This saves the step of spacing (see SPACE command) over reports not needed. A maximum of 14 values (or ranges of values) is allowed.

Example 1:

```
START J12,H2SYS
```

This command starts a print job using the H2SYS job descriptor library and the job descriptor entry, J12. It will run in multi-report mode (by default) and print the number of copies as specified in the J12 job descriptor entry. The job descriptor library, H2SYS, must reside in object form in the JDL directory. After the START command is initiated, several messages will be displayed on the keyboard/display to inform the operator of the print jobs in progress. In most instances, only one more keyin will be required of the operator. An example of the interaction is as follows:

```
OS1000 READY FOR COMMANDS
START J12,H2SYS
OS1010 STARTING JOB 00003
OS2010 MOUNT INPUT TAPE; "CONTINUE I" WHEN READY
CONTINUE I
OS0010 RESUMING INPUT
OS0020 RESUMING OUTPUT
OS1020 JOB 00003 HAS COMPLETED INPUT PHASE
OS1030 JOB 00003 HAS COMPLETED PRINTING
OS1000 READY FOR COMMANDS
```

Example 2:

```
START J12,H2SYS,,5
```

This command is the same as in Example 1 with the exception that five copies are requested. The value of 5 entered for copies will override the value specified in the J12 job descriptor entry. Note that a comma replaces the unspecified mode option.

Example 3:

```
START
```

No options are specified so the START command defaults take effect. The default for the job descriptor library will be DFAULT which must exist in the JDL directory. The job descriptor entry to be used will be DFLT (created by the user within the DFAULT job descriptor library).

Example 4:

```
START *UNDJDE,,UNJOB,S,4
```

In this example, the DJDE command file UNDJDE (cataloged in CMD directory) is initiated along with the job descriptor library UNJOB (cataloged in the JDL directory). Since no job descriptor entry is named on the START command, the system will default to the "DFLT" jde which the user has created within the UNJOB job descriptor library. The job will be run in single report mode and 4 copies will be printed.

### Stopping the System

STOP

The STOP command is used to suspend tape input and/or printer output activities. Since at any given time, the system may be processing input for one job and output for another job, input and output may be controlled separately. Stopping input suspends tape processing but allows printing to continue. Stopping output suspends printing after cycling out the paper path but allows processing of tape input to continue. The command form is:

STOP$\left\{\begin{matrix}I\\O\end{matrix}\right\}$ or (STOP) function key

where

I requests that input activities be suspended until resumed by CONTINUE command.
O requests that output activities be suspended until resumed by a CONTINUE command.

If no operand is specified or the (STOP) function key is used, both input and output activities are suspended.

### Continuation after Stopping Printing

CONTINUE

The CONTINUE command is used to resume tape input and/or printer output activities suspended by the STOP command or soft-stop mid job errors. As with the STOP command, input and output activities may be controlled separately. Continuing input allows tape input to be spooled to disk even if printer output is stopped. Continuing output allows an existing page to be printed even if input is stopped. The command form is:

CONTINUE $\left\{\begin{matrix}I\\O\end{matrix}\right\}$ or (CONTINUE) function key

where

I requests that input activities be resumed
O requests that output activities be resumed

If no operand is specified or the (CONTINUE) function key is used, both input and output activities are resumed.

### Continuation After MOVE or SPACE

CONTINUE

After a MOVE or SPACE command has been issued the operator has the option (after the operation completes) of entering a CONTINUE command (as described above) or one of the form:

CONTINUE jde, jdl

where jde and jdl are defined the same as on the START command. This form of the CONTINUE command is to be used when the jdl and jde parameters to be used for the next report to be processed from tape are different than those issued on the last START command.

### Starting a Task

Certain tasks supplied with the system are invoked by task name. These tasks can be run only when the system is in the idle state, i.e., when the print job queue is empty. Currently included in this category are the EDITOR, FDL, PDL and OSDS. The command form is:

taskname parm(1),parm(2),...

where

taskname is a 1-6 character identifier for the task

parm(i) are task specific parameters supplied with the command

Tasks invoked by name may be aborted via the ABORT command. The task name need not be given since no other jobs or tasks can be active.

### Aborting a Job

ABORT

The ABORT command removes a job from the system, aborts a report or discontinues certain system activities. A job abort is handled in this manner: if the job is in the input phase, tape processing ceases with the next tape read; if the job is queued for printing, it is removed from the queue; and if the job is printing, it is truncated and the system continues with any jobs remaining in the print queue. Other system activities may also be aborted. These include the language processors (PDL and FDL), tasks evoked by name, and other system level commands. The command form is:

ABORT {job-id / R}

where

job-id is the job identification assigned to a job by the system in response to the START command. If no job-id is specified, any non-print task is aborted; or, if printing, the job currently being printed is aborted.

R requests that the report currently being printed be aborted.

Examples:

```
ABORT 5
OS0900 JOB 5 ABORTED

ABORT
OS0950 TASK ABORTED
```

## Resetting the System

RESET

The RESET command forces all system processing activity to cease. All print jobs are removed from the job queue and any pages in the printer paper path are cycled to the stacker. If the input device is tape, it is left positioned at the point at which it was when the RESET command was given. The system remains in an idle state until the next command. The command form is:

RESET

## Status Information

### Job Queue Status

JOBS

The JOBS command is used to display system and job status information on the operator's console. System status shows the state of the system (running, idle, stopped), and job status shows information about all jobs known to the system. The command form is:

JOBS or (STATUS) function key

In response to this command, the system outputs a formatted display:

system-status

| JOB-ID | JDE-USED | STATUS | COPIES | PAGES | MODE |
|---|---|---|---|---|---|
| id | jde | s | c | p | m |

where

system-status is one of the system states

| | |
|---|---|
| RUNNING | if either input or output processing is active |
| IDLE | if the print job queue is empty |
| STOPPED | if both input and output have been suspended |

id is the job identification supplied by the system in response to a START command

jde is the JDE name supplied by the operator to start the job

s is the job status which may be

| | |
|---|---|
| QUEUED | if the job is waiting to be processed |
| INPUT | if the job is in the input phase and has not begun to print |
| PRINTING | if the job is being printed |
| ABORT | if the job is to be aborted |

c is the total copies to be printed if job status is INPUT. If job status is PRINTING, c is the remaining copies to be printed (if mode is COLLATE) or total number of copies to be printed (if mode is NON-COLLATE). If job status is QUEUED, c is either the copies value specified by operator on START command or blank for default value specified by the JDE.

p is the number of pages remaining to be printed for the current copy (if the mode is COLLATE). The value will always be zero if on the first pass. If the mode is NON-COLLATE the value will be the remaining number of copies to be printed of the current page.

m is either C (COLLATE mode) or NC (NON-COLLATE mode).

### Time and Date

TIME

The time and date as kept within the system can be obtained with the TIME command. The command form is:

TIME

which will result in the date and time displayed on the keyboard/display.

## Sample Print

The operator may request a sample print either during job printing or when the system is in the idle state. In the former case, the next page to be printed will be printed twice, with one copy being delivered to the output bin normally and the other being delivered to the sample print tray. In the case where the system is in the idle state at the time the request is given, the operator must indicate the item to be sample-printed (i.e., the operator may request a sample print of a font, form, or a logo).

### Single Sample Print

SAMPLE

To request a sample print during job printing, the operator enters the command:

SAMPLE or (SAMPLE COPY) function key

To request a sample print of a font, form, or logo, the operator enters a request of the form:

SAMPLE file-name. file-type

where

file-name specifies the name of a file to be sample-printed

file-type specifies the type of the file to be sample-printed. If only "file-type" is entered, then all files of that type will be sample printed. "file-type" may be any of the following:

FRM if the file is a form cataloged in the FRM library
FNT if the file is a font cataloged in the FNT library
LGO if the file is a logo cataloged in the LGO library

### Continuous Sample Print

SAMPLE

To request the continuous printing (up to 32,767 copies) the operator enters the command:

SAMPLE file-name.TST

where

file-name indicates the name of a form file cataloged in the FRM library

TST specifies the file is to be printed continuously

The specified form will be printed continuously until printing is halted by giving the STOP command or the ABORT command.

## Accounting and System Activity

As print jobs are processed by the 9700, usage data is accumulated and saved by the system software. The REPORT commands provide printed reports of this data, and the ACCOUNT command provides the facility to write the data to tape.

### Customer Usage Statistics

REPORT

The customer usage statistics summary report is printed and delivered to the bin by entering the following command via the keyboard/display.

REPORT USER [,CLEAR]

If "CLEAR" is specified, the summary report is printed and then the accounting log entries are set to zero.

The summary information printed as a result of the REPORT USER command contains accumulated usage data(pages, lines, etc.) for all print jobs (since the last CLEAR) on a "department/JDL" basis. A sample of this output is contained in section "Installation Accounting Report" in Chapter 5. This data may also be written to tape with the ACCOUNT command, see the following section "Accounting File Maintenance".

### System Activity Statistics

REPORT

The operator may obtain a display or printed report of system activity and machine usage information by the following command entered via the keyboard/display.

REPORT ACTIVITY [,CLEAR]

If "CLEAR" is specified, system activity information is printed from the activity log and then the entries are set to zero. The summary information printed as a result of the REPORT ACTIVITY command contains accumulated usage data for all print jobs (since the last CLEAR). An example of the output is as follows:

```
                    SYSTEM ACTIVITY REPORT

                    PAGE COUNTS
PAGES = 1065        PAPER PATH HOLES = 2

                    PROCESSING TIME
INPUT:              HOURS = 0           MINUTES = 6
OUTPUT:             HOURS = 0           MINUTES = 8

                    PROCESSING COUNTS
JOBS = 49           FILES = 47          REPORTS = 49

                    I/O STATISTICS
TAPE MOUNTS = 9     BLOCKS READ = 2065  BLOCKS SKIPPED = 0
```

The Print Descriptor Language command ACCT (left/right part "DEPT = sc") allows the user to set-up department codes under which accounting information will be maintained (see "Accounting Information" in Chapter 5). In addition to the PDL command the OSS level command ACCOUNT must also be invoked by the user. The ACCOUNT command also provides an installation with the capability of writing user and system activity usage data to tape. The format of the command is:

```
                  { ADD,department                                                  }
ACCOUNT           { DELETE,department                                               }
                  { LIST                                                            }
                  { WRITE [,{EBCDIC}] [,{1600}]                                     }
                  {       [,{ASCII }] [,{6250}]                                     }
```

where

ADD — specifies a department name is to be added to the list of departments under which accounting information will be maintained.

department — is a string of alphanumerics (A thru Z, 0 thru 9 and the character ":") specifying the name of the department. The "department" name should be the same as specified in the "DEPT = sc" left/right part of the ACCT command.

DELETE — specifies a department name is to be deleted from the list of department names. A department name cannot be deleted if data for that department exists in the file.

LIST — lists all the department names in the current list on the keyboard/display.

WRITE — specifies writing user and system usage data to tape (only for 9700 systems with writable tape drives). The user and system usage data written to tape is the same data as that printed by the REPORT commands as described previously. The format of the data on tape is defined in Appendix G.

The data will be written to tape starting at BOT and followed by two EOF's. The format will be 80 character records, unblocked, ASCII or EBCDIC (user specified), and density of 1600 or 6250 bpi (user specified).

EBCDIC / ASCII — are keywords defining the character mode in which the tape is to be written. The default is EBCDIC.

1600 / 6250 — is a value specifying the recording density in bits per inch of the tape on which the data is to be written. The default is 1600 bpi.

# Tape Control

## Logical Report Spacing

SPACE

The SPACE command allows the operator to position a tape by logical reports within the limits of the start of job tape position and the end-of-data. The operator must first enter the START command to initiate a job. The SPACE command has the following form:

SPACE n REPORTS

where

n is a decimal integer in the range -32768 to +32767. A positive number indicates forward report spacing, and a negative, backward spacing. The tape cannot be positioned beyond the start-of-job tape position and end-of-data. If the start-of-tape position is on another tape, the tape cannot be positioned to it. The default is the forward space over one report.

REPORTS indicates positioning by logical report units.

Once a job is initiated a SPACE command can be entered anytime, subject to the following restrictions:

1. Input must be stopped. This is indicated by the message "INPUT STOPPED", or when the system is suspended waiting for an operator response.
2. The "REPORTS" option of the START command must not have been invoked.

When report spacing is complete, the system will prompt with the message: ENTER 'CON I' OR 'CON JDE, JDL' to START NEXT REPORT. A "CONTINUE" or another SPACE may then be invoked.

## Page Spacing

SPACE

During the printing of a report, the SPACE command may be used to position pages forward and backward within the current report. Prior to issuance of the spacing function, printing must be stopped with the STOP or STOP O command. The SPACE command is then issued and printing is resumed with the CONTINUE command. The command form is:

SPACE n PAGES

where

n is a decimal integer in the range -32768 to +32767. A positive n results in forward spacing over n pages unless end-of-report terminates the spacing. A negative n results in backward spacing over n pages unless beginning-of-report terminates the spacing. Backward spacing is not allowed if processing the last (or only) copy of a collated report. The default is to forward space 1 page.

PAGES indicates positioning by page units.

Example:

```
STOP O
OS0500 OUTPUT STOPPED
SPACE 200 PAGES
OS0610 PAGE SPACING FORWARD
CONTINUE O
OS0020 RESUMING OUTPUT
```

## Block/File Positioning

MOVE

The MOVE command is used to physically position a magnetic tape a specified number of files or blocks either forward or backward. MOVE may be used during job processing but input must be stopped first (with STOP I). If no job is active the command may be entered directly. The command form is:

MOVE n {FILES / BLOCKS}

where

n is a decimal integer in the range -32768 to +32767. A positive number indicates forward movement and negative backward movement. The default is 1.

FILES indicates positioning by files. A positive "n" moves the tape forward over "n" tape marks unless end of volume (EOV) terminates the tape movement. A negative "n" moves the tape backward over "n" tape marks unless BOT terminates the tape movement. File positioning is the default.

BLOCKS indicates positioning by blocks. A positive "n" moves the tape forward over "n" Interblock Tape Gaps unless EOV or a tape mark terminates tape movement. A negative "n" moves the tape backward over "n" blocks unless BOT or a tape mark terminates tape movement.

If a job is in progress (while the MOVE is invoked), the system will request a "CONTINUE I" to resume. The "CONTINUE I" will not be necessary if no job has yet been initiated.

Whenever a tape move is halted by the encountering of a tape mark, the tape is always positioned at the block one greater than the absolute block number of the tape mark. This is done regardless of the direction of the move.

## Rewinding a Tape

REWIND

The REWIND command is issued by the operator to rewind the current tape on the tape drive. It should be issued after the user has completed processing reports with a tape and is going to mount another. REWIND must be used if a print job is being processed in single report mode ("S" option on the START command) and end-of-data on the tape has not been reached. The command form is:

REWIND

### Paper Tray Selection

FEED

The operator may select or allow the system to automatically select the active paper tray with the FEED command. Manual tray selection allows the selected tray to be emptied and the printer cycled down. Automatic tray selection allows the main tray to be fed until a "paper low" condition is sensed; then, an automatic switch to the aux tray by the operating system allows printing to continue while the main tray is refilled. The command form is:

FEED { MAIN / AUX / AUTO }

where

MAIN or AUX — selects the MAIN or AUX tray, respectively. If the selected tray is not ready, the printer is cycled down and an appropriate message is issued. While operating, the printer is cycled-down whenever the selected tray is emptied.

AUTO — selects the automatic tray selection mode. The operating system will treat the MAIN tray preferentially. The MAIN tray will be fed initially until emptied, then the AUX tray will be selected until either the MAIN tray becomes ready again or the AUX tray is emptied.

After system boot, the MAIN option is implicitly in effect. If no operand is specified and manual tray selection is in effect, the default is selection of the currently inactive tray. If no operand is specified and AUTO is in effect, then the command is ignored. This command form offers a convenient means of switching trays in manual mode. However, if the inactive tray is not ready, the active tray will remain selected.

### Bin Selection

SELECT

The operator may select, or allow the system to automatically select, the active output bin with the SELECT command. Manual bin selection allows the selected bin to be filled; when full, the operating system will stop the printer. Automatic bin selection allows the active bin to be filled; when full, the operating system will automatically switch to the alternate bin thus allowing printing to continue. For both selection modes, approximately seven pages will be delivered to the previously active bin after a new bin selection is made. Thereafter, pages will be delivered to the new bin if it is ready. The command form is:

```
SELECT  { 1[,SAMPLE] }
        { 2[,SAMPLE] }
        { AUTO       }
```

where

1 or 2 — selects bin 1 or 2, respectively. If the selected bin is not ready, the printer is cycled down and an appropriate message is issued. Whenever the selected bin is filled, the printer will be cycled down.

SAMPLE — selects the specified bin as the active bin to be filled and assigns the remaining bin as a logical sample print copy bin. If the selected bin is not ready the printer is cycled down and the message "OS2310 BIN n NOT READY" is issued.

AUTO — selects alternate bin automatically when active bin becomes full. Selects "active" bin if ready or automatically switches bins at initiation of printing.

If SELECT is entered with no operand the alternate bin is selected but does not change mode (i.e., if in AUTO it remains in AUTO, and if in manual it remains in manual).

### Bin Lowering

UNLOAD

The operator may lower an inactive bin with the UNLOAD command. Full bins are lowered automatically by the operating system if they are not actively receiving paper. If the operator wants to unload a partially-filled, active bin, he may do so by selecting (with the SELECT command) the opposite bin and issuing the UNLOAD command. When all paper has been removed from a bin and the bin tray has been returned to its "in" position, the system automatically raises the bin to the ready position. The command form is:

```
UNLOAD  { 1 }
        { 2 }
```

where

1 or 2 — causes bin 1 or 2, respectively, to be lowered if inactive. If no operand is specified, the default action is to unload the inactive bin.

## File Manipulation

The system maintains a catalog of disk resident files. Each of these files is classified by "type" within the catalog. Examples of various file-types are job descriptor library source, forms source, fonts, and logos. Commands are provided to list the catalog by file-type, delete files from the catalog, and copy files into the catalog. The term "file-id" used in the syntax of the commands refers to the form "file-name.file-type"

where

file-name is a 1-6 character file name
file-type is one of the file directories under which file-name is cataloged

The following "file-type" names are used in the command definitions to follow:

| | | |
|---|---|---|
| CMD - DJDE command file | FSL - Form source language | PDE - Page descriptor object |
| CME - Copy modification object | JDL - Job Descriptor object | SYS - System control |
| FNT - Font data | JSL - PDL job source | TMP - Temporary file |
| FRM - Form object language | LGO - Logo data | TSK - System task image |

### Deleting a File

DELETE

The operator may delete a file from the system using the DELETE command. Deletion of a file causes its name to be removed from the system catalog and its file space on disk to be made available for reuse. The command form is:

DELETE file-id[,file-id],...]

where

file-id is of the form "file-name.file-type" as defined above

### Displaying a Catalog

FILE

The operator may display the disk file catalog on the keyboard/display using the FILE command. Specific file types within the catalog or the entire catalog may be displayed on the screen. The command form is:

FILE [file-type [,file-type],...]

where

file-type may be any one of the system's directories defined above (i.e., CMD, CME, FNT, FRM, FSL, JDL, JSL, LGO, PDE, SYS, TMP, and TSK). If no file-type is specified, the default is to display all the files of each catalog.

### Listing a Cat

LIST

The operator may print the disk file catalog using the LIST command. Specific file types within the catalog or the entire catalog may be listed and are delivered to the sample print tray. The command form is:

LIST [file-type[,file-type],...]

where

file-type is one of the file directories under which files are cataloged (i.e., CMD, CME, FNT, FRM, FSL, JDL, JSL, LGO, PDE, SYS, TMP, and TSK). If no file-type is specified, the default is to print the files in each directory.

## Copying a File (Disk to Disk)

COPY

The operator may use the COPY command to create a copy of a file on the system disk. The command form is:

COPY file-id(input) file-id(output)

where

file-id specifies the input or output file which is of the form "file-name.file-type" as defined previously

Example:

```
COPY FORMA.FSL FORMX.FSL
CREATING FILE FORMX.FSL
```

## Copying a File (Unlabeled Tape to Disk)

COPY

COPY TAPE [ {EBCDIC / ASCII} ] file-id

where

TAPE is a keyword specifying the input file will come from tape

EBCDIC / ASCII specifies the recording format of the input tape. EBCDIC is the default.

file-id specifies the output file which is of the form "file-name.file-type" as defined previously

For this command, the input tape must be unlabeled, unblocked, EBCDIC or ASCII, 9 track and 1600 bpi. The maximum record length is 133 characters although only the first 72 characters of each record will be entered into the disk file. The largest file that can be created is 5000 records.

Example:

```
COPY TAPE UNSYS.JSL
```

Will copy a file from an unblocked, unlabeled tape to disk. The file will be named UNSYS and cataloged in the JSL file directory.

## File Maintenance

Files may be saved/restored to and from formatted 9700 labeled tapes. In order to write files to tape from the 9700, a writable tape unit must exist on the system. Before a tape may be written, it must be initialized (TAPE VOLINIT). After files have been written the TAPE REWIND must be used to rewind the tape. The COPY command is used to read/write the files.

### Labeled Tape Initialization

TAPE

When writing to a new tape on the 9700, the following command must precede any COPY commands:

TAPE VOLINIT [, 1600 / , 6250]

where

VOLINIT — is a keyword specifying that the tape is to be initialized.

1600 / 6250 — specifies the recording density in bits per inch. This is the only way to set tape density. The default is 1600 bpi.

### Disk File to Labeled Tape

COPY

COPY TAPE WRITE LABEL { file-id (input) file-id (output) / file-type / ALL }

This command will append new files at the end of the current files on the tape if the tape has been previously initialized. If there are not other files, then the new files will be written starting at BOT.

where

TAPE / WRITE / LABEL — are keywords specifying that files are to be written to a 9700 labeled tape.

file-id — specifies the name of the file to be input from disk (input) and the name to be given the file on tape (output). This name is of the form "file-name.file-type" as defined previously.

file-type — specifies the name of a catalog (FSL, FRM, FNT, LGO, JSL, JDL, PDE, CME, CMD) in which all the files of that catalog will be written to tape.

ALL — specifies that all user files are to be copied to tape. Files will be written to tape from the FSL, FRM, FNT, LGO, JSL, PDE, CME, and CMD directories.

## Labeled Tape to Disk File

COPY

COPY TAPE LABEL { file-id / NEXT / ALL }

where

TAPE LABEL } are keywords specifying that the input file(s) will come from a 9700 labeled tape.

file-id specifies the name of the output file on which the data will be stored. This name is of the form "file-name.file-type" as defined previously.

NEXT specifies that the next file on the tape is to be copied to disk.

ALL specifies all the files on tape are to be copied to disk.

Example:

```
COPY TAPE LABEL OCRA.FNT
SEARCHING FOR FILE OCRA.FNT
CREATING FILE OCRA.FNT
```

The above command will find and copy the file OCRA from the tape into the file named OCRA. It will be cataloged in the FNT file directory. If another file existed on this tape (after the one just copied) then the command,

```
COPY TAPE LABEL NEXT
```

will copy the next file from tape and store it in a "file-name.file-type" as specified in the tape label information. The command,

```
COPY TAPE LABEL ALL
```

will copy all the files from the 9700 labeled tape, store them with the same "file-name" as on tape and catalog them in their "file-type".

## Labeled Tape Rewind

TAPE

After copying files from disk to a 9700 labeled tape, the TAPE REWIND command should be used to rewind the tape. It has the form:

```
TAPE REWIND
```

# 8. EDITOR

## Introduction

The EDITOR is a 9700 system task which provides file editing facilities as part of normal operator communication and control. These facilitites are available to the user for the creation, modification and maintenance of files. The EDITOR has commands for creating and modifying source files (JDL, FDL, CME, PDE, DJDE source statements) as well as commands for the maintenance of any type of disk files.

The EDITOR allows the user to directly access each line (or record) of a source file. The user enters single line EDITOR commands, via the keyboard/display, which provide the following facilities:

- Creating a sequenced source file.
- Inserting, reordering, and replacing lines or groups of lines of text.
- Selective printing to the keyboard/display or sample print tray.
- Reordering and renumbering of records within a file.
- Merging part of one file into another.
- Matching, moving and substituting character strings within a specified range of text lines.
- File maintenance (allowing the user to copy and delete files).

Each source file has line numbers associated with its data records. These line numbers are supplied by the EDITOR during file creation. Files entered via magnetic tape will have line numbers added to them at the time they are entered into the system. When a source file is being modified, the commands entered by the user are applied to a "working file", not to the permanent file on disk. At the time a request is made to edit an existing file, the specified file is copied into working storage. All editing operations are performed upon the working file and it is saved permanently on the disk only when specifically requested by the user.

Examples of commonly used EDITOR commands are illustrated in Figures 8-1, 8-2 and 8-3.

### File Directories

A user may edit source files which are cataloged in the following "file-type" directories:

| File Type | Contents |
|---|---|
| CMD | DJDE Command File source |
| FSL | Form source language |
| JSL | PDL job source language |

When files are initially created they must be "saved" in one of the above directories. Non-source files are cataloged in the following directories:

| File Name | Contents |
|---|---|
| CME | Copy modification object |
| FNT | Font data |
| FRM | Form object |
| JDL | Job descriptor object |
| LGO | Logo data |
| PDE | Page descriptor object |
| TMP | Temporary file |
| TSK | System task image |

Files within these directories may be accessed only by a subset of the EDITOR commands. These commands allow listing of the file type directories (LIST and FILE), deleting (DELETE) files or copying (COPY) files. LIST, FILE, DELETE and COPY are also available as system level commands (see "Disk File Manipulation" in Chapter 7).

### Calling the Editor

To create a new source file the user requests the EDITOR facilities by typing EDIT on the 9700 keyboard/display. If a file already exists on disk and is to be modified, it can be edited by typing in:

EDIT [file-name.file-type](RETURN)

where "file-name" is a 1-6 character file name which exists on disk and "file-type" is one of the file directories (CMD, FSL, JSL) under which the "file-name" is cataloged.

Note that only the first three characters of the command need be entered. This is true of all EDITOR commands. The (RETURN) indicates the pressing of the RETURN key on the keyboard/display. All commands and data lines are ended by depressing the RETURN key.

After the user types in the request for the EDITOR processor, the EDITOR prompts the user for a command by typing "EDIT >" which indicates that another command may be entered.

## Editor Commands

EDITOR commands fall into the following three categories:

- File commands: Commands that apply to an entire file. These commands may be given at any time.
- Record commands: Commands that act upon the record or a group of records within a file. These commands may be given only after a file has been selected for editing.
- Intra-record commands: Commands that make changes within an individual record. These commands generally manipulate character strings and may be given only after a specific set of records has been selected.

These commands and categories are summarized in Table 8-1. The syntax of each command is explained in the following section along with examples of usage.

Table 8-1. EDITOR Command Summary

| FILE COMMANDS | FUNCTION | Reference Page |
|---|---|---|
| CLEAR | Clears the entire contents of the working file | 8-5 |
| CONVERT | Sets mode to convert | 8-5 |
| COPY | Copies files from tape/disk to disk | 8-6 |
| DELETE | Deletes a file from the system | 8-6 |
| EDIT | Begins an editing session | 8-6 |
| END | Ends an editing session | 8-7 |
| FILE | Lists disk file catalog on the operator's display | 8-7 |
| GET | Gets an existing file and copies it into working storage for editing | 8-7 |
| KEYS | Display of beginning and ending line numbers | 8-8 |
| LIST | Lists disk file catalog on the printer | 8-8 |
| MERGE | Merges an existing file into current working storage | 8-9 |
| NOCONVERT | Sets mode to no convert | 8-9 |
| REVIEW | Sequential listing of files with optional delete | 8-9 |
| SAVE | Saves the contents of working storage as a permanent file | 8-9 |
| **RECORD COMMANDS** | **FUNCTION** | **Reference Page** |
| DISPLAY | Displays source lines on the operator's console | 8-10 |
| DUPLICATE | Duplicates lines from the file into another area of the file | 8-10 |
| FIND | Finds and displays lines containing a specified text string | 8-11 |
| INSERT | Inserts lines using a specified line number into the file | 8-11 |
| MODIFY | Specifies a range of lines to be affected by intra-record commands | 8-11 |
| MOVE | Moves lines from one place to another in the file | 8-12 |
| PRINT | Produces hard copy of the working file to the sample print tray | 8-12 |
| REMOVE | Removes a specified group of lines from the file | 8-12 |
| RENUMBER | Renumbers the working file | 8-13 |
| REPLACE | Specifies that lines are to be removed and others inserted in their place | 8-13 |
| STEP | Implicitly specifies the modify range to be the next record | 8-13 |
| **INTRA-RECORD COMMANDS** | **FUNCTION** | **Reference Page** |
| D | Delete string | 8-14 |
| F | Insert following string | 8-14 |
| P | Insert prior to string | 8-14 |
| S | Substitute string | 8-14 |

## Editor Conventions

The syntax of EDITOR commands are defined in detail in the following section. The commands are in alphabetic order grouped according to their type (File, Record, Intra-record). Conventions used in defining the syntax are as follows:

- When typing in commands, only the first three characters need be entered. For example, typing in REN or RENUMBER will be interpreted as the same command by the EDITOR.

- All commands are terminated by pressing the RETURN key on the keyboard.

- The RUBOUT function key can be used to erase characters. Each time it is pressed, it erases the previous character on the current line and backspaces one position.

- Examples shown in this chapter will differentiate user typed commands from system supplied messages by underlining those printed by the system.

- The term "file-id" refers to the name of a file and the file directory in which it is cataloged. Thus, when specifying "file-id", the user must type "file-name.file-type". For example, SIGMA.JSL would be the file-id of a file named SIGMA which is cataloged in the JSL file directory.

- When "n-m" is used in the command syntax the following usage is implied:

  n-m = starting and ending line number for command
  n = only line number for this command
  n- = starting line number to end of file
  -m = beginning of file to ending line number

- The maximum line number for a file is 32,765.

## File Commands

### CLEAR

This command is used to erase all data lines from the working file. It has no parameter option and is used when a new file is to be created.

Example:

```
EDIT > CLEAR                Clear working storage
EDIT > INSERT 10,10         Issue next command
```

If a file exists in working storage (and has not yet been SAVEed) when the CLEAR command is entered, the EDITOR will respond with the following message:

```
THE FILE HAS NOT BEEN SAVED
DO YOU WANT TO SAVE IT (YES/NO)?
```

If the user types in NO the CLEAR command takes effect and the message

```
WORK FILE CLEARED
```

is printed on the console. If YES is typed the CLEAR command is ignored and the message COMMAND IGNORED is printed. The user may then save the file by typing in the SAVE command.

### CONVERT

This command allows the user to edit files containing lower case characters using the upper case only keyboard/display. In order to replace lower case characters in a file, the lower case characters must be bracketed by the # symbol. The # symbol is used by EDITOR to indicate the start and the end of conversion to lower case characters. If the file does not actually contain lower case characters, the CONVERT command need not be used. After using the CONVERT command and text modification requiring it, the CONVERT mode should be "reset" with the NOCONVERT command (described below).

Example:

The CONVERT command would be needed if the text line "This is the end" (which was entered into a disk file from tape) needs to be altered. This text line could not be created (displayed) in lower case on the 9700 keyboard/display character because it is only possible to enter (display) upper case characters from (on) it.

The following command sequence illustrates the use of the CONVERT command to change the word "This" to "That".

```
  :
EDIT > CONVERT
EDIT > MOD 10
THIS IS THE END
EDIT > S/T#HIS/T#HAT/
THAT IS THE END
EDIT > NOCONVERT
```

The keyboard/display prints only in upper case but internally to 9700 the sentence is initially represented as "This is the end" and then as "That is the end".

After the use of CONVERT command is completed, the user should "reset" this mode by using NOCONVERT (described below).

## COPY

This command allows the user to create new files on the disk. A file to be copied may be input from tape or disk. The parameters of this command are defined in "Copying a File" in Chapter 7.

## DELETE

This command allows the user to delete a file from the system. The command form is:

DELETE file-id[,file-id] ,...]

where "file-id" is of the form "file-name.file-type" as defined previously. To delete a file from the system, the user must have an appropriate level of access. See "File Access Protection" in Chapter 7 for further details.

Example:

```
DELETE UNIVAC.JSL,FORMX.FSL
UNIVAC.JSL DELETED
FORMX.FSL DELETED
```

## EDIT

This command specifies that an editing session is to begin. It has the form:

EDIT [file-id]

If the "file-id" is specified then the requested file is brought into working storage.

Example:

```
OS1000 READY FOR COMMANDS
EDIT  SYSPDL.JSL
EDIT >
```

Additional commands then may be entered.

**END**

This command terminates an editing session. Control is returned to the system and no more EDITOR commands will be accepted until the EDITOR is requested again. The message OS1000 READY FOR COMMANDS will be displayed on the console after the END takes effect.

If the user types in END and a file (one just created or updated) has not been saved (SAVE command) the following message will be displayed on the console:

```
THE FILE HAS NOT BEEN SAVED
DO YOU WANT TO SAVE IT (YES/NO)?
```

If a YES is typed, the EDITOR ignores the previous END command and the message "COMMAND IGNORED" is displayed. The user may then type in the SAVE command. If the user types in NO, the EDITOR ends the editing session and retains the contents of working storage for a future session. CAUTION: There is only one workfile, thus any intervening use of the EDITOR is likely to destroy its contents.

**FILE**

This command allows the user to display on the keyboard/display the file in each disk file directory. The command form is:

FILE [file-type [,file-type],...]

where "file-type" may be any one of the system's directories previously defined (i.e., CMD, CME, FNT, FRM, FSL, JDL, JSL, LGO, PDE, SYS, TMP, TSK). If none is specified, the default is to list all the files of each catalog.

**GET**

This command specifies that an existing file is to be brought into working storage. It has the form:

GET  file-id

Example:

```
OS1000 READY FOR COMMANDS
EDIT
EDIT > GET  SYS2.JSL
```

If the GET command is typed and a file exists in working storage that has not been saved (SAVE command) then the message:

```
THE FILE HAS NOT BEEN SAVED
DO YOU WANT TO SAVE IT (YES/NO)?
```

is printed on the console. A YES response causes the EDITOR to ignore the GET command. The user can then issue the SAVE command. If the response is NO, the GET command is performed and the previous contents of working storage is lost.

### KEYS

This command is used to request that the beginning and ending line numbers of the working file be displayed. This command has no optional parameters.

Example:

```
EDIT > GET SYS2.JSL
EDIT > KEYS
BEGINNING LINE NUMBER 000010
ENDING LINE NUMBER 000050
```

### LIST

This command allows the user to list on the printer the files in each disk file directory. It has the form:

LIST [[file-type [,file-type], ...]]

where "file-type" is defined as one of the file directories under which files are cataloged (i.e., CMD, CME, FNT, FSL, JDL, JSL, LGO, PDE, SYS, TMP, and TSK).

Examples:

```
EDIT > LIST JSL, FSL, FRM
- - - -
  :
- - - -
- - - -
- - - -
```

The system will list on the printer all the files in the specified file directories (i.e., JSL, FSL, and FRM).

```
EDIT > LIST
```

The system will list all the files in all the file directories.

## MERGE

The purpose of this command is to add the contents of a disk file to working storage. The current records in working storage will not be destroyed when the records of the new file are brought in. The MERGE command has the form:

MERGE file-id [,n [,s]]

where n is the sequence number to be assigned to the first record of the file, and s is the sequence increment.

## NOCONVERT

This command is a "reset" of the CONVERT command described previously. By "resetting" the CONVERT command no upper-to-lower case conversion will take place. This is the default mode of the EDITOR. The NOCONVERT command has no optional parameters.

## REVIEW

This command allows the user to automatically list files in a directory with the option of deleting each file. The command form is:

REVIEW [file-type [,file-type] , . . .]

where "file-type" may be any one of the systems directories previously defined (i.e., CMD, CME, FNT, FRM, FSL, JDL, JSL, LGO, PDE, SYS, TMP, TSK). To delete a file from the system, the user must have an appropriate level of access (see "File Access Protection" in Chapter 7).

After the user requests a set of files in a "file-type" to be displayed, the files are displayed on the keyboard/display one at a time. The user has the option of deleting a file (by typing in "D" followed by a RETURN), or exiting the display process of the current "file-type" (by typing in "X" followed by a RETURN).

Any user response other than "D" or "X" will result in the next file being displayed and the current file being retained. Any file name containing a period or dollar sign cannot be deleted and will be displayed followed by the message '*PROTECTED*'.

## SAVE

This command specifies that the contents of the working file are to be saved in permanent file storage. It has the form:

SAVE [file-id]

If no "file-id" is specified, the contents of the working file is saved under the name currently associated with the working file (as set by a previous GET command, a previous SAVE command, or as set when EDIT is called). Otherwise the operator is prompted for a "file-id" under which to save the contents of the working file.

## DISPLAY

This command is used to obtain a listing of all or selected lines of the current working file on the display screen.

This command has the form:

DISPLAY n-m

where n-m are as defined previously in section, "EDITOR Conventions".

Examples:

| | |
|---|---|
| DISPLAY | Display all lines |
| DISPLAY 10-20 | Display lines 10 through 20 |
| DIS 5 - | Display lines 5 through the end of the file |
| DIS - 100 | Display lines from the beginning of file through line 100 |

If the number of lines to be displayed does not exceed the screen display capacity, all requested lines are displayed on the screen. If the number of lines to be dispalyed exceeds the display screen capacity, the user will be prompted with the message:

ENTER RETURN TO CONTINUE

to indicate that all requested lines have not been displayed. To continue displaying another set of lines, the user depresses the RETURN key. If any other response is given, the display is halted.

## DUPLICATE

This command specifies that a line or group of lines will be duplicated at another point in the file. It has the form:

DUPLICATE n [- m] ,p [,s]

The specified lines are moved to a new location starting with line number p. The new line numbers will be incremented by s. the default value for s is 1. The command is identical to the MOVE command except that original lines are not deleted from the file.

## FIND

This command specifies that all lines in the specified range containing the given string are to be displayed. It has the form:

FIND [n [- m]]/string/ .

Examples:

| | |
|---|---|
| FIND /JDE/ | Display each line that has the characters "JDE" in it. |
| FIN 5-100/JOB/ | Display each line from 5 to 100 that has the characters "JOB" |
| FIN 5/SYSTEM/ | Display line 5 if it contains the word "SYSTEM" |
| FIN 5-/SYSTEM/ | Display each line from 5 to end of file if it contains the word "SYSTEM" |
| FIN -100/JOB/ | Display each line from 1 to 100 that has the characters "JOB" |

## INSERT

This command specifies that a line or group of lines are to be inserted into the file. It has the form:

INSERT n [,s]

Lines are inserted starting at n and incremented by s. Line n must not already exist. The operator will be prompted by the next sequence number and input terminates upon receiving a null line or encountering an existing record. The default value for s is 1.

Examples:

| | |
|---|---|
| INSERT 10,10 | Insert lines starting at 10 and increment by 10 for next line |
| INS 15 | Insert lines starting at 15 and increment by 1 |

Another method for inserting lines into the working file is to just enter the desired line number, one blank, and then the text. For example, to enter text on line 55 the user would key-in:

EDIT > 55 ACCT USER =(BIN,TRAY);

and then press the RETURN key. With this type of insertion the EDITOR will replace existing lines without any further notification to the user.

## MODIFY

This command specifies a range of lines which may be affected by subsequent string modification commands. These lines will be displayed on the console after the MODIFY command has been issued. String modification commands are defined below in section "Intra-Record Commands". This command has the form:

MODIFY n-m

Examples:

| | | |
|---|---|---|
| EDIT > | MODIFY 5-100 | Prepare to modify lines 5 through 100 |
| EDIT > | S/JDE/JOB/ | Subsitute occurrences of "JDE" with "JOB" in lines 5 through 100 |
| EDIT > | MOD 5 | Prepare to modify line 5 |
| EDIT > | D/HOST=OSWTR/ | Delete text "HOST=OSWTR" from line 5. |

If no match is found in the modify range, the message NO MATCH FOUND is displayed.

## MOVE

This command specifies that a line or group of lines are to be moved to new line positions. It has the form:

MOVE n [-m] ,p [,s]

The specified lines (n-m) are moved to a new location starting with line number p. The new line numbers will be incremented by s. The default value for s is 1. Lines n-m will be removed (deleted) from the file.

## PRINT

This command is used to obtain a hard copy listing delivered to the sample print tray of all or selected lines of the current working file. It has the form:

PRINT n-m

In order to print the file, the user must exit (END command) the EDITOR after the PRINT command is given. The current working file will be intact if the user re-enters the EDITOR after the file is printed.

## REMOVE

This command specifies that a line or group of lines are to be removed (deleted) from the working file. It has the form:

REMOVE n-m

For this command, n is a required parameter; if there is a range, m is also required (i.e., hyphen can not be used to specify initial or terminal line).

Examples:

| | |
|---|---|
| REMOVE 5-25 | Remove lines 5 through 25 |
| REM 15 | Remove line 15 |

Another way to remove a line from the working file is to enter the line number (of the line to be removed) followed by the RETURN key. To delete a group of lines, the user enters the beginning and ending line numbers separated by a dash.

Examples:

```
EDIT > 55 (RETURN)          Delete line 55
     1 RECORD DELETED

EDIT > 10-40 (RETURN)       Delete lines 10 through 40
     4 RECORDS DELETED      Assuming line numbers are incremented by 10
```

## RENUMBER

This command is used to renumber the lines of the working file. Renumbering begins with line number n and successive data lines are assigned numbers in increments of s. The default values for n and s are 10. This command has the form:

```
RENUMBER [n [,s]]
```

## REPLACE

This command specifies that a line or group of lines are to be removed and lines are to be inserted in their place. It has the form:

```
REPLACE   n [-m] [,s]
```

This is identical to the commands:

```
REMOVE    n -m
INSERT    n [,s]
```

## STEP

This command will display each record of a file, one at a time, starting with the next record as specified in the MODIFY command. It has the form:

```
STEP
```

Example:

```
EDIT > MODIFY 10                Set modify range
     ...line 10 is printed...
EDIT > STEP
     ...text of next line after line 10 is printed...
EDIT > S/MOST/HOST/             Change text in next line after line 10
     1 STRING CHANGED
EDIT > STEP
     ... next line is printed
EDIT > ...user may enter an appropriate intra-record command (i.e., S,F,P or D) or another STEP
     :
     :
```

## Intra-Record Commands

The following commands allow the user to modify parts of a record (or records). These commands are used with the MODIFY command which specifies the record (or records) to be altered. After a line has been altered it will be displayed.

**D**

Delete a string from a record (or records) in the working file. The command has the form:

D [k] /string/

Where k indicates that the kth occurrence of "string" in each line will be affected (k=0 indicates that all occurrences of specified string are to be affected). When k is specified, it must be preceded by one blank character. The default value of k is 1. "string" is deleted from lines specified in the MODIFY command.

Examples:

| | |
|---|---|
| EDIT > MODIFY 5 | Modify line 5 |
| EDIT > D/SYSTEM/ | Delete the first occurrence of "SYSTEM" from line 5. |
| EDIT > MOD 5-70 | Range of lines which may be altered is 5 through 70. |
| EDIT > D 0/JDE/ | Delete all occurrences (i.e., k=0) of "JDE" from each line, starting with line 5 and proceeding through line 70. If k were set to 2 (instead of 0) then the second occurrence of JDE in each line (if any) would be deleted. |

**F**

Insert following a string. This command has the form:

F [k] /string1/string2/

k has the same meaning as in D (delete string). "string2" is inserted following "string1" for lines specified in the MODIFY command.

**P**

Insert prior to string. This command has the form:

P [k] /string1/string2/

k has the same meaning as in D (delete string). "string2" is inserted just prior to "string1" for lines specified in the MODIFY command.

**S**

Substitute string. This command has the form:

S [k] /string1/string2/

k has the same meaning as in D (delete string). "string2" is substituted for "string1" for lines specified in the MODIFY command.

## Usage Examples

The following figures illustrate the use of some of the frequently used EDITOR commands. All commands in these examples are terminated by pressing the RETURN key. Underlined lines are those messages displayed on the console by the EDITOR.

```
OS1000 READY FOR COMMANDS
EDIT
EDIT > CLEAR
WORK FILE CLEARED
EDIT > INS 10,10
000010 SYSPDL:SYSTEM;
000020 VOLUME HOST = POWERVS, PLABEL = YES;
000030 BLOCK LENGTH = 2048;
000040 ACCT USER = (BIN,TRAY);
000050 21:JOB;
000060 END;
000070 (RETURN)
EDIT> SAVE SYSPDL.JSL
EDIT> CLEAR
EDIT> END
OS1000 READY FOR COMMANDS
```

Figure 8-1. Create and Save a File in the JSL File Directory.

```
OS1000 READY FOR COMMANDS
EDIT
EDIT >  CLEAR
WORK FILE CLEARED
EDIT >  GET SYSPDL.JSL
EDIT >  KEYS
BEGINNING LINE NUMBER 000010
ENDING LINE NUMBER 000060
INS 11
000011  /*  SYS JDL  */
000012  (RETURN)
EDIT >  FIND 10-60/JOB/
21:JOB;
EDIT >  MODIFY 50
21:JOB;
EDIT >  S/JOB/JDE/
21:JDE;
EDIT >  DIS 40
ACCT USER = (BIN,TRAY);
EDIT >  REMOVE 40
      1 RECORDS DELETED
EDIT >  REN
EDIT >  DISPLAY
        ____
        ____
        ____   } Lines displayed on console
        :
        ____
EDIT >  SAVE SYSPDL.JSL
EDIT >  CLEAR
WORK FILE CLEARED
EDIT >  END
OS1000 READY FOR COMMANDS
```

Figure 8-2. Modify and Save File of Figure 8-1.

```
OS1000 READY FOR COMMANDS
EDIT
EDIT >  COPY  SYSPDL.JSL  TSTSYS.JSL
CREATING FILE TSTSYS.JSL
EDIT >  FILE JSL
      OSWTR
      IBMPDL
      TSTPDL   } List of all files in JSL directory
        :
      TSTSYS
EDIT >  GET TSTSYS.JSL
EDIT >  PRINT
PRINT JOB QUEUED, MUST EXIT TO PRINT
EDIT >  END
OS1010 STARTING JOB 00003
OS1000 READY FOR COMMANDS
OS1020 JOB 00003 HAS COMPLETED INPUT PHASE
OS0020 RESUMING OUTPUT
OS1030 JOB 00003 HAS COMPLETED PRINTING
OS1000 READY FOR COMMANDS
EDIT
EDIT >  DELETE SYSPDL.JSL
FILE SYSPDL.JSL DELETED
EDIT >  END
OS1000 READY FOR COMMANDS
```

Figure 8-3. File Manipulation Commands

# 9. OPERATING SYSTEM DIAGNOSTIC SOFTWARE

## Introduction

Operating System Diagnostic Software (OSDS) is a task of the 9700 Operating System Software (OSS) which provides the primary diagnostic software support for input and output hardware subsystems including their controllers. It provides analysis and display of information logged by OSS for recoverable failures in the control subsystem. OSDS also provides guidance to the operator for recovery and generates codes for the dispatch of a customer engineer OSDS is also the user's primary software tool for verifying the operability of the 9700 hardware.

## OSDS Communications

While OSDS is operating, it provides a continuous display of information to the user as to what OSDS is doing and/or what it expects of the user. The user communicates with OSDS by requesting a particular mode (PROBLEM or VERIFY) and OSDS responds to the user via formatted displays to the keyboard/display. OSDS also produces printer test data (to test or verify operation of the printer) and a hard copy of error log information.

## PROBLEM Mode

The PROBLEM mode of OSDS is activated by the operator because of a request by OSS or by an operator recognized problem.

The system requests the operator (via a message to the keyboard/display) to initiate PROBLEM mode analysis when further productive operation of the 9700 is not possible. This is done only after all appropriate automatic and operator assisted recovery techniques have been exhausted. The operator requests the system to initiate the PROBLEM mode when a problem exists which is not detectable by OSS (e.g., copy quality) or degraded operation of the 9700 is possible but not acceptable.

The PROBLEM mode is initiated by the operator typing "PROBLEM" on the keyboard/display. After the operator's request, OSDS will respond by displaying on the console its selection/display options. An example of an operator/OSDS dialogue is illustrated in Figure 9-1.

Effective dispatch of a customer engineer is provided by OSDS in the PROBLEM mode by determining the nature of the problem from the user inputs solicited or from the OSS error log. OSDS will inform the user that a customer engineer is required under the following conditions:

- A OSS reported problem indicates the need for a service call.
- An operator reported problem indicates the need for a service call.
- A diagnostic test failure indicates the need for a service call.

## VERIFY Mode

OSDS provides the verification of the tape subsystem, the printer subsystem, and/or the entire system by performing the following actions:

- Analyzes any system reported problem.
- Reports on system or subsystem capability for job processing (i.e., no detected problems operating possible with minor problems, printer inoperable).
- Provides guidance on further operator action.

The VERIFY mode of OSDS is initiated by the operator typing "VERIFY" on the keyboard/display.

```
OS1000 READY FOR COMMANDS                          (OSS command level)
PROBLEM                                            (user types in PROBLEM)
OSDS          ANALYSIS: RUNNING                    (system responds)
OSDS          SEL ANALYSIS RUNNING
OSDS          PROBLEM ANALYSIS: RUNNING

WHICH OF THE FOLLOWING TYPES OF PROBLEMS DO YOU WISH TO REPORT?

    1.  COPY QUALITY PROBLEMS.
    2.  FREQUENT JAMS.
    3.  BIN PROBLEMS.
    4.  TRAY PROBLEMS.
    5.  TAPE PROBLEMS.
    6.  OTHER SYSTEM PROBLEMS.

        INDICATE SELECTION BY ENTERING NUMBER
>1(RETURN)                                         (user makes selection)
WHICH COPY QUALITY PROBLEM DO YOU HAVE?

    1.  COPY IS DIRTY.
    2.  COPY HAS DARK BACKGROUND.
    3.  COPY HAS LINES.
    4.  COPY HAS STREAKS.
    5.  COPY IS BLANK.
    6.  COPY HAS DELETIONS (BLANK SPOTS).
    7.  COPY IS BLACK.
    8.  COPY HAS UNFUSED TONER.
    9.  OTHER.
```

Figure 9-1. Operator/OSDS Sample Display

```
        INDICATE SELECTION BY ENTERING NUMBER
>1(RETURN)                                          (user makes selection)
DO YOU WANT TO REPORT ANY PROBLEMS IN ADDITION TO THOSE BELOW?

    10.00.50    COPY IS DIRTY

    1.  YES. I WISH TO REPORT ADDITIONAL PROBLEMS.
    2.  NO MORE PROBLEMS TO REPORT.

        INDICATE SELECTION BY ENTERING NUMBER
>2(RETURN)                                          (user makes selection)
CALL SERVICE AND REPORT THESE NUMBERS

    10.00.50

OPERATOR - INDICATE THE ACTION YOU HAVE TAKEN.

    1.  SERVICE CALL HAS BEEN PLACED. WAITING FOR SERVICE.
    2.  SERVICE CALL HAS BEEN PLACED. RETURN TO THE OPERATING SYSTEM.
    3.  SERVICE NUMBERS COPIED FOR LATER CALL. RETURN TO OPERATING SYSTEM.
    4.  NONE OF THE ABOVE. RETURN TO THE OPERATING SYSTEM.

        INDICATE SELECTION BY ENTERING NUMBER
>2(RETURN)                                          (user makes selection)
OSDS HAS RETURNED TO OSS
OS1000 READY FOR COMMANDS                           (return to OSS command level)
```

Figure 9-1. Operator/OSDS Sample Display (Continued)

# 10. ELECTRONIC FORMS

## General

The Electronic forms facility is a major feature of the Xerox 9700. Electronic forms provide the capability for precisely defining forms in order to highlight information for maximum visual impact. They also improve the readability of the printed output without the typical disadvantages associated with preprinted forms (i.e., registration and form-alignment problems, expensive inventory of preprinted forms, and expenditure of operator's time to change forms). The Forms Description Language (FDL) provides the user with the facilities to create "electronic" forms. It is an easy to use, keyword oriented language which allows the user to specify:

- Name of each form uniquely.
- Page orientation - portrait or landscape.
- Font selection - multiple fonts can be specified.
- Vertical and horizontal lines - specified and positioned in line printer measurements (line number and character position) or inches, centimeters, or dots.
- Line width and type - hairline, medium, or bold and solid, broken, or dotted.
- Shading - to define shaded areas and a variety of shading densities.
- Positioning of captions - exact location on a page or automatic placement within a specified area.
- Selection and positioning of logos.

A summary of the FDL command syntax and usage of the FDL processor is provided in the following section. Further details on using FDL to create forms are available in the Forms Creation Guide (Xerox Publication No. 91 00 01).

## FDL Processor

The FDL processor (a 9700 system task) accepts source language FDL statements created by the user. The source statements may be processed by FDL from either tape or a file on the system disk. Source statements processed by FDL from tape are automatically cataloged in the FSL library on disk, thereby allowing for corrections and modifications via the 9700's editing facilities (described in Chapter 8). The editing facilities can also be used to create FDL source files directly on disk.

The FDL processor is invoked by the following command entered from the operator's console:

FDL [filename] [,NOPRINT] [,TRAY] [,DEBUG]

where

filename specifies the name of a source Forms Description Language file in the Form Source Library (FSL) on the system disk that is to be used for source input to the FDL task. If no filename is specified, it is assumed that source input is to come from tape.

NOPRINT specifies that no listing of the FDL source statements, no form summary, and no sample form are to be printed. However, if an error is detected during the processing of the form, the source statement listing and form summary are unconditionally printed.

TRAY specifies that the sample form and listing/summary are to go to the sample print tray instead of the currently active bin. The nominal capacity of the sample print tray is 25 sheets and it is the operator's responsibility to ensure that the capacity is not exceeded.

DEBUG specifies that no attempt will be made to print the sample form. This ensures that the listing/summary can be printed even if the sample form can not be printed.

## FDL Command Syntax Summary

In defining the FDL source statement syntax the following symbols are used:

| SYMBOL | MEANING | |
|---|---|---|
| $c_o$ | Coordinate, origin | Defined in y unit, x unit, where y is measured along the vertical axis, and x along the horizontal axis. |
| $c_s$ | Coordinate, start | |
| $c_e$ | Coordinate, end | |
| $c_a$ | Coordinate, absolute | |
| $c_i$ | Coordinate, incremental | |
| d,m,n,x,y | Positive numerical values which are expressed as integers when used in conjunction with DOTS or as decimal values with two place accuracy when used with other units (i.e., 1.00 INCH or 2.54 CM). | |
| unit | IN [CH [ES] ],CM, CENTIMETERS, or DOTS | |
| id | Identifier consisting of one to six alphanumeric characters | |

### Set-up Commands

FORM id;

{LANDSCAPE / PORTRAIT} [ PAGE [SIZE] [IS] n [ IN [CH [ES]] / CM / CENTIMETERS / DOTS ] [WIDE] [BY] m [ IN [CH [ES]] / CM / CENTIMETERS / DOTS ] [HIGH] ];

GRID [UNIT] [IS] { FMTm / n IN [CH [ES]] / n CM / n CENTIMETERS / n CPI n LPI / DOTS } [ ORIGIN y [ IN [CH [ES]] / CM / CENTIMETERS / DOTS ] x [ IN [CH [ES]] / CM / CENTIMETERS / DOTS ] ];

FONT id;

FONTS id ... id;

### Section Commands

```
[BEGIN] SECTION id;
DO SECTION id AT y [unit] x [unit];
END SECTION;
```

### Form Structure Commands

LINE or LINES:

```
AT c_o [unit] [DRAW] [n] [HORIZONTAL | VERTICAL] {LINE | LINES} [IN unit] [FROM] c_s [unit]

TO c_e [unit] [USING] [SOLID | BROKEN | DOTTED] [HAIRLINE | 0 | 1 | 2]

[ [AND] [REPEAT] [HORIZONTALLY | VERTICALLY] {AT c_a [unit] [c_a [unit] ...] | EVERY c_i [unit]} ];
```

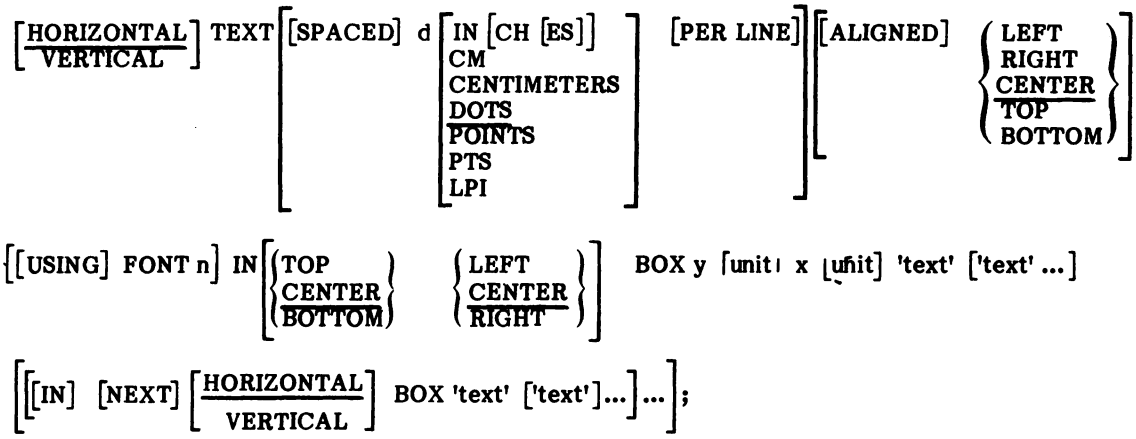
BOX or BOXES;

```
AT y [unit] x [unit] [DRAW] [n] {BOX | BOXES} [IN unit] n [unit] [WIDE] [BY] m [unit] [HIGH]

[ [USING] [SOLID | BROKEN | DOTTED] [HAIRLINE | 0 | 1 | 2]
          [SHADING] [LIGHT | MEDIUM | HEAVY] ]

[ [AND] [REPEAT] [HORIZONTALLY | VERTICALLY] {AT c_a [unit] [c_a [unit] ...] | EVERY c_i [unit]} ];
```

TEXT:

```
[HORIZONTAL | VERTICAL] TEXT [ [SPACED] d [IN [CH [ES]] | CM | CENTIMETERS | DOTS | POINTS | PTS | LPI] ]
    [PER LINE] [ [ALIGNED] { LEFT | RIGHT | CENTER | TOP | BOTTOM } ]

[ [USING] FONT n ] AT y [unit] x [unit] 'text' ['text' ...] ;
```

TEXT IN BOX:




## LOGO, COMMENT and END Commands

```
LOGO id AT y [unit] x [unit];
COMMENT text;
END;
```

# 11. FONTS

## Introduction

A font is a specific set of characters of one size, style, and orientation (portrait or landscape). A font may have up to 240 alphanumeric characters, punctuation marks, line segments, shading elements, special symbols, and form characters. The number of lines per inch (3 to 18) as well as the number of characters per inch (4 to 30) are both inversely proportional to the font size. That is, small font sizes allow more lines and characters per inch than large font sizes. Xerox 9700 fonts range from 4 to 24 point characters. Within the Xerox 9700 the number of lines per inch divided into 72 will give the approximate point size of the characters of the font (i.e., 8 lpi is 9 points).

As listed in the Xerox 9700 Font Users Guide (Publication No. 91 00 03) standard fonts are available in a variety of point sizes, character spacing (fixed and proportional), line spacing, character orientation (landscape or portrait), and styles (bold, script, etc.). Typical electronic form and variable data fonts are as follows:

- Univers
- Press Roman
- OCR A readable
- OCR B readable
- Sanserif based on Xerox 1200 Computer Printing System
- Serifed typewritier style

Special fonts may be created (or digitized) at the Xerox Digitization Center upon user's request. Fonts created at the Xerox Digitization Center are output on magnetic tape in a binary format. These fonts may be loaded from tape to the Xerox 9700 disk by entering the COPY command via the keyboard (refer to "Copying a File", Chapter 7).

## Font Usage

The fonts to be utilized in a print job are specified by the PDE command in the Job Descriptor Entry (see Chapter 5). The fonts are identified by the name under which they have been previously cataloged. In addition, if a form is required by a particular print job, additional fonts may be specified within the forms definition (see Forms Creation Guide, Xerox Publication No. 91 00 01).

During the printing of a job, any font from the list of fonts on the PDE command may be selected on a character-to-character basis. If different fonts are specified on the same print line, the largest line spacing value associated with those fonts will be used to determine the position of the print line relative to the previous line.

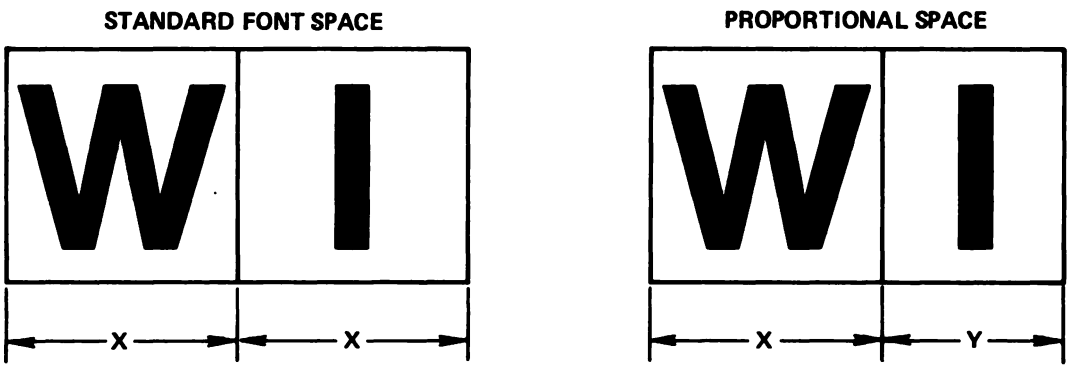
## Font Data

Font data consists of encoded bit matrices of dots which represent each character or graphic symbol in the font. Also associated with each font are optimal line spacing and character spacing parameters, and centering information. When a landscape character is printed, it is centered in its character cell horizontally; portrait characters are aligned vertically.

Font characters may also be combined to create logos. The largest character which will be created is .85" x .85". Therefore, logos which exceed this size are created from a series of font characters. Information carried in the logo file specifies the relative placement of the characters so that the user may treat the logo, as well as other complete figures (i.e., signatures), as a single character.

## Proportional Spacing

Provisions are made for proportionally spaced printing as an option. This requires a special font for each proportional style, size and orientation. Care must be taken by the customer when using a proportionally spaced font, to preclude form line data from interfering with the variable data, and to stay within character limitations stated above.



# 12. ON-LINE PRINTING SYSTEM

## Introduction

This chapter describes various features of the 9700 On-Line Printing System and how these features are made available to the user.

The 9700 on-line system emulates an IBM 3211 Printer/3811 Printer Control Unit. It may be attached to a Byte Multiplexer, Block Multiplexer or Selector Channel of an IBM 360 Computer (Models 22, 30 and up), or IBM 370 Computer (Models 135 and up). The 9700 on-line system operates in either byte multiplexer or burst mode. No programming changes to the IBM operating system software are required, providing a 3811/3211 has already been "sysgened" into the system.

The 9700 on-line system is an integrated hardware/software system consisting of a xerographic printer, laser imaging unit, an output stacker, host input interface, a disk subsystem and a system controller. The 9700 operating system software resides in the system controller.

## Description of System

The 9700 on-line system shares most of the same hardware with the off-line system. The common hardware elements are:

- Xerographic printer
- Laser imaging unit
- Output stacker
- Disk subsystem
- Keyboard/display unit
- System controller

The major difference is that input data is received through a direct interface to an IBM host computer. The 9700 on-line system interface is via the host I/O channel. The device is an On-Line Interface (OLI) unit.

## System Features

The general features of the 9700 on-line system are listed below. For a detailed description of the 9700 on-line system's processing of each 3211 command, see "Processing of 3211 Commands".

- Accepts and prints data in a manner which emulates the IBM 3211/3811.
  - Responds to 3211 diagnostic commands (as documented in the 3211/3811 Component Description Manual) which are appropriate to 9700.
  - The 3211 Universal Character Set Buffer (UCSB) feature is supported. The UCSB is used by 9700 software as a basis for generating folded and unfolded translation tables. See section "Universal Character Set Buffer" in this chapter for further details.
  - The 3211 Forms Control Buffer (FCB) feature is supported. See section "Forms control Buffer" in this chapter for further details.

## Software Features

### Differences from 9700 Off-Line System

The 9700 on-line system software is an operating system called the 9700 Operating System Software (OSS). One of the functions of OSS is to accept print data from the host computer and process it through the xerographic printer and stacker. OSS is functionally divided into tasks (as illustrated in Figure 2-3). The Executive passes logical control between INPUT and OUTPUT processing. On-line INPUT processing differs from off-line INPUT processing in the physical device handling (channel versus tape) and in the emulation of the 3211 printer characteristics. Basic off-line functions of INPUT processing are augmented by the following functions for on-line operations:

- Handles all interactions with the On-Line Interface (OLI) hardware.
- Builds translate tables based on the contents of the Universal Character Set Buffer (UCSB) and the current FOLD/UNFOLD command in effect. Default is an UNFOLDED translate table with all characters printable.
- Processes FCBs received from the host to redefine channel to line number assignments and set margins based on the Print Position Indexing Byte.

### PDL Commands

Appendix A contains a summary of PDL commands available for both on-line and off-line modes. In this appendix each command is defined as available for on-line or off-line (or both). PDL commands which are unique to on-line (or different than their off-line counterparts) are defined in Table 12-1 with more detailed information in the following paragraphs. Commands common to both on-line and off-line are referenced to in Appendix A. These commands are ACCT, CATALOG, CME, CODE, END, IDEN, JOB, MESSAGE, OUTPUT, PDE, ROUTE, and SYSTEM.

### Creating a JDL/JDE

An on-line Job Description Library and its Job Description Entries are created by the user according to the same rules and syntax as described in Chapter 3. The difference between an on-line and off-line JDL/JDE is in the PDL commands that are available to each mode. The one PDL command that must be specified to define an on-line system is VOLUME HOST = IBMONL. The default on-line PCCTYPE is IBM3211. Other defaults for on-line are defined in Table 12-1 and Appendix A.

### DJDE Processing

Printing is controlled through parameters contained in an OSS level control block called the Job Control Block (JCB). This JCB is built from the user defined JDE and may be dynamically modified by Dynamic Job Descriptor Entries. DJDEs are user created and are processed by OSS as part of the print data from the host machine. Chapter 6 defines their format and the PDL commands necessary to let the system know they will be part of the input data. Appendix A identifies those PDL parameters which may be overridden by DJDEs.

### FCB and UCSB

On-line users may optionally restrict the Forms Control Buffer (FCB) and Universal Character Set Buffer (UCSB) information from modifying the JCB. The user may control the updating of the JCB by accepting or rejecting the normal processing of host-transmitted FCB and UCSB input. Options to the LINE command which control FCB and UCSB processing are FCB = $\left\{\begin{matrix}\text{IGNORE}\\\text{PROCESS}\end{matrix}\right\}$ and UCSB = $\left\{\begin{matrix}\text{IGNORE}\\\text{PROCESS}\end{matrix}\right\}$. See Table 12-1 for more details.

### Print Positioning

The control of line positioning on the page is determined by the various JDE parameters. The user has control of font selection and line spacing. The BEGIN keyword of the PDE command specifies the location of the starting print line for each logical page on a physical page. Using this capability, print data may be located starting at any location on a physical page without requiring a change in FCB. The PDE command is discussed in section "Page Descriptors" of Chapter 5.

### Overprinting

The keyword OVERPRINT of the LINE command allows specification of the PRINT2 option which will cause OSS to overprint the first two overprint lines in a series of consecutive overprint lines. The remaining overprint lines are ignored under this option. The syntax of this command is defined in Table 12-1.

### Copy Modification Entries

CME statements may be specified in a JDE, but the associated copy range of the MODIFY left/right part (OUTPUT command) will be ignored. This means, for example, that if six copies of a report are desired and one CME is to apply to copies 1-4, and another CME is to apply to copies 5-6, then the report should be transmitted twice - the first time with a copy count of 4 and CME1 specified and the second time with a copy count of 2 and CME2 specified.

### Report Separation

An off-line job or report is defined in terms of end-of-files on a tape or by special processing criteria. For the on-line stream, report separation is defined in terms of banner page detection. Banner page criteria, which will vary from site to site, are user specifiable. Logically separated reports will be physically offset from one another in the output bins. Three PDL commands are used to control on-line report separation. They are BANNER, TABLE and CRITERIA which are defined in Table 12-1. Examples of their usage are as follows:

Example 1:
The following PDL statements define the offset criteria for a job stream with two trailer pages and one header page. A character string of 120 asterisks beginning in print position one of line 66 identifies each banner page.

```
T1: TABLE CONSTANT = (120)'*';
C1: CRITERIA CONSTANT = (0,120,EQ,T1), LINENUM = (66,1);
    BANNER TEST = C1, HCOUNT = 1, TCOUNT = 2;
```

Example 2:
If a job stream has two trailer pages with the character string DATE = mm/dd/yy appearing on line 10 or 11 and beginning in print position 20, the offset criteria could be coded as follows:

```
T2: TABLE CONSTANT = ('DATE = ::/::/::'), MASK = (':');
C2: CRITERIA CONSTANT = (19,13,EQ,T2), LINENUM = (10,2);
    BANNER TEST = C2, HCOUNT = 0, TCOUNT = 2;
```

The left part "MASK =" is used to specify a masking character. The left part "CONSTANT =" is used to specify both the constant data to be compared and the character positions which contain variable data which are not to be compared. In the above example, position 1 through 5 (DATE =) as well as positions 8 and 11 (the /'s) of the specified character string will contain fixed data. Positions 6, 7, 9, 10, 12 and 13 which are identified by the mask character (:) will contain variable data and will not be compared.

Table 12-1. On-Line PDL Commands*

| Command | Left Part | Right Parts | Default | Comments |
|---|---|---|---|---|
| ac: TABLE | CONSTANT = | (sc, . . . sc) | none | The TABLE statement is used to build a table of constants. The table identifier (referenced in the CRITERIA statement as "id") is "ac". See section "TABLE Statement" in Chapter 4 for further details. |
| | MASK = | ('char') | none | Defines a masking character. Character positions in the string constant (as defined in the left part "CONSTANT =") which contain the mask character ('char') are ignored during comparison testing for the banner page. |
| ac: CRITERIA | CONSTANT = | (offset, length, { EQ / NE }, id) | none | The CRITERIA statement is defined in Chapter 4 (section "CRITERIA Statement") except for the LINENUM left part. |
| | LINENUM = | (init, count) | all lines | The left part LINENUM specifies the range of lines on which to apply the BANNER test. If not specified, the range is all lines.<br>"init" is the initial line number.<br>"count" is the number of lines to test.<br>The test is satisfied if the criteria is met for any line within the range. |
| ac: CRITERIA | CHANGE = | (offset, length, NE, LAST) | none | The CRITERIA statement is defined in Chapter 4 (section "CRITERIA Statement") except for the LINENUM left part. |
| | LINENUM = | (init, count) | all lines | The left part LINENUM specifies the range of lines on which to apply the BANNER test. If not specified, the range is all lines.<br>"init" is the initial line number.<br>"count" is the number of lines to test.<br>The test is satisfied if the criteria is met for any line within the range. |

* Includes commands unique to on-line. Commands common to both on-line and off-line are defined in Appendix A. These commands are ACCT, CATALOG, CME, CODE, END, IDEN, JOB, MESSAGE, OUTPUT, PDE, ROUTE, and SYSTEM.

Table 12-1. On-Line PDL Commands* (cont.)

| Command | Left Part | Right Parts | Default | Comments |
|---|---|---|---|---|
| BANNER | TEST = | $\left\{ \begin{matrix} id_1 \\ (id_1, \left\{ \begin{matrix} AND \\ OR \end{matrix} \right\}, id_2) \end{matrix} \right\}$ | none | Defines the test expression for detection of a banner page. $id_1$ and $id_2$ are identifiers for either change-mode or constant-mode CRITERIA tables.<br>If only $id_1$ is present, then the test is satisfied if the record satisfies the criteria in $id_1$. If $id_1$ and $id_2$ are specified and if the keyword AND is coded, the test is satisfied only if the record satisfies the criteria in both $id_1$ and $id_2$. If the keyword OR is coded, the test is satisfied if the record satisfies the criteria in either $id_1$ or $id_2$.<br>If the test is satisfied, the page containing the record tested will be considered a banner page. |
| | HCOUNT = | value | 0 | "value" specifies the total number of consecutive header banner pages. |
| | TCOUNT = | value | 0 | "value" specifies the total number of consecutive trailer banner pages. |
| LINE | | | | Same LINE command as in Chapter 5 for off-line. |
| | FCB = | $\left\{ \begin{matrix} IGNORE \\ PROCESS \end{matrix} \right\}$ | PROCESS | The right part allows the user to accept (PROCESS) or reject (IGNORE) the processing of the host transmitted FCB. |
| | UCSB = | $\left\{ \begin{matrix} IGNORE \\ PROCESS \end{matrix} \right\}$ | PROCESS | The right part allows the user to accept (PROCESS) or reject (IGNORE) the processing of the host transmitted UCSB. |
| | PCCTYPE = | IBM3211 | IBM3211 | The right part specifies the carriage control convention for printing. Default is IBM3211 if HOST right part of VOLUME command is IBMONL. |
| | OVERPRINT = | $\left( \left\{ \begin{matrix} PRINT \\ MERGE \\ IGNORE \\ PRINT2 \end{matrix} \right\}, \left\{ \begin{matrix} DISP \\ NODISP \end{matrix} \right\} \right)$ | (PRINT, NODISP) | The right part PRINT2 specifies that up to two consecutive overprint lines will overprint. PRINT, MERGE, IGNORE, NODISP, DISP are defined the same as for off-line in Chapter 5. |
| VOLUME | HOST = | IBMONL | IBMOS | IBMONL defines the host to be an on-line system. Off-line right parts are defined in Chapter 4 (see VOLUME command). |

* Includes commands unique to on-line. Commands common to both on-line and off-line are defined in Appendix A. These commands are ACCT, CATALOG, CME, CODE, END, IDEN, JOB, MESSAGE, OUTPUT, PDE, ROUTE, and SYSTEM.

## Operating the System

Operating the 9700 On-Line System is essentially the same as the 9700 Off-Line System. The operator commands specific to on-line operations are described in Table 12-2. They cause the 9700 to go on-line, to go off-line, and to control various on-line operations. The terms READY and NOT READY refer to emulation of 3211 hardware states.

### Placing the System On-Line/Off-Line

The following procedure places the 9700 on-line.

1. At the 9700 console, issue the ONLINE command.
2. At the host console, issue the VARY ONLINE (or equivalent) command.

The above procedure enables the 9700 to go on-line and assume a NOT READY state. OSS will acknowledge the status at the 9700 console. Similarly, to revert to the off-line mode, the following procedure shall be used.

1. At the host console, issue the VARY OFFLINE (or equivalent) command.
2. At the 9700 console, issue the OFFLINE command.

### Starting an On-Line Job

The START command is used to initiate printing of an on-line print job and to make the On-Line Interface go READY. The START command has the form:

START [[jde] [,[jdl] [,[copies] [,FORM = form]]]]

Parameters of the START command are positional and must be separated by commas. A comma must be entered to replace a parameter that is not specified. Options specified on the START command will override those specified in the job descriptor library. Further details on override parameters are contained in "Hierarchy of Replacement" in Chapter 3.

Options on the START command are as follows:

jde — is a 1-6 character identifier for the Job Descriptor Entry to be used in processing the job. If the "jde" option is not specified on the START command, then the DFLT jde specified within the jdl will be used (see "Default JDE" in Chapter 3).

jdl — is a 1-6 character identifier of a disk file which contains the Job Descriptor Library for the print job. It must be cataloged in the JDL file directory. If the "jdl" option is not specified then the default jdl is used. The default "jdl" is user created with an identifier of DFAULT on the SYSTEM command (see "SYSTEM Command Set" in Chapter 3).

copies — specifies the number of copies of a report to be printed. This option overrides the "copies" value specified in the job descriptor entry, and also overrides values specified in a DJDE record from the host.

FORM — is a keyword that allows the user to specify a form to be used for the job. This option overrides a form specification in the JDE (see FORMS left part of OUTPUT command in Chapter 5). This "form" must be on disk in the FRM directory.

### Ending an On-Line Job

Situations arise when the 9700 system becomes quiescent for lack of data from the host, but there are still some pages of the present report remaining in memory which need to be printed. This condition generally occurs when reports do not contain trailer banner pages indicating an end-of-report. When this condition occurs, the 9700 system must be externally informed, via an ENDJOB command, to print out the remaining pages in memory. At the completion of this operation, the system waits for the next START command.

### Multiple Copies

The "copies" option of the START command has the same meaning in both on-line and off-line modes of operation. The multiple copies option may lead to a situation of which the operator must be aware. The object is to produce multiple copies without retransmission from the host. If the number of pages of a multicopy document is smaller than the capacity of the disk buffer, then no difficulty exists. However, if the document is greater than the disk buffer can hold, then the copies may be generated N pages at a time, where N is the number of pages that the disk can hold at one time. This situation will entail the manual merging of the sets. The operator will be able to specify via system level commands what action is to be taken in the event of the disk saturation condition. The possible actions are as follows:

1. Enter CONTINUE to continue multicopy processing and print the requested copies of the report N pages at a time.
2. Enter PRINT1 to discontinue multicopy processing for this report only, print one copy of the current report, and continue accepting data from the host.
3. Enter ENDJOB to discontinue multicopy processing, print one copy of the current report, and end the 9700 job.
4. Enter ABORT R to abort the current report; resume printing with the next report in multicopy mode.
5. Enter ABORT to go NOT READY, mark the current 9700 job as aborted and purge the current 9700 print file.

### 370 Console Messages Handling

The messages output to the 370 host console contain a class that affects the 3211 emulation. The class contains "MOUNT FORM X ON YY" where YY is the particular 3211. X can imply a change of stock, or a 9700 cataloged form or a multicopy form. The 9700 operator should key-in ENDJOB to print unimaged pages from the last report, set up for the next job based on the message content on the host console, and then enter the appropriate START command to begin the next job.

### Saving On-Line Environment

The 9700 On-Line System will save the current environment in a recovery file on disk in the event of a system failure. Upon a system restart, the system will print any unimaged pages from the disk buffer and restore the most current environment. The 9700 operator will then be able to key-in the ONLINE and START commands to reinitiate printing activities.

### On-Line Only Systems

For systems with only an on-line capability, initializing the disk program, fonts, forms, logos, and JDLs/JDEs, will be accomplished through the use of a host-load utility. This program will be disk resident at installation time and will place OSS files into the file system.

Table 12-2. On-Line Operator Commands

| Command | Function |
|---|---|
| ABORT | Causes the 9700 to go NOT READY, mark the current 9700 job as aborted and purge the current 9700 print file, i.e., any pages remaining unimaged will not be printed. |
| ABORT R | Causes the current report being printed to be aborted. Printing will resume with the next logical report in the current 9700 print file. |
| CONTINUE | Causes the 9700 to go READY for normal operations. |
| ENDJOB | Causes the 9700 to go NOT READY, mark the current 9700 job as terminated and print all remaining unimaged pages. |
| OFFLINE | Causes the 9700 to go off-line. |
| ONLINE | Causes the 9700 to go on-line and NOT READY. |
| RESET | Causes the 9700 to go NOT READY and purge the current print file, i.e., forces all system printing activities to cease. |
| START | Causes the 9700 to go READY and process host commands based upon:<br>• Host-supplied control information (FCB, UCSB)<br>• User-supplied parameters (JDE, JDL, DJDEs) |
| STOP | Causes the 9700 to go NOT READY and to stop accepting all commands from the host except SENSE. |

## Processing of 3211 Commands

While certain minor differences exist between the 3211/3811 and the 9700 on-line system's emulation of this device, the responses are appropriate and cause no programming changes to IBM software. Emulated 3211 command codes are listed in Table 12-3.

The 3211 commands are processed by 9700 on-line as follows:

- All write commands perform the included carriage functions after the line is printed.
- Carriage Control commands are processed exactly as the 3211 does, including the overprint function which will result in overprinted characters.
- The 132 or 150 character per line options are supported. User specifies appropriate fonts in the PDE.
- TEST I/O is processed exactly as the 3211 does.
- NO-OP is processed exactly as the 3211 does.
- LOAD UCSB is processed as follows:
  - The train-image field is used in responding to a CHECK READ command that is not preceded by a DIAGNOSTIC GATE command.
  - The associative field is used to build the translate table based upon the FOLD or UNFOLD command in effect. Default is an UNFOLDED translate table.
  - 3211 Dualing feature is not supported. The bits, if set, will be ignored. Characters to be printed by the 9700 are defined within the fonts.
- FOLD and UNFOLD are processed as the 3211 does.
- BLOCK (Disallow) DATA CHECK is processed as the 3211 does.
- ALLOW DATA CHECK is processed as the 3211 does.
- LOAD FCB is processed as follows:
  - The print position indexing (PPI) byte is processed as the 3211 does.
  - Line spacing control bit at FCB address 1 is ignored. Both 6 and 8 LPI are supported as a function of the fonts specified in the PDE command.
  - The current channel assignment (VFU) table is released, and a new one is built.
- The Diagnostic Commands for the 3211 are emulated as follows:
  - SENSE is processed as the 3211 does.
  - READ PLB is processed as the 3211 does.
  - READ UCSB is processed as the 3211 does.
  - READ FCB is processed as the 3211 does but will not result in forms misalignment.
  - CHECK READ is processed as the 3211 does except that if a parity error occurs, all bytes in the PLB will be flagged as having bad parity. This has no effect on operational software since this command is only used by IBM diagnostics.
  - DIAGNOSTIC WRITE is processed as in the 3211.
  - RAISE COVER is accepted but functionally ignored.
  - DIAGNOSTIC GATE command is functionally equivalent to the 3211.
    - A flag is set for use during the processing of other commands (in particular, 'CHECK READ'). This flag is cleared at the completion of the command following the 'DIAGNOSTIC GATE' command.

### Universal Character Set Buffer (UCSB)

The Universal Character Set Buffer is used by the 3211 to define the order of characters on the print train and the complete set of printable characters available on the train. The buffer is 512 bytes long but the 9700 uses primarily the "associative" field of 64 bytes in buffer locations 448 through 511. In this field, if a designated bit associated with a hexadecimal character value is set to one, the character is interpreted as printable. If the bit is zero, that character is interpreted as not printable.

The 9700 will substitute a blank for any hexadecimal character value whose associative bit is set to 1 but for which it has no printable character equivalent. It is recommended that an installation set the entire associative-bit field to binary 1's, defining all hexadecimal values as printable.

The following table defines the UCSB byte locations by function.

| Location | Function |
|---|---|
| 1-432 | Train image field - used in responding to a CHECK READ command that is not preceded by a DIAGNOSTIC GATE command. |
| 433-447 | Reserved field - ignored by the 9700. |
| 448-511 | Associative field - bits 0-3 of each location are used to define printable characters. |
| 512 | Reserved field - ignored by the 9700. |

Appendix F shows the correspondences between location in the associative field hexadecimal values, characters, and the associative bits that determine the character's printability.

### Forms Control Buffer (FCB)

The Forms Control Buffer (FCB) defines channel positions and forms length. It is analogous to a carriage tape on an impact printer. Each byte in the buffer corresponds to one line on the form.

| Hex Code | Definition |
|---|---|
| 01 to 0C | Channels 1 to 12 |
| 00 | No channel punch |
| 1N | Last line, where N may be zero or a channel code |

Indexing (shifting of print position one) may be initiated when the FCB is loaded. This is defined by a one-byte code preceding the forms definition data. The indexing code should be omitted from the FCB load data if no indexing is required.

| Hex Code | Definition |
|---|---|
| 8N | Shift right N-1 positions |
| 4N | Shift left N-1 positions (the first N-1 bytes of each record will not be printed by the 9700). |

Table 12-3. Emulated Command Codes

| Hex | Function | Hex | Function | Hex | Function |
|---|---|---|---|---|---|
| 01 | Write without spacing | 0B | Space 1 line immediate | 00 | Test I/O |
| 09 | Write and space 1 line | 13 | Space 2 lines immediate | 02 | Read Print Line buffer (PLB) |
| 11 | Write and space 2 lines | 1B | Space 3 lines immediate | 03 | No-op |
| 19 | Write and space 3 lines | 83 | Skip immediate to channel 0 | 04 | Sense |
| 89 | Write and skip to channel 1 | 8B | Skip immediate to channel 1 | 05 | Diagnostic write |
| 91 | Write and skip to channel 2 | 93 | Skip immediate to changel 2 | 06 | Check read |
| 99 | Write and skip to channel 3 | 9B | Skip immediate to channel 3 | 07 | Diagnostic Gate |
| A1 | Write and skip to channel 4 | A3 | Skip immediate to channel 4 | 0A | Read UCSB |
| A9 | Wirte and skip to channel 5 | AB | Skip immediate to channel 5 | 12 | Read FCB |
| B1 | Write and skip to channel 6 | B3 | Skip immediate to channel 6 | 23 | Unfold |
| B9 | Write and skip to channel 7 | BB | Skip immediate to channel 7 | 43 | Fold |
| C1 | Write and skip to channel 8 | C3 | Skip immediate to channel 9 | 63 | Load FCB |
| C9 | Write and skip to channel 9 | CB | Skip immediate to channel 9 | 6B | Raise cover (functionally ignored) |
| D1 | Write and skip to channel 10 | D3 | Skip immediate to channel 10 | 73 | Block data check |
| D9 | Write and skip to channel 11 | DB | Skip immediate to channel 11 | 7B | Allow data check |
| E1 | Write and skip to channel 12 | E3 | Skip immeidate to channel 12 | FB | Load UCSB |

# 13. XEROX 9700/850 COMMUNICATIONS INTERFACE

## Introduction

The Xerox 9700/850 Interface is an integrated hardware-software system which allows the Xerox 9700 Electronic Printing System to serve remote Xerox 850 work stations (Page Display or Display Typewriter) as a high-speed printing device. Communication with a remote 850 can occur concurrently with normal printing operations on the 9700. Data received from a remote station is spooled to the 9700 system's disk for printing at a later time under control of the 9700 operator. The communication software is resident on the 9700 disk and is loaded into main memory on request by the 9700 operator.

## Hardware Requirements

The Xerox 9700/850 system shares most of the same hardware with the off-line system. The common hardware elements are:

- Xerographic printer
- Laser imaging unit
- Output stacker
- Disk subsystem
- Keyboard/display unit
- System controller

In addition to these 9700 hardware elements, the Xerox 9700/850 Interface requires the following:

1. A minimum of 80K of Main Memory must exist on the system, 64K of this must be task memory (see memory organization in Figure 13-1). The additional memory (vs off-line) will allow communications concurrent with printing.

2. An 850 Display Typing System with the Display Typewriter (DT) or Page Display (PD).

3. A single-line, synchronous, communications controller for point-to-point communication over switched or non-switched networks. This consists of a printed circuit board which is equipped with a 25 foot modem cable (supplied by Xerox).

4. A customer-supplied Bell Data Set 201C with Auto-Answer option (or its equivalent).

The following table summarizes 9700/850 hardware communications specifications.

| | |
|---|---|
| Type | Synchronous |
| Speed | 2400 Baud |
| Mode | Half-Duplex |
| Line Protocol | BSC - Based |
| Communications Interface | EIA Standard RS 232-C |
| Modem | Bell Data Set 201C (or its equivalent) |
| Code | ASCII-Based (XWP) |

## System Features

The main task of the 9700/850 Interface is to provide remote Xerox 850 work stations access to a high-speed printing device. In addition to this communication capability, access to a 9700 system offers the 850 user the following capabilities:

- Printed output at up to two pages per second
- Choice of landscape or portrait page orientation
- Font selection similar to that of the 9700
- Electronic forms usage
- Multiple-copy capability
- Option for automatically offsetting copies in output bin
- Automatic printing of message-routing information
- Accounting summary
- Selection of fonts on a character, line, or page basis
- Merging of files for address list/form letter applications
- Concatenation of files into one file for printing.

## Available Documentation

In addition to this chapter and other portions of the 9700 Reference Manual there are several other documents which will be useful in understanding and using the 9700/850 facilities.

| **850—Oriented** | Publication No. |
|---|---|
| 850 Page Display Operator Manual | 9R80088 |
| 850 Page Display Reference Guide | 9R80089 |
| 850 Display Typewriter Communication Manual | 9R80124 |

| **9700—Oriented** | Publication No. |
|---|---|
| Xerox 9700 Electronic Printing System Forms Creation Guide | 90 00 01 |
| Xerox 9700 Electronic Printing System Font Users Guide | 91 00 03 |
| Xerox 9700 Electronic Printing System Operator's Guide | 600P81096 |
| Xerox 9700 Electronic Printing System Forms Description Language Self-Study Module | 500485 |
| Xerox 9700 Electronic Printing System Print Description Language Self-Study Module | 500484 |

| **9700/850 Installation** | Publication No. |
|---|---|
| Program Description - Xerox 9700 Operating System Software | 890125-11 |
| Program Description - Xerox 9700 System Software Tape | 890025-11 |

## Software Modules

The 9700/850 Interface software consists of several enhancements to the basic 9700 Operating System Software (OSS). Most of the changes to OSS are evident in the Communication Word-Processing (CWP) Task, the Communications Interface I/O Driver, and a text processing module which is part of the OSS Input Task.

The function of these software modules is defined in the following section. Their interaction in support of the 9700/850 Interface functions is illustrated in the "System Flow" diagram of Figure 13-1.

### Communications Interface I/O Driver

This module performs physical input/output to/from a remote 850, handles interrupts from the communications controller, processes communication interface errors, and does error logging.

### Communications Word—Processing (CWP) Task

- Allows the 9700 operator to control communication-related activity
- Performs logical I/O to/from a remote 850
- Handles line protocol
- Generates Cyclic Redundancy Check (CRC) and performs error checking
- Manages disk space required for document spooling, job queue building, communication accounting information, and document status files
- Interfaces with the OCS Task to manage the CWP-built job queue
- Monitors status of communication line
- Interprets Comment Line in the first Format Block of an 850 document.

### Text Processing Module

- Processes 850 Format Block parameters including the setting of left and right margins, tabs, and special tabs; line spacing; calculates inter-word spacing values for justification; interprets page-size-information; places header and trailer text; processes DJDE commands from Comment Line data.
- Interprets word processing control codes for centering between margins, optional centering, and column centering; columnar data processing, column underscoring, text underscoring, subscript and superscript printing, paragraph indentation; line termination; page termination; tabbing; flush-right text printing; backspacing.
- Performs font switching.
- Merges files (such as an address list with a form letter).
- Concatenates several files into one file for printing.

## System Flow

Figure 13-1 illustrates the general 9700/850 Interface system flow starting with the transfer of a document from the 850 to the printing of it on the 9700. The basic activities are as follows:

(1) The 9700 operator keys in a command at the 9700 console which loads the Communication Word Processing task into main memory and initiates the communication controller to receive and answer calls automatically.

(2) A document is transferred to the 9700 from the 850. The document is received by the Communication Interface I/O Driver and pre-processed by the Communications Word-Processing task. These modules perform the functions defined in the previous section.

(3) The CWP task spools the document data to a disk file. The CWP task also builds a job queue entry and a document status file entry.

(4) At the 9700 console, the 9700 operator keys in a command to begin printing the document that has been previously cataloged.

(5) The Operator Communication Subsystem (OCS) task requests the Input task be loaded. Input prepares to process the document on disk by loading in its 850 Text Processing overlay.

(6-7) The Text Processing routine reads the file from disk, performs its functions (defined in previous section) and writes its output to a disk file (the System Print File).

(8-9) The Output task is activated, reads the Print File, then prints the document on the 9700.

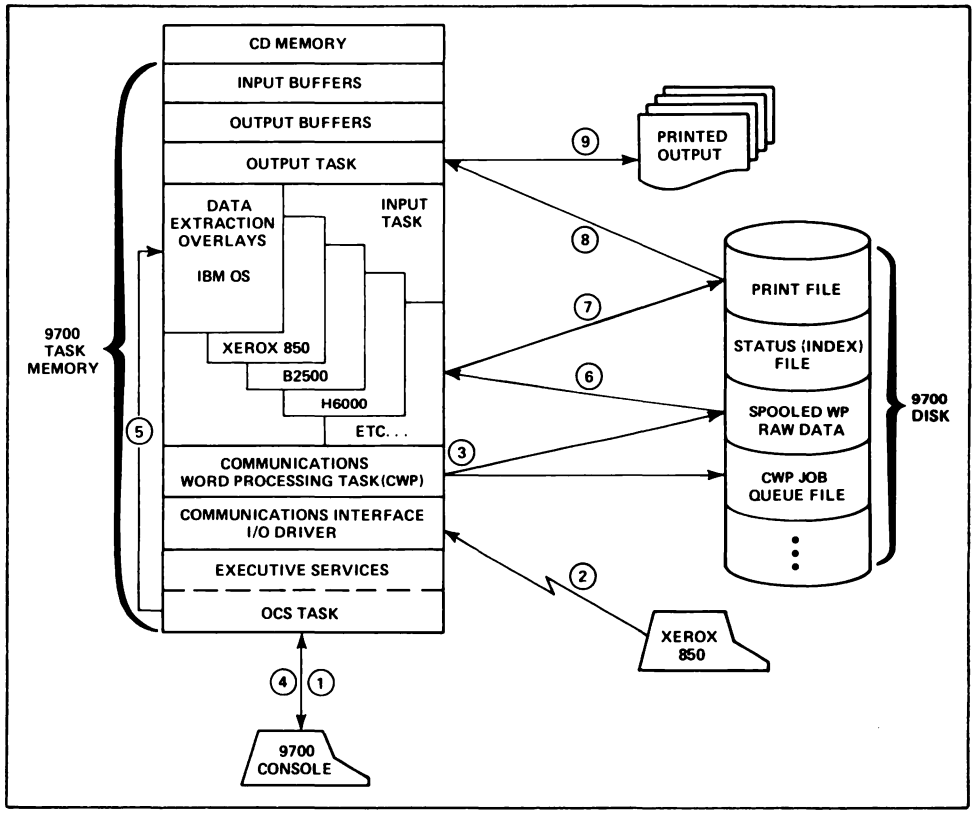



Figure 13-1. 9700/850 System Flow

## 9700/850 File Organization

There are several files on both the 850 and 9700 involved in the printing on the 9700 of an 850 generated document. They are the files created on the 850 (the document to be printed and control files) and there are 9700 files such as fonts, forms, logos, job descriptor library, etc. These file types and their functions are defined in the following section.

### 850 Files

There are 2 types of files created on the 850 which are to be transmitted to the 9700. One is a document file (contains the data to be printed) and the other is a user-created control file (containing commands related to the document to be printed).

The document file to be printed is transmitted from the 850 to a disk file on the 9700 and cataloged in a 9700 file directory (the XCS directory). Once on the 9700, the document file may be initiated for printing by the 9700 operator. After printing, it is deleted unless the user has requested it be saved (via a control file).

When a control file (will be referred to as a Communications Control Language or CCL file) is transmitted to the 9700, its print control parameters are entered into the system. The control file is not saved by the 9700, although its parameters remain in effect until another control file is transmitted. CCL files are transmitted before or after a document file depending on which CCL functions are selected by the user.

A more detailed discussion of CCL files and commands is presented in section "Communications Control Commands".

### 9700 Files

There are several types of files on the 9700 which affect the printing of a document. These files are organized into directories, each directory containing a specific type of file (e.g., file directory type JSL contains the job source language files). Most of the directories that will be used for printing documents from the 850 are the same as those used for on/off-line printing. The following directories are available for 9700/850 usage:

| File Type | Contents |
|---|---|
| FNT | Font data |
| FRM | Form object |
| FSL | Form source language |
| JDL | Job descriptor language object |
| JSL | Job source language |
| LGO | Logo data |
| PDE | Page descriptor object |
| TST | RTEXT object |
| XCS | Xerox Communications Source |
| TSK, SYS, TMP | 9700 System tasks and data files |

XCS Directory

The XCS directory and its data files are unique to 850 usage. A file in this directory will contain the variable data transmitted from the 850 to the 9700 for printing (step 3 in "System Flow"). It can be optionally saved or deleted after it is printed (see SAVE command in section "Communications Control Language Commands").

JSL/JDL Directories

Job Source Language (JSL) files define the printing characteristics for a document. These files are user created (as defined in Chapter 3) and are cataloged in the JSL directory. When these source files are compiled by PDL, the object module is stored in the JDL (Job Descriptor Language object) directory.

FSL/FRM Directories

Electronic forms are created by the user with FSL (Forms Source Language) commands, and then compiled with the FDL compiler. The source files are saved in the FSL directory and the object modules are cataloged in the FRM directory. The FSL files may be created by the user either on the 850 (and then transmitted to the 9700) or using off-line facilities of the 9700 (see Chapter 8). The Xerox 9700 Forms Creation Guide contains details on the syntax and format of FSL commands.

FNT/LGO Directories

The FNT/LGO directories contain the fonts and logos which are supplied by Xerox.

PDE Directory

Page Descriptor Entries (PDEs) specify format information (page orientation, fonts) for pages of a document. The PDE directory contains compiled PDE commands which may be referenced from a JDL or a DJDE. Further details on PDEs is contained in section "Page Descriptors" of Chapter 5.

TST Directory

The TST directory contains precompiled RTEXT commands which may be referenced from a user's Job Descriptor Library. Further details are provided in section "Cataloged RTEXT Files" in Chapter 5.

TSK/SYS/TMP Directories

These directories contain files directly relating to the functioning of the 9700 Operating System Software.

## 9700/850 Control Features

There are several features of the 9700/850 Interface which provide control over the processing and printing of a document. Control is exercised by the operator commands on the 850 and the 9700. Also, control parameters may be set from within a document to be printed, and also via a special file transmitted along with the document.

A brief overview of these control features is presented in this section. References are provided to more detailed discussions of these control features covered in subsequent sections of this chapter and other texts.

### Operational Controls

The 850 and 9700 operators have control over the order in which documents are to be transmitted, initiated for printing, aborted, and several other operational features.

Commands available to operate the 850 are presented in the appropriate 850 Reference Guide (Display Typewriter or Page Display Reference Guide). Commands available to operate the 9700 are presented in section "Operating The System" in this chapter. The 9700 Operators Guide provides further details and examples on the operational features of the system.

### Job Descriptor Library (JDL)

A Job Descriptor Library (JDL) must exist as a disk file on the 9700 in order to print a document. The JDL allows the user to specify output features such as which fonts are to be used, the number of copies to be printed, whether forms are to be merged with variable data, etc. JDL options will remain in effect for the printing of a document unless modified by Dynamic Job Descriptor Entries (see next section) specified within the document. Further details on JDLs are contained in section "Creating a Job Descriptor Library".

### Dynamic Job Descriptor Entries (DJDEs)

DJDEs are control parameters which can control certain output features on a page-to-page basis. Features that may be controlled on a page-to-page basis are the number of copies to be printed, forms to be merged with variable data, and the set of fonts to be used. DJDEs are specified within the document to be printed and override equivalent parameters specified within the JDL. Further details on this feature are contained in section "DJDE Processing".

### Communications Control Language (CCL)

Document-like files are created on the 850 containing CCL commands which transmit printing control information to the 9700. These commands and several examples of their usage are described in section "Communication Control Language Commands".

### Format Block

The Comment Line in the 850 Format Block is an important communication link between the 850 and the 9700. Within the Comment Line must be specified either a START command (and related parameters) or define the "type" of document (i.e., is it a document to be printed or for control) being transmitted. The Comment Line may also contain DJDE commands. Further details on passing control information via the Format Block is contained in section "Specifying Control Information in the 850 Format Block".

## Creating a Job Descriptor Library

A Job Descriptor Library (JDL) and its Job Descriptor Entries (JDE) are created by the user according to the same rules and syntax as defined in Chapter 3. They are typically created with the off-line facilities of the 9700. EDITOR commands (discussed in Chapter 8) are used to create these files, and the PDL compiler is used to compile the source file into its executable object form (see section "JDL Creation and Compilation" in Chapter 3).

### PDL Commands

Table 13-1 contains a definition of the PDL commands available to 9700/850 Interface users. This set of commands is a subset of those available for off-line processing. Note that under each command heading are chapter/page references for further details on the command.

A Job Descriptor Library (a source statement file) may be created on the 850 and transmitted to the 9700. This feature is discussed in section "Creating JSL/FSL Files on the 850".

### Mandatory PDL Commands

There are certain mandatory PDL statements which must be coded for each 850-oriented JDL. These commands are:

```
VOLUME HOST=X850, CODE=ASCII;
BLOCK LENGTH=512;
RECORD STRUCTURE=VB, LTHFLD=1;
```

An example of an 850-oriented JDL is shown in Figure 13-2.

```
SAM850:  SYSTEM;
PDE1:    PDE FONTS=(PR107D,L01ITB);
         VOLUME HOST=X850, CODE=ASCII;
         BLOCK LENGTH=512;
         RECORD STRUCTURE=VB, LTHFLD=1;
JOB1:    JOB;
         OUTPUT FORMAT=PDE1, FORMS=SFORM, COPIES=2;
JOB2:    JOB;
         OUTPUT FORMS=XFORM;
         ACCT USER=(BIN, TRAY);
         END; END;
```

Figure 13-2. Example of an 850-Oriented JDL

## Specifying Control Information in the 850 Format Block

The 850 Format Block is the vehicle used by the 9700/850 Interface to define information related to the type of document being sent to the 9700 and what to do with it once it is received by the 9700 from the 850. A sample Format Block (as it would appear on an 850 screen) is illustrated in Figure 13-3. Accessing the Format Block and its options is defined in the 850 Page Display Reference Guide.

### Comment Line in the First Format Block

The Comment Line (located under the heading "Page Labels" in Figure 13-3) in the first Format Block of every document transmitted from the 850 to the 9700 must contain either a 9700 START command or a TYPE/NAME command set. The Comment Line may also contain Dynamic Job Descriptor Entry commands.

### START vs TYPE/NAME

A START command should be specified in the Comment Line of the First Format Block of all document files to be printed on the 9700 unless they are to be "linked" or "merged" together (on the 9700) for printing purposes. When documents are "linked" or "merged", then the START command must be specified in a CCL command file (not in the Format Block). A START command is also not to be specified in the Format Block of a CCL command file.

The TYPE/NAME command set must be specified in the Comment Line of the first Format Block of CCL-type files. Also, it must be specified for those document files which are to be transmitted to the 9700 and then "linked" or "merged" before printing (as discussed in the above paragraph).

If the format of the START or TYPE/NAME command in the Comment Line of the first Format Block is incorrect, then the 9700 will send an appropriate message to the 850 and then disconnect the line.

Several examples of the START command, TYPE/NAME command set, and CCL commands will be given in the following section.

### START Command in Comment Line

The START command is used to inform the 9700 system which job parameters (JDL parameters) will be used to control the printing of the document. The command form is:

START [jde] ,[jdl] ,,[copies], DISC: document-name

Parameters of the START command are positional and must be separated by commas. A comma must be entered to replace a parameter that is not specified. The "copies" option specified on the START command will override the "copies" parameter specified in the job descriptor library. Options on the START command are as follows:

| | |
|---|---|
| jde | is a 1-6 character identifier for the 9700 Job Descriptor Entry to be used in processing the document. If the "jde" option is not specified on the START command, then the user-specified default jde (DFLT) within the jdl will be used (see "Default JDE" in Chapter 3). |
| jdl | is a 1-6 character identifier of a 9700 disk file which contains the Job Descriptor Library for the print job. It must be cataloged in the 9700 JDL file directory. If the "jdl" option is not specified, then the default is used. The default "jdl" is user-created with an identifier of DFAULT on the SYSTEM command (see SYSTEM Command Set in Chapter 3). |
| copies | specifies the number of copies of a document to be printed. It is an override of the "copies" value specified in the job descriptor entry, and also will override a DJDE "copies" command if present. |
| document name | specifies the name of the 850 document being transmitted to the 9700 for printing. The name should contain no embedded blanks. |

START Command Example

START JDE1,JDL1,,2,DISC:PRTME

The document with the 850 file name PRTME is to be transmitted to the 9700 disk and queued for printing. When it is initiated for printing (by the 9700 operator) it will run under the control of the jde JDE1 (a jde in the jdl JDL1). Two copies of the document will be printed. The START command is shown as it would appear in the Comment Line of the first Format Block of the document PRTME.

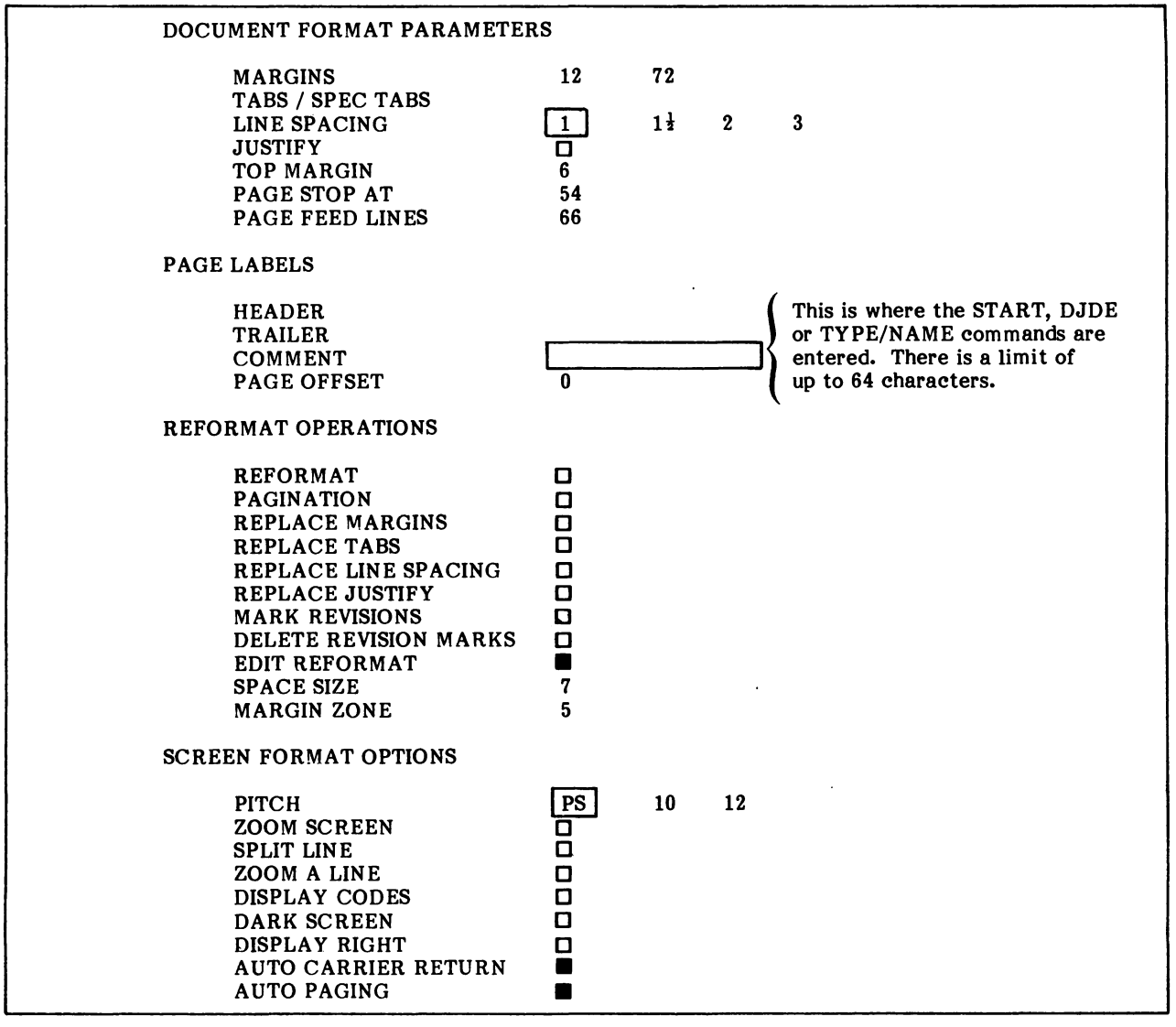



Figure 13-3. Sample 850 Format Block

### TYPE/NAME Command in Comment Line

The TYPE/NAME command set is of the form:

TYPE = document type; NAME = document name

where

| | |
|---|---|
| TYPE | is a keyword |
| document type | is either XCS or CCL. CCL is specified for all CCL-type documents. XCS is specified if the document is to be "linked" or "merged" for printing purposes on the 9700. |
| NAME | is a keyword |
| document name | is the 850-defined XCS or CCL document name. The name should contain no embedded blanks. |

All upper-case words shown above are 'key' words. The '=', ';' and ',' are field separators and must be used as shown above. The Comment Line should not contain any embedded Word Processing control characters except CRN (Carriage Return). The text within the Comment Line can be either in upper case, lower case or both, but it is recommended that all the characters within the Comment Line be in upper case.

TYPE/NAME Example 1

A CCL document (named CCLCMD on the 850) is to be transmitted to the 9700. To identify it as CCL document, the Comment Line in the first Format Block would appear as:

TYPE = CCL; NAME = CCLCMD

TYPE/NAME Example 2

The document SECT1 is one of several documents which are to be printed as one document on the 9700. Each of the individual sections will be transmitted to the 9700 and temporarily saved on the 9700 disk. When all the sections have been transmitted then they are "linked" together by a special type of command document (a Communications Control Language file discussed in the next section).

The TYPE/NAME command would appear in the Comment Line of the first Format Block of the document as:

TYPE = XCS; NAME = SECT1

### DJDE Commands in Comment Line

DJDE commands are specified in the Comment Line of the first Format Block of any page in a document. The format and syntax of these commands is discussed in the section "DJDE Processing" of this chapter. An example is shown in Figure 13-5.

## Communications Control Language (CCL) Commands

CCL commands are used for the purpose of transmitting control information about the printing of a document to the 9700 system. The CCL commands are transmitted to the 9700 as part of a CCL-type document file. Table 13-2 lists the CCL commands available to the 850 user. Note that a few of the commands may only be used if the document is being transmitted from an 850 Display Typewriter system. All others are for both the Page Display (PD) and Display Typewriter (DT).

CCL commands also allow the 9700 to maintain an "index" defining the status of the documents transmitted from the 850 to the 9700. The "index" file and its content are discussed in section "Document Status File" in this chapter.

### CCL Files

The CCL file is created like any other document on the 850, but it only contains CCL commands. Each text line of the file may contain only one CCL command, each of which must conform to the format as defined in Table 13-2 and the syntax defined in the following section.

A CCL file transmitted to the 9700 is not saved on the 9700 disk nor is any entry made in the "index" file concerning it. If there is a START command within the CCL file, then an entry is made in the Job Queue file (a file of START command information concerning all documents queued to be printed).

### CCL Syntax

All "key" words of the CCL commands of Table 13-2 are shown in upper case. Parameters which are to be filled in by the user are shown in lower case. The "key" words and parameters in the CCL command line can be in lower case, upper case or both. The '=' and ',' are field separators and must be specified as shown. There should be no embedded Word Processing codes (except those used with FONT CODE and MERGE commands) in the CCL command line, and all command lines must terminate with a Carriage Return code (CRN, PCR, PSC, SCR).

If an error is detected during the processing of the CCL file, then an appropriate message is transmitted to the 850, and then the 9700 will disconnect the line. The procedure for the 850 operator under these circumstances should be to:

1. Examine message received from the 9700.
2. Make the necessary correction in the CCL file.
3. Retransmit the CCL file.

### Types of CCL Commands

The following section will discuss the two basic types of CCL commands. Examples are provided along with suggestions on their usage.

"Communication Session" Related CCL Commands

This type of CCL command is used to communicate information between the 850 and the 9700, and does not have any effect on the printing of a document. These commands are:

TEL
850 ID
CLEAR INDEX ENTRY
C

CCL FILE Example 1

CCL file used when initiating a communications session.

```
TEL = 213/679-4511
850 ID = MY850
```

The Comment Line of the first Format Block of this CCL file must have a TYPE/NAME command set. For example:

```
TYPE = CCL; NAME = SIGNON
```

CCL FILE Example 2

CCL file used to clear 'index' entries.

```
TEL = 213/679-4511
850 ID = MY850
CLEAR INDEX ENTRY
```

It is not necessary to define TEL and 850 ID commands in all CCL files; they are required only in the first CCL file of a communications session. However, it is strongly recommended that these two CCL commands be specified in all CCL files.

"Job" or "Document" Related CCL Commands

The function of this type of CCL command is to allow the user to set parameters that will affect the format of the output printed on the 9700. As will be illustrated in the examples below, some of the CCL commands must reside in CCL files which precede (i.e., is put into the transmit queue on the 850 ahead of) the document they reference or apply to and some must follow. Commands which must be in a CCL file which precedes a document are:

SAVE
850 PITCH
FONT CODE
TOP MARGIN
PAGE OFFSET
POINT SIZE
HEADER FONT INDEX
TRAILER FONT INDEX

CCL FILE Example 3

CCL file used to "save" a file on the 9700 disk.

To "save" a file on the 9700 disk (after it has been printed), two files need to be queued for transmission to the 9700 in the following order.

1. CCL file with SAVE command
2. Document to be printed (and then saved)

The CCL file would contain the following:

```
TEL = 213/679-4511
850 ID = MY850
SAVE = MYFILE
```

The Comment Line of the first Format Block in the document to be saved would contain the following:

```
START JDE1,JDL1,,10,DISC: MYFILE
```

Care should be taken in using the SAVE command. If too many files are saved then eventually a "disk full" condition on the 9700 will occur.

CCL FILE Example 4

CCL file used for defining CCL commands which specify print parameters for all the documents following this CCL file.

```
TEL = 213/679-4511
850 ID = MY850
850 PITCH = PS
FONT CODE = 'CODE + 3'
POINT SIZE = 12
HEADER FONT INDEX = 2
TRAILER FONT INDEX = 3
```

It should be noted that the parameters defined by the above CCL commands will apply to all the documents transmitted following this CCL file during the current communications session. If any one of the above parameters is to be different (e.g., HEADER FONT INDEX = 1) from one document to the next, then the "new" value for that parameter needs to be defined in another CCL file which precedes (in the transmit queue) the document(s) to which the "new" value applies.

During a communication session, if an "850 ID" command is specified which is different than a previous one, then all CCL parameters are reset to their default state.

CCL FILE Example 5

Same as Example 4 except different values for HEADER FONT INDEX and TRAILER FONT INDEX are to be used.

```
TEL = 213/679-4511
850 ID = MY850
HEADER FONT INDEX = 1
TRAILER FONT INDEX = 4
```

Commands that would reside in a CCL file which must be queued for transmission (on the 850) to the 9700 after the document which is to be printed are:

```
LINK/START
MERGE/START
DELETE
```

CCL FILE Example 6

CCL file used for defining LINK and START commands:

```
TEL = 213/679-4511
850 ID = MY850
LINK = DOC1,DOC2,DOC3
START JDE1,JDL1,,2,DISC:DOC1
```

The following things should be kept in mind when creating this type of CCL file.

- The document name (after DISC:) specified in the START command must be the same as the first document name in the LINK command.
- The START command must follow the LINK command in the CCL file.
- The documents referenced in the LINK command must have been transmitted to the 9700 prior to transmitting this CCL file.
- The Comment Line of the first Format Block of each of the documents referenced in the LINK command must be of the format:

  TYPE = XCS; NAME = document name

The START command in a CCL file should only be specified when a LINK or MERGE command precedes it.

CCL FILE Example 7

CCL file used for defining MERGE and START commands

```
TEL = 213/679-4511
850 ID = MY850
MERGE = ALIST,FLETER, ↑↓
START JDE1,JDL1,,50,DISC:ALIST
```

The following things should be kept in mind when creating this kind of CCL file.

- The first file-name specified in the MERGE command must be the same as the file-name (after DISC:) specified in the START command.
- The merge code ( ↑↓ ) must be specified.
- The ALIST and FLETER documents must have been transmitted to the 9700 prior to transmitting this CCL file.
- The Comment Line of the first Format Block of these two documents must be of the format:

  TYPE = XCS; NAME = document name

- The order of the parameters specified in the MERGE command line must be: 1) Name - Address List 2) Form Letter and then 3) Merge Switch Code

CCL FILE Example 8

CCL file used for deleting a file saved earlier on the 9700 disk via a SAVE command.

```
TEL = 213/679-4511
850 ID = MY850
DELETE = MYFILE
```

## DJDE Processing

### Introduction

Printing of a document on the 9700 is controlled through parameters contained in a 9700 OSS level control block called the Job Control Block (JCB). This JCB is built from the user-defined JDE and may be dynamically modified by Dynamic Job Descriptor Entries. DJDEs are created by the user within the 850 document (in the Comment Line of the first Format Block of a page) and are processed by the 9700 system during the printing of a document.

DJDE commands for 9700/850 usage are a subset of the standard 9700 DJDE set as defined in Chapter 6. The 9700/850 DJDE commands are COPIES, FORMS, FORMAT and END. The syntax and meaning of these DJDE left/right parts are defined in Table 13-3.

### DJDE Creation

DJDE records are created by the user on the 850 as part of the document to be printed. They must be part of the Comment Line in the first Format Block of a page (see previous section "Format Block" for further details). The DJDE record may be up to 64 characters in length, and there may be only one record per page of the document.

### DJDE Considerations

The following points should be considered when preparing DJDE records on the 850.

- The Format Block with a DJDE must be located at the Top-of-Page (i.e., the initial Format Block paragraph).
- DJDE records should be entered into the document after the formatting of the document is complete.
- The DJDE record may contain more than one left/right part except for the case of a comment, which must be the only DJDE information in that record. Each left/right part within a record is separated from the next left/right part by a comma.
- A DJDE record cannot be greater than 64 characters.
- The DJDE "END" command must follow the last left/right part in a record; the END command itself must be terminated with a semi-colon/blank character pair.
- The IDEN command (defined in Chapter 6) which is used in identifying DJDEs for standard 9700 processing is not used for 850 DJDE processing.
- If DJDEs are to be specified in the Comment Line after a TYPE/NAME command set or a START command, then a semi-colon must be inserted between them. For example:

```
START JDE1, JDL1,,2,DISC:DOC1; FORMS = X2, END;
TYPE = XCS; NAME = BILL; COPIES = 3, END;
```

### DJDE Example

Figure 13-5 illustrates the way in which DJDEs may be used in a document.

| Document Page | Command in Comment Line of First Format Block of Page |
|---|---|
| 1 | START JDE1, JDL1,,,DISC:SAMJOB |
| 2 | FORMS = GRID, FORMAT = X2, END; |
| 3 | (no DJDE specified) |
| 4 | FORMS = NONE, END; |
| 5 | FORMS = LPAGE, FORMAT = X3, END; |

Figure 13-5. DJDE Example

## Operating The System

This section describes the procedures used to control the 9700 communications software and what the 850 operator has to do to communicate documents and job-related information to the 9700. The 9700/850 Interface communications software is resident on the 9700 disk and is loaded into main memory on request from the 9700 operator (see step 1 of "System Flow"). After this is done, communication exchanges can occur concurrently with the normal printing activity on the 9700.

### 9700 Operator Commands

The 9700/850 Interface communications software (specifically the Communications Word-Processing Task or CWP) is loaded into main memory by typing the command CWP on the 9700 operators console. After the CWP task has been loaded, the message "OS1000 READY FOR COMMANDS" is displayed on the 9700 console.

At this point it is possible for CWP communications commands (see Table 13-4) to be invoked at the 9700 operators console. There are three groups of communications commands: commands to control printing 850 documents, commands to control communicating with the 850, and utility commands to let you look at the 850 job queue, check communications status, etc. All commands begin with the three letters CWP followed by a comma and then the command word. The CWP keyword tells the 9700 that the command refers to a communications job, and not to on/off-line print jobs. CWP commands have the general format:

CWP, command name [,command parameter(s)]

where

| | |
|---|---|
| CWP | is a keyword that identifies the command as a CWP-type. |
| command name | names the function to be performed. The functions available are defined in Table 13-4. |
| command parameter(s) | names a parameter or parameters associated with a specific 'command name'. Note that not all commands have 'command parameter(s)'. |

Each time a command is entered, the system will respond by displaying a message on the 9700 operators console. If the commands EXIT, DISC or RVI are invoked by the 9700 operator and the line is in "STANDBY HOLD STATE", then it could take up to 10 seconds before a response to the keyin is displayed on the 9700 console screen. A complete list of informational and error type messages is discussed in the following section. Examples of command/response interactions are listed in Table 13-5. The 9700 Operator's Guide should be consulted for further details on the use of these commands.

## Operational Flow

Figure 13-6 illustrates the basic operational procedure for transferring a document from the 850 and printing it on the 9700. The commands required to perform the 850 functions are defined in the 850 Page Display Reference Guide.

| 850 Procedure | 9700 Procedure | Comment |
|---|---|---|
| Create CCL and document files | Create JDL/JDE, FSL files (if required) and compile them. | |
| | "CWP" command invoked by operator | CWP task is loaded into 9700 system controller memory. |
| Queue the CCL and document files on the 850 | | |
| | "CWP, RECV" command invoked by operator | This command prepares the 9700 to receive phone calls automatically. |
| Phone call to 9700 data set by 850 operator | | After call received by the data set on the 9700 then message on 9700 console will be:<br>CW1776 STANDBY MODE - LINE IN 'HOLD' STATE |
| Transmit CCL file | | Sets up CCL functions required by the user. |
| Transmit document | | CWP task on 9700 spools 850 file to a 9700 disk file in the XCS directory. Name given to file by 9700 will consist of first 3 characters of 850 name plus 3 digits. Assume for this example the 850 filename was SAMJOB. Then the name given to the file will be, for example, SAM000. |
| | "CWP.STA,ALL" command invoked by 9700 operator | Printing of documents initiated on 9700. This command may be invoked at any time after an XCS file has been built. It is also possible to start printing by specifying an individual filename (i.e., in this example "CWP,STA,SAM000.XCS"). |
| | | When printing has completed the message:<br>OS1030 JOB **** HAS COMPLETED PRINTING<br>will be displayed on the 9700 console. |

Figure 13-6. 9700/850 Operational Flow

## 9700/850 Information/Error Messages

There are two types of messages provided by the 9700/850 Interface software. One type of message is sent to the 9700 console in response to CWP commands (or to inform the operator of an error condition). The other type of message is sent to the connected 850 when an error has occurred during communications between the 850 and the 9700.

### Messages to 9700 Console

Messages sent to the 9700 operators console originate from either the CWP task or one of the other 9700 tasks (e.g., Input, Output, etc.). Messages will be of the form:

{CWNNNN / OSNNNN} Message

where

CW — denotes it is a CWP task message. These types of messages are listed in Table 13-7. The "levels" of message severity (as defined in Table 13-7) are the same as the levels for off-line messages.

OS — denotes the message originates from a task other than CWP. These types of messages are listed in Appendix B except for messages unique to CWP processing (listed in Table 13-7).

Message — status information related to printing of a document.

### Messages Sent To a Connected 850

If an "error" condition is detected by the CWP Task during communication, CWP transmits a file with two lines of information to the transmitting 850 and then disconnects the line. The name of the file written to the 850 disk has the following format:

YXXX

where

Y is a "?" or the first character of the last file received by the 850.

XXX is a three digit number between 001 and 999.

Whenever an 850 is communicating with a 9700 and the line is disconnected then a flashing "STANDBY" message is displayed on the 850 screen. It is strongly recommended that the 850 operator look for messages received from the 9700 under the file name format described above to determine the reason for the line disconnect.

The following conditions will cause the line to be disconnected:

- No Format Block at the start of the file.
- No Comment Line in the first Format Block of the file.
- Comment Line syntax error.
- CCL command syntax errors.
- 'Record' more than 72 characters long in non-XCS files.
- Index File full.
- Job Queue full.
- 9700 Disk error.
- 9700 Disk full.
- OSDS Task was invoked by 9700 operator.
- 9700 operator keyed-in "CWP DISC".
- 9700 operator keyed-in "CWP EXIT".

The first line of this error file shows the time of day along with the file name that was being transmitted to the 9700 (if any). The second line defines the 'error' condition. After transmitting the message, the CWP Task will disconnect the line.

The second line could be any one of the following messages:

*** 'RECORD' MORE THAN 72 CHARACTERS LONG. FILE WILL BE DELETED.
*** NO INDEX FILE.
*** NO COMMENT LINE IN THE FIRST FORMAT BLOCK.
*** NO FORMAT BLOCK AT START OF DOCUMENT.
*** END OF FORMAT BLOCK NOT DETECTED.
*** INDEX FILE FULL - - SEND A 'CLEAR INDEX ENTRY' IN CCL FILE.
*** JOB QUEUE FULL.
*** DATA TRANSMISSION TASK SUSPENDED BY 9700.
CW8450 DISK FULL
CW8400 DISK ERROR - - 'filename.type'
CW0900 CWP TASK EXITING
*** INVALID KEYWORD "command line in error"
*** FILE DOES NOT EXIST "command line in error"
*** NO 'VALUE' SPECIFIED "command line in error"
*** INVALID 'VALUE' SPECIFIED "command line in error"
*** NO FILENAME SPECIFIED "command line in error"
*** INVALID FILE TYPE "command line in error"
LINE IS BEING DISCONNECTED

## Document Status File

A status file (referred to as the "Index" file) is maintained on the 9700 system disk for each 850 operator to receive a status of documents transmitted to the 9700. The "Index" file is created on the 9700 disk based on the "850 ID" command in the CCL file transmitted from the 850. It should be a mandatory operational procedure that a CCL file containing an "850 ID" command be the first file transmitted from the 850 to the 9700 for each communication session.

### Index File Entry

For each document received from the 850, an entry is made in the appropriate Index file to indicate its status. During the processing of CWP-built job queue entries, the entries in the Index file are updated to reflect the latest status of the documents received from the 850. An example of an Index file content is shown in Figure 13-7.

| TITLE | SAVE FILE | STATUS | 9700 MESSAGE |
|---|---|---|---|
| ABCD | NO | PRINTED | OS1030 JOB 0003 HAS COMPLETED PRINTING |
| XYZ | YES | PRINTED | OS1030 JOB 0004 HAS COMPLETED PRINTING |
| JDL850 | YES | COMPILED SUCCESSFULLY | OS1030 JOB 0005 HAS COMPLETED PRINTING |
| DPS | NO | PRINTING | OS1020 JOB 0006 HAS COMPLETED INPUT PHASE |
| JSLTST | YES | RETAINED ON DISK | |
| MANUAL | NO | PRINTED | OS1030 JOB 0007 HAS COMPLETED PRINTING |
| TEST | YES | PRINTED | OS1030 JOB 0008 HAS COMPLETED PRINTING |

Figure 13-7. Example of an "Index" File

### Referencing An Index File

An 850 operator, when requesting the transmission of his/her "Index" file, should specify INDEX as the file name. This is the file name to be used by all 850 operators. The procedure for requesting an Index file is defined in section "Polling a Remote 850" of the 850 Page Display Reference Guide.

## Printing Integrity Information

The user can ascertain information about text processing faults and termination conditions from the accounting page. This information is contained in the accounting page entries "WP EXCEPTION CODE" (which registers text processing faults) and "WP COMPLETION CODE" (which registers job termination conditions).

### Exception Codes

The WP EXCEPTION CODE indicates the non-performance of a word processing operation. When an exception condition occurs, an error character is printed where the fault took place, and the exception code is set. The following exception codes are currently defined:

| CODE | CONDITION |
|---|---|
| 0 | No exception condition (no problem exists) |
| 1 | Justification indicated, but one or more lines could not be justified. |
| 2 | One or more TABs could not be associated with a forward assigned position. Either the user attempted a backwards tab, or there were insufficient assigned positions. A '?' will be inserted. |
| 4 | One or more non-ASCII characters could not be translated. An '*' will be inserted. |
| 8 | One or more page transitions were forced because of pagination or page overflow processing. |
| 16 | One or more centering codes could not be processed because of insufficient room to the left of the center point. |
| 32 | One or more flush right codes could not be processed because of insufficient room to the left of the last character. |
| 64 | Character code exceeded font limit (illegal character). A blank character was substituted for the illegal character. |

It should be noted that combinations of codes can occur. If combinations of codes do occur, then they are added together into a unique code, which must be uncoded by the user. For example, a code of '11' would indicate that '1', '2', and '8' were the decoded exception codes.

### Completion Codes

If Input Processing had to terminate abnormally, the WP COMPLETION CODE (as printed on the accounting page) will indicate the reason for the termination. The following codes are currently defined:

| CODE | CONDITION |
|---|---|
| 0 | Job completed normally. |
| 10 | A DJDE record within a Format Block Comment Line had a syntax error. |
| 20 | A font listed in the PDE font list did not contain the character width table required by the 850 emulation software. This is probably an old font and should be replaced with the current version. |
| 30 | During a job which concatenated two or more files, the link between the two files failed because of insufficient dynamic memory or a 9700 disk read error. |
| 40 | A 9700 disk read error occurred while accessing an XCS-type file. |
| 50 | During a merge job which concatenated two or more LETTER files, the link between two files failed because of insufficient dynamic memory or a 9700 disk read error. |
| 60 | A 9700 disk read error occurred while processing a LETTER document for a merge job. |

Only one completion code will be printed on the accounting page.

## 9700 Disk Full Recovery Procedures

When it is determined by the CWP task during a communications session with an 850 that there is no more space available on the 9700 system disk (in which to create another entry in an index file or Job Queue file entry; or insufficient file space exists on the 9700 disk to accommodate another 850 document file), then the CWP task will transmit two lines of information to the connected 850. After this "message" has been transmitted, the CWP task will disconnect the line and will get ready to answer another call.

The first line of the 9700-transmitted message shows the time of day and the name of the file which was being transmitted from the 850 at the time of the disc full condition. The second line describes the reason for disconnecting this line. The following messages describe the reason for disconnecting the line for those conditions discussed above.

*** INDEX FILE FULL - - SEND 'CLEAR INDEX ENTRY' IN CCL FILE
*** JOB QUEUE FULL
CW8450 DISK FULL

The action to be taken by the 850 operator after the line has been disconnected for the above three conditions is discussed below. The first 4 steps in the recovery procedure are the same regardless of which problem has occurred.

1. Call the 9700 again and establish the line.

2. Transmit a CCL file containing the 850 operator's ID and telephone number.

3. Request the transmission of Index file from the 9700.

4. Upon receiving the Index file from the 9700, the 850 operator should display it and may copy it under another "title".

### Index File Full Condition

Steps 1 through 4 as above.

5. Transmit a CCL file to the 9700 containing the 850 ID and CLEAR INDEX ENTRY commands and then continue with the normal procedure of transmitting more documents to the 9700. The 9700 upon receiving the CLEAR INDEX ENTRY command will clear those entries in the Index file for which a SAVE command has not been specified.

### Job Queue Full Condition

Steps 1 through 4 as above.

5. Call the 9700 operator (on a phone other than the modem phone) and request that the CWP-built Job Queue (a list of START commands for all the currently unprinted documents) be initiated via a "CWP START ALL" command as soon as possible.

6. After a while request the transmission of the Index file to check the status of the jobs in its queue.

7. If all the documents have been printed, then continue with the normal procedure of transmitting more documents to the 9700. Otherwise repeat step 6.

### Disk Saturation Condition

Steps 1 through 4 as above.

5. Transmit a CCL file to the 9700 containing the DELETE = filename1, filename2... command and then continue with the normal procedure of transmitting more documents to the 9700. The files to be deleted (filename1, etc.) are those files which were previously SAVE'ed after being printed.

   The 9700 upon receiving the DELETE command in the CCL file will delete the files as requested thus releasing disk space on the 9700 disk for future use. Whenever a file is deleted by CWP upon receiving a DELETE command in the CCL file, an entry becomes available in the Index file.

## System Restrictions

The Xerox 9700/850 Interface software emulates a communicating Xerox 850 in the receive mode with the following minor restrictions:

- Transparency mode of communication is not supported.
- System will not execute 850-generated "program" documents on the 9700. They can be transmitted to the 9700 and printed as any other document.
- Selection of "Code Print" on the Format Block (PRINT OPTIONS Line) is not supported. Thus, a printout of system codes associated with a document will not be printed on the 9700.
- "Code+9" is not supported (Bolding). This feature is discussed in section "Bold" (Code+9) in the 850 Page Display Reference Guide. The same effect as "Bolding" can be produced, however, by selection of an appropriate 9700 font.
- Minimum 850 firmware levels required for 9700/850 communications are as follows:
  Communications (DT and PD) level 4.0
  Text Processing ROM (DT) level 4.3
  Text Processing ROM (PD) level 2.6

## Saving Files on the 9700

Files created for 9700/850 usage may be saved (and restored) using the standard 9700 file maintenance commands as described in Chapter 7 (see section "File Maintenance"). Assuming a writable tape unit exists on the 9700, the tape must first be initialized with the TAPE VOLINIT command. The files are written to tape with the COPY TAPE WRITE LABEL command. Use the TAPE REWIND command to rewind the tape. If more files are to be added at a later time the tape should not be initialized again. Files may be restored by using the COPY TAPE READ LABEL command.

## Creating JSL/FSL Files on the 850

Job Source Language (JSL) and Forms Source Language (FSL) files may be created on the 850 and then transmitted to the 9700. Each source line must be 72 characters or less. The Comment Line of the first Format Block must be of the form:

TYPE= {JSL / FSL}; NAME=file-name

where

TYPE / NAME — are keywords

JSL / FSL — specifies that the file being transmitted is a Job Source Language or Forms Source Language file. The file will be cataloged in the appropriate 9700 file directory based on this parameter.

filename — specifies the 9700 file name that will be given to the file transmitted from the 850 to the 9700. This name must be 6 characters or less.

After the file has been transmitted to the 9700, it will reside in source form in the JSL or FSL directory. The JSL or FSL file may then be compiled from the 9700 operator's console by typing in the appropriate PDL (see section "JDL Compilation" in Chapter 3) or FSL (see section "FDL Processor" in Chapter 10) command.

NOTE: If a source file of the same name as that specified on the NAME command already exists on the 9700, then the original 9700 file will be written over after the file from the 850 is transmitted. Thus, it is important to determine before transmitting a file what file names already exist in the JSL or FSL directory. JSL and FSL files are automatically saved on the 9700 disk and thus do not require a SAVE command.

It is also possible to specify a PDL/FDL compilation request in the Comment Line of the first Format Block of the JSL or FSL file. The compilation request is to be specified as it would be on the 9700 console.

Example:

The PDL compilation request:

```
PDL SAMJDL,TRAY
```

in the Comment Line of the first Format Block of the JSL source file SAMJDL would compile the JSL statements, catalog the source file in the JSL directory and the object file in the JDL directory.

## Accounting Information

Accounting information is automatically accumulated by the 9700 for each document printed. The PDL command ACCT (see section "PDL Commands" in this chapter) enables the user to request a printout of the accounting summary on a report basis. The system level REPORT USER command provides an installation with a printout of overall system usage statistics.

The USER left part of the ACCT command enables the user to request that an accounting summary be output for each document printed. This summary consists of a single page of information containing job set-up information and counts of processing events. For example, the PDL statement ACCT USER = (BIN, TRAY); requests that the system print an accounting summary page to the bin and sample print tray after a report is printed. An example of an accounting page is illustrated in Figure 13-8.

### Accounting Page Entry Definitions

Many of the entries on the accounting page are the same as those defined for off-line processing in Chapter 5. Entries which differ from those previously defined are:

CALLER ID
The text in this field is taken directly from the CCL command "850 ID =" in the CCL file. This field will be blank if the command is not used.

FILE RECEIVE TIME
Elapsed time (hh:mm:ss) to transmit the document from the 850 to the 9700.

BLOCKS RECEIVED
Number of blocks (512 bytes each) on 9700 disk required for the XCS file.

CRC ERRORS
A count of the number of blocks with CRC (Cyclic Redundancy Check) errors during transmission of document from the 850 to the 9700.

WP EXCEPTION CODE
A "code" defining a processing fault. These codes are explained in section "Printing Integrity Information" in this chapter.

WP COMPLETION CODE
A "code" defining the condition under which the printing of the document terminated. These codes are explained in section "Printing Integrity Information" of this chapter.

CALLER TELEPHONE NO.
The text in this field is taken directly from the CCL command "TEL=" in the CCL file. This field will be blank if the command is not used.

SOURCE FILENAME(s)
The names of disk files transmitted from the 850 to the 9700. Note that disk file names will be 9700 names, thus will be made up of the first 3 characters of the 850 name plus a 3 digit sequence number.

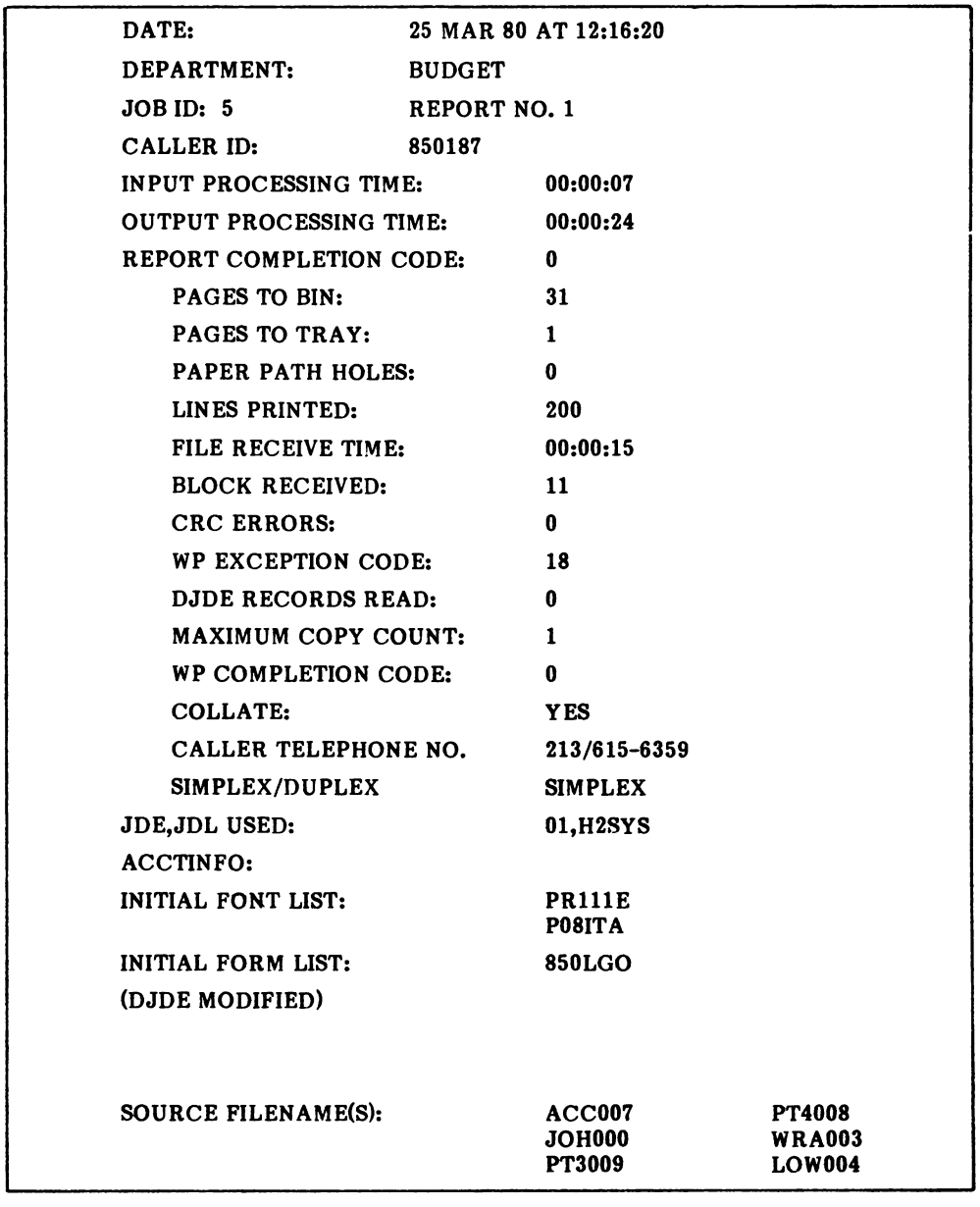



Figure 13-8. Sample Accounting Page

Table 13-1. 850 PDL Commands

| Command | Left Part | Right Parts | Default | Comments |
|---|---|---|---|---|
| ACCT (5-31) | USER = | { NONE / BIN / TRAY / (BIN,TRAY) } | BIN | ACCT command is used to request an accounting summary and to specify where the summary will be delivered. |
| | DEPT = | sc | jdl name | |
| BLOCK (4-11) | LENGTH = | 512 | 1330 | LENGTH = 512 is mandatory. |
| ac: CATALOG; (3-6) | | | | Command used to group statements which are to be included in more than one job descriptor entry. |
| END; (3-7) | | | none | Command used to terminate a job descriptor library. |
| dd: { JOB / JDE } (3-1) | [INCLUDE = | [($id_1$,...,$id_n$) ;] | none | Start of job descriptor entry. INCLUDE specified when using a catalog command set. |
| OUTPUT (5-2) | COPIES = | value | 1 | OUTPUT command is used to specify parameters concerning the format and quantity of the document to be printed. |
| | COLLATE = | { YES / NO } | YES | |
| | COVER = | { FRONT / (FRONT,SEP) / BACK / BOTH / (BOTH,SEP) / NONE } | NONE | |
| | FORMAT = | pde-id | FMT1 | |
| | FORMS = | { form-id / NONE } | NONE | |
| | DUPLEX = | { YES / NO } | NO | |
| | SHIFT = | { YES / NO } | NO | |

Table 13-1. 850 PDL Commands (cont.)

| Command | Left Part | Right Parts | Default | Comments |
|---|---|---|---|---|
| ac: PDE (5-23) | PMODE = | {PORTRAIT / LANDSCAPE} | LANDSCAPE | PDE command is used to describe formatting information for each page of a document. |
| | FONTS = | $\{(f_1 [[,f_2],\ldots]) / ((f_1,s_1) [[,(f_2,s_2)]\ldots])\}$ | NONE | A maximum of 6 fonts may be specified. |
| RECORD (4-14) | STRUCTURE = | VB | FB | STRUCTURE = VB is mandatory. |
| | LTHFLD = | 1 | 0 | LTHFLD = 1 is mandatory. |
| [ac:] ROUTE (5-28) | RTEXT = | {sc / NONE / rtext-id} | NONE | |
| | RFORM = | {form-id / NONE} | NONE | |
| dd: {SYSTEM / JDL}; (3-5) | | | | Command used to identify the beginning of a job descriptor library. |
| VOLUME (4-5 | HOST= | X850 | IBMOS | HOST = X850 is mandatory. |
| | CODE = | ASCII | EBCDIC | CODE = ASCII is mandatory. |

Table 13-2. CCL Commands

| CCL Command | Comments |
|---|---|
| TEL = Area Code/abc-1234 | Twelve characters defining caller's telephone number, e.g., 213/679-4511. Must be specified in the first document transmitted during a communication session. |
| 850 ID = operator ID | Six characters defining 850 operator's ID. Must be specified in the first document transmitted during a communication session. This ID is used for creating an "Index" file on the 9700 disk for the calling 850. |
| 850 PITCH = {value / PS} | This command needs to be specified if the pitch or font used to create a document on the 850 differs from the first font specified in the FONTS option of the PDE command (within the 9700 JDE).<br><br>"value" must be an integer between 0 and 99. |
| CLEAR INDEX ENTRY | Clears "entries" in the Index file on the 9700. Clears only those "entries" for which a SAVE command has not been previously specified. This command should be specified only when the 850 operator does not wish to maintain any "status" of the documents transmitted earlier or when the line has been disconnected by the 9700 due to the Index file being full. |
| MERGE = alist, letter, ↑↓<br><br>START jde,jdl,,copies,DISC:alist | Used to merge letter and address list files.<br>"alist" = name of 850 file which contains an address list<br>"letter" = name of 850 file which contains a form letter<br>↑↓ = required characters to define the merge "switch" code.<br><br>This START command must follow the MERGE command to specify how to print the merge documents. The START command has the same format and parameter definitions as described in section "START Command in Comment Line". |
| LINK = file-id1,file-id2,...,<br>file-idn<br><br>START jde,jdl,,copies,DISC:file-id1 | "file-id's" are names of files that are to be "linked" or concatenated into one file for the purpose of printing. Names should be specified in the order they have to be "linked".<br><br>This START command must follow the LINK command to specify how to print the "linked" documents. The START command has the same format and parameter definitions as described in section "START Command in Comment Line". The document name following the keyword DISC must be the same as the first document name in the LINK command. |

Table 13-2. CCL Commands (cont.)

| CCL Command | Comments |
|---|---|
| FONT CODE = {'Code +3' / 'Code +5' / ↓↑} | This control command defines the font switch code; it could vary from document to document. In any other context, the "Code+" inputs retain their conventional meaning as described in section "CODE + (keys)" of the 850 Page Display Reference Guide. But to invoke a font switch, the font code is repeated a number of times corresponding to the index value of the font in the current PDE-defined font list, e.g., 'CODE+3' 'CODE+3' 'CODE+3' for font 3 in the PDE font list. 'CODE+3' 'CODE+3' 'CODE+3' 'CODE+3' 'CODE+3' for the 5th font in the PDE font list. |
| SAVE = document name | This command is specified only when a document (as referenced by the "document name") needs to be saved on the 9700 disk after all copies for that document have been printed. The CCL file containing the SAVE command must immediately precede in the 850 transmit queue the document which has to be saved. The 9700 system default is to delete a document after printing. |
| DELETE = filename | This command causes 850-named files previously transmitted to the 9700 to be deleted from the 9700 system disk. Several file names may be specified, each separated by a comma. The DELETE command also causes the corresponding entry in the "Index" file to be cleared. |
| C∅ text | This command allows user commentary in the CCL file. A space character must be used to separate the C command from the user-specified text string. The "text" strings are ignored by the CWP task. |
| TOP MARGIN = value<br>(DT only) | This command specifies the line number, given by "value", where the first line of text is to be placed on a page. The default is line number 6. "Value" will override the Format Block specification for top margin. |
| PAGE OFFSET = value<br>(DT only) | This command allows page numbering of a document to begin with any desired number. The number specified by "value" should be one lower than the desired beginning number. The default is 0. |
| POINT SIZE = value | The 9700 'en' space character size corresponding to the specified point size "value" will be used by the 9700 system to calculate physical margin and tab positions on the page. The default space character size is that associated with the first font specified in the FONTS option of the PDE command. This size for a particular font can be obtained from the CHARACTER SPACING entry on a "SAMPLE font" job printout. Refer to the XEROX 9700 Electronic Printing System Font User's Guide for complete space character size definitions. |

Table 13-2. CCL Commands (cont.)

| CCL Command | Comments |
|---|---|
| HEADER FONT INDEX = value | This command allows the user to specify which font in the active PDE font list (within the 9700 JDE) is to be used in printing document Header text. "Value" is the index into the font list. Header text is described in section "HEADER" of the 850 Page Display Reference Guide. |
| TRAILER FONT INDEX = value | This command allows the user to specify which font in the active PDE font list (within the 9700 JDE) is to be used in printing document Trailer text. "Value" is the index into the font list. Trailer text is described in section "TRAILER" of the 850 Page Display Reference Guide. |

Table 13-3. 850 DJDE Left/Right Parts

| Left Part | Right Parts | Description |
|---|---|---|
| COPIES = | number | Specifies the number of copies of a document or page to produce. It takes effect on the document page associated with the Format Block that contains this DJDE command.<br><br>The right part "number" specifies the number of copies which are to be printed. The range for "number" is 1 to 32,767. |
| FORMS = | {form-id<br>NONE} | Specifies a form to be merged on the document pages. It takes effect on the document page associated with the Format Block that contains this DJDE command.<br><br>The right part "form-id" specifies a 1-6 character form file name (may be numeric, alpha, or alphanumeric) which exists on the 9700 system disk. This file is created by compiling a Forms Description Language source file with the FDL system task (see Chapter 10 for further details on FDL).<br><br>The option NONE specifies that no form is to be merged on the document page. |
| FORMAT = | pde-id | Specifies the Page Descriptor Entry (PDE) to be used for formatting control. It takes effect on the document page associated with the Format Block that contains this DJDE command.<br><br>The right part "pde-id" references a PDE which must have been defined previously in a Job Descriptor Library. It may also be a reference to a PDE file separately catalogued in the PDE library on disk. A set of standard "pde-ids" are defined in Table 5-1 (i.e., FMT1, FMT2, etc.).<br><br>Details on the PDE command are discussed in the "Page Descriptors" section Chapter 5. Creating and compiling PDE files with the PDL processor is discussed in "PDE and CME Compilation" in Chapter 3. |
| END;ø | | This command must be used to indicate the end of DJDE information. When the END command is encountered in a DJDE record, the system will apply all DJDE information specified to the document page associated with the Format Block which contains the DJDEs. "END" must terminate with a semicolon/blank character pair. |

Table 13-4. CWP Operator Commands

| Command Name | Command Parameter | Function |
|---|---|---|
| EXIT | none | CWP task will complete input/output for currently active documents and then exit. |
| CONTINUE | none | Takes the telephone line out of a "wait" state. |
| DISC | none | Disconnects the communication line. |
| DELETE | {*,XCS / file-id1,..,file-idn} | Delete all (*) or specified documents (file-id) from the XCS directory. |
| ENDQ | none | Stop printing of all 850 documents after finishing the one currently being worked on. |
| LISTQ | none | Displays on the 9700 console the START commands for all 850 documents received. This is the communications job queue. |
| LISTC | none | Continue LISTQ function after 9700 console is filled. |
| RECV | none | Gets the 9700 ready to receive a call from an 850. |
| RVI | message | Sends "message" to the 850 currently talking to the 9700. |
| START | {ALL / filename.XCS} | Initiate printing of all documents (ALL) received from 850s or a specified one (filename). |
| STATUS | none | Display communications line status (e.g., 'hold' or 'wait'). |
| WAIT | none | Puts the telephone line in 'wait' state. |

Table 13-5. Command/Response Interactions on 9700 Console

| Command | Responses to 9700 Console |
|---|---|
| CWP, RECV | CW1775 STANDBY MODE - WAITING FOR RING |
| CWP, START, filename.XCS | Command line in the requested JQE<br>CW1600 QUEUE IS EMPTY<br>CW2750 DID NOT FIND REQUESTED JOB<br>CW1630 QUEUE ALREADY ACTIVE |
| CWP, LISTQ | Command line in the first JQE<br>Command line in the second JQE<br>.<br>.<br>.<br>Command line in the last JQE<br>-or-<br>Command line in the first JQE<br>Command line in the second JQE<br>.<br>.<br>.<br>Command line in the 15th JQE<br>CW2935 ENTER 'CWP LISTC' TO CONTINUE -or-<br>CW1600 QUEUE IS EMPTY |
| CWP, EXIT | CW0900 CWP TASK EXITING |
| CWP, STATUS | CW1775 STANDBY MODE - WAITING FOR RING<br>CW1766 STANDBY MODE - LINE IN HOLD STATE<br>CW1771 RECEIVED MODE - LINE IN WAIT STATE<br>CW1770 RECEIVE MODE - LINE IS ACTIVE<br>CW1777 TRANSMIT MODE - LINE IS ACTIVE<br>CW1770 RECEIVE MODE - LINE IS ACTIVE<br>ACTIVE FILE file-id<br>CW1777 TRANSMIT MODE - LINE IS ACTIVE<br>ACTIVE FILE file-id<br>CW2930 ENTER A 'CWP RECV' TO ANSWER CALLS |
| CWP, ENDQ | CW1620 CWP QUEUE STOPPED -or-<br>CW1610 QUEUE IS NOT ACTIVE |
| CWP, WAIT | CW1771 RECEIVE MODE - LINE IS ACTIVE -or-<br>CW1772 NOTHING TO WAIT FOR |
| CWP, DISC | LINE IS BEING DISCONNECTED |

Table 13-6. Messages to 9700 Console

## Level 0 - Confirmation Messages

CW0900 CWP TASK EXITING
LINE IS BEING DISCONNECTED

## Level 1 - Information Messages

CW1350 CREATING FILE 'filename.type'
CW1351 NUMBER OF BLOCKS ALLOCATED FOR CREATING FILES
CW1600 QUEUE IS EMPTY
CW1610 QUEUE IS NOT ACTIVE
CW1620 CWP QUEUE STOPPED
CW1630 QUEUE ALREADY ACTIVE
CW1640 TRANSMIT QUEUE FULL
CW1680 MESSAGE SENT TO CALLER ID:
CW1681 MESSAGE QUEUE TO REMOTE CALLER FULL
CW1770 RECEIVE MODE — LINE IS ACTIVE
CW1771 RECEIVE MODE — LINE IN WAIT STATE
CW1772 RECEIVE MODE — TRANSMITTER IN TTD MODE
CW1773 LINE DISCONNECTED BY CALLER
CW1774 LINE DISCONNECTED BY RECEIVER
CW1775 STANDBY MODE — WAITING FOR RING
CW1776 STANDBY MODE — LINE IN 'HOLD' STATE
CW1777 TRANSMIT MODE — LINE IS ACTIVE
CW1778 TRANSMIT MODE — RECEIVER IN WAIT STATE

## Level 2 - Routine Maintenance or Action Messages

CW2710 INVALID KEY-IN
CW2711 INVALID REQUEST IN THE CURRENT CWP MODE
CW2712 INVALID FILE TYPE
CW2713 INSUFFICIENT PARAMETERS
CW2750 DID NOT FIND REQUESTED JOB
CW2915 INDEX MASTER DIRECTORY FULL
CW2920 GO TO VOICE MODE
CW2930 ENTER A 'CWP RECV' TO ANSWER CALLS
CW2935 ENTER 'CWP LISTC' TO CONTINUE
CW2940 ENTER 'CWP, ENDQ' BEFORE EXITING TASK

## Level 5 - Communication Problem Messages

CW5400 DATA SET NOT READY
CW5410 DATA TERMINAL NOT READY
CW5500 DATA ERROR - RECEIVED FIFTEEN CONSECUTIVE BAD BLOCKS

Table 13-6. Messages to 9700 Console (cont.)

## LEVEL 8—PROBABLE SEVERE SOFTWARE ERROR MESSAGES

CW8200 INVALID TMCB RECEIVED FROM A TASK
CW8400 DISK ERROR - - 'filename.type'
CW8450 DISK FULL
CW8980 UNABLE TO INITIATE JOB - - QUEUE WILL BE STOPPED

## LEVEL 9—PROBABLE SEVERE HARDWARE ERROR MESSAGES

CW9100 LINE CANNOT BE ESTABLISHED
CW9150 TELEPHONE LINE ERROR

## OS Messages Unique to 850

OS6751 INSUFFICIENT DYNAMIC MEMORY FOR WORD PROCESSING LINK JOB
OS6752 INSUFFICIENT DYNAMIC MEMORY FOR WORD PROCESSING MERGE JOB
OS6760 FONT CHAR WIDTH TABLE MISSING — WORD PROC. ABORTING JOB
OS6770 WORD PROCESSING ABORTING JOB
OS9534 WORD PROCESSING MERGE FILE I/O ERROR — ABORTING JOB
OS9535 WORD PROCESSING FILE LINKING I/O ERROR — ABORTING JOB

# APPENDIX A. PDL STATEMENT SUMMARY

Appendix A may be used as a quick reference to PDL statement syntax. Further details of each PDL statement are provided starting on the "ref. page" (shown in parentheses immediately below the command keyword). The following definitions describe the conventions and symbols used in this appendix and in the referenced pages.

## Conventions

1. Lower case letters identify an item that must be replaced by a user selected value.
2. Keywords in capital letters must be entered as shown.
3. An element in brackets [ ] is optional.
4. Elements placed inside a pair of braces { } indicate a required choice.
5. Ellipsis inside parentheses (. . .) indicate that the element preceding the ellipsis may be repeated.
6. Symbols used in this appendix are defined as follows:

| Symbol | Meaning |
|---|---|
| ac | Statement identifier. Consists of 1 to 6 alphanumeric characters (A-Z and 0-9) followed by a colon (:). One of the characters must be an alpha. |
| dd | Statement identifier. Same as 'ac' except restriction of having one alpha character is removed. This identifier only applies to SYSTEM (or JDL) and JOB (or JDE) commands. |
| id | Reference to a statement identified by an "ac" statement identifier. |
| sc | String constant. A hexadecimal, octal, ASCII, EBCDIC or character constant. |
| value | A decimal constant preferably, but may also be a string constant. |

7. The columns in this table with the heading "ON" and "OFF" indicate whether a PDL command is available for on-line (ON) and/or off-line (OFF) usage. A column with a mark in it for a particular command means that command is available for the usage type indicated. If the column is not marked, the command is not available for that mode of usage.

8. The column in this table with the heading "DJDE" indicates whether a left/right part of a command may be modified by a DJDE. If the column is marked with an "X",then the DJDE command is available for both on-line and off-line modes. If the column contains "OFF", then the corresponding DJDE command is available in off-line mode only. See Chapter 6 for further details on DJDEs.

Table A-1. PDL Statement Summary

| Command (ref. page) | Left Part | Right Parts | Default | ON | OFF | DJDE | Comments |
|---|---|---|---|---|---|---|---|
| ABNORMAL (5-36) | | | | | | | ABNORMAL command not available on 9700. Syntax acceptable by PDL for purposes of compatibility with Xerox 1200. |
| | REP = | {YES / NO} | – | | | | |
| | RES = | sev-level | – | | | | |
| | BLKSP = | {YES / NO} | – | | | | |
| | SECURITY = | {YES / NO} | – | | | | |
| ACCT (5-30) | | | | | | | Command to request an accounting summary and to specify where the summary will be delivered. |
| | USER = | {NONE / BIN / TRAY / (BIN, TRAY)} | BIN | X | X | | |
| | DEPT = | sc | jdl name | X | X | | |
| BANNER (12-5) | TEST = | {$id_1$ / ($id_1$, {AND / OR}, $id_2$)} | none | X | | | Defines test expression for the detection of a banner page. |
| | HCOUNT = | value | 0 | X | | | |
| | TCOUNT = | value | 0 | X | | | |
| BDELETE (4-24) | TEST = | {$id_1$ / ($id_1$, {AND / OR}, $id_2$)} | none | | X | | Defines test expression for deleting blocks from the printed output. |
| BLOCK (4-15) | | | | | | | Command used to specify the parameters which describe the physical structure of the tape blocks. |
| | LENGTH = | value | 1330 | | X | | |
| | LTHFLD= | size | 0 | | X | | |
| | OFFSET = | value | 0 | | X | | |

| Command (ref. page) | Left Part | Right Parts | Default | ON | OFF | DJDE | Comments |
|---|---|---|---|---|---|---|---|
| | FORMAT = | {BIN / DEC / PACK / PKSG} | BIN | | X | | |
| | ADJUST = | value | 0 | | X | | |
| | PREAMBLE = | length | 0 | | X | | |
| | POSTAMBLE = | length | 0 | | X | | |
| | CONSTANT = | sc | 0 | | X | | |
| | ZERO = | {YES / NO} | NO | | X | | |
| | LMULT = | value | 1 | | X | | |
| BSELECT (4-24) | TEST = | $\{id_1 / (id_1, \{AND / OR\}, id_2)\}$ | none | | X | | Defines test expression for selecting blocks for printing. |
| ac: CATALOG; (3-6) | | | none | X | X | | Command used to group statements which are to be included in more than one job descriptor entry. |
| ac: CME (5-18) | | | | | | | Command used to specify a copy modification entry which modifies output pages on a per copy basis. |
| | LINE = | {n / (n, m) / (n, -)} | none | X | X | | |
| | POS = | p | 1 | X | X | | |
| | FONT = | value | 1 | X | X | | |
| | CONSTANT = | sc | none | X | X | | |
| [ac:] CODE (4-14) | | | | | | | Command used to assign an input-code to output-code correspondence. |
| | DEFAULT = | code-type | EBCDIC | X | X | | "code-type" must be one of the following: ASCII, BCD, BCL, CBCD, EBCDIC, PEBCDIC, H2BCD, H6BCD, IBMBCD, value. |

Table A-1. PDL Statement Summary (cont.)

| Command (ref. page) | Left Part | Right Parts | Default | ON | OFF | DJDE | Comments |
|---|---|---|---|---|---|---|---|
| ac: CODE (cont.) | ASSIGN = | {(input, output), (input, (output, ..., output))} | none | X | X | | |
| ac: CRITERIA (4-22) | CONSTANT = | (offset, length, {EQ NE}, id) | none | X | X | | Defines test of subfield of a record or block against entries in a value table. |
| | LINENUM = | (init, count) | all lines | X | | | See Table 12-1 for on-line usage. |
| ac: CRITERIA (4-23) | CHANGE = | (offset, length, NE, LAST) | none | X | X | | Defines a subfield of each record or block which is to be compared with the same subfield of the previous record or block. |
| | LINENUM = | (init,count) | all lines | X | | | See Table 12-1 for on-line usage. |
| END; (3-7) | | | none | X | X | | Command used to terminate a job descriptor library. |
| IDEN (6-3) | | | | | | | Command used to identify a dynamic job descriptor entry. |
| | PREFIX = | sc | none | X | X | | |
| | SKIP = | value | 1 | X | X | | |
| | OFFSET = | value | 0 | X | X | | |
| | OPRINFO = | {YES NO} | NO | X | X | | |
| dd: {JOB JDE} (3-6) | [INCLUDE = | ($id_1$, ..., $id_n$)] | none | X | X | OFF | Start of job descriptor entry. INCLUDE specified when using a catalog command set. |
| LINE (5-7) | | | | | | | Command used to define and format the output print line. |
| | MARGIN = | {value, (value {b ,} . {IN CM POS})} | (1, POS) | X | X | X | |

| Command (ref. page) | Left Part | Right Parts | Default | ON | OFF | DJDE | Comments |
|---|---|---|---|---|---|---|---|
| | DATA = | (offset, length) | (1, 132) | | X | X | |
| | PCC = | (offset, {TRAN / NOTRAN}) | (0, NOTRAN) | | X | | |
| | PCCTYPE = | pcctype | ANSI | X | X | | "pcctype" must be one of the following: ANSI, B2500, B2700, B3500, B3700, B4700, B6700, H2000, H6000, IBM1401, IBM1403, IBM3211, NONE, UNIVAC, USER, US70, XEROX, id |
| | OVERPRINT = | ({PRINT / MERGE / IGNORE / PRINT2}, {DISP / NODISP}) | (PRINT, NODISP) | X | X | X | |
| | VFU = | {vfu-id / NONE} | NONE | X | X | | |
| | FONTINDEX = | {offset / NONE} | NONE | | X | X | |
| | FCB = | {IGNORE / PROCESS} | PROCESS | X | | | See Table 12-1 for on-line usage. |
| | UCSB = | {IGNORE / PROCESS} | PROCESS | X | | | See Table 12-1 for on-line usage. |
| MESSAGE (5-27) | | | | | | | Command used to cause a message to be displayed to the operator during job processing or job printing. |
| | ITEXT = | {sc / (sc ,passnum) / NONE} | NONE | X | X | X | |
| | OTEXT = | {sc / (sc [,{passnum / END}] [,WAIT]) / NONE} | NONE | X | X | X | |
| OUTPUT (5-2) | | | | | | | Command used to specify the copy count and collate/ uncollate mode, CMEs and offset control. |
| | COPIES = | value | 1 | X | X | X | |
| | COLLATE = | {YES / NO} | YES | X | X | X | |

Table A-1. PDL Statement Summary (cont.)

| Command (ref. page) | Left Part | Right Parts | Default | ON | OFF | DJDE | Comments |
|---|---|---|---|---|---|---|---|
| OUTPUT (cont.) | MODIFY = | {cme-id / (cme-id, init [,copies]) / NONE} | NONE | X | X | X | See section "Copy Modification Entries" in Chapter 12 for on-line usage restrictions. |
| | OFFSET = | {ALL / FIRST / NONE} | ALL | | X | | |
| | COVER = | {FRONT / (FRONT, SEP) / BACK / BOTH / (BOTH, SEP) / NONE} | NONE | | X | | |
| | FORMAT = | pde-id | FMT1 | X | X | X | |
| | NUMBER = | {(pnum, lnum, cnum) / NO} | NO | X | X | X | |
| | FORMS = | {form-id / (form-id, init [,copies]) / NONE} | NONE | X | X | X | |
| [ac:] PCC (5-14) | | | | | | | Command used to assign one-byte printer carriage control codes and define their actions. |
| | INITIAL = | {BOF / TOF} | TOF | | X | | |
| | DEFAULT = | {ccln / pctype} | PSP1 | | X | | "pctype" must be one of the following: ANSI, B2500, B2700, B3500, B4700, B6700, H2000, H6000, IBM1401, IBM1403, US70, UNIVAC, XEROX, NONE |
| | MASK = | value | X'FF' | | X | | |
| | ADVTAPE = | {YES / NO} | YES | | X | | |
| | ASSIGN = | {(byte, ccln) / (byte, (ccln, ..., ccln))} | none | | X | | |
| ac: PDE (5-22) | | | | | | | Command used to describe formatting information for each page of the report. |
| | PMODE = | {PORTRAIT / LANDSCAPE} | LANDSCAPE | X | X | | |

| Command (ref. page) | Left Part | Right Parts | Default | ON | OFF | DJDE | Comments |
|---|---|---|---|---|---|---|---|
| | FONTS = | $\left\{\begin{array}{l}(f_1\left[[,f_2],..\right]) \\ ((f_1,s_1)\left[[,(f_2,s_2)]...\right])\end{array}\right\}$ | none | X | X | | |
| | BEGIN = | (vpos $\left\{\begin{array}{l}CM\\IN\end{array}\right\}$ , hpos $\left\{\begin{array}{l}CM\\IN\end{array}\right\}$ ) | (0. IN, 0. IN) | X | X | | |
| RDELETE (4-26) | TEST = | $\left\{\begin{array}{l}id_1 \\ (id_1, \left\{\begin{array}{l}AND\\OR\end{array}\right\}, id_2)\end{array}\right\}$ | none | | X | | Defines test expression for deleting records from the printed output. |
| RECORD (4-18) | | | | | | | Command used to specify parameters which describe the characteristics of the logical tape records. |
| | STRUCTURE = | structure-type | FB | | X | | "structure-type" must be one of the following: F, FB, U, UB, V, VB |
| | LENGTH = | value | 133 | | X | | |
| | LTHFLD = | size | 0 | | X | | |
| | OFFSET = | value | 0 | | X | | |
| | FORMAT = | $\left\{\begin{array}{l}BIN\\DEC\\PACK\\PKSG\end{array}\right\}$ | BIN | | X | | |
| | ADJUST = | value | 0 | | X | | |
| | PREAMBLE = | length | 0 | | X | | |
| | POSTAMBLE = | length | 0 | | X | | |
| | CONSTANT = | sc | none | | X | | |
| | LMULT = | value | 1 | | X | | |
| ROFFSET (4-28) | TEST = | $\left\{\begin{array}{l}id_1 \\ (id_1, \left\{\begin{array}{l}AND\\OR\end{array}\right\}, id_2)\end{array}\right\}$ | none | | X | | Defines test expression for offsetting output to the bin. |
| | PASSES = | $\left\{\begin{array}{l}FIRST\\ALL\end{array}\right\}$ | ALL | | X | | |
| ROUTE (5-28) | | | | | | | Command to print a message on a separate page preceding a report ply or an entire report. |

Table A-1. PDL Statement Summary (cont.)

| Command (ref. page) | Left Part | Right Parts | Default | ON | OFF | DJDE | Comments |
|---|---|---|---|---|---|---|---|
| ROUTE (cont.) | RTEXT = | {sc / (sc ,passnum [,line [,col]]) / NONE} | NONE | X | X | X | |
| | RFORM = | {form-id / NONE} | NONE | | X | X | |
| RRESUME (4-31) | TEST = | {$id_1$ / ($id_1$, {AND / OR} , $id_2$)} | none | | X | | Defines test expression for resuming printing that has been suspended. |
| | BEGIN = | {CURRENT / NEXT} | NEXT | | X | | |
| RSELECT (4-26) | TEST = | {$id_1$ / ($id_1$, {AND / OR} , $id_2$)} | none | | X | | Defines test expression for selecting records for printing. |
| RSTACK (4-34) | TEST = | {$id_1$ / ($id_1$, {AND / OR} , $id_2$)} | none | | X | | Defines test expression for detecting an end-of-report. |
| | DELIMITER = | {YES / NO} | NO | | X | | |
| | PRINT = | {NONE / BIN / TRAY / BOTH} | NONE | | X | | |
| | ACCTINFO = | (offset, length) | none | | X | | |
| RSUSPEND (4-30) | TEST = | {$id_1$ / ($id_1$, {AND / OR} , $id_2$)} | none | | X | | Defines test expression for suspending pritning. |
| | BEGIN = | {CURRENT / NEXT} | NEXT | | X | | |
| dd: {SYSTEM / JDL} ; (3-5) | | | | X | X | OFF | Command used to identify the beginning of a job descriptor library. |
| ac: TABLE (4-37) | CONSTANT = | (sc, . . . , sc) | none | X | X | | Command to build a table of constants for use by the logical processing functions. |
| | MASK = | ('char') | none | X | | | See Table 12-1 for on-line usage. |

| Command (ref. page) | Left Part | Right Parts | Default | ON | OFF | DJDE | Comments |
|---|---|---|---|---|---|---|---|
| [ac:] VFU (5-11) | ASSIGN = | (channo, lineno)<br>(channo, (lineno, . . . lineno)) | none | X | X | X | Command used to specify the line position to be associated with each channel used in the carriage control convention and the top- and bottom-of-form physical margins. |
| | TOF = | value | 1 | X | X | X | |
| | BOF = | value | 66 | X | X | X | |
| VOLUME (4-9) | | | | | | | Command used to specify parameters which describe the format of the job tape. |
| | HOST = | host-type | IBMOS | X | X | | "host-type" must be one of the following: ANSI, B2500, B2700, B3500, B3700, B4700, B6700, DUMP, GRASP, H2000, H6000, IBMOS, IBMDOS, IBMONL, OCTDUMP, OSWTR, POWER, POWERVS, UNIVAC, US70, XEROX.<br>See Table 12-1 for HOST=IBMONL usage. |
| | UNPACK = | { T4X3 / T4X3H2 / UNIVAC / NONE } | NONE | | X | | |
| | LABEL = | { ANSI / COBOL / NONE / SPR / STANDARD } | STANDARD | | X | | |
| | CODE = | code-type | EBCDIC | X | X | | "code-type" must be one of the following: ASCII, BCD, BCL, CBCD, EBCDIC, H2BCD, H6BCD, IBMBCD, PEBCDIC, USER, id |
| | LCODE = | lcode-type | EBCDIC | | X | | "lcode-type" must be one of the following: ASCII, BCD, BCL, CBCD, EBCDIC, H2BCD, H6BCD, IBMBCD, USER, id |
| | EOV = | ( { PAUSE / NOPAUSE }, { EOF / NOEOF } ) | (NOPAUSE, NOEOF) | | X | | |

Table A-1. PDL Statement Summary (cont.)

| Command (ref. page) | Left Part | Right Parts | Default | ON | OFF | DJDE | Comments |
|---|---|---|---|---|---|---|---|
| VOLUME (cont.) | PLABEL = | {YES / NO} | NO | | X | | |
| | OSCHN = | value | 9 | | X | | OSCHN, OSHDP, and OSTLP are meaningful only for HOST = OSWTR. |
| | OSHDP = | value | 0 | | X | | |
| | OSTLP = | value | 0 | | X | | |
| | BMULT = | value | 1 | | X | | |
| | RMULT = | value | 1 | | X | | |
| | RMODE = | {S / M} | M | | X | | |

# APPENDIX B. SYSTEM ERROR MESSAGES

## Level 0 - Confirmation Messages

Level 0

OS0010 RESUMING INPUT
OS0020 RESUMING OUTPUT
OS0050 MAIN TRAY LOWERING
OS0110 BIN 1 LOWERING
OS0120 BIN 2 LOWERING
OS0200 MAIN TRAY SELECTED
OS0210 AUX TRAY SELECTED
OS0310 BIN 1 SELECTED
OS0320 BIN 2 SELECTED
OS0500 OUTPUT STOPPED
OS0510 INPUT STOPPED
OS0610 PAGE SPACING FORWARD
OS0620 PAGE SPACING BACKWARD
OS0900 JOB ***** ABORTED
OS0950 TASK ABORTED
OS0990 RESETTING THE SYSTEM

## Level 1 - Information Messages

Level 1

OS1000 READY FOR COMMANDS
OS1010 STARTING JOB *****
OS1020 JOB ***** HAS COMPLETED INPUT PHASE
OS1030 JOB ***** HAS COMPLETED PRINTING
OS1050 REWIND WILL BE DONE AT END-OF-JOB
OS1080 START ACCOUNTING REPORT
OS1090 END ACCOUNTING REPORT
OS1110 BIN 1 BUSY
OS1120 BIN 2 BUSY
OS1150 JOB QUEUE FULL
OS1210 BIN 1 ALREADY UNLOADED
OS1220 BIN 2 ALREADY UNLOADED
OS1300 NUMBER OF ACTIVE FORMS IS ***
OS1310 NUMBER OF ACTIVE FONTS IS ***
OS1380 ALIGNMENT IS ***** SCAN LINES AND ***** DOTS
OS1390 INVALID SCAN OR DOT VALUE. RETRY ALI
OS1400 SAMPLE IGNORED...PRINTER IDLE
OS1410 SAMPLE NOT ALLOWED BY JDE
OS1420 PAGE SPACING NOT ALLOWED BY JDE
OS1430 BLOCK SPACING NOT ALLOWED BY JDE
OS1450 CANNOT SPACE REPORTS BEFORE START COMMAND

OS1460 CANNOT SPACE IF 'REPORTS' IS SPECIFIED WITH START CMD
OS1500 PAGE SPACING STOPPED BY BEGINNING-OF-REPORT
OS1510 PAGE SPACING STOPPED BY END-OF-REPORT
OS1520 BLOCK SPACING STOPPED BY END OF FILE
OS1530 CAN NOT BACKWARD SPACE ANYMORE PAGES
OS1540 CANNOT SPACE PAST START OF JOB
OS1550 SPACE OR MOVE FUNCTION STOPPED BY END OF DATA
OS1555 FILE SPACING STOPPED BY END OF VOLUME
OS1560 TAPE REWIND COMPLETE
OS1600 INPUT TASK NOT ACTIVE
OS1610 OUTPUT TASK NOT ACTIVE
OS1650 OUTPUT PROCESSING HAS CAUGHT-UP WITH INPUT PROCESSING
OS1700 INPUT PROCESSING NOT CURRENTLY STOPPED
OS1710 INPUT PROCESSING ALREADY STOPPED
OS1730 CLEAR INVALID WITH DISPLAY, CLEAR IGNORED
OS1750 NOTHING TO ABORT
OS1760 TASK TERMINATED
OS1800 INPUT PROCESSING ABORTING
OS1810 PRINTING ABORTING
OS1820 OUTPUT PROCESSING IS ABORTING CURRENT REPORT
OS1850 JOB STARTED AT END OF DATA, NOTHING FOUND TO PRINT
OS1910 THIS FUNCTION OF INPUT NOT IMPLEMENTED
OS1920 THIS SYSTEM FUNCTION NOT IMPLEMENTED

## Level 2 - Routine Maintenance or Action Messages Level 2

OS2000 ENTER CONTINUE 0 TO RESUME PRINTING
OS2005 ENTER 'CONTINUE I' TO RESUME INPUT
OS2010 MOUNT INPUT TAPE; 'CONTINUE I' WHEN READY
OS2020 MOUNT NEXT VOLUME; 'CONTINUE I' WHEN READY
OS2030 TAPE AT EOV 'CONTINUE I' WILL REWIND TAPE
OS2060 REFILL MAIN TRAY
OS2070 REFILL AUX TRAY
OS2100 MAIN TRAY NOT READY. IS THE DOOR SHUT?
OS2110 AUX TRAY NOT READY. IS THE COVER DOWN?
OS2150 DISPLAY (Y/N) ?
OS2210 BIN 1 FULL
OS2220 BIN 2 FULL
OS2310 BIN 1 NOT READY
OS2320 BIN 2 NOT READY
OS2400 PRINTER MISFEED DETECTED. CHECK PAPER SUPPLY
OS2500 REFILL DRY INK HOPPER
OS2530 PLEASE CLEAN COROTRONS NOW
OS2550 DRY INK RECLAIM BOTTLE OR FILTER BAG FULL
CHANGE, THEN PUSH 'BOTTLE/FILTER RESET' BUTTON

OS2560 PLEASE CLEAN THE TAPE DRIVE BEFORE NEXT TAPE MOUNT
OS2580 POP SENSOR REQUIRES CLEANING
OS2600 TAPE VOLUME OUT OF SEQUENCE; MOUNT CORRECT VOLUME
OS2700 KEY-IN LOST. RE-ENTER
OS2710 INVALID COMMAND RE-ENTER
OS2720 INVALID CONTROL KEY. RETRY
OS2730 REQUESTED TASK NOT FOUND IN SYSTEM. CHECK & RETRY
OS2740 JDE NOT FOUND. CHECK AND RETRY
OS2741 JDL NOT FOUND. CHECK AND RETRY
OS2750 JOB NOT FOUND. CHECK & RETRY
OS2760 SAMPLE FILE NOT FOUND. CHECK AND RE-ENTER
OS2800 MOVE OR SPACE FUNCTION COMPLETE
ENTER 'CON I' OR 'CON JDE, JDL' TO START NEXT REPORT
OS2810 START-UP MESSAGE FROM TAPE FOLLOWS:
OS2820 TASK NOT ALLOWED TILL SYSTEM STATUS = 'IDLE'
OS2840 OUTPUT MUST BE STOPPED BEFORE PAGE SPACING CAN BE DONE
OS2850 DISK SATURATED. IF OUTPUT IS STOPPED 'SPA' OR 'CON' IT
OS2880 MAX FONTS & FORMS EXCEEDED. ENTER NEW VALUE. RESTART JOB
OS2885 MAX NUMBER OF FONTS EXCEEDED. ENTER NEW VALUE. RESTART JOB
OS2900 TAPE BLOCK LENGTH EXCEEDS JDE MAX. DO THE FOLLOWING:
* ABORT AND RETRY. SPECIFYING ANOTHER JDE/JDL
OS2910 NO ACCOUNTING FILE ENTRY FOR DEPARTMENT
OS2950 SOFTWARE TIMING HAS CAUSED A PRINTER SOFT STOP
OS2980 FUNCTION NOT AVAILABLE IN THIS LOGON CLASS
OS2990 USE 'PROBLEM' AT EARLIEST OPPORTUNITY

## Level 3 - Printer Problem Messages Level 3

OS3010 PRINTER IS WARMING UP
OS3100 XEROGRAPHIC ENGINE INTERLOCK OPEN. CLOSE DOORS & COVER
OS3120 IMAGING MODULE INTERLOCK OPEN. CLOSE COVER
OS3150 OUTPUT MODULE INTERLOCK OPEN. CLOSE COVER
OS3200 FUSER TEMPERATURE IS BELOW MINIMUM OPERATING TEMPERATURE
WAIT THREE MINUTES
OS3300 REMOVE JAMMED SHEET FROM SAMPLE TRAY
OS3310 REMOVE JAMMED SHEET FROM BIN 1
OS3320 REMOVE JAMMED SHEET FROM BIN 2
OS3330 BIN JAM DETECTED
OS3400 PRINTER JAM...CLEAR PAPER PATH
OS3450 PRINTER JAM...CLEAR PAPER PATH INCLUDING PHOTORECEPTOR
OS3700 SUSPECTED PAGE-DELIVERY ERROR. CHECK OUTPUT

## Level 4 - System or Tape Problem Messages — Level 4

OS4010 CANNOT FIND END OF TAPE RELECTOR STRIP. 'CONTINUE I'

OS4050 TOO MANY FORMS/FONTS SPECIFIED VIA FORMS/FONTS COMMANDS
ENTER NEW VALUES VIA THE FORMS/FONTS COMMAND.

OS4100 PRINTER IS IN MANUAL MODE. CALL DISPATCH. SAN 10.04.91

OS4150 TAPE DRIVE IS OFF-LINE. 'CONTINUE I' WHEN TAPE IS READY

OS4200 TAPE DRIVE NOT RESPONDING. DO ONE OF THE FOLLOWING:
CALL DISPATCH. SAN 20.00.01

OS4310 IRRECOVERABLE TAPE READ ERROR DURING TAPE DUMP

OS4500 BAD BLOCK ON TAPE. DO ONE OF THE FOLLOWING:

* MOVE - 1 BLOCKS TO RETRY READ
* ABORT JOB AND CLEAN THE TAPE DRIVE THEN RESTART JOB
* SPACE 1 REPORT
* 'CONTINUE I' IF DATA LOSS ACCEPTABLE
* IF PROBLEM CONTINUES, RUN ANOTHER JOB TAPE

OS4520 INPUT ATTEMPTING RECOVERY . . . TAPE NOT READY

OS4650 INPUT HAS RECOVERED TO PAGE BOUNDARY

OS4651 OUTPUT HAS RECOVERED TO PAGE BOUNDARY

OS4990 SYSTEM RELIABILITY LOG BEING LOST. USE 'PROBLEM'

## Level 6 - Job Integrity Messages — Level 6

OS6000 INSUFFICIENT MEMORY FOR 'ACCTINFO'; CONTINUE OR ABORT?

OS6010 LABEL ERROR : INVALID LABEL FORMAT; CONTINUE OR ABORT?

OS6011 LABEL ERROR : VOL1; CONTINUE OR ABORT?

OS6012 LABEL ERROR : HDR1; CONTINUE OR ABORT?

OS6013 LABEL ERROR : UHL, TM, OR HDR2; CONTINUE OR ABORT?

OS6014 LABEL ERROR : EOF OR EOV; CONTINUE OR ABORT?

OS6015 LABEL ERROR : TM, HDR, OR UHL; CONTINUE OR ABORT?

OS6016 LABEL ERROR : TM OR USER; CONTINUE OR ABORT?

OS6017 LABEL ERROR : TAPE MARK; CONTINUE OR ABORT?

OS6018 LABEL ERROR : EOF; CONTINUE OR ABORT?

OS6019 LABEL ERROR : EOV; CONTINUE OR ABORT?

OS6020 LABEL ERROR : UVL OR HDR1; CONTINUE OR ABORT?

OS6021 LABEL ERROR : UTL, TM, OR EOF; CONTINUE OR ABORT?

OS6022 LABEL ERROR : ANSI OPTION 3; CONTINUE OR ABORT?

OS6023 LABEL ERROR : 1HDR; CONTINUE OR ABORT?

OS6024 LABEL ERROR : 1EOR, TM, OR 1EOF; CONTINUE OR ABORT?

OS6025 LABEL ERROR : 1EOR OR 1EOF; CONTINUE OR ABORT?

OS6026 LABEL ERROR : 1EOR; CONTINUE OR ABORT?

OS6027 LABEL ERROR : 1EOF; CONTINUE OR ABORT?

OS6028 LABEL ERROR : 1ERI, TM, OR 1HDR; CONTINUE OR ABORT?

OS6029 LABEL ERROR : BASIC TAPE; CONTINUE OR ABORT?

OS6030 LABEL ERROR : EOF OR EOR; CONTINUE OR ABORT?

OS6031 LABEL ERROR : EOR; CONTINUE OR ABORT?

OS6032 LABEL ERROR : HDR1, UVL, OR VOL; CONTINUE OR ABORT?

OS6033 LABEL ERROR : HDR1 OR UVL; CONTINUE OR ABORT?
OS6034 LABEL ERROR : TM, EOF2, OR UTL; CONTINUE OR ABORT?
OS6035 LABEL ERROR : TM OR TRAILER; CONTINUE OR ABORT?
OS6036 LABEL ERROR : TM OR HDR1; CONTINUE OR ABORT?
OS6037 LABEL ERROR : STANDARD HDR; CONTINUE OR ABORT?
OS6038 LABEL ERROR : STANDARD EOF OR EOV; CONTINUE OR ABORT
OS6039 LABEL ERROR : STANDARD EOF; CONTINUE OR ABORT?
OS6040 LABEL ERROR : STANDARD EOV; CONTINUE OR ABORT?
OS6041 LABEL ERROR : SPECIAL BLOCK LBL; CONTINUE OR ABORT?
OS6042 LABEL ERROR : ILLEGAL POWER V/S FORMAT; CONTINUE OR ABORT?
OS6080 LABEL ERROR : BAD RECORD FORMAT; CONTINUE OR ABORT?
OS6090 LABEL ERROR : ILLEGAL BLOCK LENGTH; CONTINUE OR ABORT?
OS6200 LABEL AND FILE BLOCK COUNT MISMATCH. CONTINUE OR ABORT?
OS6500 CANNOT VALIDATE FIRST DATA RECORD; SPACE TO NEXT REPORT
OS6550 DATA NOT FORMATTED AS SPECIFIED; SPACE TO NEXT REPORT
OS6650 INPUT HAS RECOVERED TO NEXT REPORT IN JOB
OS6651 OUTPUT HAS RECOVERED TO NEXT REPORT IN JOB
OS6700 SYNTAX ERROR IN DJDE. CONTINUE OR ABORT?
OS6710 DISK PROCESSING ERROR. CONTINUE OR ABORT?
OS6750 OUT OF DYNAMIC MEMORY, DJDE IGNORED. CONTINUE OR ABORT?
RUN NEXT JOB AND REPORT PROBLEM
OS6800 OUTPUT TASK CRASH * RELOADING. CHECK OUTPUT AT END OF JOB
OS6900 DATA ON PAGE EXCEEDS 8½ X 11. CHECK OUTPUT
OS6950 LINE DENSITY EXCEEDED. PAGE WILL NOT BE PRINTED. ABORT OR

## Level 7 - System Problem Messages Level 7

OS7100 PCC OR TRANSLATE TABLE UNREADABLE
OS7110 CME FILE NOT FOUND
OS7120 PDE FILE NOT FOUND
OS7130 FONT FILE NOT FOUND
OS7140 FORM FILE NOT FOUND
OS7150 FORM FONT NOT FOUND
OS7300 ACCOUNT FORM FILE NOT FOUND/ACCOUNTING CAN'T PRINT
OS7500 INSUFFICIENT MEMORY FOR JDE
OS7510 INSUFFICIENT MEMORY FOR JDE TABLES
OS7520 INSUFFICIENT MEMORY FOR VFU TABLES
OS7530 INSUFFICIENT MEMORY FOR CME TABLES
OS7540 INSUFFICIENT MEMORY FOR TAPE DUMP PRINT BUFFER
OS7550 INSUFFICIENT MEMORY FOR INPUT BUFFERS
OS7800 INSUFFICIENT MEMORY FOR FORM PRINT
OS7810 INSUFFICIENT MEMORY FOR SAMPLE FONT
OS7820 INSUFFICIENT DYNAMIC MEMORY FOR SAMPLE LOGIC
OS7830 INSUFFICIENT DYNAMIC MEMORY FOR FONT INDEX
OS7850 TOO MANY DATA AND FORM FONTS AND FORMS SPECIFIED IN JDE
OS7900 FONT MEMORY SIZE EXCEEDED. RUN NEXT JOB. REPORT ERROR
OS7910 JOB TOO BIG FOR AVAILABLE MEMORY-OUTPUT
OS7950 UNABLE TO OPEN PRINT FILE . . . ENTER "RESET" THEN "REA"

## Level 8 - Probable Severe Software Error Messages — Level 8

OS8010 INPUT FOUND NOTHING TO PRINT
OS8100 REQUESTED TASK ALREADY ACTIVE - REBOOT THE SYSTEM
OS8200 INVALID TMCB RECEIVED FROM A TASK
OS8300 INPUT HAD AN I/O ERROR DURING FORM FILE READ
OS8320 INPUT CANNOT OPEN ACCOUNTING FILE
OS8410 BYTE ALIGNED DISK I/O REQUEST...INPUT ABORTING
OS8420 BAD LBN ON DISK I/O...INPUT ABORTING
OS8430 ILLEGAL ADDRESS SPACE ON DISK I/O...INPUT ABORTING
OS8500 SYSTEM RELIABILITY LOG LOST
OS8600 UNSUCCESSFUL COMPLETION OF SEND DATA DIRECTIVE
OS8650 INPUT HAS RECOVERED TO NEXT JOB IN QUEUE
OS8651 OUTPUT HAS RECOVERED TO NEXT JOB IN QUEUE
OS8700 ILLEGAL META-CODE IN DATA
OS8800 INSUFFICIENT DYNAMIC MEMORY - INPUT
OS8810 INSUFFICIENT DYNAMIC MEMORY FOR RTEXT
OS8850 FILE MANAGEMENT INITIALIZATION FAILURE. TRY REBOOTING
OS8940 INSUFFICIENT INPUT TASK MEMORY
OS8950 INSUFFICIENT DYNAMIC MEMORY TO RUN REQUESTED TASK
OS8970 CANNOT OPEN RECOVERY FILE
OS8980 UNABLE TO INITIATE INPUT TASK
OS8990 UNABLE TO INITIATE OUTPUT TASK

## Level 9 - Probable Severe Hardware Error Messages — Level 9

OS9200 HARDWARE ERROR - TAPE DRIVE. CALL DISPATCH. SAN 20.00.02
OS9300 PAGE SET-UP ERROR. PAGE WILL NOT BE PRINTED. ABORT OR
OS9380 PRINTER FAILURE ENTER 'PROBLEM'; OR
OS9420 DISK ERROR/PRINT FILE - INPUT. CALL DISPATCH. SAN 30.00.02
OS9440 DISK ERROR/ACCOUTNING FILE - OUTPUT. CALL DISPATCH. SAN 30.00.03
OS9460 DISK ERROR - OCS. CALL DISPATCH. SAN 30.00.04
OS9500 BAD BLOCK ON DISK - INPUT. CALL DISPATCH. SAN 30.00.05
OS9530 DISK ERROR - INPUT. CALL DISPATCH. SAN 30.00.06
OS9550 DISK ERROR - OUTPUT. CALL DISPATCH. SAN 30.00.07
OS9560 FONT MEMORY WRITE ERROR. CALL DISPATCH. SAN 50.00.01
OS9580 IRRECOVERABLE TAPE READ ERROR. CALL DISPATCH. SAN 20.00.03
OS9800 FATAL HARDWARE ERROR ON DISK - INPUT. CALL DISPATCH. SAN 30.00.08
OS9920 NO SYSTEM LOG. CALL DISPATCH. SAN 30.00.99
OS9950 MEMORY IS TOO DEGRADED TO USE. CALL DISPATCH. SAN 00.00.98

# APPENDIX C. XEROX 9700 CHARACTER SETS

**ASCII and EBCDIC Standard Character Sets :**

| Type | I | II | III | IV | V | VI | VII | VIII | IX |
|---|---|---|---|---|---|---|---|---|---|
| Page | C-2 | C-3 | C 4 | C-5 | C-6 | C-7 | C-8 | C-9 | C-10 |

## Standard Character Set I
## ASCII

| LEAST SIGNIFICANT BITS | | MOST SIGNIFICANT BITS | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Hex | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| | Binary | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| 0 | 0000 | RESERVED FOR XEROX USE ONLY | | | 0 | | P | | | | | | | | | | |
| 1 | 0001 | | | | 1 | A | Q | | | | | | | | | | |
| 2 | 0010 | | | | 2 | B | R | | | | | | | | | | |
| 3 | 0011 | | | | 3 | C | S | | | | | | | | | | |
| 4 | 0100 | | | | 4 | D | T | | | | | | | | | | |
| 5 | 0101 | | | | 5 | E | U | | | | | | | | | | |
| 6 | 0110 | | | | 6 | F | V | | | | | | | | | | |
| 7 | 0111 | | | | 7 | G | W | | | | | | | | | | |
| 8 | 1000 | | | | 8 | H | X | | | | | | | | | | |
| 9 | 1001 | | | | 9 | I | Y | | | | | | | | | | |
| A | 1010 | | | | | J | Z | | | | | | | | | | |
| B | 1011 | | | | | K | | | | | | | | | | | |
| C | 1100 | | | , | | L | | | | | | | | | | | |
| D | 1101 | | | – | | M | | | | | | | | | | | |
| E | 1110 | | | . | | N | | | | | | | | | | | |
| F | 1111 | | | / | | O | | | | | | | | | | | |

## Standard Character Set I
## EBCDIC

| LEAST SIGNIFICANT BITS | | MOST SIGNIFICANT BITS | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Hex | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| | Binary | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| 0 | 0000 | | | | | | | – | | | | | | | | | 0 |
| 1 | 0001 | | | | | | | / | | | | | | A | J | | 1 |
| 2 | 0010 | | | | | | | | | | | | | B | K | S | 2 |
| 3 | 0011 | | | | | | | | | | | | | C | L | T | 3 |
| 4 | 0100 | | | | | | | | | | | | | D | M | U | 4 |
| 5 | 0101 | | | | | | | | | | | | | E | N | V | 5 |
| 6 | 0110 | | | | | | | | | | | | | F | O | W | 6 |
| 7 | 0111 | | | | | | | | | | | | | G | P | X | 7 |
| 8 | 1000 | | | | | | | | | | | | | H | Q | Y | 8 |
| 9 | 1001 | | | | | | | | | | | | | I | R | Z | 9 |
| A | 1010 | | | | | | | | | | | | | | | | |
| B | 1011 | | | | | | | | | | | | | | | | |
| C | 1100 | | | | | | | | | | | | | | | | |
| D | 1101 | | | | | | | | | | | | | | | | |
| E | 1110 | | | | | | | | | | | | | | | | |
| F | 1111 | | | | | | | | | | | | | | | | |

## Standard Character Set II
## ASCII

| LEAST SIGNIFICANT BITS | | MOST SIGNIFICANT BITS | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Hex | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| | Binary | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| 0 | 0000 | RESERVED FOR XEROX USE ONLY | | | 0 | | P | | | | | | | | | | |
| 1 | 0001 | | | | 1 | A | Q | | | | | | | | | | |
| 2 | 0010 | | | | 2 | B | R | | | | | | | | | | |
| 3 | 0011 | | | | 3 | C | S | | | | | | | | | | |
| 4 | 0100 | | | $ | 4 | D | T | | | | | | | | | | |
| 5 | 0101 | | | | 5 | E | U | | | | | | | | | | |
| 6 | 0110 | | _ | & | 6 | F | V | | | | | | | | | | |
| 7 | 0111 | | ⌋ | ' | 7 | G | W | | | | | | | | | | |
| 8 | 1000 | | \| | | 8 | H | X | | | | | | | | | | |
| 9 | 1001 | | \| | | 9 | I | Y | | | | | | | | | | |
| A | 1010 | | | * | | J | Z | | | | | | | | | | |
| B | 1011 | | | + | | K | | | | | | | | | | | |
| C | 1100 | | | , | < | L | | | | | | | | | | | |
| D | 1101 | | | - | = | M | | | | | | | | | | | |
| E | 1110 | | | . | > | N | | | | | | | | | | | |
| F | 1111 | | | / | | O | _ | | | | | | | | | | |

## Standard Character Set II
## EBCDIC

| LEAST SIGNIFICANT BITS | | MOST SIGNIFICANT BITS | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Hex | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| | Binary | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| 0 | 0000 | | | | | | & | - | | | | | | | | | 0 |
| 1 | 0001 | | | | | | | / | | | | | | A | J | | 1 |
| 2 | 0010 | | | | | | | | | | | | | B | K | S | 2 |
| 3 | 0011 | | | | | | | | | | | | | C | L | T | 3 |
| 4 | 0100 | | | | | | | | | | | | | D | M | U | 4 |
| 5 | 0101 | | | | | | | | | | | | | E | N | V | 5 |
| 6 | 0110 | | | | | | | | \| | | | | | F | O | W | 6 |
| 7 | 0111 | | | | | | | | \| | | | | | G | P | X | 7 |
| 8 | 1000 | | | | | | | | | | | | | H | Q | Y | 8 |
| 9 | 1001 | | | | | | | | | | | | | I | R | Z | 9 |
| A | 1010 | | | | | | | | | | | | | | | | |
| B | 1011 | | | | | . | $ | , | | | | | ⌋ | | | | |
| C | 1100 | | | | | | * | | | | | | | ⌋ | | ⌋ | |
| D | 1101 | | | | | | | _ | | | | | | | | | |
| E | 1110 | | | | | | | | | | | | | Y | | | |
| F | 1111 | | | | | | | | | | | | _ | | | | |

## Standard Character Set III
## ASCII

| LEAST SIGNIFICANT BITS | Hex | Binary | MOST SIGNIFICANT BITS 0 (0000) | 1 (0001) | 2 (0010) | 3 (0011) | 4 (0100) | 5 (0101) | 6 (0110) | 7 (0111) | 8 (1000) | 9 (1001) | A (1010) | B (1011) | C (1100) | D (1101) | E (1110) | F (1111) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0000 | RESERVED FOR XEROX USE ONLY | | | 0 | | P | | | | | | | | | | |
| | 1 | 0001 | | | | 1 | A | Q | | | | | | | | | | |
| | 2 | 0010 | | | | 2 | B | R | | | | | | | | | | |
| | 3 | 0011 | | ! | | 3 | C | S | | | | | | | | | | |
| | 4 | 0100 | | $ | * | 4 | D | T | | | | | | | | | | |
| | 5 | 0101 | | % | | 5 | E | U | | | | | | | | | | |
| | 6 | 0110 | | _ | & | 6 | F | V | | | | | | | | | | |
| | 7 | 0111 | | ' | | 7 | G | W | | | | | | | | | | |
| | 8 | 1000 | | ( | | 8 | H | X | | | | | | | | | | |
| | 9 | 1001 | | ) | | 9 | I | Y | | | | | | | | | | |
| | A | 1010 | | | + | | J | Z | | | | | | | | | | |
| | B | 1011 | | | | | K | | | | | | | | | | | |
| | C | 1100 | | | , | | L | | | | | | | | | | | |
| | D | 1101 | | | - | | M | | | | | | | | | | | |
| | E | 1110 | | | . | | N | | | | | | | | | | | |
| | F | 1111 | | | / | | O | _ | | | | | | | | | | |

## Standard Character Set III
## EBCDIC

| LEAST SIGNIFICANT BITS | Hex | Binary | MOST SIGNIFICANT BITS 0 (0000) | 1 (0001) | 2 (0010) | 3 (0011) | 4 (0100) | 5 (0101) | 6 (0110) | 7 (0111) | 8 (1000) | 9 (1001) | A (1010) | B (1011) | C (1100) | D (1101) | E (1110) | F (1111) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0000 | | | | | | & | - | | | | | | | | | 0 |
| | 1 | 0001 | | · | | | | | / | | | | | | A | J | | 1 |
| | 2 | 0010 | | | | | | | | | | | | | B | K | S | 2 |
| | 3 | 0011 | | | | | | | | | | | | | C | L | T | 3 |
| | 4 | 0100 | | | | | | | | | | | | | D | M | U | 4 |
| | 5 | 0101 | | | | | | | | | | | | | E | N | V | 5 |
| | 6 | 0110 | | | | | | | | \| | | | | | F | O | W | 6 |
| | 7 | 0111 | | | | | | | | \| | | | | | G | P | X | 7 |
| | 8 | 1000 | | | | | | | | | | | | | H | Q | Y | 8 |
| | 9 | 1001 | | | | | | | | | | | | | I | R | Z | 9 |
| | A | 1010 | | | | | | | | | | | | | | | | |
| | B | 1011 | | | | | | $ | | | | | | ] | | | | |
| | C | 1100 | | | | | < | * | | | | | | | | | | |
| | D | 1101 | | | | | | | - | ' | | | | | | | | |
| | E | 1110 | | | | | + | | > | = | | | | | | | | |
| | F | 1111 | | | | | | | | | | | | _ | | | | |

## Standard Character Set IV
## ASCII

| LEAST SIGNIFICANT BITS | Hex | Binary | MOST SIGNIFICANT BITS 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| | 0 | 0000 | RESERVED FOR XEROX USE ONLY | | | 0 | @ | P | ¢ | | | | | | | | | |
| | 1 | 0001 | | | ! | 1 | A | Q | | | | | | | | | | |
| | 2 | 0010 | | | " | 2 | B | R | | | | | | | | | | |
| | 3 | 0011 | | | # | 3 | C | S | | | | | • | | | | | |
| | 4 | 0100 | | | $ | 4 | D | T | | | | | | | | | | |
| | 5 | 0101 | | | % | 5 | E | U | | | | | | | | | | |
| | 6 | 0110 | | | & | 6 | F | V | | | | | | | | | | |
| | 7 | 0111 | | | ' | 7 | G | W | | | | | | | | | | |
| | 8 | 1000 | | | ( | 8 | H | X | | | | | | | | | | |
| | 9 | 1001 | | | ) | 9 | I | Y | | | | | | | | | | |
| | A | 1010 | | | * | : | J | Z | | | | | | | | | | |
| | B | 1011 | | | + | ; | K | | | | | | | | | | | |
| | C | 1100 | | | , | | L | | | | | | | | | | | |
| | D | 1101 | | | - | = | M | | | | | | | | | | | |
| | E | 1110 | | ¼ | . | | N | | | | | | | | | | | |
| | F | 1111 | | ½ | / | ? | O | _ | | | | | | | | | | |

## Standard Character Set IV
## EBCDIC

| LEAST SIGNIFICANT BITS | Hex | Binary | MOST SIGNIFICANT BITS 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| | 0 | 0000 | | | | | | & | - | | ¼ | ½ | | | | | | 0 |
| | 1 | 0001 | | | | | | | / | | | | | | A | J | | 1 |
| | 2 | 0010 | | | | | | | | | | | | | B | K | S | 2 |
| | 3 | 0011 | | | | | | | | | | | | | C | L | T | 3 |
| | 4 | 0100 | | | | | | | | | | | | | D | M | U | 4 |
| | 5 | 0101 | | | | | | | | | | | | | E | N | V | 5 |
| | 6 | 0110 | | | | | | | | | | | | | F | O | W | 6 |
| | 7 | 0111 | | | | | | | | | | | | | G | P | X | 7 |
| | 8 | 1000 | | | | | | | | | | | | | H | Q | Y | 8 |
| | 9 | 1001 | | | | | | | | ¢ | | | | | I | R | Z | 9 |
| | A | 1010 | | | | | ¢ | ! | | : | | | | | | | | |
| | B | 1011 | | | | | . | $ | , | # | | | | | | | | |
| | C | 1100 | | | | | | * | % | @ | | | | | | | | |
| | D | 1101 | | | | | ( | ) | _ | ' | | | | | | | | |
| | E | 1110 | | | | | + | ; | | = | | | | | | | | |
| | F | 1111 | | | | | | | ? | " | | | | | | | | |

## Standard Character Set V
## ASCII

| LEAST SIGNIFICANT BITS | Hex | MOST SIGNIFICANT BITS | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| | Binary | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| 0 | 0000 | RESERVED FOR XEROX USE ONLY | | | 0 | @ | P | ¢ | | | | | | | | | |
| 1 | 0001 | | | ! | 1 | A | Q | | | | | | | | | | |
| 2 | 0010 | | ¤ | " | 2 | B | R | | | | | | | | | | |
| 3 | 0011 | | | # | 3 | C | S | | | | | | | | | | |
| 4 | 0100 | | | $ | 4 | D | T | | | | | | | | | | |
| 5 | 0101 | | | % | 5 | E | U | | | | | | | | | | |
| 6 | 0110 | | | & | 6 | F | V | | | | | | | | | | |
| 7 | 0111 | | | ' | 7 | G | W | | | | | | | | | | |
| 8 | 1000 | | | ( | 8 | H | X | | | | | | | | | | |
| 9 | 1001 | | | ) | 9 | I | Y | | | | | | | | | | |
| A | 1010 | | | * | : | J | Z | | | | | | | | | | |
| B | 1011 | | ¦ | + | ; | K | | | | | | | | | | | |
| C | 1100 | | | , | < | L | | | | | | | | | | | |
| D | 1101 | | ¬ | - | = | M | | | | | | | | | | | |
| E | 1110 | | | . | > | N | | | | | | | | | | | |
| F | 1111 | | | / | ? | O | _ | | | | | | | | | | |

## Standard Character Set V
## EBCDIC

| LEAST SIGNIFICANT BITS | Hex | MOST SIGNIFICANT BITS | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| | Binary | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| 0 | 0000 | | | | | | & | - | | | | | | | | | 0 |
| 1 | 0001 | | | | | | | / | | | | | | A | J | | 1 |
| 2 | 0010 | | | | | | | | | | | | | B | K | S | 2 |
| 3 | 0011 | | | | | | | | | | | | | C | L | T | 3 |
| 4 | 0100 | | | | | | | | | | | | | D | M | U | 4 |
| 5 | 0101 | | | | | | | | | | | | | E | N | V | 5 |
| 6 | 0110 | | | | | | | | | | | | | F | O | W | 6 |
| 7 | 0111 | | | | | | | | | | | | | G | P | X | 7 |
| 8 | 1000 | | | | | | | | | | | | | H | Q | Y | 8 |
| 9 | 1001 | | | | | | | | ¢ | | | | | I | R | Z | 9 |
| A | 1010 | | | | | ¢ | ! | | : | | | | | | | | ¦ |
| B | 1011 | | | | | . | $ | , | # | | | | | | | | |
| C | 1100 | | | | | < | * | % | @ | | ¤ | | | | | | |
| D | 1101 | | | | | ( | ) | _ | ' | | | | | | | | |
| E | 1110 | | | | | + | ; | > | = | | | | | | | | |
| F | 1111 | | | | | ¦ | ¬ | ? | " | | | | | | | | |

## Standard Character Set VI
## ASCII

| LEAST SIGNIFICANT BITS | Hex | Binary | MOST SIGNIFICANT BITS 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| | 0 | 0000 | RESERVED FOR XEROX USE ONLY | | | 0 | @ | P | ¢ | p | | | | | | | | |
| | 1 | 0001 | | | ! | 1 | A | Q | a | q | | | | | | | | |
| | 2 | 0010 | | | " | 2 | B | R | b | r | | | | | | | | |
| | 3 | 0011 | | | # | 3 | C | S | c | s | | | | | | | | |
| | 4 | 0100 | | | $ | 4 | D | T | d | t | | | | | | | | |
| | 5 | 0101 | | | % | 5 | E | U | e | u | | | | | | | | |
| | 6 | 0110 | | | & | 6 | F | V | f | v | | | | | | | | |
| | 7 | 0111 | | | ' | 7 | G | W | g | w | | | | | | | | |
| | 8 | 1000 | | | ( | 8 | H | X | h | x | | | | | | | | |
| | 9 | 1001 | | | ) | 9 | I | Y | i | y | | | | | | | | |
| | A | 1010 | | | * | : | J | Z | j | z | | | | | | | | |
| | B | 1011 | | | + | ; | K | | k | | | | | | | | | |
| | C | 1100 | | | , | | L | | l | | | | | | | | | |
| | D | 1101 | | | - | = | M | | m | | | | | | | | | |
| | E | 1110 | | ¼ | . | | N | | n | | | | | | | | | |
| | F | 1111 | | ½ | / | ? | O | _ | o | | | | | | | | | |

## Standard Character Set VI
## EBCDIC

| LEAST SIGNIFICANT BITS | Hex | Binary | MOST SIGNIFICANT BITS 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| | 0 | 0000 | | | | | | & | - | | ¼ | ½ | | | | | | 0 |
| | 1 | 0001 | | | | | | | / | | a | j | | | A | J | | 1 |
| | 2 | 0010 | | | | | | | | | b | k | s | | B | K | S | 2 |
| | 3 | 0011 | | | | | | | , | | c | l | t | | C | L | T | 3 |
| | 4 | 0100 | | | | | | | | | d | m | u | | D | M | U | 4 |
| | 5 | 0101 | | | | | | | | | e | n | v | | E | N | V | 5 |
| | 6 | 0110 | | | | | | | | | f | o | w | | F | O | W | 6 |
| | 7 | 0111 | | | | | | | | | g | p | x | | G | P | X | 7 |
| | 8 | 1000 | | | | | | | | | h | q | y | | H | Q | Y | 8 |
| | 9 | 1001 | | | | | | | | ¢ | i | r | z | | I | R | Z | 9 |
| | A | 1010 | | | | | ¢ | ! | | : | | | | | | | | |
| | B | 1011 | | | | | . | $ | , | # | | | | | | | | |
| | C | 1100 | | | | | | * | % | @ | | | | | | | | |
| | D | 1101 | | | | | ( | ) | _ | ' | | | | | | | | |
| | E | 1110 | | | | | + | ; | | = | | | | | | | | |
| | F | 1111 | | | | | | | ? | " | | | | | | | | |

## Standard Character Set VII
## ASCII

| | | MOST SIGNIFICANT BITS | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Hex | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| | Binary | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| LEAST SIGNIFICANT BITS 0 | 0000 | RESERVED FOR XEROX USE ONLY | | | 0 | @ | P | ` | p | | | | | | | | |
| 1 | 0001 | | | ! | 1 | A | Q | a | q | | | | | | | | |
| 2 | 0010 | | | " | 2 | B | R | b | r | | | | | | | | |
| 3 | 0011 | | | # | 3 | C | S | c | s | | | | | | | | |
| 4 | 0100 | | | $ | 4 | D | T | d | t | | | | | | | | |
| 5 | 0101 | | | % | 5 | E | U | e | u | | | | | | | | |
| 6 | 0110 | | | & | 6 | F | V | f | v | | | | | | | | |
| 7 | 0111 | | | ' | 7 | G | W | g | w | | | | | | | | |
| 8 | 1000 | | | ( | 8 | H | X | h | x | | | | | | | | |
| 9 | 1001 | | | ) | 9 | I | Y | i | y | | | | | | | | |
| A | 1010 | | | * | : | J | Z | j | z | | | | | | | | |
| B | 1011 | | | + | ; | K | [ | k | | | | | | | | | |
| C | 1100 | | | , | < | L | \ | l | | | | | | | | | |
| D | 1101 | | | - | = | M | ] | m | | | | | | | | | |
| E | 1110 | | | . | > | N | ^ | n | | | | | | | | | |
| F | 1111 | | | / | ? | O | _ | o | | | | | | | | | |

## Standard Character Set VII
## EBCDIC

| | | MOST SIGNIFICANT BITS | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Hex | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| | Binary | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| LEAST SIGNIFICANT BITS 0 | 0000 | | | | | | & | - | | | | | | { | } | \ | 0 |
| 1 | 0001 | | | | | | | / | | a | j | | | A | J | | 1 |
| 2 | 0010 | | | | | | | | | b | k | s | | B | K | S | 2 |
| 3 | 0011 | | | | | | | | | c | l | t | | C | L | T | 3 |
| 4 | 0100 | | | | | | | | | d | m | u | | D | M | U | 4 |
| 5 | 0101 | | | | | | | | | e | n | v | | E | N | V | 5 |
| 6 | 0110 | | | | | | | | | f | o | w | | F | O, | W | 6 |
| 7 | 0111 | | | | | | | | | g | p | x | | G | P | X | 7 |
| 8 | 1000 | | | | | | | | | h | q | y | | H | Q | Y | 8 |
| 9 | 1001 | | | | | | | | ` | i | r | z | | I | R | Z | 9 |
| A | 1010 | | | | | ` | ! | ^ | : | | | | | | | | ¦ |
| B | 1011 | | | | | . | $ | , | # | ( | ) | | | | | | |
| C | 1100 | | | | | < | * | % | @ | | | | | | | | |
| D | 1101 | | | | | ( | ) | _ | ' | | | [ | ] | | | | |
| E | 1110 | | | | | + | ; | > | = | | | | | | | | |
| F | 1111 | | | | | ¦ | ~ | ? | " | | | | | | | | |

## Standard Character Set VIII
## ASCII

| LEAST SIGNIFICANT BITS | Hex | Binary | MOST SIGNIFICANT BITS 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| | 0 | 0000 | RESERVED FOR XEROX USE ONLY | ≤ | | 0 | @ | P | ¢ | p | ° | † | | | | | | |
| | 1 | 0001 | | ≥ | ! | 1 | A | Q | a | q | ' | ‡ | | | | | | |
| | 2 | 0010 | | ¤ | " | 2 | B | R | b | r | ' | • | | | | | | |
| | 3 | 0011 | | | # | 3 | C | S | c | s | ' | ¤ | | | | | | |
| | 4 | 0100 | | | $ | 4 | D | T | d | t | " | + | | | | | | |
| | 5 | 0101 | | | % | 5 | E | U | e | u | " | ' | | | | | | |
| | 6 | 0110 | | _ | & | 6 | F | V | f | v | " | | | | | | | |
| | 7 | 0111 | | ⌴ | ' | 7 | G | W | g | w | · | | | | | | | |
| | 8 | 1000 | | | ( | 8 | H | X | h | x | ° | | | | | | | |
| | 9 | 1001 | | | ) | 9 | I | Y | i | y | ° | | | | | | | |
| | A | 1010 | | ( | * | : | J | Z | j | z | · | | | | | | | |
| | B | 1011 | | \| | + | ; | K | [ | k | ' | ' | | | | | | |
| | C | 1100 | | ) | . | < | L | \ | l | ± | | | | | | | | |
| | D | 1101 | | ¬ | - | = | M | ] | m | ⌙ | ' | | | | | | | |
| | E | 1110 | | | . | > | N | ^ | n | ⌐ | ¶ | | | | | | | |
| | F | 1111 | | | / | ? | O | _ | o | ¬ | ∮ | | | | | | | |

## Standard Character Set VIII
## EBCDIC

| LEAST SIGNIFICANT BITS | Hex | Binary | MOST SIGNIFICANT BITS 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| | 0 | 0000 | | | | | | & | - | | | | | • | ( | ) | \ | 0 |
| | 1 | 0001 | | | | | | | / | | a | j | ' | ' | A | J | | 1 |
| | 2 | 0010 | | | | | | | | | b | k | s | ' | B | K | S | 2 |
| | 3 | 0011 | | | | | | | | | c | l | t | ' | C | L | T | 3 |
| | 4 | 0100 | | | | | | | | | d | m | u | " | D | M | U | 4 |
| | 5 | 0101 | | | | | | | | | e | n | v | " | E | N | V | 5 |
| | 6 | 0110 | | | | | | | | | f | o | w | " | F | O | W | 6 |
| | 7 | 0111 | | | | | | | | | g | p | x | ' | G | P | X | 7 |
| | 8 | 1000 | | | | | | | | | h | q | y | " | H | Q | Y | 8 |
| | 9 | 1001 | | | | | | | | ¢ | i | r | z | " | I | R | Z | 9 |
| | A | 1010 | | | | | ¢ | ! | ' | : | | † | ‡ | | | | | \| |
| | B | 1011 | | | | | . | $ | , | # | ( | ) | ⌙ | ⌴ | | | | |
| | C | 1100 | | | | | < | * | % | @ | ≤ | ¤ | ⌐ | ¬ | | ∮ | | |
| | D | 1101 | | | | | ( | ) | _ | ' | · | · | [ | ] | | ¶ | | |
| | E | 1110 | | | | | + | ; | > | = | ' | ± | ≥ | ≠ | | | | |
| | F | 1111 | | | | | \| | ¬ | ? | " | + | • | • | _ | | | | |

## Standard Character Set IX
## ASCII

| | | MOST SIGNIFICANT BITS | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Hex | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| | Binary | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| LEAST SIGNIFICANT BITS | | | | | | | | | | | | | | | | | |
| 0 | 0000 | RESERVED FOR XEROX USE ONLY | | | | | ┬ | | | | | | | | | | |
| 1 | 0001 | | | | ┬ | ┌ | ┴ | | | | | | | | | | |
| 2 | 0010 | | | | ┴ | ┐ | ├ | | | | | | | | | | |
| 3 | 0011 | | | | ├ | ┘ | ┤ | | | | | | | | | | |
| 4 | 0100 | | | | ┤ | └ | ─ | | | | | | | | | | |
| 5 | 0101 | | | | ┼ | ┬ | ─ | | | | | | | | | | |
| 6 | 0110 | | | | ─ | ┴ | │ | | | | | | | | | | |
| 7 | 0111 | | | | │ | ├ | ┌ | | | | | | | | | | |
| 8 | 1000 | | | | : | ┤ | ┐ | | | | | | | | | | |
| 9 | 1001 | | | | : | ┼ | ┘ | | | | | | | | | | |
| A | 1010 | | | | | ─ | └ | | | | | | | | | | |
| B | 1011 | | | | | │ | | | | | | | | | | | |
| C | 1100 | | | | | ┌ | | | | | | | | | | | |
| D | 1101 | | | | | ┐ | | | | | | | | | | | |
| E | 1110 | | | | | ┘ | | | | | | | | | | | |
| F | 1111 | | | | | └ | | | | | | | | | | | |

## Standard Character Set IX
## EBCDIC

| | | MOST SIGNIFICANT BITS | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Hex | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| | Binary | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| LEAST SIGNIFICANT BITS | | | | | | | | | | | | | | | | | |
| 0 | 0000 | | | | | | | | | | | | | | | | |
| 1 | 0001 | | | | | | | | | | | | | ┌ | | | ┬ |
| 2 | 0010 | | | | | | | | | | | | | ┐ | │ | ┤ | ┴ |
| 3 | 0011 | | | | | | | | | | | | | : | ┌ | ┼ | ├ |
| 4 | 0100 | | | | | | | | | | | | | ┘ | ┐ | ─ | ┤ |
| 5 | 0101 | | | | | | | | | | | | | ┬ | ┘ | │ | ┼ |
| 6 | 0110 | | | | | | | | | | | | | └ | └ | ┌ | ─ |
| 7 | 0111 | | | | | | | | | | | | | ├ | ┬ | ┐ | │ |
| 8 | 1000 | | | | | | | | | | | | | ┤ | ┴ | ┘ | : |
| 9 | 1001 | | | | | | | | | | | | | ┼ | ├ | └ | : |
| A | 1010 | | | | | | | | | | | | | | | | |
| B | 1011 | | | | | | | | | | | | | | | | |
| C | 1100 | | | | | | | | | | | | | | | | |
| D | 1101 | | | | | | | | | | | | | | | | |
| E | 1110 | | | | | | | | | | | | | | | | |
| F | 1111 | | | | | | | | | | | | | | | | |

## Xerox 9700 EBCDIC to Extended ASCII
## Hex Translation Values

| | | | MOST SIGNIFICANT BITS | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Hex | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| | | Binary | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| LEAST SIGNIFICANT BITS | 0 | 0000 | 20 | FA | EA | DA | 20 | 26 | 2D | BO | 1E | 1F | 8C | 80 | 1A | 1C | 5C | 30 |
| | 1 | 0001 | 20 | F9 | E9 | D9 | CA | C1 | 2F | AF | 61 | 6A | 7B | 81 | 41 | 4A | 9F | 31 |
| | 2 | 0010 | 20 | F8 | E8 | D8 | C9 | CO | B8 | AE | 62 | 6B | 73 | 82 | 42 | 4B | 53 | 32 |
| | 3 | 0011 | 20 | F7 | E7 | D7 | C8 | BF | B7 | AD | 63 | 6C | 74 | 83 | 43 | 4C | 54 | 33 |
| | 4 | 0100 | 20 | F6 | E6 | D6 | C7 | BE | B6 | AC | 64 | 6D | 75 | 84 | 44 | 4D | 55 | 34 |
| | 5 | 0101 | 20 | F5 | E5 | D5 | C6 | BD | B5 | AB | 65 | 6E | 76 | 85 | 45 | 4E | 56 | 35 |
| | 6 | 0110 | 20 | F4 | E4 | D4 | C5 | BC | B4 | 19 | 66 | 6F | 77 | 86 | 46 | 4F | 57 | 36 |
| | 7 | 0111 | 20 | F3 | E3 | D3 | C4 | BB | B3 | 18 | 67 | 70 | 78 | 87 | 47 | 50 | 58 | 37 |
| | 8 | 1000 | 20 | F2 | E2 | D2 | C3 | BA | B2 | AA | 68 | 71 | 79 | 88 | 48 | 51 | 59 | 38 |
| | 9 | 1001 | 20 | F1 | E1 | D1 | C2 | B9 | B1 | 60 | 69 | 72 | 7A | 89 | 49 | 52 | 5A | 39 |
| | A | 1010 | 20 | FO | EO | DO | 60 | 21 | 5E | 3A | A9 | 90 | 91 | A8 | A7 | A3 | 9E | 1B |
| | B | 1011 | FF | EF | DF | CF | 2E | 24 | 2C | 23 | 1A | 1C | 7D | 17 | A6 | A2 | 9D | 99 |
| | C | 1100 | FE | EE | DE | CE | 3C | 2A | 25 | 40 | 10 | 12 | 7E | 7F | 13 | 8F | 15 | 98 |
| | D | 1101 | FD | ED | DD | CD | 28 | 29 | 5F | 27 | 8A | 8D | 5B | 5D | A5 | 8E | 9C | 97 |
| | E | 1110 | FC | EC | DC | CC | 2B | 3B | 3E | 3D | 8B | 7C | 11 | 95 | 14 | A1 | 9B | 96 |
| | F | 1111 | FB | EB | DB | CB | 1B | 1D | 3F | 22 | 94 | 93 | 92 | 16 | A4 | AO | 9A | 20 |

## IBM BCD Table (6-Bit Representation)[t]

| | | | Most Significant Bits | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Octal | (columns) | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| | (rows) | Binary | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
| Least Significant Bits | 0 | 000 | | 8 | | Y | – | Q | & | H |
| | 1 | 001 | 1 | 9 | / | Z | J | R | A | I |
| | 2 | 010 | 2 | 0 | S | | K | | B | & |
| | 3 | 011 | 3 | # | T | , | L | $ | C | . |
| | 4 | 100 | 4 | @ | U | % | M | * | D | < |
| | 5 | 101 | 5 | ' | V | _ | N | ) | E | ( |
| | 6 | 110 | 6 | = | W | > | O | ; | F | + |
| | 7 | 111 | 7 | | X | ? | P | | G | |

Note: 0'20' is the official blank character.

[t]Corresponds to BCD code set used by IBM users and defined by PDL command CODE = IBMBCD.

## Honeywell 200/2000 BCD Table (6-Bit Representation)[t]

| | | | Most Significant Bits | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Octal | (columns) | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| | (rows) | Binary | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
| Least Significant Bits | 0 | 000 | 0 | 8 | + | H | – | Q | < | Y |
| | 1 | 001 | 1 | 9 | A | I | J | R | / | Z |
| | 2 | 010 | 2 | ' | B | ; | K | # | S | @ |
| | 3 | 011 | 3 | = | C | . | L | $ | T | , |
| | 4 | 100 | 4 | : | D | ) | M | * | U | ( |
| | 5 | 101 | 5 | | E | % | N | " | V | { |
| | 6 | 110 | 6 | > | F | [ | O | ] | W | } |
| | 7 | 111 | 7 | & | G | ? | P | ! | X | |

Notes: 0'15' is the official blank character; 0'77' is the padding character.

[t]Corresponds to BCD code set used by Honeywell 200/2000 users and defined by PDL command CODE = H2BCD.

## Honeywell 6000 BCD Table (6-Bit Representation)[t]

| | | | Most Significant Bits | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Octal | (columns) | ø | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| | (rows) | Binary | øøø | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
| Least Significant Bits | ø | øøø | ø | 8 | space | H | ¬ 1 / ^ 2 | Q | + | Y |
| | 1 | øø1 | 1 | 9 | A | I | J | R | / | ≠ |
| | 2 | ø1ø | 2 | [ | B | & | K | - | S | _ |
| | 3 | ø11 | 3 | # | C | . | L | $ | T | , |
| | 4 | 1øø | 4 | @ | D | ] | M | * | U | % |
| | 5 | 1ø1 | 5 | : | E | ( | N | ) | V | = |
| | 6 | 11ø | 6 | > | F | < | O | ; | W | " |
| | 7 | 111 | 7 | ? | G | \ | P | ' | X | ! |

Notes: 1. Usual BCD Character

2. Printer Character

---

[t]Corresponds to BCD code set used by Honeywell 600/6000 series SSF tapes and defined by PDL command CODE = BCD or CODE = H6BCD.

## Fieldata Translation Table

| | | | Most Significant Bits | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Octal | (columns) | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| | (rows) | Binary | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
| Least Significant Bits | 0 | 000 | @ | C | K | S | ) | * | 0 | 8 |
| | 1 | 001 | [ | D | L | T | - | ( | 1 | 9 |
| | 2 | 010 | ] | E | M | U | + | % | 2 | ' |
| | 3 | 011 | # | F | N | V | < | : | 3 | ; |
| | 4 | 100 | ^ | G | O | W | = | ? | 4 | / |
| | 5 | 101 | (blank) | H | P | X | > | ! | 5 | . |
| | 6 | 110 | A | I | Q | Y | & | , | 6 | " |
| | 7 | 111 | B | J | R | Z | $ | \ | 7 | _ |

# Univac ASCII Character Set

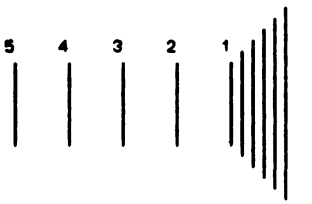
| Octal | Character | Octal | Character | Octal | Character |
|---|---|---|---|---|---|
| 040 | (blank) | 100 | @ | 140 | \` |
| 041 | ! | 101 | A | 141 | a |
| 042 | " | 102 | B | 142 | b |
| 043 | # | 103 | C | 143 | c |
| 044 | $ | 104 | D | 144 | d |
| 045 | % | 105 | E | 145 | e |
| 046 | & | 106 | F | 146 | f |
| 047 | ' | 107 | G | 147 | g |
| 050 | ( | 110 | H | 150 | h |
| 051 | ) | 111 | I | 151 | i |
| 052 | * | 112 | J | 152 | j |
| 053 | + | 113 | K | 153 | k |
| 054 | , | 114 | L | 154 | l |
| 055 | – | 115 | M | 155 | m |
| 056 | . | 116 | N | 156 | n |
| 057 | / | 117 | O | 157 | o |
| 060 | 0 | 120 | P | 160 | p |
| 061 | 1 | 121 | Q | 161 | q |
| 062 | 2 | 122 | R | 162 | r |
| 063 | 3 | 123 | S | 163 | s |
| 064 | 4 | 124 | T | 164 | t |
| 065 | 5 | 125 | U | 165 | u |
| 066 | 6 | 126 | V | 166 | v |
| 067 | 7 | 127 | W | 167 | w |
| 070 | 8 | 130 | X | 170 | x |
| 071 | 9 | 131 | Y | 171 | y |
| 072 | : | 132 | Z | 172 | z |
| 073 | ; | 133 | [ | 173 | { |
| 074 | < | 134 | \ | 174 | ¦ |
| 075 | = | 135 | ] | 175 | } |
| 076 | > | 136 | ^ | 176 | ~ |
| 077 | ? | 137 | _ | 177 | (null) |

# APPENDIX D. FDL ALIGNMENT FORM

The FDL form on the following page should be used to aid in the alignment of the laser image with the paper. This form (filename TEST) is resident on a 9700 system in the FDL directory. See section "Setting Physical Page Alignment" in Chapter 7 for further details on how to use this form.

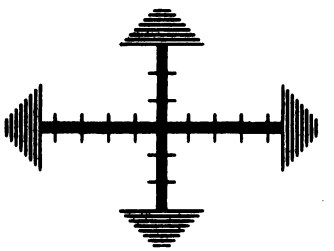


SUBTRACT
SCAN LINES




THIS IS THE
LEFT EDGE
OF THE FORM

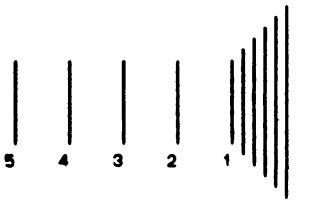
SUBTRACT
DOTS



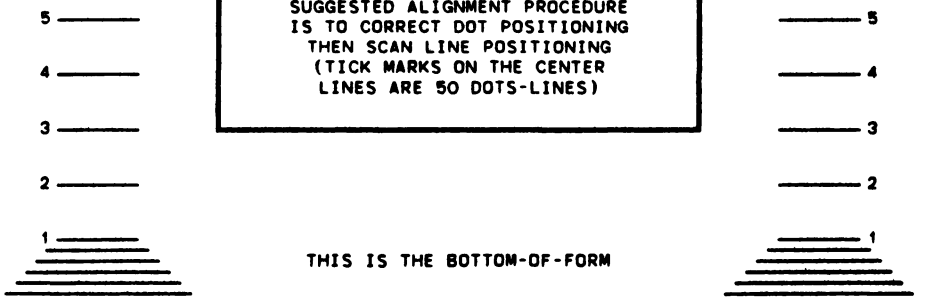
ADD
DOTS

THIS IS THE
RIGHT EDGE
OF THE FORM


ADD
SCAN LINES




SUGGESTED ALIGNMENT PROCEDURE
IS TO CORRECT DOT POSITIONING
THEN SCAN LINE POSITIONING
(TICK MARKS ON THE CENTER
LINES ARE 50 DOTS-LINES)



```
 1  FORM TEST; LANDSCAPE; GRID 1 INCHES; FONT LO112B;                          000010
 2  AT .1,.1 BOX 10.8 WIDE 8.3 HIGH USING SOLID 0;                             000020
 3  AT 4.53 DRAW 2 VERTICAL LINES FROM 4.2 TO 4.3 AND EVERY 1.94;              000030
 4  AT 4.56 DRAW 2 VERTICAL LINES FROM 4.17 TO 4.33 AND EVERY 1.88;            000040
 5  AT 4.59 DRAW 2 VERTICAL LINES FROM 4.14 TO 4.36 AND EVERY 1.82;            000050
 6  AT 4.62 DRAW 2 VERTICAL LINES FROM 4.11 TO 4.39 AND EVERY 1.76;            000060
 7  AT 4.65 DRAW 2 VERTICAL LINES FROM 4.08 TO 4.42 AND EVERY 1.70;            000070
 8  AT 4.68 DRAW 2 VERTICAL LINES FROM 4.05 TO 4.45 AND EVERY 1.64;            000080
 9  AT 4.71 DRAW 2 VERTICAL LINES FROM 4.02 TO 4.48 AND EVERY 1.58;            000090
10  AT 4.74 DRAW 2 VERTICAL LINES FROM 3.99 TO 4.51 AND EVERY 1.52;            000100
11  AT 3.53 DRAW 2 HORIZONTAL LINES FROM 5.45 TO 5.55 AND EVERY 1.44;          000110
12  AT 3.56 DRAW 2 HORIZONTAL LINES FROM 5.42 TO 5.58 AND EVERY 1.38;          000120
13  AT 3.59 DRAW 2 HORIZONTAL LINES FROM 5.39 TO 5.61 AND EVERY 1.32;          000130
14  AT 3.62 DRAW 2 HORIZONTAL LINES FROM 5.36 TO 5.64 AND EVERY 1.26;          000140
15  AT 3.65 DRAW 2 HORIZONTAL LINES FROM 5.33 TO 5.67 AND EVERY 1.20;          000150
16  AT 3.68 DRAW 2 HORIZONTAL LINES FROM 5.30 TO 5.70 AND EVERY 1.14;          000160
17  AT 3.71 DRAW 2 HORIZONTAL LINES FROM 5.27 TO 5.73 AND EVERY 1.08;          000170
18  AT 3.74 DRAW 2 HORIZONTAL LINES FROM 5.24 TO 5.76 AND EVERY 1.02;          000180
19  AT 3.5,2.5 BOX 1.5 WIDE 1.5 HIGH USING SOLID 0;                            000190
20  AT 3.5,7 BOX 1.5 WIDE 1.5 HIGH USING SOLID 0;                              000200
21  AT 2.25,4.5 BOX 2 WIDE .75 HIGH USING SOLID 0;                             000210
22  AT 5.5,4.5 BOX 2 WIDE .75 HIGH USING SOLID 0;                              000220
23  AT 6.5,4 BOX 3 WIDE 1 HIGH USING SOLID 2;                                  000230
24  TEXT ALIGNED RIGHT IN CENTER RIGHT BOX 3.5,2.5 'SUBTRACT'  'DOTS';         000240
25  TEXT ALIGNED LEFT IN CENTER LEFT BOX 3.5,7 'ADD'  'DOTS';                  000250
26  TEXT IN BOTTOM CENTER BOX 2.25,4.5 'SUBTRACT'  'SCAN LINES';               000260
27  TEXT IN TOP CENTER BOX 5.5,4.5 'ADD'  'SCAN LINES';                        000270
28  TEXT IN TOP CENTER BOX .1,.1 'THIS IS THE TOP-OF-FORM';                    000280
29  TEXT IN BOTTOM CENTER BOX .1,.1 'THIS IS THE BOTTOM-OF-FORM ;              000290
30  TEXT LEFT IN CENTER LEFT BOX .1,.1                                         000300
31        'THIS IS THE'  'LEFT EDGE'  'OF THE FORM';                           000310
32  TEXT RIGHT IN CENTER RIGHT BOX .1,.1                                       000320
33        'THIS IS THE'  'RIGHT EDGE'  'OF THE FORM';                          000330
34  TEXT IN BOX 6.5,4 'SUGGESTED ALIGNMENT PROCEDURE'                          000340
35        'IS TO CORRECT DOT POSITIONING'  'THEN SCAN LINE POSITIONING'        000350
36        '(TICK MARKS ON THE CENTER'  'LINES ARE 50 DOTS-LINES)';             000360
37  GRID IS DOTS;                                                              000370
```

```
38  AT 1269 DRAW 3 HORIZONTAL LINES FROM 1424 TO 1876 USING SOLID 2          000380
39        AND EVERY 6;                                                      000390
40  AT 1644 DRAW 3 VERTICAL LINES FROM 1124 TO 1426 USING SOLID 2           000400
41        AND EVERY 6;                                                      000410
42  AT 1450 DRAW 9 VERTICAL LINES FROM 1250 TO 1300 AND EVERY 50;           000420
43  AT 1175 DRAW 5 HORIZONTAL LINES FROM 1625 TO 1675 AND EVERY 50;         000430
44  AT 100 DRAW 5 VERTICAL LINES FROM 750 TO 900 AND EVERY 100;             000440
45  AT 100 DRAW 5 VERTICAL LINES FROM 1650 TO 1800 AND EVERY 100;           000450
46  AT 2800 DRAW 5 VERTICAL LINES FROM 750 TO 900 AND EVERY 100;            000460
47  AT 2800 DRAW 5 VERTICAL LINES FROM 1650 TO 1800 AND EVERY 100;          000470
48  AT 100 DRAW 5 HORIZONTAL LINES FROM 900 TO 1050 AND EVERY 100;          000480
49  AT 100 DRAW 5 HORIZONTAL LINES FROM 2250 TO 2400 AND EVERY 100;         000490
50  AT 2050 DRAW 5 HORIZONTAL LINES FROM 900 TO 1050 AND EVERY 100;         000500
51  AT 2050 DRAW 5 HORIZONTAL LINES FROM 2250 TO 2400 AND EVERY 100;        000510
52  TEXT AT 700,100 '1'; TEXT AT 700,200 '2'; TEXT AT 700,300 '3';          000520
53  TEXT AT 700,400 '4'; TEXT AT 700,500 '5';                               000530
54  TEXT AT 700,2778 '5'; TEXT AT 700,2878 '4'; TEXT AT 700,2978 '3';       000540
55  TEXT AT 700,3078 '2'; TEXT AT 700,3178 '1';                             000550
56  TEXT AT 1813,100 '1'; TEXT AT 1813,200 '2'; TEXT AT 1813,300 '3';       000560
57  TEXT AT 1813,400 '4'; TEXT AT 1813,500 '5';                             000570
58  TEXT AT 1813,2778 '5'; TEXT AT 1813,2878 '4'; TEXT AT 1813,2978 '3';    000580
59  TEXT AT 1813,3078 '2'; TEXT AT 1813,3178 '1';                           000590
60  TEXT AT 100,865 '1'; TEXT AT 200,865 '2'; TEXT AT 300,865 '3';          000600
61  TEXT AT 400,865 '4'; TEXT AT 500,865 '5';                               000610
62  TEXT AT 100,2413 '1'; TEXT AT 200,2413 '2'; TEXT AT 300,2413 '3';       000620
63  TEXT AT 400,2413 '4'; TEXT AT 500,2413 '5';                             000630
64  TEXT AT 2028,865 '5'; TEXT AT 2128,865 '4'; TEXT AT 2228,865 '3';       000640
65  TEXT AT 2328,865 '2'; TEXT AT 2428,865 '1';                             000650
66  TEXT AT 2028,2413 '5'; TEXT AT 2128,2413 '4'; TEXT AT 2228,2413 '3';    000660
67  TEXT AT 2328,2413 '2'; TEXT AT 2428,2413 '1';                           000670
68  AT 0 DRAW 2 VERTICAL LINES FROM 650 TO 1000 AND VERTICALLY EVERY 900;   000680
69  AT 20 DRAW 2 VERTICAL LINES FROM 670 TO 980 AND VERTICALLY EVERY 900;   000690
70  AT 40 DRAW 2 VERTICAL LINES FROM 690 TO 960 AND VERTICALLY EVERY 900;   000700
71  AT 60 DRAW 2 VERTICAL LINES FROM 710 TO 940 AND VERTICALLY EVERY 900;   000710
72  AT 80 DRAW 2 VERTICAL LINES FROM 730 TO 920 AND VERTICALLY EVERY 900;   000720
73  AT 3220 DRAW 2 VERTICAL LINES FROM 730 TO 920 AND VERTICALLY EVERY 900; 000730
74  AT 3240 DRAW 2 VERTICAL LINES FROM 710 TO 940 AND VERTICALLY EVERY 900; 000740
```

```
 75  AT 3260 DRAW 2 VERTICAL LINES FROM 690 TO 960 AND VERTICALLY EVERY 900;   000750
 76  AT 3280 DRAW 2 VERTICAL LINES FROM 670 TO 980 AND VERTICALLY EVERY 900;   000760
 77  AT 3300 DRAW 2 VERTICAL LINES FROM 650 TO 1000                            000770
 78        AND VERTICALLY EVERY 900;                                           000780
 79  AT 0 DRAW 2 HORIZONTAL LINES FROM 800 TO 1150 AND HORIZONTALLY            000790
 80        EVERY 1350;                                                         000800
 81  AT 20 DRAW 2 HORIZONTAL LINES FROM 820 TO 1130 AND HORIZONTALLY           000810
 82        EVERY 1350;                                                         000820
 83  AT 40 DRAW 2 HORIZONTAL LINES FROM 840 TO 1110 AND HORIZONTALLY           000830
 84        EVERY 1350;                                                         000840
 85  AT 60 DRAW 2 HORIZONTAL LINES FROM 860 TO 1090 AND HORIZONTALLY           000850
 86        EVERY 1350;                                                         000860
 87  AT 80 DRAW 2 HORIZONTAL LINES FROM 880 TO 1070 AND HORIZONTALLY           000870
 88        EVERY 1350;                                                         000880
 89  AT 2470 DRAW 2 HORIZONTAL LINES FROM 880 TO 1070 AND HORIZONTALLY         000890
 90        EVERY 1350;                                                         000900
 91  AT 2490 DRAW 2 HORIZONTAL LINES FROM 860 TO 1090 AND HORIZONTALLY         000910
 92        EVERY 1350;                                                         000920
 93  AT 2510 DRAW 2 HORIZONTAL LINES FROM 840 TO 1110 AND HORIZONTALLY         000930
 94        EVERY 1350;                                                         000940
 95  AT 2530 DRAW 2 HORIZONTAL LINES FROM 820 TO 1130 AND HORIZONTALLY         000950
 96        EVERY 1350;                                                         000960
 97  AT 2550 DRAW 2 HORIZONTAL LINES FROM 800 TO 1150 AND HORIZONTALLY         000970
 98        EVERY 1350;                                                         000980
 99  END;END;                                                                  000990
```

# APPENDIX E. HOST SYSTEM JDLS

Appendix E contains listings of host system Job Descriptor Libraries as supplied on the OSS system tape. After a 9700 software system is created by the user these JDLs will be resident in the JDL directory.

```
.......IBMRCA.JSL COMPILED BY PDL(AOO ) AT 08:36 ON 22-MAR-79.....                1
1.......10.......20.......30.......40.......50.......60.......70.

IBMRCA:JDL;                                                                    000010
   /*                                                                  */     000020
   /*    SYSTEM TO PRINT IBM OS STANDARD AND DOS STANDARD             */     000030
   /*    LABELED TAPES, IBM ANSI LABELED TAPES, IBM OS                */     000040
   /*    WRITER TAPES, AND US70 (RCA) LABELED TAPES.                  */     000050
   /*                                                                 */     000060
V1:       VFU       ASSIGN=(1,5), ASSIGN=(2,10), ASSIGN=(3,15),               000070
                    ASSIGN=(4,20), ASSIGN=(5,25), ASSIGN=(6,30),              000080
                    ASSIGN=(7,35), ASSIGN=(8,40), ASSIGN=(9,45),              000090
                    ASSIGN=(10,50), ASSIGN=(11,55), ASSIGN=(12,60),           000100
                    TOF=5, BOF=66;                                            000110
   /*                                                                  */     000120
   /*    VFU FOR OS WRITER WITH CHANNEL 9 ASSIGNED TO LINE 66         */     000130
   /*                                                                 */     000140
WR:       VFU       ASSIGN=(1,5), ASSIGN=(2,10), ASSIGN=(3,15),               000150
                    ASSIGN=(4,20), ASSIGN=(5,25), ASSIGN=(6,30),              000160
                    ASSIGN=(7,35), ASSIGN=(8,40), ASSIGN=(10,45),             000170
                    ASSIGN=(11,50), ASSIGN=(12,60), ASSIGN=(9,66),            000180
                    TOF=5, BOF=66;                                            000190
   /*                                                                  */     000200
          VOLUME    HOST=IBMOS, LABEL=STANDARD, CODE=EBCDIC, PLABEL=YES;      000210
          LINE      DATA=(1,132), PCCTYPE=ANSI, PCC=(0,NOTRAN),               000220
                    OVERPRINT=(MERGE,NODISP), VFU=V1;                         000230
          ACCT      USER=(BIN,TRAY);                                          000240
/*                                                                  */     000241
/*     CATALOG FOR OS VARIABLE BLOCKED TAPES                        */     000242
/*                                                                  */     000244
OSVB:CATALOG;                                                                  000245
          BLOCK     LENGTH=2660, PREAMBLE=4, LTHFLD=2, FORMAT=BIN;            000250
          RECORD    LENGTH=136, PREAMBLE=4, STRUCTURE=VB,                     000260
                    LTHFLD=2, OFFSET=0, FORMAT=BIN;                           000270
   /*                                                                  */     000280
   /*     CATALOG FOR OS WRITER TAPES                                 */     000290
   /*                                                                 */     000300
OS:CATALOG;                                                                    000310
          VOLUME    HOST=OSWTR, OSCHN=9, OSTLP=0, OSHDP=1,                    000320
                    PLABEL=YES;                                               000330
          BLOCK     LENGTH=2400, PREAMBLE=4, LTHFLD=2, FORMAT=BIN;            000340
          RECORD    LENGTH=136, PREAMBLE=4, STRUCTURE=VB,                     000350
                    LTHFLD=2, FORMAT=BIN;                                     000360
          LINE      DATA=(1,132), PCCTYPE=ANSI, PCC=(0,NOTRAN),               000370
                    OVERPRINT=(MERGE,NODISP), VFU=WR;                         000380
   /*                                                                  */     000390
   /*         CATALOG FOR UNIVAC SERIES 70                            */     000400
   /*              (FORMERLY RCA)                                     */     000410
   /*                                                                 */     000420
US:CATALOG;                                                                    000430
          VOLUME    HOST=US70, LABEL=STANDARD, PLABEL=YES;                    000440
          BLOCK     LENGTH=1330, PREAMBLE=0;                                  000450
          RECORD    LENGTH=133, STRUCTURE=FB;                                 000460
          LINE      DATA=(1,132), PCCTYPE=US70, PCC=(0,NOTRAN),               000470
                    OVERPRINT=(MERGE,NODISP), VFU=V1;                         000480
   /*                                                                  */     000490
   /*              IBM OS/DOS STANDARD LABELED TAPES                  */     000500
   /*              ----------------------------------------           */     000510
   /*                                                                 */     000520
   /*    THE FOLLOWING JDES PROVIDE SUPPORT FOR IBM OS STANDARD       */     000530
  /*     AND IBM DOS STANDARD LABELED TAPES                           */     000540
   /*                                                                 */     000550
   /*       CHARACTERISTICS                      JOB                  */     000560
   /*       ---------------                     --------              */     000570
   /*                                                                 */     000580
   /*       OS STANDARD LABELS, 1403 PCC          1                   */     000590
   /*       OS STANDARD LABELS, ANSI PCC          2                   */     000600
   /*       OS STANDARD LABELS, 1401 PCC          3                   */     000610
   /*       OS STANDARD LABELS, NO PCC            4                   */     000620
   /*       DOS STANDARD LABELS, 1403 PCC         5                   */     000630
   /*       DOS STANDARD LABELS, ANSI PCC         6                   */     000640
   /*       DOS STANDARD LABELS, 1401 PCC         7                   */     000650
   /*       DOS STANDARD LABELS, NO PCC           8                   */     000660
```

# IBM RCA (cont.)

```
.....IBMRCA.JSL COMPILED BY PDL(AOO ) AT 08:36 ON 22-MAR-79.....                2
1.....10.....20.....30.....40.....50.....60.....70.

   /*                                                                */     000670
1:JOB   INCLUDE=(OSVB);                                                    000680

BLOCK LENGTH=2660,PREAMBLE=4,LTHFLD=2,FORMAT=BIN;RECORD LENGTH=
136,PREAMBLE=4,STRUCTURE=VB,LTHFLD=2,OFFSET=0,FORMAT=BIN;

         :VOLUME    HOST=IBMOS, LABEL=STANDARD, CODE=EBCDIC;               000690
          LINE      PCCTYPE=IBM1403;                                       000700
2:JOB   INCLUDE=(OSVB);                                                    000710

BLOCK LENGTH=2660,PREAMBLE=4,LTHFLD=2,FORMAT=BIN;RECORD LENGTH=
136,PREAMBLE=4,STRUCTURE=VB,LTHFLD=2,OFFSET=0,FORMAT=BIN;

          VOLUME    HOST=IBMOS, LABEL=STANDARD, CODE=EBCDIC;               000720
          LINE      PCCTYPE=ANSI;                                          000730
3:JOB   INCLUDE=(OSVB);                                                    000740

BLOCK LENGTH=2660,PREAMBLE=4,LTHFLD=2,FORMAT=BIN;RECORD LENGTH=
136,PREAMBLE=4,STRUCTURE=VB,LTHFLD=2,OFFSET=0,FORMAT=BIN;

          VOLUME    HOST=IBMOS, LABEL=STANDARD, CODE=EBCDIC;               000750
          LINE      PCCTYPE=IBM1401;                                       000760
4:JOB   INCLUDE=(OSVB);                                                    000770

BLOCK LENGTH=2660,PREAMBLE=4,LTHFLD=2,FORMAT=BIN;RECORD LENGTH=
136,PREAMBLE=4,STRUCTURE=VB,LTHFLD=2,OFFSET=0,FORMAT=BIN;

          VOLUME    HOST=IBMOS, LABEL=STANDARD, CODE=EBCDIC;               000780
          LINE      PCCTYPE=NONE;                                          000790
5:JOB;                                                                     000800
          VOLUME    HOST=IBMDOS, LABEL=STANDARD, CODE=EBCDIC;              000810
          LINE      PCCTYPE=IBM1403;                                       000820
6:JOB;                                                                     000830
          VOLUME    HOST=IBMDOS, LABEL=STANDARD, CODE=EBCDIC;              000840
          LINE      PCCTYPE=ANSI;                                          000850
7:JOB;                                                                     000860
          VOLUME    HOST=IBMDOS,LABEL=STANDARD, CODE=EBCDIC;               000870
          LINE      PCCTYPE=IBM1401;                                       000880
8:JOB;                                                                     000890
          VOLUME    HOST=IBMDOS, LABEL=STANDARD, CODE=EBCDIC;              000900
          LINE      PCCTYPE=NONE;                                          000910
   /*                                                               */     000920
   /*            IBM ANSI LABELED AND OS WRITER TAPES               */     000930
   /*            -------------------------------------------        */     000940
   /*                                                               */     000950
   /*    THE FOLLOWING JDES PROVIDE SUPPORT FOR IBM ANSI LABELED    */     000960
   /*   TAPES AND OS WRITER TAPES.                                  */     000970
   /*                                                               */     000980
    /*       CHARACTERISTICS                        JOB             */     000990
    /*       ---------------                      --------          */     001000
    /*                                                               */     001010
    /*       ANSI LABELS, ASCII CODE, ANSI PCC       21             */     001020
    /*       ANSI LABELS, ASCII CODE, NO PCC         22             */     001030
    /*       OS WRITER, ANSI PCC                     23             */     001040
    /*       OS WRITER, 1403 PCC                     24             */     001050
   /*                                                               */     001060
21:JOB;                                                                    001070
          VOLUME    HOST=IBMOS, LABEL=ANSI, CODE=ASCII, LCODE=ASCII;       001080
          LINE      PCCTYPE=ANSI;                                          001090
22:JOB;                                                                    001100
          VOLUME    HOST=IBMOS, LABEL=ANSI, CODE=ASCII, LCODE=ASCII;       001110
          LINE      PCCTYPE=NONE;                                          001120
23:JOB    INCLUDE=(OS);                                                    001130

VOLUME HOST=OSWTR,OSCHN=9,OSTLP=0,OSHDP=1,PLABEL=YES;BLOCK
LENGTH=2400,PREAMBLE=4,LTHFLD=2,FORMAT=BIN;RECORD LENGTH=136,
PREAMBLE=4,STRUCTURE=VB,LTHFLD=2,FORMAT=BIN;LINE DATA=(1,132),
PCCTYPE=ANSI,PCC=(0,NOTRAN),OVERPRINT=(MERGE,NODISP),VFU=WR;

          VOLUME    HOST=OSWTR;                                            001140
```

# IBM RCA (cont.)

```
.....IBMRCA.JSL COMPILED BY PDL(A00 ) AT 08:36 ON 22-MAR-79.....                3
1.....10.....20.....30.....40.....50.....60.....70.

         LINE     PCCTYPE=ANSI;                                          001150
24:JOB   INCLUDE=(OS);                                                  001160

VOLUME HOST=OSWTR,OSCHN=9,OSTLP=0,OSHDP=1,PLABEL=YES;BLOCK
LENGTH=2400,PREAMBLE=4,LTHFLD=2,FORMAT=BIN;RECORD LENGTH=136,
PREAMBLE=4,STRUCTURE=VB,LTHFLD=2,FORMAT=BIN;LINE DATA=(1,132),
PCCTYPE=ANSI,PCC=(0,NOTRAN),OVERPRINT=(MERGE,NODISP),VFU=WR;

         VOLUME   HOST=OSWTR;                                           001170
         LINE     PCCTYPE=IBM1403;                                      001180
   /*                                                          */       001190
   /*             US70 (RCA) STANDARD LABELED TAPES            */       001200
   /*             ----------------------------------------     */       001210
   /*                                                          */       001220
   /*    JDES 41 AND 42 PROVIDE SUPPORT FOR US70 (FORMERLY RCA) */       001230
   /*    STANDARD LABELED TAPES.                               */       001240
   /*                                                          */       001250
41:JOB   INCLUDE=(US);                                                  001260

VOLUME HOST=US70,LABEL=STANDARD,PLABEL=YES;BLOCK LENGTH=1330,
PREAMBLE=0;RECORD LENGTH=133,STRUCTURE=FB;LINE DATA=(1,132),
PCCTYPE=US70,PCC=(0,NOTRAN),OVERPRINT=(MERGE,NODISP),VFU=V1;

         VOLUME   HOST=US70, LABEL=STANDARD;                            001270
END;END;                                                                001280

*** PD0200 REPLACE AUTHORIZED BY OPERATOR, STRING = END, COL = 01                <<<<<<<<
*** PD1050 EXITING PDL TO PRINT                                                  <<<<<<<<
```

```
.........H2000.JSL COMPILED BY PDL(A00 ) AT 07:55 ON 22-MAR-79.........                1
1........10........20........30........40........50........60........70.

H2000:JDL;                                                                      000010
   /*                                                                  */     000020
   /*              SYSTEM FOR HONEYWELL 2000                           */     000030
   /*              -------------------------                           */     000040
   /*                                                                  */     000050
   /*                                                                  */     000060
   /*     VFU FOR HONEYWELL 2000 SPR TAPES - CHANNEL 2 IS BOF          */     000070
   /*                                                                  */     000080
V1:       VFU       ASSIGN=(1,5), ASSIGN=(2,61), ASSIGN=(3,10),               000090
                    ASSIGN=(4,15), ASSIGN=(5,20), ASSIGN=(6,25),              000100
                    ASSIGN=(7,30), ASSIGN=(8,35), ASSIGN=(9,40),              000110
                    ASSIGN=(10,45), ASSIGN=(11,50), ASSIGN=(12,55),           000120
                    TOF=5, BOF=66;                                            000130
   /*                                                                  */     000140
   /*                                                                  */     000150
   /*     VFU FOR HONEYWELL 2000 STANDARD TAPES - CHANNEL 12 IS BOF    */     000160
   /*                                                                  */     000170
V2:       VFU       ASSIGN=(1,5), ASSIGN=(2,10), ASSIGN=(3,15),               000180
                    ASSIGN=(4,20), ASSIGN=(5,25), ASSIGN=(6,30),              000190
                    ASSIGN=(7,35), ASSIGN=(8,40), ASSIGN=(9,45),              000200
                    ASSIGN=(10,50), ASSIGN=(11,55), ASSIGN=(12,60),           000210
                    TOF=5, BOF=66;                                            000220
   /*                                                                  */     000230
          VOLUME    HOST=H2000, LABEL=SPR, CODE=EBCDIC, LCODE=EBCDIC,         000240
                    PLABEL=YES;                                               000250
          BLOCK     LENGTH=1331, PREAMBLE=1, LTHFLD=0;                        000260
          RECORD    LENGTH=133, LTHFLD=0, STRUCTURE=FB;                       000270
          LINE      DATA=(1,132), PCC=(0,TRAN),   PCCTYPE=H2000,              000280
                    OVERPRINT=(MERGE,NODISP), VFU=V1;                         000290
          ACCT      USER=(BIN,TRAY);                                          000300
   /*                                                                  */     000310
   /*              TABLES AND CRITERIA FOR LOGICAL PROCESSING          */     000320
   /*                                                                  */     000330
T1:       TABLE     CONSTANT='P';                                             000340
T2:       TABLE     CONSTANT='PO';                                            000350
T3:       TABLE     CONSTANT='P1';                                            000360
T4:       TABLE     CONSTANT=0'47';                                           000370
   /*                                                                  */     000380
C1:       CRITERIA CONSTANT=(0,1,EQ,T1);                                      000390
C2:       CRITERIA CONSTANT=(0,2,EQ,T2);                                      000400
C3:       CRITERIA CONSTANT=(0,2,EQ,T3);                                      000410
C4:       CRITERIA CONSTANT=(0,1,EQ,T4);                                      000420
   /*                                                                  */     000430
   /*                                                                  */     000440
   /*                                                                  */     000450
   /*              HONEYWELL 2000 SPR                                  */     000460
   /*              ----------------------------------------            */     000470
   /*                                                                  */     000480
   /*     JDES 1-5 AND 11-15 DEFINE 5 COMMON HONEYWELL 2000 SPR        */     000490
   /*     LABELED FORMATS.                                             */     000500
   /*     THE FORMATS ARE AS FOLLOWS:                                  */     000510
   /*                                                                  */     000520
   /*                                                                  */     000530
   /*        CHARACTERISTICS                    JOB                    */     000540
   /*        ---------------                   --------                */     000550
   /*                                                                  */     000560
   /*        BLOCKED 1331, SINGLE BANNER          1                    */     000570
   /*        BLOCKED 668, SINGLE BANNER, 2 BYTE   2                    */     000580
   /*           POSTAMBLE                                              */     000590
   /*        BLOCKED 1332, DOUBLE BANNER,         3                    */     000600
   /*           SINGLE REPORT PER FILE                                 */     000610
   /*        BLOCKED 1332, DOUBLE BANNER, SELECT  4                    */     000620
   /*           REPORT 0 IN INTERSPERSED REPORT                        */     000630
   /*        BLOCKED 1332, DOUBLE BANNER, SELECT  5                    */     000640
   /*           REPORT 1 IN INTERSPERSED REPORT                        */     000650
   /*                                                                  */     000660
   /*                                                                  */     000670
1:JOB;                                                                          000680
          VOLUME    HOST=H2000, LABEL=SPR  ;                                  000690
          BLOCK     LENGTH=1331, PREAMBLE=1, LTHFLD=0;                        000700
```

```
.........H2000.JSL COMPILED BY PDL(A00 ) AT 07:55 ON 22-MAR-79.........                2
1........10........20........30........40........50........60........70.

          RECORD    LENGTH=133, STRUCTURE=FB, LTHFLD=0;                          000710
          BSELECT   TEST=(C1);                                                   000720
2:JOB;                                                                           000730
          VOLUME    HOST=H2000, LABEL=SPR;                                       000740
          BLOCK     LENGTH=668, PREAMBLE=1, LTHFLD=0, POSTAMBLE=2;               000750
          RECORD    LENGTH=133, STRUCTURE=FB;                                    000760
          BSELECT   TEST=(C1);                                                   000770
3:JOB;                                                                           000780
          VOLUME    HOST=H2000, LABEL=SPR;                                       000790
          BLOCK     LENGTH=1332, PREAMBLE=2, LTHFLD=0;                           000800
          RECORD    LENGTH=133, STRUCTURE=FB;                                    000810
          BSELECT   TEST=(C1);                                                   000820
4:JOB;                                                                           000830
          VOLUME    HOST=H2000, LABEL=SPR;                                       000840
          BLOCK     LENGTH=1332, PREAMBLE=2, LTHFLD=0;                           000850
          RECORD    LENGTH=133, STRUCTURE=FB;                                    000860
          BSELECT   TEST=(C2);                                                   000870
5:JOB;                                                                           000880
          VOLUME    HOST=H2000, LABEL=SPR;                                       000890
          BLOCK     LENGTH=1332, PREAMBLE=2, LTHFLD=0;                           000900
          RECORD    LENGTH=133, STRUCTURE=FB;                                    000910
          BSELECT   TEST=(C3);                                                   000920
    /*                                                                  */       000930
    /*            HONEYWELL 2000 STANDARD, COBOL AND SPR BCD TAPES      */       000940
    /*            ------------------------------------------------      */       000950
    /*                                                                  */       000960
    /*                                                                  */       000970
/*       JDES 21-26 DEFINE 6 COMMON HONEYWELL 2000 FORMATS             */       000980
    /*    UTILIZING STANDARD 80-BYTE LABELS AND COBOL 120-BYTE LABELS*/          000990
    /*    AS WELL AS BCD-CODED SPR LABELED TAPES.  THE FORMATS ARE AS*/          001000
    /*    FOLLOWS:                                                      */       001010
    /*                                                                  */       001020
    /*       CHARACTERISTICS                          JOB               */       001030
    /*       ---------------                        --------            */       001040
    /*                                                                  */       001050
    /*       BCD-CODED SPR LABELS, BLOCKED 1201        21               */       001060
    /*       BCD-CODED SPR LABELS, BLOCKED 1332        22               */       001070
    /*       EBCDIC-CODED STANDARD LABELS,             23               */       001080
    /*          BLOCKED 1340                                            */       001090
    /*       BCD-CODED STANDARD LABELS, BLOCKED        24               */       001100
    /*          1009                                                    */       001110
    /*       EBCDIC-CODED COBOL LABELS, BLOCKED        25               */       001120
    /*          1440                                                    */       001130
    /*       EBCDIC-CODED COBOL LABELS, BLOCKED        26               */       001140
    /*          1500                                                    */       001150
    /*                                                                  */       001160
21:JOB;                                                                          001170
          VOLUME    HOST=H2000, LABEL=SPR, CODE=H2BCD, LCODE=H2BCD,              001180
                    UNPACK=T4X3H2;                                               001190
          BLOCK     LENGTH=1201, PREAMBLE=2, POSTAMBLE=2;                        001200
          BSELECT   TEST=(C4);                                                   001210
22:JOB;                                                                          001220
          VOLUME    HOST=H2000, LABEL=SPR, CODE=H2BCD, LCODE=H2BCD,              001230
                    UNPACK=T4X3H2;                                               001240
          BLOCK     LENGTH=1332, PREAMBLE=2;                                     001250
          BSELECT   TEST=(C4);                                                   001260
23:JOB;                                                                          001270
          VOLUME    HOST=H2000, LABEL=STANDARD, CODE=EBCDIC;                     001280
          BLOCK     LENGTH=1340, PREAMBLE=0;                                     001290
          RECORD    STRUCTURE=FB, LENGTH=134;                                    001300
          LINE      VFU=V2;                                                      001310
24:JOB;                                                                          001320
          VOLUME    HOST=H2000, LABEL=STANDARD, CODE=H2BCD, LCODE=H2BCD,         001330
                    UNPACK=T4X3H2;                                               001340
          BLOCK     LENGTH=1009, PREAMBLE=0, POSTAMBLE=1;                        001350
          RECORD    LENGTH=144, STRUCTURE=FB;                                    001360
          LINE      DATA=(8,132), VFU=V2;                                        001370
25:JOB;                                                                          001380
          VOLUME    HOST=H2000, LABEL=COBOL,                                     001390
                    CODE=EBCDIC;                                                 001400
```

# H2000 (cont.)

```
.........H2000.JSL COMPILED BY PDL(A00 ) AT 07:55 ON 22-MAR-79.........                3
1........10........20........30........40........50........60........70.

          BLOCK     LENGTH=1440, PREAMBLE=0;                                    001410
          RECORD    LENGTH=144, STRUCTURE=FB;                                   001420
          LINE      DATA=(2,132), VFU=V2, PCCTYPE=ANSI, PCC=(1,NOTRAN);         001430
26:JOB;                                                                         001440
          VOLUME    HOST=H2000, LABEL=COBOL,                                    001450
                    CODE=EBCDIC;                                                001460
          BLOCK     LENGTH=1500, PREAMBLE=0;                                    001470
          RECORD    LENGTH=150, STRUCTURE=FB;                                   001480
          LINE      DATA=(1,132), VFU=V2, PCCTYPE=H2000;                        001490
  END;END;     /* END OF HONEYWELL 2000 JDL */                                  001500

*** PD0200 REPLACE AUTHORIZED BY OPERATOR, STRING = END, COL = 03                <<<<<<<<
*** PD1050 EXITING PDL TO PRINT                                                  <<<<<<<<
```

```
.........H6000.JSL COMPILED BY PDL(A00 ) AT 08:35 ON 22-MAR-79..........
1........10........20........30........40........50........60........70.

H6000:JDL;                                                                  000010
   /*                                                                 */    000020
   /*     HONEYWELL 6000 JOB DESCRIPTOR  LIBRARY                      */    000030
   /*                                                                 */    000040
V1: VFU  ASSIGN=(1,10), ASSIGN=(2,15), ASSIGN=(3,20), ASSIGN=(4,25),        000050
         ASSIGN=(5,30), ASSIGN=(6,35), ASSIGN=(7,40), ASSIGN=(8,45),        000060
         ASSIGN=(9,50), ASSIGN=(10,55), ASSIGN=(11,60), ASSIGN=(12,62),     000070
         ASSIGN=(0,5), TOF=5, BOF=66;                                       000080
   /*                                                                 */    000090
         VOLUME   HOST=H6000, LABEL=STANDARD, UNPACK=T4X3, CODE=H6BCD,      000100
                  LCODE=H6BCD,  PLABEL=YES;                                 000110
         BLOCK    LENGTH=1920, LTHFLD=3, OFFSET=3, ADJUST=6, PREAMBLE=6,    000120
                  POSTAMBLE=0;                                              000130
         RECORD   STRUCTURE=VB, LENGTH=134, LTHFLD=3, OFFSET=0,             000140
                  ADJUST=6, PREAMBLE=0, POSTAMBLE=0;                        000150
         LINE     DATA=(0,132), PCC=(0,NOTRAN), PCCTYPE=H6000,              000160
                  OVERPRINT=(MERGE,NODISP), VFU=V1;                         000170
         ACCT     USER=(BIN,TRAY);                                          000180
   /*                                                               */      000190
   /*              TABLES AND CRITERIA FOR LOGICAL PROCESSING       */      000200
   /*                                                               */      000210
C1:      CRITERIA CHANGE=(5,1,NE,LAST);                                     000220
   /*                                                               */      000230
   /*                                                               */      000240
   /*                                                               */      000250
   /*              HONEYWELL 6000 SSF BCD AND ASCII TAPES           */      000260
   /*              -------------------------------------           */      000270
   /*                                                               */      000280
   /*                                                               */      000290
   /*    THE FOLLOWING JDES WILL SUPPORT 9 TRACK BCD AND ASCII TAPES*/      000300
  /*     FOR HONEYWELL 6000 SSF FORMATS.                           */      000310
   /*    THE JDES ARE AS FOLLOWS:                                   */      000320
   /*                                                               */      000330
   /*              CHARACTERISTICS                  JOB             */      000340
   /*              ---------------                 --------         */      000350
   /*                                                               */      000360
   /*              BCD-CODED SSF                      1             */      000370
   /*              BCD-CODED SSF, STACKED REPORTS     2             */      000380
   /*              ASCII-CODED SSF                    3             */      000390
   /*              ASCII-CODED SSF, STACKED REPORTS   4             */      000400
   /*                                                               */      000410
1:JOB;                                                                      000420
         VOLUME   HOST=H6000, LABEL=STANDARD, CODE=H6BCD, LCODE=H6BCD,      000430
                  UNPACK=T4X3;                                              000440
2:JOB;                                                                      000450
         VOLUME   HOST=H6000, LABEL=STANDARD, CODE=H6BCD, LCODE=H6BCD,      000460
                  UNPACK=T4X3;                                              000470
         RSTACK   TEST=(C1), PRINT=NONE;                                    000480
3:JOB;                                                                      000490
         VOLUME   HOST=H6000, LABEL=STANDARD, CODE=ASCII, LCODE=H6BCD,      000500
                  UNPACK=T4X3;                                              000510
4:JOB;                                                                      000520
         VOLUME   HOST=H6000, LABEL=STANDARD, CODE=ASCII, LCODE=H6BCD,      000530
                  UNPACK=T4X3;                                              000540
         RSTACK   TEST=(C1), PRINT=NONE;                                    000550
  END;END;/* END OF HONEYWELL 6000 JDL */                                   000560

*** PD0200 REPLACE AUTHORIZED BY OPERATOR, STRING = END, COL = 03                <<<<<<<<
*** PD1050 EXITING PDL TO PRINT                                                 <<<<<<<<
```

```
.....BUR.JSL COMPILED BY PDL(A00 ) AT 07:57 ON 22-MAR-79.....
1.....10.....20.....30.....40.....50.....60.....70.

BUR:JDL;                                                                  000010
   /*                                                                  */ 000020
   /*     SYSTEM TO PRINT BURROUGHS MEDIUM SYSTEM AND BURROUGHS       */ 000030
   /*        LARGE SYSTEM PRINTER BACKUP TAPE FORMATS                 */ 000040
   /*                                                                 */ 000050
   /*                                                                 */ 000060
V1:       VFU       ASSIGN=(1,5), ASSIGN=(2,10), ASSIGN=(3,15),           000070
                    ASSIGN=(4,20), ASSIGN=(5,25), ASSIGN=(6,30),          000080
                    ASSIGN=(7,35), ASSIGN=(8,40), ASSIGN=(9,45),          000090
                    ASSIGN=(10,50), ASSIGN=(11,55), TOF=5, BOF=66;        000100
   /*                                                                  */ 000110
   /*     USER TRANSLATION TABLE FOR BURROUGHS ALTERS LEFT AND        */ 000120
   /*     RIGHT BRACKETS AND EXCLAMATION MARK                         */ 000130
   /*                                                                 */ 000140
          CODE      DEFAULT=EBCDIC, ASSIGN=(X'4A',X'B4'), ASSIGN=(X'5A',  000150
                    X'B5'), ASSIGN=(X'DO',X'5A');                         000160
   /*                                                                  */ 000170
          VOLUME    HOST=B4700, LABEL=STANDARD,  CODE=USER, LCODE=USER,    000180
                    PLABEL=YES;                                           000190
          BLOCK     LENGTH=816, PREAMBLE=0, POSTAMBLE=0;                  000200
          RECORD    LENGTH=136, STRUCTURE=FB;                             000210
          LINE      DATA=(4,132), PCCTYPE=B4700; PCC=(0,NOTRAN),          000220
                    OVERPRINT=(MERGE,NODISP), VFU=V1;                     000230
          ACCT      USER=(BIN,TRAY);                                      000240
   /*                                                                  */ 000250
   /*                                                                  */ 000260
   /*     CATALOG TO PRINT BURROUGHS LARGE SYSTEM (B6700 AND B7700)   */ 000270
   /*             PRINTER BACKUP TAPES                                */ 000280
   /*                                                                 */ 000290
LB:CATALOG;                                                               000300
          VOLUME    HOST=B6700, LABEL=ANSI,CODE=USER, LCODE=USER,         000310
                    BMULT=6, RMULT=6, PLABEL=YES;                         000320
          BLOCK     LENGTH=1800, LTHFLD=0, PREAMBLE=0, POSTAMBLE=0;       000330
          RECORD    LENGTH=138, STRUCTURE=VB, LTHFLD=2, OFFSET=4,         000340
                    LMULT=6, FORMAT=BIN, ADJUST=6, PREAMBLE=6;            000350
          LINE      MARGIN=1, DATA=(0,132), PCCTYPE=B6700, PCC=(0,NOTRAN),000360
                    OVERPRINT=(MERGE,NODISP), VFU=V1;                     000370
   /*                                                                  */ 000380
   /*                                                                  */ 000400
   /*                                                                  */ 000410
   /*             BURROUGHS MEDIUM SYSTEM SUPPORT           */          000420
   /*             ------------------------------------------        */ 000430
   /*                                                                 */ 000440
   /*     THE FOLLOWING JDES WILL PRINT BOTH LABELED AND UNLABELED    */ 000450
   /*     TAPES FROM THE BURROUGHS MEDIUM SYSTEMS (B2500, B2700,      */ 000460
  /*     B3500,B3700, AND B4700.                                      */ 000470
   /*                                                                 */ 000480
   /*        CHARACTERISTICS                    JOB                   */ 000490
   /*        --------------                    --------               */ 000500
   /*                                                                 */ 000510
   /*        LABELED PRINTER BACKUP TAPE         1                   */ 000520
   /*        UNLABELED PRINTER BACKUP TAPE       2                   */ 000530
   /*        NON-PRINTER BACKUP ANSI TAPE        3                   */ 000540
   /*                                                                 */ 000550
1:JOB;                                                                    000560
          VOLUME    HOST=B4700, LABEL=STANDARD, CODE=USER, LCODE=USER;    000570
2:JOB;                                                                    000580
          VOLUME    HOST=B4700, LABEL=NONE, CODE=USER;                    000590
3:JOB;                                                                    000600
          VOLUME    HOST=B4700, LABEL=STANDARD, CODE=USER, LCODE=USER;    000610
          BLOCK     LENGTH=1330;                                          000620
          RECORD    LENGTH=133, STRUCTURE=FB;                             000630
          LINE      DATA=(1,132), PCCTYPE=ANSI, PCC=(0,NOTRAN);           000640
   /*                                                                  */ 000650
   /*                                                                  */ 000660
   /*             BURROUGHS LARGE SYSTEM SUPPORT            */          000670
   /*             ------------------------------------------        */ 000680
   /*                                                                 */ 000690
   /*     THE FOLLOWING JDES WILL PRINT BOTH LABELED PRINTER BACKUP   */ 000700
   /*     AND STANDARD ANSI-LABELED TAPES FROM THE BURROUGHS LARGE    */ 000710
```

```
.....BUR.JSL COMPILED BY PDL(A00 ) AT 07:57 ON 22-MAR-79.....                2
1.....10.....20.....30.....40.....50.....60.....70.

  /*    SYSTEMS (B6700 AND B7700).                                  */     000720
   /*                                                               */     000730
   /*       CHARACTERISTICS                    JOB                 */     000740
   /*       ---------------                   --------             */     000750
   /*                                                               */     000760
   /*       LABELED PRINTER BACKUP TAPE         31                */     000770
   /*       STANDARD ANSI LABELED TAPE          32                */     000780
   /*                                                               */     000790
31:JOB   INCLUDE=(LB);                                                    000800

VOLUME HOST=B6700,LABEL=ANSI,CODE=USER,LCODE=USER,BMULT=6,RMULT=
6,PLABEL=YES;BLOCK LENGTH=1800,LTHFLD=0,PREAMBLE=0,POSTAMBLE=0;
RECORD LENGTH=138,STRUCTURE=VB,LTHFLD=2,OFFSET=4,LMULT=6,FORMAT=
BIN,ADJUST=6,PREAMBLE=6;LINE MARGIN=1,DATA=(0,132),PCCTYPE=
B6700,PCC=(0,NOTRAN),OVERPRINT=(MERGE,NODISP),VFU=V1;

         VOLUME   HOST=B6700, LABEL=ANSI;                                 000810
32:JOB;                                                                   000820
         VOLUME   HOST=B6700, LABEL=ANSI, CODE=USER, LCODE=USER,          000830
                  BMULT=6, RMULT=6;                                       000840
         BLOCK    LENGTH=1330;                                           000850
         RECORD   LENGTH=133, STRUCTURE=FB;                              000860
         LINE     DATA=(1,132), PCCTYPE=ANSI, PCC=(0,NOTRAN);            000870
END;END;     /*  END OF BURROUGHS JDL */                                 000880

*** PD0200 REPLACE AUTHORIZED BY OPERATOR, STRING = END, COL = 01                <<<<<<<<
*** PD1050 EXITING PDL TO PRINT                                                 <<<<<<<<
```

```
.......POWGRP.JSL COMPILED BY PDL(AOO ) AT 08:40 ON 22-MAR-79.........
1........10........20........30........40........50........60........70.

POWGRP:JDL;                                                              000010
   /*                                                                 */ 000020
   /*              SYSTEM FOR POWER, POWER VS, AND GRASP              */ 000030
   /*                                                                 */ 000040
V1:      VFU       ASSIGN=(1,5), ASSIGN=(2,10), ASSIGN=(3,15),           000050
                   ASSIGN=(4,20), ASSIGN=(5,25), ASSIGN=(6,30),          000060
                   ASSIGN=(7,35), ASSIGN=(8,40), ASSIGN=(9,45),          000070
                   ASSIGN=(10,50), ASSIGN=(11,55), ASSIGN=(12,60),       000080
                   TOF=4, BOF=66;                                        000090
   /*                                                                 */ 000100
   /*              TABLES AND CRITERIA FOR LOGICAL PROCESSING         */ 000110
   /*                     FOR GRASP INTERLEAVED TAPES                 */ 000120
   /*                                                                 */ 000130
T1:      TABLE     CONSTANT='B';                                         000140
T2:      TABLE     CONSTANT='C';                                         000150
   /*                                                                 */ 000160
C1:      CRITERIA  CONSTANT=(27,1,EQ,T1);                               000170
C2:      CRITERIA  CONSTANT=(27,1,EQ,T2);                               000180
   /*                                                                 */ 000190
   /*              SYSTEM FOR POWER VS                                */ 000200
   /*                                                                 */ 000210
         VOLUME    HOST=POWERVS, PLABEL=YES;                           000220
         BLOCK     LENGTH=2048;                                        000230
         RECORD    LENGTH=136, STRUCTURE=VB, LTHFLD=2, OFFSET=0,       000240
                   ADJUST=0, FORMAT=BIN, PREAMBLE=3;                   000250
         LINE      DATA=(1,132), PCCTYPE=IBM1403, PCC=(0,NOTRAN),      000260
                   OVERPRINT=(MERGE,NODISP), VFU=V1;                   000270
         ACCT      USER=(BIN,TRAY);                                    000280
   /*                                                                 */ 000290
   /*              CATALOG FOR POWER VERSIONS                         */ 000300
   /*                                                                 */ 000310
PW:CATALOG;                                                            000320
         VOLUME    HOST=POWER;                                         000330
         BLOCK     LENGTH=2048, PREAMBLE=6, LTHFLD=2, FORMAT=BIN,      000340
                   OFFSET=4;                                           000350
         RECORD    LENGTH=135, STRUCTURE=VB, PREAMBLE=2,               000360
                   LTHFLD=2, FORMAT=BIN, OFFSET=0, ADJUST=3;           000370
   /*                                                                 */ 000380
   /*              CATALOG FOR GRASP                                  */ 000390
   /*                                                                 */ 000400
GR:CATALOG;                                                            000410
         VOLUME    HOST=GRASP;                                         000420
         BLOCK     LENGTH=4096, PREAMBLE=0, ZERO=YES;                  000430
         RECORD    LENGTH=135, STRUCTURE=VB, PREAMBLE=1,               000440
                   LTHFLD=1, FORMAT=BIN, OFFSET=0, ADJUST=2;           000450
   /*                                                                 */ 000460
   /*                                                                 */ 000480
   /*                                                                 */ 000490
   /*              POWER VS, POWER 4.0, AND POWER 4.1/4.2             */ 000500
   /*              -------------------------------------              */ 000510
   /*                                                                 */ 000520
   /*    THE FOLLOWING JDES PROVIDE SUPPORT FOR IBM POWER VS TAPES,   */ 000530
   /*    POWER VERSION 4.0 TAPES, AND POWER VERSIONS 4.1/4.2 TAPES    */ 000540
   /*                                                                 */ 000550
   /*       CHARACTERISTICS                        JOB                */ 000560
   /*       ---------------                      --------             */ 000570
   /*                                                                 */ 000580
   /*       POWER VS TAPES                          1                 */ 000590
   /*       POWER VERSION 4.0 TAPES                 2                 */ 000600
   /*       POWER VERSIONS 4.1/4.2 TAPES            3                 */ 000610
   /*                                                                 */ 000620
1:JOB;                                                                 000630
         VOLUME    HOST=POWERVS;                                       000640
2:JOB    INCLUDE=(PW);                                                 000650

VOLUME HOST=POWER:BLOCK LENGTH=2048,PREAMBLE=6,LTHFLD=2,FORMAT=
BIN,OFFSET=4;RECORD LENGTH=135,STRUCTURE=VB,PREAMBLE=2,LTHFLD=2,
FORMAT=BIN,OFFSET=0,ADJUST=3;

         VOLUME    HOST=POWER;                                         000660
```

# Power Grasp (cont.)

```
.....POWGRP.JSL COMPILED BY PDL(A00 ) AT 08:40 ON 22-MAR-79.....          2
1.....10.....20.....30.....40.....50.....60.....70.

        RECORD   LTHFLD=1, PREAMBLE=1, ADJUST=2;                        000670
3:JOB   INCLUDE=(PW);                                                   000680

VOLUME HOST=POWER;BLOCK LENGTH=2048,PREAMBLE=6,LTHFLD=2,FORMAT=
BIN,OFFSET=4;RECORD LENGTH=135,STRUCTURE=VB,PREAMBLE=2,LTHFLD=2,
FORMAT=BIN,OFFSET=0,ADJUST=3;

        VOLUME   HOST=POWER;                                            000690
   /*                                                            */     000700
   /*            IBM DOS GRASP TAPES                             */     000710
   /*            -------------------                             */     000720
   /*                                                            */     000730
   /*    THE FOLLOWING JDES PROVIDE SUPPORT FOR GRASP TAPES      */     000740
   /*                                                            */     000750
   /*       CHARACTERISTICS                          JOB         */     000760
   /*       ---------------                         --------     */     000770
   /*                                                            */     000780
   /*       NORMAL GRASP TAPES                        21         */     000790
   /*       GRASP WITH INTERSPERSED REPORTS - SELECT  22         */     000800
   /*         REPORTS FROM PHANTOM DEVICE B                      */     000810
   /*       GRASP WITH INTERSPERSED REPORTS - DELETE  23         */     000820
   /*         REPORTS FROM PHANTOM DEVICE C                      */     000830
   /*                                                            */     000840
21:JOB  INCLUDE=(GR);                                                   000850

VOLUME HOST=GRASP;BLOCK LENGTH=4096,PREAMBLE=0,ZERO=YES;RECORD
LENGTH=135,STRUCTURE=VB,PREAMBLE=1,LTHFLD=1,FORMAT=BIN,OFFSET=0,
ADJUST=2;

        VOLUME   HOST=GRASP;                                            000860
22:JOB  INCLUDE=(GR);                                                   000870

VOLUME HOST=GRASP;BLOCK LENGTH=4096,PREAMBLE=0,ZERO=YES;RECORD
LENGTH=135,STRUCTURE=VB,PREAMBLE=1,LTHFLD=1,FORMAT=BIN,OFFSET=0,
ADJUST=2;

        VOLUME   HOST=GRASP;                                            000880
        BSELECT  TEST=(C1);                                             000890
23:JOB  INCLUDE=(GR);                                                   000900

VOLUME HOST=GRASP;BLOCK LENGTH=4096,PREAMBLE=0,ZERO=YES;RECORD
LENGTH=135,STRUCTURE=VB,PREAMBLE=1,LTHFLD=1,FORMAT=BIN,OFFSET=0,
ADJUST=2;

        VOLUME   HOST=GRASP;                                            000910
        BSELECT  TEST=(C2);                                             000920
   /*                                                            */     000930
   /*            END OF POWER AND GRASP SYSTEM                   */     000940
END;END;                                                                000950

*** PD0200 REPLACE AUTHORIZED BY OPERATOR, STRING = END, COL = 01        <<<<<<<<
*** PD1050 EXITING PDL TO PRINT                                         <<<<<<<<
```

```
.........XEROX.JSL COMPILED BY PDL(A00 ) AT 08:49 ON 22-MAR-79.........
1........10........20........30........40........50........60........70.

XEROX:JDL;                                                              000010
   /*              SYSTEM FOR XEROX TAPES                              */ 000020
   /*                                                                  */ 000030
V1:       VFU      ASSIGN=(1,5), ASSIGN=(2,10), ASSIGN=(3,15),          000040
                   ASSIGN=(4,20), ASSIGN=(5,25), ASSIGN=(6,30),         000050
                   ASSIGN=(7,35), ASSIGN=(8,40), ASSIGN=(9,45),         000060
                   ASSIGN=(10,50), ASSIGN=(11,55), ASSIGN=(12,60),      000070
                   TOF=5, BOF=66;                                       000080
   /*                                                                  */ 000090
          VOLUME   HOST=XEROX, LABEL=ANSI, CODE=EBCDIC, PLABEL=YES;     000100
          BLOCK    LENGTH=4096, PREAMBLE=0, LTHFLD=0;                   000110
          RECORD   PREAMBLE=0, STRUCTURE=FB;                            000120
          LINE     DATA=(1,132), PCCTYPE=XEROX, PCC=(0,NOTRAN),         000130
                   OVERPRINT=(MERGE,NODISP),VFU=V1;                     000140
          ACCT     USER=(BIN,TRAY);                                     000150
   /*                                                                  */ 000350
   /*                                                                  */ 000360
   /*              XEROX ANSI LABELED AND UNLABELED TAPES              */ 000370
   /*              ----------------------------------------------      */ 000380
   /*                                                                  */ 000390
   /*    THE FOLLOWING JDES WILL PRINT A VARIETY OF XEROX TAPES -      */ 000400
   /*    BOTH ANSI LABELED AND UNLABELED                               */ 000410
   /*                                                                  */ 000420
   /*       CHARACTERISTICS                          JOB               */ 000430
   /*       ---------------                        --------            */ 000440
   /*                                                                  */ 000450
   /*       ANSI-LABELED, FB, BLOCKED 3990, XEROX     11               */ 000460
   /*          PCC                                                     */ 000470
   /*       ANSI-LABELED, U, BLOCKED 133, XEROX PCC   12               */ 000480
   /*       UNLABELED, BLOCKED 3990, FB, XEROX PCC    13               */ 000490
   /*       UNLABELED, BLOCKED 3960, FB, XEROX PCC    14               */ 000500
   /*       UNLABELED, BLOCKED 133, U, XEROX PCC      15               */ 000510
   /*                                                                  */ 000520
11:JOB;                                                                 000530
          VOLUME   HOST=XEROX, LABEL=ANSI;                              000540
          BLOCK    LENGTH=3990;                                         000550
          RECORD   LENGTH=133, STRUCTURE=FB;                            000560
12:JOB;                                                                 000570
          VOLUME   HOST=XEROX, LABEL=ANSI;                              000580
          BLOCK    LENGTH=133;                                          000590
          RECORD   LENGTH=133, STRUCTURE=U;                             000600
13:JOB;                                                                 000610
          VOLUME   HOST=XEROX, LABEL=NONE;                              000620
          BLOCK    LENGTH=3990;                                         000630
          RECORD   LENGTH=133, STRUCTURE=FB;                            000640
14:JOB;                                                                 000650
          VOLUME   HOST=XEROX, LABEL=NONE;                              000660
          BLOCK    LENGTH=3960;                                         000670
          RECORD   LENGTH=132, STRUCTURE=FB;                            000680
15:JOB;                                                                 000690
          VOLUME   HOST=XEROX, LABEL=NONE;                              000700
          BLOCK    LENGTH=133;                                          000710
          RECORD   LENGTH=133, STRUCTURE=U;                             000720
   /*                                                                  */ 000730
   /*              JDES FOR UTILITY PRINTOUTS                          */ 000740
   /*              --------------------------                          */ 000750
   /*                                                                  */ 000760
   /*    THE FOLLOWING JDES WILL ALLOW PRINTOUTS OF MOST TAPES         */ 000770
   /*    WITHOUT ATTEMPTING TO PROPERLY DEBLOCK THE TAPE.              */ 000780
   /*                                                                  */ 000790
   /*       CHARACTERISTICS              JOB                           000800
   /*       ---------------            --------                        000810
   /*                                                                  000820
   /*       UNBLOCKED, NO PCC            31                           000830
   /*       BLOCKED 4096, U, NO PCC      32                           000840
   /*       BLOCKED 3990, FB, NO PCC     33                           000850
   /*                                                                */ 000860
31:JOB;                                                                 000870
          VOLUME   LABEL=NONE;                                          000880
          BLOCK    LENGTH=133;                                          000890
```

# Xerox (cont.)

```
.........XEROX.JSL COMPILED BY PDL(A00 ) AT 08:49 ON 22-MAR-79.........              2
1.........10........20........30........40........50........60........70.

          RECORD    STRUCTURE=U;                                              000900
          LINE      PCCTYPE=NONE, DATA=(0,132);                               000910
32:JOB;                                                                       000920
          VOLUME    LABEL=NONE;                                               000930
          BLOCK     LENGTH=4096;                                              000940
          RECORD    STRUCTURE=U;                                              000950
          LINE      PCCTYPE=NONE, DATA=(0,132);                               000960
33:JOB;                                                                       000970
          VOLUME    LABEL=NONE;                                               000980
          BLOCK     LENGTH=3990;                                              000990
          RECORD    LENGTH=133, STRUCTURE=FB;                                 001000
          LINE      PCCTYPE=NONE, DATA=(0,132);                               001010
END;END; /* END OF JDL */                                                     001020

*** PD0200 REPLACE AUTHORIZED BY OPERATOR, STRING = END, COL = 01                <<<<<<<<
*** PD1050 EXITING PDL TO PRINT                                                 <<<<<<<<
```

# APPENDIX F. CHARACTER CORRESPONDENCE IN UCSB ASSOCIATIVE FIELD

| UCSB Location | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Byte | Bit 0 | | Bit 1 | | Bit 2 | | Bit 3 | |
| | Hex | Char | Hex | Char | Hex | Char | Hex | Char |
| 448 | 00 | | 40 | Sp | 80 | ¼ | C0 | { |
| 449 | 01 | | 41 | | 81 | a | C1 | A |
| 450 | 02 | | 42 | | 82 | b | C2 | B |
| 451 | 03 | | 43 | | 83 | c | C3 | C |
| 452 | 04 | | 44 | | 84 | d | C4 | D |
| 453 | 05 | | 45 | | 85 | e | C5 | E |
| 454 | 06 | | 46 | | 86 | f | C6 | F |
| 455 | 07 | | 47 | | 87 | g | C7 | G |
| 456 | 08 | | 48 | | 88 | h | C8 | H |
| 457 | 09 | | 49 | | 89 | i | C9 | I |
| 458 | 0A | | 4A | ¢ | 8A | | CA | |
| 459 | 0B | | 4B | . | 8B | { | CB | |
| 460 | 0C | | 4C | < | 8C | | CC | |
| 461 | 0D | | 4D | ( | 8D | | CD | |
| 462 | 0E | | 4E | + | 8E | | CE | |
| 463 | 0F | | 4F | ¦ | 8F | | CF | |
| 464 | 10 | | 50 | & | 90 | ½ | D0 | } |
| 465 | 11 | | 51 | | 91 | j | D1 | J |
| 466 | 12 | | 52 | | 92 | k | D2 | K |
| 467 | 13 | | 53 | | 93 | l | D3 | L |
| 468 | 14 | | 54 | | 94 | m | D4 | M |
| 469 | 15 | | 55 | | 95 | n | D5 | N |
| 470 | 16 | | 56 | | 96 | o | D6 | O |
| 471 | 17 | | 57 | | 97 | p | D7 | P |
| 472 | 18 | | 58 | | 98 | q | D8 | Q |
| 473 | 19 | | 59 | | 99 | r | D9 | R |
| 474 | 1A | | 5A | ! | 9A | | DA | |
| 475 | 1B | | 5B | $ | 9B | } | DB | |
| 476 | 1C | | 5C | * | 9C | | DC | |
| 477 | 1D | | 5D | ) | 9D | | DD | |
| 478 | 1E | | 5E | ; | 9E | | DE | |
| 479 | 1F | | 5F | ~ | 9F | | DF | |

| UCSB Location | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Byte | Bit 0 | | Bit 1 | | Bit 2 | | Bit 3 | |
| | Hex | Char | Hex | Char | Hex | Char | Hex | Char |
| 480 | 20 | | 60 | | A0 | | E0 | |
| 481 | 21 | | 61 | | A1 | | E1 | |
| 482 | 22 | | 62 | | A2 | s | E2 | S |
| 483 | 23 | | 63 | | A3 | t | E3 | T |
| 484 | 24 | | 64 | | A4 | u | E4 | U |
| 485 | 25 | | 65 | | A5 | v | E5 | V |
| 486 | 26 | | 66 | | A6 | w | E6 | W |
| 487 | 27 | | 67 | | A7 | x | E7 | X |
| 488 | 28 | | 68 | | A8 | y | E8 | Y |
| 489 | 29 | | 69 | | A9 | z | E9 | Z |
| 490 | 2A | | 6A | ^ | AA | | EA | |
| 491 | 2B | | 6B | , | AB | | EB | |
| 492 | 2C | | 6C | % | AC | | EC | |
| 493 | 2D | | 6D | _ | AD | [ | ED | |
| 494 | 2E | | 6E | > | AE | | EE | |
| 495 | 2F | | 6F | ? | AF | | EF | |
| 496 | 30 | | 70 | | B0 | | F0 | 0 |
| 497 | 31 | | 71 | | B1 | | F1 | 1 |
| 498 | 32 | | 72 | | B2 | | F2 | 2 |
| 499 | 33 | | 73 | | B3 | | F3 | 3 |
| 500 | 34 | | 74 | | B4 | | F4 | 4 |
| 501 | 35 | | 75 | | B5 | | F5 | 5 |
| 502 | 36 | | 76 | | B6 | | F6 | 6 |
| 503 | 37 | | 77 | | B7 | | F7 | 7 |
| 504 | 38 | | 78 | | B8 | | F8 | 8 |
| 505 | 39 | | 79 | ¢ | B9 | | F9 | 9 |
| 506 | 3A | | 7A | : | BA | | FA | ¦ |
| 507 | 3B | | 7B | # | BB | | FB | |
| 508 | 3C | | 7C | @ | BC | | FC | |
| 509 | 3D | | 7D | ' | BD | ] | FD | |
| 510 | 3E | | 7E | = | BE | | FE | |
| 511 | 3F | | 7F | " | BF | | FF | |

Note:

Characters in the column 'char' are shown for illustration. Actual characters printed are a function of the font specified in the PDE command.

# APPENDIX G. ACCOUNTING DATA TAPE FORMAT

The OSS command ACCOUNT provides the capability to write system usage accounting data to tape. The tape will be unlabeled with 80 byte records. Each 80 byte record is identified in its first two bytes by a value indicating the type of information contained in that record. These two-byte identifiers are 01, 02, 03, 04. The first two records written to tape are '01' and '02' type records which are defined as follows:

Record 1

| Bytes | Contents |
|---|---|
| 0-1 | Identifier ('01') |
| 40-49 | Pages |
| 50-59 | Reports |
| 60-69 | Files |
| 70-79 | Jobs |

Record 2

| Bytes | Contents |
|---|---|
| 0-1 | Identifier ('02') |
| 2-6 | Hours (input) |
| 7-11 | Minutes (input) |
| 21-25 | Hours (output) |
| 26-30 | Minutes (output) |
| 44-49 | Mounts |
| 50-59 | Blocks read |
| 60-69 | Blocks skipped |
| 70-79 | Lines |

The information contained in these records is the same as that printed by the REPORT ACTIVITY command.

The remaining records on the tape are pairs of '03' and '04' type records, one pair for each "department-name/JDL" entry in the accounting log. The information contained in these records is the same as that printed by the REPORT USER command. The format of these records is as follows:

Record 3

| Bytes | Contents |
|---|---|
| 0-1 | Identifier ('03') |
| 2-32 | Department name |
| 35-39 | Pages to tray |
| 40-79 | Same as in record 1 |

Record 4

| Bytes | Contents |
|---|---|
| 0-1 | Identifier ('04') |
| 2-69 | Same as in record 2 |
| 70-79 | Paper path holes |

# INDEX

## A



## B



## C

# D

## E



## F



## H



## I



## J



## K



## L

# M



# O



# P



# R

# S



# T



# U

# V



# X

Xerox Corporation
701 South Aviation Boulevard
El Segundo, California 90245

XEROX

# Reader Comment Form

We would appreciate your comments and suggestions for improving this publication.

| Publication No. | Rev. Letter | Title | Current Date |
|---|---|---|---|
| | | | |

How did you use this publication?

☐ Learning ☐ Installing ☐ Sales
☐ Reference ☐ Maintaining ☐ Operating

Is the material presented effectively?

☐ Fully Covered ☐ Well Illustrated ☐ Well Organized ☐ Clear

What is your overall rating of this publication?

☐ Very Good ☐ Fair ☐ Very Poor
☐ Good ☐ Poor

What is your occupation?

Your other comments may be entered here. Please be specific and give page, column, and line number references where applicable.

Your Name & Return Address

2190(12/72)

Thank You For Your Interest. (fold & fasten as shown on back, no postage needed if mailed in U.S.A.)

Staple

Staple

Fold

First Class
Permit No. 229
El Segundo,
California

**BUSINESS REPLY MAIL**

**No postage stamp necessary if mailed in the United States**

**Postage will be paid by**

Xerox Corporation
701 South Aviation Boulevard
El Segundo, California 90245

*Attn: Programming Publications*

Fold

XEROX

# Publication Revision Sheet

June 1980

ADDITION TO XEROX 9700 ELECTRONIC PRINTING SYSTEM REFERENCE MANUAL.

Publication No. 91 00 2D, March 1980

This revision package contains "Chapter 13. Xerox 9700/850 Communications Interface." It is a new chapter for the 9700 Reference Manual and will be incorporated in the next edition of the manual.

XEROX

Xerox Corporation
701 South Aviation Boulevard
El Segundo, California 90245
213 679-4511

 Printed in U.S.A. 91 00 02