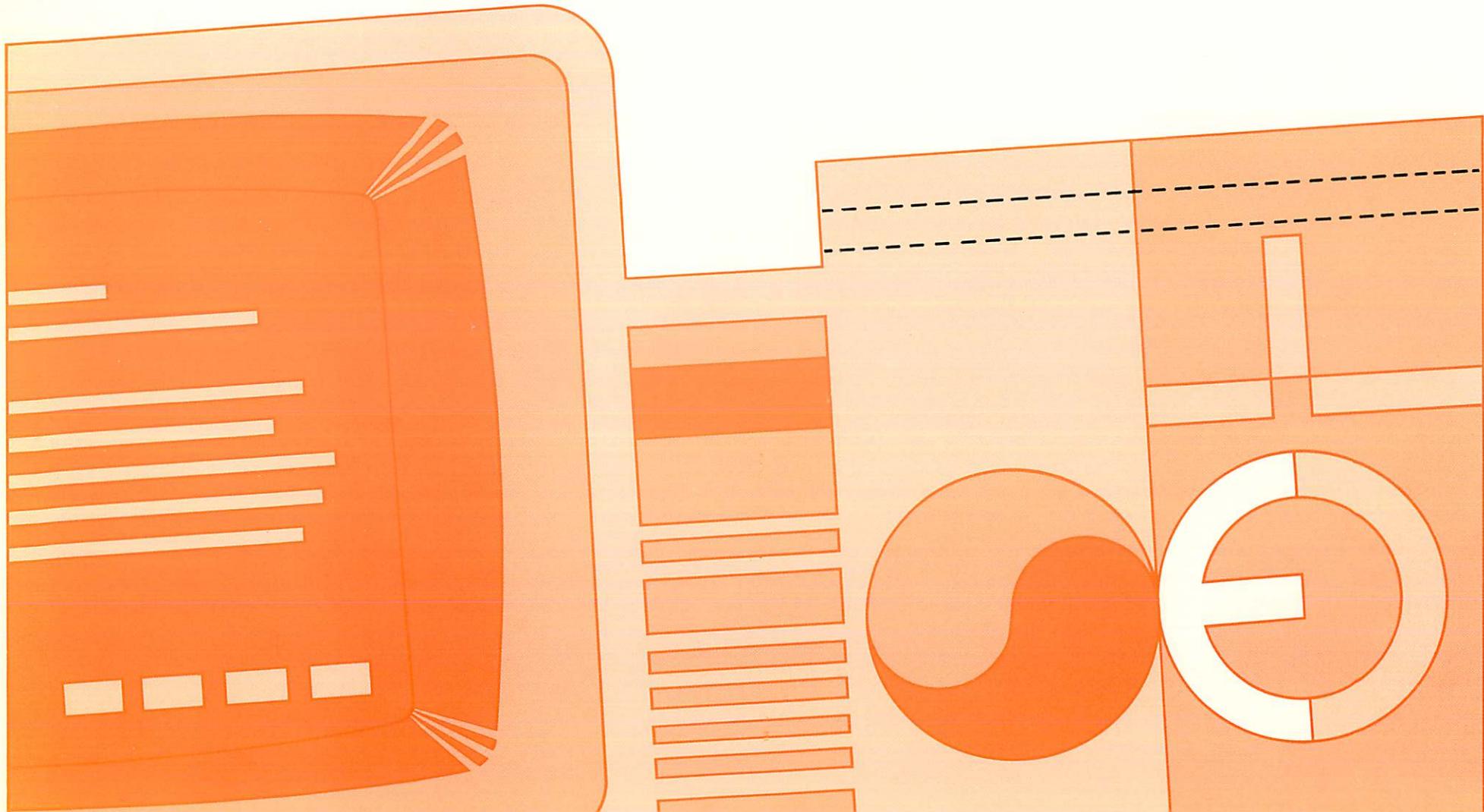


IFS/3000 Interactive Formatting System Reference Guide



HP 3000 Computer Systems

**Interactive Formatting System
IFS/3000**

Reference Guide



11311 CHINDEN BOULEVARD, BOISE, IDAHO 83707

Part No. 36580-90001

Product No. 36580A

Printed in U.S.A. 9/84

NOTICE

The information contained in this document is subject to change without notice.

HEWLETT-PACKARD MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance or use of this material.

Hewlett-Packard assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Hewlett-Packard.

This document contains proprietary information which is protected by copyright. All rights are reserved. No part of this document may be photocopied, reproduced or translated to another language without the prior written consent of Hewlett-Packard Company.

PRINTING HISTORY

New editions are complete revisions of the manual. Update packages, which are issued between editions, contain additional and replacement pages to be merged into the manual by the customer. The dates on the title page change only when a new edition or a new update is published. No information is incorporated into a reprinting unless it appears as a prior update; the edition does not change when an update is incorporated.

The software code printed alongside the date indicates the version level of the software product at the time the manual or update was issued. Many product updates and fixes do not require manual changes and, conversely, manual corrections may be done without accompanying product changes. Therefore, do not expect a one to one correspondence between product updates and manual updates.

Third Edition	January 1984.	HP 36580 A.02.01
Update 1.	September 1984.	HP 36580 A.02.03

THIS PAGE SHOULD BE BLANK

LIST OF EFFECTIVE PAGES

The List of Effective Pages gives the date of the most recent version of each page in the manual. To verify that your manual contains the most current information, check the dates printed at the bottom of each page with those listed below. The date on the bottom of each page reflects the edition or subsequent update in which that page was printed.

Effective Pages	Date
i thru xx	Sept 1984
2-3 thru 2-6.2	Sept 1984
3-33 thru 3-34	Sept 1984
4-1 thru 4-8	Sept 1984
4-19 thru 4-20	Sept 1984
5-1 thru 5-2	Sept 1984
5-21 thru 5-26	Sept 1984
Appendix B (all)	Sept 1984
D-1 thru D-6	Sept 1984
D-35 thru D-38.	Sept 1984
D-49 thru D-56.	Sept 1984
Appendix H (all)	Sept 1984
I-1 thru I-2	Sept 1984
Index (all)	Sept 1984
all other pages	Jan 1984

THIS PAGE SHOULD BE BLANK

PREFACE

Hello, and welcome.

Please take a moment to read this information.

This publication is the reference guide for the IFS/3000 Interactive Formatting System, a software subsystem that runs on the HP 3000. IFS/3000 enables you to electronically format documents to be printed by an intelligent output device such as an HP Laser Printer.

Overview

With purchase of an HP 2680A or 2688A Laser Printer, you receive a wide selection of print formats, known as environment files. A limited number of character sets (fonts) are also included. Print environments supplied with the HP 2680A are listed in Appendix C, Table C-1. HP 2688A standard environments are shown in Table C-2.

To create custom print environments using multiple character sets, electronic forms, or graphics, the following HP Software products are available:

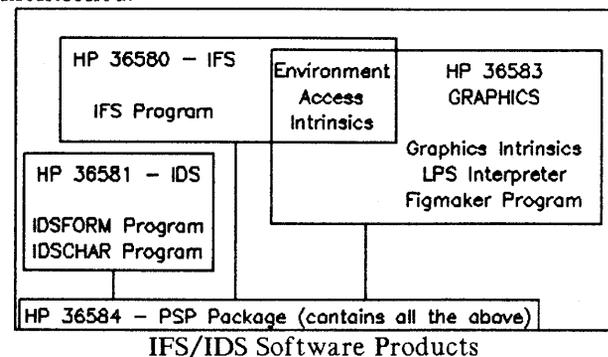
HP 36580 - Interactive Formatting System Software (IFS). Allows the user to combine selected character fonts and forms to create and compile custom environments. In addition to the environments mentioned above, when you purchase the IFS program you receive the Font Libraries listed in Appendix D.

HP 36581 - Interactive Design System (IDS). This program consists of two parts, IDIFORM and IDSCHAR, which provide the user with capabilities of designing customized character fonts, logos and forms.

HP 36583 - Printer Graphics Package. This provides a vector-to-raster conversion capability allowing printing of HP 3000 Graphics (DSG, HP Draw, Easychart) and non-HP graphics to HP Laser (HP 2680/2688) and Impact (HP 2608S, 2563A, 2565A, 2566A) printers.

HP 36584 Printer Support Package (PSP) This product bundles all the above mentioned software (36580, 36581, 36583) into one package.

The following figure provides a graphic representation of the software products just mentioned:



This manual contains information concerning the HP 36580 and HP 36583 products.

PREFACE (continued)

How To Use This Manual

The way you use this manual will be affected by the type of IFS/3000 installation you have on your system, and whether you are a programmatic or non-programmatic user.

IFS/3000 is actually a packaged subsystem with major components which may be installed in part, with limited capabilities; or with full capabilities when installed as part of the HP Laser Printer Support Package. See your System Manager to establish what is available on your system.

The HP non-graphics software includes the IFS/3000 Interactive Formatting System Program, the Programmatic Interface which gives you additional programmatic control over document format, a number of commonly used character fonts and document formats, and IDS/3000 for development of customized character fonts and forms.

The IFS graphics software includes the Graphics Programmatic Interface and the non-programmatic Laser Printing System Interpreter capabilities supporting document formatting control Laser Printer Graphics output.

With this in mind, the non-programmer should understand the concepts presented in Sections 1 and 2. This will enable you to determine if IFS supplied character fonts and formats meet your requirements. If none of the supplied formats meet your needs, you should use Section 3 to define your own format specifications.

The Programmer should be familiar with the material in Sections 1 through 4 before designing application programs with the Programmatic Interface presented in Section 5.

How The Information Is Presented

Section 1 reviews the components of a Laser Printing System and provides an introduction to IFS/3000.

Section 2 reviews the terminology and concepts utilized by the components of a Laser Printing System.

Section 3 explains the interactive operation of the IFS/3000 program and reviews the menus it displays on the terminal screen. You use these menus to define the format for your document and to supply device-specific printing instructions.

Section 4 describes the Laser Printing System Interpreter which gives you additional non-programmatic control over document format.

PREFACE (continued)

Only **Section 5**, the Programmatic Interface, requires knowledge of a programming language.

Appendix A reviews related MPE Commands and Intrinsics. **Appendix B** discusses use and updating of figure files. **Appendix C** presents Environments supplied with the IFS/3000 Formatting Package and the HP Printer Support Package. **Appendix D** presents Character Fonts supplied with the IFS/3000 Formatting Package and the HP Laser Printer Support Package. **Appendix E** defines Vertical Format Control specifications. **Appendix F** provides sample programs. **Appendix G** reviews IFS/3000 Error Messages. **Appendix H** defines the IFS/3000 PCELL File Merge Utility and its application. **Appendix I** provides information regarding impact printer graphics.

References to Other Documentation

To use IFS/3000 effectively, you should know how to operate the Hewlett-Packard terminals used at your facility. Refer to the User's Manual for the terminal you will be using.

If you require character fonts in addition to those supplied with IFS/3000, you can find information on designing them in the IDSCHAR Reference Guide.

To design your own forms, refer to the IDIFORM Reference Guide.

If you use the Programmatic Interface, you may need the manual for the programming language in which you are coding. The following manuals contain information you might need as a supplement to this reference guide:

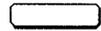
IDSCHAR Reference Guide (36581-90001)
IDIFORM Reference Guide (36581-90002)
Laser Page Printer Operator's Handbook
(02682-90912)
2680A Service Manual (02682-90904)
2688A Operator's Manual (26088-90901)
MPE Intrinsics Reference Manual
(30000-90010)
MPE Commands Reference Manual
(30000-90009)
Using the HP 3000 (03000-90121)
Using Files (30000-90102)
2645 Display Station User's Manual
(02645-90001)
2645 Display Station Reference Manual
(02645-90005)
2626A Display Station User's Manual
(02626-90001)
2626A Display Station Reference Manual
(02626-90002)
2647A Reference Manual (02647-90002)
2647F Reference Manual (02647-90037)
2648A Reference Manual (02648-90002)
TDP/3000 Reference Manual (36578-90001)
HPWord Reference Guide (32120-90001)
DSG/3000 Reference Guide (32250-90001)
HPDraw Reference Guide (32108-90001)
HPEasyChart Reference Guide (32109-90001)

THIS PAGE SHOULD BE BLANK

CONVENTIONS USED IN THIS MANUAL

NOTATION

DESCRIPTION



May be used to illustrate specific keys on the terminal keyboard corresponding to the label within the symbol, such as the **ENTER** key. On some terminal keyboards, abbreviated key labels may differ.

**Function
Label**

May be used to illustrate specific function labels currently assigned to the special function keys labeled f1 through f8 on the terminal keyboard.

OR,

Function

nonitalics

Words in syntax or format statements which are not in italics must be entered exactly as shown. Punctuation characters other than brackets, braces and ellipses must also be entered exactly as shown. For example:

EXIT;

italics

Words in syntax in lowercase italics denote a parameter which must be replaced by a user-supplied variable. For example:

CLOSE *filename*

[]

Within syntax, an element inside brackets is optional. Several elements stacked inside brackets means the user may select any one or none of these elements. For example:

$\left[\begin{array}{l} A \\ B \end{array} \right]$ User *may* select A or B or neither.

{ }

Within syntax, when elements are stacked within braces the user must select one of those elements. For example:

$\left\{ \begin{array}{l} A \\ B \end{array} \right\}$ User *must* select A or B.

CONVENTIONS (continued)

() Elements *within* parentheses *within* a word or a parameter in syntax or format statements are not entered by the user and appear for identification purposes only. For example:

C(OMMAND) User enters C *only*.
Y(*es*) User enters Y *only*.

... Within syntax, a horizontal ellipsis indicates that a previous element may be repeated. For example:

[*,itemname*]...;

underlining When necessary for clarity in an example, user input may be underlined. For example:

NEW NAME? ALPHA

In addition, brackets, braces or ellipses appearing in syntax or format statements which must be entered as shown will be underlined. For example:

LET *var*[[*subscript*]] = *value*

Δ When necessary for clarity, the symbol Δ may be used to indicate a required blank or an exact number of blanks. For example:

SET[(*modifier*)]Δ(*variable*);

TABLE OF CONTENTS

Overview	vii
How To Use This Manual	viii
How The Information Is Presented	viii
References to Other Documentation	ix
Conventions Used in this Manual	xi

Section 1

Introduction	1-1
Overview of the HP Laser Printing System.	1-1
The HP Laser Printer	1-1
Features of the Laser Printing System.	1-3
Laser Printer Hardware	1-3
The HP Non-Graphics Software.	1-6
For the Non-Programmer	1-6
For the Programmer	1-6
HP Environments Supplied With The Laser Printing System.	1-6
Customization with IDS/3000.	1-7
IDSCHAR.	1-7
IDSFORM.	1-8
Interactive Formatting with IFS/3000	1-8
The Programmatic Interface.	1-9
The LPS Interpreter	1-9
HP Graphics Software	1-11

Section 2

Terminology	2-1
Overview.	2-1
IDSCHAR Terminology	2-2
Character Cell	2-2
Character Code Number	2-3
Character Font	2-4
Character Font Number	2-4
Primary and Secondary Character Fonts	2-4
Cell File.	2-5
Character Font File.	2-5
Logo File	2-5
Character Font Orientations	2-6
Character Font Sizes	2-6.1
IDSFORM Terminology	2-7
Forms	2-7
Subforms	2-8
Forms File	2-8
Multi-Copy Forms	2-8
IFS/3000 Terminology	2-10
Default Environments	2-10
Environment File.	2-10
Compiling the Environment	2-11
Specifying the Environment	2-12
The Physical Page.	2-12
Physical Page Copies	2-13
Logical Pages	2-13
Logical Page Orientations.	2-14
Logical Page Numbers	2-14
Active and Inactive Logical Pages	2-14
Vertical Format Control	2-19

CONTENTS (continued)

Graphics Terminology	2-20
Decision Support Graphics DSG/3000	2-20
HPEasyChart	2-21
HPDraw	2-21
Figure Maker	2-22
Figure	2-22
Figure File	2-23
Partitioned Raster Image	2-23
Partitioned Raster File	2-23
Figure Rotation and Conversion	2-24
Printing a Figure	2-27
Printing a Partitioned Raster Image	2-27
Positioning a Partitioned Raster Image	2-28

Section 3

Using IFS/3000 Interactively	3-1
Overview	3-1
Requirements for Running IFS/3000	3-1
IFS Menu Conventions	3-2
Using The Terminal Keyboard	3-3
The Special Function Keys	3-4
The Terminal Control Keys	3-6
The Character Set Keys	3-6
The Display Control Keys	3-7
The Field Edit Keys	3-8
Using IFS/3000 Menus	3-9
Log-On and Run IFS/3000	3-11
The Environment File Menu	3-12
Field Discussion	3-13
Function Keys	3-12

The Main Menu	3-14
Function Keys	3-15
Field Discussion	3-15
1. The Selection Field	3-15
2. The Device Field	3-19
3. The Font/Page Number Field	3-20
4. The Character Font Name Field	3-20
5. The Recompile Everything? Field	3-21
The Initialize Menu	3-22
Function Keys	3-23
Field Discussion	3-23
1. The HP Defined Environment Field	3-23
2. The Reduction Field	3-25
3. The Another Environment Field	3-25
The Physical Page Menu	3-29
Function Keys	3-30
Field Discussion	3-30
1. The Number Of Copies Field	3-30
2. The Multi-Copy Forms? Field	3-31
3. The Physical Page Size Fields	3-31
4. The Primary and Secondary Font Fields	3-33
5. The Default Measurement System Field	3-34
6. The Default Form File Field	3-34
The Multi-Copy Forms Menu	3-35
Function Keys	3-36
Field Discussion	3-36
1. The Logical Page Field	3-37
2. The Form Field	3-37
3. The Form File Field	3-38
4. The Scale? Field	3-40
The Logical Page Menu	3-41
Function Keys	3-42
Field Discussion	3-43
1. The Logical Page Number Field	3-43
2. The Initially Active? Field	3-44

TABLE OF CONTENTS

3. The Orientation Field	3-44	The Copy Menu	3-75
4. The Change Forms or VFC? Field	3-45	Function Keys	3-76
5. The Size Specification and Positioning Fields.	3-46	Field Discussion.	3-76
6. Actual Spacing.	3-51	1. The Environment File	
The Logical Page Forms Menu.	3-52	To Copy From Field	3-76
Function Keys	3-53	2. The Parts To Copy Fields	3-77
Field Discussion.	3-54	3. The Method Of Copy Field	3-79
1. The Change VFC Specification Field	3-55	4. The Copy Associated Compiled Parts? Field	3-80
2. The Form 1 Field	3-55		
3. The Form File 1 Field	3-55		
4. The Scale? Field	3-56		
5. The Position on Page Fields	3-56		
6. The Manual Positioning Field	3-57		
The Vertical Format Control Menu	3-59		
Function Keys	3-60		
Field Discussion.	3-61		
1. The Type of VFC Field	3-61		
2. The Line Position Field	3-62		
3. The VFC File Field	3-63		
4. The Top Margin Field	3-63		
The Character Font Menu.	3-65		
Function Keys	3-66		
Field Discussion.	3-67		
1. The Font Number Field	3-67		
2. The Orientation Field	3-68		
3. The Font Name Field	3-69		
4. The Font File Field	3-69		
5. The Font Size Field	3-70		
6. The Match Size Field	3-72		
7. The Units Field	3-72		
8. The ASCII Character Codes Field	3-72		
9. The Protected Information Fields	3-73		
		Section 4	
		Laser Printing System Interpreter (LPS Interpreter)	4-1
		Overview.	4-1
		Intrinsic Calls	4-3
		Text Files	4-3
		Text.	4-3
		Commands	4-3
		Continuation Lines	4-4
		Comments.	4-4
		Default Parameters	4-4
		Shift Character	4-5
		Output File	4-5
		Carriage Control.	4-5
		Running the LPS Interpreter	4-6
		Capabilities	4-7
		Output Device.	4-7
		Environment File	4-7
		Sessions.	4-7
		Interactive	4-7
		Non-Interactive	4-8
		Batch Job	4-8
		Ending A Session.	4-8
		Terminology	4-9
		LPS Interpreter Commands	4-9

CONTENTS (continued)

Abbreviations	4-9
LPS Interpreter Commands	4-11
.ACTIVATEPAGE.	4-13
.COMMENTCHAR	4-14
.CONTINUECHAR	4-15
.CONTROLCHAR.	4-16
.CONVERTFIGURE.	4-17
.CONVERTRASTER	4-19
.DEACTIVATEPAGE	4-23
.DEFAULTCHAR	4-24
.DELETERASTER.	4-25
.ENDWRITEFIELD	4-26
.EXIT	4-28
.FLASHRASTER	4-29
.HELP	4-32
.INCLUDE.	4-34
.LOADRASTER	4-36
.MOVEPENABS	4-38
.MOVEPENREL.	4-40
.NEWFORM.	4-42
.NEWPAGE	4-43
.NEWPHYSPAGE	4-44
.NEWSUBFORM	4-45
.PRINTFIGURE	4-46
.PRINTRASTER.	4-51
.SELECTPAGE	4-55
.SHIFTCHAR	4-57
.USEFONT.	4-58
.WRITEFIELD.	4-60

LPS Interpreter Example Text Files	4-62
Example 1: Environment File	4-62
Example 1: Text File	4-63
Example 1: Output	4-66
Example 2: Environment File	4-67
Example 2: Text File	4-67
Example 2: Output	4-70
Example 3: Batch Job	4-71
LPS Error Messages	4-72

Section 5

Programmatic Interface.	5-1
Overview.	5-1
Procedure Descriptions.	5-1
Programmatic Intrinsic.	5-2
Calling the Intrinsic.	5-3
Capabilities	5-5
Parameter Data Types	5-5
Byte Arrays	5-6
Communication Area	5-7
User Area Status	5-8
Error Number.	5-8
Error Parameters 1 and 2.	5-8
String Parameter in an Error	5-8
Message Catalog File Number	5-8
Communication Area Examples	5-9

CONTENTS (continued)

IFS/3000 Programmatic Intrinsic	5-11	PINITIALIZE	5-54
HP36580.	5-13	PLOADRASTER.	5-57
HP36580V.	5-14	PLOGPAGEINFO	5-59
PACTIVATEPAGE	5-16	PMOVEPENABS.	5-62
PCONVERTFIGURE[A].	5-18	PMOVEPENREL	5-64
PCONVERTRASTER[A].	5-22	PNEWFORM	5-66
PDEACTIVATEPAGE.	5-27	PNEWPAGE.	5-68
PDELETERASTER	5-29	PNEWPHYSPAGE.	5-70
PERRMSG.	5-31	PNEWSUBFORM	5-72
PFIGUREINFO[A].	5-33	PPRINTFIGURE[A]	5-74
PFLASHRASTER[A].	5-41	PPRINTRASTER[A].	5-80
PFONTINFO.	5-45	PSELECTPAGE	5-84
PFONTNUM.	5-48	PSTATEINFO	5-86
PINITDEVICE.	5-50	PSTRINGWIDTH	5-89
		PUSEFONT	5-91
		PWRITEFIELD	5-93

CONTENTS (continued)

Appendix A

MPE Commands and Intrinsic	A-1
Commands	A-1
Intrinsic	A-4

Appendix B

Using Figure Files	B-1
The Figure Maker Utility	B-1
Converting Other Graphic	
Figures With Figure Maker	B-2
Running Figure Maker	B-2
Interactive	B-2
ASCII Files	B-3
Commands	B-3
Updating Figure Files	B-6
Figure Maker Examples	B-8
Figure Maker Error Messages	B-10

Appendix C

Environments Supplied	
With IFS/3000	C-1

Appendix D

Character Fonts Supplied	
With IFS/3000	D-1
HP 2680 Character Fonts	D-2
About Bar Codes	D-36
HP 2688 Character Fonts	D-40

Appendix E

Vertical Format Control (VFC)	E-1
Standard VFC	E-4
User Defined VFC	E-4
VFC File Format	E-4
Carriage Control	E-8

Appendix F

Sample Programs	F-1
COBOL	F-3
BASIC	F-9
FORTRAN	F-14
SPL	F-20
PASCAL	F-25

Appendix G

IFS/3000 Error Messages	G-1
-------------------------	-----

Appendix H

PCELL File Utilities	H-1
Merging PCELL Files	H-1
Updating PCELL Files	H-3

Appendix I

Impact Printer Graphics	I-1
-------------------------	-----

Index	J-1
-------	-----

TABLE OF FIGURES

1-1 Model 2680A Laser Printer	1-4
1-1a Model 2688A Laser Printer	1-5
1-2 Overview of Laser Printing System	1-10
1-3 Graphics Output in the Laser Printing System.	1-12
2-1 Character Cell	2-2
2-2 Character Font Orientations.	2-6
2-3 An Electronic Form	2-9
2-4 Form Merged With Data.	2-9
2-5 Example of Logical Pages	2-15
2-6 Logical Page and Font Orientations	2-16
2-7 Logical Page Layout	2-17
2-8 Actual Output	2-18
2-9 Raster Image Rotations.	2-26
3-1 IFS/3000 Menu Conventions	3-2
3-2 IFS/3000 Function Key Labels	3-5
3-3 IFS/3000 Menu Map.	3-10
3-4 Environment File Menu	3-12
3-5 The Main Menu	3-14
3-6 The Initialize Menu	3-22
3-7 2 to 1 Reduction with HP-defined Environment	3-27
3-8 4 to 1 Reduction with HP-defined Environment	3-28
3-9 The Physical Page Menu	3-29
3-10 The Multi-Copy Forms Menu.	3-35
3-11 Multi-Copy Forms.	3-39
3-12 The Logical Page Menu	3-41
3-13 Specifying The Location of Logical Pages.	3-49
3-14 The Logical Page Forms Menu.	3-52
3-15 The Vertical Format Control Menu	3-59
3-16 The Character Font Menu.	3-65
3-17 The Copy Menu	3-75
3-18 Copy Field Selection Results.	3-80
4-1 Overview of the LPS Interpreter.	4-2
4-2 LPS Interpreter Formatting Commands	4-11/4-12
4-3 Figure Printed with the .PRINTFIGURE Command.	4-50
4-4 Raster Images Printed with the .PRINTRASTER command	4-54

TABLE OF FIGURES (continued)

4-5 Example 1 Output	4-66
4-6 Example 2 Output	4-70
5-1 Functional Listing of the Programmatic Intrinsic.	5-2
5-2 Calling Intrinsic from Programming Languages.	5-4
5-3 Data Types Allowed for Various Languages	5-5
5-4 Outline of the Communication Area Contents.	5-7
5-5 User Area Status Word Values	5-8
5-6 IFS/3000 Intrinsic Summary	5-11/5-12
5-7 Information Available from PFIGUREINFO	5-34/5-37
5-8 Contents of Page Information Array.	5-60
5-9 Figures printed with the PPRINTFIGURE intrinsic	5-79
5-10 Figures printed with the PPRINTRASTER intrinsic	5-83
5-11 Contents of State Information Array	5-87
B-1 Figure Maker Workspace	B-1
C-1 HP 2680 Environment Formats	C-2
C-2 HP 2688 Environment Formats	C-3

Overview of the HP Laser Printing System

As outlined in the Preface to this Reference Guide, you should determine which components of the HP Laser Printer Support Package are available on your system. You may or may not have all of the capabilities of a fully integrated Laser Printing System which consists of:

- an HP Laser Printer
- the HP non-graphics software which includes:

IFS/3000 Interactive Formatting System
which consists of:

- Formatting Programmatic Interface
- LPS Interpreter Formatting Commands
- HP-supplied Environments
- HP-supplied Character Fonts
- HP Character Font Merge Utility

IDS/3000 Interactive Design System
which consists of:

- IDSCHAR Font and Logo Design
- IDSFORM Forms Design

- the HP graphics software available includes:

- IFS Graphics
 - Graphics Programmatic Interface
 - LPS Interpreter Graphics Commands
 - HP Figure Maker Utility
- HPDraw
- DSG/3000
- HPEasyChart

an HP Impact Printer supported by a subset of the HP graphics software.

Each of these components are reviewed in the discussions which follow in this section.

The HP Laser Printer

The Laser Printer is an electronic page printer which uses electrophotographic technology to print computer generated output.

It is helpful to picture output on the Laser Printer as the conversion of the file to be printed to a pattern of dots (known as bits) which are either turned on or off. These dots "cover" the entire surface area of a piece of paper and is limited to the physical size of the paper and the size of the transfer surface on the drum of the printer.

Controlling the on or off status of the dots is a function of the HP Laser Printer Support subsystem software. Communicating these instructions to the printer along with your file is also a function of the software. The Laser Printer has the capability to recognize and interpret these instructions, store and retrieve the related information when it is needed, and convert the file to a page image and print it.

INTERACTIVE FORMATTING SYSTEM

These printer capabilities are what classify the Laser Printer as an intelligent output device.

The number of dots which comprise a page is determined by the Laser Printer hardware on your system. The number of dots is usually referred to as dot density (or distribution) and is commonly measured in dots per inch. The shape of each dot is also defined by the hardware and determines the way dots fit together on the page. This effects the pattern of dots needed to form a recognizable image. Because of these two dot characteristics, most dot control software is device-dependent. This means precise instructions on dot location and on or off status depend on the output device limitations. Since this may differ between device models, you will be asked to specify the target device when defining instructions to the printer in IFS.

Groups of dots printed on the page become intelligible when the pattern is recognizable as a letter or line or image. Patterns of letters which have an identifiable type style are classified as a character Font. Patterns of lines and characters which form a static image are classified as a Form.

The software included in the HP Laser Printer Support Package enables you to direct the control of these dot patterns in the form of Fonts, Forms, and Text.

Controlling the combination and position of these patterns on the page is called Formatting.

Combining these formatting components along with the specifications for page layout into a single file is called creating an environment file. The environment file specifications are compiled into laser printer compatible formats. This is included in the compiled portion of the environment file. At print time, the environment file is passed to the Laser Printer which interprets incoming formatting commands and applies the defined specifications contained within the environment.

The HP graphics software provides the additional capability of printing Graphic "pictures" such as charts and drawings.

The Laser Printer Interpreter and the Programmatic Interface enable you to merge Graphic images with Fonts, Forms, and Text at the Laser Printer at print time. Your method of control over this integration depends on whether you are a programmatic or non-programmatic user.

Features of the Laser Printing System

The features of the Laser Printer hardware technology and the HP Laser Print Support Package software subsystem are summarized as follows:

- High print speed and print quality.
- A single interactive program for combining formatting components into a single file of output specifications which is interpreted at print time.
- Variable types, sizes and rotations of electronic character Fonts.
- Variable types, sizes and rotations of electronic Forms.
- Variable types, sizes, and rotations of electronic Graphics.
- User customization of Fonts, Forms, and Graphics.
- Hewlett-Packard supplied standard character fonts and formats.
- Words, data, and graphics integration at print time.
- Programmatic and non-programmatic interface for formatting control.

Laser Printer Hardware

Each HP Laser Printer model is described in Figure 1-1 on the following pages. It is important to note the following differences between the printer models:

- The direction of the paper path which established the default "page" orientation.
- The density of the dot distribution which establishes device dependent specifications when creating, compiling, and calling character Fonts and combinations of Forms and Fonts.
- The dot shape and how each dot is "nested" within the dot pattern on the page which affects Font and Logo design and their compiled and printed orientations.

Remember to visualize Laser Printer output as a pattern of dots whose distribution on the page is established by the hardware model on your system. You specify the target output device when working with the formatting subsystem programs (except IDIFORM). This ensures that the components of your overall Format, including the environment, fonts and forms, will be prepared for printing within the limitations of the output target device. When you compile these components in to a single environment file they are verified to be compatible with the output device hardware.

Refer to the Operator's Handbook or Operator's Manual for the the Laser Printer model installed on your system. A list of this documentation is provided in the Preface to this Reference Guide.

INTERACTIVE FORMATTING SYSTEM

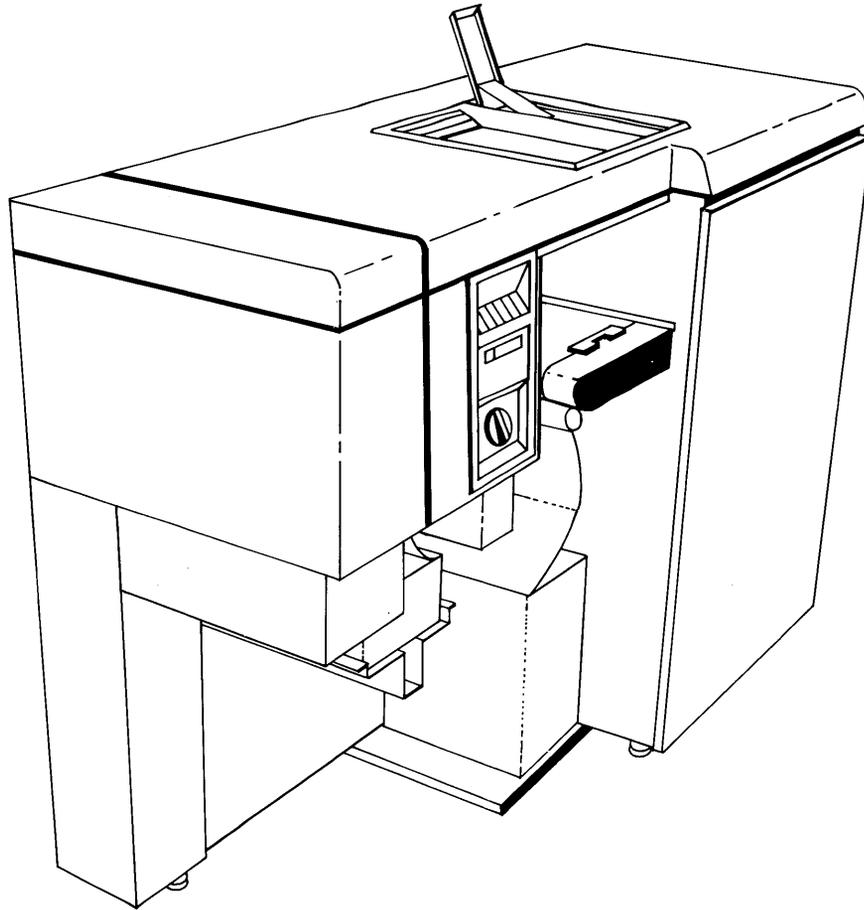


Figure 1-1. Model 2680A Laser Printer

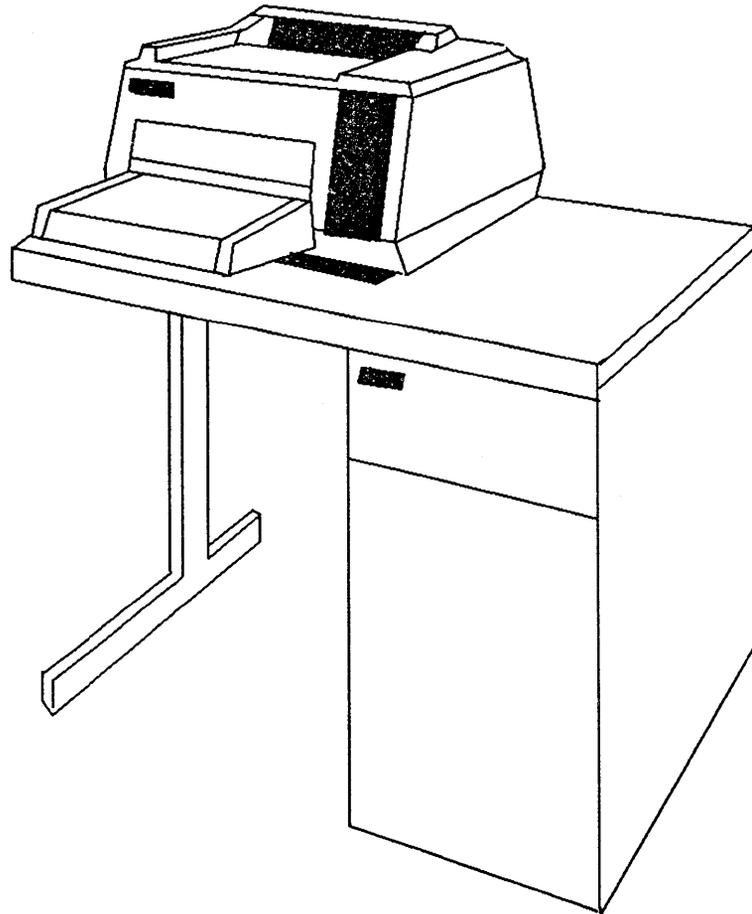


Figure 1-1a. Model 2688A Laser Printer

The HP Non-Graphics Software

Please note that while it is helpful for you to understand how the Laser Printer works, you need not perform all of the tasks necessary to create the components of a formatted document.

Hewlett-Packard supplies some of the most commonly used Environments with the with the Laser Printer and some of the most commonly used Fonts with IFS/3000. The HP graphics software is discussed separately in this section.

For the Non-Programmer

To design documents to be printed on the Laser Printer, you should understand the concepts in this section and Section 2.

If the format of your document matches one of the HP supplied document formats you do not need to run the IFS/3000 Interactive Formatting Program to create and compile a custom environment. Instead, you can use the supplied format by naming it in the MPE `:File` Command to set a file equation setting your output format to the specified system supplied environment. Refer to Appendix A for information on using the `:File` Command and to Appendix C for the formats supplied by Hewlett-Packard. If none of these formats meet your needs, you should read Section 3 which contains information on using IFS/3000 to define your own formats.

Section 4 describes the commands used to control document format using the Laser Printing System Interpreter.

For the Programmer

To design application programs which print user-supplied information in a document with a specified format, you should be familiar with the material in Sections 1 through 4. In addition, you should read Section 5 which describes the formatting procedures called intrinsics. You can use these intrinsics to programmatically direct the unique formatting features of the Laser Printer. Appendix A contains information on the MPE intrinsics and commands which can also be used to control document format.

HP Environments Supplied With the Laser Printer System

The environments supplied with the HP laser printing system include environments which simulate common printing devices such as line printers and typewriters. Using these environments is an effective way to become familiar with the default page orientation of your printer model. It is suggested that you print a test file through a number of these supplied environments so that you become comfortable with the differences between the orientation of a page and the formatting within the page. A discussion of sample applications is included along with the specifications of each environment provided. You will find this approach helpful when progressing to the design of custom formats and environments.

Customization with IDS/3000

The Interactive Design System/3000 consists of two parts:

- IDSCHAR
- IDIFORM

These two interactive, menu-driven programs are included with the HP non-graphics software to provide the capabilities of designing customized character fonts and logos, and forms. They are designed for the experienced user and require an understanding of character design and forms design. Section 2 reviews terminology and concepts you should be familiar with before using IDS/3000. You should refer to the reference documentation for these products as listed in the Preface to this Reference Guide.

When you

```
:RUN IDSCHAR.PUB.SYS
```

OR,

```
:RUN IDIFORM.PUB.SYS
```

the programs will present instructions on your terminal screen in a format called a menu. This menu also lists the choices of functions the program is able to perform at the current menu location within the program. You specify your function choice by using the currently assigned special function keys on your keyboard or filling in blank fields. Any required data is typed within the areas indicated on the menu. These

capabilities are what characterize these programs as interactive, menu-driven programs.

You would use these programs to modify existing character fonts and forms. You use the interactive IFS/3000 program to prepare (compile) the character fonts and forms designed with IDS/3000 for inclusion in an environment file which is the format required to be utilized by the Laser Printer.

IDSCHAR

IDSCHAR is a program which enables you to design and modify character fonts and logos. You do not need to know a programming language to design character fonts with IDSCHAR. You should be familiar with the file system used to store character fonts and the individual characters and logos within a font. You will need a 264X graphics terminal to run IDSCHAR and display the cell file wherein you create or modify your character or logo. For more information on character fonts, refer to Section 2 on Terminology and to the *IDSCHAR Reference Guide (part number 36581-90001)*.

Character fonts are made up of characters where a character can be an alphabetic character, a numeral, a special symbol, or a logo. Hewlett-Packard supplies many of the most commonly used character fonts. Using IDSCHAR, you can add symbols to these fonts, create your own special character fonts, or design business artwork, such as company logos or signatures.

Figure 1-2 illustrates the use of IDSCHAR within the HP Formatting System.

INTERACTIVE FORMATTING SYSTEM

IDSFORM

IDSFORM is a program which enables you to design and modify forms. You do not need to know a programming language to design forms with IDSFORM.

A form is the fixed, static, non-data portion of a document. Your form can contain characters (character fonts, special symbols, logos, or artwork) designed by you with IDSCHAR or provided by Hewlett-Packard. You can also include form graphics such as lines, boxes, and shaded areas.

Using IFS/3000, you indicate the forms you want to use to print your document. For more information on forms, refer to Section 2 on Terminology and to the *IDSFORM Reference Guide (part number 36581-90002)*.

Figure 1-2 illustrates the use of IDSFORM within the HP Formatting System.

Interactive Formatting with IFS/3000

The Interactive Formatting System/3000 is a menu-driven program which enables you to combine selected character fonts and forms with your formatting specifications for document layout.

IFS/3000 is run once prior to running your print job. IFS/3000 then deals with specifications to be

used at run time and ultimately controls the output at print time. You are guided by terminal screen menus presented in logical sequence. Each menu contains highlighted areas called fields in which you type your requirements. Menu conventions and instruction on running IFS/3000 appear at the beginning of Section 3.

IFS/3000 converts format components into a disc file that can be accepted and interpreted by the Laser Printer. This conversion process is called compiling and the resulting package of Fonts, Forms, and Formats is called an Environment File. At print time, the MPE Operating System passes the environment file to the Laser Printer prior to generating the output. This allows system utilities (such as FCOPY, EDITOR and QUERY) or a user-written application program to accept user-supplied data (such as a text file with formatting commands) and print the document as specified in the environment file.

When you compile an environment with IFS/3000, the subsystem confirms that all components of the environment meet the dot distribution specifications of the specified output device. Warnings or error messages are issued when components do not match the target device specified or the file for the component cannot be found. See Appendix G for a listing of the IFS/3000 error messages, meanings, and suggested actions.

The environment file contains the selected components in their compiled format. It can be accessed and transferred within MPE file conventions. Refer to the *Using Files Manual (Part Number 30000-90102)* for more information on maintaining files. Once an

environment has been successfully compiled, it does not need access to the original environment components in order to operate as compiled.

The Programmatic Interface

The programmatic interface consists of procedures called intrinsics which enable a user-written application program to control the Laser Printer. Environment files created in IFS/3000 are callable by the programmatic interface. The user program may be written in COBOL, COBOL II, BASIC, FORTRAN, SPL, or PASCAL. This flexibility allows you to programmatically vary output characteristics such as character fonts or forms while the application program is being run. Thus, the document format can be changed to accommodate user-supplied data.

One of the features of the programmatic interface enables you to print data into a form symbolically using named fields defined in IDIFORM, thus eliminating the need for "print and space" positional forms interfacing. IDIFORM continues to support "print and space" protocol to provide compatibility with existing application programs. The programmatic interface also enables you to do such things as vary the direction of printing and extract information about the widths of proportionally spaced characters for text processing applications.

If you have the HP graphics software, you can also use the programmatic interface to print graphics

output on the Laser Printer, and to merge text with graphics at print time.

Section 5 discusses the details of the programmatic interface.

The LPS Interpreter

The Laser Printing System Interpreter consists of formatting commands which are embedded in a text file to give the Laser Printer formatting instructions at print time. These commands give you the same formatting control as the programmatic intrinsics, but you do not need to know a programming language to use the LPS Interpreter. Text files can be any ASCII file created with an HP 3000 text editor. The LPS Interpreter requests that you specify an environment file as well as the target output device before accepting your text file.

The LPS Interpreter allows you to do such things as print text with varying character fonts on different page layouts. If you have the HP graphics software, you can also print documents which merge text and graphics. Like the programmatic interface, the LPS Interpreter enables you to print data into a form using named fields.

Section 4 discusses the details of the LPS Interpreter.

INTERACTIVE FORMATTING SYSTEM

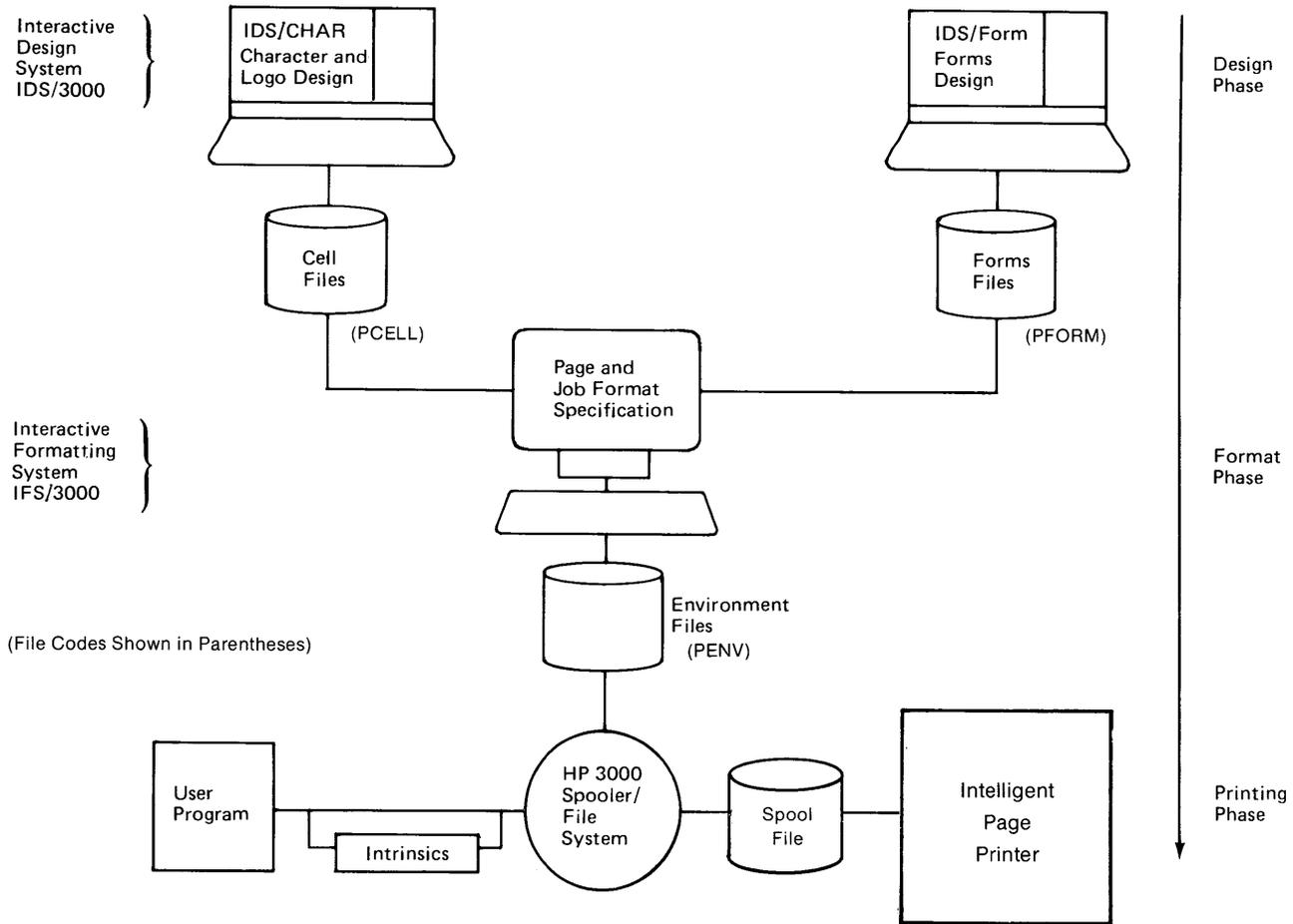


Figure 1-2. Overview of Laser Printing System

HP Graphics Software

The HP graphics software allows you to print figures on the Laser Printer. These figures must first be created with DSG/3000, HPDraw, HPEasyChart, or the graphics Figure Maker utility.

A figure must be converted to a partitioned raster image file before it can be accepted by the Laser Printer.

The IFS/3000 graphics intrinsics and the LPS Interpreter graphics commands allow the programmer and the non-programmer to convert a figure in a figure file into a partitioned raster image file. A copy of this partitioned raster image file is then loaded into Laser Printer memory, printed, and if desired, deleted from Laser Printer memory.

By saving the partitioned raster file you may reprint the raster image without reconvertng the original figure file. Thus, printing a previously converted image saves time and system resources.

Section 4 describes the LPS Interpreter Graphics commands used to control graphics integration with your document by passing formatting commands directly to the Laser Printing System Interpreter at print time.

Section 5 describes the formatting procedures called intrinsics which enable you to programmatically convert figures and integrate them with your document at print time.

Figure 1-3 gives an overview of graphics output on the Laser Printer.

INTERACTIVE FORMATTING SYSTEM

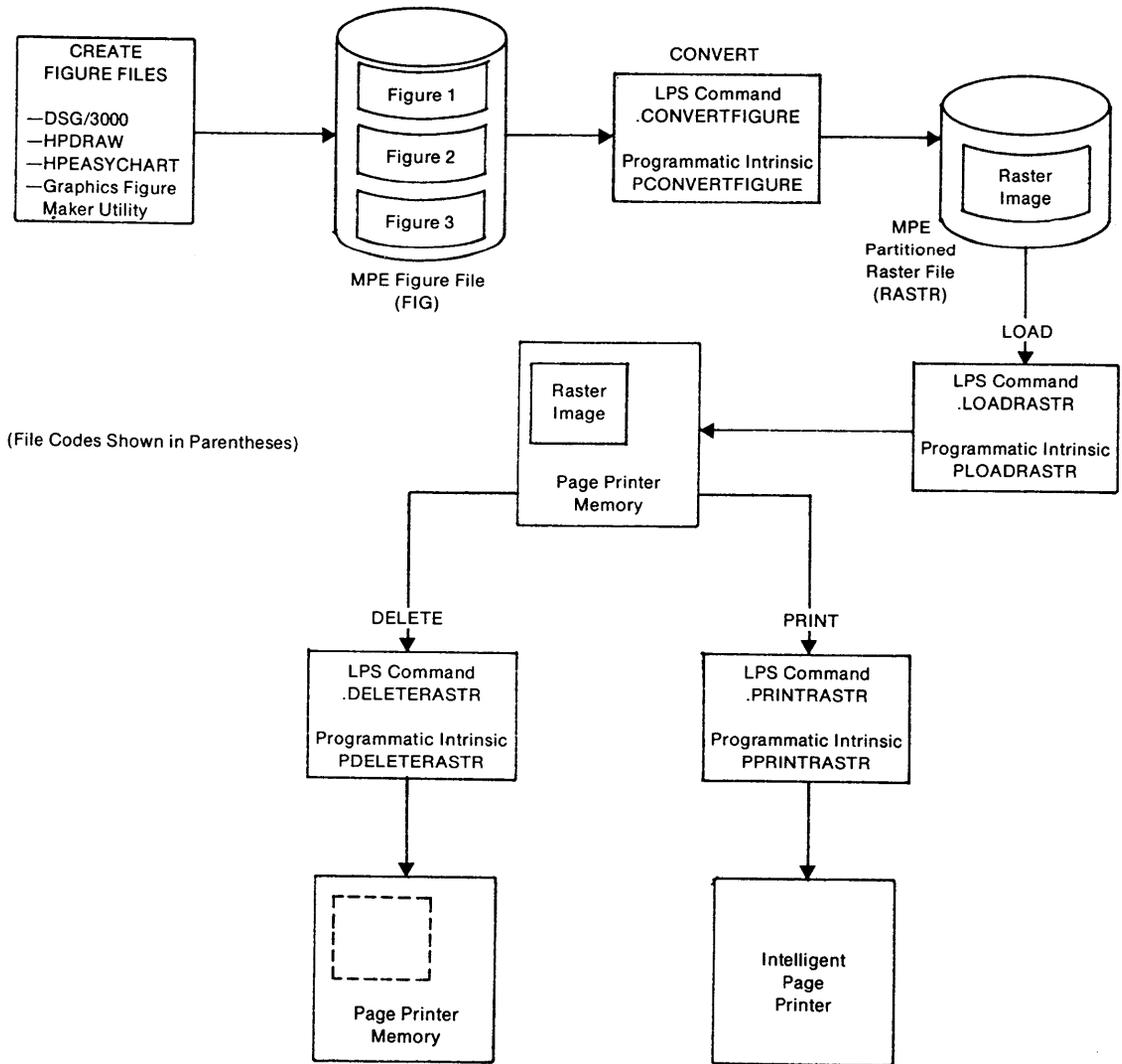


Figure 1-3. Graphics output in the Laser Printing System

Overview

The HP Laser Printer and the formatting software subsystems provide extensive flexibility in defining the format of your printed document. The Interactive Formatting System/3000 enables you to combine the components of the HP non-graphics software into a single file called an *environment file*. This environment file contains specifications for the physical page, character fonts, forms and logical pages. A logical page is a rectangular area on the physical page used for formatting. The environment file provides the following printing capabilities:

- multiple character fonts per line of print
- use of the different sizes available of the same character font or logo
- different orientations of the same character font or logo
- multiple logical pages per physical page
- multiple orientations of logical pages
- up to two forms on each logical page
- multiple copies of each physical page
- graphics charts and drawings on any page

Some of these components are supplied by Hewlett-Packard. All of these components are customizable. You should be familiar with the Terminology and Concepts presented in this section when using IFS/3000 to design custom environments, and when using IDS/3000 to design custom components for the environment.

This section will be particularly helpful when using the the IFS/3000 Merge Utility which enables you to merge different character fonts, sizes, and rotations within the same Character Font File.

For more detailed information on electronic characters and forms and their design, refer to the *IDSCHAR Reference Guide (part number 36581-90001)* and the *IDSFORM Reference Guide (part number 36581-90002)*.

TERMINOLOGY

IDSCHAR Terminology

IDSCHAR is a subsystem of IDS/3000 and is included in the HP non-graphics software.

The IDSCHAR Reference Guide is written primarily for graphics designers and others who have a basic understanding of typesetting terminology. This section will review the basic terminology you will encounter when using IDSCHAR with IFS/3000 to supply customized character fonts or logos for your environment.

You should also review the IFS/3000 Merge Utility in Appendix H which enables you to combine different fonts, sizes, and rotations within the same Character Font File.

IDSCHAR is device dependent, as dot density and dot shape are defined by the printer hardware installed on your system. This will affect the design of Character Fonts and Logos and their use with IFS/3000.

Character Cell

A character designer defines a character cell using IDSCHAR. The content of a character cell is a character which is a pattern of dots within the cell, as shown in Figure 2-1. Typical contents of a character cell could be a letter such as the "B" in our example; or a logo, or other artwork.

The dot pattern within a character cell may range from a single dot to a maximum of 255 dots wide by 255 dots high.

You will need to convert the cell dot size to the standard point size measurement used to designate type sizes on the Character Font Menu in IFS/3000. There are 72 points to the inch. Your Laser Printer dot resolution is expressed in dot density per inch. Divide the dots per cell into the printer dots per inch to determine the cell size in inches for the target output device. Multiply the standard 72 points per inch by the cell size in inches to determine your cell point size. To be consistent, the character within the cell should be positioned within the cell so that it maintains the point size you intend to specify in IFS/3000.

If a character within a character font has not been specified, a solid black rectangle will be printed in place of the undefined character (cell).

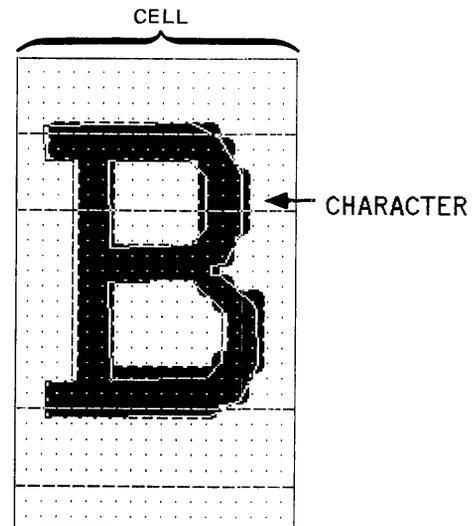


Figure 2-1. Character Cell

Character Code Number

The character designer assigns a character code number to each character during the design process. The Laser Printer uses the character code number along with the character font number to refer to a character when printing it. It is not necessary for you to know the character code number in order to use it for printing. Refer to the paragraph on Character Font Number in this section.

A character font can contain up to 128 characters. The location of the character (cell) within the character font is determined by the character code assigned to a position from 0 to 127. Character codes 0 through 32, and code 127 are reserved for use by IFS/3000. The remaining 94 code positions, 33 through 126, are available for design use. Code position 32 is defined as a blank to be consistent throughout font design.

Since more than 95 character codes are required to fully define any European language, a pair of character sets are linked together. The first 95 characters in the pair are the standard ASCII character set; the second 95 characters in the pair are the Roman Extension to the ASCII character set. Appendix D provides details of these paired character sets and Appendix C lists the supplied environments in which they are used.

A character code number can be used in more than one character font. If a given character is included in several character fonts, the same code number is normally used for that character in each of the fonts. For example, the character code number decimal 67 would normally represent the

letter "C" in all character fonts containing the English alphabet.

For consistency, the character code number is normally the ASCII number for the character. For example, the character codes for the following line of print appear beneath each character, read vertically:

```

ASCII Characters and Character Codes
686773619199111139113619199111361111
537332707179101127102707179101271001
  4 4  61452 00  4 4  614 1615

```

For information regarding ASCII codes refer to the *MPE Pocket Guide (30000-90049)*.

If you do not need all the characters in a given character font, you can save space in your environment file and in the Laser Printer memory by selecting only the block of characters you need. You can identify the desired range of characters by entering the starting and ending character code numbers on the Character Font Menu in IFS/3000.

This is particularly useful when an environment contains a large number of character fonts, or when only a few characters have been designed in a specific font, especially if a large cell size is used. If you ever encounter the printer error "Not enough room for character set download", you may solve the problem by specifying that only a range of characters be used.

TERMINOLOGY

Character Font

A character font is a collection of character cells, where each character is normally in the same typeface and size. Each character in a character font is assigned a character code number as described above. An example of a character font is the Times Roman typeface and 10-point typesize. Hewlett-Packard supplies a selection of character fonts as described in Appendix D.

You can use up to 32 different character fonts to print a document. Laser Printer memory limitations may allow less than this, but 32 is the maximum number you can use in one environment file. You can use these fonts in any combination on a page and can interchange them within a single line of print.

Character Font Number

After you choose the character fonts you want to use to print data in your document, you identify them using the Character Font Menu in IFS/3000. For reference purposes, you associate a number and an optional name with each character font you select. The character font number is used for character font selection at print time.

You can print character fonts in any one of four orientations: Landscape, Portrait, Reverse Landscape, and Reverse Portrait. To reference the various orientations, you must assign a different character font number to each orientation and size

of any given typeface when specifying an environment in IFS/3000.

Primary and Secondary Character Fonts

When fonts are defined using the Character Font Menu, the printer will default to the font which has been assigned the lowest font number as the primary character font. This font will always be initially active. In other words, unless you do something to change it, the printer will use the primary character font when it begins to print a document.

The font which has been assigned the next higher number (using the Character font Menu), will be considered the secondary character font. You can change to the secondary character font and back again at any time as follows:

CHANGING FROM PRIMARY TO SECONDARY CHARACTER FONTS

You can direct the printer to change between primary and secondary character fonts by using the SHIFT OUT (ASCII 14=CTRL N) and SHIFT IN (ASCII 15=CTRL O) characters. You can include these signals in the data or programmatically send them to the printer.

You can also cause a change to the secondary

character set by setting the eighth bit to 1 (on).* If you set the eighth bit on, the Laser Printer uses the secondary character font until you turn the eighth bit off. If no SHIFT OUT was previously in effect, then the printer reverts to the primary character font.

If the eighth bit is off, the printer uses a secondary character font whenever it encounters a SHIFT OUT (CTRL N) and the primary character set when it encounters a SHIFT IN (CTRL O).

In the following example, the secondary character set is a bold font, and SHIFT OUT and SHIFT IN are used to print only the word "BOLD" in the secondary character set:

This is a CTRL N **BOLD** CTRL O font.

You can also change fonts in a text file using the LPS Interpreter USEFONT command and | shift character (i.e. the vertical bar accessed in the shift position). See the USEFONT command description in Section 4 for more information.

Cell File

A cell file is a disc file in which IDSCHAR stores the character cells that make up a character font. Refer to the *IDSCHAR Reference Guide (part number 36581-90001)* for detailed information on cell files. Appendix D describes the cell files supplied with the HP non-graphics software. If you are designing an environment file from scratch, you need to know the name of the cell

*For example, you might set the eighth bit on by configuring some HP terminals to eight bit mode for use with various foreign language keyboards.

files containing the character fonts or logos you want to use to print your document. The character font designer assigns the cell filename when designing the cell file with IDSCHAR.

There are two kinds of cell files:

- Character Font File
- Logo File

Character Font File

A character font file normally contains multiple sizes of the same character font. However, it can contain any combination of character fonts and sizes. You would use the IFS/3000 Merge Utility to merge character fonts of the same font designed for different Laser Printer devices. It is recommended that you do not merge character fonts of different font types.

In a character font file, an "A" chosen from one size should appear almost identical to any other "A" in the same file, except for its size. Appendix D lists the sizes available for the character fonts supplied with IFS/3000.

Logo File

A logo file contains only one logo. There may be several sizes of the logo in a logo file.

TERMINOLOGY

Logos designed with IDSCHAR may be included in a form designed with IDIFORM. A letterhead with a company logo would be an example of this application. The forms designer specifies at design time the cell filename in which the logo resides on the Field Font and Subfield Description Menu in IDIFORM.

To print the form with the logo, you specify the form and form filename on the Logical Page Forms Menu in IFS/3000. IFS/3000 verifies that the logo or any other character font file references can be found and compiled with the form. At this point the logo is part of the logical page and will print when the logical page is activated.

If you wish to print a logo within the text of a document, you specify the logo filename as the cell file on the Character Font Menu in IFS/3000. The logo is automatically given the character code number "76", the ASCII code number for the uppercase letter "L". IFS/3000 automatically includes the logo character in the character font. Since there is only one logo per logo file, both the highest and lowest character code numbers are set to character code number "76". To print the logo, you select the logo file as the character font you want to print with and then print the uppercase character "L".

Depending on the logo size, you could consider creating a character font of logos, where each logo has a character cell position within the character font and is called by character position as a special character font. The cell size required would determine if this application is practical.

Character Font Orientations

The capability of printing characters in different orientations is one of the features of the Laser Printer hardware. Since dot distribution and page orientation are device dependent, you should establish these two specifications for the printer model installed on your system.

The Laser Printer does not rotate characters as it prints them. It uses your instructions to make a selection from the character font orientations included in the environment file.

IDSCHAR allows you to design characters in all four orientations: 0, 90, 180, and 270 degree rotations. Designing a character in more than one orientation is important if the output printing device has asymmetrical dots or grids. You should determine the dot shape for the Laser Printer model installed on your system. If the dot shape is other than round or square, characters will may appear distorted when rotated by IFS/3000. Directly opposite rotations (reverse) should not appear distorted regardless of the dot shape. If orientations are specified, when creating an environment, which IFS/3000 cannot find within the filename specified, IFS/3000 will rotate the existing font within the Landscape, Portrait, Reverse Landscape, and Reverse Portrait orientations as specified on the Character Font Menu. It will not verify that the resulting font may print with a distortion. A separate character font must be defined in IFS for each orientation desired in the environment.

TERMINOLOGY



Figure 2-2. Character Font Orientations

Character Font Sizes

The Laser Printer can print the different sizes of any character font generated by IDSCHAR. The different character font sizes must be compiled for the Laser Printer prior to printing. IFS/3000 will prepare the selected character font for the Laser Printer from the sizes listed in Appendix D, or from a font size you created with IDSCHAR.

You can use different print sizes for purposes such as printing more information in a given space, thus saving paper, or for emphasizing various parts of your document. You can use different character sizes defined in the environment in any combination on a particular page including interchanging them within a single line of print. This feature enables you to use character sizes for titles or illustrations that are different from the character sizes used in the body of the text.

For example, the environment file used to produce this Reference Guide contains the Helvetica Character Font in the following sizes:

10 Point BOLD
12 Point BOLD
14 Point BOLD
20 Point BOLD
24 Point BOLD

IFS\3000 does not generate character font sizes by scaling new dot patterns. Instead, it looks for the character font size you specify on the Character Font Menu in IFS/3000 within the character font filename you specify. It does provide the option of specifying the next smaller or larger size found within the filename provided if the specified size is not found. However, the font size selected with or without this option must exist as a character font.

The designer uses IDSCHAR to design each size of a particular typeface as a separate character font, prior to any reference to it in IFS\3000.

THIS PAGE SHOULD BE BLANK

IDSFORM Terminology

IDSFORM is a subsystem of IDS/3000 and is included in the HP non-graphics software.

The IDSFORM Reference Guide is written primarily for graphics designers and others who have a basic understanding of graphics layout terminology. This section will review the basic terminology you will encounter when using IDSFORM with IFS/3000 to supply customized forms for you environment.

IDSFORM is not device dependent if your form only contains lines, boxes, and shading graphics.

WARNING

If your forms contain Titles, Headings, Logos, or Text, the character fonts specified on the Field Font and Subfield Description Menu in IDSFORM are device dependent. When converting forms for an alternate Laser Printer device, IFS/3000 will issue a error message when compiling the form specified on the Logical Page Menu if the character fonts or logos specified within the form in IDSFORM are not available in the size specified for the device specified in IFS/3000. If the fonts or logos required are not available for the device specified, you will need to run IDSFORM and respecify the character font and logo references to ones that are available. If no existing alternatives are suitable, you would use IDSCHAR to design fonts and logos suitable for the form to be

converted for the alternate printer device. These new fonts and logos would need to be specified in IDSFORM.

Forms

A form is the fixed, non-data portion of a document or report. A form can include titles, headings, logos, lines, boxes, and shading. Figure 2-3 is an example of a form which the Laser Printer can print as a blank form or as a form filled out with data as illustrated in Figure 2-4. Data issued to the form may be formatted (positioned) within the form at print time either programatically or non-programatically. Refer to the PNEWFORM Programmatic Interface intrinsic in Section 5, or the .NEWFORM LPS Interpreter command in Section 4. Your text editor may also provide formatting commands which enable you to position data on the logical page which prints within the form boundaries.

The electronic forms capability allows an application program to dynamically select different forms, possibly dependent on data, and print the data in the form without requiring manual intervention by an operator. A program which reported periodic data on different forms only if data associated with a specific form was present would be an example of this application. You can use electronic forms to replace preprinted forms, eliminating the need to purchase, store, and load different forms for each document.

TERMINOLOGY

Subforms

Subforms are sections within a form and are accessible using the PNEWSUBFORM Programmatic Intrinsic or the .NEWSUBFORM LPS Interpreter command. They are designed in IDIFORM along with associated form. Subforms enhance your capabilities to simulate multi-copy carbons where duplicated formats have modified subforms on subsequent copies.

Forms File

A forms designer defines electronic forms using IDIFORM (see figure 2-3). The forms are stored in a disc file called a forms file. Normally all forms used by a specific application or environment are stored in the same forms file. You can associate forms with the logical page, and multi-copy forms with the physical page. To do this, you enter the form name and forms filename on the Logical Page Menu or the Multi-Copy Forms Table Menu in IFS/3000 when specifying your environment. To fill in the form with data, an application program accesses the form through the associated logical page (see figure 2-4).

The Laser Printer can store up to 30 forms simultaneously depending on the complexity of the forms. You can merge forms in different combinations on each physical page. This is useful

when simulating carbon copies where part of the form is changed from copy to copy. For more information on multi-copy forms, refer to the Section 3 discussion of the Physical Page and Environment Defaults, and Multi-Copy Forms Table Menus.

Multi-Copy Forms

You can simulate multi-copy forms where parts of a form vary from one copy to the next while the data printed to the form remains the same. To do this, you can use the Multi-Copy Forms Table Menu to specify up to eight pairs of forms to be printed on the physical page. In each pair of forms, one form may define the parts which vary from one copy to the next, while the second form may define the parts which remain constant for all copies. By overlaying a pair of forms on the physical page, you simulate one of the pages in a multi-copy form. The next pair of forms can simulate the next page of the multi-copy form, and so forth.

IFS/3000 automatically rotates and positions multi-copy forms so that they are centered on the logical page for the HP2680A model Laser Printer, only.

For more information on multi-copy forms, refer to Multi-Copy Forms Table Menu in Section 3.

EMPLOYEE INFORMATION FORM			
EMPLOYEE NO.	LAST NAME	FIRST	MIDDLE
STREET ADDR		SOCIAL SECURITY NO.	
CITY & STATE		ZIP	
HOME PHONE NUMBER	WORK EXTENSION NUMBER	DATE OF BIRTH (M/D/Y)	
LOCATION NUMBER	DISTRIBUTION	DATE OF HIRE (M/D/Y)	

Figure 2-3. An Electronic Form

EMPLOYEE INFORMATION FORM			
EMPLOYEE NO.	LAST NAME	FIRST	MIDDLE
04987	CHASE	SAM	NOAH
STREET ADDR 1234 EAST APPLE ROAD		SOCIAL SECURITY NO.	
CITY & STATE BOISE, IDAHO		ZIP 83707	
HOME PHONE NUMBER	WORK EXTENSION NUMBER	DATE OF BIRTH (M/D/Y)	
922-4328	2680	12/31/53	
LOCATION NUMBER	DISTRIBUTION	DATE OF HIRE (M/D/Y)	
4340	PERSONNEL	7/04/73	

Figure 2-4. Form Merged With Data

IFS/3000 Terminology

IFS/3000 is a menu-driven program supplied with the HP non-graphics software. It enables you to combine selected character fonts and forms with your formatting specifications for document layout.

When using an environment you have created yourself, IFS/3000 is run once prior to running your print job. A compiled version of your specifications is prepared which can be passed to the Laser Printer, and which ultimately controls your output at print time. You are guided by terminal screen menus presented in logical sequence. Each menu contains highlighted areas called fields in which you type your requirements. Menu conventions and instruction on running IFS/3000 appear at the beginning of Section 3.

IFS/3000 is device dependent which will affect your ability to combine device dependent components of your formatting specifications. Warning and Error Messages are issued when problems are encountered while compiling your specifications for the specified output device. Refer to Appendix G to review Error Messages, their meaning, and proposed actions.

Default Environments

The Laser Printer contains a default environment consisting of a simulated line printer character font, no forms, and a single logical page per physical page with 132 characters per line and 66 potential print lines per page. The physical page in the default environment has proportions similar

to a 14 x 11 inch line printer page, but is only 11 inches wide and 8.5 inches high. This default environment is always present in the Laser Printer unless it is overwritten by an environment you define. Refer to Appendix C for the default environment specifications.

If you do not wish to use the Laser Printer default environment just described, you can use one of the environments supplied with the Laser Printing System. Appendix C describes the supplied environments. You can also create your own environment using IFS/3000. To do this, refer to Section 3 of this manual after reviewing the terminology presented in this section.

Environment File

The environment file identifies the character fonts, forms, and other printing requirements needed by a particular print job or a set of print jobs. The environment file is in a format that is specific to a particular target output device.

An environment file consists of two parts:

- The physical page, logical page(s), and character font(s) specifications.
- The compiled version of these components and specifications.

You use IFS/3000: IDSCHAR and IDIFORM, to design and modify specification components of an

TERMINOLOGY

environment. IFS/3000 combines these components in five steps:

- Requests the definition of the physical page.
- Accepts character cell file references for character fonts and logos you wish to have available to print on the physical page.
- Accepts logical page definitions.
- Accepts forms file references for forms you wish to include in the definition of logical pages to be printed on the physical page within, or less than, the size limits of the physical page.
- Compiles these specifications by locating the files referenced and converting them to into a format that can be passed as a single file, called an environment file, to the Laser Printer.

Compiling the Environment

Any character cell or forms file references you make on the Character Font and Logical Page Menus in IFS/3000 are verified when you press **ENTER** when entering the filenames. Warning messages are issued when a file reference cannot be found, or a specification does not fit within the limits of the physical or logical page. A successful file reference only indicates that the file is accessible to IFS/3000. It does not verify that the contents of the file meet the specifications of the target output device.

When you have completed your environment component references, you are ready to compile

the environment to convert components that are not already device specific to a file format that is device specific.

This procedure is called compiling the environment. Each page, font and form are prepared for the Laser Printer model you specify; including the orientation, size, and formatting specifications of each component. IFS/3000 assigns the number you specify on the Logical Page and Character Font Menus to the compiled components, along with optional names if also specified. The assigned number is used by the Laser Printer to locate, activate, or deactivate fonts, logical pages, forms or subforms as instructed by the application program or data file. Refer to the discussions on character fonts and logical pages. Instructions can be issued either programmatically or non-programmatically, within or along with a data file sent to the Laser Printer preceded by the environment file designation. The font, form, or page must exist in its compiled form in the environment specified to be used by the Laser Printer and is applied to the data file at print time.

The dot density and dot shape are determined by the hardware model you specify on the Main Menu in IFS/3000. You should review the IDSCHAR and IDIFORM terminology in this section to determine the effect of converting and compiling dot specific patterns for the target output device.

The compiled environment file maintains the specified components in the Laser Printer format. It no longer needs the files referenced unless the environment file is opened with IFS/3000 and completely recompiled.

TERMINOLOGY

If you make changes to the specifications without compiling the environment, the compiled part which existed prior to the changes still exists. The changed specifications are available for printing the document after the environment has been recompiled. However, if the compile step does not complete successfully, the compiled part of the environment file is not available. You cannot use the environment for printing the document until it is successfully recompiled.

If the compile step fails, IFS/3000 displays error messages indicating the problems encountered. You may have to exit IFS/3000 to modify a design component if the error is due to a file reference or error within the component. After you correct the errors, return to the IFS/3000 Main Menu to recompile the environment.

IFS/3000 recompiles the parts of the environment which have not already successfully compiled and always recompiles the logical and physical pages. You may repeatedly make corrections or changes and recompile the environment until all parts are successfully compiled.

Specifying the Environment

Your compiled environment file must be associated with the document you want to print. You can do this in several ways:

1. Include your environment filename as a parameter in the MPE :FILE command for your output print file. You may use a :FILE command with supplied subsystems such as EDITOR, TDP/3000, QUERY or FCOPY, as well as with your own programs. For example:

```
:FILE JOB;DEV=PP;ENV=PICA.HPENV.SYS
```

OR,

```
:FILE JOB;DEV=PP88;ENV=PICA.HPENV.SYS
```

2. Use your environment filename in the MPE FOPEN intrinsic within the related application program.

Refer to Appendix A for information on identifying the environment file in either the MPE :FILE command or the FOPEN intrinsic. For more information on using files on the HP 3000, refer to the *Using Files Manual (part number 30000-90102)*.

3. Use the \ENVIRONMENT *filename* command if you are printing a TDP/3000 text file. For more information, refer to the *TDP Reference Manual (part number 36578-90001)*.
4. Supply the name of your environment file to the LPS Interpreter if you are printing a text file. For more information, refer to Section 4 of this reference guide.

The Physical Page

The physical page refers to the actual paper surface that passes through the printer and is exposed to the printer's electrophotographic transfer drum. Its physical size is determined by the Laser Printer model installed on your system. Refer to the Operator's Handbook for your printer model to determine the paper and, therefore, the physical page sizes available.

The physical page dimensions are specified on the Physical Page and Environment Defaults Menu in IFS/3000. Its size establishes the maximum dimensions available for printing all other components of your environment. Each printer model also has unique border areas along the edges of the physical page which are outside of the transfer area of the printer transfer drum. Any specifications you define which extend beyond the physical page size or fall within the border areas will result in warning or error messages specific to the limitations exceeded.

The Laser Printer and the software subsystems which support its operation enable you to format the print area on the physical page. Formatting of the physical page consists of two basic sets of specifications:

- subdividing the physical page area in the form of logical pages.
- defining the vertical spacing per print line and between the lines of print.

Physical Page Copies

You can simulate carbon copies of a single physical page document by entering the number of copies on the Physical Page and Environment Defaults Menu. Copies defined in this way are generated by the Laser Printer itself, rather than by the host computer system.

To produce multiple copies of a multi-page document using the Laser Printer, you have two alternatives:

- request multiple copies of each page on the Physical Page and Environment Defaults Menu and then collate the pages manually.
- retransmit the entire document from the host multiple times. If retransmitting is desired, you do not have to run the application program multiple times but rather, you can specify the additional copies via the spooler.

Logical Pages

A logical page is a rectangular area of any dimensions within the limits of the physical page, usually associated with a form, subform, or print area. It can be used to define the print boundary and alternate print margins as well as interline spacing. This enables you to vary the interline spacing from one logical page overlay to the next within one physical page.

It is helpful to view a logical page as an overlay on the physical page that varies in size from the physical page. You can specify a minimum of 1 and a maximum of 32 logical page overlays for the physical page. Figure 2-5 shows the placement of several logical pages on a single physical page.

You specify the logical page dimensions and its location on the physical page on the Logical Page Menu. It may contain up to two forms. The forms may extend outside the logical page print boundaries as long as they remain within the physical page boundaries, but you cannot print

data to the area of the form which extends beyond the logical page boundaries.

Logical Page Orientations

When defining a logical page on the Logical Page Menu, you must specify one of the following page orientations: Landscape, Portrait, Reverse Landscape, or Reverse Portrait. When referencing a logical page programmatically the orientation is given in degrees. The initial page orientation (0 degrees) is dependent in the direction of the paper movement through the output device and is, therefore, device dependent. The following table illustrates this relationship.

LOGICAL PAGE ORIENTATION		
LOGICAL PAGE MENU	2680A	2688A
Landscape	0	270
Portrait	90	0
Reverse Landscape	180	90
Reverse Portrait	270	180

The logical page orientations rotate in increments of 90 degrees and are illustrated in Figure 2-6.

You can compile multiple logical page orientations within an environment.

You can also rotate the direction of data to be printed on the logical page as shown in Figure 2-8. The compiled rotation of the character font to be used to print the data must be included in the environment file and should agree with the

orientation of the logical page. Figure 2-7 shows the use of several character font and logical page orientations.

Logical Page Numbers

You specify the order in which you want logical pages printed on the physical page by assigning a number to the page when you define it on the Logical Page Menu. The Laser Printer prints (overlays) logical pages on each physical page in numerical order, from 0 to 31. A physical page eject follows the highest ranked page and the numerical sequence is repeated on the next physical page. This cycle is maintained for the entire print job unless your application program or data file provides instructions to alter the compiled logical page hierarchy.

Active and Inactive Logical Pages

You may choose to set logical pages as initially active or inactive and compile your environment. Setting them all inactive requires each logical page used to be activated individually by your application program or through commands within your data file. You specify whether the page is initially active or inactive when defining it on the Logical Page Menu.

You can change the initial page status for the current print job using the PACTIVATEPAGE and PDEACTIVATEPAGE or the PSELECTPAGE programmatic intrinsics; or, the LPS Interpreter .ACTIVATEPAGE and .DEACTIVATEPAGE or .SELECTPAGE commands.

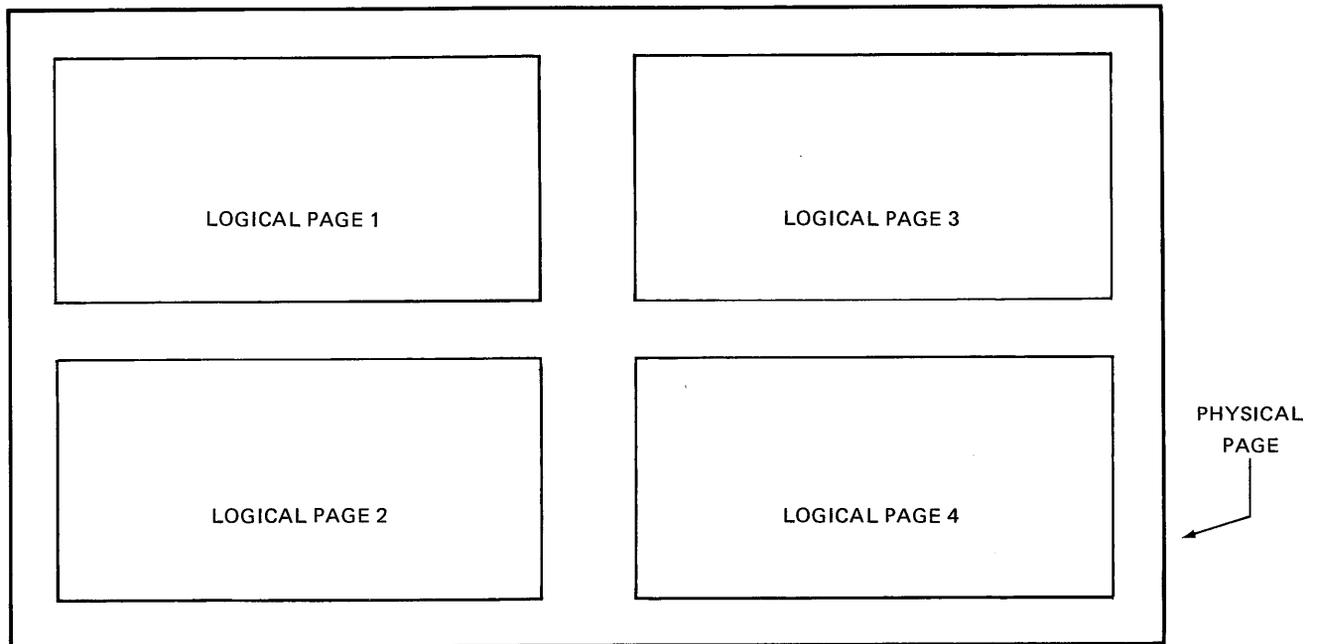


Figure 2-5. Example of Logical Pages

TERMINOLOGY

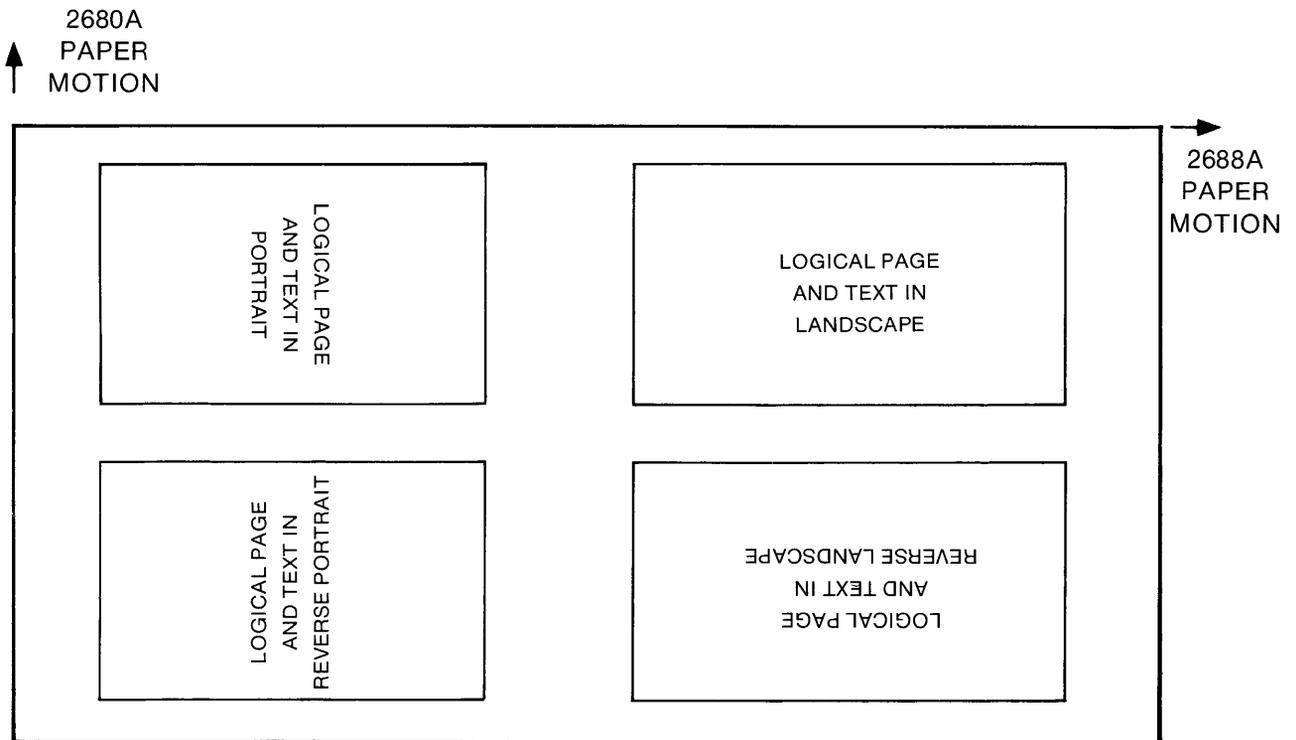


Figure 2-6. Logical Page and Font Orientations

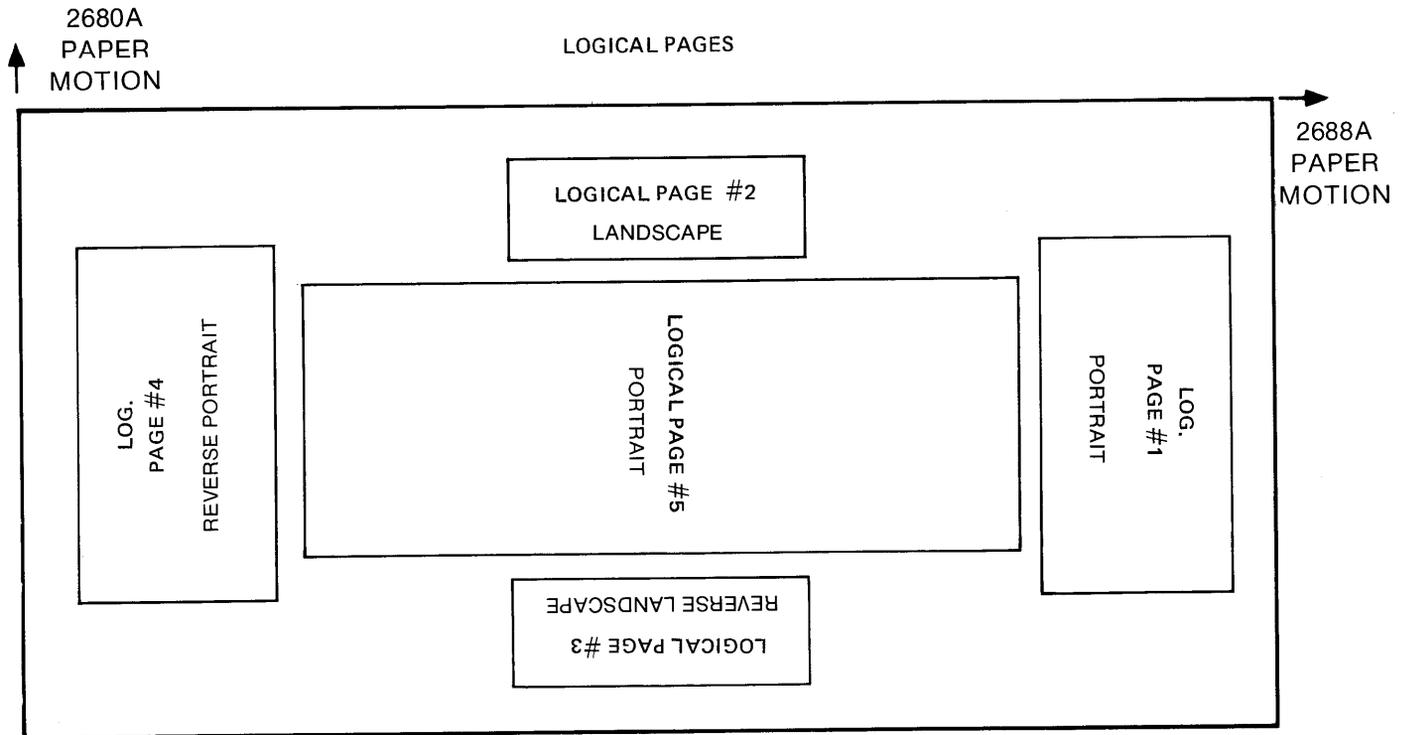


Figure 2-7. Logical Page Layout

TERMINOLOGY

HP 2680A INTELLIGENT PAGE PRINTER
WORDS/DATA/GRAPHICS/WORDS/DATA/GRAPHICS

Table Of Contents

Line Printer Applications

"General Ledger Report" - Creating more manageable line printer output.

"Calendar Compiler Listing" - Line printer output in a portrait format.

"General Ledger" - 2:1 and 4:1 reduction. High information density for maximum efficiency.

"Calendar Compiler" - 2:1 and 4:1 reduction. High information density in portrait format.

2680 Forms Applications

"Marketing Composite" - Hard-to-read reports made easy with shading.

"Routing Master" - Printing of forms and data simultaneously.

"Purchase Order" - Replacing multi-part forms with custom duplicates.

"Bar Code" - Bar code technology for state-of-the-art materials handling.

"Form 1040" - Reproduction of complex forms is featured.

Text and Graphics Applications

"Memo" - Text, data and graphics features. Enhanced reporting of information.

"Production Cost Reductions" - A typical technical documentation application.

"Raster/Vector" - Graphics reproduced from raster and vector format.

"HP and Non-HP Input" - The complete solution for electronic merging of words, data and graphics.

HP 2680 Character Sets

Listed alphabetically and by point size, characters per inch, lines per inch and proportional or non-proportional configuration.

HP 2680A INTELLIGENT PAGE PRINTER
WORDS/DATA/GRAPHICS/WORDS/DATA/GRAPHICS

Figure 2-8. Actual Output

Vertical Format Control

The Laser Printer has the ability to print different character font sizes within a page. If font size differences are extreme, or you wish to redefine areas of the physical page for different font sizes, you can change the Vertical Format Control via the Logical Page Menu. Vertical Format Control (VFC) establishes the number of print lines on a page, and the amount of space to be placed between each line of print.

IFS/3000 automatically generates an appropriate VFC for each logical page calculated on the font for VFC specified for the page or the environment, unless you specify otherwise.

NOTE

When you design character fonts, or combine character fonts within an environment that contain line drawing sets or simulated graphics produced by character ordered symbols, the VFC must match the VFC specifications of the page. Otherwise, characters or symbols designed to meet end to end vertically, may not connect. If your special character fonts share the standard point size and spacing as the font for VFC used for the Logical Page VFC, your environment logical page and character font combinations should produce satisfactory results.

If your application requires a unique VFC, you can specify a custom-designed VFC on the VFC Selection Menu via the Logical Page Menu. You can also specify "No VFC", in which case the spacing between the lines depends on the cell height of the character currently being used for printing. In environments where there is a large range of point sizes, mixing point sizes on a single line of print may cause overlapping or irregular vertical spacing.

Graphics Terminology

The HP graphics software includes a number of subsystems which support the design of charts, graphs, drawings, and illustrations. Section 1 summarizes how the HP graphics software provides document illustration capabilities.

This section provides a brief review of the terminology you will encounter when selecting a program, saving the illustration you have created as a figure file, and converting a figure to a partitioned raster image for printing on the Laser Printer.

These discussions also make references to the LPS Interpreter graphics commands and the Programmatic Interface graphics intrinsics which you will find in Sections 4 and 5 respectively.

If you are using your text processor to call the graphics commands and intrinsics indirectly, you should also be familiar with any terminology and procedures specific to your processor. It may be that your processor provides commands which you embed within your data file that call the graphics intrinsics for you when you final your document to the Laser Printer. TDP/3000 and HPWORD provides extensive capabilities for integrating graphics by embedding commands within your text file that automatically call the graphics intrinsics. Refer to the Reference Manuals for these products for more information on editor/formatter assisted graphics integration.

Decision Support Graphics DSG/3000

DSG/3000 is a graphics design subsystem that runs on the HP 3000. You may call the DSG/3000 intrinsics programatically to produce charts using an application program; or, you may run the DSG/3000 menu-driven program, GRAPH, to design charts and graphs interactively.

GRAPH enables you to enter tabular data to be converted to a chart or graph, make annotations within the graph format, and specify the fonts to be used for the annotations, headings, and labels.

It enables you to print the chart to the terminal screen, a plotter, or the Laser Printer. When you specify the Laser Printer as the target output device on the Graphing Options Menu, GRAPH automatically converts the graph figure to a temporary partitioned raster image, loads the partitioned raster image into Laser Printer memory, prints the image, and purges it from printer memory.

In addition, you may save a graph as a data file which can be modified or expanded; or save the graph as a figure in a figure file which can be converted to a partitioned raster image. A partitioned raster image is the format necessary to integrate your graph with a document on the Laser Printer at print time.

To integrate a GRAPH figure file with a document at print time, you must initiate the figure to partitioned raster file conversion yourself. The PCONVERTFIGURE programmatic intrinsic or .CONVERTFIGURE LPS Interpreter command enables you to convert GRAPH figure files for the

Laser Printer. The partitioned raster file created may be retained for future use without the need to reprocess the figure file, or it may be deleted to save memory space.

For more information on designing charts, refer to the *DSG/3000 Reference Guide, (part number 32250-90001)*.

HPEasyChart

HPEasyChart is a graphics subsystem designed for the inexperienced user that runs on the HP 3000. The HPEasyChart interactive program, EZCHART, converts tabular data entered via a logical sequence of menus into a chart image. It enables you to print the chart to the terminal screen, a plotter, or the Laser Printer. When you specify the Laser Printer as the target output device on the Plot Chart Menu, EZCHART automatically converts the chart figure to a temporary partitioned raster image, loads the partitioned raster image into Laser Printer memory, prints the image, and purges it from printer memory.

In addition, you may save a chart as a chart file which can be modified or expanded; or save the chart as a figure in a figure file which can be converted to a partitioned raster image. A partitioned raster image is the format necessary to integrate your chart with a document on the Laser Printer at print time.

To integrate an EZCHART figure file with a document at print time, you must initiate the figure to partitioned raster file conversion yourself. The PCONVERTFIGURE programmatic intrinsic or .CONVERTFIGURE LPS Interpreter

command enables you to convert EZCHART figure files for the Laser Printer. The partitioned raster file created may be retained for future use without the need to reprocess the figure file, or it may be deleted to save memory space.

For more information on creating charts with HPEasyChart, refer to the *HPEasyChart Reference Guide, (part number 32109-90001)*.

HPDraw

HPDraw is an interactive graphics design subsystem that runs on the HP 3000. It provides the capability to create illustrations which include charts, graphs, drawings, figures, and text. You can design original illustrations; use charts or graphs saved in HPEasyChart or DSG/3000 as chart or data files; use any figure in an MPE figure file; or, create an illustration using any combination of these sources. HPDraw enables you to create, modify, and save your drawing through a logical sequence of menus and displays your drawing on the terminal screen.

When you specify the Laser Printer as the target output device on the Plotting Menu, HPDraw automatically converts the drawing to a temporary partitioned raster image, loads the partitioned raster image into Laser Printer memory, prints the image, and purges it from printer memory.

In addition, you may save a drawing as a draw file which can be modified or expanded; save the drawing as a figure in a multi-figure file which can be recalled as a figure; or, save the drawing as a partitioned raster image in a partitioned raster file which is the format necessary to integrate your

TERMINOLOGY

drawing with a document on the Laser Printer at print time. You may not recall, enhance, or modify a drawing saved as a partitioned raster image.

To integrate a figure in an HPDraw figure file with a document at print time, you must initiate the figure to partitioned raster file conversion yourself. The PCONVERTFIGURE programmatic intrinsic or .CONVERTFIGURE LPS Interpreter command enables you to convert HPDraw figure files for the Laser Printer. The partitioned raster file created may be retained for future use without the need to reprocess the figure file; or it may be deleted to save disc space.

For more information on designing drawings, refer to the *HPDraw Reference Guide (part number 32108-90001)*.

Figure Maker

Figure Maker is an HP 3000 Graphics Utility which enables you to create a figure and save it in a multi-figure file, independent of any other graphics subsystem.

The Figure Maker Utility may be operated interactively by supplying the utility commands and corresponding parameters at the "@" prompt. Or, you may supply an ASCII filename which includes predefined figure design specifications as a series of utility commands and parameters.

The figure file you create with the Figure Maker Utility can be used in the same way as figure files created with any other HP 3000 graphics subsystem, including conversion programatically or

non-programatically to the partitioned raster image format for output on the Laser Printer.

Appendix B describes the Figure Maker Utility.

Figure

Within the HP 3000 environment, a figure is a graphic image or "picture" which can be displayed on a graphics terminal screen, drawn on a plotter, or printed on a Laser Printer. The HP graphics software subsystems enable you to create these images.

When creating a chart, graph, or drawing, a series of vector instructions is generated which contains the beginning and ending coordinates for each line of active (on) bits which make up the image. The terminal display and plotter image are drawn by activating the bits between these coordinates as it displays or plots the figure.

If you wish to modify your figure at another time, you must save your figure as a chart, data, or drawing file depending on the graphic subsystem used to create the figure. When you save your figure in one of these file formats, the original data used to calculate the vector coordinates is saved. Each subsystem provides a means of retrieving the data file and modifying the information.

If you do not want to modify the figure, you can specify the Laser Printer as the target output device and print the figure; or, you can save the figure in a figure file.

Figure File

The file code for a figure file is 1146 or FIG.

When you save a chart, graph, or drawing as a figure, the vector coordinates are recorded for each line of the image and are stored in a figure file. The data associated with the initial creation of the figure is not saved. Since the figure file does not contain the original data, figures within the figure file cannot be modified. However, if you also saved the chart, data, or drawing file, you can modify the data, save it as a figure of the same name, and overwrite the existing figure.

You can use HPDraw to include a figure as part of a drawing, but you cannot modify the figure itself. When you save a multiple-figure drawing in HPDraw as a figure, all coordinates are stored in vector format and are, therefore, unchangeable.

Since figure files contain the vector coordinates of an image and not the actual bit pattern of the image, the individual figures can be rotated and scaled when prepared for the Laser Printer. This is done by supplying the desired values for the *imageheight*, *units*, and *imagerotation* parameters of the .CONVERTFIGURE or the .PRINTFIGURE LPS Interpreter command, or the PCONVERTFIGURE or the PPRINTFIGURE programmatic intrinsic. Refer to the discussion on Figure Conversion and Rotation in this section.

The Laser Printer does not have the capability to process the individual lines of coordinates stored in a figure file. Therefore, the figure file must be interpreted and converted to a dot pattern representing the figure before it can be accepted and stored in Laser Printer memory.

You would purge a figure file using the MPE :PURGE command.

Partitioned Raster Image

A figure in a figure file which is converted to a format that can be passed to the Laser Printer becomes a partitioned raster image. A partitioned raster image is the actual dot distribution of the entire figure fixed in size and orientation. Each figure must be converted individually and the partitioned raster image created is stored as a single partitioned raster image in a partitioned raster file. The partitioned raster image can be modified, rotated, or scaled using the PCONVERTRASTER command. An alternative would be to reconvert the figure file to a different size and rotation.

It is helpful to view the partitioned raster image as the compiled format of a figure similar to an environment being the compiled format of fonts, forms, and logical pages. In each case, the figure, font, or form is passed to the Laser Printer as a dot distribution pattern that is only available for printing in the dot pattern format passed.

Partitioned Raster File

The file code for a partitioned raster file is 1114 or RASTR.

A partitioned raster file contains a single partitioned raster image. You convert a figure in a figure file to a partitioned raster image in a partitioned raster file using the CONVERTFIGURE

TERMINOLOGY

or PRINTFIGURE LPS Interpreter commands; or, the PCONVERTFIGURE or PRINTFIGURE programmatic intrinsics.

You use a partitioned raster image stored in a partitioned raster file by specifying the *rasterfilename* with the .FLASHRASTER or .PRINTRASTER commands or the PFLASHRASTER or PPRINTRASTER intrinsics.

You would purge a partitioned raster file using the MPE :PURGE command.

Figure Rotation and Conversion

When converting a figure to a partitioned raster image, you specify the rotation value as a parameter to the procedure.

You can invoke the conversion procedure in a number of ways:

- interactively, within the graphics subsystem you used to create the figure.
- interactively, using the LPS Interpreter .CONVERTFIGURE or the .PRINTFIGURE command; or, by specifying a data file containing these commands to be processed by the LPS Interpreter.
- programmatically, using the Programmatic Interface PCONVERTFIGURE or PPRINTFIGURE intrinsics in your application program.
- indirectly using the TDP/3000 Formatter's \ILLUSTRATION command or HPWORD's Figure Space function key.

The method you select will determine the affect of figure rotation in relation to its placement on the physical and logical page at print time.

It is helpful to view the initial figure as being device dependent for the Physical Page. The same figure would be device independent for the Logical Page where the orientation is specified by the environment file.

The initial orientation of a figure is specified as having 0 degrees of rotation when you create the figure. The upper left hand corner of the figure is used to establish the beginning pen location when printing the picture. Refer to figure 2-9 for the various partitioned raster image rotations.

When you specify a rotation other than the initial orientation, the upper left corner vector is rotated in increments of 90 degrees from the upper left hand corner of the Physical Page as specified. All vector coordinates are then recalculated on the new beginning position.

The height of the figure is also viewed in the figure's initial orientation. Therefore, as you rotate a figure, your view of its height also rotates to maintain the initial top-to-bottom perspective.

Your alternatives in invoking the conversion procedure have the following effects:

- When you specifying the Laser Printer as the target output device within the graphic subsystem program and press **ENTER**, the temporary partitioned raster image created is loaded into printer memory. It will be printed at the Physical Page orientation for the Laser

Printer model specified. You can rotate the figure on the Physical Page if the subsystem enables you to provide alternate printing specifications. If you save your figure in a figure file you would use either of the following alternatives to rotate and convert your figure.

- The .CONVERTFIGURE command or the PCONVERTFIGURE intrinsic's parameter strings require the default or a respecified *imagerotation* value. Rotation values are specified as 0, 90, 180, or 270 degrees of rotation on the Physical Page for the Laser Printer model specified. (See figure 2-9).
- The .PRINTFIGURE command or the PPRINTFIGURE intrinsic's parameter strings require the default or a respecified *imagerotation* value. Rotation values are specified as 0, 90, 180, or 270 degrees of rotation on the Logical Page in the environment specified (See figure 4-3). Refer to the Logical Page Orientation discussion in this section.
- When you use the TDP/3000 Formatter's \ILLUSTRATION command, you specify a

figure file name and figure name along with optional rotation, scaling, and positional parameters. Rotation values are specified as 90, 180, or 270 degrees from the 0, or default orientation of the target Logical Page. The orientation of the Logical Page is established by the environment associated with the print job. The TDP/3000 Formatter calls the PPRINTFIGURE intrinsic supplying the parameter values you specify. A temporary partitioned raster image is created, loaded in Laser Printer memory, printed on the target Logical Page, and purged from printer memory. Refer to the *TDP/3000 Reference Manual (part number 36578-90001)* to review the Formatter \ILLUSTRATION command.

- For information regarding use of figures in HPWORD refer to the Figure Space function key in the HPWORD Reference Guide.

For more information on the LPS Interpreter commands, refer to Section 4. For more information on the Programmatic Interface intrinsics, refer to Section 5.

TERMINOLOGY

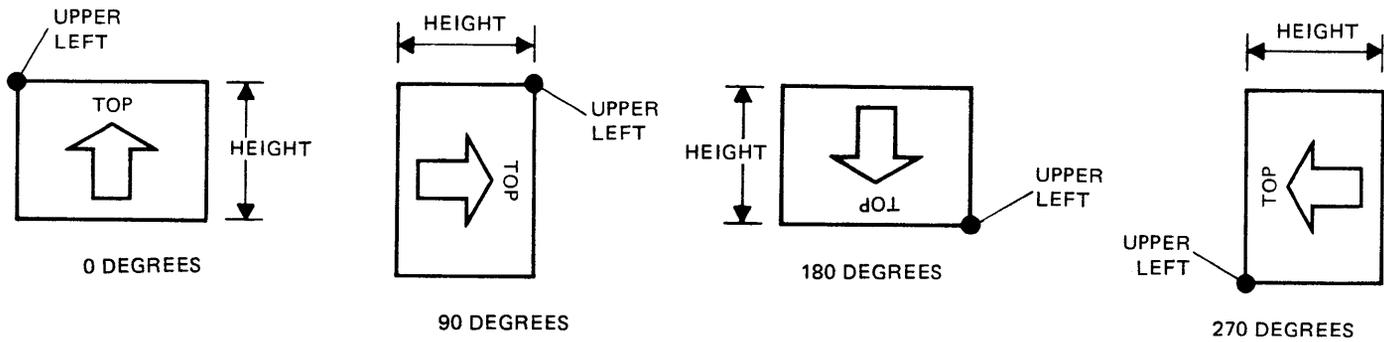


Figure 2-9. Relationship of Height and Origin for Raster Image Rotations

Printing a Figure

The LPS Interpreter .PRINTFIGURE command and the Programmatic Interface PPRINTFIGURE intrinsic perform a number of functions within a single procedure. A figure in a figure file can be rotated and scaled in relation to the target Logical Page, converted to a partitioned raster image, loaded in the Laser Printer, printed on the target logical page, and deleted from printer memory.

You would use this procedure to print a figure that has not been previously converted to a partitioned raster image, and which is to be associated with a specific Logical Page.

Refer to the above discussion on rotation and conversion, and to the procedure discussions in Sections 4 and 5.

Printing a Partitioned Raster Image

When a partitioned raster image in a partitioned raster file is designated for the Laser Printer, a copy of the partitioned raster image is loaded into the printer memory. This copy is classified as either permanent or temporary.

A permanent partitioned raster image is loaded into the Laser Printer once, is available for printing as many times as you require, and is deleted from Laser Printer memory by you after you have printed the last copy required. You would use permanent partitioned raster images if you had to print the same image many times in a print job. You would purge the permanent

partitioned raster image to maintain space in the printer memory for other partitioned raster images you may have specified in that job. All partitioned raster images in the printer are purged automatically at the end of the job.

A temporary partitioned raster image is loaded into the Laser Printer, printed once, and deleted from Laser Printer memory automatically after it has been printed. You would use temporary partitioned raster images if you wished to make a single print of each image where there are a large number of images. By automatically purging each temporary partitioned raster image, you maintain printer memory space for subsequent partitioned raster images.

Since the partitioned raster image is a copy, its permanent or temporary classification refers to its status in Laser Printer memory and not to the status of the MPE partitioned raster file.

Permanent and temporary partitioned raster images can be used simultaneously in a print job. You can specify up to 32 permanent or 64 temporary partitioned raster images, but the combined total must not exceed 64 on any one Physical Page. You should note, however, that the actual number depends on the size of each partitioned raster image and is limited by the amount of memory available in the Laser Printer.

You should also consider the amount of printer memory required to store your environment file. A copy of the environment file is loaded in printer memory first. This determines the available space

TERMINOLOGY

remaining for incoming partitioned raster images as the print job proceeds.

Your use of the following LPS Interpreter commands or the Programmatic intrinsics determines the permanent or temporary status of the partitioned raster image.

When you use the `.LOADRASTER` command or the `PLOADRASTER` intrinsic to copy your partitioned raster image into printer memory, the partitioned raster image is classified as permanent. It will remain in printer memory for the remainder of the print job, or until it is purged with the `.DELETERASTER` command or `PDELETERASTER` intrinsic. You print the partitioned raster image on the current Logical Page using the `.PRINTRASTER` command or `PPINTRASTER` intrinsic. The parameter strings for these procedures enable you to specify the print position for the image on the Logical Page.

The `.FLASHRASTER` command or the `PFLASHRASTER` intrinsic copies a temporary partitioned raster image into printer memory, prints it, and purges it from memory. The parameter string for this procedure enables you to specify the print position for the image on the Logical Page.

It is important to remember that partitioned raster images created using the `.CONVERTFIGURE` command or the `PCONVERTFIGURE` intrinsic

maintain their association with the Physical Page orientation. Since you cannot rotate a partitioned raster image you should consider the orientation of the logical page on the physical page when converting figures for a specific print job. This will ensure that the rotation value you specify using the `.CONVERTFIGURE` command or the `PCONVERTFIGURE` intrinsic will result in a partitioned raster image that prints in the desired orientation on the target logical page.

Positioning a Partitioned Raster Image

When printing a partitioned raster image, the *x*- and *y-position* parameters to the `.PRINTRASTER` and `.FLASHRASTER` commands, or `PPINTRASTER` and `PFLASHRASTER` intrinsics refer to the vertical and horizontal coordinates on the logical page where the image is to be printed (see figure 4-3).

These coordinates establish the beginning print position of the upper left hand corner of the partitioned raster image as rotated in relation to the upper left hand corner of the target logical page. This enables you to specify exactly where you want an image to print within a logical page. The values you specify here have no effect on the orientation of the partitioned raster image and refer only to the print position on the logical page.

Overview

This section provides information you should be familiar with when using the IFS/3000 interactive program to create a custom environment file.

The environment file is the format necessary to associate your document formatting specifications with your data file. The Laser Printer accepts the environment file at the beginning of a print job, applies initially active specifications contained in the environment to the data file, and activates or deactivates alternate specifications based on instructions you associate with the data file as the print job progresses.

Some of the specifications within the environment file can be developed by using data such as fonts or forms stored in files created by other software systems. The process of assimilating these components, providing a means of identifying and activating them individually, and converting them to a single file that can be accepted by the Laser Printer is the function of the IFS/3000 program.

Part one of this section reviews the requirements, procedures, and conventions you need to know to initiate an interactive session with the program on your terminal.

Part two provides a Menu Map of the IFS/3000 program, presents each menu in the logical sequence established by the Main Menu procedures, reviews the operation of the function keys for each

menu, reviews the data entry fields and the effect of the field operation on the information you specify.

Since IFS/3000 enables you to combine components created with other subsystems, all file references you specify must be accessible to the program. In addition, the formatting specifications must be consistent with the hardware specifications of the target output device. In order to assist you when inconsistencies occur, a series of warning and error messages will be issued by the program.

Part three of this section presents the error messages, their general meaning, and suggested courses of action.

Requirements for Running IFS/3000

IFS/3000 requires a CRT terminal connected to an HP 3000 host computer. A graphics terminal is not required as the components of your environment specifications reside as disc files within the MPE file system. IFS/3000 accesses these files but does not display their contents on your terminal screen.

In addition, your terminal must support VPLUS/3000 in order to display the IFS/3000 menus which are the main communication format for your session.

IFS/3000 Menu Conventions

IFS/3000 will present instructions on your terminal screen along with available procedures and areas for entering data and specifications. Each logical set of procedures will appear in a format called a menu.

Each menu will use the same general conventions so that you can expect to find the information presented in a consistent format.

You use the terminal keyboard to type data or specifications on an IFS/3000 Menu. The areas reserved for entering data are called *unprotected fields*. Unprotected fields are the **highlighted** areas in the midsection of the menu.

```
IFS      Menu Title Banner / Error and Status Messages  Environment:  SAMPLE

Sample protected field inquiry?
Sample unprotected field response

Number 1
Name Sample (sample menu comment)
File filename.groupname.accountname

D Protected field specification list
A - Choice A      1 Number -OR- Sample data Name
B - Choice B      Number
C - Choice C
D - Sample specification response

N Sample default value? Y/N

Function Function Function Function      Function Refresh Main Function
```

Figure 3-1. IFS/3000 Menu Conventions

The cursor is positioned by IFS/3000 at the first unprotected field available for data input. Field instructions or a list of available specifications will appear directly above, or adjacent to each data entry field. The field instruction tells you what information should be entered in the field.

Some fields only provide enough space to enter a few characters representing your selection from the choices available. You may type as many characters as will fit within the unprotected highlighted area, but you do not have to fill the entire field. You can use either the **ENTER** key or the cursor control keys to move the cursor to the next field.

Not all of the areas are intended for your use. For example, the menu banner line, the instructions, and the function key labels are *protected fields*. You cannot overwrite or add data in these fields.

When you press the **ENTER** key on your terminal keyboard, the information entered on the menu is issued to the computer. At this time the IFS/3000 program may issue Status, Warning, or Error Messages which will be displayed in the menu banner window. These messages generally indicate that the program performed the menu operation successfully, or encountered difficulty in one of the following areas:

- the program could not find or access the filename entered on the menu, or a designated component within the file is not available as specified.
- the program found inconsistencies between the specifications you entered on the menu and the

target output device limitations, or other specifications you have entered previously.

- the program encountered difficulty or failed to perform the specified operation.

The menu remains on the screen for you to review or modify. IFS/3000 only proceeds to another menu when you press **ENTER** and there have been no changes to the current menu since you last pressed the **ENTER** key, or when you press an appropriate function key.

Using the Terminal Keyboard

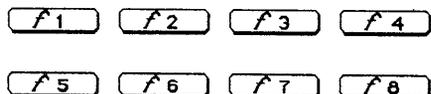
Unless all terminals in your area are identical you will find that some keyboards may have different keyboard arrangements. Take a few moments to become familiar with your keyboard. The IFS/3000 program enables you to use some of the standard keys as they are labeled. Other keys are reassigned by the program to enable you to move from one location in the program to another. When you are comfortable with your keyboard arrangement, you will have a much more efficient IFS/3000 session.

There are five key groups on the keyboard which you will be using as follows:

- The special function keys.
- The terminal control keys.
- The character set keys.
- The display control keys.
- The field edit keys.

The Special Function Keys

The function keys are labeled f1 through f8 and are located on the upper half of your keyboard. On some terminals they are spread across the top; on others they are placed in two rows of four keys each as follows:



Every menu has eight function Key Labels along the bottom. This **Key Label** specifies what operation can be performed if you press the corresponding special function key on your keyboard. Some function keys change labels with each menu change. Experimenting with the function keys is the best way to become comfortable with the logical sequence of IFS/3000 Menus and how the function keys operate.

CAUTION

Any data typed on a menu is not issued to the computer until you press **ENTER**. If you press a function key to move to another menu before you press **ENTER**, the current data will be lost.

The Function Key Layout in Figure 3-2 and the IFS/3000 Menu Map in Figure 3-3 will guide you through the IFS/3000 program. Below are definitions for those functions which you will see frequently.

Key Label

Function



There are eight function key labels along the bottom of each menu. A blank label highlight indicates that no function has been assigned to the function key for that menu.



This will be the function of **f6** on every menu. If the **TERMINAL RESET** key is accidentally pressed or in case of a power failure, press Refresh and the screen will be redisplayed with the last data or information that was issued to the computer when you pressed **ENTER**. Any data typed on the menu before using the **ENTER** key is not available and will not be redisplayed.



This will be the function of **f7** on every menu except the Environment File and Main Menus. You may return to the Main Menu from any menu with this function key. You can then specify another operation or exit the program from the Main Menu.



If you change your mind when entering data on a menu or after you have pressed the **ENTER** key, press **f8** to delete the data and return to the Main Menu.

IFS/3000 Function Key Labels

IFS/3000 FUNCTION KEY LABELS								
MENU NAME	f1	f2	f3	f4	f5	f6	f7	f8
ENVIRONMENT						Refresh		Exit
MAIN						Refresh		Done
INITIALIZE						Refresh	Main	Cancel
COPY						Refresh	Main	Cancel
PHYSICAL PAGE						Refresh	Main	Cancel
MULTI-COPY FORMS						Refresh	Main	Cancel
CHARACTER FONT	First Ft	Last Ft	Prev Ft	Next Ft		Refresh	Main	Cancel
LOGICAL PAGE	First LP	Last LP	Prev LP	Next LP		Refresh	Main	Cancel
LOGICAL PAGE FORMS	First LP	Last LP	Prev LP	Next LP		Refresh	Main	Cancel
VERTICAL FORMAT	First LP	Last LP	Prev LP	Next LP		Refresh	Main	Cancel

Figure 3-2. IFS/3000 Function Key Labels

The Terminal Control Keys

There are four keys in this group which will effect your IFS/3000 session. They are the BLOCK MODE, ENTER, **BREAK**, and RESET TERMINAL keys. Normally, the BLOCK MODE key is used only with the HP2640B terminal. The **ENTER** key may be located in the Character Set Group on some terminal keyboards. You should not need to use the **BREAK** key and RESET TERMINAL keys unless recovery procedures are required. The key functions are defined as follows:

BLOCK MODE Pressing this key puts the terminal in block mode. This enables you to type all data required for a specific menu and make modifications before it is issued to the computer. IFS/3000 automatically puts terminals in BLOCK MODE, except the HP2640B where it displays a message asking you to depress the BLOCK MODE key.

ENTER Pressing this key sends all data you have typed on a menu to the computer. It is your only means of issuing data to the IFS/3000 program. Any data you type on a menu will be lost if you leave the menu before pressing the enter key. Once you press **ENTER**, the program processes your data and may issue Status or Error Messages.

BREAK If you accidentally press this key while in session, you will be exited from the IFS/3000 program. You can recover with the following steps:

1. Press the RESET TERMINAL key. On an HP2645A, HP2647A/F, HP2648A, or HP2640B; press the RESET TERMINAL key rapidly, twice. On the HP2640B, also press the BLOCK MODE key to return to character mode. On an HP262X terminal, press the CTRL (control) and SHIFT keys while pressing RESET TERMINAL.
2. Press **RETURN** to get the MPE colon prompt (:).
3. Press the ESC (escape) key at the left of the top row of the character set group. Then press the colon (:) key to display what you are typing (echo) on the terminal screen.
4. Type the word resume and press the **RETURN** key. The message "READ PENDING" will be displayed on your screen. On an HP2640B terminal, press the BLOCK MODE key.
5. Press the f6 function key to redisplay (REFRESH) the current menu. On an HP262X terminal, press the USER KEYS key before pressing REFRESH.

These procedures can also be used to recover from a power failure or from your terminal being turned off while you are in session.

The Character Set Keys

This group includes the standard typewriter alphanumeric keys which you use to enter data values, names, and specifications. In addition, you may use the **RETURN**, **TAB**, SHIFT, and BACKSPACE keys to control the cursor. The

cursor is the blinking underscore on the terminal screen which indicates the position where a character you type on the keyboard will appear on the screen.

RETURN The **RETURN** key moves the cursor to the beginning of the next line available in the current unprotected field. It will move to the next field only if it is at the end of the current field.

The **RETURN** key does not issue data to the computer. You would use the **ENTER** key to issue data to the computer.

TAB The **TAB** key moves the cursor from its current position to the beginning of the next unprotected field. With each subsequent **TAB** the cursor moves to the beginning of each field, from left to right, on each line where there is more than one unprotected field. A **TAB** from the last field on a menu will reposition the cursor at the beginning of the first unprotected field at the top of the menu. In addition, holding down the CNTL (control) key while pressing the **TAB** key will move the cursor backwards through the previous field positions. For the HP262X series of terminals, the SHIFT key is used with the **TAB** key to return to the beginning of the previous fields. The reverse TAB procedure is not available on the HP2640B.

SHIFT The SHIFT key performs the standard typewriter upper case shift. In addition, on 262X terminals the SHIFT key acts as a control key by reversing the forward motion of the **TAB** key cursor control function, where **SHIFT/TAB** moves the cursor to the previous unprotected field.

BACKSPACE The BACKSPACE key moves the cursor back one character.

The Display Control Keys

This group of keys can be used to move the cursor and to clear data from the unprotected fields. When you use the cursor positioning keys, you can inadvertently type outside of the limits of an unprotected field. When this happens, the program will automatically tab to the next unprotected field and continue the typing display. The keys in this group are defined as follows:

CLEAR Clears all data in all unprotected fields from the current position of the cursor to the last field of the menu.

**CNTL/
CLEAR
DISPLAY** Clears the data from all lines within the unprotected field at the cursor position. Data in other unprotected fields is not cleared.

**CURSOR
LEFT** Moves the cursor back one character, or continuously without changing any existing characters.

USING IFS/3000 INTERACTIVELY

CURSOR RIGHT Moves the forward one character, or continuously without changing any existing characters. (The SPACE BAR would clear existing characters.)

CURSOR UP/DOWN Moves the cursor up or down one line, or continuously without changing any existing characters.

CURSOR HOME Moves the cursor to the beginning of the first unprotected field on the menu.

CNTL/CURSOR HOME Positions the cursor after the last unprotected field on the menu. On an HP262X terminal, use the SHIFT and CURSOR HOME keys.

The Field Edit Keys

The keys in this group enable you to insert or delete characters at the cursor position in an unprotected field as follows:

INSERT CHAR After you press this key, any character you type will be inserted in front of the character at the cursor position. If the field is full, inserting characters will push the rightmost characters beyond the field limits and they will be lost. You turn the INSERT CHAR function off by pressing the INSERT CHAR key a second time to return to normal mode, where any character typed replaces the character at the cursor position. The INSERT CHAR key should always be turned off immediately after use to avoid accidentally inserting data.

DELETE CHAR Each time you press this key, you delete the character at the cursor position. All subsequent characters in the field move one space to the left, creating blanks in the rightmost positions of the field. You do not need to turn the DELETE CHAR key off as it operates only when pressed.

Using IFS/3000 Menus

When you log on and :RUN IFS.PUB.SYS you enter the program at the Environment File Menu. There are ten menus in the IFS/3000 program as follows:

- Environment File Menu
- Main Menu
- Initialize Menu
- Physical Page Menu
- Multi-Copy Forms Menu
- Logical Page Menu
- Logical Page Forms Menu
- Vertical Format Control Menu
- Character Font Menu
- Copy Menu

Once you supply a valid environment file name on the **Environment File Menu**, you will automatically be presented with the **Main Menu**.

The **Main Menu** lists all operations you can perform when creating or modifying an environment file. Subsequent menus in the program will only be presented if they support the task you select on the **Main Menu**.

The **Initialize Menu** enables you to use an existing environment's specifications as the starting point for creating a new environment.

The **Physical Page Menu** enables you to define the physical page size. It also provides the menu path to the **Multi-Copy Forms Menu** where you specify forms to be used to simulate carbon copy output or multiple-copies of the physical page.

The **Logical Page Menu** enables you to specify alternate page layouts for the physical page. It also provides the menu path to the **Logical Page Forms Menu**, where you specify a form residing in a forms file to be included in the logical page layout. Or, you may proceed to the **Vertical Format Control Menu** to specify vertical line spacing unique to the logical page.

The **Character Font Menu** enables you to specify fonts residing in a character font file which are to be included in the environment.

The **Copy Menu** enables you to copy all or part of an existing environment into the environment you are creating or modifying.

Any data you enter on any menu in the program is only issued to the computer when you press **ENTER**. If you use any function key before you use the **ENTER** key, the data just typed will be lost. When you press **ENTER**, the status of the operation on the field data will be acknowledged by a message in the banner, or a bell.

The **Environment File Menu** enables you to exit the program with the **F8** function key. Once you proceed to the **Main Menu**, you must return to the **Main Menu** to specify that you wish to exit the program. If you have not compiled the current environment file, a warning message will be issued and you will be asked to repeat your exit instruction in order to exit the program.

On all menus except the **Environment File Menu**, you may return to the **Main Menu** using the **F7** function key. Figure 3-3 provides a Menu Map of the IFS/3000 program.

The IFS/3000 Menu Map

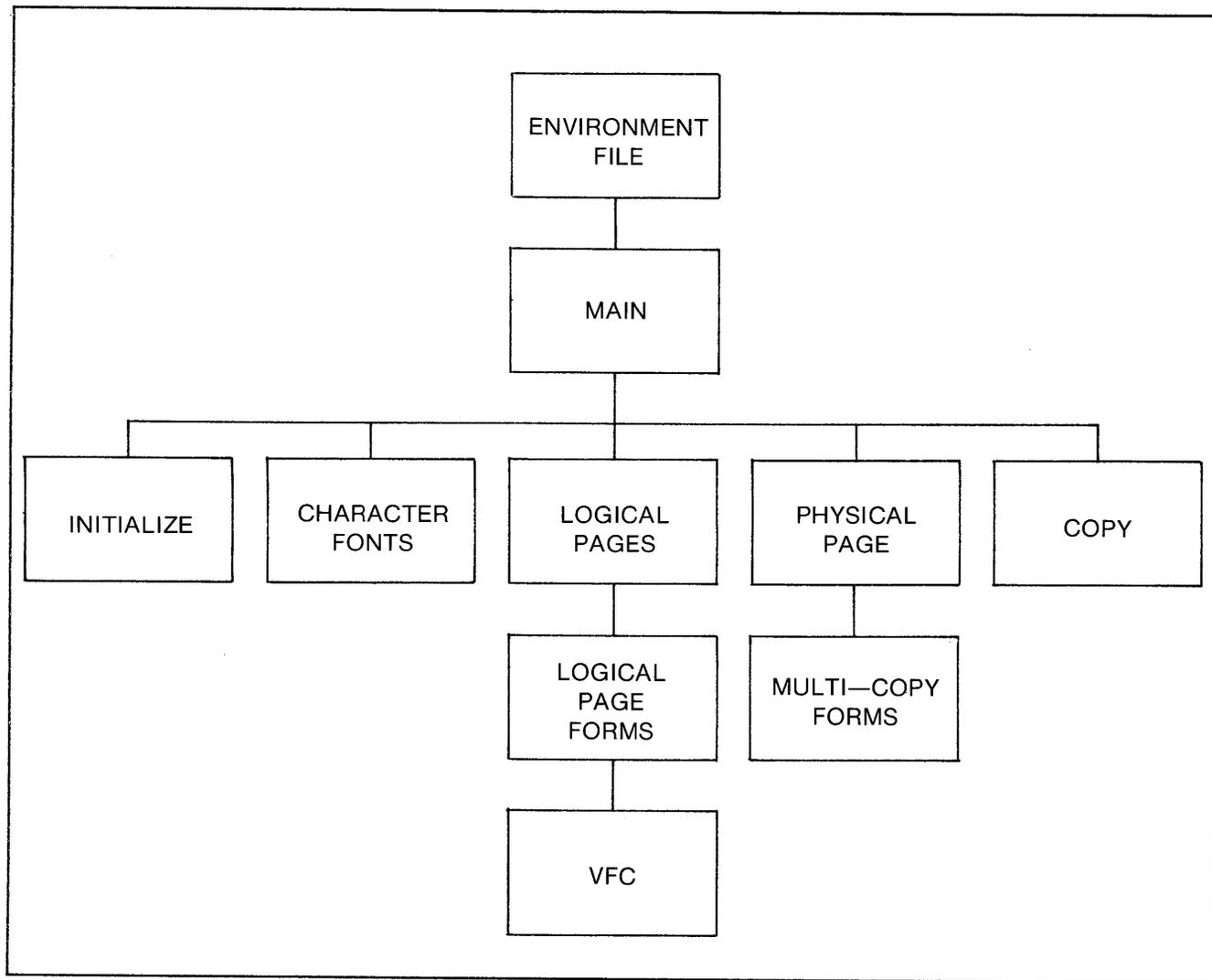


Figure 3-3. The IFS/3000 Menu Map

Log-On and Run IFS/3000

Before you can run IFS/3000 you have to complete several basic steps. If you are using a 264X series terminal, you should make sure that the REMOTE key, on the second row from the top, is depressed.

The next step is to inform the system that you wish to start working. To do this, you must identify yourself with a user name and account name.

First, press the RETURN key on your keyboard until a colon (:) appears on the screen.

Type the following at the colon, substituting your user and account names:

```

      Space      Period
      ↓          ↓
:HELLO username.accountname

```

Press the RETURN key. After a brief pause, the system acknowledges your hello with a message.

```

HP3000 / MPE IV C.D0.20.   SUN, JAN 30, 1983,  2:51 PM
:
```

When a colon (:) appears following the message, the system is ready to accept your instruction to run the IFS/3000 program. At the colon, type:

```

:RUN IFS.PUB.SYS
```

Press the RETURN key. The first message you will see will identify the version of IFS/3000 and VPLUS/3000 available on your system displayed as follows:

```

IFS/3000      (A.02.01) HP36580 (c) COPYRIGHT Hewlett-Packard Co. 1983
V/3000       (B.03.20)
```

After a brief pause, the Environment File Menu will be displayed.

The Environment File Menu

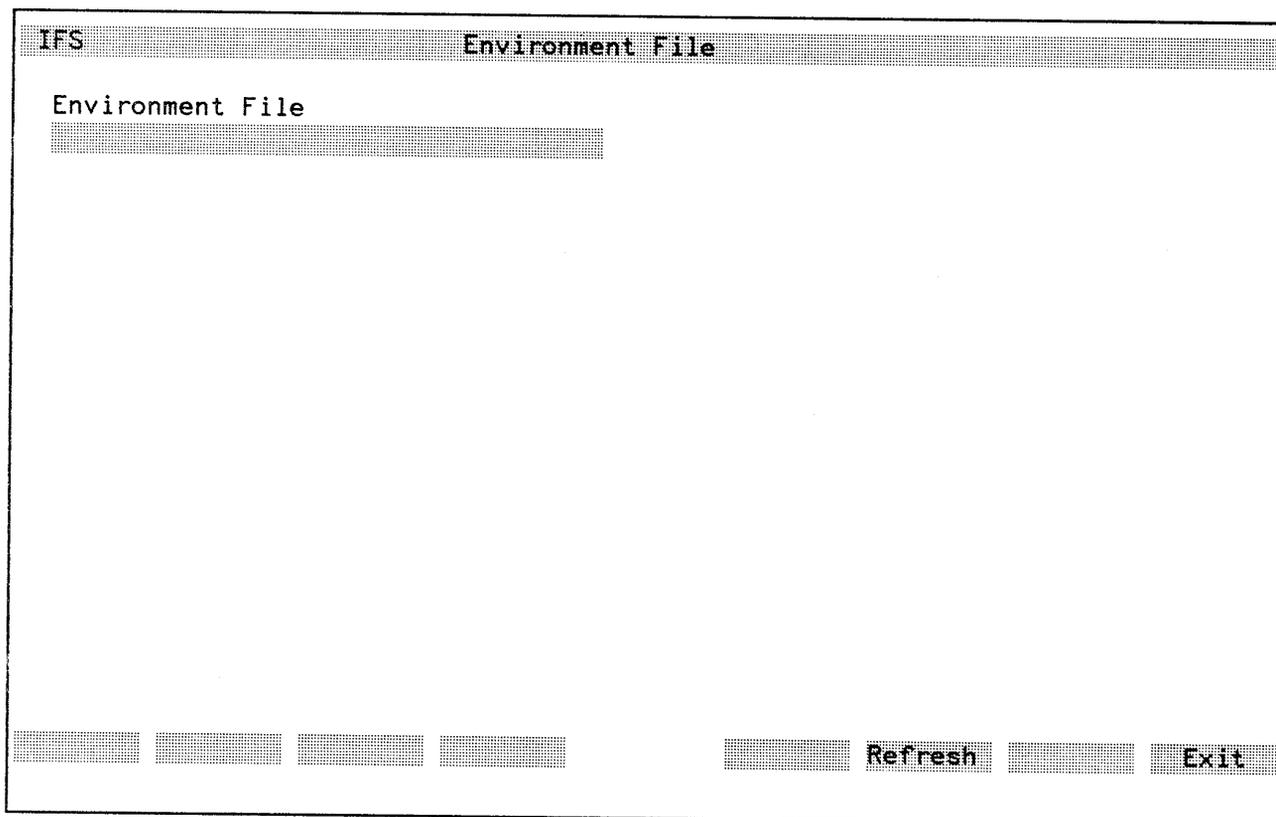


Figure 3-4. Environment File Menu

The Environment File Menu is the first menu in the IFS/3000 program.

You use this menu to access an existing environment file, or create a new environment file. When you specify a valid filename and press **ENTER**, the Main Menu will be presented where you indicate the operation you wish to perform. You can return to this menu by pressing **f a** on the Main Menu.

Function Keys

The function keys used on this menu are:

ENTER

Once you have typed the environment file name you wish to access or create, press the **ENTER** key to issue the filename

to the computer. IFS/3000 will issue status messages if the file does not exist as specified, or the Main Menu will be presented.

Refresh Press to redisplay the terminal screen.

Exit Press to EXIT the IFS/3000 program.

Field Discussion

The Environment File Menu provides a single data entry field to specify a filename. You enter the environment filename using standard MPE File System naming conventions, where the filename may have up to 8 characters, and can be any letter followed by any letter or number. If the file resides in a group other than the sign-on group, you must specify a groupname and accountname. You may also specify a lockword if this form of security applies to the file.

If you are creating a new environment file, it will reside in the sign-on group and account.

If you used all of these file designation conventions the field data entry would appear as follows:

Environment File

The IFS/3000 program checks to see that the filename specified exists, and that it is an environment file. If the file does not exist as

specified, a status message will be issued to the banner window indicating that:

- The filename does not meet MPE filename conventions as specified.
- The file does not reside in the group and account specified, where the group and account is different from the group where the program is being run.
- You cannot access the file in the group and account specified due to security violations.
- The file specified is not an environment file.

OR, if the filename does not exist in the current group, you will be asked if you wish to create a new environment file.

- If you do not want to create a new environment file, you can change the environment filename and enter it again. This process can be continued until you enter a filename which meets the IFS/3000 and MPE requirements and is accessible as specified.
- If you want to create a new environment file with the name specified, press as requested.
- OR, press to EXIT the IFS/3000 program.

Once you supply a valid filename and press , the Main Menu will be presented.

the option to proceed to a subsequent menu path from the current menu.

You may return to the Environment File Menu using the **Done** function key, or exit the program by entering E. However, once you create or modify any environment specifications on subsequent menus, you must complete the operation by compiling the new or modified specifications for the environment file. A warning message will be issued before you are allowed to leave the Main Menu if the environment file is not in the compiled state. Your environment file must be compiled to be successfully associated with a data file and be accepted by the Laser Printer.

Function Keys

The function keys used with this menu are:

ENTER

Once you have typed your operation selection and any associated data in the menu fields, press the **ENTER** key to issue your instructions to the computer. IFS/3000 will issue status messages to the menu banner if it has difficulty performing the operation; OR, a menu which supports the specified operation will be presented.

Refresh

Press **F6** to redisplay the terminal screen with any data entered using the **ENTER** key. It is useful when the terminal is inadvertently turned off or reset.

Done

Press **F8** when you are done with the current environment file and wish to return to the Environment File Menu and specify another environment file.

Field Discussion

There are five data entry fields on the Main Menu.

The Selection Field and the Device Field are required fields. A warning message will be issued to the message banner if you specify a selection code not listed, leave the selection field blank, or make a selection without specifying the target output device. If you are reviewing or modifying an existing environment file, the device will default to the device specified in the existing environment.

1. The Selection Field

Selection

The first field on the Main Menu is the Selection field. The alternate selections provide the menu path to all operations which can be performed on an environment file. You enter a one or two letter code for the operation you wish to perform. If you do not enter a selection, or you specify an erroneous one, and press **ENTER**, an error message is displayed in the menu banner. To proceed, you either enter a valid selection, also specify a device if an existing file default doesn't apply, or press

THE MAIN MENU

D Done to return to the Environment File Menu.

Your alternate selections for this field are defined as follows:

I Initialize environment

This selection along with a new environment device number will display the Initialize Menu. You would not use this option if you specified an existing environment file on the Environment File Menu. Trying to initiate another set of basic specifications using another environment file on the Initialize Menu while in an existing file will destroy the current specifications.

You could use this option to initialize a new environment by starting with the specifications in the file you enter on the Initialize Menu.

Refer to the Initialize Menu discussion which follows in this section.

C Character Fonts

This selection along with an existing default or new environment device number will display the Character Font Menu. You use this menu to review or modify the character fonts included in an existing environment. OR, you use this menu to identify a character font to be added to the environment. In order for a character font to be available for a print job, it must be included in the environment in the size and rotation desired.

You can enter up to 32 different sets of character font specifications. You can go directly to a character font by also entering its numerical identification number in the optional Font/Page number field or its name in the Character Font Name field on this menu. If you do not specify a font ID-number or no fonts are defined in the environment, the program starts with lowest character font ID-number usually 0 (zero).

Refer to the Character Font Menu discussion which follows in this section.

L Logical pages, forms and VFCs

This selection along with an existing default or new environment device number will display the Logical Page Menu. You use this menu to review or modify the logical pages included in an existing environment. OR, you use this menu to define a logical page to be added to a new or existing environment. You would also use this menu path to access the Logical Page Forms Menu where you associate up to two forms in a forms file with a logical page. And, proceed to the Vertical Format Control Menu to specify the vertical line spacing unique to a logical page.

You can enter up to 32 different sets of logical page specifications. You can go directly to a logical page by also entering its numerical identification number in the optional Font/Page number field on this menu. If you do not specify a logical page ID-number or no logical pages are defined in

the environment, the program starts with the lowest logical page, usually 0 (zero).

Refer to Logical Page, Logical Page Forms, and Vertical Format Control Menu discussions which follow in this section.

P Physical page and multi-copy forms

This selection along with an existing default or new environment device number will display the Physical Page Menu. When creating a new environment from scratch, you would use this menu to specify the paper, and therefore, physical page size for the environment. You can also specify the **ID**-number of the initial primary and secondary character fonts for the environment. And, you can specify if multiple copies of the physical page are to be printed when the print job commences.

This menu provides the menu path to the Multi-Copy Forms Menu where you specify pairs of forms in a forms file to be used to simulate carbon copy output.

Refer to the Physical Page and Multi-Copy Forms Menu discussions which follow in this section.

CC Copy from another environment

This selection along with an existing default or new environment device number will display the Copy Menu. You use this menu to copy all or selected parts of another environment file into the current environment file. This alternate differs

from the Initialize selection in that you may select the parts you wish to use.

Refer to the Copy Menu discussion which follows in this section.

DC Delete character font

This selection along with an existing default device number AND the character font **ID**-number entered in the Font\Page number field, or optional character font name entered in the Character font name field, perform the operation directly from the Main Menu when you press **(ENTER)**. The character font specifications will be deleted from the environment and are no longer available through the Character Font Menu. The compiled parts of the deleted character font will be deleted when you recompile the environment. A bell signal will indicate the operation is complete and the Font/Page number, and **characterfontname** field entries will be returned to blanks. A status message will be issued if the font specified is invalid or cannot be found.

The recompile operation must be selected as a separate operation as discussed below.

DL Delete logical page

This selection along with an existing default device number AND the logical page **ID**-number entered in the Font\Page number field perform the operation directly from the Main Menu when you press **(ENTER)**. The logical page specifications will

THE MAIN MENU

be deleted from the environment and are no longer available through the Logical Page Menu. The compiled parts of the deleted logical page will be deleted when you recompile the environment. A bell signal will indicate the operation is complete and the **DL** and **ID** field entries will be returned to blanks. A status message will be issued if the logical page specified is invalid or cannot be found.

The recompile operation must be selected as a separate operation as discussed below.

X Compile environment

This selection along with an existing default or new environment device number perform the compile operation directly from the Main Menu when you press **ENTER**. You would use this operation when you are satisfied with the specifications defined for an environment and you wish to prepare it for use by the Laser Printer. You would also use this operation to recompile an environment you have modified. You have the option to recompile all components of the modified environment by changing the default **N** (=no) to **Y** (=yes) in the **Recompile everything?** inquiry field. If you do not want to recompile everything, press **ENTER** without changing the default.

If problems are encountered during the compile operation, warning or error messages are displayed in the banner window. Warning messages indicate the specified part was compiled or retained as previously compiled, but a condition exists in the

compiled format which could affect its use. Since warning messages do not stop the compile, each subsequent warning replaces the preceding one. You must note each message as it occurs, or recompile the environment again and watch the warnings as they are displayed.

An error stops the compile operation and a corresponding error message is displayed. The third part of this section lists error messages, their meaning and the suggested action.

You can return to the part in error to make modifications by selecting one of the alternate operations which will enable you to proceed to the menu containing the specifications. If the error is due to a component file outside of IFS/3000, you may exit the program to modify the file, and a warning message will be issued that the current file needs to be compiled.

When the environment has compiled successfully, the message:

Successful compilation

is displayed in the banner window. The amount of space required for the environment **COMAREA** which is used by the Programmatic Interface intrinsics is displayed as follows:

Comarea space required
number words

E Exit

This selection indicates that you wish to EXIT the program. If the environment file was modified and not recompiled or is new and needs to be compiled, the program will clear the **E** selection and issue a warning message and an instruction. If you wish to exit without compiling the environment, reselect **E** and press **(ENTER)**. You will be returned to the MPE operating system as follows:

END OF PROGRAM

:

The version number of IFS/3000 is also displayed above End of Program.

NOTE

The uncompiled environment remains unavailable for use with a data file or the Laser Printer. Any unchanged parts previously compiled remain compiled. You can run IFS/3000 at any time to complete the compile or recompile operation.

2. The Device Field

Device



The Device Field is a required field. When creating a new environment, you must specify the target output device in this field. IFS/3000 uses the device number when performing the selected

operation to determine if the specifications for the physical page, all logical pages, and all character fonts are consistent with the target device limitations.

Once the device field is specified, it remains device specific. When you specify an existing environment on the Environment File Menu, the field will default to the device associated with the existing file. The device model number will appear in this field and you replace it only if you are converting the entire environment to an alternate device.

Environments are device specific. All components within an environment are compiled using the dot pattern and density specific to the Laser Printer Hardware specified as the device. Most of the components of the environment, with the exception of some forms, are also device dependent when created. Hewlett-Packard supplied fonts and environments are also device specific.

If you plan to convert an existing environment, it is suggested that you consider the availability of the device dependent components for the new device. Refer to Sections 1 and 2 in this Reference Guide to review the characteristics of device dependent components. Appendix H describes the IFS/3000 Merge Utility which you can use to store device dependent fonts in a single character font file. Appendices C and D lists Hewlett-Packard supplied environments and fonts respectively.

THE MAIN MENU

The format for data entry in this field is as follows:

2680A The HP 2680A Laser printer is a tractor-feed system printer which prints 180 dots per inch.

2688A The HP 2688A Laser Printer is a single sheet office printer which prints 300 dots per inch.

default When you have specified an existing environment on the Environment File Menu, the device associated with the environment becomes the default for this field. This field cannot be blank when a new environment is specified.

3. The Font/Page Number Field

Font/Page number

You access the specifications for a specific font or logical page number by typing its **ID**-number in this field. The **ID**-number appears on the Logical Page or Character font menu if you need to review existing font or logical page numbers.

When you are deleting a logical page by specifying **DL** in the Selection field, its **ID**-number is required in this field.

When you are deleting a character font by specifying **DC** in the Selection field, its **ID**-number is required in this field, or its name is required in the Character Font Name field.

When you are using the delete operation and specify the **ID**-number in this field and press **ENTER**, no other menus are required or presented. The program issues a status message or acknowledges a successful deletion by returning the filed to blanks and sounding a bell.

This field is optional if you entered **C** or **L** in the Selection field. You can use this field to select a specific character font or logical page that you want to add, modify, or review. IFS/3000 displays either the Character Font or Logical Page Menu with existing specifications, or as a blank menu to enter new specifications.

4. The Character Font Name Field

Character font name

.....

You use the Character Font Name field only if you entered **C** or **DC** in the Selection field. It is not necessary to use both the character font **ID**-number and the character font name. Either of these is sufficient to select the specific character font you want to work with or delete. However, one or the other is required when you are using the delete operation.

You can use this field to select a specific character font that you want to add, modify, or review. IFS/3000 displays the Character Font Menu with existing specifications, or as a blank menu to enter new specifications.

5. The Recompile Everything? Field

N Recompile everything?

This field contains the default value **N** (=no). If you use the default value, the program compiles only those specifications which are new or have been modified since the last compile. The logical page and physical page are exceptions as they are always recompiled.

If a form associated with a logical page was modified using **IDSFORM**, **IFS/3000** detects the change to the form even though the logical page specifications were not changed on the Logical Page Menu. Also, if a character font has been modified using **IDSCHAR**, **IFS/3000** detects the change even though the character font specifications were not changed on the Character Font Menu.

Since **IFS/3000** detects these changes, it will automatically recompile the associated specifications.

However, if you modify a character font that is referenced on a form created using **IDSFORM**, **IFS/3000** WILL NOT detect the change and thus WILL NOT recompile the form. To recompile the form, you must either change this field to **Y** to recompile everything, or access the form using **IDSFORM** and modify the field containing the character font in question. Once this field has been accessed, **IFS/3000** recognizes a change has been made and will recompile it.

For more information on forms design, refer to the *IDSFORM Reference Guide (part number 36581-90002)*.

To recompile all specifications in the environment, change the default in this field to **Y** and press **(ENTER)**. Status and Error messages will be issued to the menu banner. Large environments may take considerable time to recompile if this option is selected.

Function Keys

The function keys used with this menu are:

ENTER

Once you have typed your environment selection in a menu field, press the **ENTER** key to issue your instruction to the computer. IFS/3000 will issue status messages to the menu banner if it has difficulty accessing or copying the environment. OR, the Main Menu will be displayed for you to continue with additions or modifications to the copied specifications.

Refresh

Press **F6** to redisplay the terminal screen with any data entered using the **ENTER** key. It is useful when the terminal is inadvertently turned off or reset.

Main

Press **F7** to return to the Main Menu before you have entered an environment selection.

Cancel

Press **F8** to cancel the environment selection you have typed on the menu and have not entered.

Field Discussion

There are three data entry fields on the Initialize Menu. You must specify either the HP Defined Environment or a name in the Another Environment field in order for IFS/3000 to locate, access, and copy the specifications in an environment file. The fields are defined as follows:

1. The HP Defined Environment Field

HP defined environment

Hewlett-Packard supplies a number of environment files with IFS/3000. For ease of use, the most commonly used HP-defined environments are listed as alternate selections adjacent to this field. This enables you to use one of these environments as an initial environment by typing the appropriate letter in the data entry field.

If you also select a reduction in the Reduction field, IFS/3000 rotates and scales the logical page in these environments so that the desired number of logical pages fit on the physical page. Any character fonts included in these environments are also rotated so that printing is in the same orientation as the logical page. Refer to the Reduction field discussion which follows in this section.

The environment alternates available in this field are defined as follows:

R Regular line printer

This environment includes a logical page with a width that is greater than the height. This page format will be referred to as a Landscape orientation throughout IFS/3000 procedures.

This environment simulates a line printer environment except that the physical page is 11 by 8 1/2 inches instead of the standard line printer page size of 14 3/8 by 11 inches. As with a standard line printer, the logical page has 66 print lines and 132 character positions on each line. An 8 point simulated line printer font is also included.

You can duplicate a line printer application requiring a preprinted form by adding a form to the logical page on the Logical Page Menu.

T Typewriter, 10 pitch

This environment includes a logical page with a height that is greater than the width. This page format will be referred to as a Portrait orientation throughout IFS/3000 procedures.

The physical page is 8 1/2 by 11 inches. There are six lines per inch, or 66 lines per physical page. There are 75 characters on each line, leaving a 1/2 inch margin on both the left and right sides of the logical page. The environment includes a 10 pitch font

which simulates a typewriter with Pica type (10 characters per inch).

You can produce the effect of varying typing elements on a typewriter by adding other character fonts to this environment using the Character Font Menu.

E Typewriter, 12 pitch

This environment has the same physical and logical page specifications as the above environment.

This alternate prints 12 characters per inch, or 90 characters per line. It includes a 12 pitch font which simulates an Elite typewriter font. A 12 pitch italics font is also included as the secondary font.

D Document

This environment has the same physical and logical page specifications as the **P** Portrait environment.

The document environment includes a proportionally spaced Roman character font in 12 point type as the primary font, 12 point Bold as the secondary font, and 12 point italics as an alternate font.

L Landscape

This environment includes a logical page with a width that is greater than the height. This page format will be referred to as a Landscape orientation throughout IFS/3000 procedures.

The physical page is 8 1/2 by 11 inches. This environment contains no character fonts. You would use this alternate to initialize the specifications for a Landscape environment. You would add logical pages and character fonts using the Logical Page and Character Font Menus.

P Portrait

This environment includes a logical page with a height that is greater than the width. This page format will be referred to as a Portrait orientation throughout IFS/3000 procedures.

The physical page is 8 1/2 by 11 inches. This environment contains no character fonts. You would use this alternate to initialize the specifications for a Portrait environment. You would add logical pages and character fonts using the Logical Page and Character Font Menus.

2. The Reduction Field

This field is device dependent and can only be used if you have also specified the **2680A** Laser Printer in the Device field on the Main Menu.

This field is also an option which can ONLY be used with the environments listed as alternates in the HP Defined Environment field.

When you choose a reduction, IFS/3000 uses an environment which includes a form in the logical page definition. This form consists of a single line

used as a border around the logical page. You can remove this border by modifying the environment file. To do this, use the Logical Page Forms Menu to remove the names of the border form and form file.

Refer to Figures 3-7 and 3-8 which illustrate the effect of specifying one of the following reduction options:

2 2 to 1

Choosing a 2 to 1 reduction enables you to print twice as much in the same amount of space by rotating and scaling two logical pages to a single physical page. See Figure 3-7.

4 4 to 1

Choosing a 4 to 1 reduction enables you to print four times as much in the same amount of space by rotating and scaling four logical pages to a single physical page. See Figure 3-8.

3. The Another Environment Field

Another environment

This field allows you to supply the name of any existing environment file that you want to use as an initial environment.

You enter the environment filename using standard MPE File System naming conventions where the filename may have up to 8 characters,

THE INITIALIZE MENU

and can be any letter followed by any letter or number. If the file resides in a group other than the sign-on group, you must specify a groupname and accountname. You may also specify a lockword if this form of security applies to the file.

If you used all of these file designation conventions the field data entry would appear as follows:

Another environment

`filename/lockword.groupname.accountname`

When you press **ENTER**, IFS/3000 checks to see that the filename specified exists, and that it is an environment file. If the file does not exist as specified, a status message will be issued to the banner window indicating that:

- The filename does not meet MPE filename conventions as specified.

- The file does not reside in the group and account specified, where the group and account is different from the group where the program is being run.
- You cannot access the file in the group and account specified due to security violations.
- The file specified is not an environment file.

OR, the environment specified will be copied and the Main Menu will be displayed.

If you use this field, you may not enter a reduction selection. To achieve reduction with a custom environment, you must include the appropriate logical page and character font definitions using the Logical Page and Character Font Menus.

THE INITIALIZE MENU

PAGE 0005/COBTEXT CALENDAR				PAGE 0006/COBTEXT CALENDAR			
00147	015900			00184	019700		
00148	016000 01	FONT-NUMBERS.		00185	019800 01	DAYS-TABLE.	
00149	016100	05 MONTH-YR-FONT	PIC S9(4) COMP VALUE 0.	00186	019900	05 SUNDAY	PIC X(10) VALUE " SUNDAY".
00150	016200	05 CAL-DAYS-FONT	PIC S9(4) COMP VALUE 1.	00187	020000	05 MONDAY	PIC X(10) VALUE " MONDAY".
00151	016300	05 WEEK-DAY-FONT	PIC S9(4) COMP VALUE 2.	00188	020100	05 TUESDAY	PIC X(10) VALUE " TUESDAY".
00152	016330	05 DOT-LINE-FONT	PIC S9(4) COMP VALUE 2.	00189	020200	05 WEDNESDAY	PIC X(10) VALUE " WEDNESDAY".
00153	016400	05 JUL-DAYS-FONT	PIC S9(4) COMP VALUE 3.	00190	020300	05 THURSDAY	PIC X(10) VALUE " THURSDAY".
00154	016500	05 DEFAULT-FONT	PIC S9(4) COMP VALUE 4.	00191	020400	05 FRIDAY	PIC X(10) VALUE " FRIDAY".
00155	016700			00192	020500	05 SATURDAY	PIC X(10) VALUE " SATURDAY".
00156	016800 01	DOTTED-LINE.		00193	020600 01	WEEK-DAYS REDEFINES DAYS-TABLE.	
00157	016900	05 FILLER	PIC X(18) VALUE ALL "...".	00194	020700	05 DAYS-TABLE OCCURS 7 TIMES.	
00158	017000			00195	020800	10 WEEK-DAY	PIC X(10).
00159	017100 01	MONTH-FIELD-DATA.		00196	020900		
00160	017200	05 CALENDAR-LOGO.		00197	021000 01	FIELD-DAYS.	
00161	017300	10 CALENDARLOGO	PIC X VALUE SPACES.	00198	021100	05 SUNDAY	PIC X(6) VALUE "DAYS1".
00162	017400	10 CAL-CHARSET	PIC S9(4) COMP.	00199	021200	05 MONDAY	PIC X(6) VALUE "DAYS2".
00163	017500	05 CREATOR-NAME.		00200	021300	05 TUESDAY	PIC X(6) VALUE "DAYS3".
00164	017600	10 CREATORNAME	PIC X(30) VALUE SPACES.	00201	021400	05 WEDNESDAY	PIC X(6) VALUE "DAYS4".
00165	017700	10 CREATOR-FONT	PIC S9(4) COMP.	00202	021500	05 THURSDAY	PIC X(6) VALUE "DAYS5".
00166	017800	05 CAL-HEADING.		00203	021600	05 FRIDAY	PIC X(6) VALUE "DAYS6".
00167	017900	10 CALHEADING	PIC X(52) VALUE SPACES.	00204	021700	05 SATURDAY	PIC X(6) VALUE "DAYS7".
00168	018000	10 HEADER-FONT	PIC S9(4) COMP.	00205	021800 01	FIELD-TABLE REDEFINES FIELD-DAYS.	
00169	018100	05 CAL-NOTE.		00206	021900	05 FIELD-TAB OCCURS 7 TIMES.	
00170	018200	10 CALNOTE	PIC X(52) VALUE SPACES.	00207	022000	10 DAY-FLD-NAME	PIC X(6).
00171	018300	10 CALNOTE-FONT	PIC S9(4) COMP.	00208	022100		
00172	018400	05 SALES-NAME.		00209	022200 01	FORMATED-DATE.	
00173	018500	10 SALESNAME	PIC X(52) VALUE SPACES.	00210	022300	05 YY	PIC XX.
00174	018600	10 SALE-NAME-FONT	PIC S9(4) COMP.	00211	022400	05 MM	PIC XX.
00175	018700	05 SALES-ADDRESS.		00212	022500	05 DD	PIC XX.
00176	018800	10 SALESADDRESS	PIC X(52) VALUE SPACES.	00213	022600		
00177	018900	10 SALES-ADDR-FONT	PIC S9(4) COMP.	00214	022700 01	CALENDAR-YEAR.	
00178	019000	05 SALES-CITY.		00215	022800	05 FIRST-TWO-NO	PIC XX VALUE "19".
00179	019100	10 SALESCITY	PIC X(52) VALUE SPACES.	00216	022900	05 CALENDAR-YR	PIC XX.
00180	019200	10 SALES-CITY-FONT	PIC S9(4) COMP.	00217	023000		
00181	019300	05 SALES-INFO.		00218	023100 01	FIRST-LAST-DAYS.	
00182	019400	10 SALESINFO	PIC X(52) VALUE SPACES.	00219	023200	05 FIRST-CAL-DAY	PIC X(6) VALUE SPACES.
00183	019500	10 SALES-INFO-FONT	PIC S9(4) COMP.	00220	023300	05 LAST-CAL-DAY	PIC X(6) VALUE SPACES.
				00221	023400	05 BEGIN-WK-DAY	PIC 9 COMP VALUE 0.
				00222	023500	05 CUR-PRT-CAL-DAY	PIC X(6).
				00223	023600	05 CUR-CAL-DATE REDEFINES CUR-PRT-CAL-DAY.	
				00224	023700	10 YY-DSPL	PIC 99.
				00225	023800	10 MM-DSPL	PIC 99.
				00226	023900	10 DD-DSPL	PIC 99.
				00227	024000	10 DD-DSPL-2 REDEFINES DD-DSPL.	
				00228	024100	15 DD-FIRST	PIC 9.
				00229	024200	15 DD-SECOND	PIC 9.
				00230	024300	05 CAL-CNTR	
				00231	024400	10 YY-COMP	PIC S9(4) COMP.
				00232	024500	10 MM-COMP	PIC S9(4) COMP.
				00233	024600	10 DD-COMP	PIC S9(4) COMP.
				00234	024700		

Figure 3-7. 2 to 1 Reduction With an HP-defined Environment

Function Keys

The function keys used with this menu are:

ENTER

Once you have typed your physical page specifications in the menu fields, press the **ENTER** key to issue this data to the computer. IFS/3000 will issue status messages to the menu banner if inconsistencies are found between the physical limitations of the device specified on the Main Menu and the physical page specifications you have requested. OR, the program will issue a bell to indicate the field data is acceptable. The menu will remain on the screen for your review; OR, the Multi-Copy Forms Menu will be displayed, if specified. If you do not make any changes, you can press **ENTER** again, or press the **Main** function key to return to the Main Menu.

Refresh

Press **F6** to redisplay the terminal screen with any data entered using the **ENTER** key. It is useful when you have used the **Cancel** key.

Main

Press **F7** to return to the Main Menu. Note that any field data you have typed but have not entered will be lost.

Cancel

Press **F8** to delete any data you have just typed, but have decided not to enter.

Field Discussion

There are six groups of data entry fields on the Physical Page Menu.

You use these fields to define the dimensions of the physical page, the primary and secondary character font defaults, and whether or not you want the Laser Printer to print multiple copies of the physical page as a single print job.

In addition, you can also specify that you wish to go to the Multi-Copy Forms Menu where you can specify up to eight pairs of forms to be associated with the physical page. You would use this menu path if your environment was to be used to simulate carbon copy production.

The field data and its effect on the environment is defined as follows:

1. The Number of Copies Field

1 Number of copies

The default value for this field is **1** (=1). You may request up to 32,767 copies of the physical page by entering the number of copies desired in this field.

Copies of a physical page are generated by the Laser Printer by issuing the specified number of physical page ejects for each new page instruction associated with the data file. In a multi-page print job, each page will print the number of times specified in this field, and you would have to collate the copies manually.

If you want collated copies of a multi-page print job, you must specify the number of copies within the print file equation (see Appendix A). Copies of multi-page documents generated in this manner are reissued by the host computer once for each copy requested. The environment file is reloaded in the Laser Printer on each issue from the host computer.

2. The Multi-Copy Forms? Field

Multi-copy forms? Y/N

This field defaults to (=no). It provides the menu path to the Multi-Copy Forms Menu if you change to (=yes). You would use the Multi-Copy Forms Menu to specify up to eight pairs of forms which are to be used to simulate carbon copy output. Refer to the Multi-Copy Forms Menu discussion which follows in this section.

3. The Physical Page Size Fields

Width
 Length
 Units

There are three data entry fields in this group. You use these fields to define the dimensions of the

physical page. The first field defines the page width. The second defines the page length. The third field defines the units of measure to be used to calculate the width and length.

These fields default to the standard 8 1/2 by 11 paper size dimensions for the device specified in the Device field on the Main Menu. You can change these defaults by changing the dimensions to the alternate paper size for your printer model.

The physical page refers to the actual paper surface that passes through the printer and is exposed to the printer's electrophotographic transfer drum. Its dimensions are determined by the Laser Printer model installed on your system. Refer to the Operator's Handbook for your printer model to determine the paper and, therefore, the physical page sizes available.

The physical page size establishes the maximum dimensions available for printing all other components of your environment. Each printer model also has unique border areas along the edges of the physical page which are outside of the transfer area of the printer transfer drum. Any logical page specifications you define on the Logical Page Menu which extend beyond the physical page size, or fall within the unprintable border areas will result in error or warning messages specific to the limitations exceeded.

IFS/3000 uses the physical page dimensions you specify in these fields when compiling your environment. This does not ensure that the paper in the output device has these dimensions. If your system changes paper sizes frequently, you can tell the operator the size of paper to load into the

THE PHYSICAL PAGE MENU

printer using the MPE :FILE command or the FOPEN intrinsic. Refer to Appendix A.

You define the physical page dimensions using these fields as follows:



Width

You define the page width in this field as a whole number with a decimal fraction. It will be calculated in the units specified in the associated Units field.

The width of the page is measured perpendicular to the direction of the paper movement through the Laser Printer, and is device dependent. The printer model was a required specification in the Device field on the Main Menu.

Where the device has been specified as **2680A**, and the standard 8 1/2 by 11 tractor-fed paper is being used; the default entry would be:

11.0 Width

Where the device has been specified as **2688A**, and the standard 8 1/2 by 11 single sheet paper is being used; the default entry would be:

8.5 Width



Length

(in direction of paper motion)

You define the page length in this field as a whole number with a decimal fraction. It will be calculated in the units specified in the associated Units field.

The length of the page is measured parallel to the direction of the paper movement through the Laser Printer, and is device dependent. The printer model was a required specification in the Device field on the Main Menu.

Where the device has been specified as **2680A**, and the standard 8 1/2 by 11 tractor-fed paper is being used; the default entry would be:

8.5 Length

Where the device has been specified as **2688A**, and the standard 8 1/2 by 11 single sheet paper is being used; the default entry would be:

11.0 Length

Units

The default for this field is **I** (=Inches). You may respecify this field by changing the **I** to **C** (=Centimeters).

The unit value in this field is used to calculate the physical page

dimensions specified in the associated width and length fields.

4. The Primary and Secondary Font Fields

Initial primary font number
-OR- name

Initial secondary font number
-OR- name

These fields are useful for controlling which fonts will be used as primary and secondary character fonts. If you do not specify a primary and secondary character font using these fields, IFS/3000 defaults to the character font which has been assigned the lowest ID-number (00) as the primary character font, and the next highest ID-number (01) as the secondary character font. If only one character font is defined in the environment, it is used for both the primary and secondary fonts. If no character fonts have been defined in the environment, IFS/3000 uses character font number 31, the Laser Printer default, for both the primary and secondary fonts.

You define the primary and secondary character fonts using these fields as follows:

You choose the primary character font by typing the character font ID-number, OR, character font name in these fields:

ID Initial primary font number
-OR- userfontname name

The font ID-number OR name can be that of any character font which has been defined in the environment. NOTE: You may not use the ID-number and the name.

You choose the secondary character font by typing the desired character font's ID-number; OR, the character font's name in these fields as follows:

ID Initial secondary font number
-OR- userfontname name

The font ID-number OR name can be that of any character font which has been defined in the environment. NOTE: You may not use the ID-number and the name.

If you need to review character font ID-numbers or any corresponding user font names, you would use the Character Font Menu (described later in this section).

You may associate instructions with the data file which will activate the secondary character font. Refer to the discussion, "Primary and Secondary Character Fonts" in Section 2.

5. The Default Measurement System Field

E Default measurement system

The default value for this field is **E** (=English). You can change the default **E** to **M** (=Metric). The units you select here will establish the default system of measurement for all specifications entered on the Logical Page, Logical Page Forms, Vertical Format Control, and Character Font Menus.

6. The Default Form File Field

Default Form File

This is an optional field. IF you specify a forms filename in this field, IT becomes the default forms file for the environment.

All forms for an environment are usually contained in a single forms file. You can save time when specifying each form on the Logical Page Forms and Multi-Copy Forms menus by establishing the forms filename as the default in this field. This enables you to enter the form name on these menus without having to repeat the filename for each entry by defaulting to the filename you have established here.

You enter the name of the forms file using standard MPE File System naming conventions, where the filename may have up to 8 characters, and can be any letter followed by any letter or number. If the file resides in a group other than the sign-on group, you must also specify a groupname and accountname. You may also specify a lockword if this form of security applies to the file.

If you need to use all of these file designation conventions, the field data entry would appear as follows:

Default Form File

filename/lockword.groupname.accountname

When you press **ENTER**, the IFS/3000 program checks to see that the filename specified exists and that it is a forms file. If the file does not exist as specified, a status message will be issued indicating that:

- The filename does not meet MPE filename conventions.
- The file does not reside in the group and account specified.
- You cannot access the file due to security violations.
- The file specified is not a forms file.

The Multi-Copy Forms Menu

IFS		Multi Copy Forms		Environment: SAMPLE	
<input type="checkbox"/> Logical page that will be used to write to these forms					
Form	Form File (if not default)	Copy	Scale? Y/N		
		Copy 1	N		
		Copy 2			
		Copy 3			
		Copy 4			
		Copy 5			
		Copy 6			
		Copy 7			
		Copy 8			
		<input type="button" value="Refresh"/>	<input type="button" value="Main"/>	<input type="button" value="Cancel"/>	

Figure 3-10. The Multi-Copy Forms Menu

If you enter **Y** in the Multi-Copy Forms? field on the Physical Page Menu and press **ENTER**, the Multi-Copy Forms Menu is displayed.

You use this menu to specify up to eight pairs of forms to be printed on the physical page. If you want to write data to the forms at print time, you must also identify the logical page which has been designed to position data within the forms. You would use the Logical Page Menu to design the

associated logical page before you identify it on this menu.

This procedure can be used to simulate carbon copy production where different configurations of the same general form appear on subsequent physical pages.

Function Keys

The function keys used with this menu are:

ENTER

Once you have typed the required field data and any optional data in the menu fields, press the **ENTER** key to issue the data to the computer. IFS/3000 will issue status messages to the menu banner IF it has difficulty accessing the specified form, form file, or logical page; OR, if the form(s) specified will not fit within the limitations of the physical page. OR, the program will issue a bell to indicate that the field data is acceptable. The menu will remain on the screen for your review. IF you do not make any changes, you can press **ENTER** again, OR press the **Main** function key, to return to the Main Menu.

Refresh

Press **F6** to redisplay the terminal screen with any data entered using the **ENTER** key. It is useful when you have used the **Cancel** key.

Main

Press **F7** to return to the Main Menu. Note that any field data you have typed but have not entered will be lost.

Cancel

Press **F8** to delete any data you have just typed, but have decided not to enter.

Field Discussion

There are four groups of data entry fields on the Multi-Copy Forms Menu.

You use these fields to identify the forms you wish to use, the filename in which they reside, the logical page to be used to write data to the forms, and if the program is to scale the forms to fit within the logical page limitations.

When you associate a logical page, IFS/3000 centers the forms ON the logical page. If you also scale the forms, IFS/3000 centers the forms WITHIN the printable area of the logical page. You should determine if your form includes borders which are to extend beyond the printable area of the logical page, or if the entire form is to be positioned within the printable area. You cannot write data to any part of a form which extends beyond the printable area defined in a logical page layout.

You MUST associate a logical page with your forms IF you want to scale and position the forms on the physical page, AND you want the forms to have the same orientation as the logical page used to write data at print time.

You can use a pair of forms to define each copy of a multiple part form. Normally you use the first form in the pair to define the parts of the form which are constant from one copy to the next. You use the second form in the pair to define the parts of the form which vary from one copy to the next. IFS/3000 overlays the first form in each pair with the second form in the pair, centering

each form on the associated logical page. Each form must be designed so that it is placed appropriately when the forms are overlaid. This is easier if both parts of the form are the same size and shape.

The field data and its effect on the environment is defined as follows:

1. The Logical Page Field

Logical Page that will be used to write to these forms

This is an optional field. You would use this field to identify the logical page to be used to write data to the form specified in the associated Form field.

The logical page you identify here should define the position where data will print within the form layout. In addition, the forms will be rotated so that the forms and the print have the same orientation as the logical page. You may also specify that the forms be scaled to fit within the logical page using the Scale? Field on this menu.

You may specify the ID-number of any logical page that has been defined in the environment. If you do not specify a logical page, IFS/3000 uses the logical page with the lowest logical page ID-number (00). If there are no logical pages defined in the environment, IFS/3000 uses the Laser Printer default logical page.

If the logical page you specify in this field has not been defined, the program issues a status message to the menu banner.

2. The Form Field

Form



This is a required field. You use this field to identify the forms you wish to associate with the physical page. The form's name is the name assigned when it was created in IDIFORM. The field provides enough space to enter up to eight pairs of form names. You may repeat any form name as many times as desired. The form name can contain up to 16 characters, beginning with a letter, followed by any letter or number.

If the form does not exist in the form file specified, IFS/3000 displays an error message in the menu banner.

The forms print in the order that they are specified in this field.

You can specify up to two forms to be used to produce the first copy. In this case, the Laser Printer prints the first form and then overprints it with the second to give the appearance of a single form. If you are also writing data to the form at print time, the data is printed to the position defined by the associated logical page. The Laser Printer then issues a physical page eject.

The second copy is printed on the next page in the same pattern: the first form specified for copy two prints, the second form specified overprints, and the same data then overprints the two forms.

THE MULTI-COPY FORMS MENU

The third, and each subsequent copy follows this procedure.

The field entry for the example shown in figure 3-11 would be as follows:

Form	
formA	
formB	Copy 1
formA	
formC	Copy 2
formA	
formD	Copy 3
	Copy 4
	Copy 5
	Copy 6
	Copy 7
	Copy 8

3. The Form File Field

Form File (if not default)

--

This field is optional if you elected to specify a default forms file on the Physical Page Menu, and it contains the forms you wish to identify in the Form Name Field on this menu. You can override

the default forms filename by specifying a filename for any form which does not reside in the default forms file. IF you do not use the default forms file you **MUST** specify each forms filename individually.

Normally all forms created using IDIFORM for a specific environment are stored in the same forms file.

You enter the name of the forms file using standard MPE File System naming conventions, where the filename may have up to 8 characters, and can be any letter followed by any letter or number. If the file resides in a group other than the sign-on group, you must also specify a groupname and accountname. You may also specify a lockword if this form of security applies to the file.

If you need to use all of these file designation conventions, the field data entry would appear as follows:

Form File (if not default)

filename/lockword.groupname.accountname

When you press **ENTER**, the IFS/3000 program checks to see that the filename specified exists, that it is a forms file, and that the form name resides within the file. The form definition is then included in the environment file and will be compiled when you perform that operation using the Main Menu.

THE MULTI-COPY FORMS MENU

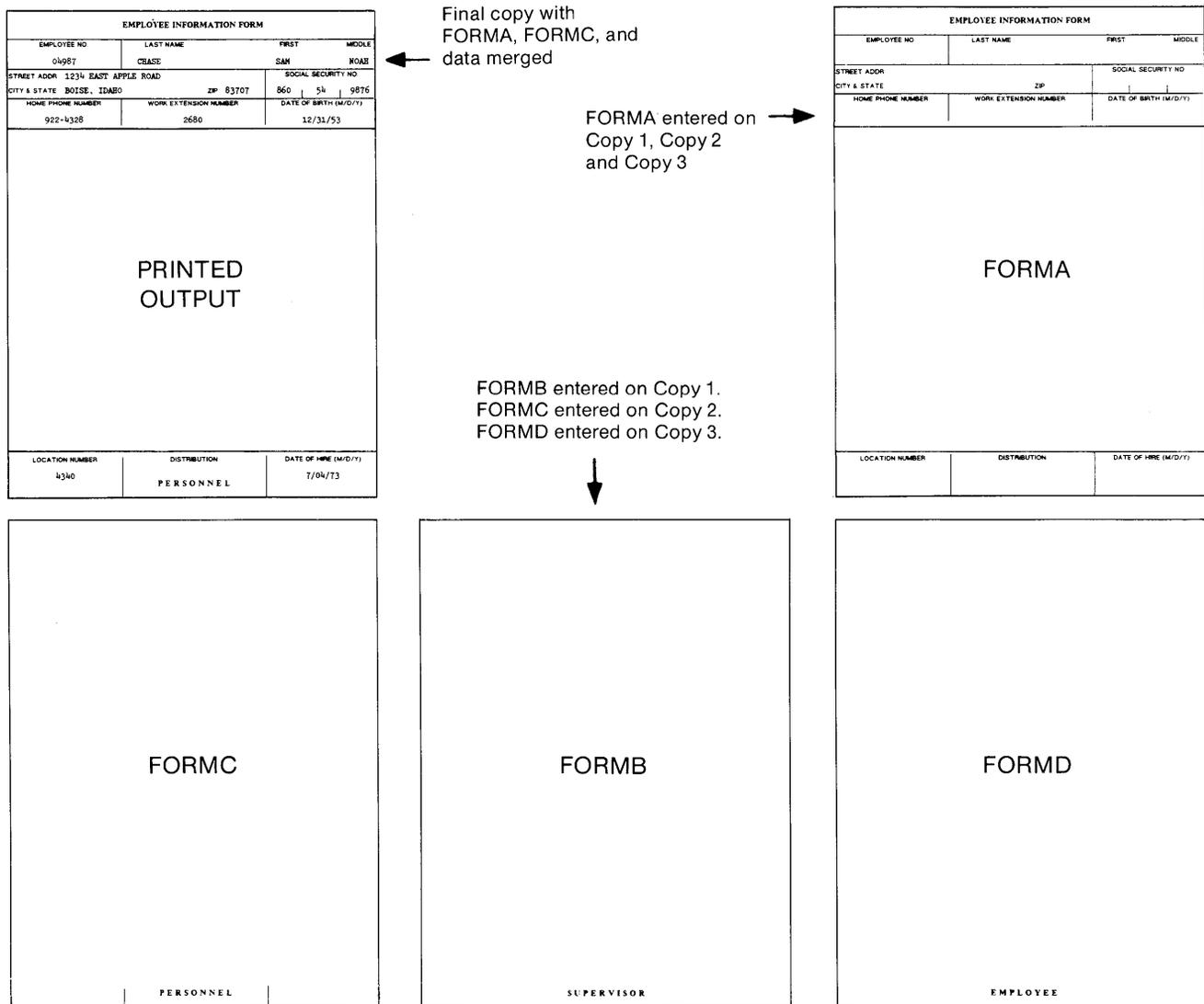


Figure 3-11. Multi-Copy Forms

THE MULTI-COPY FORMS MENU

If the file does not exist as specified, a status message will be issued indicating that:

- The filename does not meet MPE filename conventions.
- The file does not reside in the group and account specified.
- You cannot access the file due to security violations.
- The form name does not exist within the file.
- The file specified is not a forms file.

If you did not elect to use the default forms filename on the Physical Page Menu, your entries in this field could have the following variations.

Form File (if not default)

filename.groupname.accountname
filename.groupname.accountname
filename

filename
filename

4. The Scale? Field

Scale? Y/N

N

The default for this field is N (=no). IF you wish to scale your forms, you MUST also specify a logical page in the Logical Page Field on this menu, AND change the default N to Y (=yes) in this field. The forms will then be rotated and scaled to fit within the logical page identified on this menu and all portions of the forms will be within the printable area defined on the logical page.

When scaling forms, you must be aware of the process IFS/3000 uses.

All lines and boxes are scaled to fit within the parameters of the logical page defined. When the form dimensions reach the limitations of the logical page either in width or height, the scaling process stops. Therefore, your form may be either narrow or short depending on the logical page parameters.

When IFS/3000 scales the fonts it simply looks at the character font referenced in IDIFORM and selects from the sizes resident in that font. If the smallest available character font is too big, the type will extend beyond the lines on the form.

The Logical Page Menu

IFS	Logical Page	Environment: SAMPLE
<input type="checkbox"/>	Logical page number	<input type="checkbox"/> Initially active? Y/N
<input type="checkbox"/>	Orientation: L - Landscape, RL - Reverse Landscape, P - Portrait, RP - Reverse Portrait	
<input type="checkbox"/>	Change Forms or VFC? Y/N	Units I-in, C-cm, M-mm, D-dots -OR-
<input type="checkbox"/>	Width	<input type="checkbox"/> NC - number of characters
<input type="checkbox"/>	Height	<input type="checkbox"/> NL - number of lines
<input type="checkbox"/>	Distance from left	
<input type="checkbox"/>	Distance from top	
		-OR-
<input type="checkbox"/>	Overriding line spacing	<input type="checkbox"/> LI - lines/inch, LC - lines/cm
<input type="checkbox"/>	Left margin	<input type="checkbox"/> NC - number of characters
<input type="checkbox"/>	Font number for VFC -OR-	<input type="checkbox"/> name
	Actual spacing information:	
	_____ dots character width	_____ dots line height
<input type="checkbox"/>	First LP	<input type="checkbox"/>
<input type="checkbox"/>	Last LP	<input type="checkbox"/>
<input type="checkbox"/>	Prev LP	<input type="checkbox"/>
<input type="checkbox"/>	Next LP	<input type="checkbox"/>
		<input type="checkbox"/> Refresh
		<input type="checkbox"/> Main
		<input type="checkbox"/> Cancel

Figure 3-12. The Logical Page Menu

If you enter **L** in the Main Menu Selection field, the Logical Page Menu is displayed.

This menu allows you to define the logical pages to be included in your environment file. For each logical page, you define its orientation, size, and location on the physical page.

The Logical Page Menu contains the path to the Logical Page Forms Menu; the Logical Page Forms Menu contains the path to the VFC Selection Menu.

When you go to the Logical Page Menu from the Main Menu, IFS/3000 displays the first logical page number. This is usually page number zero, unless you have entered a specific logical page

number on the Main Menu. The function keys allow you to access other logical page definitions.

Function Keys

The function keys used with this menu are:

ENTER

Once you have typed your logical page specifications in the menu fields, press the **ENTER** key to issue this data to the computer. IFS/3000 will issue status messages to the menu banner if any required data is not specified, or if it has difficulty processing the data. OR, the program will issue a bell to indicate that the field data was acceptable. The menu will remain on the screen for your review, OR the Logical Page Forms menu will be displayed, if specified. You can use the function keys to proceed directly to another logical page, OR press the **Main** function key, to return to the Main Menu. Pressing **ENTER** again without making a change will also return you to the Main Menu.

First LP

Press **F1** to display the first logical page defined. If you have not defined specifications for this logical page, the menu will appear blank except for the logical page **ID**-number and field defaults.

Last LP

Press **F2** to display the last logical page defined. If you have not defined specifications for this logical page, the menu will appear blank except for the logical page **ID**-number and field defaults.

Prev=LP

Press **F3** to display the logical with an **ID**-number ONE LESS than the current logical page. If you have not defined specifications for this logical page, the menu will appear blank except for the logical page **ID**-number and field defaults.

Next LP

Press **F4** to display the Logical Page Menu for the logical page with an **ID**-number ONE GREATER than the current logical page. IF you have not defined specifications for this logical page, the menu will appear blank except for the logical page **ID**-number. The Logical Page Menu provides the path to the Logical Page Forms Menu when you specify **Y** in the Change Forms or VFC? Field and press **ENTER**.

Refresh

Press **F6** to redisplay the terminal screen with any data entered using the **ENTER** key.

Main

Press **F7** to return to the Main Menu. Note that any field data you have typed but have not entered will be lost.

Cancel

Press **F8** to delete any data you have just typed, but have decided not to enter.

Field Discussion

There are six groups of fields on the Logical Page Menu.

The first field established the **ID**-number for the logical page specifications you are entering or reviewing on this menu. This **ID**-number remains associated with the specifications unless you change it. It enables you to identify an individual logical page in a data file or application program which can then be activated or deactivated by the Laser Printer at print time.

The second field allows you to specify whether or not the logical page is active at the beginning of the print job.

The third field establishes the orientation of the logical page as either landscape, portrait, reverse landscape or reverse portrait with respect to the physical page. Each orientation specifies a rotation of the logical page on the physical page. For more information refer to the description of logical page in Section 2.

The fourth field is used to proceed to the Logical Page Forms Menu and subsequently to the Vertical Format Control Menu.

The remaining fields allow you to specify the dimensions of the logical page, to position the logical page exactly on the physical page, to describe some information relating to VFC specification and to set the left margins for printing characters on the logical page.

At the bottom of this menu, protected fields are used to display the width and height of the character font chosen in the fields directly above. If no font is chosen here, a default is chosen as described below.

The field data and its effect on the environment is defined as follows:

1. The Logical Page Number Field

Logical page number

This is a required field. You can identify up to 32 logical pages to be included in your environment. Each logical page must be defined individually and given an **ID**-number in this field.

A logical page **ID**-number from **00** to **31** will always appear in this field when the Logical Page Menu is displayed. The number that appears depends on the method you use to display the menu as follows:

- If you did not enter a specific page number in the Font/Page number field on the Main Menu, logical page **00** will be displayed. If you have not defined specifications for this logical page, the menu will appear blank except for the **ID**-number and field defaults.

THE LOGICAL PAGE MENU

- If you DID enter a specific page number in the Font/Page Number, the logical page specified will be displayed. If you have not defined specifications for this font number, the menu will appear blank except for the ID-number and field defaults.
- If you pressed a function key on this menu, the logical page ID-number corresponding with the action described on the key label will be displayed. Refer to the function key discussions.

You would change a logical page number ONLY if you wish to assign a different number than the number currently assigned on the menu. To change a number, type a number from 00 to 31 over the number in this field before using the **ENTER** key. This makes an identical copy of the logical page. Therefore, you may wish to delete the logical page you copied.

2. The Initially Active? Field

Y Initially active? Y/N

This is a required field. The default for this field is **Y** (yes), meaning that the logical page is active at the beginning of the print job. Entering **N** (no) means you do not want the page to be initially active.

A logical page must be active in order to print to it. Logical pages can be activated and deactivated during a print job either programmatically using the **PACTIVATEPAGE** and **PDEACTIVATEPAGE** intrinsics or using the **.ACTIVATEPAGE** and

.DEACTIVATEPAGE commands in the LPS interpreter.

3. The Orientation Field

Orientation:

This is a required field for which there is no default. You use this field to specify the orientation of the logical page in relation to the physical page. Orientation can be specified as follows:

L Landscape

P Portrait

RL Reverse Landscape

RP Reverse Portrait

Refer to Section 2, figure 2-6 for examples of the orientations available.

4. The Change Forms or VFC? Field

N Change Forms or VFC? Y/N

This field is required. The default for this field is **N** (no). You can specify forms to be associated with the logical page and/or change the vertical format control (VFC) by entering **Y** (yes).

When you specify **Y** (yes), IFS/3000 displays the Logical Page Forms Menu. This menu is used to

add, change or review forms associated with the logical page. The logical Page Forms Menu also contains a path to the Vertical Format Control (VFC) menu.

When you specify **N** (no) in this field, you will be returned to the Main Menu after you press **ENTER**.

5. The Size Specification and Positioning Fields

UNITS= I, C, M, D
-OR-
Width NC - number of characters
Height NL - number of lines

Distance from left
Distance from top

-OR-
Overriding line spacing LI - lines/inch, LC - lines/cm
Left margin NC - number of characters

Font number for VFC -OR- name

The fourteen fields in this group allow you to specify the exact size of the logical page and its position on the menu. The first two fields, specifying width and height for the logical page, are required. The other fields are optional.

You can use inches, centimeters, millimeters, dots, or characters and lines to measure the logical page dimensions and specify its position on the page. If you specify the logical page width using number of characters, IFS/3000 uses the Font number for VFC width to calculate the logical page width. Likewise, if you specify the logical page height using the number of lines, IFS/3000 uses the Font number for VFC height to calculate the logical page height.

Physical characteristics of the HP Laser Printer force an unprintable margin area on the paper. This requires that you leave space on some edges of the physical page. You can define logical pages which extend into the unprintable area, but you cannot print clearly in the unprintable area. You

cannot define a logical page which is bigger than the physical page defined in the physical page menu.

The alternates for size specification and positioning fields are described below.

Width

The width actually consists of two required specifications, the unit of measurement specified in the Units field and the number of units specified in the Width field. If the space required to print a line exceeds the width specified, IFS/3000 may truncate characters during printing. Valid unit specifications for logical page width are inches, centimeters, millimeters, dots and number of characters, defined as follows:

I, C, M

When using width in inches, centimeters and millimeters, the width must be smaller than

the corresponding dimension specified for the physical page.

D

When using number of dots to specify width, you should remember that different models of HP Laser Printers may have varying dot density. By using number of dots to specify dimensions, you are restricting use of the environment to a single device.

NC

When specifying **NC**, number of characters, a number or name is used to compute logical page width. It is convenient to use the same character font you are using to print your document, but this is not required. For more information, refer to the discussion of the font number for VFC below.

Height

Height, like Width, is specified using two required fields. The unit of measurement is specified in the Units field and the number of units is specified in the Height field. The HP Laser Printer uses the height of the font font for VFC plus any inter-line spacing specified using IDSCHAR to calculate vertical spacing. You may change the method IPS/3000 uses to calculate vertical spacing by using the Vertical Format Control Menu. Valid units for measurement of height are inches, centimeters, millimeters, dots and number of lines specified as follows:

I, C, N

When specifying height in inches, centimeters or millimeters, the height must be smaller than the corresponding dimension specified for the physical page.

D

When using number of dots to specify height, you should remember that different models of HP Laser Printers may have varying dot density. By using number of dots to specify dimension, you are effectively restricting use of the environment to a single device.

NL

When specifying number of lines, the VFC font number or name or the overriding line spacing, is used to compute logical page height.

It is convenient to use the same character font you are using to print your document, but this is not required. For more information, refer to the discussion of the VFC font number and name below.

**Distance From Left
Distance From Top**

The location of the logical page on the physical page is specified using the optional Distance From Left and Distance From Right fields along with their associated Units fields. When specifying the location of the logical page on the physical page, you view the physical page in the same orientation as the logical page. This means that you measure

THE LOGICAL PAGE MENU

from different places on the physical page for different logical page orientations. Figure 3-13 shows examples of specifying the location of logical pages with varying orientations. To specify location, enter the unit of measurement in the corresponding Units fields and the number of units in the Distance From Left and Distance From Top fields. Units can be specified in inches, centimeters, millimeters or dots. If these fields are left blank, the logical page will be centered on the physical page.

Overriding Line Spacing

IFS/3000 uses the Overriding Line Spacing field to control line spacing for the logical page. This allows you to vary the line spacing within the logical page, overriding the interline spacing defined using the font for VFC as a unit of measurement.

If you do not enter a value in the Overriding Line Spacing field, IFS/3000 uses the cell height of the font for VFC. If you do not specify a font for VFC below, IFS/3000 uses the initial primary character font as specified on the Physical Page Menu or the lowest number character font defined.

If you enter a value in the Overriding Line Spacing field, IFS/3000 uses this value instead of the cell height of the font for VFC. After you enter your specifications, IFS/3000 displays the actual line spacing to be used in the Line Height field in the Actual Spacing information at the bottom of the menu. This includes the inter-line spacing supplied in IDSCHAR. Refer to the *IDSCHAR Reference Guide (part number 36581-90001)*.

You enter a value in the Overriding Line Spacing field usually based on the cell height of the character font you are using to print your document. If you use a character font with a cell height that differs from that indicated by the Overriding Line Spacing field, the result may not be what you intended.

You may change the method of controlling line spacing by selecting "No VFC" on the Vertical Forms Control menu. With "No VFC", the cell height of the character being printed controls the line spacing, and the Overriding Line Spacing field is ignored.

Since the default for the HP/3000 is post-space mode, "No VFC" causes a line feed after printing a line equal to the height of the last character on the line (refer to Type of VFC Field in Vertical Format Control Menu).

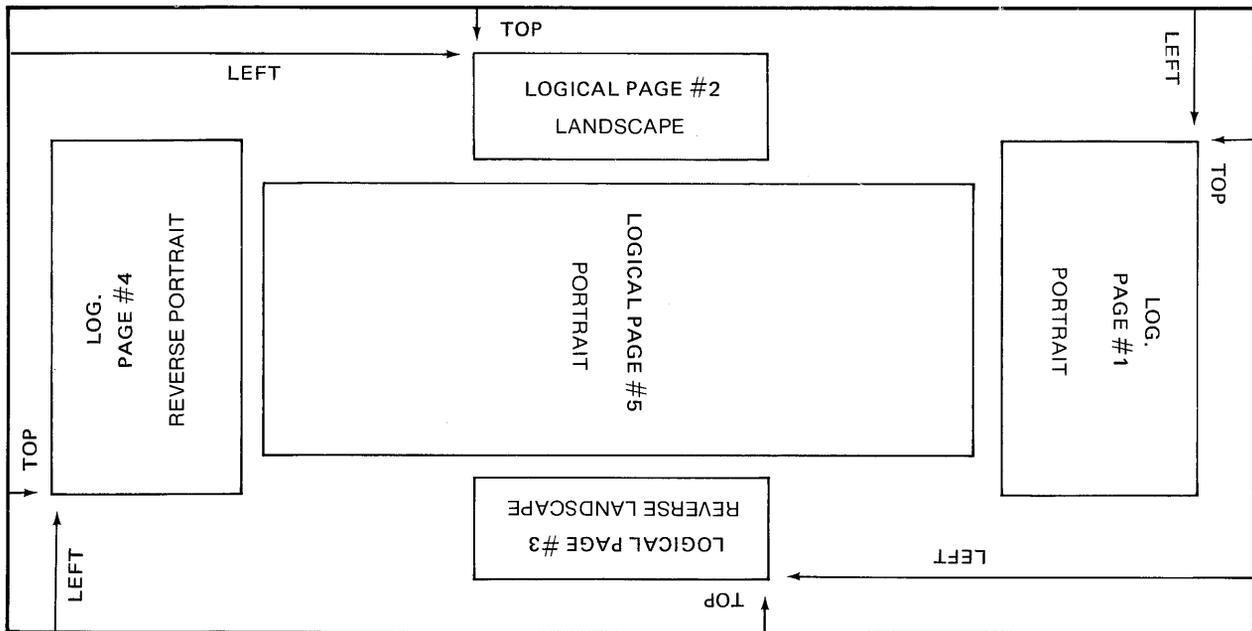


Figure 3-13. Specifying The Location of Logical Pages

THE LOGICAL PAGE MENU

You enter the unit of measurement in the Units field and the number of units in the Overriding Line Spacing field. You can use inches, centimeters, millimeters, or dots to measure line spacing, or you can express line spacing in lines per inch **LI** or lines per centimeter **LC**.

Left Margin

The Left Margin field determines the distance printing will start from the left edge of the logical page. The valid units of measurement for the left margin are inches, centimeters, millimeters, dots, or number of characters.

The left margin defaults to zero, causing the first print line to start at the left edge of the logical page. In most cases, you can use the default since the size and location of the logical page defines the apparent margin with respect to the physical page.

Font number for VFC -OR- name

You select the font for VFC from the fonts defined in your environment, but you do not have to use it to print your document. IFS/3000 refers to the font for VFC to calculate the vertical line spacing. You enter the number or name of the font in the Font for VFC fields. If you want to use the character font resident in the HP Laser Printer as the font for VFC, refer to the paragraphs on Effect of Character Font Number 31 below.

If you use number of characters (NC) as the unit of measurement for Width, you must also specify the font for VFC. IFS/3000 uses the width of the font for VFC, plus any inter-character spacing specified with IDSCHAR, to calculate the logical page width. IFS/3000 multiplies this total by the

number of characters to give the logical page width.

If the font for VFC is proportionally spaced, it is inappropriate to specify the logical page width in number of characters. If you use a proportionally spaced character font for the font for VFC and also specify page width in number of characters, IFS/3000 uses the maximum cell width of the font as the font for VFC width and displays a warning message at the bottom of the screen.

IFS/3000 checks that the character font orientation matches the logical page orientation. If they do not match, the font is still used as the font for VFC, but a warning message is displayed. To print in the normal direction of left to right across a page, the character font you are using must have the same orientation as the logical page you are printing on. The character font resident in the HP Laser Printer has a landscape orientation.

Normally it is convenient to use one of the character fonts you are using to print your document as the font for VFC, but this is not required. The width of the font for VFC is used only for calculation purposes; it is not used during the actual printing.

The actual horizontal spacing used at print time depends on the width of the character font selected for printing, not the font for VFC width. In addition to the character font width, the spacing used at print time includes the inter-character spacing defined with IDSCHAR. Refer to the *IDSCHAR Reference Guide (Part Number 36581-90001)*. If the width of the character font selected for printing is larger than the font for VFC width, lines may be truncated when printed.

If you use number of lines (NL) as the unit of measurement for logical page height you must also specify the font for VFC. IFS/3000 uses the height of the font for VFC to calculate the logical page height. The font for VFC height multiplied by the number of lines gives the logical page height.

The actual vertical line spacing used at print time also includes any inter-line spacing defined with IDSCHAR. In contrast to horizontal character spacing, vertical line spacing does not depend on the character font used for printing (unless you choose the "No VFC" option on the VFC Selection Menu).

If you do not specify a value in the Font Number for VFC or Name field, IFS/3000 selects the initial primary character font for the font for VFC. You select the primary character font on the Physical Page Menu.

If you do not select a primary character font, IFS/3000 uses the character font with the lowest ID-number. If there are no character fonts defined in the environment, IFS/3000 uses the character font resident in the HP Laser Printer as the font for VFC. The orientation of this font is landscape.

There may be a special case when you have defined character fonts in the environment file, but do not want to use any of them for the font for VFC. Instead, you may want to use the character font resident in the HP Laser Printer. If this is the case,

enter "31" in the Font Number for VFC field. IFS/3000 uses this as the font for VFC even though you have not defined it in the environment file. If you have defined a character font with the number 31, the HP Laser Printer default character font is no longer accessible.

6. Actual Spacing

IFS/3000 displays the width and height of the font for VFC in the fields labeled Actual Spacing Information. If you do not specify a font for VFC, IFS/3000 selects one as described in the paragraphs on default font for VFC above.

IFS/3000 uses the value in the Height field to determine line spacing. The Height field determines the actual vertical line spacing used at print time. In contrast, the actual horizontal character spacing used at print time depends on the character font being printed, not the value in the Width field. If the character font you are printing with does not match the size of the font for VFC, print lines may be truncated or overlaid on top of each other.

In the Dots character width field, IFS/3000 displays the width of the font for VFC plus inter-character spacing expressed in dots. For proportionally spaced fonts, the maximum cell width for the font specified is used.

The Logical Page Forms Menu

```

IFS                               Logical Page Forms                               Environment:  SAMPLE
N Change VFC specification?

Form 1                               Form File 1 (if not default)           Scale? Y/N
_____                               _____                               N

      Position on Page                -OR-                Manual Positioning
      █ █ █                            █                █ Distance to move right
      █ █ █                            █                █ Distance to move down
      █ █ █                            █                █ Units:  I - in,  C - cm,
      █ █ █                            █                █                M - mm,  D - dots

Form 2                               Form File 2 (if not default)           Scale? Y/N
_____                               _____                               N

      █ █ █                            █                █ Distance to move right
      █ █ █                            █                █ Distance to move down
      █ █ █                            █                █ Units:  I - in,  C - cm
      █ █ █                            █                █                M - mm,  D - dots

First LP  Last LP  Prev LP  Next LP                Refresh  Main  Cancel
  
```

Figure 3-14. The Logical Page Forms Menu

If you enter **N** in the Change Forms or VFC? Field on the Logical Page Menu and press **ENTER**, the Logical Page Forms Menu is displayed.

You would use this menu to identify up to two forms which reside in a forms file that are to be associated with the logical page. The data entry fields also enable you to specify where you want the forms to be positioned on the logical page.

Once your environment is compiled, the forms will print as part of the logical page layout.

Review Sections 2, 4, and 5 for information on writing data to the forms you have associated with a logical page, at print time.

This menu also provides the menu path to the Vertical Format Control Menu when you specify **N** in the Change VFC Specification Field.

Function Keys

The function keys used with this menu are:

ENTER

Once you type the Form name and any associated field data, press the **ENTER** key to issue this data to the computer. IFS/3000 will issue status messages to the menu banner if any required data is not specified, if it cannot access the files referenced, or if the forms specified are inconsistent with the dimensions specified on the Logical Page or Physical Page Menus. OR, the program will issue a bell to indicate that the field data was acceptable. The menu will remain on the screen for your review. You can use the function keys to proceed directly to another logical page, OR press the **Main** function key to return to the Main Menu. Pressing **ENTER** again without making changes also returns you to the Main Menu.

First LP

Press **F1** to display the Logical Page Menu for first logical page defined. IF you have not defined specifications for this logical page, the menu will appear blank except for the logical page **ID**-number. The Logical Page Menu provides the path to the Logical Page Forms Menu when

Last LP

you specify **Y** in the Change Forms or VFC? Field and press **ENTER**.

Press **F2** to display the Logical Page Menu for last logical page defined. IF you have not defined specifications for this logical page, the menu will appear blank except for the logical page **ID**-number. The Logical Page Menu provides the path to the Logical Page Forms Menu when you specify **Y** in the Change Forms or VFC? Field and press **ENTER**.

Prev LP

Press **F3** to display the Logical Page Menu for the logical page with an **ID**-number ONE LESS than the current logical page. IF you have not defined specifications for this logical page, the menu will appear blank except for the logical page **ID**-number. The Logical Page Menu provides the path to the Logical Page Forms Menu when you specify **Y** in the Change Forms or VFC? Field and press **ENTER**.

Next LP

Press **F4** to display the Logical Page Menu for the logical page with an **ID**-number ONE GREATER than the current logical page. IF you have not defined specifications for this logical page, the menu will appear

blank except for the logical page ID-number. The Logical Page Menu provides the path to the Logical Page Forms Menu when you specify Y in the Change Forms or VFC? Field and press **ENTER**.

Refresh

Press **^6** to redisplay the terminal screen with any data entered using the **ENTER** key.

Main

Press **^7** to return to the Main Menu. Note that any field data you have typed but have not entered will be lost.

Cancel

Press **^8** to delete any data you have just typed, but have decided not to enter.

Field Discussion

There are six groups of data entry fields on the Logical Page Forms Menu.

The first field is used to specify that you wish to proceed to the Vertical Format Control Menu. The remaining fields enable you to identify the forms to be associated with the logical page, and to specify the position of the form on the logical page. These fields are repeated on the menu to enable you to specify two forms to be associated with the logical page.

IF you do not change the default position field, IFS/3000 automatically centers each form ON the

logical page. IF you also specify that the forms be scaled, they will be centered WITHIN the printable area of the logical page.

Where you have specified two forms, the forms can be positioned independently, or overlay each other partially or completely. You can combine the Position On Page Field with the Manual Positioning Field to provide precise control on forms placement on the logical page.

When you have specified the field data and press **ENTER**, IFS/3000 will display status messages to the menu banner if it encounters inconsistencies as follows:

- The form has been positioned such that part of the form extends beyond the limits of the logical page.
- The form has been positioned such that part of the form extends into the unprintable border area of the physical page.
- The form is too large to fit on the physical page.

The field data and its effect on the environment is defined in the following discussions for Form 1. The fields repeat on the menu for Form 2. The same definitions apply to the Form 2 Fields.

1. The Change VFC Specification Field

N Change VFC specification? Y/N

The default for this field is **N** (=no). IF you want to proceed to the Vertical Format Control Menu, you **MUST** change **N** to **Y** (=yes) and press **ENTER**. You do not need to specify any other field.

2. The Form 1 Field

Form 1
.....

This field is used to identify the form you wish to associate with the logical page. The form's name is the name assigned when it was created in IDSFORM. The form name can contain up to 16 characters, beginning with a letter, followed by any letter or number.

If the form does not exist in the form file specified, IFS/3000 displays an error message in the menu banner.

3. The Form File 1 Field

Form File 1 (if not default)
.....

This field is optional if you elected to specify a default forms file on the Physical Page Menu, and it contains the forms you wish to identify in the Form 1 Field on this menu. You can override the default forms filename by specifying a filename for any form which does not reside in the default

forms file. IF you do not use the default forms file you **MUST** specify the forms filename in this field.

Normally all forms created using IDSFORM for a specific environment are stored in the same forms file.

You enter the name of the forms file using standard MPE File System naming conventions where the filename may have up to 8 characters, and can be any letter followed by any letter or number. If the file resides in a group other than the sign-on group, you must also specify a groupname and accountname. You may also specify a lockword if this form of security applies to the file.

If you need to use all of these file designation conventions, the field data entry would appear as follows:

Form File 1 (if not default)
Filename/lockword.groupname.accountname

When you press **ENTER**, the IFS/3000 program checks to see that the filename specified exists, that it is a forms file, and that the form name resides within the file. The form definition is then included in the environment file and will be compiled when you perform that operation using the Main Menu.

If the file does not exist as specified, a status message will be issued indicating that:

- The filename does not meet MPE filename conventions.

THE LOGICAL PAGE FORMS MENU

- The file does not reside in the group and account specified.
- You cannot access the file due to security violations.
- The form name does not exist within the file.
- The file specified is not a forms file.

4. The Scale? Field

Scale? Y/N

N

The default for this field is N (=no). IF you wish to scale your form, you **MUST** change the default N to Y (=yes) in this field.

IF you do not scale a form, the form is centered ON the logical page. IF you scale a form, the form is centered **WITHIN** the printable area on the logical page. Scaling is proportional.

You should determine if your form includes borders which are to extend beyond the printable area of the logical page, or if the entire form is to be positioned within the printable area. You cannot write data to any part of a form which extends beyond the printable area defined in a logical page layout.

In addition, when a form is scaled, IFS/3000 does not scale character fonts and logos in the same way it scales the lines and boxes in a form. Instead, it selects the font point size or logo which is as close

as possible to the scaling of the rest of the form. If the available font selection does not include an appropriate size, the text embedded in the form may not be positioned properly. You should therefore use care when scaling forms.

5. The Position On Page Fields

Position on Page

There are nine data entry fields in this group. The default for this group is in the centered position. You can change the default by typing an in any other field position. Reading from left to right, the positioning of the form on the logical page is defined as follows:

Top: left, center, right

Center: left, centered, right

Bottom: left, center, right

By placing an X in the top left field, your form would be positioned in the upper left corner of the logical page. By placing an X in the centermost field of the center line, your form would be positioned in the center of the logical page.

You can adjust this positioning field by also specifying the Manual Positioning Field.

6. The Manual Positioning Field

Manual Positioning

Distance to move right
Distance to move down
Units: I - IN, C - CM,
M - MM, D - Dots

This is an optional data entry field group. There are three fields in this positioning group. You use the first two to specify the number of units to move the form on the logical page. You use the third field to specify the value of the units.

IF you do not enter specifications in these fields, IFS/3000 defaults to the Position On Page Field. You can use the Manual Positioning field to offset from the automatic Position On Page Field entry. Any time you enter data in these fields which is inconsistent with the limitation of the logical page or physical page, the program will issue status messages to the menu banner.

The effect of this field data is defined as follows:

Distance to move right

The number you specify in this field can be either a positive or negative whole number or whole number and decimal fraction; if appropriate to the units specified in the associated Units Field.

The number you specify here will reposition the form starting at the position indicated in the Position On Page Field. This field moves the form to the right the number of units entered in this field, calculated at the value you specify in the associated Units Field. IF you specify a negative number, the form is moved to the left.

Distance to move down

The number you specify in this field can be either a positive or negative whole number or whole number and decimal fraction; if appropriate to the units specified in the associated Units Field.

The number you specify here will reposition the form starting at the position indicated in the Position On Page Field. This field moves the form down the number of units entered in this field, calculated at the value you specify in the associated Units Field. IF you specify a negative number, the form is moved up.

THE LOGICAL PAGE FORMS MENU

The number you specify in these positioning fields will be calculated in the unit value you specify in the Units Field. The alternates to the Units Field are defined as follows:

I IN

This alternate will calculate the number you specify in INCHES.

C CM

This alternate will calculate the number you specify CENTIMETERS.

M MM

This alternate will calculate the number you specify MILLIMETERS.

D Dots

This alternate will calculate the number you specify in the number of dots specified. Refer to the operator's manual for the device you specified in the Device Field on the Main Menu to review the number of dots per inch when specifying this alternate.

Field groups 2 through 6 above are repeated on the menu to allow you to load 2 forms on a single logical page.

The Vertical Format Control Menu

```
IFS                               Vertical Format Control                               Environment:  SAMPLE

$ Type of VFC.

S - Standard, automatically computed VFC
  3 lines before first line
  3 lines after last line

F - Special VFC from a file
  VFC File
  _____

N - No VFC (space according to character height)
  0 Top margin
  I Units  I - IN, C - CM, M - MM, D - Dots

First LP  Last LP  Prev LP  Next LP  Refresh  Main  Cancel
```

Figure 3-15. The Vertical Format Control Menu

If you enter **Y** in the Change VFC Specification? Field on the Logical Page Forms Menu and press **ENTER**, the Vertical Format Control Menu is displayed.

The number of lines of print and the amount of blank space between the lines establishes the vertical format control for the logical page. Each print line represents a horizontal channel with a vertical height which you specify as equal to a font

cell size or specific unit height. You define these VFC specifications on the Logical Page Menu.

You would use this menu **ONLY IF** you want to change the default method of computing VFC or provide additional specifications for an alternate method.

Your environment may contain a number of character fonts in various point sizes. The Laser

Printer enables you to vary the fonts and point sizes within each channel of print on a logical page. When the range of font point sizes is extreme, you may need to define a logical page where the VFC has been adjusted to accommodate specific point sizes. This menu enables you to specify a unique VFC for any logical page.

Function Keys

The function keys used with this menu are:

ENTER

Once you type VFC selection and any associated field data, press the **ENTER** key to issue this data to the computer. IFS/3000 will issue status messages to the menu banner if it has difficulty processing the VFC. OR, the program will issue a bell to indicate that the field data was acceptable. The menu will remain on the screen for your review. You can use the function keys to proceed directly to another logical page. OR, IF you do not make any changes, you can press **ENTER** again, OR press the **Main** function key to return to the Main Menu.

First LP

Press **F1** to display the Logical Page Menu for first logical page defined. IF you have not defined specifications for this logical page, the menu will appear blank except for the logical page **ID**-number. The Logical Page

Menu provides the path to the Vertical Format Control Menu when you specify **Y** in the Change Forms or VFC? Field and press **ENTER**.

Last LP

Press **F2** to display the Logical Page Menu for last logical page defined. IF you have not defined specifications for this logical page, the menu will appear blank except for the logical page **ID**-number. The Logical Page Menu provides the path to the Vertical Format Control Menu when you specify **Y** in the Change Forms or VFC? Field and press **ENTER**.

Prev LP

Press **F3** to display the Logical Page Menu for the logical page with an **ID**-number ONE LESS than the current logical page. IF you have not defined specifications for this logical page, the menu will appear blank except for the logical page **ID**-number. The Logical Page Menu provides the path to the Vertical Format Control Menu when you specify **Y** in the Change Forms or VFC? Field and press **ENTER**.

Next LP

Press **F4** to display the Logical Page Menu for the logical page with an **ID**-number ONE GREATER than the current logical page. IF you have not

defined specifications for this logical page, the menu will appear blank except for the **ID** page **ID**-number. The Logical Page Menu provides the path to the Vertical Format Control Menu when you specify **Y** in the Change Forms or VFC? Field and press **ENTER**.

Refresh

Press **F6** to redisplay the terminal screen with any data entered using the **ENTER** key.

Main

Press **F7** to return to the Main Menu. Note that any field data you have typed but have not entered will be lost.

Cancel

Press **F8** to delete any data you have just typed, but have decided not to enter.

Field Discussion

There are four groups of data entry fields on the Vertical Format Control Menu.

The first field is used to specify the method to be used to compute the VFC for the current logical page. The remaining fields are used to specify data that may be required to perform the computation for the method selected.

The field data and its effect on the environment is defined as follows:

1. The Type of VFC Field

S Type of VFC?

The default for this field is **S** (=Standard). You may select an alternate method of computing VFC by changing the **S** to either of the other alternates. All alternates for this field are defined as follows:

S Standard, automatically computed VFC

This selection is the default method of computing VFC. This method automatically computes the VFC using the specifications entered on the Logical Page Menu as follows:

- The logical page height is divided by the number of lines specified in the Overriding Line Spacing Field if overriding line spacing is defined.

OR,

- The logical page height is divided by the height of the character cell for the font for VFC specified in the Font Number or Font Name For VFC Field if font for VFC is defined.

OR,

- The logical page height is divided by the height of the lowest number character font if a character font is defined.

OR,

THE VERTICAL FORMAT CONTROL MENU

- The logical page height is divided by the height of the font resident in the HP Laser Printer.

This calculation determines the number of channels reserved for print, with blank line spacing of equivalent size between each of the print lines.

F Special VFC from a file

You would use this method if your environment has to be compatible with an existing program which uses a special VFC. The user-defined VFC must be contained in a standard ASCII file containing 80 byte records. The associated VFC File Field is a required field when selecting this method. Refer to the VFC File Field discussion in this section. Refer to Appendix E for discussions and examples of user-defined VFC specifications.

NOTE

The user-defined VFC file provides programmatic access to print positioning by establishing a VFC table with predefined channel positions. The height of the channel and the line spacing between channels is calculated in the same procedure as defined in the Standard VFC discussion above. Therefore, the specifications you enter on the Logical Page Menu are required in order to use this method of VFC

computation; OR, defaults for the Logical Page Menu Fields will apply.

N No VFC (space according to character height)

This selection does not use the Overriding Line Spacing or Font Number or Font Name Field specifications on the logical page menu.

It provides the most flexibility of the three methods by determining the line spacing based on the height of the character cell being written to the logical page. If the character point size does not change within a page, the number of lines is determined by dividing the logical page height by the current character cell height. However, if the character point size in the last character position on a line has a different character cell height than the initial character point size, the remainder of the logical page is recalculated until another point size is encountered in the last position on a line, or the page is filled.

2. The Line Position Fields

- 3** lines before first line
- 3** lines after last line

These two fields are associated with the **S** Standard VFC method alternate. Each field has a default value of **3** (=3 lines).

The number specified in these fields determines the beginning and ending print positions within the

THE VERTICAL FORMAT CONTROL MENU

printable area on the logical page. The default number sets the beginning position after the third line, and leaves three blank lines at the bottom of the printable area. It is helpful to remember that the printable area on a logical page may not be the same as the size of the logical page.

The number of channels (lines) available for print is reduced by the total number of lines entered in these fields.

You can change the default by entering to begin printing on the first line available and continuing through the last line; OR, you may specify any combination of lines to be skipped before the first line of print and after the last line of print.

3. The VFC File Field

This field is associated with the Special VFC Field and is a required field when using this method.

You enter the name of the VFC file you wish to use in this field. You must use standard MPE File System naming conventions, where the filename may have up to 8 characters, and can be any letter followed by any letter or number. If the file resides in a group other than the sign-on group, you must also specify a groupname and accountname. You may also specify a lockword if this form of security applies to the file.

If you need to use all of these file designation conventions, the field data entry would appear as follows:

VFC File

filename/lockword.groupname.accountname

When you press , the IFS/3000 program checks to see that the filename specified exists, and that it contains a valid VFC format. If the file does not exist as specified, a status message will be issued indicating that:

- The filename does not meet MPE filename conventions.
- The file does not reside in the group and account specified.
- You cannot access the file due to security violations.
- The file does not contain information in the required VFC format.

4. The Top Margin Field

Top Margin
 Units

There are two fields in this group and both are associated with the N - No VFC Field alternate.

The defaults for these fields are (=zero), and (=inches). You can change the defaults by typing your specifications in the fields.

The top margin is specified as a whole number, or as a whole number with a decimal fraction if

THE VERTICAL FORMAT CONTROL MENU

appropriate to the units specified in the associated Units Field.

The number you specify in the Top Margin Field will be calculated in the unit value you specify in the Units Field. The alternates to the Units Field are defined as follows:

I IN

This selection is the default value for this field. The number you specify in the Top Margin Field will be calculated in INCHES.

C CM

This alternate will calculate the number you specify in the Top Margin Field in CENTIMETERS.

M MM

This alternate will calculate the number you specify in the Top Margin Field in MILLIMETERS.

D Dots

This alternate will calculate the number you specify in the Top Margin Field in the number of dots specified. Refer to the operator's manual for the device you specified in the Device Field on the Main Menu to review the number of dots per inch when specifying this alternate.

The Character Font Menu

IFS		Character Font		Environment: SAMPLE		
Character Font:						
Number	00			Orientation		
Name		(user defined)		L - Landscape		
File				P - Portrait		
				RL - Reverse Landscape		
				RP - Reverse Portrait		
	Point size	-OR-		Match size		
	Cell height in dots	-OR-		N - Nearest		
	Height of Capital letters			S - Smaller		
	I Units: I - IN			L - Larger.		
	M - MM					
ASCII Character code: Lowest Highest						

Proportional spacing:	_	Point Size	Ht of Capitals	Cell Width	Cell Height	Baseline
Actual size to be used	_____	_____	_____	_____	_____
Next Larger size available	_____	_____	_____	_____	_____
Next smaller size available	_____	_____	_____	_____	_____
First Ft	Last Ft	Prev Ft	Next Ft	Refresh	Main	Cancel

Figure 3-16. The Character Font Menu

If you enter **C** in the Main Menu Selection field and press **ENTER**, the Character Font Menu is displayed.

You use this menu to identify the character fonts or logos you wish to include in your environment. OR, to review or modify fonts or logos in an existing environment; OR which you copied using the Copy or Initialize Menu.

When you associate your environment with a data file, The Laser Printer activates or deactivates the character fonts you have specified with this menu as instructed by the data file or application program. It is suggested that you review the IDSCHAR Terminology in Section 2. This will be helpful in specifying the data entry fields on this menu. It will also be useful when determining the effect of your environment on your data file at print time.

Function Keys

The function keys used with this menu are:

ENTER

Once you type the Character Font filename and all other required field data, press the **ENTER** key to issue this data to the computer. IFS/3000 will issue status messages to the menu banner if any required data is not specified, or if it has difficulty processing the data. OR, the program will issue a bell to indicate that the field data was acceptable. The menu will remain on the screen for your review. You can use the function keys to proceed directly to another character font. OR, IF you do not type any changes, you can press **ENTER** again, OR press the **Main** function key, to return to the Main Menu.

First Ft

Press **F1** to display the first font defined. IF you have not defined specifications for this character font, the menu will appear blank except for the font **ID-number** and field defaults.

Last Ft

Press **F2** to display the last font defined. IF you have not defined specifications for this character font, the menu will appear blank except for the font **ID-number** and field defaults.

Prev Ft

Press **F3** to display the character font with an **ID-number** ONE LESS than the current character font. IF you have not defined specifications for this character font, the menu will appear blank except for the font **ID-number** and field defaults.

Next Ft

Press **F4** to display the character font with an **ID-number** ONE GREATER than the current character font. IF you have not defined specifications for this character font, the menu will appear blank except for the font **ID-number** and field defaults.

Refresh

Press **F6** to redisplay the terminal screen with any data entered using the **ENTER** key.

Main

Press **F7** to return to the Main Menu. Note that any field data you have typed but have not entered will be lost.

Cancel

Press **F8** to delete any data you have just typed, but have decided not to enter.

Field Discussion

There are nine groups of data entry fields on this menu.

The first field establishes the **ID**-number for the character font specifications you are entering or reviewing on the menu. This **ID**-number remains associated with the specifications unless you change it. It enables you to identify individual character fonts in a data file or application program which can then be activated or deactivated by the Laser Printer at print time.

The second field is used to establish the orientation of the character font in relation to the physical page. You must define a separate character font for each orientation you wish to include in your environment.

The third field enables you to provide an optional font name. This is useful where you prefer to use your own naming conventions by supplying a font name that is more descriptive of your application than the font file name.

The remaining fields are used to specify the file where the character font resides, the size of the character, what alternate sizes are acceptable, and whether you wish to save space in the environment by specifying a range of characters to be included.

The menu is divided into two sections by a dashed ----- line. The protected fields below this line are used by the program to display the font characteristics that the program finds in the file when you press the **(ENTER)** key. Once you have reviewed this information you would re-type any

changes in the data entry fields above the line, and press **(ENTER)**.

You may respecify any data on this menu until the data is acceptable.

The field data and its effect on the environment is defined as follows:

1. The Font Number Field

Character Font:
Number **00**

This is a required field. You can identify up to 32 sets of character font or logo specifications to be included in your environment. Each font or logo **MUST** be defined individually and given an **ID**-number in this field.

A font **ID**-number from **00** to **31** will always appear in this field when the Character Font Menu is displayed. The number that appears depends on the method you use to display the menu as follows:

- IF you DID NOT enter a specific font number in the Font/Page Number, OR Character Font Name Field on the Main Menu; font **00** will be displayed. If you have not defined specifications for this font number, the menu will appear blank except for the **00 ID**-number and field defaults.
- IF you DID enter a specific font number in the Font/Page Number, OR Character Font Name Field on the Main Menu; the font number

THE CHARACTER FONT MENU

entered, or number associated with the name entered will be displayed. If you have not defined specifications for this font number, the menu will appear blank except for the ID-number and field defaults. You can not specify a font name unless you have already defined the font specifications and used the **ENTER** key.

- IF you used a function key on this menu, the font ID-number corresponding with the key label will be displayed. Refer to the function key discussions.

You would change a font ID-number ONLY IF you wish to assign a different number than the number currently assigned on the menu. To change a number, type a number from 00 to 31 over the number in this field before using the **ENTER** key. This makes an exact duplicate so you may want to delete the copied font.

NOTE

IF an existing font specification is similar to a new font you want to include in your environment, you can change the ID-number and font name and use the current size specifications without having to retype them. You would also provide a new filename if the font resides in another file.

CAUTION

HOWEVER, if you assign an ID-number that is currently defined and press **ENTER**, the program will ask you to confirm that you want to overwrite the existing specification. IF you do so, the existing specification will be lost. You should exercise caution when reassigning font ID-numbers using existing specifications.

The font number is used by IFS/3000 to provide access to an individual set of font specifications when working with an environment file. The font number is used by the Laser Printer to locate and activate or deactivate the compiled font. The font number is identified in the data file or application program along with instructions to the Laser Printer as to when the font number is to be activated or deactivated. When a font is active, the data file prints in the active font.

2. The Orientation Field

Orientation

This is a required field. You use this field to specify the orientation of the character font in relation to the physical page. The orientation of the logical page is device dependent, specific to the device you identified in the Device Field on the Main Menu. You must define a separate character font for each orientation of a font you wish to include in your environment.

The alternate character font orientations you may specify in this field are defined as follows:

- L** Landscape
- P** Portrait
- RL** Reverse Landscape
- RP** Reverse Portrait

Refer to Section 2, figure 2-6 for examples of the above orientations.

3. The Font Name Field

Name (user defined)

This is an optional field. You would use this field to provide a descriptive font name. This is useful where you wish to use a name which you feel is easier to remember, or is more descriptive of the font in its current application.

You can use the name you specify here in the Character Font Name Field on the Main Menu to access the specifications you define on this menu.

The Laser Printer does not use the font name when identifying fonts. Any instructions associated with a data file or application program must interpret the font name and provide the font ID-number to the Laser Printer.

The font name may include up to 16 characters, beginning with a letter, followed by any letter or number. A common usage identifying a font in a 12 point size and bold typeface could be entered as follows:

Name (user defined)

4. The Font File Field

File

This is a required field. You use this field to identify the character cell file where the font or logo resides.

You enter the name of the character font file using standard MPE File System naming conventions where the filename may have up to 8 characters, and can be any letter followed by any letter or number. If the file resides in a group other than the sign-on group, you must also specify a groupname and accountname. You may also specify a lockword if this form of security applies to the file.

If you need to use all of these file designation conventions, the field data entry would appear as follows:

When you press **ENTER**, the IFS/3000 program checks to see that the filename specified exists, that it is a character font file, and that the font size defined in the associated font size field resides within the file. The character font definition is then included in the environment file and will be compiled when you perform the compile operation using the Main Menu.

THE CHARACTER FONT MENU

If the file does not exist as specified, a status message will be issued indicating that:

- The filename does not meet MPE filename conventions.
- The file does not reside in the group and account specified.
- You cannot access the file due to security violations.
- The character font size does not exist within the file.
- The font is inconsistent with the device specified in the Device Field on the Main Menu.
- The file specified is not a character font file.

5. The Font Size Fields

	Point size	-OR-
	Cell height in dots	-OR-
	Height of Capital letters	

There are three alternate fields in this group. You **MUST** use **ONE** of the alternates to specify the size of the character font you wish to include in your environment.

A character font file contains all characteristics of a character font, including its cell point size, cell height, and dot equivalents. A font is also device dependent as the dot pattern required to form a

letter depends on the dot density for the printer hardware.

You use **ONE** of these fields to specify the font size in the characteristic you have available.

You would use the associated Match Size Field to indicate the next size which would be acceptable if the actual size specified is not available in the font file. When you press the **(ENTER)** key, the program will issue a bell if the font size requested is available. In addition, all of the characteristics of the font size will be displayed in the protected fields below the dashed ----- line. This information will appear on the line which indicates the actual size to be used.

If the size requested is not available, a status message will be issued and the alternate sizes available in the file will be displayed in the protected fields below the dashed ----- line. The alternate sizes displayed are within the limits of your entry in the Match Size Field. Once you have reviewed this information, you would re-type any changes in the font size field above the line, and press **(ENTER)**.

The alternates for the font size fields are defined as follows:

	Point size
---	------------

The point size is the standard measurement used to designate font sizes and is normally the most frequently used field for this specification.

The point size is determined by the character designer at the time the character font is created using IDSCHAR. OR, the point size is included in the listing of character fonts supplied by Hewlett-Packard in Appendix D. You would type the point size in this field as a whole number. A 12 point font would be specified as follows:

12 Point size

The program will look for the point size in the character font file you identified on this menu.

Cell height in dots

This selection requires that you define the character cell height in dots. Since character fonts are device specific, you must define this field for the device entered in the Device Field on the Main Menu.

The dot height is determined by the character designer at the time the character font is created using IDSCHAR. OR, the dot height is included in the listing of character fonts supplied by Hewlett-Packard in Appendix D. You would type the dot height in this field as a whole number.

To specify the cell height in dots for a 12 point font and a 2680A Device, this field would be defined as follows:

30 Cell height in dots

OR, for the 2688A Device as follows:

50 Cell height in dots

The program will look for the cell height in the character font file you identified on this menu.

Height of Capital Letters

If you do not know the point size or the height of the character cell in dots, you can measure the height of a capital letter and enter it in this field. The height can be expressed as a whole number or whole number and decimal fraction. A capital letter that measures .198 inches would be specified as follows:

.198 Height of Capital letters

The program will calculate this dimension in the units specified in the associated Units Field. It determines the equivalent height in dots for the device specified in the Device Field on the Main Menu. It then looks for the dot height of capitals in the character font file you identified on this menu.

THE CHARACTER FONT MENU

6. The Match Size Field

 Match size

The default for this field is  (=nearest). You can change the default to either of the other alternates by replacing the  with  (=Smaller) OR  (=Larger).

This field establishes what alternative information is to be displayed in the protected fields below the dashed ----- line on the menu. When you specify a font size field and press the **ENTER** key, the program will display the actual font characteristics if there is a match. If it cannot find the exact size specified, it will display the next smaller or next larger font characteristics which also resides in the file. If no alternatives are available, the program will issue a status message.

7. The Units Field

 Units: I - IN
M - MM

This is an optional field UNLESS you have specified the associated Height Of Capital Letters Field. The default for this field is  (=inches). You can change the default  to  (=millimeters).

The unit value you specify in this field is used to calculate the height of the capital letter. The alternates to the Units Field are defined as follows:

 IN

This alternate will calculate the number you specify in INCHES.

 MM

This alternate will calculate the number you specify in MILLIMETERS.

8. The ASCII Character Code Fields

 Lowest  Highest

There are two data entry fields in this group. These are optional fields.

The Lowest and Highest Character Code fields are used to specify the range of characters you want to include in the environment. This enables you to minimize the space required for your environment file, both on disc and in Laser Printer memory.

Each character in a character font has a character code number associated with it. If you do not need all the characters in a given character font, you can enter the range of character codes you want to use in printing your document. Your environment file then includes only the characters with character code numbers in the range you specify.

If you do not type anything in these fields, the system defaults to the entire range of characters between character code numbers 32 and 126 inclusive. Normally the character code number assigned to a character is the ASCII code. In ASCII, character code numbers 32 through 126 include the upper and lower case alphabet, the integers 0 through 9, and several common punctuation symbols. Character number 32 is a space. If the characters in a character font were

THE CHARACTER FONT MENU

not defined as ASCII characters, these character code numbers may result in characters other than those just described.

To include the ASCII characters between 32 and 58, the entry would be as follows:

32 Lowest **58** Highest

If the character font is a logo file, IFS/3000 automatically supplies the number 76 as the value for both the Lowest and Highest Character Code fields. A logo file contains only one character which is always assigned character code number 76, the uppercase letter "L" in ASCII.

9. The Protected Information Fields

There are six protected display fields in this group. Five of the fields provide space for displaying three sets of information.

The information in Section 2 on IDSCHAR terminology will be useful in determining how to apply the data displayed in these fields.

When you type the font filename and font size you wish to locate in the file, and press **ENTER**; IFS/3000 uses these fields to display actual and alternate information. Once you review this information and type any changes in the data entry fields above the dashed ----- line, press **ENTER**. The information displayed here enables you to review the alternates within the file and modify your specification until the font is defined successfully.

An example of the information displayed in these fields follows.

The first information field displays if the character font is proportionally spaced (Y=yes, N=no). The other fields vary in the information they supply.

If you select a point size that does not exist in the character font selected, the information displayed will show you the actual size to be used along with the next larger size and next smaller size available and the baseline of each. The sizes displayed may not be the only ones resident in the cell file. For example, you might select 7 as the point size and, if it does not exist, the information displayed might be as follows:

Proportional spacing: N	Point Size	Ht of Capitals	Cell Width	Cell Height	Cell Baseline
Actual size to be used. . .	6.00000	.055556	7	15	5
Next Larger size available.	8.00000	.066667	10	20	6
Next Smaller size available.	6.00000	.055556	7	15	5

THE CHARACTER FONT MENU

If you change your selection from 7 to 10 and size 10 does not exist, the information displayed might then be as shown below:

Proportional spacing: N	Point Size	Ht of Capitals	Cell Width	Cell Height	Cell Baseline
Actual size to be used. . .	8.00000	.066667	10	20	6
Next Larger size available.	12.00000	.094444	15	30	10
Next Smaller size available.	8.00000	.066667	10	20	6

Lastly, if you select a point size that does exist in the character font selected, the information displayed would show only the actual size to be used.

Proportional spacing: N	Point Size	Ht of Capitals	Cell Width	Cell Height	Cell Baseline
Actual size to be used. . .	12.00000	.094444	15	30	10
Next Larger size available.					
Next Smaller size available.					

For proportionally spaced characters, IFS/3000 displays the maximum cell width. Without proportional spacing, the character cell width is the same for each character in a given character font. If you know the cell width and the page size, you can calculate the number of characters that can be printed in a line on the page.

The cell width does not include any inter-character spacing which may have been specified when the cell was defined using IDSCHAR. Refer to the *IDSCHAR Reference Guide (Part Number 36581- 90001)*.

The cell height does not include any inter-line spacing which may have been specified when the cell was defined using IDSCHAR. Refer to the *IDSCHAR Reference Guide*.

The baseline is a line used for reference purposes when designing characters; it is not an actual line to be printed along with the character. A character or logo sits on its baseline. When characters are printed, they are aligned according to the baseline. All capital and lower case letters, except for the descender, should sit on the baseline. The character designer specifies the location of the baseline during the character design process.

The Copy Menu

IFS	Copy	Environment: SAMPLE
Environment file to copy from		
Parts to Copy		
C	- Character fonts, -OR-	Number -OR- Name
L	- Logical pages, -OR-	Number
P	- Physical page and environment defaults	
E	- Entire environment	
Method of Copy		
O	- Overwrite ... Copies all items, deleting the existing values.	
R	- Replace Copies all items found in both environments.	
A	- Add Copies all items, renumbering if necessary.	
F	- Fill Copies only those items not already in this environment.	
Copy associated compiled parts? Y/N		
		Refresh Main Cancel

Figure 3-17. The Copy Menu

If you enter **CO** in the Main Menu Selection field, the Copy Menu is displayed. You use this menu to copy all or part of an another environment file into the environment you are working on. An environment file contains two types of data:

- The physical page, logical page(s), and character font(s) specifications.
- The compiled version of these specifications.

The Copy Menu enables you to copy an environment specification with or without the corresponding compiled part. It also enables you to identify a single specification to be copied.

Any specifications and compiled parts which are added to your environment using the Copy Menu will be detected as a change by the IFS/3000 program. When you specify the compile function on the Main Menu, they will automatically be recompiled.

Function Keys

The function keys used with this menu are:

ENTER

Once you have typed the environment file name and the associated data in the menu fields, press the **ENTER** key to issue the field data to the computer. IFS/3000 will issue status messages to the menu banner if it has difficulty accessing or copying the specified environment. The program will issue a bell if the operation is successful. The menu will remain displayed for you to copy additional parts from the designated file or from another environment file. If you do not specify any new data, you can press **ENTER** again, OR press the **Main** function key to return to the Main Menu.

Refresh

Press **F6** to redisplay the terminal screen with any data entered using the **ENTER** key. It is useful when you have used the **Cancel** key.

Main

Press **F7** to return to the Main Menu. Note that any field data you have typed but have not entered will be lost.

Cancel

Press **F8** to delete any data you have just typed, but have decided not to enter.

Field Discussion

There are four groups of data entry fields on the Copy Menu.

You use these fields to specify the environment file you wish to copy from, the parts you wish to copy, the method you want used when the part is copied, and to indicate whether or not the associated compiled part is to be copied along with the specifications.

You may respecify any field data which results in a warning or error message, until the copy operation is successful.

The field data and its effect on the environment is defined as follows:

1. The Environment File to Copy From Field

Environment file to copy from

This is a required field. You must also specify the Parts to Copy field before you press **ENTER**. You do not have to change the default values for the Method of Copy or Copy Associated Parts fields unless you wish to choose an alternate selection.

You enter the name of the environment file you wish to copy from in this field. You must use standard MPE File System naming conventions, where the filename may have up to 8 characters, and can be any letter followed by any letter or

number. If the file resides in a group other than the sign-on group, you must also specify a groupname and accountname. You may also specify a lockword if this form of security applies to the file.

If you need to use all of these file designation conventions, the field data entry would appear as follows:

Environment file to copy from
 filename/lockword.groupname.accountname

When you press **ENTER**, the IFS/3000 program checks to see that the filename specified exists, that it is an environment file, and that the part to be copied is available. If the file does not exist as specified, a status message will be issued indicating that:

- The filename does not meet MPE filename conventions.
- The file does not reside in the group and account specified.
- You cannot access the file due to security violations.
- The part to be copied does not exist as specified.
- The file specified is not an environment file.

2. The Parts To Copy Fields

Parts to Copy

-OR- Number

-OR- Name

There are three data entry fields in this group. The first field is used to select the parts you wish to copy. This selection will determine if you can use the other two fields to specify a logical page or character font ID-number, or alternate character font name. These three fields enable you to specify exactly what you want to copy into your current environment.

Your alternate selections for these fields are defined as follows:

C all Character fonts,

-OR- ID Number

-OR- userfontname Name

This selection enables you to copy ALL character fonts in the specified environment: OR, also specify a single character font ID-number or name in the associated fields.

If you wish to copy a number of individual character fonts, you would need to repeat the **C** copy procedure for each character font, using the Number or Name fields. OR, you could copy all fonts, and then use the Main Menu to delete the ones you did not want to include in the current environment.

When this copy operation is complete, you

THE COPY MENU

would use the Character Font Menu to display or modify the copied specifications.

L all Logical pages -OR- ID Number

This selection enables you to copy ALL logical pages in the specified environment: OR, also specify a single logical page ID-number in the associated field.

If you wish to copy a number of individual logical pages, you would need to repeat the **L** copy procedure for each logical page using the Number field. OR, you could copy all logical pages, and then use the Main Menu to delete the ones you did not want to include in the current environment.

When this copy operation is complete, you would use the Logical Page Menu to display or modify the copied specifications.

P Physical page and environment defaults

This selection enables you to copy the physical page specifications in the specified environment. Since the physical page defines all other environment defaults, this selection will establish the current environment's defaults.

In addition, any multi-copy forms associated with the physical page will be copied.

When this copy operation is complete, if you display the Physical Page Menu, the number of physical page copies, multi-copy forms,

and primary and secondary character font specifications will have the same values as those in the environment you copied.

Any time you add or modify the physical page specifications, IFS/3000 will notice the change and automatically require you to compile or recompile all other specifications in the environment.

E Entire environment

This selection enables you to copy all specifications in the specified environment. If you also specify that the copy method is to overwrite any existing specifications, the effect will be the same as if you had used the Initialize Menu.

3. The Method Of Copy Field

R Method of Copy

The default for this field is R (=Replace). You do not need to change this default unless you wish to select another method of copy.

You use this field to select the procedure to be used when the program copies the designated parts. In order to select an alternate method, change the R (=Replace) to the selection desired.

Your alternate selections for this field are defined as follows:

O Overwrite

This selection will replace any existing specifications for the same part with the copied specifications. The part or parts replaced are those identified in the Parts to Copy fields.

WARNING

Be careful when selecting the parts to be copied. IF you do not specify a single font or logical page, AND you also select this method of copying, any existing font or logical page specifications which do not also exist in the designated environment will be deleted from the current environment. The results will be that the current environment will ONLY contain specifications which

were present in the environment you copied.

R Replace

This is the default value for this field. This selection will replace corresponding specifications in the current environment with the copied specifications. The parts replaced are those identified in the Parts to Copy fields. If you have specified that all parts be copied using this method, any current part that does not have a corresponding part to be copied WILL NOT be deleted. Once you replace a part, you cannot retrieve the previous data.

A Add

This selection will add the specified part to the current environment. If the logical page or character font ID-number or Name already is assigned in the current environment, the copied part will be renumbered, or its name will be deleted. This method has no effect on any existing parts, their ID-numbers, or Names.

Since you may not have more than one physical page defined for an environment, the add selection will ONLY be allowed for the physical page IF it is not defined in the current environment.

F Fill

This selection would be used if you wish to copy all specifications in the specified file which do not exist in the current

THE COPY MENU

environment, for the parts selected in the Parts to Copy field.

Figure 3-18 shows the results of each of the above selections. The final environment will appear as shown in the columns under the function selected.

4. The Copy Associated Compiled Parts? Field

Copy associated compiled parts? Y/N

The default for this field is (=no). You can specify that you want to copy the compiled part along with its specifications by changing (=no) to (=yes).

When you specify (=yes) in this field, the method you specify in the Method of Copy field has the same effect on the existing compiled parts as it does on the existing specifications.

IFS/3000 COPY FUNCTIONS					
		DESTINATION AFTER COPY FUNCTION			
SOURCE ENVIRONMENT	DESTINATION ENVIRONMENT	OVERWRITE	REPLACE	ADD	FILL
LP0 SOURCE	LP0 DEST.	LP0=0 SOURCE	LP0=0 SOURCE	LP0=0 DEST.	LP0=0 DEST.
LP1 SOURCE	LP1 ----	LP1=1 SOURCE	LP1=1 SOURCE	LP1=0 SOURCE	LP1=1 SOURCE
LP2 SOURCE	LP2 ----	LP2=2 SOURCE	LP2=2 SOURCE	LP2=1 SOURCE	LP2=2 SOURCE
LP3 SOURCE	LP3 ----	LP3=3 SOURCE	LP3=3 SOURCE	LP3=2 SOURCE	LP3=3 SOURCE
	LP4 DEST.		LP4=4 DEST.	LP4=4 DEST.	LP4=4 DEST.
				LP5=3 SOURCE	

IFS looks to see if location is filled, it DOES NOT KNOW what is there.

Figure 3-18. Copy Field Selection Results

Laser Printing System Interpreter (LPS Interpreter) SECTION 4

Overview

The LPS Interpreter, included with the Printer Graphics Package (HP 36583), provides a non-programmatic interface to the IFS/3000 intrinsics and gives you access to the features of the Laser Printer via commands embedded in text files.

Text files consist of text to be printed and formatting commands. These commands allow you to control such things as the character font or form being printed, or the location and direction of

of the text. You can also print data into a form using named fields and, if you have the HP graphics software, you can merge graphics output with your text at print time.

The LPS Interpreter sends environment files and text files to the Laser Printer and executes formatting commands by calling the IFS/3000 programmatic intrinsics. It is not necessary to have knowledge of a programming language to use the LPS Interpreter.

Figure 4-1 provides an illustration of the LPS Interpreter in the HP 3000 environment.

LPS INTERPRETER

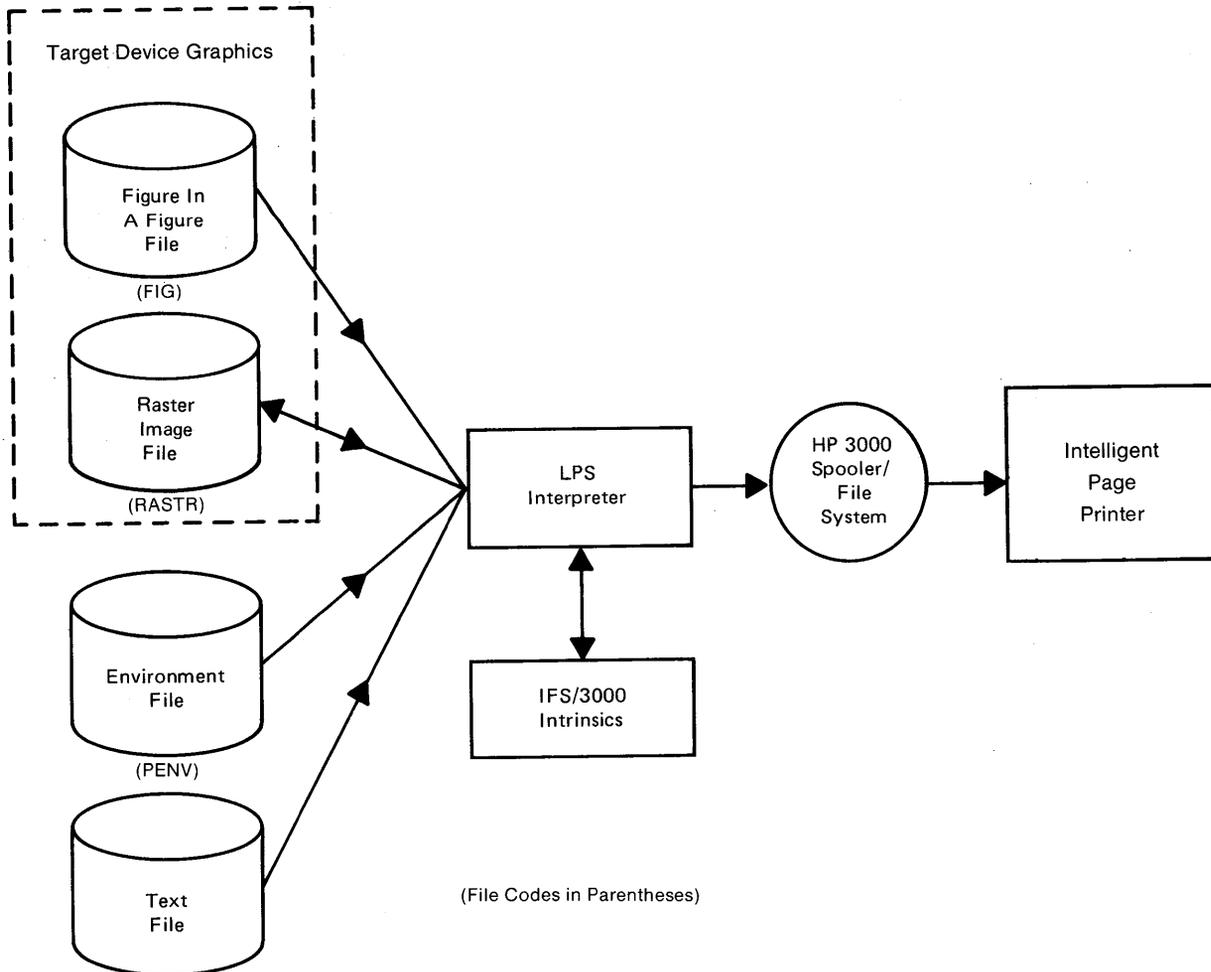


Figure 4-1. Overview of the LPS Interpreter

Intrinsic Calls

The LPS Interpreter automatically calls the MPE FOPEN intrinsic and the IFS/3000 PINITDEVICE and PERRMSG intrinsics. Also, in contrast to the programmatic intrinsics, it is not necessary to supply a *comarea* parameter with any of the commands or a *datalen* parameter with the .WRITEFIELD command. Data length is determined by the length of the text itself.

With the release of LPS version A.01.07, there is no longer a fixed limit on the environment size which may be run with LPS. However, if you use the LPS commands .CONVERTRASTER, .CONVERTFIGURE, or .PRINTFIGURE, you should have an environment with a *comarea* length less than 3000 words. Otherwise there may not be enough system stack area to manipulate the desired image. The Interpreter issues a warning to this effect if a larger environment file is specified (see "Running the LPS Interpreter", later in this section).

Section 5 of this manual describes the IFS/3000 programmatic intrinsics.

Text Files

Your text file can be an ASCII file created with any HP 3000 text editing subsystem. All LPS Interpreter commands are used in the same way, regardless of the text editor you choose.

CAUTION

Text files must be kept unnumbered for use with the LPS Interpreter. For information on creating a text file, refer to the Reference Manual for the text editor you are using.

LPS Interpreter text files consist of text to be printed, formatting commands, and control characters which are used to distinguish commands from the text to be printed.

Text

Text is whatever you want to appear in your printed document. Anything that is not preceded by a control character is recognized as text to be printed. Text can also be printed to a named field on a form with the .WRITEFIELD command.

Commands

Formatting commands are used to give the Laser Printer directions on how you want your document printed. All commands are preceded by a command control character and most are followed by one or more parameters. The default control character is a period (.)

LPS INTERPRETER

If there is more than one parameter, they should be separated by:

- one or more spaces, or
- a comma with no spaces, or
- a comma and one or more spaces.

For example:

```
.COMMAND parameter1 parameter2
.COMMAND parameter1,parameter2
.COMMAND parameter1, parameter 2
```

Descriptions of all commands appear later in this section. To change the default command control character, refer to the description of the .CONTROLCHAR command.

Continuation Lines

You can continue a command on to the next line by putting a continuation character at the end of the first line of the command. The continuation character must be preceded by a blank and must be the last character on the line. The default continuation character is an ampersand (&). For example:

```
.COMMAND parameter1 parameter2 &
      parameter3 parameter4 parameter5
```

To change the default continuation character, refer to the description of the .CONTINUECHAR command.

Comments

A comment character is placed before each comment line. Comments serve as notes and comments to you when you are creating or reviewing a text file. The default comment character is an exclamation point (!). For example:

```
!This is a comment line
```

To change the default comment character, refer to the description of the .COMMENTCHAR command.

Default Parameters

You can use an asterisk (*) in place of a command parameter to select the default. For example:

```
.COMMAND parameter1 * * parameter4
```

To change the default character, refer to the description of the .DEFAULTCHAR command.

Supplied defaults are:

.COMMENTCHAR	!
.CONTINUECHAR	&
.CONTROLCHAR	.
.DEFAULTCHAR	*
.SHIFTCHAR	
<i>figure rotation</i>	0 degrees
<i>raster image type</i>	0 (temporary)
<i>position mode</i>	0 (relative)
<i>units</i>	0 (dots)
<i>xposition</i>	0
<i>yposition</i>	0
<i>subfield number</i>	Only subfield
<i>carriage control</i>	NOCCTL

Shift Character

A shift character is used to change from primary to secondary character font, or vice versa. The default shift character is a vertical bar (|). For example:

```
This line |will print in two| typefaces
```

To change the default shift character, refer to the description of the .SHIFTCHAR command.

Output File

The formal file designator for the LPS Interpreter output spool file is LPSOUT. The Laser Printer is automatically used as the output device (device class PP or PP88) unless you specify otherwise. To do this, you must set up a file equation to direct the file LPSOUT to another device or device class. For example:

```
:FILE LPSOUT;DEV=device;ENV=envfile
```

The device class can be specified by device class name or logical device number.

If you wish to direct your output to a device class other than PP (2680A) or PP88 (2688A), you must set up this file equation prior to running the LPS Interpreter.

Be sure to specify your environment file name in the file equation. When the LPSOUT file is redirected, the environment file name you give to the LPS Interpreter **Environment file?** prompt is ignored.

Carriage Control

Carriage control in a data file is recognized by using the INCLUDE command with the CCTL *parameter*.

Running the LPS Interpreter

After you have successfully logged on and received the MPE colon prompt (:), type the command:

```
:RUN LPS.PUB.SYS  
and press RETURN.
```

The first message you see will display the version of the LPS Interpreter available on your system followed by the version of the IFS Intrinsic as follows:

```
LPS Interpreter (A.01.07) HP36580 (c) COPYRIGHT Hewlett-Packard Co. 1982  
IFS/3000 Intrinsic (A.02.03)
```

The LPS Interpreter will then prompt for the following information or exit instructions:

What device are you printing to (2680A or 2688A)?

Environment file (or 'Exit' to leave)?

Environment size is XXXXX words.

Your output is being spooled to 'LPSOUT'.

If your environment size is greater than 3000 words, the LPS Interpreter will display:

```
WARNING: Your environment file has a length greater than 3000 words. If you use  
the LPS commands .convertraster, .convertfigure, or .printfigure the program may  
abort if there is not enough area left on the system stack to manipulate the  
desired image.
```

If your environment size is 3000 words or less, this warning will not be issued. In either case, the LPS Interpreter continues as follows (giving you the option to exit):

Interactive (Y/N/Exit)?

For help type '.help'.

>

Capabilities

Users of some commands in the LPS Interpreter will need to have DS (extra Data Segments) capability. Check with your system manager if you have any questions about your user capabilities.

Output Device

Type the name of the output device you wish to print to at the **Output Device?** (e.g., 2680A or 2688A) prompt and press **RETURN**.

If you press **RETURN** without specifying an output device, the request will be repeated until you specify a device.

Environment File

The LPS Interpreter sends your text file and environment file to the Laser Printer to print your formatted document. If you do not want to use one of the environments supplied with IFS/3000, you must create your own environment file before printing your document.

Type the name of your environment file beside the **Environment File?** prompt and press **RETURN**.

If you press **RETURN** without specifying an environment, the Laser Printer will default to the standard line printer environment for the output device specified at the **Output Device?** prompt (LP.HPENV.SYS or LP88.HPENV.SYS).

IFS/3000 checks to verify that the filename you supply is an existing environment file. If it is, IFS displays the file size. If it is not a valid environment file, you receive an error message and are given the option of exiting or specifying another environment file name.

Sessions

When your environment filename has been accepted, you are asked if you want to work in an interactive session. The LPS Interpreter can be used in two ways:

- you may enter formatting commands and text interactively.
- you may supply the name of a formatted text file to be printed.

Interactive

Type Y beside the **Interactive?** prompt and press **RETURN**.

You have now begun an interactive session and the > prompt indicates that you can enter the text and formatting commands for the document you want to print.

NOTE

It is important to note that the LPS Interpreter is not a text editor. Text is spooled to the Laser Printer as each line is entered, making it impossible to go back and make changes to text lines. You would normally work interactively if you wanted to create a test file of commands for a skeleton document, rather than to build a complete document.

When you have entered all your text and formatting commands, type the command:

.EXIT

This closes the spoolfile making it ready to print your document and returns you to the MPE Operating System.

If you want to print another file, you must run the LPS Interpreter again.

Non-Interactive

Type N beside the **Interactive?** prompt and press **(RETURN)**.

You are now asked for the name of your input file. Type the name of your formatted text file beside the **Input File?** prompt and press **(RETURN)**.

The message:

Reading input file ...

is displayed to tell you that your text file is being processed and printed. After your file has been processed to the spooler, you are returned to the MPE Operating System.

If you want to print another file, you must run the LPS Interpreter again.

Batch Job

You can also run the LPS Interpreter by creating a stream file to be run as a batch job. If you use this method, the commands in your stream file must exactly correspond to the LPS Interpreter prompts. A sample stream file is given in the examples at the end of this section.

Ending A Session

When you want to end an LPS Interpreter session, type the command:

.EXIT

Your spoolfile is then closed and is ready for printing, even if you exit at a time when no text has been entered. You are returned to the MPE Operating System and the following message is displayed:

END OF PROGRAM

:

Terminology

Before using the commands described in this section, be sure you are familiar with the terminology explained in Section 2.

LPS Interpreter Commands

The LPS Interpreter commands are divided into two groups. Each group requires different software subsystem capabilities as follows:

- LPS Formatting commands can be used if you have IFS/3000 installed on your system.
- Graphics commands are available with the HP graphics software.

Figure 4-2 summarizes the formatting and graphics commands. A full description of each command is listed alphabetically, following the summary.

The HP non-graphics software includes all LPS Interpreter commands.

Abbreviations

In the descriptions that follow, all commands may be abbreviated as indicated by the square brackets []; that is, letters that appear in the brackets are optional. For example,

.U[SEFONT]

may be specified in any of the following ways:

.USEFONT
 .USEFON
 .USEFO
 .USEF
 .USE
 .US
 .U

THIS PAGE SHOULD BE BLANK

LPS Interpreter Commands

INTRINSICS	FUNCTION
<p>.ACTIVATEPAGE .COMMENTCHAR .CONTINUECHAR .CONTROLCHAR .CONVERTFIGURE .CONVERTRASTER</p>	<p>Activates a logical page. Changes the default comment character. Changes the default continuation character. Changes the default control character. Converts a figure in a figure file to a raster image file. Accepts dot-per-bit raster data graphics. The basic image can be manipulated, including rotation and scaling, and a formatted raster image file is created which is printable on the Laser Printer.</p>
<p>.DEACTIVATEPAGE .DEFAULTCHAR .DELETERASTER .ENDWRITEFIELD .EXIT</p>	<p>Deactivates a logical page. Changes the default parameter character. Deletes a raster image from Laser Printer memory. Stops printing text into a field on a form. Leaves the LPS Interpreter and returns you to the MPE Operating System.</p>
<p>.FLASHRASTER</p>	<p>Loads a raster image into Laser Printer memory, prints it, then deletes it from Laser Printer memory.</p>
<p>.HELP</p>	<p>Provides help on LPS Interpreter commands.</p>
<p>.INCLUDE</p>	<p>Allows you to include text and commands from another text file.</p>
<p>.LOADRASTER</p>	<p>Loads a raster image into Laser Printer memory.</p>
<p>.MOVEPENABS</p>	<p>Moves the Laser Printer "pen" relative to the upper left corner of the logical page.</p>

Figure 4-2. LPS Interpreter Commands

LPS Interpreter Commands

INTRINSICS	FUNCTION
<p>.MOVEPENREL</p> <p>.NEWFORM</p> <p>.NEWPAGE</p> <p>.NEWPHYSPAGE</p> <p>.NEWSUBFORM</p> <p>.PRINTFIGURE</p> <p>.PRINTRASTER</p> <p>.SELECTPAGE</p> <p>.SHIFTCHAR</p> <p>.USEFONT</p> <p>.WRITEFIELD</p>	<p>Moves the Laser Printer "pen" relative to its current position.</p> <p>Selects the form you want to print to.</p> <p>Advances to the next active logical page.</p> <p>Advances to the next physical page.</p> <p>Selects the subform you want to print to.</p> <p>Converts a figure in a figure file to a raster image file. then loads and prints the raster image and deletes it from the printer memory.</p> <p>Print a raster image loaded via LOADRASTER command.</p> <p>Activates and advances to the selected logical page. Optionally performs a physical page eject.</p> <p>Changes the default shift character.</p> <p>Selects primary and secondary character fonts</p> <p>Prints text in a field or subfield on a form.</p>

Figure 4-2. LPS Interpreter Commands (con't.)

.ACTIVATEPAGE

Activates a logical page.

Syntax

```
.A[CTIVATEPAGE] logicalpagenumber
```

Parameters

logicalpagenumber The page number of the logical page you want to activate; must be between 0 and 31, inclusive, and must correspond to a logical page in the environment you are using to print your document.

Discussion

The Laser Printer manages logical pages by number and determines which page to print next by cycling through the logical page numbers of active pages. If a logical page is not currently active, it is skipped and is not printed. .ACTIVATEPAGE is used to make an inactive logical page active so that it will be printed. Once activated, the logical page will continue to print until deactivated with the .DEACTIVATEPAGE command. There must be at least one active logical page in a job when attempting to print to the printer.

Example

```
.A 1
```

Logical page 1 was defined as initially inactive in the environment file. When you are ready to print the page, this command makes it available for printing. If the page was initially active in the environment file it would not be necessary to use the .ACTIVATEPAGE command unless the .DEACTIVATEPAGE command had been used previously to make it inactive.

.COMMENTCHAR

Changes the default comment character.

Syntax

```
.COM[MENTCHAR] character
```

Parameters

character

The character you want to use to indicate a comment line in your text. You can use any character except an ampersand (&), a comma (,), a space, or any character currently defined as the control or continuation character.

Discussion

You can include comment lines in your text file to serve as notes or comments to yourself when you are entering text. These comment lines are not printed with your document. Each comment line must be preceded by a comment character to distinguish it from text that is to be printed. The default comment character is an exclamation point (!). If you do not want this to be your comment character, use the .COMMENTCHAR command to change it to the character of your choice.

Example

```
!The next command will change the comment character  
.COM $  
$This is a comment
```

The comment character is now a dollar sign (\$). All comment lines you enter must now be preceded by this character.

.CONTINUECHAR

Changes the default continuation character.

Syntax

```
.CONTI[NUECHAR] character
```

Parameters

character

The character you want to use to indicate a continuation line in your text file. You can use any character except a comma (,), a space, or any character currently defined as the control or comment character.

Discussion

You can continue a command on to the next line by placing a continuation character at the end of the first line of text. This character must be preceded by a blank and must be the last character on the line. The default continuation character is an ampersand (&). If you do not want this to be your continuation character, use the .CONTINUECHAR command to change it to the character of your choice.

Example

```
.CONTI +
```

The continuation character is now a plus sign (+). When you want to continue a command on to the next line, the first line of the command must now end with a blank followed by a plus sign.

.CONTROLCHAR

Changes the command control character.

Syntax

```
.CONTR[OLCHAR] character
```

Parameters

character

The character you want to use to indicate a command line in your text. You can use any character except an ampersand (&), a comma (,), a space, or any character currently defined as the comment or continuation character.

Discussion

Each formatting command must be preceded by a control character to distinguish it from text to be printed. The default control character is a period (.). If you do not want this to be your control character, use the .CONTROLCHAR command to change it to the character of your choice.

Example

```
.CONTR ;  
;ACTIVATE 3
```

The control character becomes a semicolon (;). All formatting commands must now be preceded by this character.

.CONVERTFIGURE

Converts a figure in a figure file into a partitioned raster image file which is the format required for printing by the Laser Printer.

Syntax

```
.CONVERTF[IGURE] figurefilename figurename rasterfilename outputdevice  
imageheight units imagerotation
```

Parameters

- figurefilename* The name of the figure file that contains the figure you want to convert.
- figurename* The name of the figure you want to convert.
- rasterfilename* The name of the partitioned raster image file that will contain the converted figure.
- outputdevice* The name of the output device (2680A or 2688A) that will print the figure. This name can be up to six characters long.
- imageheight* The height of the partitioned raster image viewed in the direction of its orientation and given in the units specified in the *units* parameter. The width of the partitioned raster image is automatically calculated from the height you specify. Refer to Section 2 on Terminology for more information on partitioned raster image height and orientation. One dot is added to the height you specify when the figure is converted; that is, a specified height of 300 dots creates a partitioned raster image 301 dots high. You only need to take this into account if you are working with extremely precise dot measurements.

.CONVERTFIGURE

units The unit of measurement used for the partitioned raster image height. Must be 0, 1, 2, or 3.

0 = dots
1 = inches
2 = centimeters
3 = millimeters

You can specify the default unit of measurement (dots) by using the default character in place of this parameter.

imagerotation The orientation of the printed image with respect to the physical page. Must be either 0, 90, 180 or 270. You can specify the default rotation (0 degrees) by using the default character in place of this parameter. Refer to Section 2 on Terminology for more information on partitioned raster image rotation.

Discussion

.CONVERTFIGURE enables you to convert a figure in a figure file into a partitioned raster image file. A partitioned raster image file is the format required for printing by the Laser Printer. The *rasterfilename* you assign here is also used with the .LOADRASTER or .FLASHRASTER command.

Example

```
.CONVERTF FIGFILE FIG RASTFILE 2680A 6 1 90
```

The figure, FIG, which is stored in the file, FIGFILE, is converted for HP 2680A Laser Printer output. The converted file is called RASTFILE and is an image 6 inches high, at a 90 degree rotation.

.CONVERTRASTER

Accepts dot-per-bit raster data graphics. The basic image can be manipulated, including rotation and scaling, and a formatted partitioned raster image file is created which is printable on the Laser Printer.

Syntax

```
.CONVERTR[ASTER] inrasterfilename outrasterfilename outputdevice  
imageheight imagewidth units imagerotation  
scalingmethod thresholdvalue inverseoption
```

Parameters

- inrasterfilename* The name of the dot-per-bit raster file that you want to convert. This name can be up to 35 characters (upper or lower case) beginning with a letter, and cannot include any embedded blanks.
- outrasterfilename* The name of the partitioned raster image file that will contain the partitioned raster image. This array can be up to 35 characters to allow for a fully qualified MPE file name.
- outputdevice* The name of the output device for which the figure is to be converted. This name can be up to six characters long.
- imageheight* The height of the partitioned raster image, viewed in the direction of its orientation, prior to rotation, and given in the units specified in the first word of the *units* parameter. Refer to Section 2 on Terminology for more information on partitioned raster image height and orientation.
- imagewidth* The width of the partitioned raster image, prior to rotation, given in the units specified in the first word of the *units* parameter.

.CONVERTRASTER

Note: Both imageheight and imagewidth are required parameters and the ratio aspects must be calculated for each.

units

The unit of measurement used for the partitioned raster image height and width. Must be 0, 1, 2 or 3.

- 0 = dots
- 1 = inches
- 2 = centimeters
- 3 = millimeters

You can specify the default unit of measurement (dots) by using the default character in place of this parameter.

imagerotation

The orientation of the printed image with respect to the physical page. Must be either 0, 90, 180 or 270. You can specify the default rotation (0 degrees) by using the default character in place of this parameter. Refer to Section 2 on Terminology for more information on partitioned raster image rotation.

The temporary file, ROTATEFL, is created which can be saved and used separately. (ROTATEFL is a dot-per-bit raster file.)

scalingmethod

Number used to specify the scaling method which will be used to scale the raster data. For any one of the following integers specified, the temporary file, SCALEFL, is created which can be saved and used separately. (SCALEFL is a dot-per-bit raster file.)

- 0 = Threshold testing only. This method uses the input *thresholdvalue*, or calculates a default threshold which serves as a scaled dot comparison value. This method has the best general performance and does well at scaling line art graphics.
- 1 = Pattern recognition. This method attempts to duplicate the pattern of the source image by repeating the pattern in the scaled image. It is more CPU intensive but should give better results in scaled patterns and text.

.CONVERTRASTER

- 2 = Randomized threshold testing. This method is best used in grey tone picture scaling where there are few defined hard lines. This method is CPU intensive. The *thresholdvalue* can be used to generally increase or decrease the lightness of the scaled image.

thresholdvalue Number used to specify a threshold between 1 and 255 which is used in the scaling routines. If the value is zero the scaling routine will calculate a default threshold based on the scaling factor. This value is passed only if *scalingmethod* = 0 or 2. By increasing the threshold the scaled image becomes generally lighter.

inverseoption Number used to specify inverse image as = 1, or normal (non-inverse) image as = 0. The temporary file, INVERSFL, is created and can be saved and used separately. (INVERSFL is a dot-per-bit raster file.)

Discussion

.CONVERTR[ASTER] accepts a dot-per-bit raster file. The basic image can be manipulated, including rotation and scaling, and a partitioned raster image file is created which is printable on the Laser Printer. The *outrasterfilename* you specify here is used later with the .LOADRASTER or .FLASHRASTER commands.

The requirements for the input raster data are:

1. The existing partitioned raster file must be a permanent disc file with fixed record length, where each logical record represents a left to right raster scan line.
2. The raster data records must be binary with one bit-per-pixel where there is a one-to-one correspondence between each bit turned on and each dot of the printed scan line.
3. Each scan line (record) can be a maximum of 124 words (1980 dots) in length, and there cannot be more than 3060 scan lines (records) in total.

The command performs as many as four separate functions depending on the values assigned in the parameter

.CONVERTRASTER

list. Each function creates a separate temporary file of raster data which can be saved and used separately. These temporary files are created only if the related function is requested through the parameter list.

Efficiency is increased by determining which function will create a basic temporary file suitable for multiple format manipulation without repeating the basic function. As an example, scaling a basic image before multiple rotations of the image could be more efficient than rescaling multiple rotations.

Example

```
.CONVERTR EXTERNFL NEWFL 2680A 4.5 3.75 1 0 1 128 0
```

.DEACTIVATEPAGE

Deactivates a logical page.

Syntax

```
.DEA[CTIVATEPAGE] logicalpagenumber
```

Parameters

logicalpagenumber

The page number of the logical page you want to deactivate; must be between 0 and 31, inclusive, and must correspond to a logical page in the environment you are using to print your document.

Discussion

The Laser Printer manages logical pages by number and determines which page to print next by cycling through the logical page numbers of active pages. If a logical page is not currently active, it is skipped and is not printed. This command is used to make an active logical page inactive so that it is not printed.

There must be at least one logical page active in a job when attempting to print to the printer. If you try to deactivate the only active page, you will receive a warning message.

Example

```
.DEA 0
```

Logical page 0 was defined as initially active in the environment file. If you no longer want to print to this page, this command makes it unavailable for printing.

.DEFAULTCHAR

Changes the default character used in command parameters.

Syntax

```
.DEF[AULTCHAR] character
```

Parameters

character

The character you want to use as the default character. You can use any character except an ampersand (&), a comma (,) or a space.

Discussion

When entering formatting commands, you can specify a default parameter by using a default character in place of the parameter. The default character is an asterisk (*). If you do not want this to be your default character, use the .DEFAULTCHAR command to change it to the the character of your choice. Supplied defaults are listed in the discussion on Default Parameters earlier in this section.

Example

```
.DEF =  
.CONVERTFIGURE FIGFILE FIG RASTFILE = 540 = =
```

The default character becomes an equal sign (=). The three default parameters in the above command are now indicated by this character.

.DELETERASTER

Deletes a partitioned raster image from Laser Printer memory.

Syntax

```
.DEL[ETERASTER] imagenumber
```

Parameters

imagenumber

The number of the partitioned raster image to be deleted. Must be between 0 and 31 inclusive and must be the number of a partitioned raster image that has already been loaded with the .LOADRASTER command.

Discussion

A partitioned raster image can be deleted from Laser Printer memory only if it has already been loaded with the .LOADRASTER command. You would normally use the .DELETERASTER command after you had finished printing a raster image with the .PRINTRASTER command.

Example

```
.DEL 5
```

Raster image number 5, which you have already loaded with the .LOADRASTER command, is deleted from Laser Printer memory.

.ENDWRITEFIELD

Stops printing text to a field on a form.

Syntax

```
.EN[DWRITEFIELD]
```

Parameters

NONE

Discussion

The .WRITEFIELD command is used to print text to a named field on a form. Text will be printed even if it extends beyond the end of the field. The .WRITEFIELD command moves the laser printer "pen" to the baseline at the end of the text that has been printed on the field. When you want to stop printing in the field, you can leave it with the .ENDWRITEFIELD command. This command is optional in most cases, since the LPS Interpreter stops printing to a field as soon as it encounters any other command in your text file. The .ENDWRITEFIELD command is required only when there is no other command present to distinguish the end of field text from the beginning of text to be printed outside the field. This usually occurs after printing to the last field on a form.

.ENDWRITEFIELD

Example

```
.WRITEFIELD SUBJECT 1
Quarterly Report
.EN
Attached is the Quarterly Report.
```

Prints the line "Quarterly Report" to subfield 1 of the field SUBJECT. Prints the line "Attached is the Quarterly Report" outside the field as data on the logical page. The .ENDWRITEFIELD command is required since there is no other command present to distinguish this line from the field text.

```
.WRITEFIELD NAME *
Mr. Jim Small
.WRITEFIELD ADDRESS 1
3410 Central Expressway
```

In this case, there is no need to use the .ENDWRITEFIELD command after printing to the field NAME. The second .WRITEFIELD command automatically takes you out of the current field.

.EXIT

Leaves the LPS Interpreter and returns you to the MPE Operating System.

Syntax

```
.EX[IT]
```

Parameters

NONE

Discussion

The .EXIT command is used to leave the LPS Interpreter and return you to the MPE Operating System from the LPS prompt (>). If you decide to exit before printing a file, you can use the EXIT command without the control character (.) after the **Environment File?**, **Interactive?**, or **Input File?** prompts. Using the .EXIT command after the > prompt in an interactive session automatically prints any text you have entered before returning you to the MPE Operating System.

Example

```
.EX
```

You are returned to the MPE Operating System and the following message is displayed:

```
END OF PROGRAM  
:
```

.FLASHRASTER

Loads a partitioned raster image into Laser Printer memory, prints it, then deletes it from Laser Printer memory.

Syntax

```
.F[LASHRASTER] rasterfilename xposition yposition units positionmode
```

Parameters

- rasterfilename* The name of the partitioned raster image file you created with the .CONVERTFIGURE command.
- xposition* The x-coordinate of the upper left corner of the partitioned raster image on the current logical page. This is measured in the units specified in the *units* parameter and is either relative or absolute as specified in the *positionmode* parameter. With absolute positioning, only positive values can be used. With relative positioning, a positive value moves the image to the right; a negative value moves it to the left. You can specify the default x-coordinate of 0 by using the default character in place of this parameter.
- yposition* The y-coordinate of the upper left corner of the partitioned raster image on the current logical page. This is measured in the units specified in the *units* parameter and is either relative or absolute as specified in the *positionmode* parameter. With absolute positioning, only positive values can be used. With relative positioning, a positive value moves the image down; a negative value moves it up. You can specify the default y-coordinate of 0 by using the default character in place of this parameter.

For more information on positioning, refer to Section 2 on Terminology.

.FLASHRASTER

units

The unit of measurement for the *x* and *y-position* parameters must be 0, 1, 2 or 3.

0 = Dots
1 = Inches
2 = Centimeters
3 = Millimeters

You can specify the default unit of measurement (dots) by using the default character in place of this parameter.

positionmode

Number used to specify whether the *x* and *y-positions* are relative or absolute. The relative option positions the partitioned raster image relative to the current pen location on the logical page; the absolute option positions the image at the absolute location on the logical page. Must be 0 or 1.

0 = Relative
1 = Absolute

You can specify the default position mode (relative) by using the default character in place of this parameter.

Discussion

.FLASHRASTER combines the effects of .LOADRASTER, .PRINTRASTER and .DELETERASTER to load, print and delete a partitioned raster image. Since .FLASHRASTER uses temporary partitioned raster images, and each image is referenced only once in a job, this command makes it possible to print up to 64 images on a physical page. Remember, however, that 64 is an absolute maximum and the actual number is affected by the size of the raster images.

By allowing you to specify the exact location of the partitioned raster image, this command gives you complete control over its placement on the logical page. The image can be positioned relative to the current location of the pen, or at an absolute location on the logical page.

.FLASHRASTER

The values of the x - and y -coordinates depend on the size of the current logical page. For absolute positioning, only positive values can be used. The value for x moves the image to the right, the value for y moves it down. For relative positioning, negative values can be used to move the partitioned raster image left and up, for x and y respectively.

x - and y positions are checked to ensure that they lie within the boundaries of the logical page, but does not check to verify that the entire image fits on the physical page.

The position of the Laser Printer "pen" is unaffected by printing a partitioned raster image. This means that the pen remains where it was prior to printing the image and does not move to a new location. You may have to use the `.MOVEPENABS` or `.MOVEPENREL` command before you continue printing text.

Example

```
.F RASTFILE 100 -50 * *
```

Loads the partitioned raster image in the file `RASTFILE` into Laser Printer memory, then prints the image 100 dots right and 50 dots up from the current location of the pen on the logical page. The image is then deleted from Laser Printer memory.

.HELP

Provides help on any LPS Interpreter command.

Syntax

```
.H[ELP] command
```

Parameters

command

The LPS Interpreter command for which you want help.

Discussion

When you are running the LPS Interpreter, you can request help with any of the commands. A description of the command, its parameters, and an example of its use will be displayed on the terminal.

Example

```
.H ACTIVATEPAGE
```

Displays information on the .ACTIVATEPAGE command.

```
.H A
```

Also displays information on the .ACTIVATEPAGE command; note that the abbreviated form of commands can be used.

```
.H
```

Lists commands for which help is available.

```
.H *
```

Displays information on all commands.

.INCLUDE

Allows you to include text and commands from another text file.

Syntax

```
.I[INCLUDE] filename carriagecontrol
```

Parameters

filename

The name of the formatted text file you want to include in the file you are currently using.

carriagecontrol

You may choose whether or not carriage control should be recognized in the file you are including. You can specify CCTL for carriage control, or NOCCTL for no carriage control. Using the default character in this parameter has the same effect as specifying NOCCTL.

Discussion

The .INCLUDE command tells the LPS Interpreter to change to another text file and print the text and execute the commands that it contains. The included file is processed in its entirety unless an .EXIT or another .INCLUDE command is encountered in the included file. You can have up to 15 levels of embedded .INCLUDE commands.

.INCLUDE

Example

Attached is the sales report for the Western Region.

!Sales report is stored in WESTFILE

.I WESTFILE *

!Sales report has been included here

The file WESTFILE is included and printed in the file you are currently using. Carriage control is not recognized in the included file.

.LOADRASTER

Loads a partitioned raster image into Laser Printer memory.

Syntax

```
.L[OADRASTER] rasterfilename imagenumber
```

Parameters

<i>rasterfilename</i>	The name of the partitioned raster image file you created with the .CONVERTFIGURE command.
<i>imagenumber</i>	An identification number which you supply for the partitioned raster image. Must be between 0 and 31 inclusive. This number is used later with the .PRINTRASTER or .DELETERASTER command.

Discussion

A maximum of 32 permanent partitioned raster images can be loaded into Laser Printer memory. This maximum is affected by the size of each image. Large images occupy more of the printer memory and the number 32 is the absolute maximum.

If you use the .LOADRASTER command a second time and supply the same partitioned raster image number without having deleted the image, the previously loaded image is changed from permanent to temporary. This means that it is deleted after it has been printed. All future references to this partitioned raster image number will refer to the newly loaded image.

.LOADRASTER

Example

```
.L RASTFILE 5
```

The partitioned raster image in the file **RASTFILE** is assigned number **5** and is loaded into Laser Printer memory.

.MOVEPENABS

Moves the Laser Printer "pen" relative to the upper left corner of the current logical page.

Syntax

```
.MOVEPENA[BS] xposition yposition
```

Parameters

- xposition* The number of dots that the pen should be moved to the right of the upper left corner of the logical page; must be greater than or equal to zero and must be within the boundaries of the logical page.
- yposition* The number of dots that the pen should be moved down from the upper left corner of the logical page; must be greater than or equal to zero and must be within the boundaries of the logical page.

Discussion

It is helpful to imagine that the Laser Printer writes with a "pen". The pen starts writing a character at its cell baseline and left edge. To determine where you want to move the pen, be sure to view the logical page in the direction of its orientation. The number of dots expressed in the x position parameter moves the pen to the right; the y position parameter moves it down.

You would use this command when you want the most exact placement of text on the logical page. If you try to move the pen outside the boundaries of the logical page, you will receive an error message.

Example

`.MOVEPENA 25 40`

The leftmost dot of the cell on the baseline begins printing 25 dots to the right and 40 dots down from the upper left hand corner of the current logical page.

.MOVEPENREL

Moves the Laser Printer "pen" relative to its current position on the current logical page.

Syntax

```
.MOVEPENR[EL] xposition yposition
```

Parameters

xposition

The number of dots you want to move the pen to the right of its current position on the logical page; a negative number moves the pen to the left. The pen must remain within the boundaries of the logical page.

yposition

The number of dots you want to move the pen down from its current position on the logical page; a negative number moves the pen up. the pen must remain within the boundaries of the logical page.

Discussion

It is helpful to imagine that the Laser Printer writes with a "pen". The pen starts writing a character at its cell baseline and left edge. To determine where you want to move the pen, be sure to view the logical page in the direction of its orientation. The number of dots expressed in the x position parameter moves the pen to the right of its current position; the y position parameter moves it down. Negative values move the pen to the left or up, for x position and y position respectively.

You would use this command when you want the most exact placement of text on the logical page.

.MOVEPENREL

Example

`.MOVEPENR -25 30`

The leftmost dot of the cell on the baseline begins printing 25 dots to the left and 30 dots down from the current position of the pen on the current logical page. After a character is printed the pen remains on the baseline of the cell. If the character is proportional the pen is on the right bound of the character just printed, otherwise on the right-hand edge of the cell.

.NEWFORM

Selects the form you want to print.

Syntax

```
.NEWF[ORM] formname
```

Parameters

formname

The name of the form you want to print to; must be on the current logical page.

Discussion

If there is only one form on the logical page, it is automatically selected as the current form. Use the .NEWFORM command to select the form you want to write to only when there is more than one form on the current logical page. It is used to select the particular form you want to write to. You can also use the values "1" and "2" instead of using the form name. These are forms 1 and 2 as specified on the LOGICAL PAGE FORMS Menu in IFS/3000.

Example

```
.NEWF MEMO
```

The form MEMO is selected as the form on the current logical page that you want to print to. The .WRITEFIELD command is used to print the actual text.

.NEWPAGE

Advances to the next active logical page.

Syntax

```
.NEWPA[GE]
```

Parameters

NONE

Discussion

This command is used to go to the next active logical page. If you are already on the last logical page, it causes a physical page eject and takes you back to the first active logical page.

.NEWPAGE automatically makes the first form on the first active logical page the current form. This eliminates the need to use .NEWFORM since the first active form is the one you normally want to print to. .NEWPAGE causes any forms on the current logical page to be printed before advancing to the next page.

Example

```
.NEWPA
```

The next active logical page is used for printing.

.NEWPHYSPAGE

Advances to the next physical page.

Syntax

```
.NEWPH[YSPAGE]
```

Parameters

NONE

Discussion

.NEWPHYSPAGE advances to the next physical page and begins printing with the first active logical page, including any forms associated with it. .NEWPHYSPAGE determines the current form using the same method as .NEWPAGE.

Example

```
.NEWPH
```

The Laser Printer does a physical page eject and begins printing with the first active logical page.

.NEWSUBFORM

Selects the current subform to which you want to print.

Syntax

```
.NEWS[UBFORM] subformname
```

Parameters

subformname

The name of the current subform to which you want to print.

Discussion

.NEWSUBFORM is used only when there are multiple subforms within a form. .NEWSUBFORM is required when you are printing to a form containing duplicate field names. If all the fields have different names, this command is optional.

Example

```
.NEWS ADDRESS  
.WRITEFIELD NAME *
```

The form you are printing to has more than one field called NAME. This command specifies that you want to print to the NAME field on the subform ADDRESS, as opposed to the NAME field on any other subform.

.PRINTFIGURE

Converts a figure in a figure file to a partitioned raster image file, then prints the partitioned raster image.

Syntax

```
.PRINTF[IGURE] figurefilename figurename rasterfilename imageheight  
units imagerotation xposition yposition positionmode  
rasterimagetype
```

Parameters

- figurefilename* The name of figure file that contains the figure you want to convert.
- figurename* The name of the figure that you want to convert.
- rasterfilename* The name of the partitioned raster image file that will contain the converted figure. This parameter is optional; if it is not used, you must use the default character in this position. See the Discussion for more information on supplying the partitioned raster file name.
- imageheight* The height of the partitioned raster image, viewed in the direction of its orientation, and given in the units specified in the *units* parameter. The width of the partitioned raster image is automatically calculated from the height you supply in this parameter. Refer to Section 2 on Terminology for more information on partitioned raster image height and orientation. One dot is added to the height you specify when the figure is converted; that is, a specified height of 300 dots creates a partitioned raster image 301 dots high. You only need to take this into account if you are working with extremely precise dot measurements.

.PRINTFIGURE

units

The unit of measurement to be used for the *imageheight* and *x* and *yposition* parameters. Must be 0, 1, 2 or 3.

- 0 = Dots
- 1 = Inches
- 2 = Centimeters
- 3 = Millimeters

You can specify the default unit of measurement (dots) by using the default character in place of this parameter.

imagerotation

The orientation for the printed image with respect to the logical page. Must be either 0, 90, 180 or 270 degrees. You can specify the default rotation (0 degrees) by using the default character in place of this parameter. For more information, refer to Section 2 on Terminology. Also refer to Figure 4-3 in the Examples of this command.

xposition

The x-coordinate of the upper left corner of the partitioned raster image on the current logical page. The coordinate is measured in the units specified in the *units* parameter and is either relative or absolute as specified in the *positionmode* parameter. With absolute positioning, only positive values can be used. With relative positioning, a positive value moves the image to the right; a negative value moves it to the left. You can specify the default x coordinate of 0 by using the default character in place of this parameter.

yposition

The y-coordinate of the upper left corner of the partitioned raster image on the current logical page. The coordinate is measured in the units specified in the *units* parameter and is either relative or absolute as specified in the *positionmode* parameter. With absolute positioning, only positive values can be used. With relative positioning, a positive value moves the image down; a negative value moves it up. You can specify the default y coordinate of 0 by using the default character in place of this parameter.

positionmode

Number used to specify whether the *x*- and *ypositions* are relative or absolute. The relative option positions the partitioned raster image relative to the current pen location on the logical page; the absolute option positions the

.PRINTFIGURE

image at the absolute location on the logical page. Must be 0 or 1.

0 = Relative

1 = Absolute

You can specify the default position mode (relative) by using the default character in place of this parameter.

rasterimagetype

Specifies whether the partitioned raster image is permanent or temporary. Must be 0 or 1.

0 = Temporary

1 = Permanent

A temporary image is deleted from Laser Printer memory immediately after it has been printed. A permanent image remains in memory for future use.

You can specify the default partitioned raster image type (temporary) by using the default character in place of this parameter.

Discussion

.PRINTFIGURE allows you to specify the figure to be printed in one of two ways: either with or without the *rasterfilename* as a parameter. If you use the partitioned raster file name, the partitioned raster image can be printed directly. .PRINTFIGURE compares the rotation, size and creation date of the partitioned raster image with those of the figure in the figure file. If the rotation or size do not match or if the figure date is more recent than the partitioned raster image date, a new partitioned raster image file is created. This new partitioned raster image file replaces the file you specified and is given the same name. Specifying an existing partitioned raster file name can save time and system resources since there is no need to reconvert a figure that has not been changed.

If you do not specify a partitioned raster file name, .PRINTFIGURE converts the figure in the figure file to a partitioned raster image and places it in a temporary MPE file named OUT2680A or OUT2688A.

.PRINTFIGURE

.PRINTFIGURE also deletes permanent partitioned raster images from Laser Printer memory on a least-recently used basis when it is necessary to make room for another permanent partitioned raster image. When an image has been deleted from memory, it has to be reloaded if needed again.

The position of the Laser Printer "pen" is unaffected by printing a figure. This means that the pen remains where it was prior to printing the figure and does not move to a new location. You may have to use the .MOVEPENABS or .MOVEPENREL command before you continue printing text.

Example

```
.PRINTF FIGFILE FIG RASTFILE 5 1 90 3 2 1 1
```

Prints the figure FIG which is stored in the file FIGFILE and has been converted to the file RASTFILE. It is 5 inches high and is printed at 90 degree orientation, 3 inches to the right and 2 inches down from the top left corner of the logical page. The partitioned raster image is specified as permanent and remains in Laser Printer memory after printing.

```
.PRINTF DRAWFILE DRAW * 360 * * -180 180 * *
```

The figure DRAW which is stored in the file DRAWFILE is converted to a partitioned raster image and printed. It is 361 dots high and is printed at 0 degree orientation, 180 dots to the left and 180 dots down from the current location of the pen on the logical page. The image is specified as temporary and is deleted from Laser Printer memory after printing.

Figure 4-3 shows figures printed at 0 and 90 degree rotations on different logical page orientations with the .PRINTFIGURE command. The logical page dimensions are 10 x 7 inches, and the figure is 5 inches high. The rotation of the figure is specified with respect to the logical page.

.PRINTFIGURE

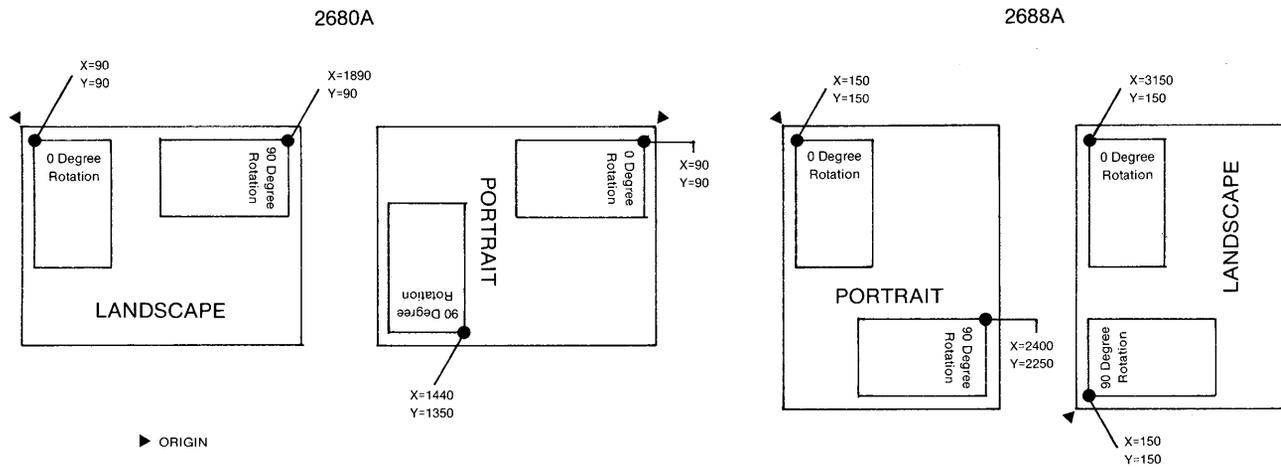


Figure 4-3. Figures printed with the .PRINTFIGURE command

.PRINTRASTER

Prints a partitioned raster image which has been loaded into Laser Printer memory.

Syntax

```
.PRINTR[ASTER] imagenumber xposition yposition units positionmode
```

Parameters

- imagenumber* The number of the image to be printed, as specified in the .LOADRASTER command. Must be between 0 and 31 inclusive.
- xposition* The x-coordinate of the upper left corner of the partitioned raster image on the current logical page. This is measured in the units specified in the *units* parameter and is either relative or absolute as specified in the *positionmode* parameter. With absolute positioning, only positive values can be used. With relative positioning, a positive value moves the image to the right; a negative value moves it to the left. You can specify the default x-coordinate of 0 by using the default character in place of this parameter.
- yposition* The y-coordinate of the upper left corner of the partitioned raster image on the current logical page. This is measured in the units specified in the *units* parameter and is either relative or absolute as specified in the *positionmode* parameter. With absolute positioning, only positive values can be used. With relative positioning, a positive value moves the image down; a negative value moves it up. You can specify the default y-coordinate of 0 by using the default character in place of this parameter.
- For more information on positioning, refer to Section 2 on Terminology and to Figure 4-4 in the Examples of this command.
- units* The unit of measurement for the x and y-position parameters. Must be 0, 1, 2

.PRINTRASTER

or 3.

- 0 = Dots
- 1 = Inches
- 2 = Centimeters
- 3 = Millimeters

You can specify the default unit of measurement (dots) by using the default character in place of this parameter.

positionmode

Number used to specify whether the x- and ypositions are relative or absolute. The relative option positions the partitioned raster image relative to the current pen location on the logical page; the absolute option positions the image at the absolute location on the logical page. Must be 0 or 1.

- 0 = Relative
- 1 = Absolute

You can specify the default position mode (relative) by using the default character in place of this parameter.

Discussion

.PRINTRASTER prints a partitioned raster image on the Laser Printer. The image must already have been loaded into Laser Printer memory with the .LOADRASTER command.

By allowing you to specify the exact location of the partitioned raster image, this command gives you complete control over its placement on the logical page. The image can be positioned relative to the current location of the pen, or at an absolute location on the logical page.

The values of the x and y-coordinates depend on the size of the current logical page. For absolute positioning, only positive values can be used. The value for x moves the image to the right, the value for y moves it down. For relative positioning, negative values can be used to move the image left and up, for x and y respectively. x and ypositions are checked to insure that they lie within the boundaries of the logical page, but does not check to verify that the entire image fits on the physical page.

.PRINTRASTER

The position of the Laser Printer "pen" is unaffected by printing a partitioned raster image. This means that the pen remains where it was prior to printing the image and does not move to a new location. You may have to use the .MOVEPENABS or .MOVEPENREL command before you continue printing text.

Example

```
.PRINTR 4 * * * 1
```

Raster image number 4 is printed in the absolute top left corner of the logical page.

```
.PRINTR 1 2 -6 1 *
```

Raster image number 1 is printed with its upper left corner 2 inches to the right and 6 inches up from the current location of the pen on the logical page. Figure 4-4 shows partitioned raster images at 0 and 90 degree rotations printed on different logical page orientations with the .PRINTRASTER command. The logical page dimensions are 10 x 7 inches and the partitioned raster image is 5 inches high. The rotation of the image was assigned with the .CONVERTFIGURE command and is specified with respect to the physical page.

.PRINTRASTER

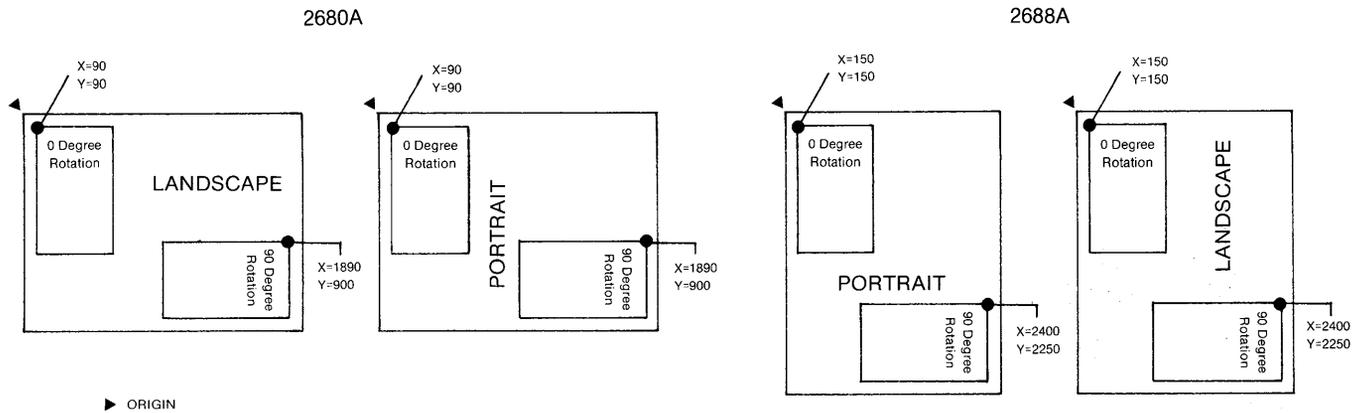


Figure 4-4. Raster images printed with the .PRINTRASTER command

.SELECTPAGE

Activates and advances to the selected logical page. Optionally performs a physical page eject.

Syntax

```
.SE[LECTPAGE] logicalpagenumber ejectoption
```

Parameters

<i>logicalpagenumber</i>	Number of the logical page to be activated.
<i>ejectoption</i>	Optional specification of PHYS for physical page eject; or, NOPHYS for no physical page eject. Using the default character in this parameter has the same effect as specifying NOPHYS.

Discussion

The Laser Printer manages logical pages by number and determines which page to print next by cycling through the logical page numbers of active logical pages. .SELECTPAGE enables you to activate any logical page in the environment, in any order, so that it will be printed once. The logical page selected remains activated and will then print in numerical order as the Laser Printer continues cycling through the active logical pages.

.SELECTPAGE

Example

.SE 2 NOPHYS

(Advances to logical page 2, no eject)

.SE 0 PHYS

(Advances to logical page 0, with eject)

.SE 14 *

(Advances to logical page 14, no eject)

.SHIFTCHAR

Changes the shift character used to change between primary and secondary character fonts.

Syntax

```
.SH[IFTCHAR] character
```

Parameters

character

The character you want to use as the shift character. You can use any character except an ampersand (&), a comma (,), a space, or any character currently defined as the comment, control, or continuation character.

Discussion

The .SHIFTCHAR is used to change to the alternate character font within a text file; that is, to the secondary font if primary is currently being used, or to primary if secondary is being used. The default shift character is a vertical bar (|). If you do not want this to be your shift character, use the .SHIFTCHAR command to change it to the character of your choice.

Example

```
.SH ^
```

This line of text includes ^ *an italic character font* ^ which was defined as the secondary font.

The shift character becomes ^. The words "an italic character font" are printed in the alternate font.

.USEFONT

Selects primary and secondary character fonts.

Syntax

```
.U[SEFONT] primaryfont secondaryfont
```

Parameters

primaryfont

The number of the character font you want to use as the primary font; must be between 0 and 31 inclusive and must correspond to a character font in the environment you are using to print your document. You can specify the current primary font by using the default character in place of this parameter.

secondaryfont

The number of the character font you want to use as the secondary font; must be between 0 and 31 inclusive and must correspond to a character font in the environment you are using to print your document. You can specify the current secondary font by using the default character in place of this parameter.

Discussion

The primary and secondary character fonts become the fonts with the numbers specified. If the default character is used, the font remains unchanged. The shift character (|) is used to change fonts within text.

Example

.U * 3

The character font the printer is using to print (primary character font) remains unchanged and the alternate (secondary character font) becomes character font number 3.

.U 0 2

The primary and secondary character fonts are now 0 and 2, respectively.

.WRITEFIELD

Prints text in a field or subfield in a form.

Syntax

```
.W[RITEFIELD] fieldname subfieldnumber
```

Parameters

fieldname

The name of the field you want to print to.

subfieldnumber

The number of the subfield within the field you want to print to. If the field has only one subfield, use the default character for this parameter.

Discussion

.WRITEFIELD enables you to print text in a field or subfield. Data is centered vertically and left justified horizontally in the field or subfield. Text will be printed even if it extends beyond the end of the field. The .WRITEFIELD command moves the Laser Printer "pen" to the baseline at the end of the text that has been printed in the field. You may have to use the .MOVEPENABS, .MOVEPENREL, or .ENDWRITEFIELD command to reposition the pen before you continue printing.

Example

.W HEADING 2
Section 4

Prints the text "Section 4" into subfield 2 of the field HEADING.

.W MEMO *
Distribution

Prints the text "Distribution" into the only subfield of the field MEMO.

LPS Interpreter Example Text Files

Example 1: Environment File

The environment file used to print the document in figure 4-5 is defined as follows:

Output Device

2680A

Character Font 0

ROM.CHARSETS.SYS

10 point

Portrait orientation

Character Font 1

ROMITAL.CHARSETS.SYS

10 point

Portrait orientation

Logical Page 0

Portrait orientation

7.0 inches wide

4.25 inches high

0.625 inches from left

1 form with 4 fields,
each with 1 subfield

Initially active

Logical Page 1

Portrait orientation

7.0 inches wide

4.25 inches high

0.625 inches from top

0.5 inches from left

No forms

Not initially active

Example 1: Text File

```
1 .W FROM *
2 Marketing Manager
3 .W TO *
4 Product Managers
5 .W DATE *
6 July 1, 1982
7 .W SUBJECT *
8 Quarterly Report
9 .MOVEPENA * 540
10 Attached is the Quarterly Report for our divisions's Marketing Department.
11
12 Please review this and return any comments to me by the end of the week.
13
14 .U * 1
15 There will be a meeting to discuss |Second Quarter Objectives| in the Board
16 Room on |Wednesday, July 7 at 3:00 p.m.|
17 !Page 1 ends here
18 .A 1
19 .DEA 0
20 .NEWPH
21 !Report will go on this page
22 .IN RPTFILE *
```

A detailed explanation of the commands used in this text file is given on the next page.

LPS EXAMPLES

LINE NO.	COMMAND AND PARAMETERS	DESCRIPTION
1	.W[RITEFIELD] FROM *	Prints text in a named field on a form. The field name. Specifies that the field has only one subfield.
9	.MOVEPENA[BS] * 540	Moves the LASER PRINTER pen relative to the absolute top left corner of the logical page; this positions the pen to begin printing the text of the memo. Moves the pen 0 dots to the left which is the default (*) position. Moves the pen 540 dots (1 inches) down.
14	.U[SEFONT] * 1	Specifies which character fonts are to be used as primary and secondary fonts. The primary font becomes the font currently being used for printing. The secondary font becomes character font number 1 from the environment file, in this case, Roman Italics.
15-16		Shift character used to change between primary and secondary fonts when the document is printed
17	!	This line is a comment line.
18	.A[CTIVATEPAGE] 1	Activates a logical page that is currently inactive which is now required for printing. Logical page number to be activated. In this case, this is the logical page without forms to be used for the second physical page of the document.

LINE NO.	COMMAND AND PARAMETERS	DESCRIPTION
19	.DEA[CTIVATEPAGE] 0	Deactivates a logical page that is currently active which is no longer required for printing. Logical page number to be deactivated. In this case, this is the logical page with the form which was used for the first physical page of the document.
20	.NEWPH[YSPAGE]	Ejects to the next physical page.
22	.I[NCLUDE] RPTFILE *	Includes and prints another text file at this point. The name of the file to be included. Specifies no carriage control (default) on the included file.

Example 2: Environment File

The environment file used to print this document is the supplied environment LP.HPENVSYS.

Example 2: Text File

```
1 The following figure is 1.5 inches high and is printed at 0
2 degree rotation with respect to the physical page.
3 .CONVERTF FIGFILE PRINTER RAST1 2680A 1.5 1 *
4 .LOAD RAST1 1
5 .PRINTR 1 2 .7 1 1
6 .MOVEPENA * 495
7 The same figure has been moved to the right so that this text
8 can be printed to the left of it.
9 .PRINTR 1 4.5 2.5 1 1
10 .MOVEPENA * 709
11 The next figure is 0.5 inches high and is printed at all four
12 rotations; i.e. 0, 90, 180, and 270 degrees.
13 .CONVERTF FIGFILE ARROW RAST2 * 0.5 1 *
14 .LOAD RAST2 2
15 .PRINTR 2 2 4.25 1 1
16 .CONVERTF FIGFILE ARROW RAST3 * 0.5 1 90
17 .LOAD RAST3 3
18 .PRINTR 3 4 4.25 1 1
19 .CONVERTF FIGFILE ARROW RAST4 * 0.5 1 180
20 .LOAD RAST4 4
21 .PRINTR 4 4 5.75 1 1
22 .CONVERTF FIGFILE ARROW RAST5 * 0.5 1 270
23 .LOAD RAST5 5
24 .PRINTR 5 2 5.75 1 1
25 .DEL 4
26 .DEL 5
```

A detailed explanation of the commands used in this text file is given on the next page.

LPS EXAMPLES

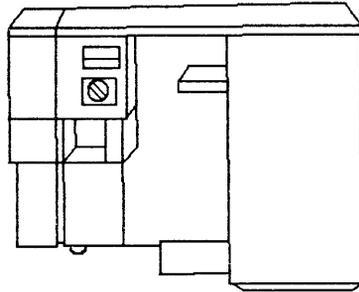
LINE NO.	COMMAND AND PARAMETERS	DESCRIPTION
3	.CONVERTF[IGURE] FIGFILE PRINTER RAST1 2680a 1.5 1 *	Converts figure in a figure file to a raster image file Name of the figure file that contains the figure. Name of the figure to be converted. A name for the raster file that will contain the converted image. output device Height of the figure, measured in units specified in the next parameter. Units of measurement, where 1 = inches. Orientation for the converted figure with respect to the physical page; default rotation is 0 degrees.
4	.L[OADRASTER] RAST1 1	Loads the raster image created with CONVERTFIGURE into the Laser Printer memory. Name of the raster image file assigned with CONVERTFIGURE. A reference number assigned to the raster image.
5	.PRINTR[ASTER] 1 2 .7 1 1	Prints the raster image created with CONVERTFIGURE and loaded into Laser Printer memory with LOADRASTER. Raster image number assigned with LOADRASTER. X-coordinate of the upper left corner of the printed image. Y-coordinate of the upper left corner of the printed image. Unit of measurement for X- and Y-coordinates, where 1 = inches. Position mode of X- and Y-coordinates, where 1 specifies that the origin is the absolute top left corner of logical page.

LINE NO.	COMMAND AND PARAMETERS	DESCRIPTION
6	.MOVEPENA[BS] * 495	Moves the Laser Printer pen relative to the absolute top left corner of the logical page; this positions the pen to begin printing the text following the figure Moves the pen 0 dots to the left which is the default (*) position. Moves the pen 654 dots down.
25	.DEL[ETERASTER] 4	Deletes a raster image from Laser Printer memory. Number of the raster image to be deleted.

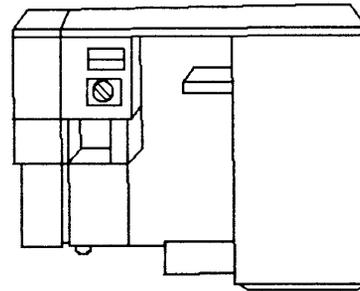
LPS EXAMPLES

Example 2: Output

The following figure is 1.5 inches high and is printed at 0 degree rotation with respect to the physical page.



The same figure has been moved to the right so that this text can be printed to the left of it.



The next figure is 0.5 inch high and is printed at all four rotations - 0, 90, 180 and 270 degrees.

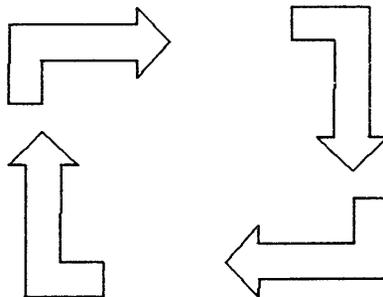


Figure 4-6. Example 2 Output

Example 3: Batch Job

The following is an example of a stream file that would initiate a batch job to run the LPS interpreter.

```
!JOB JOBNAME.USER.ACCT
!RUN LPS.PUB.SYS
2680A                (output device)
LP.HPENV.SYS        (environment file name)
N                  (non-interactive session)
MYFILE.MYGROUP.MYACCT (input file)
!EOJ
```

ERROR MESSAGE	MEANING	ACTION
20 Warning: ENDWRITEFIELD with no WRITEFIELD in progress - ignored.	The Writefield command has not been used to begin printing text in a field on a form.	Check all ENDWRITEFIELD commands to ensure that they are paired with a WRITEFIELD command.
21 Warning: INCLUDE commands nested too deep; 16th INCLUDE file ignored.	More than 15 files have been included, one within the other.	Remove one of the INCLUDE commands from your text file.
22 Invalid or missing include filename	The filename has been omitted or is not a valid filename.	Check the INCLUDE command and verify the filename.
23 Invalid or missing logical page number.	The logical page number has been omitted or does not correspond to a logical page number in the environment file you are using.	Check the ACTIVATEPAGE and DEACTIVATEPAGE commands in your text file; check the logical page numbers in your environment file.
24 Invalid or missing form name.	The form name has been omitted or is not on the current logical page.	Check the NEWFORM command in your text file; check the form name using IDIFORM; check that the form has been associated with a logical page in your environment file.
25 Invalid or missing subform name	The subform name has been omitted or is not a subform name on the current form.	Check the NEWSUBFORM command in your text file; check the subform name using IDIFORM; check that the form is associated with a logical page in your environment file.
26 Invalid or missing primary font.	The primary font number has been omitted or does not correspond to a character font number in your environment file.	Check the USEFONT command in your text file; check the character font numbers in your environment file.

ERROR MESSAGE	MEANING	ACTION
27 Invalid or missing secondary font.	The secondary font number has been omitted or does not correspond to a character font number in your environment file.	Check the USEFONT command in your text file; check the character font numbers in your environment file.
28 Invalid or missing field name.	The field name has been omitted or is not a field name on the current form.	Check the WRITEFIELD command in your text file; use IDSFORM to check the field names in your form.
29 Invalid or missing subfield.	The subfield number has been omitted or is not a subfield in the current field.	Check the WRITEFIELD command in your text file; use IDSFORM to check the number of subfields in the current field.
30 Invalid or missing comment character.	The comment character has been omitted or is not a useable character.	Check the COMMENTCHAR command in your text file; you cannot use an ampersand, a comma or a space as a comment character.
31 Invalid or missing continuation character.	The continuation character has been omitted or is not a useable character.	Check the CONTINUECHAR command in your text file; you cannot use a comma or a space as a continuation character.
32 Invalid or missing control character.	The control character has been omitted or is not a useable character.	Check the CONTROLCHAR command in your text file; you cannot use an ampersand, a comma or a space as a control character.
33 Invalid or missing default character.	The default character has been omitted or is not a useable character.	Check the DEFAULTCHAR command in your text file; you cannot use an ampersand, a comma or a space as a default character.

ERROR MESSAGE	MEANING	ACTION
34 Invalid or missing shift character.	The shift character has been omitted or is not a useable character.	Check the SHIFTCHAR command in your text file; you cannot use an ampersand, a comma or a space as a shift character.
35 Invalid or missing raster image number.	The raster image number has been omitted, is not between 0 and 31, or does not correspond to a raster image number assigned with LOADRASTER	Check the LOADRASTER command to make sure the raster image number is between 0 and 31; check the DELETERASTER and PRINTRASTER commands to make sure the image you are deleting or printing corresponds to the raster image number assigned with LOADRASTER.
36 Invalid or missing figure name.	The figure name has been omitted or is not an existing figure in the specified figure file.	Check the CONVERTFIGURE and PRINTFIGURE commands in your text file; use HPDRAW, HPEASYCHART or DSG/3000 to check that your drawing or chart has been saved as a figure in the specified figure file.
37 Invalid or missing figure filename.	The figure filename has been omitted or is not an existing figure file.	Check the CONVERTFIGURE and PRINTFIGURE commands in your text file; use HPDRAW, HPEASYCHART or DSG/3000 to check the name of your figure file.
38 Invalid or missing output device.	The output device has been omitted or is not a useable device.	Check the CONVERTFIGURE command in your text file; output device is either 2680A or 2688A.

ERROR MESSAGE	MEANING	ACTION
39 Invalid or missing raster filename.	The raster filename has been omitted or is not the name of a raster file created with CONVERTFIGURE.	Check the CONVERTFIGURE command to make sure a raster filename has been assigned; check the FLASHRASTER, LOADRASTER and PRINTFIGURE commands to make sure the raster filename corresponds to the file created with CONVERTFIGURE.
40 Invalid or missing raster image height.	The raster image height has been omitted or is outside the boundaries of the logical page.	Check the CONVERTFIGURE and PRINTFIGURE commands in your text file; check the logical page dimensions in you environment file.
41 Invalid or missing units.	Units of measurement has been omitted or is not between 0 and 3.	Check the CONVERTFIGURE, FLASHRASTER, PRINTFIGURE and PRINTRASTER commands to make sure units are 0, 1, 2 or 3; where 0=dots, 1=inches, 2=centimeters, 3=millimeters.
42 Invalid or missing raster image rotation.	The raster image rotation has been omitted or is not a useable rotation.	Raster image rotation must be 0, 90, 180, or 270 degrees; CONVERTFIGURE rotation is with respect to the physical page; for PRINTFIGURE rotation is with respect to the logical page.
43 Invalid or missing x position.	The x position has been omitted, is outside the boundaries of the logical page, or is a negative value used with absolute positioning.	Check the MOVEPENABS, MOVEPENREL, FLASHRASTER, PRINTFIGURE and PRINTRASTER commands; check the logical page dimensions in your environment file; you cannot use a negative value with absolute positioning.

ERROR MESSAGE	MEANING	ACTION
44 Invalid or missing y position.	The y position has been omitted, is outside the boundaries of the logical page, or is a negative value used with absolute positioning.	Check the MOVEPENABS, MOVEPENREL, FLASHRASTER, PRINTFIGURE and PRINTRASTER commands; check the logical page dimensions in your environment file; you cannot use a negative value with absolute positioning.
45 Invalid or missing position mode.	The position mode has been omitted or is not a useable value.	Check the FLASHRASTER, PRINTFIGURE and PRINTRASTER commands; position mode must be 0 for relative or 1 for absolute.
46 Invalid or missing raster image type.	The raster image type has been omitted or is not a useable value.	Check the PRINTFIGURE command; raster image type must be 0 for temporary, or 1 for permanent.
47 Invalid or missing carriage control option.	The carriage control option has been omitted or is not a useable value.	Carriage control option must be CCTL or NOCCTL; check the INCLUDE command.
53 Invalid or missing input raster file name.	The input raster file name has been omitted or is not a useable value.	Check the CONVERTRASTER command to make sure an output raster file name has been assigned.
54 Invalid or missing output raster file name.	The output raster file name has been omitted or is not a useable value.	Check the CONVERTRASTER command to make sure an output raster file name has been assigned.
55 Unrecognized command.	Invalid command.	Check command spelling and syntax.

ERROR MESSAGE	MEANING	ACTION
56 Invalid or missing image width.	The image width has been omitted or is not a useable value.	Check the raster image width in the CONVERTRASTER command.
57 Invalid or missing raster scaling method.	The raster scaling method has been omitted or is not a useable value.	Check the CONVERTRASTER command, scaling method must be 0, 1 or 2.
58 Invalid or missing raster scaling threshold value.	The raster scaling threshold value has been omitted or is not a useable value.	Check the CONVERTRASTER command, threshold value must be between 1 and 255.
59 Invalid or missing inverse image option.	The inverse image option value has been omitted or is not a useable value.	Check the CONVERTRASTER command; the inverse image option must be 0 or 1.
60 Warning: No more room in WRITEFIELD buffer; remaining data ignored.	Only 4096 characters are allowed per field or subfield.	Print less data or create a smaller field with IDIFORM.
61 Invalid or missing physical page eject option.	The physical page eject option has been omitted or is not a useable value.	Check the SELECTPAGE command; eject option must be PHYS or NOPHYS.

THIS PAGE SHOULD BE BLANK

Overview

The programmatic interface can be used in conjunction with IFS/3000 to control the Laser Printer. Prior to application run time, you use IFS/3000 to create an environment file containing formatting specifications for the Laser Printer. The programmatic interface gives you additional control by enabling you to supply instructions to the printer dynamically at application run time.

You can control the Laser Printer based on conditions that vary during run time, since application programs can programmatically detect these changes. You can control things such as the character font or form being printed or the location and direction of printing. You can print data into a form symbolically using field name independent of the location of the field. This means that you can change the position of fields in your form without having to make changes to the program that writes data into the form. Also, if you have the HP graphics software, you can print graphics output on the Laser Printer, as well as merge text and graphics at print time.

The programmatic interface consists of procedures called intrinsics which manage the interface between your program and the instructions to the Laser Printer.

These intrinsics refer to specifications in your environment file, such as character fonts and forms and the layout to be used to print the document. The intrinsics can be called from user programs written in COBOL, COBOL II, BASIC, FORTRAN, SPL, or PASCAL. See Appendix F for sample programs.

Procedure Descriptions

Figure 5-1 lists all programmatic intrinsics by function. The intrinsics can be divided into two groups. Each group requires specific software subsystem capabilities as follows:

- Formatting intrinsics can be used if you have IFS/3000 (HP 36580) installed on your system.
- The HP Graphics software (HP 36583) allows access to the Formatting intrinsics plus Graphics intrinsics.

Figure 5-6 summarizes the formatting and graphics intrinsics. A full description of each intrinsic is listed alphabetically, following the summary.

The HP non-graphics software includes all intrinsics in both groups.

Programmatic Intrinsic

FUNCTION	INTRINSIC (HP36580 & HP36583 unless noted)
Initialization	PINITDEVICE PINITIALIZE
Obtain information from the system	HP36580 HP36580V PERRMSG PFONTINFO PFONTNUM PLOGPAGEINFO PSTATEINFO PSTRINGWIDTH
Provide information to the system	PACTIVATEPAGE PDEACTIVATEPAGE PMOVEPENABS PMOVEPENREL PNEWFORM PNEWPAGE PNEWPHYSPAGE PNEWSUBFORM PSELECTPAGE PUSEFONT PWRITEFIELD
Graphics Procedures	PCONVERTFIGURE (only with HP 36583) PCONVERTRASTER (only with HP 36583) PDELETERASTER (only with HP 36583) PFIGUREINFO (only with HP 36583) PFLASHRASTER (only with HP 36583) PLOADRASTER (only with HP 36583) PPRINTFIGURE (only with HP 36583) PPRINTRASTER (only with HP 36583)

Figure 5-1. Functional Listing of the Programmatic Intrinsic

Calling the Intrinsic

For all programming language references in this manual, the following rules apply to the intrinsic parameters:

- All parameters are passed by reference; a literal value cannot be used as a parameter.
- Most intrinsics require a logical array, referred to as the *comarea*, to be used for the communication area.

- No condition codes are returned; the status of the call is returned in the first word of the *comarea*.
- All parameters are required. For those parameters that have defaults, you can pass values designated to indicate that the defaults should be used.

PINITDEVICE or PINITIZE must be called before any other intrinsics are used.

Figure 5-2 shows the format of calls to the intrinsics from each language.

PROGRAMMATIC INTERFACE

LANGUAGE	INTRINSIC CALL Format
COBOL	CALL intrinsic "INTRINSICNAME" USING <i>parameter1, parameter2...</i>
COBOL II	CALL "INTRINSICNAME" USING <i>parameter1, parameter2...</i>
FORTRAN	CALL INTRINSICNAME (<i>parameter1,parameter2...</i>)
BASIC	linenumber CALL INTRINSICNAME (<i>parameter1,parameter2...</i>)
SPL	INTRINSICNAME (<i>parameter1,parameter2...</i>);
PASCAL	INTRINSICNAME (<i>parameter1,parameter2...</i>);

Figure 5-2. Calling Intrinsic from Programming Languages

Where:

PROCEDURENAME

Identifies the intrinsic being called.

parameter

At least one parameter is required for most intrinsics; the particular parameters used with an intrinsic are listed in the individual intrinsic descriptions which follow in this section. When more than one parameter is specified, each is separated by a comma; for COBOL, they are separated by an optional comma and a required space.

Capabilities

All programs which call the IFS/3000 intrinsics must be PREPARED with the following user capabilities:

- PH (Process Handling)
- DS (Extra Data Segments)
- MR (Multiple RINs)

For more information on the MPE :PREPARE command, refer to the *MPE Segmenter Reference*

Manual (part number 30000-90045). If you have any questions about your user capabilities, contact your System Manager.

Parameter Data Types

The data types that are allowed for the parameters used in the intrinsics are shown in Figure 5-3.

Data Type	SPL	COBOL	FORTRAN	BASIC	PASCAL
BA	BYTE ARRAY	DISPLAY PIC X(n)	CHARACTER	STRING	PACKED ARRAY OF CHAR
LA	LOGICAL ARRAY	PIC S9(4) COMP occurs n times	LOGICAL ARRAY	INTEGER ARRAY	ARRAY OF SmallInteger (SmallInteger= -32768..32767)
I	INTEGER	PIC S9(4) COMP	INTEGER	INTEGER	INTEGER SUBRANGE WITHIN THE RANGE -32768..32767
R	REAL	*	REAL	REAL	REAL

On the HP 3000, each word is two bytes which is equivalent to 16 bits

Figure 5-3. Data Types Allowed for Various Languages

PROGRAMMATIC INTERFACE

In the discussion of the intrinsics, each parameter is identified according to its type as defined in SPL. This table is provided for those languages that do not call their data types by these particular names.

NOTE

*Note the special handling of real values in COBOL in Figure 5-3. Since COBOL does not handle real data values, IFS/3000 provides an ASCII version for intrinsics that must pass real values. These intrinsics are denoted by an "A" after the intrinsic name. For example, PCONVERTFIGURE becomes PCONVERTFIGUREA when used to pass real values in COBOL. The real parameter is a string that must contain a real number 16 characters long, padded with blanks if necessary. The real number itself consists of an integer part, a decimal point, and a decimal fraction part. A leading sign can be used; if not used, the default is a positive value.

When you pass a parameter which is a fixed length character array, you must terminate the array with a blank, an EOS (end of string), or use the entire array. An EOS character is the ASCII value 255. When an intrinsic returns a fixed length character array to you, it fills the array with trailing blanks until it reaches the maximum length. Fixed length character arrays are used for parameters such as character font name, form name, and subform name.

In BASIC, the intrinsic determines the length of a byte array from the array itself, so you do not have to terminate the array as required for other languages. For variable length arrays, you must supply a dummy variable in the array length parameter even though BASIC does not use this parameter. When an intrinsic returns a byte array to you in BASIC, it sets the array to the appropriate length (not including any trailing blanks) and, where appropriate, returns the length of the array.

Byte Arrays

For all languages except BASIC, byte arrays are terminated in one of two ways depending on whether the array is of fixed or variable length.

Communication Area

Every program must allocate a data area for communication with the intrinsics. Most intrinsics require as their first parameter a logical array referred to as the *comarea*, or communication area. This area sets up space for:

- communication with you
- communication among intrinsics
- information necessary for performing symbolic access to fields in forms
- information about character fonts and logical pages
- information about raster images

The space required for the *comarea* depends on the contents of the environment file. After you compile your environment, IFS/3000 displays the space required for the *comarea* on the IFS/3000 Main Menu. The space required is expressed in number of words. It is displayed only if the environment file has not been changed since it was last compiled. As soon as the file is changed, the number is blanked out. Once the changed file has been successfully compiled, the space required for the *comarea* is displayed.

As shown below, the *comarea* array contains information about errors that occur when the intrinsics are called. The status of the intrinsic call and error information is returned in the first 6 words of the *comarea*. Each word is two bytes which is equivalent to 16 bits.

DATA TYPE	(SPL) WORD	OFFSET	NAME	FUNCTION
Integer	1	0	User area status	status of calls
Integer	2	1	Error number	See Appendix G
Integer	3	2	Error parameter 1	error information
Integer	4	3	Error parameter 2	error information
Integer	5	4	Message catalog file number	error information
BA	6	5	Error string - 80 bytes	
Integer	27-end	26-end	RESERVED FOR SYSTEM USE	

Figure 5-4. Outline of the *comarea* Contents

User Area Status

The first word (offset 0) is the user area status word. When an intrinsic is called, it sets the user area status word to indicate the status of the intrinsic call as shown in Figure 5-5.

A successful call to any intrinsic sets the value of the user area status word to 0. If an intrinsic call causes a warning, the user area status word is set to 1. If a call to an intrinsic fails, it sets the user area status word to 2.

VALUE	MEANING
0	Successful call to intrinsic
1	WARNING
2	ERROR OCCURRED set by failed intrinsic

Figure 5-5. User Area Status Word Values

Error Number

An appropriate error or warning message can be obtained by calling PERRMSG. The error number is

available in the second word of the *comarea*. The error message corresponding to this number can be found in Appendix G.

Error Parameters 1 and 2

If an error occurs, the third and fourth words of the *comarea* are used to return information about the error, such as the file system error number.

String Parameter in an Error

If an error contains a string parameter, the string parameter begins in the sixth word of the *comarea*. The maximum length of the string is 80 bytes, including the EOS used to terminate the string. The string parameter is used to return items such as invalid character font or filename.

Message Catalog File Number

This is the file system number for the IFS/3000 message catalog (PMSGCAT.PUB.SYS).

To run programs calling PERRMSG, the IFS/3000 message catalog must be available on your system.

Communication Area Examples

Examples of defining the *comarea* follow. In all examples, nnnn is greater than or equal to the number of words required for the *comarea* as displayed on the IFS/3000 Main Menu. In the detailed examples that follow the description of each intrinsic, the *comarea* definition is not shown each time. Refer to the examples below for all instances of defining the *comarea*.

COBOL II: DATA DIVISION.

```
01 comarea.
  03 Status          pic S9(4) comp.
  03 ErrorNumber     pic S9(4) comp.
  03 ErrorParm1      pic S9(4) comp.
  03 ErrorParm2      pic S9(4) comp.
  03 Msgfilenumber   pic S9(4) comp.
  03 ErrorString     pic X(40).
  03 SystemInfo      pic S9(4) comp
                    occurs nnnn times.
```

BASIC:

```
540 REM C1 is comarea
550 REM C1[1] is Status
560 REM C1[2] is ErrorNumber
570 REM C1[3] is ErrorParm1
580 REM C1[4] is ErrorParm2
590 REM C1[5] is Msgfilenumber
600 REM C1[6] is ErrorString
610 Integer C1[nnnn]
```

FORTTRAN:

```
Integer comarea(nnnn)
Integer Status,ErrorNumber,ErrorParm1,ErrorParm2,Msgfilenumber
character *80 ErrorString
Equivalence (comarea(1),Status)
Equivalence (comarea(2),ErrorNumber)
Equivalence (comarea(3),ErrorParm1)
Equivalence (comarea(4),ErrorParm2)
Equivalence (comarea(5),Msgfilenumber)
Equivalence (comarea(6),ErrorString(1))
```

```

SPL:      Logical Array  comarea(0:nnnn);

          Define      Status      = comarea(0) #,
                  ErrorNumber = comarea(1) #,
                  ErrorParm1  = comarea(2) #,
                  ErrorParm2  = comarea(3) #,
                  MsgfileNumber = comarea(4) #;

PASCAL:   type
          SmallInteger = -32768..32767;

          var
          comarea = record
            Status,
            ErrorNumber,
            ErrorParm1,
            ErrorParm2,
            MsgfileNumber : SmallInteger;
            ErrorString   : packed array [1..80] of char;
            SystemInfo    : array [1..nnnn] of SmallInteger;
          end;

```

IFS/3000 Programmatic Intrinsic

INTRINSICS	FUNCTION
HP36580 HP36580V PACTIVATEPAGE PCONVERTFIGURE PCONVERTRASTER	Prints the version number of the IFS/3000 intrinsic. Returns the version number of the IFS/3000 intrinsic. Activates a logical page. Converts a figure in a figure file to a raster image file. Accepts raster data graphics generated by custom software, or external hardware, or which is otherwise incompatible with HP/3000 supported subsystem processing. The basic image can be manipulated, including rotation and scaling, and a formatted raster image file is created which is printable on the Laser Printer.
PDEACTIVATEPAGE PDELETERASTER PERRMSG PFIGUREINFO	Deactivates a logical page. Deletes a raster image from Laser Printer memory. Returns the message associated with an error code. Returns information about a raster imagefile or a figure in a figure file.
PFLASHRASTER	Loads a raster image into Laser Printer memory, prints it, then deletes it from Laser Printer memory.
PFONTNUM PINITDEVICE	Returns the number of a character font. Initializes the COMAREA, identifies the programming language used, the spool file number, and the output device.
PINITIALIZE	Initializes the COMAREA, identifies the programming language used and the spool file number.
PLOADRASTER PLOGPAGEINFO	Loads a raster image into Laser Printer memory. Returns logical page information.

Figure 5-6. IFS/3000 Intrinsic Summary

IFS/3000 Programmatic Intrinsic

INTRINSICS	FUNCTION
PMOVEPENABS	Moves the Laser Printer pen relative to the upper left corner of the logical page.
PMOVEPENREL	Moves the Laser Printer pen relative to its current position.
PNEWFORM	Selects the form you want to write to.
PNEWPAGE	Advances to the next logical page.
PNEWPHYSPAGE	Advances to the next physical page.
PNEWSUBFORM	Selects the subform you want to write to.
PPRINTFIGURE	Converts a figure in a figure file to a raster image file, then loads and prints the raster image and deletes it from printer memory.
PPRINTRASTER	Prints a raster image loaded via the LOADRASTER command
PSELECTPAGE	Activates and advances to the selected logical page. Optionally performs a physical page eject.
PSTATEINFO	Returns information about the current state of the print job
PSTRINGWIDTH	Returns the width of a character string in dots.
PUSEFONT	Selects primary and secondard character fonts.
PWRITEFIELD	Writes data to a field or subfield in a form.

Figure 5-6. IFS/3000 Intrinsic Summary (con't.)

HP36580

Prints the version number of the IFS/3000 intrinsics currently installed on your system.

Syntax

```
HP36580
```

Parameters

NONE

Discussion

The HP36580 intrinsic prints the following message to \$STDLIST:

```
The current version of the IFS/3000 intrinsics is v.uu.ff
```

where *v* is the major product enhancement level, *uu* is the product update level, and *ff* is the product fix level.

HP36580V

Returns the version number of the IFS/3000 intrinsics currently installed on your system.

Syntax

```
LA  
HP36580V (version)  
out
```

Parameters

version Logical array used to return the version number of the intrinsics installed on your system. This array must be at least 8 words long and is terminated by an EOS when returned to you.

Discussion

When communicating information or problems to Hewlett-Packard, it is helpful to provide the IFS/3000 version number.

The version number returned to you is in the format HP36580v.uu.ff, where HP36580 is the IFS/3000 product number, v is the major product enhancement level uu is the product update level, and ff is the product fix level.

Example

```
COBOL II:      DATA DIVISION.  
  
               version          pic S9(4) comp  
                               occurs 8 times.
```

HP36580V

PROCEDURE DIVISION.

CALL INTRINSIC "HP36580V" USING *version*.

BASIC: 500 CALL HP36580V (V1[*])

FORTRAN: CALL HP36580V (*version*)

SPL: HP36580V (*version*);

PASCAL: HP36580V (*version*);

PACTIVATEPAGE

Activates a particular logical page.

Syntax

```
LA      I
PACTIVATEPAGE (comarea, pagenumber)
           in/out      in
```

Parameters

comarea

Logical array containing information for internal use. The first word returns the status of the intrinsic call.

pagenumber

Integer variable used to supply the number of the logical page you want to activate. *pagenumber* must have a value between 0 and 31, inclusive.

Discussion

IFS/3000 manages logical pages by number in the Logical Page Table, and determines which page to print next by cycling through this table. By calling PACTIVATEPAGE and its counterpart PDEACTIVATEPAGE, you set an indicator in the Logical Page Table indicating which logical pages are currently active. If a logical page is not currently active, it is skipped and is not printed.

If there are no logical pages defined in the environment, the Laser Printer default logical page is assumed to be active.

PACTIVATEPAGE

Example

```
COBOL II:  DATA DIVISION.  
           01  pagenumber                               pic S9(4) comp.  
           PROCEDURE DIVISION.  
           CALL INTRINSIC "PACTIVATEPAGE" USING comarea,  
                                                pagenumber.  
  
BASIC:    700 CALL PACTIVATEPAGE (C1[*],P1)  
FORTRAN:  CALL PACTIVATEPAGE (comarea,pagenumber)  
SPL:     PACTIVATEPAGE (comarea,pagenumber);  
PASCAL:  PACTIVATEPAGE (comarea,pagenumber);
```

PCONVERTFIGURE[A]

Converts a figure in a figure file into a partitioned raster image file which is the format required for printing by the Laser Printer. The ASCII version, callable from COBOL, is denoted by an "A" after the intrinsic name.

Syntax

```
PCONVERTFIGURE (LA      BA      BA      BA
                 in/out   in      in      in
                 BA      R      I      I
                 in      in      in      in
                 outputdevice,imageheight,units,imagerotation)
```

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- figurefilename* Byte array used to supply the name of the figure file that contains the figure you want to convert. This array can be up to 35 characters to allow for a fully qualified MPE filename. If the filename is less than 35 characters, it must be terminated by a blank or an EOS. In BASIC a blank or EOS is not necessary since the length is determined from the array itself.
- figurename* Byte array used to supply the name of the figure within the specified figure file that you want to convert. This array can be up to 16 characters (upper or lower case) beginning with a letter, and cannot include any embedded blanks. If *figurename* is less than 16 characters, it must be terminated by a blank or an EOS. In BASIC a blank or EOS is not necessary since the length is determined from the array itself.

PCONVERTFIGURE[A]

- rasterfilename* Byte array used to supply the name of the partitioned raster image file that will contain the converted figure. This array can be up to 35 characters to allow for a fully qualified MPE filename. It must be terminated by a blank, an EOS, or the 35th character, except in BASIC.
- outputdevice* Byte array used to supply the name of the output device for which the figure is to be converted. This array can be up to 7 characters and must be terminated by a blank, an EOS, or the 7th character. In BASIC, the length is determined from the array itself.
- imageheight* Real variable used to supply the height of the partitioned raster image, viewed in the direction of its orientation, and given in the units specified in the *units* parameter. The width is automatically calculated from the *imageheight* value. For COBOL, the value is passed as an ASCII string. Refer to Section 2 on Terminology for more information on image height and orientation. One dot is added to the height when the image is converted; that is, a specified height of 300 dots creates an image 301 dots high. You only need to take this into account if you are working with extremely precise dot measurements.
- units* Integer variable used to supply the unit of measurement to be used for *imageheight*. *units* must have a value of 0, 1, 2 or 3.
- 0 = Dots
 - 1 = Inches
 - 2 = Centimeters
 - 3 = Millimeters
- imagerotation* Integer variable used to supply the orientation of the partitioned raster image with respect to the physical page; must have a value of 0, 90, 180 or 270 degrees. Refer to Section 2 on Terminology for more information on partitioned raster image rotation.

PCONVERTFIGURE[A]

Discussion

PCONVERTFIGURE converts a figure in a figure file into a partitioned raster image file which is the format required for printing by the Laser Printer. The *rasterfilename* you specify here is later used with the PLOADRASTER or PFLASHRASTER intrinsic.

Example

COBOL II: DATA DIVISION.

```
01 figurefilename          pic X(35).  
01 figurename             pic X(16).  
01 rasterfilename        pic X(35).  
01 outputdevice         pic X(8).  
01 imageheight          pic X(16).  
01 units                 pic S9(4) comp.  
01 imagerotation        pic S9(4) comp.
```

PROCEDURE DIVISION.

```
CALL INTRINSIC "PCONVERTFIGUREA" USING comarea,  
                                         figurefilename,  
                                         figurename,  
                                         rasterfilename,  
                                         outputdevice,  
                                         imageheight,  
                                         units,  
                                         imagerotation.
```

BASIC: CALL PCONVERTFIGURE (C1[*],F1\$,F2\$,R1\$,D1\$,H,U,R)

FORTRAN: CALL PCONVERTFIGURE (*comarea*,*figurefilename*,*figurename*,
& *rasterfilename*,*outputdevice*,*imageheight*,*units*,
& *imagerotation*)

PCONVERTFIGURE[A]

SPL: PCONVERTFIGURE (*comarea,figurefilename,figurename,rasterfilename,*
outputdevice,imageheight,units,imagerotation);

PASCAL: PCONVERTFIGURE (*comarea,figurefilename,figurename,rasterfilename,*
outputdevice,imageheight,units,imagerotation);

PCONVERTRASTER[A]

Accepts dot-per-bit raster data graphics. The basic image can be manipulated, including rotation and scaling, and a partitioned raster image file is created which is printable on the Laser Printer. The ASCII version, callable from COBOL, is denoted by an "A" after the intrinsic name.

Syntax

```
PCONVERTRASTER (LA      BA      BA      BA
                 in/out   in      in      in
                 R      R      IA
                 imageheight, imagewidth, convertarray)
                 in      in      in
```

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- inrasterfilename* Byte array used to supply the name of the partitioned raster file that you want to manipulate. This array can be up to 35 characters (upper or lower case) beginning with a letter, and cannot include any embedded blanks. If *inrasterfilename* is less than 35 characters, it must be terminated with a blank or an EOS. In BASIC, a blank or EOS is not necessary since the length is determined from the array itself.
- outrasterfilename* Byte array used to supply the name of the dot-per-bit raster image file that will contain the partitioned partitioned raster image. This array can be up to 35 characters to allow for a fully qualified MPE filename. It must be terminated by a blank, an EOS, or the 35th character, except in BASIC. The file code for a partitioned raster image file is 1114 or RASTR.

PCONVERTRASTER[A]

outputdevice

Byte array used to supply the name of the output device for which the figure is to be converted. This array can be up to 7 characters and must be terminated by a blank, an EOS, or the 7th character. In BASIC, the length is determined from the array itself.

imageheight

Real variable used to supply the height of the partitioned raster image, viewed in the direction of its orientation, prior to rotation, and given in the units specified in the *units* parameter. For COBOL, the value is passed as an ASCII string. Refer to Section 2 on Terminology for more information on image height and orientation.

imagewidth

Real variable used to supply the width of the partitioned raster image, prior to rotation, given in the units specified in the *units* parameter. For COBOL, the value is passed an ASCII string.

Note: Both *imageheight* and *imagewidth* are required parameters and the ratio aspects must be calculated for each.

convertarray

Integer array containing the following five integer values:

units

Integer variable used to supply the unit of measurement to be used for *imageheight* and *imagewidth*. The *units* must have a value of 0, 1, 2 or 3, where:

- 0 = dots
- 1 = inches
- 2 = centimeters
- 3 = millimeters

imagerotation

Integer variable used to supply the orientation of the printed raster image with respect to the physical page. *Imagerotation* must have a value of 0, 90, 180 or 270. Refer to Section 2 on Terminology for more information on partitioned raster image rotation.

The temporary file, ROTATEFL, is created which can be saved and used separately.

PCONVERTASTER[A]

scalingmethod

Integer variable used to supply the method used to scale the raster image. *scalingmethod* must have a value of 0, 1, 2. For any one of the following integers specified, the temporary file, SCALEFL, is created which can be saved and used separately.

- 0 = Threshold testing only. This method uses the input threshold defined in word four, or calculates a default threshold which serves as a scaled dot comparison value. This method has the best general performance and does well at scaling line art graphics.
- 1 = Pattern recognition. This method attempts to duplicate the pattern of the source image by repeating the pattern in the scaled image. It is more CPU intensive but should give better results in scaled patterns and text.
- 2 = Randomized threshold testing. This method is best used in grey tone picture scaling where there are few defined hard lines. This method is CPU intensive. The *thresholdvalue* in word four can be used to generally increase or decrease the lightness of the scaled image.

thresholdvalue

Integer variable used to specify a threshold between 1 and 255 which is used in the scaling routines. If the value is zero the scaling routine will calculate a default threshold based on the scaling factor. This value is passed only if *scalingmethod* = 0 or 2. By increasing the threshold the scaled image becomes generally lighter.

inverseoption

Integer variable used to specify inverse image as = 1, or normal (non-inverse) image as = 0. The temporary file, INVERSFL, is created and can be saved and used separately.

PCONVERTRASTER[A]

Discussion

PCONVERTRASTER accepts a dot-per-bit raster file. The basic image can be manipulated, including rotation and scaling, and a partitioned raster image file is created which is printable on the Laser Printer. The *outrasterfilename* you specify here is used later with the PLOADRASTER or PFLASHRASTER intrinsics.

The requirements for the input raster data are:

1. The existing partitioned raster file must be a permanent disc file with fixed record length, where each logical record represents a left to right raster scan line.
2. The raster data records must be binary with one bit-per-pixel where there is a one-to-one correspondence between each bit turned on and each dot of the printed scan line.
3. Each scan line (record) can be a maximum of 124 words (1980 dots) in length, and there cannot be more than 3060 scan lines (records) in total.

PCONVERTRASTER performs as many as four separate functions depending on the values assigned in the parameter list. Each function creates a separate temporary file of raster data which can be saved and used separately.

Efficiency is increased by determining which function will create a basic temporary file suitable for multiple format manipulation without repeating the basic function. As an example, scaling a basic image before multiple rotations of the image would be more efficient than rescaling multiple rotations.

Example

```
COBOL II:  DATA DIVISION.

           01 inrasterfilename                pic X(35).
           01 outrasterfilename              pic X(35).
           01 outputdevice                   pic X(8).
```

PCONVERTRASTER[A]

```
01 imageheight                pic X(16).
01 imagewidth                  pic X(16).
01 convertarray.
    05 units                    pic S9(4) comp.
    05 imagerotation           pic S9(4) comp.
    05 scalingmethod           pic S9(4) comp.
    05 thresholdvalue         pic S9(4) comp.
    05 inverseoption          pic S9(4) comp.
    05 Filler                  pic S9(4) comp
                                occurs 15 times.
```

PROCEDURE DIVISION.

```
CALL INTRINSIC "PCONVERTRASTERA" USING comarea,
                                         inrasterfilename,
                                         outrasterfilename,
                                         outputdevice,
                                         imageheight,
                                         imagewidth,
                                         convertarray.
```

```
BASIC:    CALL PCONVERTRASTER (C[*],F1$,F2$,D1$,H,W,L[*])
```

```
FORTRAN:  CALL PCONVERTRASTER (comarea,inrasterfilename,outrasterfilename,
&                               outputdevice,imageheight,imagewidth,convertarray)
```

```
SPL:      PCONVERTRASTER (comarea,inrasterfilename,outrasterfilename,
                           outputdevice,imageheight,imagewidth,convertarray);
```

```
PASCAL:   PCONVERTRASTER (comarea,inrasterfilename,outrasterfilename,
                           outputdevice,imageheight,imagewidth,convertarray);
```

PDEACTIVATEPAGE

Deactivates a particular logical page.

Syntax

```
PDEACTIVATEPAGE (comarea, pagenumber)
                  LA      I
                  in/out  in
```

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- pagenumber* Integer variable used to supply the page number of the logical page you want to deactivate. *pagenumber* must have a value between 0 and 31, inclusive.

Discussion

Refer to the discussion of PACTIVATEPAGE in this section.

There must always be at least one logical page active in a job when attempting to print to the printer. PDEACTIVATEPAGE returns a warning if you try to deactivate the only active page. If there are no logical pages defined in the environment, the Laser Printer default logical page is assumed to be active.

PDEACTIVATEPAGE

Example

```
COBOL II:    DATA DIVISION.

              01  pagenumber                pic S9(4) comp.

              PROCEDURE DIVISION.

              CALL INTRINSIC "PDEACTIVATEPAGE" USING comarea,
                                                       pagenumber.

BASIC:      700 CALL PDEACTIVATEPAGE (C1[*],P1)

FORTRAN:    CALL PDEACTIVATEPAGE (comarea,pagenumber)

SPL:       PDEACTIVATEPAGE (comarea,pagenumber);

PASCAL:    PDEACTIVATEPAGE (comarea,pagenumber);
```

PDELETERASTER

Deletes a partitioned raster image from Laser Printer memory.

Syntax

```
LA      I
PDELETERASTER (comarea,imagenumber)
           in/out      in
```

Parameters

comarea Logical array containing information for internal use. The first word returns the status of the intrinsic call.

imagenumber Integer variable used to supply the number of the partitioned raster image to be deleted. *imagenumber* must have a value between 0 and 31 and must be the number of an image that has already been loaded with the PLOADRASTER intrinsic.

Discussion

A partitioned raster image can be deleted from Laser Printer memory only if it has already been loaded with the PLOADRASTER intrinsic. PDELETERASTER would normally be used after your last reference to the partitioned raster image with the PPRINTRASTER intrinsic.

Example

```
COBOL II:  DATA DIVISION.
           01  imagenumber      pic S9(4) comp.
```

PDELETERASTER

PROCEDURE DIVISION.

CALL INTRINSIC "PDELETERASTER" USING *comarea*,
imagenumber.

BASIC: 600 CALL PDELETERASTER (C1[*],I2)

FORTRAN: CALL PDELETERASTER (*comarea*,*imagenumber*)

SPL: PDELETERASTER (*comarea*,*imagenumber*);

PASCAL: PDELETERASTER (*comarea*,*imagenumber*);

PERRMSG

Returns the error or warning message associated with the error number received when a call to an intrinsic is not successful.

Syntax

	LA	BA	I	I
PERRMSG	(<i>comarea</i> ,	<i>errormessage</i> ,	<i>arraylength</i> ,	<i>messagelength</i>)
	in/out	out	in	out

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- errormessage* Byte array used to return the error message for an error which occurs after a call to an IFS/3000 intrinsic. The error message is padded with blanks to the end of array. This array must be at least 80 bytes long to avoid truncating any error messages.
- arraylength* Integer variable used to supply the length of the error message array. In BASIC, the maximum length of the error message is determined from the array itself, but you must supply a dummy variable for this parameter.
- messagelength* Integer variable used to return the length of the error message itself, not including any trailing blanks. In BASIC, you can use a dummy variable since the length is set by the string itself.

PERRMSG

Discussion

The first word of the *comarea* returns the status of an intrinsic call. If the status is not set to 0, the intrinsic call has failed, and you should call PERRMSG to get the message associated with the error. If the call to PERRMSG is successful, PERRMSG sets the status word in the *comarea* to 0. If PERRMSG fails, it returns an appropriate error message but does not reset the error indication in the *comarea*.

If you call PERRMSG twice in a row, PERRMSG fails since there is no error the second time. It returns an appropriate error message in the error message array, but does not reset the error indication in the *comarea*.

PERRMSG depends on the IFS/3000 message catalog (PMSGCAT.PUB.SYS). To run programs calling PERRMSG, you must have the IFS/3000 message catalog available on your system.

Example

COBOL II: DATA DIVISION.

```
01 errormessage                pic X(80).  
01 arraylength                 pic S9(4) comp.  
01 messagelength              pic S9(4) comp.
```

PROCEDURE DIVISION.

```
CALL INTRINSIC "PERRMSG" USING comarea,  
                             errormessage,  
                             arraylength,  
                             messagelength.
```

BASIC: 550 CALL PERRMSG (C1[*],E1\$,E1,E2)

FORTRAN: CALL PERRMSG (*comarea*,*errormessage*,*arraylength*,*messagelength*)

SPL: PERRMSG (*comarea*,*errormessage*,*arraylength*,*messagelength*);

PASCAL: PERRMSG (*comarea*,*errormessage*,*arraylength*,*messagelength*);

PFIGUREINFO[A]

Returns information about a partitioned raster image file or a figure in a figure file. The ASCII version, callable by COBOL when a real data type is required, is denoted by an "A" after the intrinsic name.

Syntax

```

                LA      BA      I      I      LA      I
PFIGUREINFO (comarea,filename,filetype,itemnumber,itemvalue,itemlength)
                in/out   in     in/out   in     out     out
```

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- filename* Byte array used to supply the name of the figure and figure file or the partitioned raster image file for which you want information. This array can be up to 52 characters long, terminated by a blank, an EOS, or the 52nd character. In BASIC, the length is determined by the array itself. The figure filename must be followed by a figure name which must be separated from the filename by a colon (:) and followed by a blank or an EOS (e.g. *filename:figurename*). You cannot specify a figure name with a partitioned raster image file.
- filetype* Integer variable used to supply or return the kind of file specified in the *filename* parameter. File type must have a value of -1, 0, or 1.
- 1 = Unknown Type (the actual type is returned in this parameter)
 - 0 = Raster Image File
 - 1 = Figure in a Figure File
- itemnumber* Integer variable used to supply the number of the item which is to be returned. See Figure 5-7 for a description of item numbers.

PFIGUREINFO[A]

itemvalue

Returns the information requested for a particular item. See Figure 5-7 for data types to be used with different items. For COBOL, an ASCII string is returned if this parameter is a real value.

itemlength

Integer variable used to return the length of the *itemvalue* parameter in bytes. The *itemlengths* required for different *itemnumbers* are provided in Figure 5-7 on the following pages.

ITEM NO.	FILE TYPE	DATA TYPE	LENGTH	ITEM
1	Raster	Byte Array	8 bytes	Name of the output device
2	Raster	Real Array	8 bytes*	Resolution of the output device, measured in number of dots per unit given in item 3. Array(1) = horizontal resolution Array(2) = vertical resolution
3	Raster	Integer	2 bytes	Units used to measure output device resolution given in item 2. 1 = Inches 2 = Centimeters 3 = Millimeters

Figure 5-7. Information Available from PFIGUREINFO

PFIGUREINFO[A]

ITEM NO.	FILE TYPE	DATA TYPE	LENGTH	ITEM
4	Raster or Figure	Real Array	16 bytes*	<p>Minimum and maximum X and Y values of the raster image or figure. Raster image units are given in item 3. Figure units are given in a 100 x 100 unit figure space.</p> <p>Array(1) = minimum X value Array(2) = maximum X value Array(3) = minimum Y value Array(4) = maximum Y value</p>
5	Raster	Real	4 bytes*	Orientation of the raster image. Ranges from 0.0 to 360.0 degrees.
6	Raster	Integer	2 bytes	<p>Source from which the raster image was generated.</p> <p>0 = unknown 1 = figure in figure file</p>
7	Raster	Byte Array	Variable Length	<p>Name of the source given in item 6.</p> <p>Bytes 1-16 = figure name Bytes 17-52 = figure file name</p>
8	Raster or Figure	Byte Array	8 bytes	<p>Version of IFS/3000 that created the raster image, in the format v.uu.ff</p> <p>v = major enhancement level uu = update level ff = fix level</p>

Figure 5-7. Information Available from PFIGUREINFO (cont.)

PFIGUREINFO[A]

ITEM NO.	FILE TYPE	DATA TYPE	LENGTH	ITEM
9	Raster or Figure	Unsigned Logical	2 bytes	Date the raster image or figure was created or last modified, in the same format as the MPE CALENDAR intrinsic.
10	Raster or Figure	Doubleword	4 bytes	Time the raster image or figure was created or last modified, in the same format as the MPE CLOCK intrinsic.
11	Raster or Figure	Unsigned Logical	2 bytes	System crash indicator. If TRUE, integrity of raster image or figure is questionable.
12	Raster or Figure	Real Array	8 bytes*	Smallest and largest text sizes contained in the image. Values represent the proportion of the cell height to the total figure or raster image height. Array(1) = Smallest Text Proportion Array(2) = Largest Text Proportion
13	RESERVED FOR SYSTEM USE			
14	Raster	Doubleword	4 bytes	Number of vectors used to create the raster image
15	Raster	Integer Array	8 bytes	Distance in dots from the 4 borders of the raster image to the upper left corner of the image. Array(1) = Left border Array(2) = Right border Array(3) = Top border Array(4) = Bottom border

Figure 5-7. Information Available from PFIGUREINFO (cont.)

PFIGUREINFO[A]

ITEM NO.	FILE TYPE	DATA TYPE	LENGTH	ITEM
16	Figure	Unsigned Logical	2 bytes	Date the figure was created, in the same format as the MPE CALENDAR intrinsic.
17	Figure	Doubleword	4 bytes	Time the figure was created, in the same format as the MPE CLOCK intrinsic.
18	Raster	Integer	2 bytes	The number of bits used to store each element in the raster image file.
19	Figure	Byte Array	128 bytes	Comment stored in the figure header.

Figure 5-7. Information Available from PFIGUREINFO (cont.)

- * COBOL requires a larger array for real values than other languages. The length given in Figure 5-7 should be multiplied by 4 when calling PFIGUREINFOA from COBOL; that is, a stated length of 4 bytes requires 16 bytes in COBOL, a length of 8 bytes requires 32, and so forth.

Discussion

PFIGUREINFO can be used to return information about a partitioned raster image file or a figure in a figure file. If you specify a figure file, you must also specify a figure name. Figure name is not applicable to a partitioned raster image file. If you specify a figure name with a raster image file, you will receive an error message.

If you specify a figure in a figure file, only the MPE permanent file domain is searched. If you specify a partitioned raster file, the MPE temporary domain is searched first, followed by the permanent domain.

If you request information about an item that is not applicable to the file type specified, PFIGUREINFO returns an item length of 0 and the contents of item value remain unchanged.

PFIGUREINFO[A]

Example

COBOL II: DATA DIVISION.

```
77 filename                pic X(52) value spaces.
77 filetype                pic S9(4) comp.
77 itemlength             pic S9(4) comp.
77 itemnumber            pic S9(4) comp.

01 itemvalueDescription.
  03 itemvalue            pic S9(4) comp
                        occurs 64 times.
  03 ByteValue redefines itemvalue
                        pic X(1)
                        occurs 128 times.
  03 DoubleValue redefines itemvalue
                        pic S9(9) comp
                        occurs 32 times.
  03 IntegerValue redefines itemvalue
                        pic S9(4) comp
                        occurs 64 times.
  03 RealValue redefines itemvalue
                        pic X(16)
                        occurs 8 times.
```

PROCEDURE DIVISION.

```
CALL INTRINSIC "PFIGUREINFOA" USING comarea,
                                     filename,
                                     filetype,
                                     itemnumber,
                                     itemvalue,
                                     itemlength.
```

```
BASIC: 10 REM C1 is comarea
        20 REM F1 is filetype
```

PFIGUREINFO[A]

```
30 REM I1 is itemnumber
40 REM L1 is itemlength
50 REM F$ is filename
60 DIM C1[1000], F$(35)
100 REM Substitute a variable of the appropriate type
110 REM for the "X", depending on the item requested
120 REM
130 CALL PFIGUREINFO (C1[*],F$,F1,I1,X,L1)
```

```
FORTRAN:  Integer filetype,itemnumber,itemlength
          Logical itemvalue(64)
          character*128 ByteValue
          Integer*4 DoubleValue
          Integer IntegerValue(4)
          Logical LogicalValue
          Real RealValue(4)
          Equivalence (itemvalue(1),ByteValue,DoubleValue,IntegerValue,
&                    LogicalValue,RealValue(1))
          character*35 filename

          CALL PFIGUREINFO (comarea,filename,filetype,itemnumber,
&                          itemvalue,itemlength)
```

```
SPL:      Integer filetype,itemnumber,itemlength;
          Byte Array filename(0:34);
          Logical Array itemvalue(0:63);
          Byte Array ByteValue(*)=itemvalue;
          Double DoubleValue=itemvalue;
          Integer Array IntegerValue(*)=itemvalue;
          Logical LogicalValue=itemvalue;
          Real Array RealValue(*)=itemvalue;

          PFIGUREINFO (comarea,filename,filetype,itemnumber,itemvalue,
&                    itemlength);
```

```
PASCAL:   type
          SmallInteger = -32768..32767;
```

PFIGUREINFO[A]

```
char52array = packed array[1..52] of char;
char128array = packed array[1..128] of char;
logical64array = array[1..64] of logical;

itemrec = record
    dummy      : 0..16; {align on word boundary}
    case retag : itemtype of
        zlogical  : (LogicalValue : logical64array);
        zchar     : (ByteValue    : char128array);
        zdouble   : (DoubleValue  : integer);
        zreal     : (RealValue    : array[1..4] of real);
        zword     : (WordValue    : array[1..4] of SmallInteger);
    end;

var
    filename   : char52array;
    filetype   : SmallInteger;
    itemnumber  : SmallInteger;
    itemvalue  : itemrec;
    itemlength : SmallInteger;

PFIGUREINFO (comarea,filename,filetype,itemnumber,itemvalue,
            itemlength);
```

PFLASHRASTER[A]

Loads a partitioned raster image into Laser Printer memory, prints it, then deletes it from Laser Printer memory. The ASCII version, callable from COBOL, is denoted by an "A" after the intrinsic name.

Syntax

	LA	BA	R	R	I	I
PFLASHRASTER	(<i>comarea</i> ,	<i>rasterfilename</i> ,	<i>xposition</i> ,	<i>yposition</i> ,	<i>units</i> ,	<i>positionmode</i>)
	in/out	in	in	in	in	in

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- rasterfilename* Byte array used to supply the name of the partitioned raster image file created with the PCONVERTFIGURE intrinsic. This array can be up to 35 characters to allow for a fully qualified MPE filename. It must be terminated by a blank, an EOS, or the 35th character, except in BASIC.
- xposition* Real variable used to supply the x-coordinate of the upper left corner of the raster image on the current logical page. The coordinate is measured in the units specified in the *units* parameter and is either relative or absolute as specified in the *positionmode* parameter. With absolute positioning, only positive values are allowed. With relative positioning, a positive value moves the image to the right; a negative value moves it to the left. For COBOL, the value is passed as an ASCII string.
- yposition* Real variable used to supply the y-coordinate of the upper left corner of the raster image on the current logical page. The coordinate is measured in the units specified in the *units* parameter and is either relative or absolute as specified in the *positionmode* parameter. With absolute positioning, only

PFLASHRASTER[A]

positive values are allowed. With relative positioning, a positive value moves the image down; a negative value moves it up. For COBOL, the value is passed as an ASCII string.

For more information on positioning, refer to Section 2 on Terminology.

units

Integer variable used to supply the units of measurement for the *x*- and *y*position parameters. *units* must have a value of 0, 1, 2 or 3.

- 0 = Dots
- 1 = Inches
- 2 = Centimeters
- 3 = Millimeters

positionmode

Integer variable used to specify whether the *x*- and *y*-coordinates are relative or absolute. The relative option positions the partitioned raster image relative to the current pen location on the logical page; the absolute option positions the image at the absolute location on the logical page.

- 0 = Relative
- 1 = Absolute

Discussion

PFLASHRASTER combines the effects of PLOADRASTER, PPRINTRASTER and PDELETERASTER to load, print and delete a partitioned raster image. Since PFLASHRASTER uses temporary raster images, and each image is referenced only once in a job, this intrinsic makes it possible to print up to 64 images on a physical page. Remember, however, that 64 is an absolute maximum and the actual number is affected by the size of the raster images.

For the raster filename, the MPE temporary file domain is searched first, followed by the permanent file domain.

PFLASHRASTER[A]

By allowing you to specify the exact location of the partitioned raster image, PFLASHRASTER gives you complete control over its placement on the logical page. The image can be positioned relative to the current location of the pen, or at an absolute location on the logical page.

The values of the x - and y -coordinates depend on the size of the current logical page. For absolute positioning, only positive values can be used. The value for x moves the image to the right, the value for y moves it down. For relative positioning, negative values can be used to move the image left and up, for x and y respectively.

x - and y position are checked to insure that they lie within the boundaries of the logical page, but no checking is done to verify that the entire image fits on the physical page.

The position of the Laser Printer "pen" is unaffected by printing a partitioned raster image. This means that the pen remains where it was prior to printing the image and does not move to a new location.

Example

```
COBOL II:  DATA DIVISION.

           01  rasterfilename           pic X(35).
           01  xposition                 pic X(16).
           01  yposition                 pic X(16).
           01  units                     pic S9(4) comp.
           01  positionmode              pic S9(4) comp.

PROCEDURE DIVISION.

CALL INTRINSIC "PFLASHRASTER" USING comarea,
                                     rasterfilename,
                                     xposition,
                                     yposition,
                                     units,
                                     positionmode.

BASIC:    300 CALL PFLASHRASTER (C1[*],R1$,X,Y,U,P)
```


PFONTINFO

Returns information about a particular character font in the environment.

Syntax

LA	I	BA	LA	
PFONTINFO	(<i>comarea</i> ,	<i>fontnumber</i> ,	<i>fontname</i> ,	<i>fontinformation</i>)
	in/out	in	out	out

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- fontnumber* Integer variable used to supply the number of the character font. Character font number must have a value between 0 and 31, inclusive.
- fontname* Byte array in which the name of the character font is returned. This array must be 16 bytes long.
- fontinformation* Logical array in which information about the character font is returned. This array must be at least 30 words long. Its contents are:

PFONTINFO

WORD	CONTENTS
0	ORIENTATION (0,90,180,270)
1	HEIGHT IN DOTS
2	CELL WIDTH IN DOTS (MAXIMUM WIDTH FOR PROPORTIONAL SPACING)
3	BASELINE (1=BOTTOM OF CELL)
4	EXTRA CHARACTER SPACING
5	EXTRA LINE SPACING
6	PROPORTIONAL SPACING INDICATOR: 0=NO, -1*=YES
7-29	Reserved for future use

Discussion

The character font is specified by number in this intrinsic. If necessary, you can use PFONTNUM to get the character font number.

The fontname is returned in upper case and padded with blanks out to 16 bytes. In BASIC, the array is returned with its length set to the number of non-blank characters.

If there is no name assigned to the character font, all blanks are returned in the fontname array in COBOL, SPL, FORTRAN and PASCAL. In BASIC, the array is returned with length 0.

Example

COBOL II: DATA DIVISION.

```

01 fontnumber                pic S9(4) comp.
01 fontname                  pic X(16).
01 fontinformation          pic S9(4) comp
                               occurs 30 times.
    
```

PROCEDURE DIVISION.

```

CALL INTRINSIC "PFONTINFO" USING comarea,
                                fontnumber,
                                fontname,
                                fontinformation.
    
```

BASIC: 1020 CALL PFONTINFO (C1[*],I,C9\$,L1[*])

FORTRAN: CALL PFONTINFO (*comarea,fontnumber,fontname,fontinformation*)

SPL: PFONTINFO (*comarea,fontnumber,fontname,fontinformation*);

PASCAL: PFONTINFO (*comarea,fontnumber,fontname,fontinformation*);

PFONTNUM

Accepts a *fontname* and returns the corresponding *fontnumber*, as defined in the environment file.

Syntax

	LA	BA	I
PFONTNUM	(<i>comarea</i> ,	<i>fontname</i> ,	<i>fontnumber</i>)
	in/out	in	out

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- fontname* Byte array used to supply the name of the character font. This array can be up to 20 characters long, beginning with a letter and terminated by a blank, an EOS, or the 20th character. In BASIC, the length is determined from the array itself.
- fontnumber* Integer variable in which the font number is returned.

Discussion

If the *fontname* begins with a blank or an EOS (or its length is 0 in BASIC), the number of the current primary character font is returned.

The *fontnumber* is used to refer to the character font in the PUSEFONT, PFONTINFO and PSTRINGWIDTH intrinsics.

PFONTNUM

Example

```
COBOL II:  DATA DIVISION.

           01  fontname                pic X(16).
           01  fontnumber             pic S9(4) comp.

PROCEDURE DIVISION.

CALL INTRINSIC "PFONTNUM" USING comarea,
                               fontname,
                               fontnumber.

BASIC:    500 CALL PFONTNUM (C1[*],N1$,N2)

FORTRAN:  CALL PFONTNUM (comarea,fontname,fontnumber)

SPL:     PFONTNUM (comarea,fontname,fontnumber);

PASCAL:  PFONTNUM (comarea,fontname,fontnumber);
```

PINITDEVICE

Initializes the *comarea*, and identifies the programming language used, the system output file number, and the output device.

Syntax

```
PINITDEVICE (LA      I      I      I      BA
              in/out   in     in     in     in
```

Parameters

comarea Logical array containing information for internal use. The first word returns the status of the intrinsic call.

arealength Integer variable used to specify how much space in 16 bit words you have supplied for the *comarea*. This must be at least as much space as indicated on the IFS/3000 Main Menu for the environment file you are using.

language Integer variable specifying which programming language you are using. This is necessary for byte arrays to be handled properly. *language* must have a value of 0, 1, 2, 3 or 5.

- 0 = COBOL I, or COBOL II if not using CALL INTRINSIC statement
- 1 = BASIC
- 2 = FORTRAN
- 3 = SPL or COBOL II with the CALL INTRINSIC statement, or TRANSACT/3000
- 5 = PASCAL

PINITDEVICE

filenumber

Integer variable used to supply the system file number of the output file opened for the device class of the device you are using. This file must also be associated with an environment file. You can get the system file number as follows:

COBOL:	Formal file designator from the SELECT clause.
BASIC:	SFN (BASIC file number)
FORTRAN:	FNUM (FORTRAN file number)
SPL:	The number returned from FOPEN
PASCAL:	FNUM (PASCAL file identifier)

If the file number is set to -1, this indicates that no spoolfile is open, and only the PCONVERTFIGURE and PFIGUREINFO intrinsics can be used.

outputdevice

Byte array used to supply the device name of the output device. This array can be up to 7 characters long, and must be terminated with a blank or EOS. In BASIC, the length is determined from the array itself.

Discussion

PINITDEVICE supplies IFS/3000 with the length of the *comarea*, the language you are using, and the output file number in the same way as PINITIZE. The difference is that PINITDEVICE also specifies the output device and allows you to set the file number to -1 which indicates that you do not require an open spoolfile. Without an open spoolfile, you cannot use any of the intrinsics that produce printed output. However, you can access the PCONVERTFIGURE and PFIGUREINFO intrinsics.

Depending on your requirements, you should call either PINITIZE or PINITDEVICE once before calling any other intrinsic. Refer to the discussion of PINITIZE in this section.

Example

```
COBOL II:  ENVIRONMENT DIVISION.  
           CONFIGURATION SECTION.  
           INPUT-OUTPUT SECTION.
```

PINITDEVICE

FILE-CONTROL.

SELECT OUT ASSIGN TO "OUTPUT".

DATA DIVISION.

File Section.

FD PrintFile

label records are omitted.

01 PrintFileRec pic X(132).

WORKING-STORAGE SECTION.

77 *language*. pic S9(4) comp
value 3.

77 *comarelength* pic S9(4) comp
value 5000.

77 *outputdevice* pic X(7)
value "2680A ".

PROCEDURE DIVISION.

CALL INTRINSIC "PINITDEVICE" USING *comarea*,
comarelength,
language,
PrintFile,
outputdevice.

BASIC: 790 REM Get the system file number for the file to which
800 REM we assigned the number "1" and set the *language* to
810 REM 1 (BASIC)
820 A=5000,L=1,F=SFN(1),O\$="2680A "
830 REM Initialize the *comarea*
840 REM
850 CALL PINITDEVICE (C1[*],A,L,F,O\$)

FORTTRAN: Integer *comarelength,language,filenumber*
character *7 *outputdevice*
comarelength=5000
language=2

PINITDEVICE

```
FileNum=FOPEN(OutputFile, ...)
outputdevice="2680A "
CALL PINITDEVICE (comarea,comarealength,language,filenumber,
&                outputdevice)
```

```
SPL:      Integer comarealength,language,filenumber;
          Byte Array outputdevice;
          comarealength:=5000;
          language:=3;
          filenumber:=FOPEN(OutputFile, ...);
          outputdevice:="2680A ";
          PINITDEVICE (comarea,comarealength,language,filenumber,outputdevice);
```

```
PASCAL:   type SmallInteger = -32768..32767;
          var
            comarealength,language,filenumber: SmallInteger;
            outputdevice: packed array [1..7] of char;

          comarealength:=5000;
          language:=5;
          filenumber:=FOPEN(OutputFile, ...);
          outputdevice:="2680A ";
          PINITDEVICE (comarea,comarealength,language,filenumber,outputdevice);
```

PINITIALIZE

Initializes the *comarea*, and identifies the programming language being used, along with the system file number of the output file.

Syntax

```
PINITIALIZE (comarea, arealength, language, filenumber)
             LA      I      I      I
             in/out  in     in     in
```

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- arealength* Integer variable specifying how much space in 16 bit words you have supplied for the *comarea*. This must be at least as much space as indicated on the IFS/3000 Main Menu for the environment file you are using.
- language* Integer variable specifying which programming language you are using. This is necessary for byte arrays to be handled properly. (COBOL passes byte arrays with word addresses. BASIC requires a length byte to be set on return.) *language* must have a value of 0, 1, 2, 3 or 5.
- 0 = COBOL I, or COBOL II if not using the CALL INTRINSIC statement
 - 1 = BASIC
 - 2 = FORTRAN
 - 3 = SPL or COBOL II with the CALL INTRINSIC statement
 - 5 = PASCAL
- filenumber* The system file number of the output file opened for the device class of the Laser Printer. This file must also be associated with an environment file. You

PINITIALIZE

can get the system file number as follows:

COBOL:	Formal file designator from the SELECT clause
BASIC:	SFN (BASIC file number)
FORTRAN:	FNUM (FORTRAN file number)
SPL:	The number returned from FOPEN
PASCAL:	FNUM (PASCAL file identifier)

Discussion

PINITIALIZE or PINITDEVICE must be called once before any other intrinsic is called. Refer to the description of PINITDEVICE in this section.

Each language that uses the IFS/3000 intrinsics calls them with the same parameters which are essentially of the same type and size. In order for the intrinsics to adjust to any peculiarities of the calling language, you must specify the language of the calling program in PINITIALIZE.

Example

```
COBOL II:  ENVIRONMENT DIVISION.
           CONFIGURATION SECTION.
           INPUT-OUTPUT SECTION.
           FILE-CONTROL.
             SELECT OUT ASSIGN TO "OUTPUT".
           DATA DIVISION.
           File Section.
           FD  PrintFile
             label records are omitted.
           01  PrintFileRec          pic X(132).
           WORKING-STORAGE SECTION.
           77  language             pic S9(4) comp
                                     value 3.
           77  comarealength       pic S9(4) comp
                                     value 5000.
           PROCEDURE DIVISION.
```

PINITIALIZE

```
CALL INTRINSIC "PINITIALIZE" USING comarea,  
                                   comarealength,  
                                   language,  
                                   PrintFile.
```

```
BASIC:    790 REM Get the system file number for the file to which  
          800 REM we assigned the number "1" and set the language to  
          810 REM 1 (BASIC)  
          820 A=5000,L=1,F=SFN(1)  
          830 REM Initialize the comarea  
          840 REM  
          850 CALL PINITIALIZE (C1[*],A,L,F)
```

```
FORTRAN:  Integer comarealength,language,filenumber  
          comarealength=5000  
          language=2  
          filenumber=FOPEN(OutputFile, ...)  
          CALL PINITIALIZE (comarea,comarealength,language,filenumber)
```

```
SPL:      Integer comarealength,language,filenumber;  
          comarealength:=5000;  
          language:=3;  
          filenumber:=FOPEN(OutputFile, ...);  
          PINITIALIZE (comarea,comarealength,language,filenumber);
```

```
PASCAL:   type  
          SmallInteger = -32768..32767;  
          var  
          comarealength,language,filenumber : SmallInteger;  
  
          comarealength:=5000;  
          language:=5;  
          filenumber:=FOPEN(OutputFile, ...);  
          PINITIALIZE (comarea,comarealength,language,filenumber);
```

PLOADRASTER

Loads a partitioned raster image into Laser Printer memory.

Syntax

```
          LA      BA      I
PLOADRASTER (comarea,filename,imagenumber)
           in/out   in      in
```

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- filename* Byte array used to supply the name of the partitioned raster image file created with the PCONVERTFIGURE intrinsic. This array can be up to 35 characters to allow for a fully qualified MPE filename. It must be terminated by a blank, an EOS, or the 35th character, except in BASIC.
- imagenumber* Integer variable used to supply an identification number for the partitioned raster image. *imagenumber* must have a value between 0 and 31, inclusive. *imagenumber* is used later with the PPRINTRASTER or PDELETERASTER intrinsic.

Discussion

A maximum of 32 permanent raster images can be loaded into Laser Printer memory. This maximum is affected by the size of each image. Very large images occupy more of the printer memory, and the number 32 is an absolute maximum. For the raster filename, the MPE temporary file domain is searched first, followed by the permanent file domain.

PLOADRASTER

If you call PLOADRASTER a second time and again supply the same partitioned raster image number without having deleted the image, the previously loaded image is changed from permanent to temporary. This means that it is deleted from Laser Printer memory after it has been printed. All future references to this partitioned raster image number will refer to the newly loaded image.

Example

COBOL II: DATA DIVISION.

```
01 rasterfilename                pic X(35).  
01 imagenumber                   pic S9(4) comp.
```

PROCEDURE DIVISION.

```
CALL INTRINSIC "PLOADRASTER" USING comarea,  
                                   rasterfilename,  
                                   imagenumber.
```

BASIC: 800 CALL PLOADRASTER (C1[*],R1\$,I1)

FORTRAN: CALL PLOADRASTER (*comarea*,*rasterfilename*,*imagenumber*)

SPL: PLOADRASTER (*comarea*,*rasterfilename*,*imagenumber*);

PASCAL: PLOADRASTER (*comarea*,*rasterfilename*,*imagenumber*);

PLOGPAGEINFO

Returns information about a particular logical page.

Syntax

	LA	I	LA
PLOGPAGEINFO	(<i>comarea</i> ,	<i>pagenumber</i> ,	<i>pageinformation</i>)
	in/out	in	out

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- pagenumber* Integer variable used to supply the number of the logical page. *pagenumber* must have a value between 0 and 31 inclusive. To specify the current logical page, you can use -1 for this parameter.
- pageinformation* Logical array used to return information about the indicated logical page. *pageinformation* requires a minimum of 30 words. The contents of this array are presented in Figure 5-8 on the next page.

PLOGPAGEINFO

WORD	MEANING
0	ORIENTATION (0, 90, 180, 270)
1	LOGICAL HEIGHT (in dots)
2	LOGICAL WIDTH (in dots)
3	DISTANCE FROM LEFT TO EDGE OF PAPER (in dots)
4	DISTANCE FROM TOP TO EDGE OF PAPER (in dots)
5	LEFT MARGIN (in dots)
6	ACTIVE? 0 = NO, -1 = YES
7	DOES THIS PAGE HAVE A VFC? 0 = NO, -1 = YES
8	HORIZONTAL CHARACTER SPACING (in dots)
9	VERTICAL LINE SPACING (in dots)
10	BASELINE OF BASE CHARACTER FONT (in dots)
11	IF NO VFC: TOP MARGIN (in dots)
12	Fractional Character Spacing (0,1,2,3)
13	Fractional Line Spacing (0,1,2,3)
14-29	RESERVED FOR SYSTEM USE

The value -1 is equivalent to the octal %177777.

Figure 5-8. Contents of *pageinformation* Array

PLOGPAGEINFO

Discussion

You specify vertical line spacing using the Logical Page Menu. Your specifications may result in line spacing that is a fractional number of dots between lines. In the case of a fractional dot line spacing, the HP Laser Printer provides spacing that on the average equals the value specified in the Logical Page Menu.

In addition to vertical spacing, the horizontal spacing between characters may also be fractions of a dot. For horizontal character spacing, PLOGPAGEINFO returns the integer value in word 8 and the fractional value in word 12. For vertical spacing, PLOGPAGEINFO returns the integer value in word 9 and the fractional value in word 13. The valid values for the fractional values are 0, 1, 2, and 3 representing 0/4, 1/4, 2/4, and 3/4 of a dot.

Example

COBOL II: DATA DIVISION.

```
01 pagenumber                pic S9(4) comp.  
01 pageinformation           pic S9(4) comp  
                                occurs 30 times.
```

PROCEDURE DIVISION.

```
CALL INTRINSIC "PLOGPAGEINFO" USING comarea,  
                                     pagenumber,  
                                     pageinformation.
```

BASIC: 450 CALL PLOGPAGEINFO (C1[*],L1,L2[*])

FORTRAN: CALL PLOGPAGEINFO (*comarea*,*pagenumber*,*pageinformation*)

SPL: PLOGPAGEINFO (*comarea*,*pagenumber*,*pageinformation*);

PASCAL: PLOGPAGEINFO (*comarea*,*pagenumber*,*pageinformation*);

PMOVEPENABS

Moves the Laser Printer "pen" relative to the upper left hand corner of the logical page.

Syntax

```
LA      I I
PMOVEPENABS (comarea,x,y)
           in/out in in
```

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- x* Integer variable which supplies the x-coordinate of the position on the logical page to which you want to move the Laser Printer pen. *x* is expressed in dots relative to the left edge of the logical page, where the logical page is viewed in the direction of its orientation. The value must be greater than or equal to zero.
- y* Integer variable used to supply the y-coordinate of the position on the logical page to which you want to move the Laser Printer pen. *y* is expressed in dots relative to the top edge of the logical page, where the logical page is viewed in the direction of its orientation. The value must be greater than or equal to zero.

Discussion

It is helpful to visualize that the Laser Printer writes with an imaginary "pen". When determining where you want to move the pen, be sure to view the logical page in the direction of its orientation, as specified on the IFS/3000 Logical Page Menu. Then visualize a coordinate system with the origin in the upper left corner of

PMOVEPENABS

the logical page. To identify where you want to move the pen, specify the x and y coordinates in number of dots using positive integers. This new pen location is where the character cell left edge/baseline will start.

Example

```
COBOL II:      DATA DIVISION.

                01  x                                pic S9(4) comp.
                01  y                                pic S9(4) comp.

                PROCEDURE DIVISION.

                CALL INTRINSIC "PMOVEPENABS" USING comarea,
                                                    x,
                                                    y.

BASIC:         650 CALL PMOVEPENABS (C1[*],X,Y)

FORTRAN:       CALL PMOVEPENABS (comarea,x,y)

SPL:          PMOVEPENABS (comarea,x,y);

PASCAL:       PMOVEPENABS (comarea,x,y);
```

PMOVEPENREL

Moves the Laser Printer "pen" relative to its current position.

Syntax

```
LA      I I
PMOVEPENREL (comarea,x,y)
           in/out in in
```

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- x* Integer variable used to supply the x-coordinate of the position on the logical page to which you want to move the Laser Printer pen. *x* is expressed in dots relative to the pen's current position on the logical page, where the logical page is viewed in the direction of its orientation. Positive values move the pen to the right; negative values move the pen to the left.
- y* Integer variable used to supply the y-coordinate of the position on the logical page to which you want to move the Laser Printer pen. *y* is expressed in dots relative to the pen's current position on the logical page, where the logical page is viewed in the direction of its orientation. Positive values move the pen down; negative values move the pen up.

Discussion

PMOVEPENREL is similar to PMOVEPENABS except that PMOVEPENREL moves the pen relative to its current position. Also, PMOVEPENREL allows you to enter negative as well as positive values for x- and y-coordinates.

PMOVEPENREL

Positive values move the pen to the right and down for x and y respectively; negative values move it left and up. Refer to the discussion of PMOVEPENABS.

Example

```
COBOL II:      DATA DIVISION.

                01  x                                pic S9(4) comp.
                01  y                                pic S9(4) comp.

                PROCEDURE DIVISION.

                CALL INTRINSIC "PMOVEPENREL" USING comarea,
                                                    x,
                                                    y.

BASIC:         650 CALL PMOVEPENREL (C1[*],X,Y)

FORTRAN:       CALL PMOVEPENREL (comarea,x,y)

SPL:          PMOVEPENREL (comarea,x,y);

PASCAL:       PMOVEPENREL (comarea,x,y);
```

PNEWFORM

Selects the form you want to write to.

Syntax

```
LA      BA
PNEWFORM (comarea,formname)
         in/out   in
```

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- formname* Byte array used to supply the name of the form you want to write to. The form must be on the current logical page. *formname* can be up to 16 characters long beginning with a letter and terminated by a blank, an EOS, or the 16th character. In BASIC, the length is determined from the array itself. It is the name assigned to the form when it was designed in IDIFORM.

Discussion

This intrinsic is used only when there is more than one form on the current logical page. You use PNEWFORM to select which of the forms you want to write to.

You may not have to call PNEWFORM if you have called PNEWPHYSPAGE or PNEWPAGE. If there is a form on the current logical page, PNEWPHYSPAGE and PNEWPAGE automatically make it the current form.

You can also use the special values "1" and "2" for the *formname* to specify either form 1 or form 2 on the logical page. Forms 1 and 2 are specified on the Logical Page Forms Menu in IFS/3000. If you want to get the names of the forms, you can use PSTATEINFO.

PNEWFORM

Multi-copy forms can be accessed using PNEWFORM. A multi-copy form is made current automatically if it is the only form on the current logical page.

Example

```
COBOL II:          DATA DIVISION.

                   01 formname                pic X(16).

                   PROCEDURE DIVISION.

                   CALL INTRINSIC "PNEWFORM" USING comarea,
                                                    formname.

BASIC:            350 CALL PNEWFORM (C1[*],F1$)

FORTRAN:          CALL PNEWFORM (comarea,formname)

SPL:              PNEWFORM (comarea,formname);

PASCAL:           PNEWFORM (comarea,formname);
```

PNEWPAGE

Advances to the next active logical page.

Syntax

```
LA  
PNEWPAGE (comarea)  
in/out
```

Parameters

comarea Logical array containing information for internal use. The first word returns the status of the intrinsic call.

Discussion

PNEWPAGE is used to advance to the next active logical page. If you are already on the last logical page, PNEWPAGE causes a physical page eject and takes you to the first active logical page again. The pages are cycled through in order of the logical page number as assigned in IFS/3000.

When you call PNEWPAGE, it keeps track of which logical page you are printing and passes that information to other intrinsics for error checking and forms control. For example, if you call PNEWPAGE followed by PNEWFORM, PNEWFORM verifies that the form you want to print is on the current logical page. If it is not, PNEWFORM issues an error message. Also, PNEWPAGE automatically makes the first form on the active logical page the current form. This usually eliminates the need to call PNEWFORM since the first form is the first form you normally want to write to. PNEWPAGE also prints any forms on an active logical page before advancing to the next physical page.

If only one form is accessible from the logical page, it is made the current form. If multiple forms are accessible, PNEWFORM selects form 1.

If no forms exist on the logical page, and this is the page to be used with multi-copy forms, the first form for the first copy in the multi-copy form table is selected. If there is no form for the first copy, PNEWPAGE looks for a form in each successive copy until a form is found or until it reaches the last copy.

Example

```
COBOL II:          DATA DIVISION.

                   PROCEDURE DIVISION.

                   CALL INTRINSIC "PNEWPAGE" USING comarea.

BASIC:            220 CALL PNEWPAGE (C1[*])

FORTRAN:          CALL PNEWPAGE (comarea)

SPL:              PNEWPAGE (comarea);

PASCAL:           PNEWPAGE (comarea);
```

PNEWPHYSPAGE

Advances to the next physical page.

Syntax

```
La  
PNEWPHYSPAGE (comarea)  
in/out
```

Parameters

comarea

Logical array containing information for internal use. The first word returns the status of the intrinsic call.

Discussion

PNEWPHYSPAGE advances to the next physical page and prints the first active logical page.

PNEWPHYSPAGE determines the current form using the same method as PNEWPAGE. If only one form is accessible from the logical page, it is made the current form. If multiple forms are accessible, PNEWPHYSPAGE selects form 1 if it is on this logical page; otherwise, form 2 is selected.

If no forms exist on the logical page, and this is the page to be used with multi-copy forms, the first form for the first copy in the multi-copy form table is selected. If there is no form for the first copy, PNEWPHYSPAGE looks for a form in each successive copy until a form is found or until it reaches the last copy.

Example

COBOL II: DATA DIVISION.
 PROCEDURE DIVISION.
 CALL INTRINSIC "PNEWPHYSPAGE" USING *comarea*.

BASIC: 220 CALL PNEWPHYSPAGE (C1[*])

FORTRAN: CALL PNEWPHYSPAGE (*comarea*)

SPL: PNEWPHYSPAGE (*comarea*);

PASCAL: PNEWPHYSPAGE (*comarea*);

PNEWSUBFORM

Selects the current subform you want to write to.

Syntax

	LA	BA
PNEWSUBFORM	(<i>comarea</i> , <i>subformname</i>)	
	in/out	in

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- subformname* Byte array used to supply the name of the current subform you want to write to. This array can be up to 16 characters long, and must be terminated by a blank, an EOS, or the 16th character. In BASIC, the length is determined from the array itself.

Discussion

PNEWSUBFORM is optional when you are writing data into a form containing multiple subforms, but is required when you are writing data into a form containing duplicate field names.

When you write data programmatically into a form with the PWRITEFIELD intrinsic, you must provide the name of the field you want to write to. PNEWSUBFORM directs PWRITEFIELD to the appropriate subform when duplicate field names exist.

For a form with many subforms, using PNEWSUBFORM can result in a performance improvement since it restricts the fields searched to those within a particular subform only. An exception to this is when the

PNEWSUBFORM

subform name begins with a blank or EOS or has length zero in BASIC. In this case, all the fields in the entire form are searched for the *fieldname* you indicated in PWRITEFIELD.

Fields are always defined within subforms. You can have duplicate field names within the same form, as long as the fields are not in the same subform. When the same name is used for more than one field in a form, you must call PNEWSUBFORM first to identify which subform contains the field, and then call PWRITEFIELD to write to the field. Since the field names are stored in no particular order, PWRITEFIELD uses the first field it encounters with that name unless you identify the appropriate subform.

Example

```
COBOL II:      DATA DIVISION.

                01 subformname          pic X(16).

                PROCEDURE DIVISION.

                CALL "PNEWSUBFORM" USING comarea,
                                         subformname.

BASIC:         210 CALL PNEWSUBFORM (C1[*],S1$)

FORTRAN:       CALL PNEWSUBFORM (comarea,subformname)

SPL:          PNEWSUBFORM (comarea,subformname);

PASCAL:       PNEWSUBFORM (comarea,subformname);
```

PPRINTFIGURE[A]

Converts a figure in a figure file to a partitioned raster image file, then prints the partitioned raster image. The ASCII version, callable from COBOL, is denoted by an "A" after the intrinsic name.

Syntax

```
PPRINTFIGURE (LA      BA      BA      BA
              in/out   in      in      in
              R      I      I      R      R
              imageheight,units,imagerotation,xposition,yposition,
              in      in      in      in      in
              I      I
              positionmode,rasterimagetype)
              in      in
```

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- figurefilename* Byte array used to supply the name of the figure file that contains the figure you want to convert. This array can be up to 35 characters long to allow for a fully qualified MPE filename. If the filename is less than 35 characters, it must be terminated by a blank or an EOS. In BASIC a blank or EOS is not necessary since the length is determined from the array itself.
- figurename* Byte array used to supply the name of the figure within the specified figure file that you want to convert. This array can be up to 16 characters (upper or lower case) beginning with a letter, and cannot include any embedded blanks.

PPRINTFIGURE[A]

If the *figurename* is less than 16 characters, it must be terminated by a blank or an EOS. In BASIC a blank or EOS is not necessary since the length is determined from the array itself.

rasterfilename

Byte array used to supply the name of the partitioned raster file that will contain the converted figure. This array can be up to 35 characters to allow for a fully qualified MPE file name. It must be terminated by a blank, an EOS, or the 35th character, except in BASIC. This parameter is optional; if it is not used, you must leave at least one blank in this position. The figure is then converted to a temporary partitioned raster file called OUT2680A or OUT2688A.

imageheight

Real number used to supply the height of the partitioned raster image, viewed in the direction of its orientation, and given in the units specified in the *units* parameter. The width of the partitioned raster image is automatically calculated from the height. For COBOL, the value is passed as an ASCII string. One dot is added to the *imageheight* when it is converted; that is, a specified height of 300 dots creates an image 301 dots high. You only need to take this into account if you are working with extremely precise dot measurements.

units

Integer specifying the unit of measurement to be used for the *imageheight*. *units* must have a value of 0, 1, 2 or 3.

- 0 = Dots
- 1 = Inches
- 2 = Centimeters
- 3 = Millimeters

imagerotation

Integer variable used to supply the orientation for the partitioned raster image with respect to the logical page. *imagerotation* must have a value of 0, 90, 180 or 270. For more information, refer to Section 2 on Terminology. Also refer to Figure 5-9 in the Examples of this intrinsic.

xposition

Real variable used to supply the x-coordinate of the upper left corner of the raster image on the current logical page. The coordinate is measured in the units specified in the *units* parameter and is either relative or absolute as

PPRINTFIGURE[A]

specified in the *positionmode* parameter. With absolute positioning, only positive values can be used. With relative positioning, a positive value moves the image to the right; a negative value moves it to the left. For COBOL, the value is passed as an ASCII string.

yposition

Real variable used to supply the y-coordinate of the upper left corner of the raster image on the current logical page. The coordinate is measured in the units specified in the *units* parameter and is either relative or absolute as specified in the *positionmode* parameter. With absolute positioning, only positive values can be used. With relative positioning, a positive value moves the image down; a negative value moves it up. For COBOL, the value is passed as an ASCII string.

For more information on positioning, refer to Section 2 on Terminology and Figure 5-9 in the Examples of this intrinsic.

positionmode

Integer variable used to specify whether the x- and y-coordinates are relative or absolute. The relative option positions the image relative to the current pen location on the logical page; the absolute option positions the image at the absolute location on the current page. *positionmode* mode must have a value of 0 or 1.

0 = Relative
1 = Absolute

rasterimagetype

Integer variable used to specify whether the image is permanent or temporary. *rasterimagetype* must have a value of 0 or 1.

0 = Temporary
1 = Permanent

A temporary partitioned raster image is deleted immediately after printing; a permanent image remains in Laser Printer memory for future use.

PPRINTFIGURE[A]

Discussion

PPRINTFIGURE allows you to specify the partitioned raster image to be printed in one of two ways: either with or without the partitioned raster filename as a parameter.

If you use the partitioned raster filename, the partitioned raster image can be printed directly. PPRINTFIGURE compares the rotation, size and creation date of the partitioned raster image with those of the figure in the figure file. If the rotation or size do not match or if the figure date is more recent than the partitioned raster image date, a new raster image file is created. This new raster image file replaces the file you specified and is given the same name. Specifying an existing raster filename can save time and system resources since there is no need to reconvert a figure that has not been changed.

If you do not specify a partitioned raster filename, PPRINTFIGURE converts the figure to a partitioned raster image file, prints the image, and places it in a temporary file.

PPRINTFIGURE also deletes raster images from Laser Printer memory on a least recently used basis. This occurs when there is not enough space to load another image into Laser Printer memory.

Example

```
COBOL II:  DATA DIVISION.

           01  figurefilename                pic X(35).
           01  figurename                   pic X(16).
           01  rasterfilename               pic X(35).
           01  imageheight                  pic X(16).
           01  units                         pic S9(4) comp.
           01  imagerotation                pic S9(4) comp.
           01  xposition                    pic X(16).
           01  yposition                    pic X(16).
           01  positionmode                 pic S9(4) comp.
           01  imagetype                    pic S9(4) comp.

           PROCEDURE DIVISION.
```

PPRINTFIGURE[A]

```
CALL INTRINSIC "PPRINTFIGUREA" USING comarea,  
figurefilename,  
figurename,  
rasterfilename,  
imageheight,  
units,  
imagerotation,  
xposition,  
yposition,  
positionmode,  
imagetype.
```

BASIC: 500 CALL PPRINTFIGURE (C1[*],F1\$,N1\$,R1\$,H,U,R,X,Y,M,T)

FORTRAN: CALL PPRINTFIGURE (*comarea,figurefilename,figurename,rasterfilename,*
& *imageheight,units,imagerotation,xposition,yposition,*
& *positionmode,imagetype)*

SPL: PPRINTFIGURE (*comarea,figurefilename,figurename,rasterfilename,*
 imageheight,units,imagerotation,xposition,yposition,
 positionmode,imagetype);

PASCAL: PPRINTFIGURE (*comarea,figurefilename,figurename,rasterfilename,*
 imageheight,units,imagerotation,xposition,yposition,
 positionmode,imagetype);

PPRINTFIGURE[A]

Figure 5-9 shows figures printed at 0 and 90 degree rotations on different logical page orientations with the PPRINTFIGURE intrinsic. The logical page dimensions are 10 x 7 inches and the figure is 5 inches high. The rotation of the figure is specified with respect to the logical page.

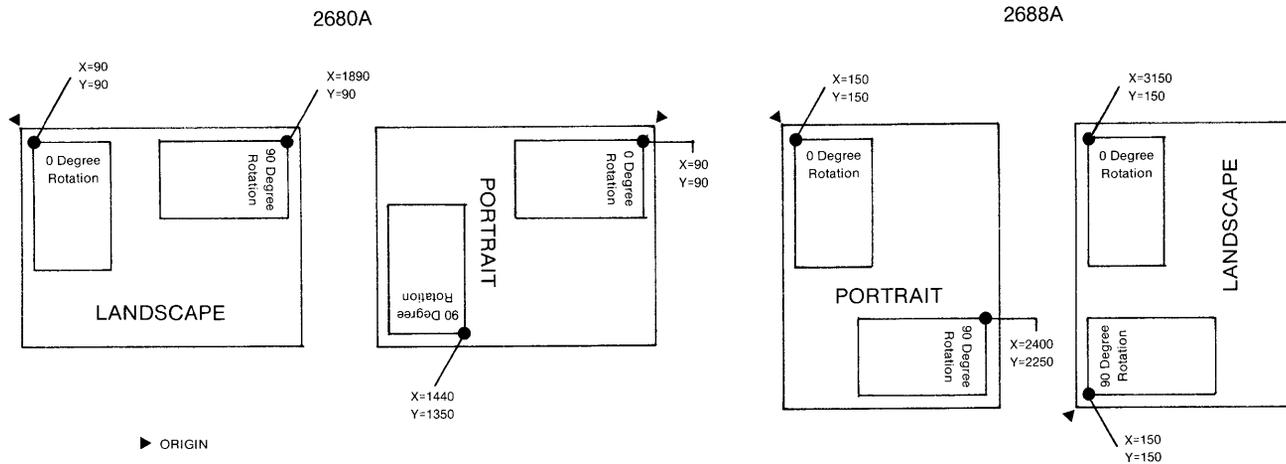


Figure 5-9. Figures printed with the PPRINTFIGURE intrinsic

PPRINTRASTER[A]

Prints a partitioned raster image which has been loaded into Laser Printer memory. The ASCII version, callable from COBOL, is denoted by an "A" after the intrinsic name.

Syntax

	LA	I	R	R	I	I
PPRINTRASTER	(<i>comarea</i> ,	<i>imagenumber</i> ,	<i>xposition</i> ,	<i>yposition</i> ,	<i>units</i> ,	<i>positionmode</i>)
	in/out	in	in	in	in	in

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- imagenumber* Integer variable used to supply the number of the partitioned raster image to be printed, as specified in the PLOADRASTER intrinsic. *imagenumber* must have a value between 0 and 31 inclusive.
- xposition* Real variable used to supply the x-coordinate of the upper left corner of the raster image on the current logical page. The coordinate is measured in the units specified in the *units* parameter and is either relative or absolute as specified in the *positionmode* parameter. With absolute positioning, only positive values can be used. With relative positioning, a positive value moves the image to the right; a negative value moves it to the left. For COBOL, the value is passed as an ASCII string.
- yposition* Real variable used to supply the y-coordinate of the upper left corner of the raster image on the current logical page. The coordinate is measured in the units specified in the *units* parameter and is either relative or absolute as specified in the *positionmode* parameter. With absolute positioning, only positive values can be used. With relative positioning, a positive value moves

PPINTRASTER[A]

the image down; a negative value moves it up. For COBOL, the value is passed as an ASCII string.

For more information on positioning, refer to Section 2 on Terminology and to Figure 5-10 in the Examples of this intrinsic.

units

Integer variable used to supply the units of measurement for the *x*- and *y*position parameters. *units* must have a value of 0, 1, 2 or 3.

- 0 = Dots
- 1 = Inches
- 2 = Centimeters
- 3 = Millimeters

positionmode

Integer variable used to specify whether the *x*- and *y*-coordinates are relative or absolute. The relative option positions the image relative to the current pen location on the logical page; the absolute option positions the image at the absolute location on the current page. *positionmode* must have a value must be 0 or 1.

- 0 = Relative
- 1 = Absolute

Discussion

PPINTRASTER prints a partitioned raster image on the Laser Printer. The image must already have been loaded with the PLOADRASTER intrinsic.

By allowing you to specify the exact location of the partitioned raster image, PPRINTRASTER gives you complete control over its placement on the logical page. The image can be positioned relative to the current location of the pen, or at an absolute location on the logical page.

The values of the *x*- and *y*-coordinates depend on the size of the current logical page. For absolute positioning, only positive values can be used for *x* and *y*. The value for *x* moves the image to the right, the

PPRINTRASTER[A]

value for y moves it down. For relative positioning, negative values can be used to move the image left and up, for x and y respectively.

X and Y positions are checked to insure that they lie within the boundaries of the logical page, but no checking is done to verify that the entire image fits on the physical page.

The position of the Laser Printer "pen" is unaffected by printing a partitioned raster image. This means that the pen remains where it was prior to printing the image and does not move to a new location.

Example

COBOL II: DATA DIVISION.

```
01 Figure-Number           pic S9(4) comp.
01 xposition                pic X(16).
01 yposition                pic X(16).
01 units                   pic S9(4) comp.
01 positionmode           pic S9(4) comp.
```

PROCEDURE DIVISION.

```
CALL INTRINSIC "PPRINTRASTERA" USING comarea,
                                     imagenumber,
                                     xposition,
                                     yposition,
                                     units,
                                     positionmode.
```

BASIC: 900 CALL PPRINTRASTER (C1[*],I1,X,Y,U,P)

FORTRAN: CALL PPRINTRASTER (*comarea*,*imagenumber*,*xposition*,*yposition*,*units*,
& *positionmode*)

SPL: PPRINTRASTER (*comarea*,*imagenumber*,*xposition*,*yposition*,*units*,
positionmode);

PPRINTRASTER[A]

PASCAL: PPRINTRASTER (*comarea,imagenumber,xposition,yposition,units,positionmode*);

Figure 5-10 shows partitioned raster images printed at 0 and 90 degree rotations on different logical page orientations with the PPRINTRASTER intrinsic. The logical page dimensions are 10 x 7 inches and the raster image is 5 inches high. The rotation of the image was assigned with the PCONVERTFIGURE intrinsic and is specified with respect to the physical page.

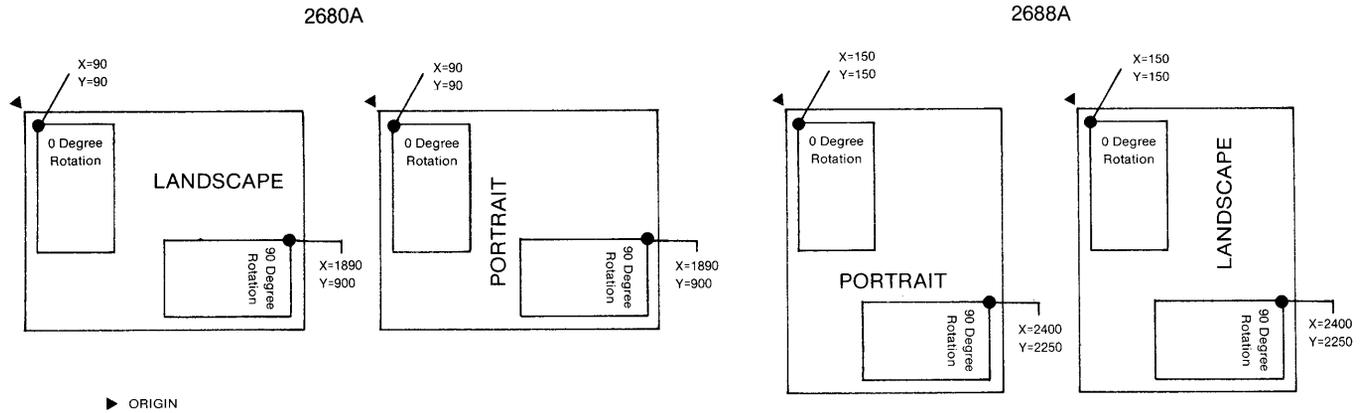


Figure 5-10. Figures printed with the PPRINTRASTER intrinsic

PSELECTPAGE

Activates and advances to the selected logical page. Optionally performs a physical page eject.

Syntax

```
PSELECTPAGE (LA          I          I
             in/out      in          in)
```

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- logicalpagenumber* Integer variable used to supply the number of the logical page you want to activate and print. *logicalpagenumber* must have a value between 0 and 31.
- ejectoption* Integer variable used to specify an optional physical page eject, where 0 = no physical page eject and 1 = physical page eject.

Discussion

The Laser Printer manages logical pages by number and determines which page to print next by cycling through the logical page numbers of active logical pages. PSELECTPAGE enables you to activate any logical page in the environment, in any order. The logical page selected remains activated and will then print in numerical order as the Laser Printer continues cycling through the active logical pages. Using the physical page eject option allows printing of one logical page per physical page even though other logical pages are active.

PSELECTPAGE

Example

```
COBOL II:  DATA DIVISION.

           01  logicalpagenumber                pic S9(4) comp.
           01  ejectoption                      pic S9(4) comp.

PROCEDURE DIVISION.

CALL INTRINSIC "PSELECTPAGE" USING comarea,
                                   logicalpagenumber,
                                   ejectoption.

BASIC:    400 CALL PSELECTPAGE (C1[*],L1,E1)

FORTRAN:  CALL PSELECTPAGE (comarea,logicalpagenumber,ejectoption)

SPL:      PSELECTPAGE (comarea,logicalpagenumber,ejectoption);

PASCAL:   PSELECTPAGE (comarea,logicalpagenumber,ejectoption);
```

PSTATEINFO

Returns information about the current state of the print job.

Syntax

	LA	BA	BA	LA
PSTATEINFO	(<i>comarea</i> ,	<i>formname</i> ,	<i>subformname</i> ,	<i>stateinformation</i>)
	in/out	out	out	out

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- formname* Byte array used to return the name of the current form on the current logical page. This array must be at least 16 characters long. The form name is returned with all letters in upper case and with blanks padded out to 16 bytes. In BASIC, the array is returned with its length set to the number of nonblank characters in the form name. If there is no form name, all blanks are returned in COBOL, SPL, FORTRAN and PASCAL, and in BASIC a length of zero is returned.
- subformname* Byte array used to return the name of the current subform in the current form. This array must be at least 16 characters long. The subform name is returned with all letters in upper case and with blanks padded out to 16 bytes. In BASIC, the array is returned with its length set to the number of nonblank characters in the subform name. If there is no subform name, all blanks are returned in COBOL, SPL, FORTRAN and PASCAL, and in BASIC a length of zero is returned.
- stateinformation* Logical array used to return information about the current state of the print job. This array must be at least 30 words long. The contents of this array are:

WORD	CONTENTS
0	CURRENT LOGICAL PAGE NUMBER
1	CURRENT PRIMARY CHARACTER FONT NUMBER
2	CURRENT SECONDARY CHARACTER FONT NUMBER
3	PHYSICAL PAGE WIDTH IN DOTS (as defined in IFS/3000)
4	PHYSICAL PAGE LENGTH IN DOTS (as defined in IFS/3000)
5-29	RESERVED FOR SYSTEM USE

Figure 5-11. Contents of *stateinformation* Array

Discussion

The physical page dimensions returned in words 3 and 4 of the *stateinformation* parameter are those you defined with the IFS/3000 Physical Page Menu. If the wrong size paper is loaded in the Laser Printer and the printer did not detect the error, PSTATEINFO does not detect the error either. The physical page length is measured in the direction of paper motion. The physical page dimensions are included so that an intelligent text processing system can determine the apparent margins created when using logical pages which are smaller than the paper.

Example

```

COBOL II:  DATA DIVISION.

           01  formname                pic X(16).
           01  subformname            pic X(16).
           01  stateinformation
           03  CurrLogPageNum          pic S9(4) comp.

```

PSTATEINFO

03	CurrPriCharNumm	pic S9(4) comp.
03	CurrSecCharNumm	pic S9(4) comp.
03	PhysPageWidth	pic S9(4) comp.
03	PhysPageLength	pic S9(4) comp.
03	Filler	pic S9(4) comp

occurs 25 times.

PROCEDURE DIVISION.

CALL INTRINSIC "PSTATEINFO" USING *comarea*,
formname,
subformname,
stateinformation.

BASIC: 650 PSTATEINFO (C1[*],F1\$,S1\$,S2[*])

FORTRAN: CALL PSTATEINFO (*comarea*,*formname*,*subformname*,*stateinformation*)

SPL: PSTATEINFO (*comarea*,*formname*,*subformname*,*stateinformation*);

PASCAL: PSTATEINFO (*comarea*,*formname*,*subformname*,*stateinformation*);

PSTRINGWIDTH

Returns the width of a string in number of dots.

Syntax

	LA	I	BA	I	I
PSTRINGWIDTH	(<i>comarea</i> ,	<i>fontnumber</i> ,	<i>string</i> ,	<i>stringlength</i> ,	<i>stringwidth</i>)
	in/out	in	in	in	out

Parameters

<i>comarea</i>	Logical array containing information for internal use. The first word returns the status of the intrinsic call.
<i>fontnumber</i>	Integer variable used to supply the number of the character font used to print the string. You can specify the current character font by supplying a value of -1 for this parameter.
<i>string</i>	Byte array used to supply the string for which you want to know the printed width.
<i>stringlength</i>	Integer variable used to supply the length of the <i>string</i> in number of bytes. BASIC requires a dummy variable for this parameter.
<i>stringwidth</i>	Integer variable used to return the width in dots of the <i>string</i> when printed using the character font indicated.

PSTRINGWIDTH

Discussion

The character font you specify must have been defined in IFS/3000. An exception to this is font number 31 which is the Laser Printer default character font.

In determining the width of a string, PSTRINGWIDTH counts the width of leading and embedded blanks, but ignores trailing blanks. To determine the width of a blank, supply a *string* with an initial blank and a *stringlength* of 1.

Example

COBOL II: DATA DIVISION.

```
01 fontnumber                pic S9(4) comp.
01 string                    pic X(72).
01 stringlength              pic S9(4) comp.
01 stringwidth               pic S9(4) comp.
```

PROCEDURE DIVISION.

```
CALL INTRINSIC "PSTRINGWIDTH" USING comarea,
                                     fontnumber,
                                     string,
                                     stringlength,
                                     stringwidth.
```

BASIC: 220 CALL PSTRINGWIDTH (C1[*],N1,S1\$,L1,W1)

FORTRAN: CALL PSTRINGWIDTH (*comarea*,*fontnumber*,*string*,*stringlength*,
& *stringwidth*)

SPL: PSTRINGWIDTH (*comarea*,*fontnumber*,*string*,*stringlength*,*stringwidth*);

PASCAL: PSTRINGWIDTH (*comarea*,*fontnumber*,*string*,*stringlength*,*stringwidth*);

PUSEFONT

Selects primary and secondary character fonts.

Syntax

```
          LA           I           I
PUSEFONT (comarea,primaryfontnumber,secondaryfontnumber)
          in/out      in           in
```

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- primaryfontnumber* Integer variable used to supply the number of the character font you want to use as the primary character font. *primaryfontnumber* must have a value between 0 and 31.
- secondaryfontnumber* Integer variable used to supply the number of the character font you want to use as the secondary character font. *secondaryfontnumber* must have a value between 0 and 31.

Discussion

If you specify -1 as the *primary*- or *secondaryfontnumber*, the character font currently being used as the primary or secondary character font is not changed.

To determine a font number, you can use PFONTNUM.

PUSEFONT

Example

```
COBOL II:  DATA DIVISION.

           01  primaryfontnumber                pic S9(4) comp.
           01  secondaryfontnumber            pic S9(4) comp.

PROCEDURE DIVISION.

CALL INTRINSIC "PUSEFONT" USING comarea,
                                primaryfontnumber,
                                secondaryfontnumber.

BASIC:    350 CALL PUSEFONT (C1[*],P1,S1)

FORTRAN:  CALL PUSEFONT (comarea,primaryfontnumber,secondaryfontnumber)

SPL:     PUSEFONT (comarea,primaryfontnumber,secondaryfontnumber);

PASCAL:  PUSEFONT (comarea,primaryfontnumber,secondaryfontnumber);
```

PWRITEFIELD

Writes data to a field or subfield in a form.

Syntax

```
                LA      BA      I      BA      I
PWRITEFIELD (comarea,fieldname,subfieldnumber,data,datalength)
                in/out   in      in      in      in
```

Parameters

- comarea* Logical array containing information for internal use. The first word returns the status of the intrinsic call.
- fieldname* Byte array used to supply the name of the field you want to write to. The field name must have been specified in the form using `IDSFORM`. The field name can be up to 16 characters beginning with a letter and terminated by a blank, an EOS or the 16th character. In BASIC the length is determined from the array itself.
- subfieldnumber* Integer indicating which subfield you want to write to. Subfields are numbered beginning with 1. The subfield number is edited, and an error occurs if the subfield number is greater than the number of subfields in the field. Use a value of -1 when the field has only 1 subfield.
- data* Byte array containing the data you want to write into the field or subfield.
- datalength* Integer variable indicating the length in bytes of the array you supplied for the *data* parameter. In BASIC, you can use a dummy variable for this parameter.

PWRITEFIELD

Discussion

PWRITEFIELD enables you to write one or more lines of data into a single subfield. Lines must be separated by a carriage return (ASCII 13) in the data array. If the last line of data is blank, a carriage return is required after the last line; otherwise a carriage return after the last line is optional.

If there are any trailing blanks in the data array, they are stripped from each line of data before it is written to the field. Since all lines of data, even blank lines, are counted when centering the data, you may want to include blank lines in your array to correctly position the data.

Data is centered vertically and left-justified horizontally in the subfield area. There is some padding between the left edge of the subfield and the start of the data. This padding is equal to 1/3 the cell height of the character font being used for printing.

If underlines are specified for any portion of the field, they are placed in fixed positions by IDIFORM. If a form is not a print and space form, the printed data is lined up with the underlines rather than centered vertically. This means that the height of the IDIFORM graphics grid is used for vertical character spacing, so you should be careful that the character font you are using to fill in the form is appropriate for the IDIFORM grid spacing. For more information, refer to the *IDIFORM Reference Guide (part number 36581-90002)*.

The maximum length of a print line is 250 bytes, but there is no limit to the length of the data array. The length of the data array is determined by the value you supply in the *datalength* parameter. You can use trailing blanks or an EOS (character 255) to terminate a data array before the end of the array is reached.

Example

```
COBOL II:      DATA DIVISION.

                01  fieldname                pic X(16).
                01  subfieldnumber          pic S9(4) comp.
                01  FieldDataDescription.
                   03  data                  pic X(80).
                   03  FieldDataRedef redefines data.
                       05  FieldDataDetail    pic X(59).
                       05  ReturnChar         pic X value %15.
```

PWRITEFIELD

```
05 SecondData          pic X(9).
05 ReturnChar          pic X value %15.
05 ThirdData           pic X(10).
01 datalength         pic S9(4) comp.
```

PROCEDURE DIVISION.

```
CALL INTRINSIC "PWRITEFIELD" USING comarea,
                                   fieldname,
                                   subfieldnumber,
                                   data,
                                   datalength.
```

BASIC: 250 CALL PWRITEFIELD (C1[*],F1\$,S,D1\$,L)

FORTRAN: CALL PWRITEFIELD (*comarea*,*fieldname*,*subfieldnumber*,*data*,
& *datalength*)

SPL: PWRITEFIELD (*comarea*,*fieldname*,*subfieldnumber*,*data*,*datalength*);

PASCAL: PWRITEFIELD (*comarea*,*fieldname*,*subfieldnumber*,*data*,*datalength*);

THIS PAGE SHOULD BE BLANK

MPE Commands and Intrinsic

APPENDIX A

This section lists the MPE commands and intrinsic that will be useful when operating the HP Laser Printer Support Package subsystems. Your environment file and component files are disc files which reside within the HP 3000 MPE operating system domain. You can use the following MPE commands and intrinsic to associate an environment file with your data file and direct your output to the Laser Printer. In addition, you can monitor the status of your print job, verify where files reside, verify a file's code and type; and copy files between groups, account, and systems.

The commands provide non-programmatic access to the MPE operating system. The intrinsic provide programmatic access. For more information, refer to the *MPE Commands Reference Manual (part number 30000-90009)* and the *MPE Intrinsic Reference Manual (part number 30000-90010)*.

MPE Commands

:FILE

You can use the :FILE command to set a file equation specifying the Laser Printer as the output target device. With the DEV parameter, most installations will use device class PP for the 2680A and PP88 for the 2688A. Other device classes can also be used. The equation must also contain an ENV parameter which specifies an environment file to be used when printing the associated data file. The environment file can be one of those supplied by Hewlett-Packard and listed in Appendix C, or a custom environment file you designed using IFS/3000.

EDITOR

```
:FILE MYFILE;DEV=PP;ENV=PICA.HPENV.SYS
:EDITOR *MYFILE
HP32201A.7.12 EDIT/3000 MON. JUL 12, 1982, 11:09 AM
(C) HEWLETT-PACKARD CO. 1979
/T PRINTIFS
FILE UNNUMBERED
/L ALL,OFFLINE
*** OFF LINE LISTING BEGUN. ***
/EXIT
```

MPE COMMANDS AND INTRINSICS

END OF SUBSYSTEM

:

For multiple colated copies of output, specify the number of copies as the third DEV parameter as shown in the following:

```
FILE MYFILE;DEV=PP, ,3;;ENV=PICA.HPENV.SYS
```

To specify a unique paper size, add the FORMS=FORMSMSG to the file equation. FORMSMSG is a string of not more than 49 characters terminated with a period. This message will be displayed on the console and must be verified by the operator before the user can access the file.

:FCOPY

You can use the :FCOPY command to copy an environment file or component file from one group and account to another. You specify the :FCOPY command within the group and account where you want the copied file to reside. You must have previously released the file you wish to copy in the group and account where it resides or MPE security will not allow :FCOPY access to the file.

You can also use :FCOPY to copy a file to a specific device and a specific environment.

```
FCOPY
```

```
:FILE FILEOUT;DEV=PP88;ENV=LP88.HPENV.SYS
```

```
:FCOPY
```

```
HP32212A.3.15 FILE COPIER (C) HEWLETT-PACKARD CO. 1982
```

```
>FROM=PRINTIFS;TO=*FILEOUT
```

```
*200*WARNING:FROMFILE RECSIZE IS 72BYTES,TOFILE RECSIZE IS 256 BYTES.
```

```
CONTINUE OPERATION (Y OR N) ? Y
```

```
EOF FOUND IN FROMFILE AFTER RECORD 622
```

```
623 RECORDS PROCESSED *** 0 ERRORS
```

```
>EXIT
```

END OF SUBSYSTEM

:

For additional information see the *FCOPY Reference Manual* (part number 0300-90064).

:STORE

You can use the :STORE command to store an environment file or component file on tape.

```
:FILE TAPE;DEV=TAPE
:STORE filename;*TAPE
```

:RESTORE

You use the :RESTORE command to restore an environment file or component file from tape.

```
:FILE TAPE;DEV=TAPE
:RESTORE *TAPE;filename
```

:LISTF ,2

You can use the :LISTF ,2 command to display a listing of all files within you group, account, or another group or account on the same computer system. You may use the :LISTF command without the ,2 parameter, however, the main emphasis here is to display the different file code and types for the various components available to an environment file.

```
:LISTF @.CHARSETS.SYS,2
```

will list all files in the CHARSETS group of the SYS account.

:SHOWOUT

You can use the :SHOWOUT command to report the status of output device files.

MPE Intrinsic

- FCHECK** The FCHECK intrinsic returns error codes indicating errors encountered in processing the environment file, such as environment file open, close, or read errors, and uncompiled environment file error.
- FFILEINFO** You can use FFILEINFO to obtain the environment file name.
- FOPEN** You can use FOPEN to specify the Laser Printer as the output device, and to specify the environment file name.

Using Figure Files

APPENDIX B

When you purchase the IFS Graphics software (HP 36583), you can create new figure files through usage of the Figure Maker program, and update older graphics files using the Figure Update program. These two programs may be used as follows:

The Figure Maker Utility

Figure Maker is an HP3000 Graphics Utility which allows you to create your own figures in a figure file independent of any other graphics subsystem. Figure Maker is provided with the HP graphics software.

Figures created with Figure Maker are treated in the same way as figures created using HPDraw and can be used with any of the following:

- IFS/3000 graphics intrinsics
- LPS Interpreter graphics commands
- HPDraw
- TDP/3000

Figure Maker gives you a workspace 100 units square for the creation of your figures. You can move anywhere on this grid, draw lines as coordinates, and include text of various sizes as part of the figure. You determine the actual size of the figure when you use it in an application.

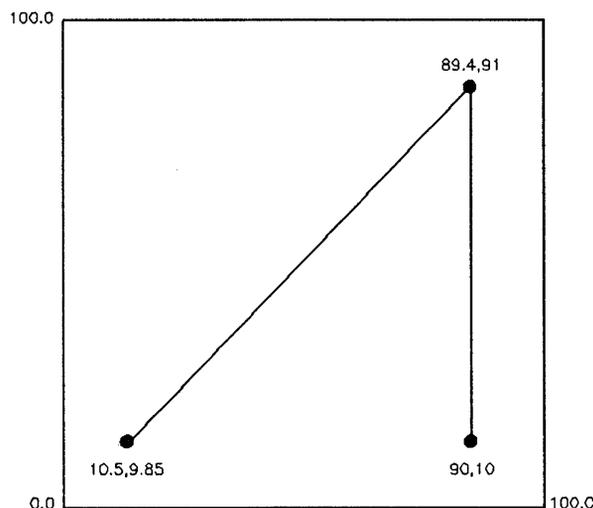


Figure B-1. Figure Maker Workspace

Converting Other Graphics Figures with Figure Maker

If you have figures created from an editor file or drawing program other than HPDraw, you can use Figure Maker to convert the file to a figure file. For example, a file of line segment endpoints can be converted into a figure in a figure file which can then be used on the HP laser printer or other graphics output devices. Since most other drawing applications generate a different type of output than Figure Maker expects as input, a short intermediate program is necessary to format the vectors properly for input into the Figure Maker.

Running Figure Maker

Users of Figure Maker require DS (Extra Data Segments) Capability. If you have any questions about your user capabilities, check with your System Manager.

After you have successfully logged on and received the MPE colon (:) prompt, type:

```
:RUN FIGMAKER.PUB.SYS
```

and press RETURN.

The @ sign which is then displayed is Figure Maker's prompt.

You can create a figure by supplying an ASCII file of Figure Maker commands, or you can enter these commands interactively.

Interactive

When you enter Figure Maker commands interactively, your figure is created as you enter each command. This means that you cannot go back and change a previously entered command. If you want to modify the resulting figure, you have to re-enter all commands required to create the figure.

An example of interactive use is given at the end of this appendix.

ASCII File

The name of the file you supply can be an EDIT/3000 text file that contains Figure Maker commands. In this case, you can modify the figure by changing the text file and recreating the figure from this new file.

You can use Figure Maker to convert the output from your own graphics application to a format compatible with Hewlett-Packard graphics subsystems. This is done by converting your vector file into a figure in a figure file. Since different applications generate different types of output, a short intermediate program will be necessary in order to format the vectors properly for input to Figure Maker.

Examples of using an EDIT/3000 file and an outline of a conversion program are given at the end of this appendix.

Commands

The commands explained below are given in the order in which they would typically be used in a Figure Maker session.

OPENFILE *filename*

Opens the figure file named in the filename parameter. If the file does not exist, it is created now. You must use this command to open a figure file before you can begin creating a figure. Only one figure file can be open at a time. The file name must conform to the MPE file naming conventions.

The number of figures that can be stored in a figure file depends on the size of each figure. If the file you specify is full, you receive an error message and should give another figure file name.

A figure file is always created with a fixed amount of disc space. For the most effective use of space, it is best to put several figures in one figure file.

BEGINFIG *figurename*

Begins a new figure in the figure file specified with the OPENFILE command. The figure name parameter can be up to 16 characters long, must begin with a letter and can contain only letters, numbers and underscores (_).

USING FIGURE FILES

MOVE *x,y*

Moves to the specified coordinate point in the workspace. X and y are both real numbers and must have a positive value. The origin of the workspace is the lower left corner where x and y both have a value of 0. The value specified in the x parameter moves to the right; the y parameter moves up. The minimum value for x and y is 0.0; the maximum is 100.0. The MOVE command simply moves to the specified point; it does not draw a line as it moves.

DRAW *x,y*

Draws a straight line from the current position to the point specified. Like MOVE, x and y are real numbers and have a positive value. They should be specified relative to the origin of 0,0 and not relative to the current position. The minimum value for x and y is 0.0; the maximum is 100.0.

FONT *number*

Allows you to select different fonts for text. The font number is an integer between 1 and 4 and corresponds to the following: Font 1 - Modern Light, Font 2 - Modern Medium, Font 3 - Modern Bold, and Font 4 - Classic. The default is Font 1 - Modern Light.

SIZE *height*

A real number that defines the height of the text you want to draw. Height is specified in number of units. A height of 100.0 would fill the entire workspace. The height of the text remains in effect until a different height is specified in another SIZE command. Text height is measured from the top of uppercase to the bottom of the character descender, and includes some padding. The default text size is 2.92 units.

At the time of designing the figure, measurable size is not determined. You specify the actual size of the figure when it is used in a specific application. When creating the figure with Figure Maker, size is merely a proportional representation of 100 x 100 units.

TEXT *length "string"*

Draws the specified text at the current position. The length parameter gives the length of the text string in number of characters and must be an integer greater than or equal to 0.

The string is the text you want to draw, and must be enclosed within quotes.

TEXTDIR *text direction*

Positions the text at different angles by selecting the degrees at which you want the text positioned. If you select 45 for your angle, the text is positioned counter-clockwise at 45 degrees relative to the text position you chose. The angle is an integer between 0 and 359 inclusive. The default is 0.

PEN *pen number*

Selects the pen number you wish to use. Pen number applies to any multiple color device. The pen number parameter is an integer between 1 and 64 inclusive. The default pen number is 1.

ARC *xcenter, ycenter, degree length*

Draws an arc beginning at the current position. The center of the arc is specified by *xcenter*, *ycenter*. The length of the arc is specified by *degree length*. A positive degree length indicates a counterclockwise arc.

COMMENT

Allows you to include a comment line. You can have multiple line comments, but each line must be preceded by the **COMMENT** command.

INCLUDE *filename*

Executes the Figure Maker commands in the ASCII file named in the file name parameter. Included files can be nested up to ten levels.

An example of including an ASCII file is given at the end of this section.

ENDFIG

Ends the current figure. If you are creating more than one figure in a session, the **ENDFIG** command must be used before using **BEGINFIG** to start a new figure.

CLOSEFILE *filename*

Closes the figure file named in the file name parameter. The file name must conform to the MPE file naming conventions and is the name of the figure file that is currently open.

USING FIGURE FILES

EXIT

Ends the Figure Maker session and returns you to the MPE Operating System as indicated by the colon prompt.

END OF PROGRAM

:

Updating Figure Files

Files created using DSG/3000 version A.01.00, HPEasyChart version A.00.00, or HPDraw versions A.00.00 and A.00.01 will need to be converted for use with the present version of IFS. Files need to be converted only once. To use the figure update program:

1. Log into account as original creator.
2. When the colon prompt appears, type:

```
:RUN FUPDATE.PUB.SYS
```

3. A banner will appear on the screen, followed by:

```
Enter name of figure file to be converted (RETURN to exit):
```

4. Enter the old file name after the colon. The following instruction appears:

```
Enter a unique name for the old figure file:
```

5. When you enter the new file name, the screen displays:

Figure file successfully converted.

The updated figure file now exists under the original file name. This is so that any references to the original file name (in HPDraw) do not have to be changed.

You can continue to enter names of figure files to be updated if you wish. The first instruction will be repeated. Press **RETURN** to exit.

Note that the two file names cannot be the same. The old file is renamed (to the name specified in step 4). This enables you to archive the old file if you wish. Once you have successfully printed your updated figure file, you may want to purge the old files from the system.

For figure files created using a character set other than USASCII, run FUPDATE as follows:

```
:RUN FUPDATE.PUB.SYS[;parm=lang]
```

Where *lang* is as follows:

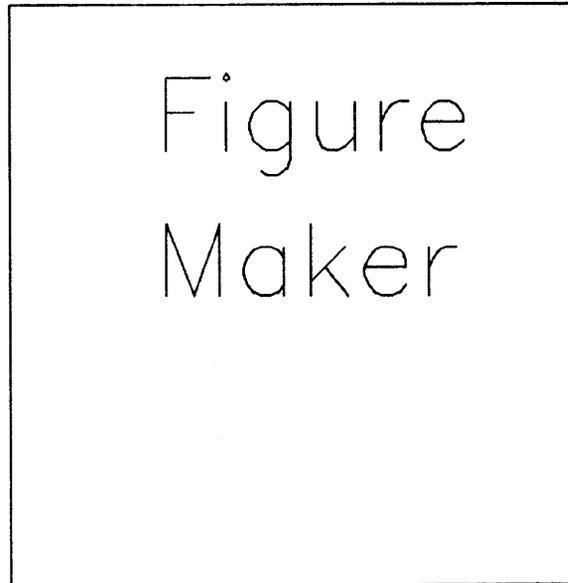
- 2 - Swedish/Finnish
- 3 - Norwegian/Danish
- 4 - French
- 5 - German
- 6 - United Kingdom
- 7 - Spanish

USING FIGURE FILES

FIGMAKER EXAMPLES

1. Interactive Commands

```
@OPENFILE FIGURES
@BEGINFIG FIG1
@MOVE 20,20
@DRAW 20,40
@DRAW 40,40
@DRAW 40,20
@DRAW 20,20
@MOVE 25.2,35
@FONT 1
@SIZE 5
@TEXTDIR 0
@TEXT 6 "Figure"
@MOVE 25.2,30.3
@TEXT 5 "Maker"
@ENDFIG
@CLOSEFILE FIGURES
@EXIT
```



2. ASCII FILE - this produces the same figure as that created in example 1.

Figure Maker Session:

```
@OPENFILE FIGURES
@BEGINFIG FIG1
@USE MAKEFIG
@ENDFIG
@CLOSEFILE FIGURES
@EXIT
```

EDIT/3000 Listing of MAKEFIG:

```

1 COMMENT Draw a Box
2 MOVE 20,20
3 DRAW 20,40
4 DRAW 40,40
5 DRAW 40,20
6 DRAW 20,20
7 COMMENT Draw the Text
8 MOVE 25.2,35
9 SIZE 5
10 TEXT 6 "Figure"
11 MOVE 25.2,30.3
12 TEXT 5 "Maker"

```

3. PROGRAM OUTLINE FOR VECTOR CONVERSION

In the following example, it is assumed that each vector is stored as a record consisting of four real numbers which determine its endpoints; that is (X1,Y1) (X2,Y2). Scaling of vector endpoints is necessary to conform to Figure Maker's 100 x 100 unit workspace.

```

OPEN Vector-List-File
OPEN Figure-Maker-File

```

```

WRITE (OPENFILE DIAGRAMS) to Figure-Maker-File
WRITE (BEGINFIG SAMPLE_DIAGRAM) to Figure-Maker-File

```

Determine scaling factors so endpoints lie in a 100x100 workspace.

```

WHILE not End-of-File on Vector-List-File
  BEGIN
    READ (X1,Y1,X2,Y2) from Vector-List-File
    SCALE X's and Y's
    WRITE (MOVE X1 Y1) to Figure-Maker-File in ASCII
    WRITE (DRAW X2 Y2) to Figure-Maker-File in ASCII
  END
WRITE (ENDFIG) to Figure-Maker-File

```

USING FIGURE FILES

WRITE (CLOSEFILE DIAGRAMS) to Figure-Maker-File
CLOSE Vector-List-File CLOSE Figure-Maker-File

ERROR MESSAGES

Error Number	Meaning
401	Height specified is invalid. Must be a real number greater than 0.
0707	A figure file has already been opened.
0708	The specified figure file has not been opened.
1004	Text length is less than 0.
1501	A figure by that name already exists in the figure file.
1508	A figure is currently being created. You must end the curr figure before doing a BEGINFIG.
1509	File open intrinsic failed.
1510	File close intrinsic failed.
1517	A BEGINFIG command must be done before an ENDFIG command
1518	Not a legal figure name.
1519	Figure file is full. The current figure cannot be put in the file.

THIS PAGE SHOULD BE BLANK

Environments Supplied with IFS/3000

APPENDIX C

The environments supplied with IFS/3000 simulate common printing devices such as line printers and typewriters. This appendix includes the specifications for these environments and gives possible logical page locations.

These environments are installed in the HPENV group in the SYS account.

Environment files with more than one logical page use the same font for VFC, actual spacing and forms unless a 2:1 or 4:1 reduction is specified in which case a form is defined on each logical page.

Figure C-1 illustrates the format of the HP 2680A environments listed in Table C-1 and Figure C-2 illustrates the format of the HP 2688A environments listed in Table C-2.

One major difference in these environments is the orientation of the logical pages.

In all environments, the character fonts used are rotated to the same orientation as the logical page so that printing begins in the upper left corner of the logical page, as viewed in the direction of its orientation.

ENVIRONMENTS SUPPLIED WITH IFS/3000

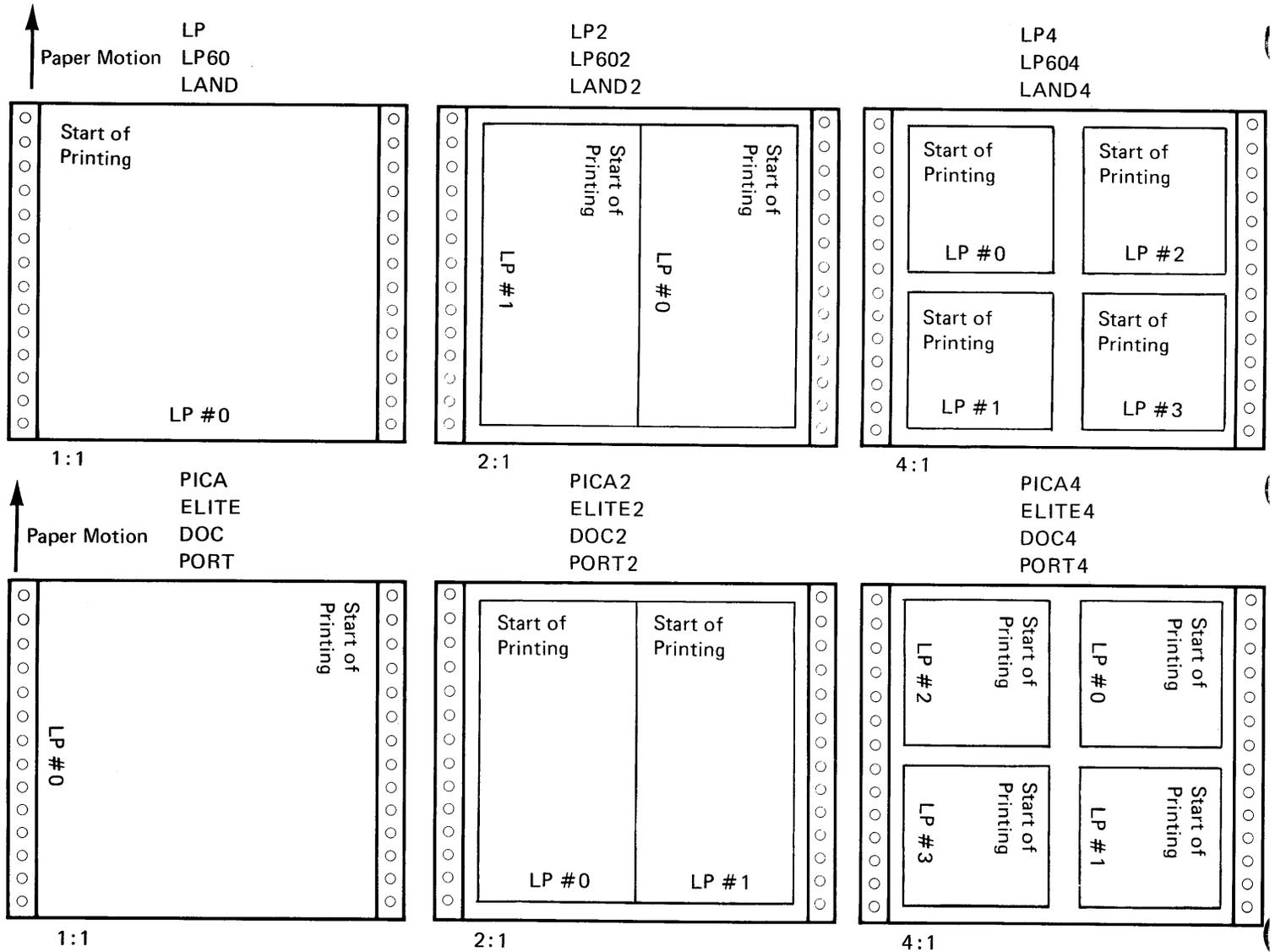


Figure C-1. HP 2680 Environment Formats

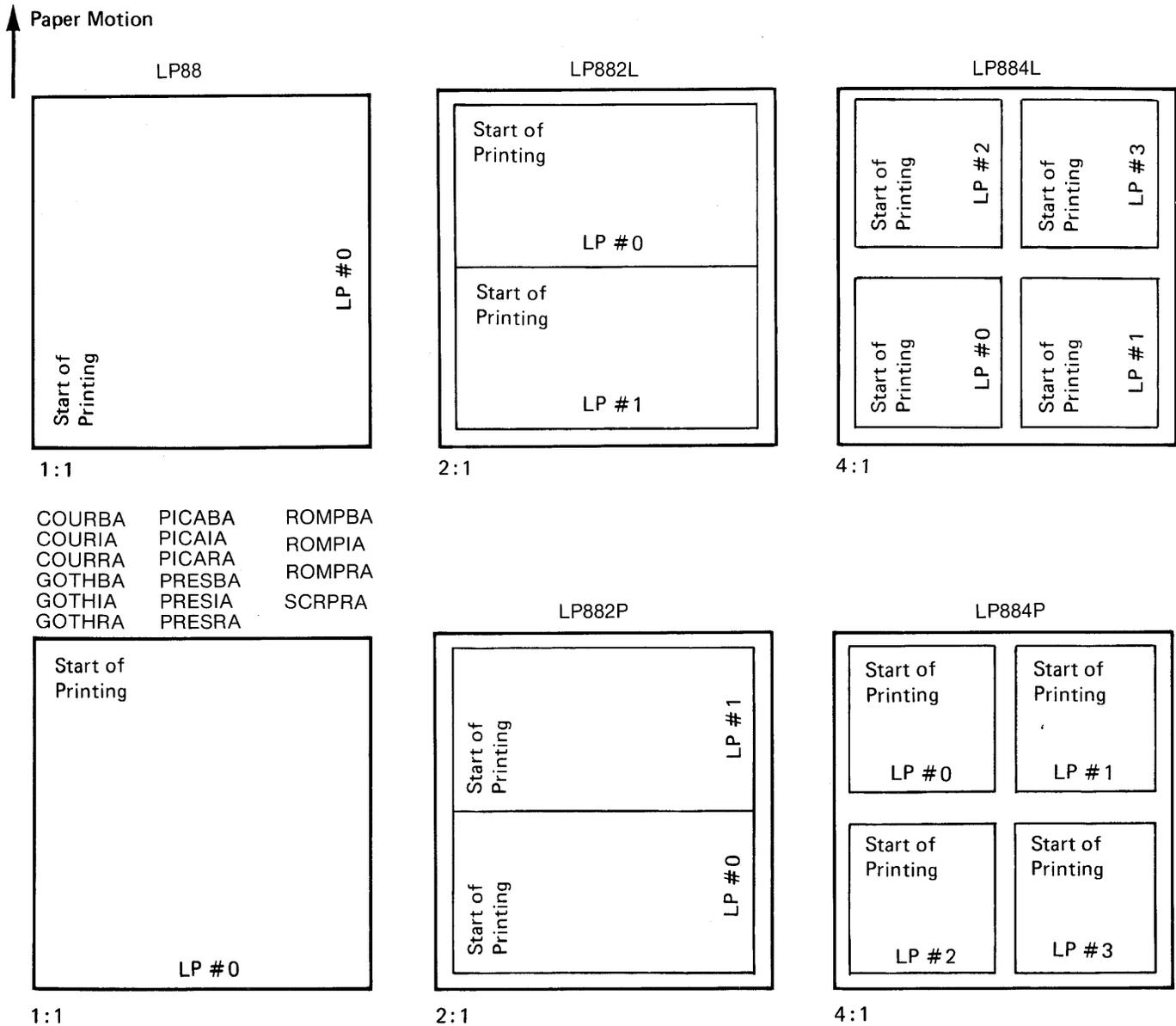


Figure C-2. HP 2688 Environment Formats

Table C-1. HP 2680A Environments Supplied with IFS/3000

ENVIRONMENT FILE NAME ¹	Description	Reduction	Logical Pages Per Physical Page	Orientation of Logical Pages	Characters Per Line	Lines Per Logical Page	Lines Skipped Before/After Print ²
LP	Line Printer		1	Landscape	132	66	3/3
LP2	Line Printer	2:1	2	Portrait	132	66	3/3
LP4	Line Printer	4:1	4	Landscape	132	66	3/3
LP60	Line Printer		1	Landscape	132	61	0/1
LP602	Line Printer	2:1	2	Portrait	132	61	0/1
LP604	Line Printer	4:1	4	Landscape	132	61	0/1
PICA	Pica Typewriter		1	Portrait	75	66	3/3
PICA2	Pica Typewriter	2:1	2	Landscape	75	66	3/3
PICA4	Pica Typewriter	4:1	4	Portrait	75	66	3/3
ELITE	Elite Typewriter		1	Portrait	90	66	3/3
ELITE2	Elite Typewriter	2:1	2	Landscape	90	66	3/3
ELITE4	Elite Typewriter	4:1	4	Portrait	90	66	3/3

 Base Character Font ³ Name/ w x h Number (Dots)	Actual Spacing Character Line Width ⁴ Height (Dots)	Character Font Files ⁵ P=Primary S=Secondary	Form Name	Sample Applications
#0 #0 #0	14 20 10 15 7 10	(P) LINEPR66 (P) LINEPR66 (P) LINEPR66	BORDERHPLP2 BORDERHPLP4	This environment simulates a line printer including Auto-Page Eject off and skip to channel 11. With Auto-Page Eject off, 66 lines per page are printed, but printing does not extend to the edge of the paper because of the 1/2 inch margin required at the top and bottom of each physical page. For more information on Auto-Page Eject, refer to FCONTROL in the MPE Intrinsic Reference Manual (30000-90010).
 #0 #0 #0	14 22 10 16 7 11	(P) LINEPR60 (P) LINEPR60 (P) LINEPR60	BORDERLP602 BORDERLP604	This environment simulates a line printer with Auto-Page Eject enabled except that a skip to channel 11 advances to top of form instead of one line above top of form. Use this environment for listings requiring only 60 lines, such as compiler listings.
Pica Pica Pica	18 30 12 20 9 15	(P) PICA (P) PICA (P) PICA	BORDERHPPICA2 BORDERHPPICA4	This environment simulates a typewriter with Pica type (10 characters per inch). You can produce the effect of varying typing elements by adding character fonts to this environment.
Elite Elite  Elite	15 30 10 20 7 15	(P) ELITE (S) LT ITAL (P) ELITE (S) LT ITAL (P) ELITE (S) LT ITAL	BORDERHPELITE2 BORDERHPELITE4	This is the same as the Pica environment except that Elite prints 12 characters per inch.

Table C-1. HP 2680A Environments Supplied with IFS/3000 (continued)

ENVIRONMENT FILE NAME ¹	Description	Reduction	Logical Pages Per Physical Page	Orientation of Logical Pages	Characters Per Line	Lines Per Logical Page	Lines Skipped Before/After Print ²
DOC	Document		1	Portrait	Proportionally Spaced	66	3/3
DOC2	Document	2:1	2	Landscape	Proportionally Spaced	66	3/3
DOC4	Document	4:1	4	Portrait	Proportionally Spaced	66	3/3
LAND	Landscape		1	Landscape			
LAND2	Landscape	2:1	2	Portrait			
LAND4	Landscape	4:1	4	Landscape			
PORT	Portrait		1	Portrait			
PORT2	Portrait	2:1	2	Landscape			
PORT4	Portrait	4:1	4	Portrait			

 Base Character Font³ Name/ w x h Number (Dots)	Actual Spacing Character Line Width⁴ Height (Dots)	Character Font Files⁵ P=Primary S=Secondary	Form Name	Sample Applications
Roman 24 x 25 Roman 18 x 19 Roman 7 x 11	24 30 18 20 7 13	(P) ROM (S) ROMBOLD ROMITAL (P) ROM (S) ROMBOLD ROMITAL (P) ROM (S) ROMBOLD ROMITAL		The DOC (document) environment prints 60 lines of proportionally spaced characters (Auto-Page Eject off gives 66 lines). The horizontal spacing is equivalent to 10-point characters and the vertical spacing is equivalent to 12-point characters. Use this environment to produce high-quality printed material.
			BORDERHPLP2 BORDERHPLP4	The Landscape environment has logical pages similar to the line printer environment except that there are no character fonts. You normally use Landscape as the basis for creating new environments.
			BORDERHPDOC2 BORDERHPDOC4	Portrait is used in the same way as Landscape. The difference between Landscape and Portrait is that with Landscape the logical page width exceeds the height, while with Portrait, the height exceeds the width. Portrait has logical pages similar to the DOC environment.

Table C-1. HP 2680A Environments Supplied with IFS/3000 (continued)

ENVIRONMENT FILE NAME ¹	Description	Reduction	Logical Pages Per Physical Page	Orientation of Logical Pages	Characters Per Line	Lines Per Logical Page	Lines Skipped Before/After Print ²
PICA1F	Pica Typewriter (French)		1	Portrait	75	66	3/3
PICA2F	Pica Typewriter (French)	2:1	2	Landscape	75	66	3/3
ELITE1F	Elite Typewriter (French)		1	Portrait	90	66	3/3
ELITE2F	Elite Typewriter (French)	2:1	2	Landscape	90	66	3/3
PICA1G	Pica Typewriter (German)		1	Portrait	75	66	3/3
PICA2G	Pica Typewriter (German)	2:1	2	Landscape	75	66	3/3
ELITE1G	Elite Typewriter (German)		1	Portrait	90	66	3/3
ELITE2G	Elite Typewriter (German)	2:1	2	Landscape	90	66	3/3



Base Character Font ³ Name/ w x h Number (Dots)	Actual Spacing Character Line Width ⁴ Height (Dots)	Character Font Files ⁵ P=Primary S=Secondary	Form Name	Sample Applications
Pica	18 30	(P) PICAF		This is the same as the Pica environment except that it contains ISO substitution characters for the French language.
Pica	12 20	(P) PICAF	BORDERHPPICA2	
Elite	15 30	(P) ELITEF		This is the same as the Elite environment except that it contains ISO substitution characters for the French language.
Elite	10 20	(P) ELITEF	BORDERHPELITE2	
Pica	18 30	(P) PICAG		This is the same as the Pica environment except that it contains ISO substitution characters for the German language.
Pica	12 20	(P) PICAG	BORDERHPPICA2	
Elite	15 30	(P) ELITEG		This is the same as the Elite environment except that it contains ISO substitution characters for the German language.
Elite	10 20	(P) ELITEG	BORDERHPELITE2	

Table C-1. HP 2680A Environments Supplied with IFS/3000 (continued)

ENVIRONMENT FILE NAME ¹	Description	Reduction	Logical Pages Per Physical Page	Orientation of Logical Pages	Characters Per Line	Lines Per Logical Page	Lines Skipped Before/After Print ²
PICA1N	Pica Typewriter (Norwegian/Danish)		1	Portrait	75	66	3/3
PICA2N	Pica Typewriter (Norwegian/Danish)	2:1	2	Landscape	75	66	3/3
ELITE1N	Elite Typewriter (Norwegian/Danish)		1	Portrait	90	66	3/3
ELITE2N	Elite Typewriter (Norwegian/Danish)	2:1	2	Landscape	90	66	3/3
PICA1S	Pica Typewriter (Spanish)		1	Portrait	75	66	3/3
PICA2S	Pica Typewriter (Spanish)	2:1	2	Landscape	75	66	3/3
ELITE1S	Elite Typewriter (Spanish)		1	Portrait	90	66	3/3
ELITE2S	Elite Typewriter (Spanish)	2:1	2	Landscape	90	66	3/3



Base Character Font ³ Name/ Number	w x h (Dots)	Actual Spacing Character Line Width ⁴ Height (Dots)	Character Font Files ⁵ P=Primary S=Secondary	Form Name	Sample Applications
Pica		18 30	(P) P ICAN		This is the same as the Pica environment except that it contains ISO substitution characters for the Norwegian and Danish languages.
Pica		12 20	(P) P ICAN	BORDERHPPICA2	
Elite		15 30	(P) ELITEN		This is the same as the Elite environment except that it contains ISO substitution characters for the Norwegian and Danish languages.
Elite		10 20	(P) ELITEN	BORDERHPELITE2	
Pica		18 30	(P) P ICAS		This is the same as the Pica environment except that it contains ISO substitution characters for the Spanish language.
Pica		12 20	(P) P ICAS	BORDERHPPICA2	
Elite		15 30	(P) ELITES		This is the same as the Elite environment except that it contains ISO substitution characters for the Spanish language.
Elite		10 20	(P) ELITES	BORDERHPELITE2	

Table C-1. HP 2680A Environments Supplied with IFS/3000 (continued)

ENVIRONMENT FILE NAME ¹	Description	Reduction	Logical Pages Per Physical Page	Orientation of Logical Pages	Characters Per Line	Lines Per Logical Page	Lines Skipped Before/After Print ²
PICA1W	Pica Typewriter (Swedish/Finnish)		1	Portrait	75	66	3/3
PICA2W	Pica Typewriter (Swedish/Finnish)	2:1	2	Landscape	75	66	3/3
ELITE1W	Elite Typewriter (Swedish/Finnish)		1	Portrait	90	66	3/3
ELITE2W	Elite Typewriter (Swedish/Finnish)	2:1	2	Landscape	90	66	3/3
PICA1U	Pica Typewriter (U.K. English)		1	Portrait	75	66	3/3
PICA2U	Pica Typewriter (U.K. English)	2:1	2	Landscape	75	66	3/3
ELITE1U	Elite Typewriter (U.K. English)		1	Portrait	90	66	3/3
ELITE2U	Elite Typewriter (U.K. English)	2:1	2	Landscape	90	66	3/3



Base Character Font ³ Name/ w x h Number (Dots)	Actual Spacing Character Line Width ⁴ Height (Dots)	Character Font Files ⁵ P=Primary S=Secondary	Form Name	Sample Applications
Pica	18 30	(P) PICA W		This is the same as the Pica environment except that it contains ISO substitution characters for the Swedish and Finnish languages.
Pica	12 20	(P) PICA W	BORDERHPPICA2	
Elite	15 30	(P) ELITE W		This is the same as the Elite environment except that it contains ISO substitution characters for the Swedish and Finnish languages.
Elite	10 20	(P) ELITE W	BORDERHPELITE2	
Pica	18 30	(P) PICA U		This is the same as the Pica environment except that it contains ISO substitution characters for the United Kingdom language.
Pica	12 20	(P) PICA U	BORDERHPPICA2	
Elite	15 30	(P) ELITE U		This is the same as the Elite environment except that it contains ISO substitution characters for the United Kingdom language.
Elite	10 20	(P) ELITE U	BORDERHPELITE2	

Table C-1. HP 2680A Environments Supplied with IFS/3000 (continued)

ENVIRONMENT FILE NAME ¹	Description	Reduction	Logical Pages Per Physical Page	Orientation of Logical Pages	Characters Per Line	Lines Per Logical Page	Lines Skipped Before/After Print ²
ELITE1X	Elite Typewriter with Roman Extension		1	Portrait	90	66	3/3
ELITE2X	Elite Typewriter with Roman Extension	2:1	2	Landscape	90	66	3/3
LTITAL1X	Elite Typewriter Italics with Roman Extension		1	Portrait	90	66	3/3
LTITAL2X	Elite Typewriter Italics with Roman Extension	2:1	2	Landscape	90	66	3/3

Base Character Font ³ Name/ w x h Number (Dots)	Actual Spacing Character Line Width ⁴ Height (Dots)	Character Font Files ⁵ P=Primary S=Secondary	Form Name	Sample Applications
Elite Elite	15 30 10 20	(P) ELITE (S) ELITEX (P) ELITE (S) ELITEX	BORDERHPELITE2	This has the same format as the Elite environment. The primary font is Elite, the secondary font is the Roman Extension to Elite.
Ltital Ltital	15 30 10 20	(P) LTITAL (S) LTITALX (P) LTITAL (S) LTITALX	BORDERHPELITE2	This has the same format as the Elite environment. The primary font is Elite Italics; the secondary font is the Roman Extension to Elite Italics.

Table C-1. HP 2680A Environments Supplied with IFS/3000 (continued)

ENVIRONMENT FILE NAME ¹	Description	Reduction	Logical Pages Per Physical Page	Orientation of Logical Pages	Characters Per Line	Lines Per Logical Page	Lines Skipped Before/After Print ²
PICA1X	Pica Typewriter with Roman Extension		1	Portrait	75	66	3/3
PICA2X	Pica Typewriter with Roman Extension	2:1	2	Landscape	75	66	3/3
DOC1X	Document with Roman Extension		1	Portrait	Proportionally Spaced	66	3/3
DOC2X	Document with Roman Extension	2:1	2	Landscape	Proportionally Spaced	66	3/3
SCRIPT1X	Script Typewriter with Roman Extension		1	Portrait	90	66	3/3



Base Character Font ³ Name/ w x h Number (Dots)	Actual Spacing Character Line Width ⁴ Height (Dots)	Character Font Files ⁵ P=Primary S=Secondary	Form Name	Sample Applications
Pica Pica	18 30 12 20	(P) PICA (S) PICAX (P) PICA (S) PICAX	BORDERHPPICA2	This has the same format as the Pica environment. The primary font is Pica; the secondary font is the Roman Extension to Pica.
Roman Roman	24 30 18 20	(P) ROM (S) ROMX (P) ROM (S) ROMX	BORDERHPDOC2	This has the same format as the DOC environment. The primary font is Roman; the secondary font is the Roman Extension to Roman.
Script	15 30	(P) SCRIPT (S) SCRIPTX		This has the same format as the Elite environment. The primary font is Script; the secondary font is the Roman Extension to Script.



Table C-1. HP 2680A Environments Supplied with IFS/3000 (continued)

ENVIRONMENT FILE NAME ¹	Description	Reduction	Logical Pages Per Physical Page	Orientation of Logical Pages	Characters Per Line	Lines Per Logical Page	Lines Skipped Before/After Print ²
LPK	Katakana Charsets		1	L	132	66	3/3
LPK2	Katakana Charsets	2:1	2	P	132	66	3/3
LPK4	Katakana Charsets	4:1	4	L	132	66	3/3
PICA4K	Katakana Charsets		1	P	75	66	3/3
PICA2K	Katakana Charsets	2:1	2	L	75	66	3/3
PICA4K	Katakana Charsets	4:1	4	P	132	66	3/3

Table C-2. HP 2688A Environments Supplied with IFS/3000

ENVIRONMENT FILE NAME ¹	Description	Reduction	Logical Pages Per Physical Page	Orientation of Logical Pages	Characters Per Line	Lines Per Logical Page	Lines Skipped Before/After Print ²
COURBA	Courier Typewriter Bold		1	P	75	66	3/3
COURIA	Courier Italics Typewriter Italics		1	P	75	66	3/3
COURRA	Courier Typewriter Regular		1	P	75	66	3/3
GOTHBA	Gothic Typewriter Bold		1	P	90	66	3/3
GOTHIA	Gothic Typewriter Italics		1	P	90	66	3/3
GOTHRA	Gothic Typewriter Regular		1	P	90	66	3/3
LP88	Line Printer		1	L	132	66	3/3
LP882L	Line Printer	2:1	2	L	132	66	3/3
LP882P	Line Printer	2:1	2	P	75	66	3/3
LP884L	Line Printer	4:1	4	L	132	66	3/3
LP884P	Line Printer	4:1	2	P	75	66	3/3
PICABA	Pica Typewriter Bold		1	P	75	66	3/3
PICAIA	Pica Typewriter Italics		1	P	75	66	3/3
PICARA	Pica Typewriter Regular		1	P	75	66	3/3

 Base Character Font³ Name/ w x h Number (Dots)	Actual Spacing Character Line Width⁴ Height (Dots)		Character Font Files⁵ P=Primary S=Secondary	Form Name	Sample Applications
#0 #0 #0	30 30 30	50 50 50	(P) COURB88S (S) COURB88X (P) COURI88S (S) COURI88X (P) COURR88S (S) COURR88X		
 #0 25 50 #0 25 50 #0 25 50	30 30 30	50 50 50	(P) GOTHB88S (S) GOTHB88X (P) GOTHI88S (S) GOTHI88X (P) GOTHR88S (S) GOTHR88X		
#0 #0 #0 #0 #0	20 14 18 10 14	35 22 30 16 20	(P) LINEP88s (S) LINEP88X	BORDERHPLP882L BORDERHPLP882P BORDERHPLP884L BORDERHPLP884L	
 #0 #0 #0	30 30 30	50 50 50	(P) PICAB88S (S) PICAB88X (P) PICAI88S (S) PICAI88X (P) PICAR88S (S) PICAR88X		

Table C-2. HP 2688A Environments Supplied with IFS/3000 (continued)

ENVIRONMENT FILE NAME ¹	Description	Reduction	Logical Pages Per Physical Page	Orientation of Logical Pages	Characters Per Line	Lines Per Logical Page	Lines Skipped Before/After Print ²
PRESBA	Prestige Elite Typewriter Bold		1	P	90	66	3/3
PRESIA	Prestige Elite Typewriter Italics		1	P	90	66	3/3
PRESRA	Prestige Elite Typewriter Regular		1	P	90	66	3/3
ROMPRA	Times Roman Prop. (Doc) Bold		1	P	(P.S.)	66	3/3
ROMPIA	Times Roman Prop. (Doc) Italics		1	P	(P.S.)	66	3/3
ROMPRA	Times Roman Prop. (Doc) Regular		1	P	(P.S.)	66	3/3
SCRPRA	Typewriter Script		1	P	90	66	3/3

	Base Character Font ³		Actual Spacing Character Line		Character Font Files ⁵ P=Primary S=Secondary	Form Name	Sample Applications
	Name/ Number	w x h (Dots)	Width ⁴ (Dots)	Height (Dots)			
	#0	25 50	30	50	(P) PRESB88S (S) PRESB88X		
	#0	25 50	30	50	(P) PRESI88S (S) PRESI88X		
	#0	25 50	30	50	(P) PRESR88S (S) PRESR88X		
	#0	40 50			(P) ROMPB88S (S) ROMPB88X		
	#0	40 50			(P) ROMPI88S (S) ROMPI88X		
	#0	40 50			(P) ROMPR88S (S) ROMPR88X		
	#0	25 50	30	50	(P) SCRPT88S (S) SCRPT88X		
							

Table C-2. HP 2688A Environments Supplied with IFS/3000 (continued)

ENVIRONMENT FILE NAME ¹	Description	Reduction	Logical Pages Per Physical Page	Orientation of Logical Pages	Characters Per Line	Lines Per Logical Page	Lines Skipped Before/After Print ²
COUR88A	HPWORD Charsets		3	P	82	66	3/3
COUR88E	HPWORD Charsets		3	P	80	70	3/3
GOTH88A	HPWORD Charsets		3	P	99	66	3/3
GOTH88E	HPWORD Charsets		3	P	96	70	3/3



**Base
Character Font³
Name/ w x h
Number (Dots)**

**Actual Spacing
Character Line
Width⁴ Height
(Dots)**

**Character Font
Files⁵
P=Primary
S=Secondary**

**Form
Name**

Sample Applications

30 50

COURR88S
COURR88X
COURB88S
COURB88X
COURI88S
COURI88X

30 50

COURR88S
COURR88X
COURB88S
COURB88X
COURI88S
COURI88X

25 50

GOTHR88S
GOTHR88X
GOTHB88S
GOTHB88X
GOTHI88S
GOTHI88X

25 50

GOTHR88S
GOTHR88X
GOTHB88S
GOTHB88X
GOTHI88S
GOTHI88X



Table C-2. HP 2688A Environments Supplied with IFS/3000 (continued)

ENVIRONMENT FILE NAME ¹	Description	Reduction	Logical Pages Per Physical Page	Orientation of Logical Pages	Characters Per Line	Lines Per Logical Page	Lines Skipped Before/After Print ²
PROP88A	HPWORD Charsets		3	P		66	3/3
PROP88E	HPWORD Charsets		3	P		66	3/3



**Base
Character Font³
Name/ w x h
Number (Dots)**

**Actual Spacing
Character Line
Width⁴ Height
(Dots)**

**Character Font
Files⁵
P=Primary
S=Secondary**

**Form
Name**

Sample Applications

40 50

ROMPR88S
ROMPR88X
ROMPB88S
ROMPB88X
ROMPI88S
ROMPI88X

40 50

ROMPR88S
ROMPR88X
ROMPB88S
ROMPB88X
ROMPI88S
ROMPI88X



1. These are HP-defined environment files located in the HPENV group of the SYS account.
2. If Auto-Page Eject is disabled, the lines skipped before/after print values are ignored resulting in 0 lines skipped before the first line of print and after the last one.
3. Font for VFC width and height are listed only when different from the actual spacing.
4. For proportionally spaced characters fonts, the maximum cell width is listed.
5. The character font files are in the CHARSETS group of the SYS account.
6. The forms are contained in the forms file named BORDERS in the HPENV group in the SYS account.

Character Fonts Supplied with IFS/3000

APPENDIX D

This appendix contains the file names and samples of all the character fonts supplied with the purchase of IFS/3000 for use on the HP 2680 and 2688 Laser Printers. The information is organized as follows:

HP 2680 CHARACTER FONTS	D-2
Proportionally Spaced Characters	D-2
Fixed Space Characters	D-6
ISO Substitution Characters	D-14
Roman Extension Characters	D-26
Special Characters	D-30
Bar Codes	D-35.1
 MORE ABOUT BAR CODES	 D-36
 HP 2688 CHARACTER FONTS	 D-40
Fixed, Proportional, and Roman Extension Characters	D-40
Special Characters	D-52
Bar Codes	D-54

D-1. HP 2680 Character Fonts Supplied With IFS/3000

CHARACTER FONT FILE NAME	Cell Size		Baseline	ADDITIONAL INFORMATION
	Point Size	Width (in dots)	Height (in dots)	
<p>Proportionally Spaced Characters:</p> <p>HELV.CHARSETS.SYS</p>	11.2	27	28	<p>The digits 0-9, the plus sign (+), minus sign (-) & dollar sign (\$) have fixed spacing.</p>
	10	25	25	
	8	20	20	
	6	15	15	
<p>HELVITAL.CHARSETS.SYS</p>	11.2	27	28	<p>The digits 0-9, the plus sign (+), minus sign (-) & dollar sign (\$) have fixed spacing.</p>
	8	20	20	

CHARACTERS

!"#\$%&'()*+,-./0123456789;:<=>?@
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`
abcdefghijklmnopqrstuvwxyz{|}~

Here is some text printed in
the selected font. The line
spacing is set by the font

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

*Here is some text printed in
the selected font. The line
spacing is set by the font*

*Here is some text printed in
the selected font. The line
spacing is set by the font also.*

D-1. HP 2680 Character Fonts Supplied With IFS/3000

CHARACTER FONT FILE NAME	Cell Size		Baseline	ADDITIONAL INFORMATION
	Point	Width	Height	
	Size	(in dots)	(in dots)	
HELVBOLD.CHARSETS.SYS	24	60	60	For 8, 11.2, & 14 point, the digits 0-9, the plus sign (+), minus sign (-) & dollar sign (\$) have fixed spacing. For 24 point, all characters are proportionally spaced.
	14	35	35	
	11.2	27	28	
	8	20	20	

CHARACTERS

**!"#\$%&'()*+,-./01234
ABCDEFGHIJKLMN
OP
abcdefghijklmnopqrst**

**!"#\$%&'()*+,-./0123456789;:<=>?
ABCDEFGHIJKLMN
OPQRSTUVWXYZ
abcdefghijklmnopqrstuv
xyz{ }~**

**!"#\$%&'()*+,-./0123456789;:<=>?@
ABCDEFGHIJKLMN
OPQRSTUVWXYZ[\]^_`
abcdefghijklmnopqrst
uvwxyz{ }~**

**!"#\$%&'()*+,-./0123456789;:<=>?@
ABCDEFGHIJKLMN
OPQRSTUVWXYZ[\]^_`
abcdefghijklmnopqrst
uvwxyz{ }~**

Here is some text

**Here is some text printed in
the selected font. The line
spacing is set by the font also.**

**Here is some text printed in
the selected font. The line
spacing is set by the font**

**Here is some text printed in
the selected font. The line
spacing is set by the font also.**

D-1. HP 2680 Character Fonts Supplied With IFS/3000

CHARACTER FONT FILE NAME	Cell Size		Baseline	ADDITIONAL INFORMATION	
	Point	Width	Height		
	Size	(in dots)	(in dots)		
ROM.CHARSETS.SYS	11.2	27	28	7	The digits 0-9, the plus sign (+), minus sign (-) & dollar sign (\$) have fixed spacing.
	10	24	25	7	
	8	18	19	6	
	5	12	12	4	
ROMITAL.CHARSETS.SYS	11.2	27	28	7	
	10	24	25	7	
	8	18	19	6	
	5	12	12	4	

CHARACTERS

!"#\$%&'()*+,-./0123456789;<=>?@
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`
abcdefghijklmnopqrstuvwxyz{|}~

Here is some text printed in
the selected font. The line
spacing is set by the font

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

*Here is some text printed in
the selected font. The line
spacing is set by the font*

*Here is some text printed in
the selected font. The line
spacing is set by the font also.*

*Here is some text printed in
the selected font. The line
spacing is set by the font also.*

*Here is some text printed in
the selected font. The line
spacing is set by the font also.*

D-1. HP 2680 Character Fonts Supplied With IFS/3000

CHARACTER FONT FILE NAME	Cell Size		Baseline	ADDITIONAL INFORMATION
	Point Size	Width (in dots)	Height (in dots)	
ROMBOLD.CHARSETS.SYS	11.2	27	28	
	10	25	25	
	8	18	19	
	5	12	12	

CHARACTERS

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`
abcdefghijklmnopqrstuvwxyz{|}~

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`
abcdefghijklmnopqrstuvwxyz{|}~

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`
abcdefghijklmnopqrstuvwxyz{|}~

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`
abcdefghijklmnopqrstuvwxyz{|}~

Here is some text printed in
the selected font. The line
spacing is set by the font

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

D-1. HP 2680 Character Fonts Supplied With IFS/3000 (cont.)

CHARACTER FONT FILE NAME	Cell Size			Baseline	ADDITIONAL INFORMATION
	Point Size	Width (in dots)	Height (in dots)	(in dots)	
Fixed Space Characters:					
BIGONES.CHARSETS.SYS	99	250	250	64	This character font file contains only the digits 0-9 & the section symbol §. This symbol is printed by typing a tilde(~).
COUR.CHARSETS.SYS	12	18	31	12	
	8	12	20	5	
	6	9	15	5	
COUR12.CHARSETS.SYS	12	15	30	12	
	8	10	20	6	
	6	7	15	5	

CHARACTERS

248

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`
abcdefghijklmnopqrstuvwxyz{|}~^-

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

D-1. HP 2680 Character Fonts Supplied With IFS/3000 (cont.)

CHARACTER FONT FILE NAME	Cell Size			Baseline (in dots)	ADDITIONAL INFORMATION
	Point Size	Width (in dots)	Height (in dots)		
Fixed Space Characters: (Continued)					
COURITAL.CHARSETS.SYS	12	17	30	12	
	8	11	20	7	
	6	9	15	6	
ELITE.CHARSETS.SYS	12	15	30	10	
	8	10	20	6	
	6	7	15	5	
LRGELITE.CHARSETS.SYS	12	15	30	10	
	8	10	20	6	

CHARACTERS

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMN^oPQRSTU^vVWXYZ[\]^_`
abcdefghijklmnopqrstu^vwxyz{|}~^-

*Here is some text printed in
the selected font. The line
spacing is set by the font also.*

*Here is some text printed in
the selected font. The line
spacing is set by the font also.*

*Here is some text printed in
the selected font. The line
spacing is set by the font also.*

*Here is some text printed in
the selected font. The line
spacing is set by the font also.*

*Here is some text printed in
the selected font. The line
spacing is set by the font also.*

*Here is some text printed in
the selected font. The line
spacing is set by the font also.*

*Here is some text printed in
the selected font. The line
spacing is set by the font also.*

*Here is some text printed in
the selected font. The line
spacing is set by the font also.*

D-1. HP 2680 Character Fonts Supplied With IFS/3000 (cont.)

CHARACTER FONT FILE NAME	Cell Size			Baseline	ADDITIONAL INFORMATION
	Point Size	Width (in dots)	Height (in dots)	(in dots)	
Fixed Space Characters: (Continued)					
LRGELITE.CHARSETS.SYS	6	7	15	5	
LTITAL.CHARSETS.SYS	12	17	30	10	
	8	11	20	7	
	6	9	15	6	
PICA.CHARSETS.SYS	12	18	30	8	
	8	12	20	5	
	6	9	15	5	

CHARACTERS

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNopqrstuvwxyz[\]^_`
abcdefghijklmnopqrstuvwxyz{|}~^-

Here is some text printed in
the selected font. The line
spacing is set by the font also.

*Here is some text printed in
the selected font. The line
spacing is set by the font also.*

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

D-1. HP 2680 Character Fonts Supplied With IFS/3000 (cont.)

CHARACTER FONT FILE NAME	Cell Size			Baseline	ADDITIONAL INFORMATION
	Point Size	Width (in dots)	Height (in dots)	(in dots)	
Fixed Space Characters: (Continued)					
LINEPR60.CHARSETS.SYS	9	14	22	5	
	6.5	10	16	5	
	4	7	11	3	
LINEPR61.CHARSETS.SYS	8	14	21	7	
	6	10	15	5	
	4	7	10	3	
LINEPR66.CHARSETS.SYS	8	14	20	6	
	6	10	15	5	
	4	7	10	3	

CHARACTERS

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNPOQRSTUVWXYZ[\]^_`
abcdefghijklmnopqrstuvwxy{|}~

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

D-1. HP 2680 Character Fonts Supplied With IFS/3000 (cont.)

CHARACTER FONT FILE NAME	Cell Size			Baseline	ADDITIONAL INFORMATION
	Point Size	Width (in dots)	Height (in dots)	(in dots)	
Fixed Space Characters: (Continued)					
SCRIPT.CHARSETS.SYS	12	15	30	8	
ISO Substitution Characters:					
ELITEF.CHARSETS.SYS (FRENCH)	12	15	30	7	This is the same as the Elite character font except that it contains ISO substitution characters for the French Language
	8	10	20	4	
	6	7	18	5	
ELITEG.CHARSETS.SYS (GERMAN)	12	15	30	7	This is the same as the Elite character font except that it contains ISO substitution characters for the German Language.
	8	10	20	4	
	6	7	18	5	

CHARACTERS

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMN^oOPQRSTUVWXYZ[\]^_`
abcdefghijklmnopqrstu^ovwxyz{|}~"

!"#\$%&'()*+,-./0123456789:;<=>?à
ABCDEFGHIJKLMN^oOPQRSTUVWXYZ^oç\$^`
abcdefghijklmnopqrstu^ovwxyzéùè"

!"#\$%&'()*+,-./0123456789:;<=>?à
ABCDEFGHIJKLMN^oOPQRSTUVWXYZ^oç\$^`
abcdefghijklmnopqrstu^ovwxyzéùè"

!"#\$%&'()*+,-./0123456789:;<=>?à
ABCDEFGHIJKLMN^oOPQRSTUVWXYZ^oç\$^`
abcdefghijklmnopqrstu^ovwxyzéùè"

!"£\$%&'()*+,-./0123456789:;<=>?§
ABCDEFGHIJKLMN^oOPQRSTUVWXYZ^oÄÖÜ^`
abcdefghijklmnopqrstu^ovwxyzäöüß

!"£\$%&'()*+,-./0123456789:;<=>?§
ABCDEFGHIJKLMN^oOPQRSTUVWXYZ^oÄÖÜ^`
abcdefghijklmnopqrstu^ovwxyzäöüß

!"£\$%&'()*+,-./0123456789:;<=>?§
ABCDEFGHIJKLMN^oOPQRSTUVWXYZ^oÄÖÜ^`
abcdefghijklmnopqrstu^ovwxyzäöüß

*Here is some text printed in
the selected font. The line
spacing is set by the font also.*

*Here is some text printed in
the selected font. The line
spacing is set by the font also.*

*Here is some text printed in
the selected font. The line
spacing is set by the font also.*

*Here is some text printed in
the selected font. The line
spacing is set by the font also.*

*Here is some text printed in
the selected font. The line
spacing is set by the font also.*

*Here is some text printed in
the selected font. The line
spacing is set by the font also.*

*Here is some text printed in
the selected font. The line
spacing is set by the font also.*

D-1. HP 2680 Character Fonts Supplied With IFS/3000 (cont.)

CHARACTER FONT FILE NAME	Cell Size			Baseline (in dots)	ADDITIONAL INFORMATION
	Point Size	Width (in dots)	Height (in dots)		
ISO Substitution Characters: (Continued)					
ELITEN.CHARSETS.SYS (NORWEGIAN/ DANISH)	12	15	30	7	This is the same as the Elite character font except that it contains ISO substitution characters for the Norwegian and Danish Languages.
	8	10	20	4	
	6	7	18	5	
ELITES.CHARSETS.SYS (SPANISH)	12	15	30	7	This is the same as the Elite character font except that it contains ISO substitution characters for the Spanish Language.
	8	10	20	4	
	6	7	18	5	

CHARACTERS

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMN~~OP~~QRSTU~~VW~~XYZ~~æ~~øå^`
abcdefghijklmnopqrstu~~vw~~xyzæøå~`

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMN~~OP~~QRSTU~~VW~~XYZ~~æ~~øå^`
abcdefghijklmnopqrstu~~vw~~xyzæøå~`

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMN~~OP~~QRSTU~~VW~~XYZ~~æ~~øå^`
abcdefghijklmnopqrstu~~vw~~xyzæøå~`

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMN~~OP~~QRSTU~~VW~~XYZ;Ñ;°`
abcdefghijklmnopqrstu~~vw~~xyz{ñ}~`

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMN~~OP~~QRSTU~~VW~~XYZ;Ñ;°`
abcdefghijklmnopqrstu~~vw~~xyz{ñ}~`

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMN~~OP~~QRSTU~~VW~~XYZ;Ñ;°`
abcdefghijklmnopqrstu~~vw~~xyz{ñ}~`

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

D-1. HP 2680 Character Fonts Supplied With IFS/3000 (cont.)

CHARACTER FONT FILE NAME	Cell Size			Baseline	ADDITIONAL INFORMATION
	Point Size	Width (in dots)	Height (in dots)	(in dots)	
ISO Substitution Characters: (Continued) ELITEU.CHARSETS.SYS (U.K. ENGLISH) ELITEW.CHARSETS.SYS (SWEDISH/FINNISH)	12	15	30	7	This is the same as the Elite character font except that it contains ISO substitution characters for the United Kingdom. This is the same as the Elite character font except that it contains ISO substitution characters for the Swedish and Finnish Languages.
	8	10	20	4	
	6	7	18	5	
	12	15	30	7	
	8	10	20	4	
	6	7	18	5	

CHARACTERS

!"£\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNPOQRSTUVWXYZ[\]^_`
abcdefghijklmnopqrstuvwxy{|}~

!"£\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNPOQRSTUVWXYZ[\]^_`
abcdefghijklmnopqrstuvwxy{|}~

!"£\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNPOQRSTUVWXYZ[\]^_`
abcdefghijklmnopqrstuvwxy{|}~

!"#%&'()*+,-./0123456789:;<=>?É
ABCDEFGHIJKLMNPOQRSTUVWXYZÄÖÅÛ_é
abcdefghijklmnopqrstuvwxyäöåû

!"#%&'()*+,-./0123456789:;<=>?É
ABCDEFGHIJKLMNPOQRSTUVWXYZÄÖÅÛ_é
abcdefghijklmnopqrstuvwxyäöåû

!"#%&'()*+,-./0123456789:;<=>?É
ABCDEFGHIJKLMNPOQRSTUVWXYZÄÖÅÛ_é
abcdefghijklmnopqrstuvwxyäöåû

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

D-1. HP 2680 Character Fonts Supplied With IFS/3000 (cont.)

CHARACTER FONT FILE NAME	Cell Size			Baseline (in dots)	ADDITIONAL INFORMATION
	Point Size	Width (in dots)	Height (in dots)		
PICAF.CHARSETS.SYS (FRENCH)	12	18	30	7	This is the same as the Pica character font except that it contains ISO substitution characters for the French Language.
	8	12	20	4	
	6	9	18	4	
PICAG.CHARSETS.SYS (GERMAN)	12	18	30	7	
	8	12	20	4	
	6	9	18	4	

CHARACTERS

!"#\$%&'()*+,-./0123456789:;<=>?`à
ABCDEFGHIJKLMN O PQRSTU VWXYZ°ç\$^`
abcdefghijklmnopqrstuvwxyzéùè"'

!"#\$%&'()*+,-./0123456789:;<=>?`à
ABCDEFGHIJKLMN O PQRSTU VWXYZ°ç\$^`
abcdefghijklmnopqrstuvwxyzéùè"'

!"#\$%&'()*+,-./0123456789:;<=>?`à
ABCDEFGHIJKLMN O PQRSTU VWXYZ°ç\$^`
abcdefghijklmnopqrstuvwxyzéùè"'

!"&\$%&'()*+,-./0123456789:;<=>?§
ABCDEFGHIJKLMN O PQRSTU VWXYZÄÖÜ^`
abcdefghijklmnopqrstuvwxyzäöüß

!"&\$%&'()*+,-./0123456789:;<=>?§
ABCDEFGHIJKLMN O PQRSTU VWXYZÄÖÜ^`
abcdefghijklmnopqrstuvwxyzäöüß

!"&\$%&'()*+,-./0123456789:;<=>?§
ABCDEFGHIJKLMN O PQRSTU VWXYZÄÖÜ^`
abcdefghijklmnopqrstuvwxyzäöüß

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

D-1. HP 2680 Character Fonts Supplied With IFS/3000 (cont.)

CHARACTER FONT FILE NAME	Cell Size			Baseline	ADDITIONAL INFORMATION
	Point Size	Width (in dots)	Height (in dots)	(in dots)	
PICAN.CHARSETS.SYS (NORWEGIAN/ DANISH)	12	18	30	7	This is the same as the Pica character font except that it contains ISO substitution characters for the Norwegian and Danish Languages.
	8	12	20	4	
	6	9	18	4	
PICAS.CHARSETS.SYS (SPANISH)	12	18	30	7	
	8	12	20	4	
	6	9	18	4	

CHARACTERS

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMN O PQRSTU VWXYZÆØÅ^`~
abcdefghijklmnopqrstu vwxyzæøå~`

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMN O PQRSTU VWXYZÆØÅ^`~
abcdefghijklmnopqrstu vwxyzæøå~`

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMN O PQRSTU VWXYZÆØÅ^`~
abcdefghijklmnopqrstu vwxyzæøå~`

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMN O PQRSTU VWXYZ;Ñ¿°`~
abcdefghijklmnopqrstu vwxyz{ñ}~`

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMN O PQRSTU VWXYZ;Ñ¿°`~
abcdefghijklmnopqrstu vwxyz{ñ}~`

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMN O PQRSTU VWXYZ;Ñ¿°`~
abcdefghijklmnopqrstu vwxyz{ñ}~`

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

CHARACTERS

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`
abcdefghijklmnopqrstuvwxyz{|}~^-

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`
abcdefghijklmnopqrstuvwxyz{|}~^-

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`
abcdefghijklmnopqrstuvwxyz{|}~^-

!"#0%&'()*+,-./0123456789:;<=>?É
ABCDEFGHIJKLMNOPQRSTUVWXYZÄÖÅÛ_é
abcdefghijklmnopqrstuvwxyzäöåü

!"#0%&'()*+,-./0123456789:;<=>?É
ABCDEFGHIJKLMNOPQRSTUVWXYZÄÖÅÛ_é
abcdefghijklmnopqrstuvwxyzäöåü

!"#0%&'()*+,-./0123456789:;<=>?É
ABCDEFGHIJKLMNOPQRSTUVWXYZÄÖÅÛ_é
abcdefghijklmnopqrstuvwxyzäöåü

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

Here is some text printed in
the selected font. The line
spacing is set by the font also.

D-1. HP 2680 Character Fonts Supplied With IFS/3000 (cont.)

CHARACTER FONT FILE NAME	Cell Size			Baseline	ADDITIONAL INFORMATION
	Point Size	Width (in dots)	Height (in dots)	(in dots)	
Roman Extension Characters:					
ELITEX.CHARSETS.SYS	12	15	30	7	This character font is the Roman Extension to Elite. In the character samples, a black rectangle denotes undefined characters.
	8	10	23	6	
	6	7	18	5	
LTITALX.CHARSETS.SYS	12	17	30	8	This character font is the Roman Extension to Elite Italics. In the character samples, a black rectangle denotes undefined characters.
	8	11	23	7	
	6	9	19	6	

CHARACTERS

éòúáéóúáèòúáèòúÁíØÆáíøæÁíÖÛËÏß

éòúáéóúáèòúáèòúÁíØÆáíøæÁíÖÛËÏß

éòúáéóúáèòúáèòúÁíØÆáíøæÁíÖÛËÏß

éòúáéóúáèòúáèòúÁíØÆáíøæÁíÖÛËÏß

éòúáéóúáèòúáèòúÁíØÆáíøæÁíÖÛËÏß

éòúáéóúáèòúáèòúÁíØÆáíøæÁíÖÛËÏß

D-1. HP 2680 Character Fonts Supplied With IFS/3000 (cont.)

CHARACTER FONT FILE NAME	Cell Size			Baseline	ADDITIONAL INFORMATION
	Point Size	Width (in dots)	Height (in dots)	(in dots)	
SCRIPTX.CHARSETS.SYS	12	15	30	7	This character font is the Roman Extension to Script. In the character samples, a black rectangle denotes undefined characters.
PICAX.CHARSETS.SYS	12	18	30	7	This character font is the Roman Extension to Pica. In the character samples, a black rectangle denotes undefined characters.
	8	12	23	5	
	6	9	19	5	
ROMX.CHARSETS.SYS	10	25	30	7	This character font is the Roman Extension to Roman. In the character samples, a black rectangle denotes undefined characters.
	8	20	25	6	

CHARACTERS

ε-° çÑñ; ;Dε § â
êôúáéóúàèòüäëöüÀíØÆáíøæÀìÖÛËß

D-1. HP 2680 Character Fonts Supplied With IFS/3000 (cont.)

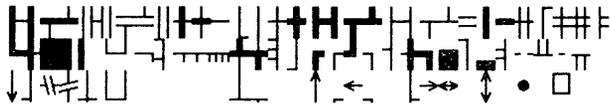
CHARACTER FONT FILE NAME	Cell Size			Baseline	ADDITIONAL INFORMATION
	Point Size	Width (in dots)	Height (in dots)	(in dots)	
Special Characters:					
MATH.CHARSETS.SYS	12	18	30	8	This font contains Mathematical symbols.
OCRA.CHARSETS.SYS	13.5	18	34	7	This font is used with Optical card readers.
OCRB.CHARSETS.SYS	11	20	27	8	This font is used with Optical card readers.
PICALINE.CHARSETS.SYS	12	18	30	8	This is a Line drawing character font.

CHARACTERS

√|\$∇±α∫÷≅ΠΓΥ≡ΦΞ°¹²³⁴⁵⁶⁷⁸⁹ΩΛ∞}†Σ∏
 αβψφεθληιθκωμνρπγθστξδδχυζ↑→↵←↓∩
 αβψφεθληιθκωμνρπγθστξδδχυζ↑→↵←

!"#\$%&'()*+,-./0123456789:;<=>?@
 ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`
 abcdefghijklmnopqrstuvwxyz{|}~

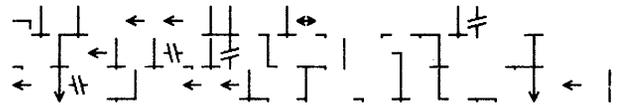
!"#\$%&'()*+,-./0123456789:;<=>?@
 ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`
 abcdefghijklmnopqrstuvwxyz{|}~



πεθε ισ σρμε τεχτ πθιντεφ ιν
 τηε σεωεψτεφ θρντφ τηε ωινε
 σπαψινλ ισ σετ βυ τηε θρντ αωσρφ

Here is some text printed in
 the selected font. The line
 spacing is set by the font also.

Here is some text printed in
 the selected font. The line
 spacing is set by the font also.



D-1. HP 2680 Character Fonts Supplied With IFS/3000 (cont.)

CHARACTER FONT FILE NAME	Cell Size			Baseline	ADDITIONAL INFORMATION
	Point Size	Width (in dots)	Height (in dots)	(in dots)	
Special Characters:					
PICAK.CHARSETS.SYS	12	18	30	8	Katakana substitution with Katakana 8 extension
	8	12	20	5	
	6	9	15	5	
PICAKX.CHARSETS.SYS	12	18	30	8	Katakana substitution with Katakana 8 extension
	8	12	20	5	
	6	9	15	5	
OCRBK.CHARSETS.SYS	11	20	27	8	Katakana substitution with Katakana 8 extension
OCRBKX.CHARSETS.SYS	11	20	27	8	

CHARACTERS

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNopqRstUvwXyZ[¥]^-_~

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNopqRstUvwXyZ[¥]^-_~

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNopqRstUvwXyZ[¥]^-_~

「」・フアイウエオヤユヨツーアイウエオカキクケコサシスセソタ
チツテトナニヌネノハヒフヘホマミムメモヤユヨラリルロロクン〃タ

「」・フアイウエオヤユヨツーアイウエオカキクケコサシスセソタ
チツテトナニヌネノハヒフヘホマミムメモヤユヨラリルロロクン〃タ

「」・フアイウエオヤユヨツーアイウエオカキクケコサシスセソタ
チツテトナニヌネノハヒフヘホマミムメモヤユヨラリルロロクン〃タ

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNopqRstUvwXyZ[¥]^_~

「」・フアイウエオヤユヨツーアイウエオカキクケコサシスセソタ
チツテトナニヌネノハヒフヘホマミムメモヤユヨラリルロロクン〃タ

Here is some text printed in
the selected font. The line
spacing is set by the font

Here is some text printed in
the selected font. The line
spacing is set by the font as well.

Here is some text printed in
the selected font. The line
spacing is set by the font as well.

Here is some text printed in
the selected font. The line
spacing is set by the font

D-1. HP 2680 Character Fonts Supplied With IFS/3000 (cont.)

CHARACTER FONT FILE NAME	Cell Size			Baseline	ADDITIONAL INFORMATION
	Point Size	Width (in dots)	Height (in dots)	(in dots)	
Special Characters: LNPR66K.CHARSETS.SYS	8	14	20	6	Katakana substitution with Katakana 8 extension
	6	10	15	5	
	4	7	10	3	
LNPR66KX.CHARSETS.SYS	8	14	20	6	Katakana substitution with Katakana 8 extension
	6	10	15	5	
	4	7	10	3	

CHARACTERS

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNOpqrsTUVWxyz{|}~_

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNOpqrsTUVWxyz{|}~_

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNOpqrsTUVWxyz{|}~_

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNOpqrsTUVWxyz{|}~_

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNOpqrsTUVWxyz{|}~_

!"#\$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNOpqrsTUVWxyz{|}~_

Here is some text printed in
spacing is set by the font as well.

Here is some text printed in
the selected font. The line
spacing is set by the font as well.

Here is some text printed in
the selected font. The line
spacing is set by the font as well.

D-1. HP 2680 Character Fonts Supplied With IFS/3000

CHARACTER FONT FILE NAME	Cell Size		Baseline	ADDITIONAL INFORMATION
	Point	Width	Height	
	Size	(in dots)	(in dots)	
Bar Codes:				CODE 39™ ALPHANUMERIC BAR CODE*
BAR39W14.CHARSETS.SYS	17.5	64	75	CODE39™, 14 mil minimum bar width
BAR39W19.CHARSETS.SYS	18	28	45	CODE39™, 19 mil minimum bar width
BAR39W25.CHARSETS.SYS	18	35	45	CODE39™, 25 mil minimum bar width
				*Code 39 is a trademark of Interface Mechanism, Inc.
UPCA20.CHARSETS.SYS	18	49	45	UPC-A, 20 mil minimum bar width
UPCA25.CHARSETS.SYS	18	28	45	UPC-A, 25 mil minimum bar width
UPCA35.CHARSETS.SYS	18	35	45	UPC-A, 35 mil minimum bar width

CHARACTERS



D-1. HP 2680 Character Fonts Supplied With IFS/3000

CHARACTER FONT FILE NAME	Cell Size			Baseline (in dots)	ADDITIONAL INFORMATION
	Point	Width	Height		
	Size	(in dots)	(in dots)		
UPCE20.CHARSETS.SYS	18	49	45		UPC-E, 20 mil minimum bar width
UPCE25.CHARSETS.SYS	18	45	45		UPC-E, 25 mil minimum bar width
UPCE35.CHARSETS.SYS	18	61	45		UPC-E, 35 mil minimum bar width

CHARACTERS



More About Bar Codes

Several Bar Code character fonts have been provided for use on the HP Laser printing systems. These include different sizes of the Uniform Product Code (UPC) versions A and E for the HP 2680, and Bar Code 39 for the HP 2680 and HP 2688. These fonts were designed using the appropriate Code 39 and UPC specifications, and have been read with different bar code readers to verify readability. There is, however, no guarantee they will work with your particular readers.*

Three widths have been provided for each of the HP 2680 bar code types as shown in Table D-2 (one width for Bar Code 39 on the HP 2688). The nominal width for the narrow bar/space would be 12, 14, 19, or 25 thousandths of an inch depending on the character set selected (Table D-2).

To print a bar code string:

1. Select the appropriate character font.
2. Enter the desired string including guard bars.

To print the character 0 with check digit (another 0) and the guard bars, the character string would be:

00

and would print as:



If a bar height greater than the character height is required, the string may be printed multiple times, each pass doubling the normal height. For example:

00

00

would print as:



***HEWLETT-PACKARD MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THE BAR CODE CHARACTER FONTS, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE OR ABILITY TO BE READ BY A PARTICULAR BAR CODE READER. Hewlett-Packard shall not be liable for errors contained therein or for incidental or consequential damages in connection with the furnishing, performance, or use of these fonts.**

Bar Code 39®

Code 39® is a registered trademark of Interface Mechanisms, Inc.

For more details on the formats of Bar Code 39 refer to Hewlett-Packard Application Brief No. 406, "Use of the IPS 7000 to Print Bar Codes"

See Table D-4 for the locations in the character fonts for the various Bar Code 39 characters.

UPC VERSIONS A AND E

For more information on the formats of the UPC Code, refer to:

UPC SYMBOL SPECIFICATION
 Uniform Product Code Council, Inc.
 7051 Corporate Way, Suite 201
 Dayton, Ohio 45450

See Table D-3 for the locations in the character fonts for the various UPC version A and E characters.

Table D-2. Bar Code Files

File Name	Bar Code Type	Nominal Width of Narrow Bar/Space (thousandths of inch)
HP 2680A		
UPCA20	UPC Version A	20
UPCA 25	UPC Version A	25
UPCA35	UPC Version A	35
UPCI20	UPC Version E	20
UPCE25	UPC Version E	25
UPCE35	UPC Version E	35
BAR39W14	CODE 39®	14
BAR39W19	CODE 39®	19
BAR39W25	CODE 39®	25
HP 2688		
BAR12W88	CODE 39®	12

D-3. HP 2680 UPC Character Locations

UPC Character	Character Font Character	
	Version A	Version E
Left.Guard	@	@
Right Guard	@	=
Center Guard	?	not applicable
<u>Odd Characters</u>		
0	0	0
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	8
9	9	9
<u>Even Characters</u>		
0	A	A
1	B	B
2	C	C
3	D	D
4	E	E
5	F	F
6	G	G
7	H	H
8	I	I
9	J	J

D-4. HP 2680 and HP 2688 CODE 39™
Character Locations

Code 39 Character	Character Font Character
Guard	*
0	0
through	through
9	9
A	A
through	through
Z	Z
-	-
.	.
Space	,
\$	\$
/	/
+	+
%	%

THIS PAGE SHOULD BE BLANK

D-5. HP 2688 Character Fonts Supplied With IFS/3000

CHARACTER FONT FILE NAME	Cell Size			Baseline	ADDITIONAL INFORMATION
	Point Size	Width (in dots)	Height (in dots)	(in dots)	
COURB88S.CHARSETS.SYS	12	30	50	14	
COURB88X.CHARSETS.SYS	12	30	50	14	Roman 8 Extension
COURI88S.CHARSETS.SYS	12	30	50	14	
COURI88X.CHARSETS.SYS	12	30	50	14	Roman 8 Extension
COURR88S.CHARSETS.SYS	12	30	50	14	
COURR88X.CHARSETS.SYS	12	30	50	14	Roman 8 Extension

D-5. HP 2688 Character Fonts Supplied With IFS/3000 (cont.)

CHARACTER FONT FILE NAME	Cell Size			Baseline	ADDITIONAL INFORMATION
	Point Size	Width (in dots)	Height (in dots)	(in dots)	
GOTHB88S.CHARSETS.SYS	12	25	50	14	
GOTHB88X.CHARSETS.SYS	12	25	50	14	Roman 8 Extension
GOTHI88S.CHARSETS.SYS	12	25	50	14	
GOTHI88X.CHARSETS.SYS	12	25	50	14	Roman 8 Extension
GOTHR88S.CHARSETS.SYS	12	25	50	14	
GOTHR88X.CHARSETS.SYS	12	25	50	14	Roman 8 Extension

D-5. HP 2688 Character Fonts Supplied With IFS/3000 (cont.)

CHARACTER FONT FILE NAME	Cell Size			Baseline	ADDITIONAL INFORMATION
	Point Size	Width (in dots)	Height (in dots)	(in dots)	
PICAB88S.CHARSETS.SYS	12	30	50	14	
PICAB88X.CHARSETS.SYS	12	30	50	14	Roman 8 Extension
PICAI88S.CHARSETS.SYS	12	30	50	14	
PICAI88X.CHARSETS.SYS	12	30	50	14	Roman 8 Extension
PICAR88S.CHARSETS.SYS	12	30	50	14	
PICAR88X.CHARSETS.SYS	12	30	50	14	Roman 8 Extension

D-5. HP 2688 Character Fonts Supplied With IFS/3000 (cont.)

CHARACTER FONT FILE NAME	Cell Size			Baseline	ADDITIONAL INFORMATION
	Point Size	Width (in dots)	Height (in dots)	(in dots)	
PRESB88S.CHARSETS.SYS	12	25	50	14	
PRESB88X.CHARSETS.SYS	12	25	50	14	Roman 8 Extension
PRESI88S.CHARSETS.SYS	12	25	50	14	
PRESI88X.CHARSETS.SYS	12	25	50	14	Roman 8 Extension
PRESR88S.CHARSETS.SYS	12	25	50	14	
PRESR88X.CHARSETS.SYS	12	25	50	14	Roman 8 Extension

D-5. HP 2688 Character Fonts Supplied With IFS/3000 (cont.)

CHARACTER FONT FILE NAME	Cell Size		Baseline	ADDITIONAL INFORMATION	
	Point Size	Width (in dots)	Height (in dots)		
ROMPB88S.CHARSETS.SYS	12	40	50	14	
ROMPB88X.CHARSETS.SYS	12	40	50	14	Roman 8 Extension
ROMPI88S.CHARSETS.SYS	12	40	50	14	
ROMPI88X.CHARSETS.SYS	12	40	50	14	Roman 8 Extension
ROMPR88S.CHARSETS.SYS	12	40	50	14	
ROMPR88X.CHARSETS.SYS	12	40	50	14	Roman 8 Extension

CHARACTERS

!"#\$%&'()*+,-./0123456789;:<=>?@
 ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`
 abcdefghijklmnopqrstuvwxyz{|}~

À Á Ê Ë Ì Í Î Ï Ñ Ò Ó Ô Õ Ö × Ø Ù Ú Û Ü Ý Þ ß à á â ã ä å æ ç è é ê ë ì í î ï ð ñ ò ó ô õ ö ÷ ø ù ú û ü ý þ ß — † ‡ « » ±

!"#\$%&'()*+,-./0123456789;:<=>?@
 ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`
 abcdefghijklmnopqrstuvwxyz{|}~

À Á Ê Ë Ì Í Î Ï Ñ Ò Ó Ô Õ Ö × Ø Ù Ú Û Ü Ý Þ ß à á â ã ä å æ ç è é ê ë ì í î ï ð ñ ò ó ô õ ö ÷ ø ù ú û ü ý þ ß — † ‡ « » ±

!"#\$%&'()*+,-./0123456789;:<=>?@
 ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`
 abcdefghijklmnopqrstuvwxyz{|}~

À Á Ê Ë Ì Í Î Ï Ñ Ò Ó Ô Õ Ö × Ø Ù Ú Û Ü Ý Þ ß à á â ã ä å æ ç è é ê ë ì í î ï ð ñ ò ó ô õ ö ÷ ø ù ú û ü ý þ ß — † ‡ « » ±

Here is some text printed in
 the selected font. The line
 spacing is set by the font as well.

á ã é ê ë ì í î ï ð ñ ò ó ô õ ö ÷ ø ù ú û ü ý þ ß à á â ã ä å æ ç è é ê ë ì í î ï ð ñ ò ó ô õ ö ÷ ø ù ú û ü ý þ ß — † ‡ « » ±

Here is some text printed in
 the selected font. The line
 spacing is set by the font as well.

á ã é ê ë ì í î ï ð ñ ò ó ô õ ö ÷ ø ù ú û ü ý þ ß à á â ã ä å æ ç è é ê ë ì í î ï ð ñ ò ó ô õ ö ÷ ø ù ú û ü ý þ ß — † ‡ « » ±

Here is some text printed in
 the selected font. The line
 spacing is set by the font as well.

á ã é ê ë ì í î ï ð ñ ò ó ô õ ö ÷ ø ù ú û ü ý þ ß à á â ã ä å æ ç è é ê ë ì í î ï ð ñ ò ó ô õ ö ÷ ø ù ú û ü ý þ ß — † ‡ « » ±

D-5. HP 2688 Character Fonts Supplied With IFS/3000 (cont.)

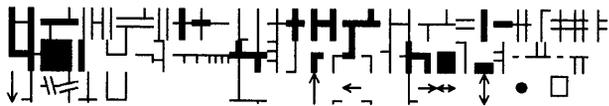
CHARACTER FONT FILE NAME	Cell Size		Baseline	ADDITIONAL INFORMATION	
	Point Size	Width (in dots)	Height (in dots)		
LINPR88S.CHARSETS.SYS	8.4	20	35	10	
LINPR88X.CHARSETS.SYS	8.4	20	35	10	Roman 8 Extension
SCRPT88S.CHARSETS.SYS	12	25	50	14	
SCRPT88X.CHARSETS.SYS	12	25	50	14	Roman 8 Extension

D-5. HP 2688 Character Fonts Supplied With IFS/3000

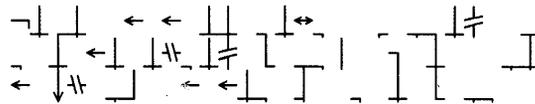
CHARACTER FONT FILE NAME	Cell Size		Baseline	ADDITIONAL INFORMATION	
	Point Size	Width (in dots)	Height (in dots)		
Special Characters:					
BIGNR88S.CHARSETS.SYS	60	252	250	1	This character font contains only the digits 0-9.
DRAWR88S.CHARSETS.SYS	12	30	50	12	This is a line drawing character font.
MATHR88S.CHARSETS.SYS	12	30	50	14	This font contains mathematical symbols.

CHARACTERS

2 4 8



√ | § ∇ ± α ρ + ≈ π Γ Ψ ≡ Φ Ξ ° 1 2 3 4 5 6 7 8 9 Ω Λ ∞) † Σ ¶
α β ψ φ ε θ λ η ι θ κ ω μ ν ρ π γ θ σ τ ε δ δ χ υ ζ ↑ → † ← ↓ ¶
α β ψ φ ε θ λ η ι θ κ ω μ ν ρ π γ θ σ τ ε δ δ χ υ ζ ↑ → † ←



π ε θ ε ι σ σ ρ μ ε τ ε χ τ π θ ι ν τ ε φ ι ν
τ η ε σ ε ω ε ψ τ ε φ θ ρ ν τ φ τ η ε ω ι ν ε
σ π α ψ ι ν λ ι σ σ ε τ β υ τ η ε θ ρ ν τ

D-5. HP 2688 Character Fonts Supplied With IFS/3000

CHARACTER FONT FILE NAME	Cell Size Point Width Height Size (in dots) (in dots)	Baseline (in dots)	ADDITIONAL INFORMATION
<p>Bar Codes:</p> <p>B39W1288.CHARSETS.SYS</p>	<p>18 77 45</p>	<p>CODE 39™ ALPHANUMERIC BAR CODE*</p> <p>CODE39™, 12 mil minimum bar width</p> <p>*Code 39 is a trademark of Interface Mechanism, Inc.</p>	

CHARACTERS



THIS PAGE SHOULD BE BLANK

Vertical Format Control (VFC)

VFC is the printer industry standard for Vertical Format Control. A VFC allows a programmer to instruct the printer to skip to predetermined lines on a page with typically one or two commands instead of counting and printing a number of blank lines. The definition of the predetermined lines is contained in the VFC.

The HP laser printer uses a VFC similar to that used for the HP 2608 printer except for the differences discussed in this appendix. A VFC for the HP laser printer contains 16 channels. Each channel performs a standard skip (or slew) function such as skip to top of next page, skip one line, skip to next quarter page, and so forth. The standard functions for each of the channels in a standard VFC are shown in table E-1.

For the VFC in table E-1, the line position for each skip is based on a page with 60 potential print lines. For documents with other than 60 lines per page, the number of lines skipped varies with the length of the page. For example, the mid-point of a 60-line page is a different line number than the mid-point of an 80-line page. Compare the line positions in table E-2 which shows the standard 8 line per inch VFC with 80 potential print lines per page to those in table E-1 which shows 6 lines per inch and 60 print lines per page.

VFC SPECIFICATIONS

CHANNEL	FUNCTION	LINE POSITIONS OF LOGICAL ONE*
1	Slew to top of next form	0
2	Slew to bottom of form	59
3	Single space	0, 1, 2, . . . 59
4	Slew to next double space line	0, 2, 4, . . . 58
5	Slew to triple space line	0, 3, 6, . . . 57
6	Slew to half page line	0, 30
7	Slew to next quarter page line	0, 15, 30, 45
8	Slew to next tenth line	0, 10, 20, . . . 50
9	Slew to bottom of form	59
10	Slew to one line previous to bottom of form	58
11	Slew to one line previous to top of next form	65
12	Slew to top of next form	0
13	Slew to next seventh line	0, 7, 14, . . . 56
14	Slew to next sixth line	0, 6, 12, . . . 54
15	Slew to next fifth line	0, 5, 10, . . . 55
16	Slew to next fourth line	0, 4, 8, . . . 56

* Logical one is similar to the hole in a paper tape or destination point for that channel.

NOTE: A ten-inch printed form area on an 11-inch page with 60 lines possible is assumed.

Table E-2. Standard Eight Line Per Inch Format

CHANNEL	FUNCTION	LINE POSITIONS OF LOGICAL ONE*
1	Slew to top of next form	0
2	Slew to bottom of form	79
3	Single space	0, 1, 2, . . . 79
4	Slew to next double space line	0, 2, 4, . . . 78
5	Slew to triple space line	0, 3, 6, . . . 77
6	Slew to half page line	0, 40
7	Slew to next quarter page line	0, 20, 40, 60
8	Slew to next tenth line	0, 10, 20, . . . 70
9	Slew to bottom of form	79
10	Slew to one line previous to bottom of form	78
11	Slew to one line previous to top of next form	87
12	Slew to top of next form	0
13	Slew to next seventh line	0, 7, 14, . . . 77
14	Slew to next sixth line	0, 6, 12, . . . 78
15	Slew to next fifth line	0, 5, 10, . . . 75
16	Slew to next fourth line	0, 4, 8, . . . 76

* Logical one is similar to the hole in a paper tape or destination point for that channel.

NOTE: A ten-inch printed form area on an 11-inch page with 80 lines possible is assumed.

Standard VFC

With a standard VFC, you use IFS/3000's Logical Page Menu to specify the number of print lines on the page. You can use the number of lines to specify the logical page height, or you can specify the page height and actual line height from which IFS/3000 calculates the number of print lines per page. IFS/3000 computes the number of lines to be skipped for each VFC channel and stores them in a VFC table. When a program issues a skip command using a VFC channel, the HP laser printer scans the VFC table to determine the line it should skip to.

User Defined VFC

The predefined skips in standard VFC channels may not fit your application. To accommodate unique applications, IFS/3000 allows you to define your own VFCs. With a user-defined VFC, each channel and its function is defined by you.

To specify a user-defined VFC, you must first create and store your VFC file unnumbered, using an editor. The name of this file is then entered on the IFS/3000 Vertical Format Control Menu.

VFC File Format

A user-defined VFC file is an ASCII file containing 80-byte records. For each record, only the first 16 columns are significant. IFS/3000 displays an error message when it encounters an unrecognized line in the VFC file. The HP laser printer ignores unrecognized VFC records. Refer to table E-3 for a sample VFC file format.

Table E-3. Sample VFC File Format

Record 1	MARGIN = nn	<p>Key:</p> <p>nn = A number between 1 and 16 inclusive. Specifies position of left margin indentation for the HP 2608. Ignored by the HP Laser Printer.</p> <p>x = 6,8,blank, or 0 specifies vertical spacing in lines per inch for the HP 2608, but not for the HP Laser Printer. Default is 6 lines per inch. If this is the only number, it is interpreted by the HP Laser Printer as number of lines per logical page.</p> <p>y = A number between 0 and 127 inclusive. Specifies number of lines in VFC.</p> <p>zzz = Comments to describe the VFC file. Ignored by the HP Laser Printer.</p> <p>A blank or 0 indicates a 0 (no-punch) and a non-blank indicates a 1 (a punch).</p>
Record 2	VFC,x,y,zzz	
Record 3	1011111100011111	
Record 4	001	
Record 5	0011	
Record 6	00101	
.	.	
Record m	01	m = Number of lines in the VFC. Each line corresponds to a potential print line on the page.

VFC SPECIFICATIONS

If you need to change from the default post-space mode to pre-space mode, use a carriage control value of "A". Use carriage control "@" to change back to post-space mode.

In general, a VFC file begins with an optional record which sets the left margin. While this record is meaningful to the HP 2608, the HP laser printer ignores the MARGIN record, if present. To set the left margin for the HP laser printer, you use the IFS/3000 Logical Page Menu.

The VFC record, which is required, follows the MARGIN record. The VFC record gives the number of lines per inch and the number of potential print lines on the page.

In the VFC record, the only parameter which is significant to the HP laser printer is the number of potential print lines per page. The other parameters have significance only for the HP 2608. Vertical line spacing for the HP laser printer is set on the IFS/3000 Logical Page Menu.

Table E-4 illustrates the number of lines per page used by the HP laser printer as a result of various VFC record formats. If the number of lines parameter is 0, the number used for lines per page is 11 times the value of the lines per inch parameter. If there is only one number following "VFC", the HP laser printer interprets it as the number of lines per page, not the lines per inch. If neither lines per inch nor lines per page is specified, the HP laser printer defaults to 66 lines per page. In any case, the number of lines before the first print line and after the last print line are set as close as possible to the unprintable area of the HP laser printer.

Table E-4. Sample VFC Record Formats

	Number of Lines Per Page
VFC 8,0	88
VFC 6,0	66
VFC 11,0	121
VFC 66	66
VFC	66

VFC SPECIFICATIONS

As shown in table E-1, the subsequent records are a paper tape image defining the lines to be skipped to for each channel. Column 1 represents the line to be skipped to when a program issues a skip to channel 1; column 2 represents channel 2, and so forth, through channel 16. To define the function for each channel, locate the column for the desired channel and put a non-blank or non-zero character (usually a 1) in the row representing the line to be skipped to.

These records can be used to define the number of lines to be skipped before the first print line. If zero records are skipped before the first print line, a skip to channel 11 (one line before the first print line) does not function correctly.

For a standard line printer, there are usually three lines skipped before the first print line and three lines after the last print line, with 60 potential print lines per page. To define this VFC, you specify 66 lines on a logical page with the Logical Page Menu. Using the VFC Selection Menu, you select the standard VFC option and specify three lines before and three lines after printing.

VFC SPECIFICATIONS

Carriage Control

The first byte of each output record can be used as the carriage control for that line of print.

```
:FILEPRINTOUT;DEV=LP;CCTL
```

With the "CCTL" specified in the file statement, the first byte of each output record will be used to determine carriage control.

The laser printer assumes that channel 3 is for single spacing and all carriage returns will generate a channel 3 call. A double space carriage control ("0") will generate a channel 4 call which will

normally slew paper to the next odd line (0,1,3,5...). A triple space carriage control ("-") will generate a channel 5 call which will normally slew paper to the next triple space line (0,3,6...). A page eject ("1") will select VFC channel 1.

If you need to space a set number of lines, use %2nn carriage control where nn is the number of lines to space.

Some sample carriage control characters are listed below.

OCTAL CODE	ASCII SYMBOL	CARRIAGE ACTION	LASER PRINTER CHANNEL CALLED
%40	" "	Single line feed	3
%53	"+"	Suppress line feed	
%55	"-"	Triple line feed	5
%60	"0"	Double line feed	4
%61	"1"	Logical page eject	1

Sample Programs

APPENDIX F

THIS PAGE SHOULD BE BLANK

COBOL

```
00001000 $CONTROL USLINIT
00002000 *
00003000 *       Quarterly Report Program.
00004000 *
00005000 * This sample program illustrates the calling sequences for
00006000 * several of the IFS/3000 intrinsics from COBOL. The program
00007000 * opens a SpoolFile indicated by the file equation:
00008000 *
00009000 *       :FILE SPOOLOUT;DEV=PP;ENV=MEMOENV
00010000 *
00011000 * The environment MEMOENV has the following characteristics:
00012000 *
00013000 * - 2 logical pages:
00014000 *   #0 at 90 degrees, no form, initially inactive.
00015000 *   #1 at 90 degrees, letterhead form, initially active.
00016000 *
00017000 * - 2 character sets:
00018000 *   #0 Elite.
00019000 *   #1 Elite Italics.
00020000 *
00021000 * - Standard 3/3 VFC.
00022000
00023000
00024000
00025000 Identification Division.
00026000 Program-Id. QREPORT.
00027000 Author. J. S.
00028000 Date-Written. July 1982.
00029000 Remarks. This program was written to illustrate
00030000 several of the IFS/3000 intrinsics.
00031000
00032000 Environment Division.
00033000 Configuration Section.
00034000 Source-Computer. HP-3000.
00035000 Object-Computer. HP-3000.
00036000
00037000 Input-Output Section.
00038000 File-Control.
00039000
00040000 Select PrintFile assign to "SPOOLOUT"
00041000 access is sequential.
00042000
00043000 Data Division.
00044000 File Section.
00045000
00046000 FD PrintFile
00047000 label records are omitted.
00048000 01 PrintFileRec pic X(132).
00049000
00050000 Working-Storage Section.
00051000
00052000 77 Language pic S9(4) comp.
00053000 77 XPosition pic X(16).
```

00054000	77	YPosition	pic X(16).
00055000	77	PrimaryFont	pic S9(4) comp.
00056000	77	SecondaryFont	pic S9(4) comp.
00057000	77	AreaLen	pic S9(4) comp.
00058000	77	FieldName	pic X(17).
00059000	77	SubFieldNum	pic S9(4) comp.
00060000	77	Info	pic X(255).
00061000	77	InfoLen	pic S9(4) comp.
00062000	77	OutputDev	pic X(6).
00063000	77	XLoc	pic S9(4) comp.
00064000	77	YLoc	pic S9(4) comp.
00065000	77	LogPageNum	pic S9(4) comp.
00066000	77	FigureName	pic X(17).
00067000	77	FigureFileName	pic X(36).
00068000	77	RasterFileName	pic X(36).
00069000	77	Height	pic X(16).
00070000	77	Rotation	pic S9(4) comp.
00071000	77	Units	pic S9(4) comp.
00072000	77	RasterNumber	pic S9(4) comp.
00073000	77	PositionMode	pic S9(4) comp.
00074000	77	ErrorMsg	pic X(80).
00075000	77	ErrorMsgLen	pic S9(4) comp.
00076000	77	MsgLen	pic S9(4) comp.
00076100	77	DisplayErrorNum	pic X(5).
00077000			
00078000			
00079000	01	ComArea.	
00080000	03	IFSStatus	pic S9(4) comp.
00081000	03	ErrorNum	pic S9(4) comp.
00082000	03	ErrorParm1	pic S9(4) comp.
00083000	03	ErrorParm2	pic S9(4) comp.
00084000	03	MsgFileNum	pic S9(4) comp.
00085000	03	Filler	pic S9(4) comp.
00086000			occurs 1995 times.
00087000			
00088000			
00089000			
00090000		Procedure Division.	
00091000			
00092000		Main-Process Section.	
00093000			
00094000		Mission-Control.	
00095000			
00096000		Open Output, PrintFile.	
00097000		Perform Generate-Report.	
00098000		Close PrintFile.	
00099000		Stop Run.	
00100000			
00101000			
00102000		Generate-Report Section.	
00103000			
00104000		Quarterly-Report.	
00105000			

Jan 84

F-4

```

00106000      *
00107000      * Initialize the IFS/3000 intrinsics.
00108000      *
00109000          Move 2000 to AreaLen.
00110000          Move Zero to Language.
00111000          Move "2680A " to OutputDev.
00112000          Call "PINITDEVICE" using ComArea,
00113000          AreaLen,
00114000          Language,
00115000          PrintFile,
00116000          OutputDev.
00117000          Perform Check-For-Error.
00118000
00119000      *
00120000      * Now write to the four fields.
00121000      *
00122000      * All fields have only 1 subfield.
00123000
00124000          Move -1 to SubFieldNum.
00125000
00126000
00127000
00128000          Move "FROM " to FieldName.
00129000          Move "      Profit Monitor" to Info.
00130000          Move 20 to InfoLen.
00131000          Call "PWRITEFIELD" using ComArea,
00132000          FieldName,
00133000          SubFieldNum,
00134000          Info,
00135000          InfoLen.
00136000          Perform Check-For-Error.
00137000
00138000          Move "TO " to FieldName.
00139000          Move "      All Regional Managers" to Info.
00140000          Move 27 to InfoLen.
00141000          Call "PWRITEFIELD" using ComArea,
00142000          FieldName,
00143000          SubFieldNum,
00144000          Info,
00145000          InfoLen.
00146000          Perform Check-For-Error.
00147000
00148000          Move "DATE " to FieldName.
00149000          Move "      August 1, 1982" to Info.
00150000          Move 24 to InfoLen.
00151000          Call "PWRITEFIELD" using ComArea,
00152000          FieldName,
00153000          SubFieldNum,
00154000          Info,
00155000          InfoLen.
00156000          Perform Check-For-Error.
00157000
00158000          Move "SUBJECT " to FieldName.

```

```

00159000          Move "          Last Quarter's Sales" to Info.
00160000          Move 34 to InfoLen.
00161000          Call "PWRITEFIELD" using ComArea,
00162000                                fieldName,
00163000                                SubFieldNum,
00164000                                Info,
00165000                                InfoLen.
00166000          Perform Check-For-Error.
00167000
00168000          *
00169000          * Now move the pen: 1 inch over, 4 inches down.
00170000          *
00171000          Move 180 to XLoc.
00172000          Move 720 to YLoc.
00173000          Call "PMOVEPENABS" using Comarea,
00174000                                XLoc,
00175000                                YLoc.
00176000          Perform Check-For-Error.
00177000
00178000          *
00179000          * Switch the primary font to italics.
00180000          *
00181000          Move 1 to PrimaryFont.
00182000
00183000          * Secondary font remains unchanged.
00184000
00185000          Move -1 to SecondaryFont.
00186000
00187000          Call "PUSEFONT" using ComArea,
00188000                                PrimaryFont,
00189000                                SecondaryFont.
00190000          Perform Check-For-Error.
00191000
00192000          *
00193000          * Now write out some text in the new font.
00194000          *
00195000          Move Spaces to Info.
00196000          Move "See page 2 for last quarter's sales figures."
00197000          to Info.
00198000          Write PrintFileRec From Info; Invalid Key Stop Run.
00199000
00200000          *
00201000          * Now, activate logical page 0 (no form) and deactivate
00202000          * logical page 1 (letterhead form) and then skip to the
00203000          * new page.
00204000          *
00205000          Move Zero to LogPageNum.
00206000          Call "PACTIVATEPAGE" using ComArea,
00207000                                LogPageNum.
00208000          Perform Check-For-Error.
00209000
00210000          Move 1 to LogPageNum.
00211000          Call "PDEACTIVATEPAGE" using ComArea,
00212000                                LogPageNum.

```

Jan 84

F-6

```

00213000         Perform Check-For-Error.
00214000
00215000         Call "PNEWPAGE" using ComArea.
00216000         Perform Check-For-Error.
00217000
00218000 *
00219000 * Now convert the figure created by DSG/3000 into a
00220000 * raster image.
00221000 *
00222000         Move "QTRSALES " to FigureFileName.
00223000         Move "SECOND QUARTER " to FigureName.
00224000         Move "R2NDQTR " to RasterFileName.
00225000
00226000 * Indicate inches.
00227000         Move 1 to Units.
00228000         Move "6.0 " to Height.
00229000         Move 90 to Rotation.
00230000         Call "PCONVERTFIGUREA" using ComArea,
00231000         FigureFileName,
00232000         FigureName,
00233000         RasterFileName,
00234000         OutputDev,
00235000         Height,
00236000         Units,
00237000         Rotation.
00238000         Perform Check-For-Error.
00239000
00240000 *
00241000 * Now load the image as #17, print it at a position relative
00242000 * to the Upper-left corner of the logical page, then delete it.
00243000 *
00244000         Move 17 to RasterNumber.
00245000         Call "PLOADRASTER" using ComArea,
00246000         RasterFileName,
00247000         RasterNumber.
00248000         Perform Check-For-Error.
00249000
00250000 * 1.5 Inches right; 2.0 inches down.
00251000         Move "1.5 " to XPosition.
00252000         Move "2.0 " to YPosition.
00253000 * Indicate relative positioning.
00254000         Move 0 to PositionMode.
00255000         Call "PPRINTRASTERA" using ComArea,
00256000         RasterNumber,
00257000         XPosition,
00258000         YPosition,
00259000         Units,
00260000         PositionMode.
00261000         Perform Check-For-Error.
00262000
00263000         Call "PDELETERASTER" using ComArea,
00264000         RasterNumber.

```

```

00265000           Perform Check-For-Error.
00266000
00267000           Generate-Exit.
00268000
00269000           Exit.
00270000
00271000
00272000           Check-For-Error Section.
00273000
00274000           *
00275000           * This routine checks the ComArea's IFSStatus to see if an error
00276000           * occurred. If so, then PERRMSG is called, the error is
00277000           * reported, and the program terminates.
00278000           *
00279000           If IFSStatus of ComArea Not = 0 then
00279100             Move Spaces to DisplayErrorNum
00279200             Move ErrorNum of ComArea to DisplayErrorNum
00280000             Display "{IFS Error ", DisplayErrorNum, "}"
00281000             Move Spaces to ErrorMessage
00282000             Move 80 to ErrorMessageLen
00283000             Move Zero to MsgLen
00284000             Call "PERRMSG" using ComArea,
00285000                                 ErrorMessage,
00286000                                 ErrorMessageLen,
00287000                                 MsgLen
00288000             Display ErrorMessage
00289000             Stop Run.
00290000
00291000           Check-Exit.
00292000
00293000           Exit.

```

```

DATA AREA IS %005017 WORDS.
CPU TIME = 0:00:04. WALL TIME = 0:00:18.
END COBOL/3000 COMPILATION. NO ERRORS. NO WARNINGS.

```

BASIC

```
10 REM Program to generate a Quarterly report
20 REM
30 REM
40 REM This sample program illustrates the calling sequences for
50 REM several of the IFS/3000 intrinsics from BASIC.
60 REM The program opens a SpoolFile with an environment with
70 REM the following characteristics:
80 REM
90 REM - 2 logical pages:
100 REM #0 at 90 degrees, no form, initially inactive.
110 REM #1 at 90 degrees, letterhead form, initially active.
120 REM
130 REM - 2 character sets:
140 REM #0 Elite.
150 REM #1 Elite Italics.
160 REM
170 REM - Standard 3/3 VFC.
180 REM
190 REM
200 REM *****
210 REM
220 REM Variable          Definition          Equivalent SPL Type
230 REM
240 REM C1              ComArea              Logical Array
250 REM D1$            DeviceClass          Byte Array
260 REM F1$            FormalDesignator    Byte Array
270 REM F2              File System ErrNum  Integer
280 REM L1              Language             Integer
290 REM X1              XLocation            Integer
300 REM Y1              YLocation            Integer
310 REM X2              XPosition            Real
320 REM Y2              YPosition            Real
330 REM P1              PrimaryFont          Integer
340 REM P2              SecondaryFont        Integer
350 REM S1              SpoolFileNum         Integer
360 REM A1              AreaLen              Integer
370 REM F2$            FileName             Byte Array
380 REM S2              SubFieldNum          Integer
390 REM D2$            Data                  Byte Array
400 REM D2              DataLength           Integer
410 REM O1$            OutputDevice         Byte Array
420 REM L2              LogPageNum           Integer
430 REM N1$            FigureName           Byte Array
440 REM N2$            FigureFileName       Byte Array
450 REM R1$            RasterFileName       Byte Array
460 REM H1              Height               Real
470 REM R1              Rotation             Integer
480 REM U1              Units                Integer
490 REM R2              RasterNumber         Integer
500 REM M1              PositionMode         Integer
510 REM E1$            ErrorString          Byte Array
520 REM M2              MessageLength        Integer
```

```

530 REM          L3                ErrorStringLen      Integer
540 REM          Z1                SpoolFileOpenErr    Integer
550 REM          Z1$              SpoolFileOpenString  Byte Array
560 REM
570 REM
580 REM
590 REM
600 INTEGER C1[2000]
610 INTEGER F2,L1,X1,Y1,P1,P2,S1,A1,S2,D2,L2,R1,U1,R2,M1,M2,L3,Z1
620 REM
630 DIM D1$[15],F1$[9],F2$[16],D2$[127],O1$[6],N1$[35],N2$[35]
640 DIM R1$[35],E1$[80],Z1$[80]
650 REM
660 FILES *
670 REM
680 REM
690 REM First, Open the SpoolFile
700 REM
710 LET F1$="SPOOLOUT "
720 LET D1$="PP:ENV=MEMOENV "
730 LET Z1$="FILE "+F1$+";DEV="+D1$
740 SYSTEM Z1,Z1$
750 IF Z1<>0 THEN DO
760     PRINT "Cannot issue file equation for SpoolFile."
770     PRINT "System error number = ";Z1
780     STOP
790 DOEND
800 ASSIGN F1$,1,Z1
810 IF Z1<>8 THEN DO
820     PRINT "Cannot open SpoolFile."
830     PRINT "System error number = ";Z1
840     STOP
850 DOEND
860 REM
870 REM Get the file system number using standard function SFN.
880 REM
890 LET S1=SFN(1)
900 REM
910 REM Next, Initialize the IFS/3000 intrinsics.
920 REM
930 LET A1=2000
940 LET L1=1
950 LET O1$="2680A"
960 CALL PINITDEVICE(C1[*],A1,L1,S1,O1$)
970 GOSUB 2040
980 REM
990 REM Now Write to the four fields
1000 REM
1010 REM All fields have only 1 subfield.
1020 LET S2=-1
1030 LET F2$="FROM"
1040 LET D2$="          Profit Monitor"

```

Jan 84

F-10

```

1050 LET D2=20
1060 CALL PWRITEFIELD(C1[*],F2$,S2,D2$,D2)
1070 GOSUB 2040
1080 REM
1090 LET F2$="TO"
1100 LET D2$="          All Regional Managers"
1110 LET D2=27
1120 CALL PWRITEFIELD(C1[*],F2$,S2,D2$,D2)
1130 GOSUB 2040
1140 REM
1150 LET F2$="DATE"
1160 LET D2$="          August 1, 1982"
1170 LET D2=24
1180 CALL PWRITEFIELD(C1[*],F2$,S2,D2$,D2)
1190 GOSUB 2040
1200 REM
1210 LET F2$="SUBJECT"
1220 LET D2$="          Last Quarter's Sales"
1230 LET D2=34
1240 CALL PWRITEFIELD(C1[*],F2$,S2,D2$,D2)
1250 GOSUB 2040
1260 REM
1270 REM
1280 REM Now move the pen.
1290 REM
1300 REM Go Over 1 inch.
1310 LET X1=180
1320 REM Go down 4 inches.
1330 LET Y1=720
1340 CALL PMOVEPENABS(C1[*],X1,Y1)
1350 GOSUB 2040
1360 REM
1370 REM Switch the Primary font to Italics.
1380 REM
1390 LET P1=1
1400 REM The secondary font remains unchanged.
1410 LET P2=-1
1420 CALL PUSEFONT(C1[*],P1,P2)
1430 GOSUB 2040
1440 REM
1450 REM Now write out some text in the new font.
1460 REM
1470 PRINT #1:"See page 2 for last quarter's sales figures."
1480 REM
1490 REM Now, activate logical page 0 (no form) and deactivate
1500 REM logical page 1 (letterhead form) and skip to the new page.
1510 REM
1520 LET L2=0
1530 CALL PACTIVATEPAGE(C1[*],L2)
1540 GOSUB 2040
1550 LET L2=1
1560 CALL PDEACTIVATEPAGE(C1[*],L2)

```

```

1570 GOSUB 2040
1580 REM
1590 CALL PNEWPAGE(C1[*])
1600 GOSUB 2040
1610 REM
1620 REM Now convert the figure created with DSG/3000 into a raster
1630 REM image.
1640 REM
1650 LET N2$="QTRSALES"
1660 LET N1$="SECOND QUARTER"
1670 LET R1$="R2NDQTR"
1680 REM specify inches
1690 LET U1=1
1700 REM 6 inches high
1710 LET H1=6
1720 REM 90 degree rotation
1730 LET R1=90
1740 CALL PCONVERTFIGURE(C1[*],N2$,N1$,R1$,O1$,H1,U1,R1)
1750 GOSUB 2040
1760 REM
1770 REM Now load the image as #17, print it at a position relative
1780 REM to the Upper-left corner of the logical page, and then
1790 REM delete it.
1800 REM
1810 LET R2=17
1820 CALL PLOADRASTER(C1[*],R1$,R2)
1830 GOSUB 2040
1840 REM
1850 REM 1.5 inches to the right
1860 LET X2=1.5
1870 REM 2.0 inches down the page
1880 LET Y2=2
1890 REM indicate relative positioning
1900 LET M1=0
1910 CALL PPRINTRASTER(C1[*],R2,X2,Y2,U1,M1)
1920 GOSUB 2040
1930 REM
1940 CALL PDELETERASTER(C1[*],R2)
1950 GOSUB 2040
1960 REM
1970 REM Done.
1980 END
1990 REM
2000 REM
2010 REM
2020 REM
2030 REM
2040 REM      Subroutine CHECK_FOR_ERROR
2050 REM
2060 REM This routine checks the ComArea's Status to see if an error
2070 REM occurred. If so, then PERRMSG is called, the error is
2080 REM reported and the program terminates.

```

Jan 84

F-12

```
2090 REM
2100 REM   C1[1] is Status of the ComArea,
2110 REM   C1[2] is the Error number.
2120 IF C1[1]=0 THEN RETURN
2130 REM
2140 REM An error/warning has occurred.
2150 REM
2160 PRINT "(IFS Error ";C1[2];)"
2170 REM L3 is a dummy variable here.
2180 LET L3=0
2190 CALL PERRMSG(C1[*],E1$,L3,M2)
2200 PRINT E1$
2210 STOP
```

FORTRAN

```
00001000 $CONTROL USLINIT
00002000 C
00003000     program QUARTERLYREPORT
00004000 C
00005000 C
00006000 C
00007000 C
00008000 C This sample program illustrates the calling sequences for several
00009000 C of the IFS/3000 intrinsics from FORTRAN. The program opens a
00010000 C SpoolFile with an environment with the following characteristics:
00011000 C
00012000 C - 2 logical pages:
00013000 C   #0 at 90 degrees, no form, initially inactive.
00014000 C   #1 at 90 degrees, letterhead form, initially active.
00015000 C
00016000 C - 2 character sets:
00017000 C   #0 Elite.
00018000 C   #1 Elite Italics.
00019000 C
00020000 C - Standard 3/3 VFC.
00021000 C
00022000 C
00023000 C Note that all error messages are written to logical unit number 5,
00024000 C which must be equated with a file or device; e.g.
00025000 C
00026000 C   :FILE FTN05=$STDLIST
00027000 C
00028000 C
00029000 C
00030000 C
00031000     external CHECKFORERROR
00032000 C
00033000     system intrinsic FOPEN, FCLOSE, FCHECK, FWRITE, TERMINATE,
00034000     &PINITDEVICE, PERRMSG, PWRITEFIELD, PMOVEPENABS, PMOVEPENREL,
00035000     &PUSEFONT, PACTIVATEPAGE, PDEACTIVATEPAGE, PNEWPAGE,
00036000     &PCONVERTFIGURE, PLOADRASTER, PPRINTRASTER, PDELETERASTER
00037000 C
00038000     character*16 DeviceClass
00039000     character*1 DevClassEquiv(16)
00040000     equivalence (DeviceClass, DevClassEquiv)
00041000     character*255 CharArray
00042000     character*9 FormalDesig
00043000     integer FSErr
00044000     logical ComArea(2000)
00045000     integer Status
00046000     integer ErrorNum
00047000     integer ErrorParm1
00048000     integer ErrorParm2
00049000     integer MsgFileNum
00050000     equivalence (ComArea(1), Status)
00051000     equivalence (ComArea(2), ErrorNum)
00052000     equivalence (ComArea(3), ErrorParm1)
```

```

00053000      equivalence (ComArea(4), ErrorParm2)
00054000      equivalence (ComArea(5), MsgFileNum)
00055000      integer Language
00056000      real XPosition
00057000      real YPosition
00058000      integer PrimaryFont
00059000      integer SecondaryFont
00060000      integer SpoolFileNum
00061000      integer OldFileNum
00062000      integer AreaLen
00063000      character*16 FieldName
00064000      integer SubFieldNum
00065000      character*255 Data
00066000      integer DataLen
00067000      character*6 OutputDev
00068000      integer XLoc
00069000      integer YLoc
00070000      integer LogPageNum
00071000      character*17 FigureName
00072000      character*36 FigureFileName
00073000      character*36 RasterFileName
00074000      real Height
00075000      integer Rotation
00076000      integer Units
00077000      integer RasterNumber
00078000      integer PositionMode
00079000      C

00081000      C
00082000      C
00083000      C      First, Open the Spoolfile.
00084000      C
00085000      C
00086000      FormalDesig = "SPOOLOUT "
00087000      DeviceClass = "PP;ENV=MEMOENV"
00088000      DevClassEquiv(15) = %15C
00089000      C      FOPTIONS are: CCTL, Undefined, Formal, ASCII, New file.
00090000      C      AOPION are: Write Only.
00091000      SpoolFileNum = FOPEN (FormalDesig,
00092000      &                               \%604L\,
00093000      &                               \1\,
00094000      &                               DeviceClass)
00095000      &
00096000      if (.CC.) 10, 20, 10
00097000      C      FOPEN error.
00098000      10      write (5,10000)
00099000      10000  format(1X, "Cannot Open SpoolFile.")
00100000      call FCHECK (SpoolFileNum, FSErr)
00101000      write (5,20000) FSErr
00102000      20000  format(1X, "{FSERR ", I3, "}")
00103000      call TERMINATE
00104000      C
00105000      20      continue

```

```

00106000 C
00107000 C      Associate logical unit number 11 with the SpoolFileNum
00108000 C
00109000      call FSET(11, SpoolFileNum, OldFileNum)
00110000 C
00111000 C
00112000 C      Next, Initialize the IFS/3000 intrinsics.
00113000 C
00114000 C
00115000      AreaLen = 2000
00116000      Language = 2
00117000      OutputDev = "2680A "
00118000      call PINITDEVICE (ComArea,AreaLen,Language,SpoolFileNum,OutputDev)
00119000      call CHECKFORERROR (Status, ErrorNum)
00120000 C
00121000 C
00122000 C      Now Write to the four fields.
00123000 C
00124000 C
00125000 C      All Fields have only 1 subfield.
00126000      SubFieldNum = -1
00127000 C
00128000      FieldName = "FROM "
00129000      Data = "      Profit Monitor"
00130000      DataLen = 20
00131000      call PWRITEFIELD (ComArea,Fieldname,SubFieldNum,Data,DataLen)
00132000      call CHECKFORERROR (Status, ErrorNum)
00133000 C
00134000      FieldName = "TO "
00135000      Data = "      All Regional Managers"
00136000      DataLen = 27
00137000      call PWRITEFIELD (ComArea,FieldName,SubFieldNum,Data,DataLen)
00138000      call CHECKFORERROR (Status, ErrorNum)
00139000 C
00140000      FieldName = "DATE "
00141000      Data = "      August 1, 1982"
00142000      DataLen = 24
00143000      call PWRITEFIELD (ComArea,FieldName,SubFieldNum,Data,DataLen)
00144000      call CHECKFORERROR (Status, ErrorNum)
00145000 C
00146000      FieldName = "SUBJECT "
00147000      Data = "      Last Quarter's Sales"
00148000      DataLen = 34
00149000      call PWRITEFIELD (ComArea,FieldName,SubFieldNum,Data,DataLen)
00150000      call CHECKFORERRCR (Status, ErrorNum)
00151000 C
00152000 C
00153000 C      Now move the pen.
00154000 C
00155000 C
00156000 C      1 inch over
00157000      XLoc = 180

```

```

00158000 C      4 inches down.
00159000      YLoc = 720
00160000      call PMOVEPENABS (ComArea, XLoc, YLoc)
00161000      call CHECKFORERROR (Status, ErrorNum)
00162000 C
00163000 C
00164000 C      Switch The Primary Font to Italics.
00165000 C
00166000 C
00167000      PrimaryFont = 1
00168000 C      SecondaryFont remains unchanged.
00169000      SecondaryFont = -1
00170000      call PUSEFONT (ComArea, PrimaryFont, SecondaryFont)
00171000      call CHECKFORERROR (Status, ErrorNum)
00172000 C
00173000 C
00174000 C      Now write out some text in the new font.
00175000 C
00176000 C
00177000      write (11, 30000)
00178000 30000 format(IX, "See page 2 for last quarter's sales figures.")
00179000 C
00180000 C
00181000 C      Now, activate logical page 0 (no form) and deactivate logical
00182000 C      page 1 (letterhead form) and skip to the new page.
00183000 C
00184000 C
00185000      LogPageNum = 0
00186000      call PACTIVATEPAGE (ComArea, LogPageNum)
00187000      call CHECKFORERROR (Status, ErrorNum)
00188000 C
00189000      LogPageNum = 1
00190000      call PDEACTIVATEPAGE (ComArea, LogPageNum)
00191000      call CHECKFORERROR (Status, ErrorNum)
00192000 C
00193000      call PNEWPAGE (ComArea)
00194000      call CHECKFORERROR (Status, ErrorNum)
00195000 C
00196000 C
00197000 C      Now Convert the figure created with DSG/3000 into a raster
00198000 C      image.
00199000 C
00200000 C
00201000      FigureFileName = "QTRSALES "
00202000      FigureName = "SECOND QUARTER "
00203000      RasterFileName = "R2NDQTR "
00204000 C      Indicate inches.
00205000      Units = 1
00206000      Height = 6.0
00207000      Rotation = 90
00208000      call PCONVERTFIGURE (ComArea, FigureFileName, FigureName,
00209000 &      RasterFileName, OutputDev, Height, Units, Rotation)

```

```

00210000      call CHECKFORERROR (Status, ErrorNum)
00211000      C
00212000      C
00213000      C      Now Load the image as #17, print it at a position relative to
00214000      C      the Upper-left corner of the logical page, and then delete it.
00215000      C
00216000      C
00217000      RasterNumber = 17
00218000      call PLOADRASTER (ComArea, RasterFilename, RasterNumber)
00219000      call CHECKFORERROR (Status, ErrorNum)
00220000      C
00221000      C      Place it 1.5 inches to the right, 2.0 inches down the page.
00222000      XPosition = 1.5
00223000      YPosition = 2.0
00224000      C      relative positioning.
00225000      PositionMode = 0
00226000      call PPRINTRASTER (ComArea, RasterNumber, XPosition, YPosition,
00227000      & Units, PositionMode)
00228000      call CHECKFORERROR (Status, ErrorNum)
00229000      C
00230000      call PDELETERASTER (ComArea, RasterNumber)
00231000      call CHECKFORERROR (Status, ErrorNum)
00232000      C
00233000      C
00234000      C      Finally, close the spoolfile.
00235000      C
00236000      C
00237000      call FCLOSE (SpoolFileNum, 0, 0)
00238000      C
00239000      C
00240000      stop
00241000      end
00243000      subroutine CHECKFORERROR (Status, ErrorNum)
00244000      C
00245000      integer Status
00246000      integer ErrorNum
00247000      C
00248000      C
00249000      C      This routine checks the ComArea's Status to see if an error
00250000      C      occurred. If so, then PERMSG is called, the error is reported,
00251000      C      and the program terminates.
00252000      C
00253000      C
00254000      character*1 ErrorMessage(80)
00255000      integer ArrayLen
00256000      integer MsgLen
00257000      data ErrorMessage /80*" "/
00258000      C
00259000      if (Status .EQ. 0) return
00260000      C      An Error/Warning occurred.
00261000      write (5,40000) ErrorNum
00262000      40000 format(1X, "(IFS Error ", I4, ")")

```

Jan 84

F-18

```
00263000      ArrayLen = 80
00264000      call PERRMSG (ComArea, ErrorMessage, ArrayLen, MsgLen)
00265000      display ErrorMessage
00266000      call TERMINATE
00267000      return
00268000      end
```

SPL

```
00001000 00000 0 $CONTROL USLINIT
00002000 00000 0
00003000 00000 0 begin << Quarterly Report Program >>
00004000 00000 1
00005000 00000 1
00006000 00000 1
00007000 00000 1 <<
00008000 00000 1 << This sample program illustrates the calling sequences for several
00009000 00000 1 << of the IFS/3000 intrinsics from SPL. The program opens a
00010000 00000 1 << SpoolFile with an environment with the following characteristics:
00011000 00000 1 <<
00012000 00000 1 << - 2 logical pages:
00013000 00000 1 << #0 at 90 degrees, no form, initially inactive.
00014000 00000 1 << #1 at 90 degrees, letterhead form, initially active.
00015000 00000 1 <<
00016000 00000 1 << - 2 character sets:
00017000 00000 1 << #0 Elite.
00018000 00000 1 << #1 Elite Italics.
00019000 00000 1 <<
00020000 00000 1 << - Standard 3/3 VFC.
00021000 00000 1 <<
00022000 00000 1 << >>
00023000 00000 1
00024000 00000 1 equate
00025000 00000 1 cMaxComAreaLen = 2000,
00026000 00000 1 cSPL = 3,
00027000 00000 1 cStatusOk = 0;
00028000 00000 1
00029000 00000 1 byte array DeviceClass (0:14);
00030000 00000 1 logical array LogArray(0:128);
00031000 00000 1 byte array CharArray (*) = LogArray;
00032000 00000 1 byte array FormalDesignator(0:8);
00033000 00000 1 integer FSErr;
00034000 00000 1 logical array ComArea(0:cMaxComAreaLen-1);
00035000 00000 1 define Status = ComArea(0)#;
00036000 00000 1 define ErrorNum = ComArea(1)#;
00037000 00000 1 define ErrorParm1 = ComArea(2)#;
00038000 00000 1 define ErrorParm2 = ComArea(3)#;
00039000 00000 1 define MsgFileNum = ComArea(4)#;
00040000 00000 1 integer Language;
00041000 00000 1 real XPosition;
00042000 00000 1 real YPosition;
00043000 00000 1 integer PrimaryFont;
00044000 00000 1 integer SecondaryFont;
00045000 00000 1 integer SpoolFileNum;
00046000 00000 1 integer AreaLen;
00047000 00000 1 byte array FieldName(0:15);
00048000 00000 1 integer SubFieldNum;
00049000 00000 1 byte array Data(0:255);
00050000 00000 1 integer DataLen;
00051000 00000 1 byte array OutputDev(0:5) := "2680A ";
00052000 00004 1 integer XLoc;
00053000 00004 1 integer YLoc;
```

Jan 84

F-20

```

00054000 00004 1 integer LogPageNum;
00055000 00004 1 byte array FigureName(0:16);
00056000 00004 1 byte array FigureFileName(0:35);
00057000 00004 1 byte array RasterFileName(0:35);
00058000 00004 1 real Height;
00059000 00004 1 integer Rotation;
00060000 00004 1 integer Units;
00061000 00004 1 integer RasterNumber;
00062000 00004 1 integer PositionMode;
00063000 00004 1 byte array AsciiNum(0:9);
00064000 00004 1 logical array L'AsciiNum(*) = AsciiNum;
00065000 00004 1 byte array Msg(0:79);
00066000 00004 1 logical array L'Msg(*) = Msg;
00067000 00004 1 integer NumChar;
00068000 00004 1
00069000 00004 1 intrinsic
00070000 00004 1 FOPEN, FCLOSE, FCHECK, FWRITE, TERMINATE, ASCII, PRINT,
00071000 00004 1 PINITDEVICE, PERRMSG, PWRITEFIELD, PMOVEPENABS, PMOVEPENREL,
00072000 00004 1 PUSEFONT, PACTIVATEPAGE, PDEACTIVATEPAGE, PNEWPAGE, PCONVERTFIGURE,
00073000 00004 1 PLOADRASTER, PPRINTRASTER, PDELETERASTER;
00074000 00004 1
00076000 00004 1 procedure CHECK'FOR'ERROR;
00077000 00000 1
00078000 00000 1 <<
00079000 00000 1 << This routine checks the ComArea's Status to see if an error
00080000 00000 1 << occurred. If so, then PERRMSG is called, the error is reported,
00081000 00000 1 << and the program terminates.
00082000 00000 1 << >>
00083000 00000 1
00084000 00000 1 begin <<CHECK'FOR'ERROR>>
00085000 00000 2 byte array ErrorMsg(0:79);
00086000 00000 2 integer ArrayLen;
00087000 00000 2 integer MsgLen;
00088000 00000 2 if Status <> cStatusOk then
00089000 00011 2 begin <<Get error/warning message>>
00090000 00011 3 move Msg := "(IFS Error ";
00091000 00026 3 PRINT (L'Msg, -11, %320);
00092000 00032 3 NumChar := ASCII (ErrorNum, 10, AsciiNum);
00093000 00041 3 PRINT (L'AsciiNum, -NumChar, %320);
00094000 00046 3 move Msg := ")";
00095000 00055 3 PRINT (L'Msg, -1, 0);
00096000 00061 3 ArrayLen := 80;
00097000 00063 3 PERRMSG (ComArea, ErrorMsg, ArrayLen, MsgLen);
00098000 00070 3 move Msg := ErrorMessage, (MsgLen);
00099000 00074 3 PRINT (L'Msg, -MsgLen, 0);
00100000 00100 3 TERMINATE;
00101000 00101 3 end; <<Get error/warning message>>
00102000 00101 2 end; <<CHECK'FOR'ERROR>>
00103000 00000 1
00105000 00000 1
00106000 00000 1 <<
00107000 00000 1 << First, Open the Spoolfile.

```

```

00108000 00000 1      << >>
00109000 00000 1
00110000 00000 1      move FormalDesignator := "SPOOLOUT ";
00111000 00013 1      move DeviceClass := "PP;ENV=MEMOENV";
00112000 00030 1      DeviceClass(14) := %15; << Required to terminate string properly >>
00113000 00033 1      SpoolFileNum := FOPEN (FormalDesignator,
00114000 00035 1          %604, <<CTL, Undef, Formal, ASCII, New>>
00115000 00036 1          1, <<Write only>>
00116000 00037 1          <<Default RecSize>>
00117000 00037 1          DeviceClass);
00118000 00045 1      if <> then
00119000 00046 1      begin <<FOPEN error>>
00120000 00046 2          move Msg := "Cannot Open SpoolFile";
00121000 00072 2          PRINT (L'Msg, -21, 0);
00122000 00076 2          FCHECK (SpoolFileNum, FSErr);
00123000 00103 2          NumChar := ASCII (FSErr, 10, AsciiNum);
00124000 00111 2          move Msg := "(FSERR ";
00125000 00123 2          PRINT (L'Msg, -7, %320);
00126000 00127 2          PRINT (L'AsciiNum, -NumChar, %320);
00127000 00134 2          move Msg := ")";
00128000 00143 2          PRINT (L'Msg, -1, 0);
00129000 00147 2          TERMINATE;
00130000 00150 2      end; <<FOPEN error>>
00131000 00150 1
00132000 00150 1      <<
00133000 00150 1      << Next, Initialize the IFS/3000 intrinsics.
00134000 00150 1      << >>
00135000 00150 1
00136000 00150 1      AreaLen := cMaxComAreaLen;
00137000 00152 1      Language := cSPL;
00138000 00154 1      PINITDEVICE (ComArea, AreaLen, Language, SpoolFileNum, OutputDev);
00139000 00162 1      CHECK'FOR'ERROR;
00140000 00163 1
00141000 00163 1      <<
00142000 00163 1      << Now Write to the four fields.
00143000 00163 1      << >>
00144000 00163 1
00145000 00163 1      SubFieldNum := -1; <<All fields have only 1 subfield>>
00146000 00165 1
00147000 00165 1      move FieldName := "FROM ";
00148000 00177 1      move Data := "          Profit Monitor";
00149000 00217 1      DataLen := 20;
00150000 00221 1      PWRITEFIELD (ComArea, Fieldname, SubFieldNum, Data, DataLen);
00151000 00227 1      CHECK'FOR'ERROR;
00152000 00230 1
00153000 00230 1      move FieldName := "TO ";
00154000 00240 1      move Data := "          All Regional Managers";
00155000 00264 1      DataLen := 27;
00156000 00266 1      PWRITEFIELD (ComArea, FieldName, SubFieldNum, Data, DataLen);
00157000 00274 1      CHECK'FOR'ERROR;
00158000 00275 1

```

```

00159000 00275 1      move fieldName := "DATE ";
00160000 00306 1      move Data := "          August 1, 1982";
00161000 00330 1      DataLen := 24;
00162000 00332 1      PWRITEFIELD (ComArea, fieldName, SubFieldNum, Data, DataLen);
00163000 00340 1      CHECK'FOR'ERROR;
00164000 00341 1
00165000 00341 1      move fieldName := "SUBJECT ";
00166000 00353 1      move Data := "          Last Quarter's Sales";
00167000 00402 1      DataLen := 34;
00168000 00404 1      PWRITEFIELD (ComArea, fieldName, SubFieldNum, Data, DataLen);
00169000 00412 1      CHECK'FOR'ERROR;
00170000 00413 1
00171000 00413 1      <<
00172000 00413 1      << Now move the pen.
00173000 00413 1      << >>
00174000 00413 1
00175000 00413 1      XLoc := 180; <<1 Inch>>
00176000 00415 1      YLoc := 720; <<4 Inches>>
00177000 00417 1      MOVEPENABS (ComArea, XLoc, YLoc);
00178000 00423 1      CHECK'FOR'ERROR;
00179000 00424 1
00180000 00424 1      <<
00181000 00424 1      << Switch The Primary Font to Italics.
00182000 00424 1      << >>
00183000 00424 1
00184000 00424 1      PrimaryFont := 1;
00185000 00426 1      SecondaryFont := -1; <<Remains unchanged>>
00186000 00430 1      PUSEFONT (ComArea, PrimaryFont, SecondaryFont);
00187000 00434 1      CHECK'FOR'ERROR;
00188000 00435 1
00189000 00435 1      <<
00190000 00435 1      << Now write out some text in the new font.
00191000 00435 1      << >>
00192000 00435 1
00193000 00435 1      move CharArray := "See page 2 for last quarter's sales figures.";
00194000 00472 1      DataLen := 45;
00195000 00474 1      FWRITE (SpoolFileNum, LogArray, -DataLen, 0);
00196000 00501 1      if <> then
00197000 00502 1      begin <<FWRITE error>>
00198000 00502 2          move Msg := "FWRITE failed.";
00199000 00517 2          PRINT (L'Msg, -14, 0);
00200000 00523 2          TERMINATE;
00201000 00524 2      end; <<FWRITE error>>
00202000 00524 1
00203000 00524 1      <<
00204000 00524 1      << Now, activate logical page 0 (no form) and deactivate logical page
00205000 00524 1      << 1 (letterhead form) and skip to the new page.
00206000 00524 1      << >>
00207000 00524 1
00208000 00524 1      LogPageNum := 0;
00209000 00526 1      PACTIVATEPAGE (ComArea, LogPageNum);
00210000 00531 1      CHECK'FOR'ERROR;
00211000 00532 1

```

```

00212000 00532 1      LogPageNum := 1;
00213000 00534 1      PDEACTIVATEPAGE (ComArea, LogPageNum);
00214000 00537 1      CHECK'FOR'ERROR;
00215000 00540 1
00216000 00540 1      PNEWPAGE (ComArea);
00217000 00542 1      CHECK'FOR'ERROR;
00218000 00543 1
00219000 00543 1      <<
00220000 00543 1      << Now Convert the figure created with DSG/3000 into a raster image.
00221000 00543 1      << >>
00222000 00543 1
00223000 00543 1      move FigureFileName := "QTRSALES ";
00224000 00556 1      move FigureName := "SECOND QUARTER "; <<Terminated with a blank>>
00225000 00574 1      move RasterFileName := "R2NDQTR "; <<Terminated with a blank>>
00226000 00614 1      Units := 1; <<Inches>>
00227000 00616 1      Height := 6.0;
00228000 00620 1      Rotation := 90;
00229000 00622 1      PCONVERTFIGURE (ComArea, FigureFileName, FigureName, RasterFileName,
00230000 00626 1      OutputDev, Height, Units, Rotation);
00231000 00633 1      CHECK'FOR'ERROR;
00232000 00634 1
00233000 00634 1      <<
00234000 00634 1      << Now Load the image as #17, print it at a position relative to the
00235000 00634 1      << Upper-left corner of the logical page, and then delete it.
00236000 00634 1      << >>
00237000 00634 1
00238000 00634 1      RasterNumber := 17;
00239000 00636 1      PLOADRASTER (ComArea, RasterFilename, RasterNumber);
00240000 00642 1      CHECK'FOR'ERROR;
00241000 00643 1
00242000 00643 1      XPosition := 1.5; <<Inches to the right>>
00243000 00645 1      YPosition := 2.0; <<Inches down the page>>
00244000 00647 1      PositionMode := 0; <<Relative>>
00245000 00651 1      PPRINTRASTER (ComArea, RasterNumber, XPosition, YPosition, Units,
00246000 00656 1      PositionMode);
00247000 00660 1      CHECK'FOR'ERROR;
00248000 00661 1
00249000 00661 1      PDELETERASTER (ComArea, RasterNumber);
00250000 00664 1      CHECK'FOR'ERROR;
00251000 00665 1
00252000 00665 1      <<
00253000 00665 1      << Finally, close the spoolfile.
00254000 00665 1      << >>
00255000 00665 1
00256000 00665 1      FCLOSE (SpoolFileNum, 0, 0);
00257000 00670 1
00258000 00670 1      end. <<Quarterly Report Program>>
PRIMARY DB STORAGE=%045; SECONDARY DB STORAGE=%04503
NO. ERRORS=0000; NO. WARNINGS=0000
PROCESSOR TIME=0:00:05; ELAPSED TIME=0:00:34

```

Pascal

```
1.000      0 0  $USLINIT, STANDARD_LEVEL 'HP3000'$
2.000      0 0
3.000      0 0  program Quarterly_Report (input, output);
4.000      0 0
5.000      0 0  {
6.000      ** 0  { This sample program illustrates the calling sequences for several
7.000      ** 0  { of the IFS/3000 intrinsics from Pascal. The program opens a
8.000      ** 0  { SpoolFile with an environment with the following characteristics:
9.000      ** 0  {
10.000     ** 0  { - 2 logical pages:
11.000     ** 0  {   #0 at 90 degrees, no form, initially inactive.
12.000     ** 0  {   #1 at 90 degrees, letterhead form, initially active.
13.000     ** 0  {
14.000     ** 0  { - 2 character sets:
15.000     ** 0  {   #0 Elite.
16.000     ** 0  {   #1 Elite Italics.
17.000     ** 0  {
18.000     ** 0  { - Standard 3/3 VFC.
19.000     ** 0  {
20.000     0 0  {
21.000     0 0
22.000     0 0  const
23.000     0 0      CCE                = 2;
24.000     0 0      cCR                 = chr(13);
25.000     0 0      cStatusOk           = 0;
26.000     0 0
27.000     0 0  type
28.000     0 0      SmallInteger        = -32768..32767;
29.000     0 0      ComAreaRec          = record
30.000     0 0          Status           : SmallInteger;
31.000     0 0          ErrorNum        : SmallInteger;
32.000     0 0          ErrParm1       : SmallInteger;
33.000     0 0          ErrParm2       : SmallInteger;
34.000     0 0          MsgFileNum     : SmallInteger;
35.000     0 0          Reserved       : array [1..1995]
36.000     0 0              of SmallInteger;
37.000     0 0      end;
38.000     0 0      DevArray            = packed array [1..15] of char;
39.000     0 0
40.000     0 0  const
41.000     0 0      cDevParm            = DevArray ['PP;ENV=MEMOENV',cCR];
42.000     0 0
43.000     0 0  type
44.000     0 0      TagType              = (ByteAddr, WordAddr);
45.000     0 0      CharArrayType        = packed array [1..80] of char;
46.000     0 0      LogArrayType         = array [1..40] of SmallInteger;
47.000     0 0      LogicalRec          = record
48.000     0 0          dummybyte : char; {To keep data aligned}
49.000     0 0          case AddrTag : TagType of
50.000     0 0              ByteAddr : (CharArray : CharArrayType);
51.000     0 0              WordAddr : (LogArray  : LogArrayType);
52.000     0 0      end;
53.000     0 0  $PAGE$
```

```

54.000      0 0      var
55.000      0 0      DeviceClass          : DevArray;
56.000      0 0      FormalDesignator      : packed array [1..9] of char;
57.000      0 0      FSErr                 : SmallInteger;
58.000      0 0      ComArea                : ComAreaRec;
59.000      0 0      Language                : SmallInteger;
60.000      0 0      XPosition               : real;
61.000      0 0      YPosition               : real;
62.000      0 0      PrimaryFont             : SmallInteger;
63.000      0 0      SecondaryFont           : SmallInteger;
64.000      0 0      TextArray               : LogicalRec;
65.000      0 0      SpoolFileNum            : SmallInteger;
66.000      0 0      AreaLen                 : SmallInteger;
67.000      0 0      FieldName                : packed array [1..16] of char;
68.000      0 0      SubFieldName            : SmallInteger;
69.000      0 0      Data                    : packed array [1..256] of char;
70.000      0 0      Datalen                 : SmallInteger;
71.000      0 0      OutputDev                : packed array [1..6] of char;
72.000      0 0      XLoc                    : SmallInteger;
73.000      0 0      YLoc                    : SmallInteger;
74.000      0 0      LogPageNum              : SmallInteger;
75.000      0 0      FigureName              : packed array [1..16] of char;
76.000      0 0      FigureFileName          : packed array [1..35] of char;
77.000      0 0      RasterFileName          : packed array [1..35] of char;
78.000      0 0      Height                  : real;
79.000      0 0      Rotation                : SmallInteger;
80.000      0 0      Units                    : SmallInteger;
81.000      0 0      RasterNumber            : SmallInteger;
82.000      0 0      PositionMode            : SmallInteger;
83.000      0 0
84.000      0 0      function FOPEN : SmallInteger; intrinsic;
85.000      0 0      procedure FCLOSE; intrinsic;
86.000      0 0      procedure FCHECK; intrinsic;
87.000      0 0      procedure FWRITE; intrinsic;
88.000      0 0      procedure TERMINATE; intrinsic;
89.000      0 0      procedure PINITDEVICE; intrinsic;
90.000      0 0      procedure PERRMSG; intrinsic;
91.000      0 0      procedure PWRITEFIELD; intrinsic;
92.000      0 0      procedure PMOVEPENABS; intrinsic;
93.000      0 0      procedure PUSEFONT; intrinsic;
94.000      0 0      procedure PMOVEPENREL; intrinsic;
95.000      0 0      procedure PACTIVATEPAGE; intrinsic;
96.000      0 0      procedure PDEACTIVATEPAGE; intrinsic;
97.000      0 0      procedure PNEWPAGE; intrinsic;
98.000      0 0      procedure PCONVERTFIGURE; intrinsic;
99.000      0 0      procedure PLOADRASTER; intrinsic;
100.000     0 0      procedure PPRINTRASTER; intrinsic;
101.000     0 0      procedure PDELETERASTER; intrinsic;
102.000     0 0
103.000     0 0      $PAGE$

```

```

104.000      0 0      procedure CHECK_FOR_ERROR;
105.000      0 0
106.000      0 0      {
107.000      ** 0      { This routine checks the ComArea's Status to see if an error
108.000      ** 0      { occurred. If so, then PERRMSG is called, the error is reported,
109.000      ** 0      { and the program terminates.
110.000      0 0      { }
111.000      0 0
112.000      0 0
113.000      0 0      var
114.000      0 0      ErrorMsg      : packed array [1..80] of char;
115.000      0 0      ArrayLen      : SmallInteger;
116.000      0 0      MsgLen        : SmallInteger;
117.000      0 0
118.000      0 0
119.000      0 1      begin {CHECK_FOR_ERROR}
120.000      0 1      with ComArea do
121.000      1 2      begin {with}
122.000      1 2      if Status <> cStatusOk then
123.000      2 3      begin {Get error/warning message}
124.000      2 3      writeln(output, '(IFS Error ', ErrorNum:1, ')');
125.000      3 3      ArrayLen := 80;
126.000      4 3      PERRMSG (ComArea, ErrorMsg, ArrayLen, MsgLen);
127.000      5 3      writeln(output, ErrorMsg:MsgLen);
128.000      6 3      TERMINATE;
129.000      6 3      end; {Get error/warning message}
130.000      6 2      end; {with}
131.000      6 1      end; {CHECK_FOR_ERROR}
132.000      7 0
133.000      7 0      $PAGES$
134.000      0 1      begin {Quarterly_Report}
135.000      0 1
136.000      0 1      {
137.000      ** 1      { First, Open the Spoolfile.
138.000      0 1      { }
139.000      0 1
140.000      0 1      FormalDesignator := 'SPOOLOUT ';
141.000      1 1      DeviceClass := cDevParm;
142.000      2 1      SpoolFileNum := FOPEN (FormalDesignator,
143.000      3 1      octal('604'), {CCTL, Undef, Formal, ASCII, New}
144.000      3 1      1, {Write only}
145.000      3 1      {Default RecSize}
146.000      3 1      DeviceClass);
147.000      3 1
148.000      4 2      if ccode <> CCE then
149.000      4 2      begin {FOPEN error}
150.000      5 2      writeln(output, 'Cannot Open Spoolfile');
151.000      6 2      FCHECK (SpoolFileNum, FSErr);
152.000      7 2      writeln(output, '(FSERR ', FSErr:1, ')');
153.000      7 2      TERMINATE;
154.000      7 2      end; {FOPEN error}
154.000      8 1

```

```

155.000      8  1      {
156.000     **  1      { Next, Initialize the IFS/3000 intrinsics.
157.000      8  1      { }
158.000      8  1
159.000      8  1      AreaLen := 2000;
160.000      9  1      Language := 5;
161.000     10  1      OutputDev := '2680A';
162.000     11  1      PINITDEVICE (ComArea, AreaLen, Language, SpoolFileNum, OutputDev);
163.000     12  1      CHECK_FOR_ERROR;
164.000     13  1
165.000     13  1      {
166.000     **  1      { Now Write to the four fields.
167.000     13  1      { }
168.000     13  1
169.000     13  1      SubFieldNum := -1; {All fields have only 1 subfield}
170.000     14  1
171.000     14  1      FieldName := 'FROM';
172.000     15  1      Data := '      Profit Monitor';
173.000     16  1      DataLen := 20;
174.000     17  1      PWRITEFIELD (ComArea, Fieldname, SubFieldNum, Data, DataLen);
175.000     18  1      CHECK_FOR_ERROR;
176.000     19  1
177.000     19  1      FieldName := 'TO';
178.000     20  1      Data := '      All Regional Managers';
179.000     21  1      DataLen := 27;
180.000     22  1      PWRITEFIELD (ComArea, FieldName, SubFieldNum, Data, DataLen);
181.000     23  1      CHECK_FOR_ERROR;
182.000     24  1
183.000     24  1      FieldName := 'DATE';
184.000     25  1      Data := '      August 1, 1982';
185.000     26  1      DataLen := 24;
186.000     27  1      PWRITEFIELD (ComArea, FieldName, SubFieldNum, Data, DataLen);
187.000     28  1      CHECK_FOR_ERROR;
188.000     29  1
189.000     29  1      FieldName := 'SUBJECT';
190.000     30  1      Data := '      Last Quarter's Sales';
191.000     31  1      DataLen := 34;
192.000     32  1      PWRITEFIELD (ComArea, FieldName, SubFieldNum, Data, DataLen);
193.000     33  1      CHECK_FOR_ERROR;
194.000     34  1
195.000     34  1      {
196.000     **  1      { Now move the pen.
197.000     34  1      { }
198.000     34  1
199.000     34  1      XLoc := 180; {1 Inch}
200.000     35  1      YLoc := 720; {4 Inches}
201.000     36  1      PMOVEPENABS (ComArea, XLoc, YLoc);
202.000     37  1      CHECK_FOR_ERROR;
203.000     38  1
204.000     38  1      {
205.000     **  1      { Switch The Primary Font to Italics.
206.000     38  1      { }
207.000     38  1

```

Jan 84

F-28

```

208.000 38 1 PrimaryFont := 1;
209.000 39 1 SecondaryFont := -1; {Remains unchanged}
210.000 40 1 PUSEFONT (ComArea, PrimaryFont, SecondaryFont);
211.000 41 1 CHECK_FOR_ERROR;
212.000 42 1
213.000 42 1 {
214.000 ** 1 { Now write out some text in the new font.
215.000 42 1 { }
216.000 42 1
217.000 42 1 with TextArray do
218.000 43 2 begin {with}
219.000 43 2 AddrTag := ByteAddr;
220.000 44 2 CharArray := 'See page 2 for last quarter's sales figures.';
221.000 45 2 DataLen := 45;
222.000 46 2 AddrTag := WordAddr;
223.000 47 2 FWRITE (SpoolFileNum, LogArray, -DataLen, 0);
224.000 48 2 if ccode <> CCE then
225.000 49 3 begin {FWRITE error}
226.000 49 3 writeLn(output, 'FWRITE failed.');
```

227.000 50 3 TERMINATE;

228.000 50 3 end; {FWRITE error}

229.000 50 2 end; {with}

230.000 51 1

231.000 51 1 {

232.000 ** 1 { Now, activate logical page 0 (no form) and deactivate logical page

233.000 ** 1 { 1 (letterhead form) and skip to the new page.

234.000 51 1 { }

235.000 51 1

236.000 51 1 LogPageNum := 0;

237.000 52 1 PACTIVATEPAGE (ComArea, LogPageNum);

238.000 53 1 CHECK_FOR_ERROR;

239.000 54 1

240.000 54 1 LogPageNum := 1;

241.000 55 1 PDEACTIVATEPAGE (ComArea, LogPageNum);

242.000 56 1 CHECK_FOR_ERROR;

243.000 57 1

244.000 57 1 PNEWPAGE (ComArea);

245.000 58 1 CHECK_FOR_ERROR;

246.000 59 1

247.000 59 1 {

248.000 ** 1 { Now Convert the figure created with DSG/3000 into a raster image.

249.000 59 1 { }

250.000 59 1

251.000 59 1 FigureFileName := 'QTRSALES ';

252.000 60 1 FigureName := 'SECOND QUARTER ' ; {Terminated with a blank}

253.000 61 1 RasterFileName := 'R2NDQTR ' ; {Terminated with a blank}

254.000 62 1 Units := 1; {Inches}

255.000 63 1 Height := 6.0;

256.000 64 1 Rotation := 90;

257.000 65 1 PCONVERTFIGURE (ComArea, FigureFileName, FigureName, RasterFileName,

258.000 66 1 OutputDev, Height, Units, Rotation);

259.000 66 1 CHECK_FOR_ERROR;

```

260.000      67  1
261.000      67  1      {
262.000      **  1      { Now Load the image as #17, print it at a position relative to the
263.000      **  1      { Upper-left corner of the logical page, and then delete it.
264.000      67  1      {
265.000      67  1
266.000      67  1      RasterNumber := 17;
267.000      68  1      PLOADRASTER (ComArea, RasterFilename, RasterNumber);
268.000      69  1      CHECK_FOR_ERROR;
269.000      70  1
270.000      70  1      XPosition := 1.5;    {Inches to the right}
271.000      71  1      YPosition := 2.0;    {Inches down the page}
272.000      72  1      PositionMode := 0; {Relative}
273.000      73  1      PPRINTRASTER (ComArea, RasterNumber, XPosition, YPosition, Units,
274.000      74  1      PositionMode);
275.000      74  1      CHECK_FOR_ERROR;
276.000      75  1
277.000      75  1      PDELETERASTER (ComArea, RasterNumber);
278.000      76  1      CHECK_FOR_ERROR;
279.000      77  1
280.000      77  1      {
281.000      **  1      { Finally, close the spoolfile.
282.000      77  1      {
283.000      77  1
284.000      77  1      FCLOSE (SpoolFileNum, 0, 0);
285.000      78  1
286.000      77  1      end.    {Quarterly_Report}

NUMBER OF ERRORS = 0      NUMBER OF WARNINGS = 0
PROCESSOR TIME 0: 0:12   ELAPSED TIME 0: 0:26
NUMBER OF LINES = 287    LINES/MINUTE = 1435.0

```

IFS/3000 Error Messages

APPENDIX G

ERROR MESSAGES

THIS PAGE SHOULD BE BLANK

ERROR MESSAGE	MEANING	ACTION
101 Internal error: Expected to write record type <i>nnn</i> , not <i>nnn</i> . (PSPERR 101)	Probably a bad environment file or possibly an internal software error, not under user control.	Write down error message and contact system manager.
102 Internal error: (PSPERR 102) Assertion <i>nnn</i> failed in procedure <i>xxx</i> .	Internal software error, not under user control.	Contact system manager.
104 Internal error: bad orientation request: <i>nnn</i> (PSPERR 104)	Internal software error, not under user control.	Contact system manager.
106 Internal error: bad low ASCII number: <i>nnn</i> (PSPERR 106)	Internal software error, not under user control.	Contact system manager.
107 Internal error: bad high ASCII number: <i>nnn</i> (PSPERR 107)	Internal software error, not under user control.	Contact system manager.
108 Can't write to the environment file. (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information and possible action.
109 Form names must begin with a letter.	A nonalphabetic character begins the form name.	Check the form name.
110 This program doesn't support this terminal, since V/3000 doesn't.	Terminal is not supported by V/3000.	Consult the V/3000 Reference manual (P/N 32209-90001) for supported terminals.
111 This program doesn't support this device, since V/3000 doesn't.	Terminal is not supported by V/3000.	Consult the V/3000 Reference manual (P/N 32209-90001) for supported terminals.

ERROR MESSAGE	MEANING	ACTION
201 This file is not an environment file.	Specified file exists, but has the wrong file code.	Specify an environment file (code=P2680).
203 Internal error: Bad record type: Expected <i>nnn</i> , not <i>nnn</i> . (PSPERR 203)	Probably a bad environment file or possibly an internal software error, not under user control.	Write down error message and contact system manager.
204 Bad env file: bad pointer <i>nnn</i> . (PSPERR 204)	Error in environment file, not under user control.	Write down error message and contact system manager.
206 Bad env file: default meas. sys. bad in phys page. (PSPERR 206)	Error in environment file, not under user control.	Write down error message and contact system manager.
207 Bad env file: # recs inconsistent in free header. (PSPERR 207)	Error in environment file, not under user control.	Write down error message and contact system manager.
208 Bad env file: inconsistent log page # in LP desc. (PSPERR 208)	Error in environment file, not under user control.	Write down error message and contact system manager.
209 Bad env file: bad orientation in log page desc. (PSPERR 209)	Error in environment file, not under user control.	Write down error message and contact system manager.
210 Bad env file: bad units in log page desc. (PSPERR 210)	Error in environment file, not under user control.	Write down error message and contact system manager.
211 Bad env file: expected to find compiled phys page. (PSPERR 210)	Error in environment file, not under user control.	Write down error message and contact system manager.
212 Bad env file: cset # out of range in phys pg desc. (PSPERR 212)	Error in environment file, not under user control.	Write down error message and contact system manager.
216 Bad env file: expected compiled form descriptor. (PSPERR 216)	Error in environment file, not under user control.	Write down error message and contact system manager.

ERROR MESSAGE	MEANING	ACTION
217 Bad env file: expected form number <i>nnn</i> , got <i>nnn</i> . (PSPERR 217)	Error in environment file, not under user control.	Write down error message and contact system manager.
218 Bad env file: wrong name in compiled form descriptor. (PSPERR 218)	Error in environment file, not under user control.	Write down error message and contact system manager.
219 Bad env file: uncompiled form found in log page desc. (PSPERR 219)	Error in environment file, not under user control.	Write down error message and contact system manager.
220 Bad env file: invalid forms part of log pg desc. (PSPERR 220)	Error in environment file, not under user control.	Write down error message and contact system manager.
221 Bad env file: expected multi-copy forms table. (PSPERR 221)	Error in environment file, not under user control.	Write down error message and contact system manager.
222 Bad env file: MC form should already be compiled. (PSPERR 222)	Error in environment file, not under user control.	Write down error message and contact system manager.
223 Bad env file: expected VFC descriptor. (PSPERR 223)	Error in environment file, not under user control.	Write down error message and contact system manager.
224 Bad env file: expected a VFC file name. (PSPERR 224)	Error in environment file, not under user control.	Write down error message and contact system manager.
225 Bad env file: VFC descriptor didn't match table. (PSPERR 225)	Error in environment file, not under user control.	Write down error message and contact system manager.
226 Bad env file: invalid VFC part of log page desc. (PSPERR 226)	Error in environment file, not under user control.	Write down error message and contact system manager.
227 Bad env file: VFC should already have been compiled. (PSPERR 227)	Error in environment file, not under user control.	Write down error message and contact system manager.

ERROR MESSAGE	MEANING	ACTION
228 Bad env file: pointer to end was nil, start wasn't. (PSPERR 228)	Error in environment file, not under user control.	Write down error message and contact system manager.
229 Bad env file: bad orientation in char descriptor. (PSPERR 229)	Error in environment file, not under user control.	Write down error message and contact system manager.
230 Bad env file: expected Y or N in PS in char desc. (PSPERR 230)	Error in environment file, not under user control.	Write down error message and contact system manager.
231 Bad env file: illegal value in log page desc. (PSPERR 231)	Error in environment file, not under user control.	Write down error message and contact system manager.
237 Bad env file: incorrect version number. (PSPERR 237)	Error in environment file, not under user control.	Write down error message and contact system manager.
239 Bad env file: bad orientation in char font desc. (PSPERR 239)	Error in environment file, not under user control.	Write down error message and contact system manager.
241 Bad env file: inconsistent char font table. (PSPERR 241)	Error in environment file, not under user control.	Write down error message and contact system manager.
242 One or more character fonts do not match for the current device.	Environment file converted from one device to another. Character references invalid for new device.	Change the font reference(s) to file(s) that contain fonts for new device.
249 A logical page needs char font "XXX" which isn't in the file.	The base character font of a logical page refers to a character font which is not defined in the environment.	Check logical page definitions or define the character font.
250 A logical page needs a char font, but there are none in the file.	The base character font of a logical page refers to a character font which is not defined in the environment.	Check logical page definitions or define the character font.

ERROR MESSAGE	MEANING	ACTION
251 A logical page needs char font # <i>nnn</i> which isn't in the file.	The base character font of a logical page refers to a character font which is not defined in the environment.	Check logical page definitions or define the character font.
256 This file can be examined, but not changed.	MPE file security violation; user does not have appropriate access capability	Contact your system manager.
257 Bad srce env file: compiled list was inconsistent. (PSPERR 257)	Error in source environment file, not under user control.	Write down error message and contact system manager.
258 Bad srce env file: expected a form table. (PSPERR 258)	Error in source environment file, not under user control.	Write down error message and contact system manager.
259 Bad dest env file: expected a form table. (PSPERR 259)	Error in destination environment file, not under user control.	Write down error message and contact system manager.
260 Can't copy any more forms. Maximum is 32. (PSPERR 260)	Too many forms; each size and orientation of any given form counts as a form.	Check forms in destination file. Recompiling destination file eliminates any deleted forms. Then try the copy again.
261 Bad srce env file: expected a VFC table. (PSPERR 261)	Error in source environment file, not under user control.	Write down error message and contact system manager.
262 Bad dest env file: expected a VFC table. (PSPERR 262)	Error in destination environment file, not under user control.	Write down error message and contact system manager.
263 Can't copy any more VFCs. Maximum is 32. (PSPERR 263)	Too many VFCs.	Recompile destination file to eliminate any deleted VFCs. Then try the copy again.

ERROR MESSAGE	MEANING	ACTION
265 Can't continue compiling. (No existing compiled font to use.)	Could not access the specified character font.	Identify the problem character font, specify a valid character font, then recompile.
267 Can't continue compiling. (No existing compiled form to use.)	Could not access the specified form.	Identify the problem form, specify a valid form, then recompile.
269 Can't continue compiling. (No existing compiled VFC to use.)	Could not access the specified VFC.	Identify the problem VFC, specify a valid VFC, then recompile.
270 Warning: Required ComArea length is too big. (PSPERR 270)	Cannot use the environment with the intrinsics.	Reduce the size of the environment file. (By deleting forms, character fonts, etc.)
271 Error opening form file "xxx" . . .	First line of a 2 line message.	
272 Error accessing form "xxx" . . .	First line of a 2 line message.	
273 Out of space in env. file - copy to a bigger file. (PSPERR 273)	File reached its physical end of file.	Create a larger environment file using the BUILD command and copy existing environment into it using FCOPY.
280 Warning: there are no active logical pages in this environment.	There are logical pages defined, but none of them are active.	Call PACTIVATEPAGE before printing any data or moving the pen.
281 Can't use primary font, since "xxx" (in phys. page) is undefined).	Primary font name specified in PHYSICAL PAGE menu is not defined in this environment.	Define the character font or change the primary font name.

ERROR MESSAGE	MEANING	ACTION
282 Can't use primary font, since # <i>nnn</i> (in phys. page) is undefined.	Primary font number specified in PHYSICAL PAGE menu is not defined in this environment.	Define the character font or change the primary font number.
403 Internal error: no char. font supplied to delete. (PSPERR 403)	Internal software error, not under user control.	Contact system manager.
408 Internal error: no log. page supplied to delete. (PSPERR 408)	Internal software error, not under user control.	Contact system manager.
409 IDSCHAR file " <i>xxx</i> " had wrong version #: <i>nnn</i> *	Probably means the character font file has been damaged.	Contact system manager.
414 Bad char font file: the storage fmt wasn't dot/bit. (PSPERR 414)	Probably means the character font file has been damaged.	Contact system manager.
420 Bad env file: ran out of records to free. (PSPERR 420)	Error in environment file, not under user control.	Write down error message and contact system manager.
421 Bad env file: list to free was inconsistent. (PSPERR 421)	Error in environment file, not under user control.	Write down error message and contact system manager.
428 Char font # <i>nnn</i> in phys page has not been defined. Can't compile.	Primary or secondary character font specified in the PHYSICAL PAGE menu is not defined in the environment.	Define the character font number or change the primary or secondary font.
429 Font " <i>xxx</i> " in phys page is undefined. Can't compile.	Primary or secondary character font specified in the PHYSICAL PAGE menu is not defined in the environment.	Define the character font name or change the primary or secondary font.
*For Error Numbers 410-413 see page G-42		

ERROR MESSAGE	MEANING	ACTION
431 Internal error: bad parms to ADD'TO'DELETE'LIST. (PSPERR 431)	Internal software error, not under user control.	Contact system manager.
441 Copy would result in more than 32 character fonts -- no copy done.	Too many character fonts in the environment.	Recompile destination file to eliminate deleted fonts, if any.
442 <i>Nnn</i> char fonts to copy, but current env only has room for <i>nnn</i> more.	Too many character fonts in the environment.	Recompile destination file to eliminate deleted fonts, if any.
448 Copy would result in more than 32 logical pages -- no copy done.	Too many logical pages in the environment.	Recompile destination file to eliminate deleted logical pages, if any.
450 <i>Nnn</i> logical pages to copy, but current env only has room for <i>nnn</i> more.	Too many logical pages in the environment.	Recompile destination file to eliminate deleted logical pages, if any.
452 Couldn't open HP defined env file . . . xxx	First line of a 2 line message. MPE file system error follows.	Contact system manager.
453 Couldn't close HP defined env file . . . xxx	First line of a 2 line message. MPE file system error follows.	Contact system manager.
455 File code error while accessing form file.	Form file does not have valid file code.	Specify a file with file code of PFORM.
456 Internal error: Tried to add a duplicate key to form file.	Internal software error, not under user control.	Contact system manager.
457 Internal error: # <i>nnn</i> occurred while accessing form file.	Internal software error, not under user control.	Contact system manager.

ERROR MESSAGE	MEANING	ACTION
458 Internal error: Tried to open new form file with existing name.	Internal software error, not under user control.	Contact system manager.
459 Internal error: Record not found in form file.	Internal software error, not under user control.	Contact system manager.
460 Internal error: not enough space form directory in form file open.	Internal software error, not under user control.	Contact system manager.
461 This file is not an IDS/FORM file.	Form file is an incorrect type of file.	Check form file name. If valid, contact system manager.
462 The form file is at EOF.	Form file has been damaged.	Contact system manager.
463 File name supplied is not an IDIFORM file. (Wrong file code.)	Form file specified does not have valid file code.	Specify a file with file code of PFORM.
464 Form file error: bad units found in form header. (PSPERR 464)	Bad IDS/FORM file.	Contact system manager.
465 Internal error: KKA count returned didn't match request.	Internal software error, not under user control.	Contact system manager.
466 No file name available for form "xxx".	Form was specified, but not the forms file name and there was no default forms file name in PHYSICAL PAGE AND ENVIRONMENT DEFAULTS menu.	Specify the forms file name.

ERROR MESSAGE	MEANING	ACTION
467 There is no default form file for this environment.	Form was specified, but not the forms file name and there was no default forms file name in PHYSICAL PAGE AND ENVIRONMENT DEFAULTS menu.	Specify the forms file name.
470 Too many different forms in env. Can't compile. (PSPERR 470)	More than 32 forms; each size and orientation of any given form counts as a form.	Delete forms until there are only 32 forms.
478 Can't open VFC file "xxx". (FSERR nnn)	File system error occurred.	Refer to the file system error number for more information.
479 Can't close VFC file. (FSERR nnn)	File system error occurred.	Refer to the file system error number for more information.
480 "Xxx " not a VFC file: not ASCII. (PSPERR 480)	File specified as the VFC file is a binary file; VFC file must be ASCII.	Specify a valid VFC file.
481 Too many VFCs. Can't compile. (PSPERR 481)	More than 32 VFCs, not under user control.	Contact system manager.
482 Bad "VFC" line in file "xxx". (PSPERR 482)	Record beginning with "VFC" is in error.	Check VFC files for valid format.
483 Can't read from VFC file "xxx". (FSERR nnn)	FWRTEDIR failed.	Refer to the file system error number for more information.
484 "Xxx " not a VFC file: bad 1st line. (PSPERR 484)	First record was not in VFC file format.	Check VFC file for valid format.

ERROR MESSAGE	MEANING	ACTION
485 No "VFC" line in file "xxx". (PSPERR 485)	Missing record beginning with "VFC" in the file specified for VFC.	Check VFC file for valid format.
486 This is not an IDSCHAR cell file.	Cell file specified has invalid file code, should be PCHAR.	Verify the name of the cell file specified.
490 Can't open IDSCHAR file "xxx". (FSERR !)	FWRITEDIR failed.	Refer to the file system error number for more information.
491 File "xxx" is not an IDSCHAR file.	Cell file specified has invalid file code, should be PCHAR.	Verify the name of the cell file specified.
492 Can't read from file "xxx".	Second line of a 2 line message. First line displayed file system error.	Refer to the file system error number for more information.
493 Can't find right size in "xxx". (PSPERR 493)	Character font file has been changed since you defined it, actual size is not in the file.	Use CHARACTER FONT menu to define an actual size which exists in the file.
494 Can't read cell versions from file "xxx".	Second line of a 2 line message. First line displayed file system error.	Refer to the file system error number for more information.
495 Bad IDSCHAR file -- ran out of dot/bit. (PSPERR 495)	Probably means the character font file has been damaged.	Contact system manager.
496 Can't close "xxx". (FSERR !)	FWRITEDIR failed.	Refer to the file system error number for more information.

ERROR MESSAGE	MEANING	ACTION
801 Internal error: bad parameters to COMPILE'FORM. (PSPERR 801)	Internal software error, not under user control.	Contact system manager.
802 Bad form file: form head missing for form "nnn". (PSPERR 802)	System is unable to process form file. Form file has been damaged and must be restored.	Contact system manager to restore file.
803 Bad form file: form head bad in form "nnn". (PSPERR 803)	System is unable to process form file. Form file has been damaged and must be restored.	Contact system manager to restore file.
804 Warning: scan line nnn. Too many triplets. (PSPERR 804)	Form is too complicated; too many graphics (line, box, shading, etc.) and characters or logos on a given line.	Simplify the form.
806 Can't do FPOINT on triplet temp file. (FSERR nnn) (PSPERR 806)	FWRITEDIR failed.	Refer to the file system error number for more information
807 Can't sort the triplet file. SORTINITIAL failed. (PSPERR 807)	Error in SORT/3000 processing.	Contact system manager.
808 Can't sort the triplet file. SORTOUTPUT failed. (PSPERR 808)	Error in SORT/3000 processing.	Contact system manager.
810 Can't write to dot/bit temp file. (FSERR !) (PSPERR 810)	FWRITEDIR failed.	Refer to the file system error number for more information
811 Can't read from dot/bit temp file. (FSERR !) (PSPERR 811)	FWRITEDIR failed.	Refer to the file system error number for more information

ERROR MESSAGE	MEANING	ACTION
812 Bad form file: found a diagonal line. (PSPERR 812)	System is unable to process form file. Form file has been damaged and must be restored.	Contact system manager to restore file.
813 Warning: the form had to be clipped.	Text or logo extends beyond the form boundaries.	Check placement of text or logo.
815 Bad form file: found a duplicate form head. (PSPERR 815)	System is unable to process form file. Form file has been damaged and must be restored.	Contact system manager to restore file.
816 Bad form file: found illegal element type: <i>nnn</i> . (PSPERR 816)	System is unable to process form file. Form file has been damaged and must be restored.	Contact system manager to restore file.
817 Bad form file: bad element length. (PSPERR 817)	System is unable to process form file. Form file has been damaged and must be restored.	Contact system manager to restore file.
818 Internal error: can't find form "xxx". (PSPERR 818)	Internal software error, not under user control.	Contact system manager.
819 Bad form file: hit EOR before EOR element found. (PSPERR 819)	System is unable to process form file. Form file has been damaged and must be restored.	Contact system manager to restore file.
820 Can't open the triplet temp file. (FSERR <i>nnn</i>) (PSPERR 820)	FWRITEDIR failed.	Refer to the file system error number for more information.
821 Can't open the dot/bit temp file. (FSERR <i>nnn</i>) (PSPERR 821)	FWRITEDIR failed.	Refer to the file system error number for more information.

ERROR MESSAGE	MEANING	ACTION
822 Bad form file: bad thickness value in line element. (PSPERR 822)	System is unable to process form file. Form file has been damaged and must be restored.	Contact system manager to restore file.
823 Bad form file: can't find subform. (PSPERR 823)	System is unable to process form file. Form file has been damaged and must be restored.	Contact system manager to restore file.
824 Internal error: can't switch to subform. (PSPERR 824)	Internal software error, not under user control.	Contact system manager.
825 Internal error: can't switch back from subform. (PSPERR 825)	Internal software error, not under user control.	Contact system manager.
826 Bad form file: a subform contained another subform. (PSPERR 826)	System is unable to process form file. Form file has been damaged and must be restored.	Contact system manager to restore file.
827 Bad form file: bad line type <i>nnn</i> in line element. (PSPERR 827)	System is unable to process form file. Form file has been damaged and must be restored.	Contact system manager to restore file.
828 Can't write to triplets temp file. (FSERR <i>nnn</i>) (PSPERR 828)	FWRITEDIR failed.	Refer to the file system error number for more information.
829 Bad form file: bad shading percentage <i>nnn</i> . (PSPERR 829)	System is unable to process form file. Form file has been damaged and must be restored.	Contact system manager to restore file.

ERROR MESSAGE	MEANING	ACTION
830 Bad form file: corners of a box were out of order. (PSPERR 830)	System is unable to process form file. Form file has been damaged and must be restored.	Contact system manager to restore file.
831 Warning: missing character(s) or logo(s). Black rectangle used.	In a field heading or logo, tried to print a character that is not defined in the cell file.	Check the field heading text or the character font or logo file.
852 Bad form file: bad character orientation. (PSPERR 852)	System is unable to process form file. Form file has been damaged and must be restored.	Contact system manager to restore file.
853 Bad form file: bad text direction. (PSPERR 853)	System is unable to process form file. Form file has been damaged and must be restored.	Contact system manager to restore file.
854 Couldn't close cell file "xxx". (PSPERR 854)	Second line of a 2 line message, first line displayed file system error.	Refer to MPE file system error.
855 Couldn't open cell file "xxx". (PSPERR 855)	Second line of a 2 line message, first line displayed file system error.	Refer to MPE file system error.
856 Warning: Expected logo file "xxx" not font.	Specified a logo file in the FIELD FONT menu but file is a character font file. Form may not print as expected.	Change the specification or the file name.
857 Warning: Expected font file "xxx" not logo.	Did not specify a logo file in the FIELD FONT menu but file is a logo file. Form may not print as expected.	Change the specification or the file name.
859 Error storing dot/bit. (PSPERR 859)	Second line of a 2 line message, first line displayed file system error.	Refer to MPE file system error.

ERROR MESSAGE	MEANING	ACTION
860 No font sizes in cell file "xxx". (PSPERR 860)	Cell file is empty.	Check the specified cell file.
861 Form file containing form "xxx" is bad. (PSPERR 861)	System is unable to process form file. Form file has been damaged and must be restored.	Contact system manager to restore file.
862 Error getting XDS. GETDSEG returned <i>nnn</i> . (PSPERR 862)	GETDSEG failed.	Refer to the error number.
863 Error altering size of XDS, possible internal error. (PSPERR 863)	Internal software error, not under user control.	Contact system manager.
864 Error moving character description from XDS. (PSPERR 864)	Internal software error, not under user control.	Contact system manager.
865 Error expanding XDS. (PSPERR 865)	Internal software error, not under user control.	Contact system manager.
866 Error moving character description to XDS. (PSPERR 866)	Internal software error, not under user control.	Contact system manager.
867 Form file containing form "xxx" is bad. (PSPERR 867)	System is unable to process form file. Form file has been damaged and must be restored.	Contact system manager to restore file.
868 Bad cell file "xxx": bad units. (PSPERR 868)	Cell file has probably been damaged.	Contact system manager.
869 Couldn't free the extra data segment. (PSPERR 869)	Internal software error, not under user control.	Contact system manager.

ERROR MESSAGE	MEANING	ACTION
870 Configured extra data seg size too small. Need (PSPERR 870)	Internal software error, not under user control.	Contact system manager.
871 Bad baseline in cell file. "xxx". (PSPERR 871)	Cell file has probably been damaged.	Contact system manager.
872 Warning: a field has an unspecified character font.	Cell file name was not specified in the FIELD FONT menu.	Check the cell file name.
873 ALTDSEG error while checking what size dseg we got. (PSPERR 873)	Internal software error, not under user control.	Contact system manager.
874 Bad cell file: max P.S. bound less than min. (PSPERR 874)	Cell file has probably been damaged.	Contact system manager.
875 Error accessing cell file "xxx". (PSPERR 875)	Second line of a 2 line message, first line displayed file system error.	Refer to MPE file system error.
900 Couldn't close the environment file. (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information and possible action.
901 Not enough room in the PSP Com Area. Need <i>nnn</i> words.	User program specified insufficient space.	Check the ComArea length parameter in PINITIALIZE and PINITDEVICE.
902 Couldn't open the environment file. (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information and possible action.
903 File wasn't an environment file. Wrong file code.	Environment file associated with output print file is in error.	Check the FILE command to determine problem file.

ERROR MESSAGE	MEANING	ACTION
904 File wasn't an environment file. Wrong record size.	Environment file associated with output print file is in error.	Check the FILE command to determine problem file.
905 Couldn't read the environment file's header. (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information and possible action.
908 Bad environment file.	Environment file has been damaged.	Contact system manager.
909 FGETINFO failed on the environment file. (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information and possible action.
910 Bad env file: file header had wrong type.	Environment file has been damaged.	Contact system manager.
911 Bad env file: illegal pointer in file header.	Environment file has been damaged.	Contact system manager.
912 Couldn't read the char font table. (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information.
913 Couldn't read a char font descriptor. (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information.
914 Bad env file: char font table had wrong type.	Environment file has been damaged.	Contact system manager.
915 Bad env file: a char font descriptor had wrong type.	Environment file has been damaged.	Contact system manager.

ERROR MESSAGE	MEANING	ACTION
916 Couldn't read the logical page table. (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information
917 Bad env file: the logical page table had the wrong type.	Environment file has been damaged.	Contact system manager.
918 Couldn't read a logical page descriptor. (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information
919 Bad env file: a logical page descriptor had the wrong type.	Environment file has been damaged.	Contact system manager.
920 Couldn't read compiled log page rec. (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information
921 Bad env file: compiled log page rec had wrong type.	Environment file has been damaged.	Contact system manager.
922 Couldn't read the physical page descriptor. (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information
923 Bad env file: the physical page descriptor had the wrong type.	Environment file has been damaged.	Contact system manager.
924 Couldn't read the compiled physical page. (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information
925 Bad env file: the compiled physical page had the wrong type.	Environment file has been damaged.	Contact system manager.

ERROR MESSAGE	MEANING	ACTION
926 Bad env file: bad pointer in phys page desc.	Environment file has been damaged.	Contact system manager.
927 Bad env file: nil pointer to compiled phys page in PP desc.	Environment file has been damaged.	Contact systems manager
928 Couldn't read the multi-copy forms descriptor. (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information.
929 Bad env files: MC forms descriptor had wrong type.	Environment file has been damaged.	Contact system manager.
930 Couldn't read a single-copy forms descriptor. (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information.
931 Bad env file: Single copy forms descriptor had wrong type.	Environment file has been damaged.	Contact system manager.
932 Ran out of space while building the PSP Com Area.	Internal software error, not under user control.	Contact system manager.
933 Couldn't read the form table. (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information.
934 Bad env file: the form table had the wrong type.	Environment file has been damaged.	Contact system manager.

ERROR MESSAGE	MEANING	ACTION
935 Couldn't read a compiled form descriptor. (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information.
936 Bad env file: a compiled form descriptor had the wrong type.	Environment file has been damaged.	Contact system manager
937 Bad env file: bad symbol table.	Environment file has been damaged.	Contact system manager.
938 Can't read symbol table record. (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information.
939 Bad env file: a symbol table record had the wrong type.	Environment file has been damaged.	Contact system manager.
940 Logical page orientation in ComArea was bad.	ComArea has been damaged.	Check your program.
941 No current form -- can't write to fields.	Called PWRITEFIELD on a logical page that has no form.	Check your program.
942 The current logical page has no entry in the PSPComArea.	ComArea has been damaged.	Check your program.
943 The logical page entry for the current LP is bad.	ComArea has been damaged.	Check your program.
944 Can't move the pen. (FDEVICECONTROL error <i>nnn</i>)	FDEVICECONTROL failed.	Refer to FDEVICECONTROL error number.

ERROR MESSAGE	MEANING	ACTION
945 Can't write to field. (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information.
946 Form "xxx " is not on the current logical page.	PNEWFORM was called using a form that is not on the logical page. (The current logical page is not the MULTICOPY FORMS logical page.)	Check your program.
947 Form "xxx " isn't on the current log. page or the multi-copy table.	PNEWFORM was called using a form that is neither on the logical page nor in the MULTICOPY FORMS TABLE.	Check your program.
948 Bad character font entry.	ComArea has been damaged.	Check your program.
949 Couldn't find field "xxx ".	Called PWRITEFIELD using a field which is not in the current Subform. If no current subform, then field is not in any subform.	Check your program.
950 There are no active logical pages.	All pages have been deactivated or no initially active pages were defined in the environment.	Activate a logical page.
951 Can't go to the next page. (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information.
952 <i>Nnn</i> is an illegal logical page number. Should be 0 to 31 or -1.	An intrinsic was called using an invalid logical page number.	Check your program.

ERROR MESSAGE	MEANING	ACTION
953 Can't activate logical page. (FDEVICECONTROL error <i>nnn</i>)	FDEVICECONTROL failed.	Refer to FDEVICECONTROL error number.
954 Can't deactivate logical page. (FDEVICECONTROL error <i>nnn</i>)	FDEVICECONTROL failed.	Refer to FDEVICECONTROL error number.
955 Can't select character fonts. (FDEVICECONTROL error <i>nnn</i>)	FDEVICECONTROL failed.	Refer to FDEVICECONTROL error number.
956 A font number of <i>nnn</i> is illegal. Must be 0-31 or -1.	An intrinsic was called using an invalid character font number.	Check your program.
957 The specified primary font (<i>nnn</i>) is not in the environment.	An intrinsic was called using a primary font number which is not defined.	Check your program.
958 The specified secondary font (<i>nnn</i>) is not in the environment.	An intrinsic was called using a secondary character number font which is not defined.	Check your program.
959 Warning: line too long – truncated.	A given line of data exceeded PWRITEFIELD's buffer size (250 characters).	Check your program.
960 <i>Nnn</i> isn't a legal subfield number. Should be greater than 0, or -1.	An intrinsic was called using an invalid subfield number.	Check your program.
961 Subform "xxx" is not in the current form.	PNEWSUBFORM was called using a subform name that is not defined in the current form.	Check your program.

ERROR MESSAGE	MEANING	ACTION
962 The string is too long – its length would be > 32767 dots.	PSTRINGWIDTH intrinsic was called using too large a value in the string length parameter.	Check your program
963 Bad current character font number in the ComArea.	ComArea has been damaged.	Check your program.
964 Character font number <i>nnn</i> is not in the environment.	An intrinsic was called using an undefined character font number.	Check your program.
965 Character font “xxx ” is not in the environment.	An intrinsic was called using an undefined character font.	Check your program.
966 The primary character font number is not in the environment.	An intrinsic was called using a primary font number which is not defined.	Check your program.
967 The secondary character font number is not in the environment.	An intrinsic was called using a secondary character font number which is not defined.	Check your program.
968 Bad current logical page number in the ComArea.	ComArea has been damaged.	Check your program.
969 Logical page number <i>nnn</i> is not in the environment.	An intrinsic was called using an undefined logical page number.	Check your program.
970 Invalid language parameter	PINITIALIZE or PINITDEVICE was called with an invalid value in the language parameter.	Check your program.
971 <i>Nnn</i> isn't a legal subfield for this field. Should be 1 or -1.	PWRITEFIELD was called using an invalid subfield number.	Check your program.

ERROR MESSAGE	MEANING	ACTION
972 Can't get environment file name (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information and possible action.
973 The file supplied apparently had no environment file.	FFILEINFO indicated there was no environment file associated with output print file.	Check file equation or FOPEN.
974 PERRMSG was called when error had occurred.		Check your program.
975 PNEWFORM was called with a blank form name.		Check your program.
976 Bad form name "xxx". (Must begin with a letter, or be "1" or "2")	PNEWFORM was called using an invalid form name.	Check your program.
977 A -1 was used as the subfield number in a field with <i>nnn</i> subfields.	Value of -1 for subfield number is valid only when there is one subfield.	Check your program.
978 Can't go to the next physical page. (FSERR <i>nnn</i>)	FWRITEDIR failed.	Refer to the file system error number for more information.
979 <i>Nnn</i> isn't a legal subfield for this field. Should be 1 to <i>nnn</i> , or -1.	PWRITEFIELD was called using an invalid number in the subfield parameter.	Check your program.
980 Warning: logical page <i>nnn</i> was already active.	PACTIVATEPAGE was called using the page number of an active page.	Check your program.

ERROR MESSAGE	MEANING	ACTION
981 Warning: logical page <i>nnn</i> was already inactive.	PDEACTIVATEPAGE was called using the page number of an inactive page.	Check your program.
982 Warning: there are no initially active pages in the environment.	No logical pages were defined as initially active.	Activate a logical page.
983 There is no currently active logical page to go to.	PNEWPAGE or PNEWPHYSPAGE was called when there were no active logical pages.	Activate a logical page.
984 There is no current logical page.	No logical pages were defined as initially active in the environment.	Activate a logical page.
985 -1 was used for a logical page #, but there is no current page.	The value -1 used in the logical page number parameter in PLOGPAGEINFO means to use the current page, but there is none.	Check your program.
986 Deactivating the default logical page is not allowed.	PDEACTIVATEPAGE was called using the default page number.	Check your program.
987 Warning: all logical pages have been deactivated.	There must always be one active logical page. Activate a logical page before moving off the last deactivated logical page.	Check your program.
988 Not enough space to even begin building the PSP ComArea.	Need at least 158 words in the ComArea to determine the actual space needed for ComArea. User program specified insufficient space.	Check the ComArea length parameter in PINITIALIZE or PINITDEVICE.

ERROR MESSAGE	MEANING	ACTION
989 Not enough space to even begin building the PSP ComArea.	Need at least 158 words in the ComArea to determine the actual space needed for ComArea. User program specified insufficient space.	Check the ComArea length parameter in PINITIALIZE or PINITDEVICE.
990 Not enough space to even begin building the PSP ComArea.	Need at least 158 words in the ComArea to determine the actual space needed for ComArea. User program specified insufficient space.	Check the ComArea length parameter in PINITIALIZE or PINITDEVICE.
991 Not enough space to even begin building the PSP ComArea.	Need at least 158 words in the ComArea to determine the actual space needed for ComArea. User program specified insufficient space.	Check the ComArea length parameter in PINITIALIZE or PINITDEVICE.
992 Illegal character font name "xxx". (Must begin with a letter)	PFONTNUM was called using an invalid character font name.	Check your program.
993 Bad ComArea — probably hasn't been initialized yet.	PINITIALIZE or PINITDEVICE must be called once before calling any other intrinsic.	Check your program.
994 (<i>Nnn, nnn</i>) is illegal. Can't have negative absolute coordinates.	PMOVEPENABS was called using a negative number in either the x or y coordinate parameters or both.	Check your program.

ERROR MESSAGE	MEANING	ACTION
995 X coordinate <i>nnn</i> is too big. Logical page width is <i>nnn</i> .	PMOVEPENABS or PMOVEPENREL was called using a value in the x-coordinate parameter which exceeds the logical page width.	Check your program.
996 Y coordinate <i>nnn</i> is too big. Logical page height is <i>nnn</i> .	PMOVEPENABS or PMOVEPENREL was called using a value in the y-coordinate parameter which exceeds the logical page height.	Check your program.

ERROR MESSAGE	MEANING	ACTION
997 This intrinsic is not applicable to the initialized device xxx.	The intrinsic being called cannot be used with the output device specified.	Check your program.
998 This intrinsic is not useable without an opened spoolfile.	Only the PCONVERTFIGURE and PFIGUREINFO intrinsics can be used without an open spoolfile.	Check your program. Spoolfile should not be set to -1 in a call to PINITDEVICE.
1000 Image number nn is out of range; must be between 0 and 31 inclusive.	The raster image number must be between 0 and 31 inclusive.	Check your program.
1001 Can't open raster file xxx (FSERR nnn).	Unable to access specified file.	Refer to the file system error number for more information.
1002 FGETINFO failed; can't read the file label (FSERR nnn).	Internal software error; not under user control.	Note the file system error number and contact system manager.
1003 The file is not a raster file; invalid file code nnn.	The specified file has a code that is not a raster file code.	Check your file. The raster file code is 1114 or RASTR.
1004 The file is not a raster file; invalid record size.	The specified file has the wrong record size to be a raster file.	Check your file. Raster file record size is 512 words.
1005 No raster file name was specified.	The raster file name parameter is all blanks.	Check your program.
1006 Can't close raster file xxx (FSERR nnn).	File System error occurred in attempt to close raster file.	Refer to the file system error number for more information.
1007 An error occurred while reading the file header (FSERR nnn).	Internal software error; not under user control.	Note file system error number and contact system manager.

ERROR MESSAGE	MEANING	ACTION
1008 Reached EOF while trying to read the file header (FSERR nnn).	Error in raster file; probably bad number of file header records.	Recreate the raster file.
1009 DMOVIN failed; bounds violation error.	Internal software error; not under user control.	Contact system manager.
1010 DMOVOUT failed; bounds violation error.	Internal software error; not under user control.	Contact system manager.
1011 XDS access failed; invalid Figure Load Table index.	Internal software error; not under user control.	Contact system manager.
1012 The raster file was converted for a device other than that initialized.	Output device specified for initialization must be the same as that specified for conversion.	Check your program.
1013 The partitioned raster file contains no raster data.	Error in raster file.	Create a raster file that contains data.
1014 An error occurred while trying to read the raster file (FSERR nnn).	Internal software error; not under user control.	Note file system error number and contact system manager.
1015 Encountered FDEVICECONTROL error nnn while trying to load image nn.	Internal software error; not under user control.	Contact system manager.
1016 xxx is not a device supported by IFS/3000.	The specified output device cannot be used with the IFS/3000 intrinsics.	Check your program or contact system manager.
1017 The header record count does not match the number found.	Error in raster file.	Recreate the raster file or create a new raster file.

ERROR MESSAGE	MEANING	ACTION
1018 Internal problem; downloaded image count nnn is invalid.	Internal software error or corrupted COMAREA.	Check your COMAREA or contact system manager.
1019 Warning: New image replaces previously loaded image nn.	Image nn replaces the image previously assigned number nn.	None – warning only.
1020 Encountered FDEVICECONTROL error nn while trying to delete image nn.	Internal software error; not under user control.	Contact system manager.
1021 Image nn has not been loaded.	An image cannot be printed or deleted before it has been loaded.	Check your program.
1022 Bad data word count nn found in raster file.	Error in raster file.	Recreate the raster file.
1023 Invalid units parameter n; must be 0, 1, 2 or 3.	Units must be 0 for dots, 1 for inches, 2 for cm., or 3 for mm.	Check your program.
1024 Invalid position mode parameter n; must be 0 or 1.	Position mode must be 0 for relative or 1 for absolute positioning.	Check your program.
1025 Encountered FDEVICECONTROL error nn while trying to print image nn.	Internal software error; not under user control.	Contact system manager.
1026 Relative position (n, n in dots) would be off the logical page.	X or Y position is too large or negative and too large.	Check your program and logical page dimensions.
1027 Absolute position (n, n in dots) would be off the logical page.	X or Y position is too large.	Check your program and logical page dimensions.
1028 X-position does not reduce to a value in the range -32768 . . 32767.	X position is too large or too small.	Check your program.

ERROR MESSAGE	MEANING	ACTION
1029 Y-position does not reduce to a value in the range -32768 . . 32767.	Y position is too large or too small.	Check your program.
1030 Can't specify a negative absolute X-position (nnn in dots).	Only positive values are allowed with absolute positioning.	Check your program.
1031 Can't specify a negative absolute Y-position (nnn in dots).	Only positive values can be used with absolute positioning.	Check your program.
1032 Invalid device xxx for figure file conversion.	Output device specified cannot accept a figure file for conversion.	Check your program.
1033 Invalid raster image rotation n; must be 0, 90, 180 or 270.	Only 0, 90, 180 and 270 are valid raster image rotations.	Check your program.
1034 AGLERR — the device xxx is not supported.	Internal problem; not under user control.	Contact system manager.
1035 Not enough space to create AGLComArea; requested nnn.	Internal software error; not under user control.	Contact system manager.
1039 Can't open temporary file xxx (FSERR nnn).	Specified file has not been opened.	Refer to the file system error number for more information.
1040 Can't rename temporary file xxx (FSERR nnn).	Specified file has not been renamed.	Refer to the file system error number for more information.
1041 Can't close temporary file xxx (FSERR nnn).	Specified file has not been closed.	Refer to the file system error number for more information.
1042 Can't FCLOSE renamed temporary file; failed on re-rename (FSERR nnn).	File has not been renamed; attempt to rename back to original name also failed.	Refer to the file system error number for more information.

ERROR MESSAGE	MEANING	ACTION
1043 Can't FCLOSE renamed temporary file (FSERR nnn).	File has been renamed, but attempt to close failed.	Refer to the file system error number for more information.
1044 Internal problem; no Device Driver matches xxx in case selection.	Internal software error; not under user control.	Contact system manager.
1045 Internal problem; no Units matches xxx in case selection.	Internal software error; not under user control.	Contact system manager.
1046 Encountered FDEVICECONTROL error nnn while printing temporary image.	Internal software error; not under user control.	Contact system manager.
1047 Image height (nnn in MM) is larger than the device will allow.	Image has not been printed.	Change image height parameter to a smaller value.
1048 Image height (nnn in MM) cannot be 0 or negative.	Image has not been printed.	Change image height parameter to a larger positive value.
1049 The file header crash word is set to true.	Error in raster file.	Recreate raster file.
1050 Image width (nnn in MM) larger than the device will allow.	Image height is too large, resulting in width also being too large.	Change image height parameter to a smaller value.
1051 Figure height read from figure file is 0; no image to convert.	Error in figure file.	Recreate figure file.
1052 Figure height read from figure file is <0; no image to convert.	Error in figure file.	Recreate figure file.
1053 Figure width read from figure file is <0; no image to convert.	Error in figure file.	Recreate figure file.

ERROR MESSAGE	MEANING	ACTION
1054 Invalid raster image type n; must be 0 or 1.	Raster image type must be 0 for temporary or 1 for permanent.	Check your program.
1055 Bad Image Height; can't reduce to a value in the range -32768 . . 32767.	Image height is too large.	Check your program.
1056 Internal error; PFIGUREINFO item nn not applicable to raster file.	Internal software error; not under user control.	Contact system manager.
1057 GETDSEG error. Returned index of n; invalid length.	Internal software error; not under user control.	Contact system manager.
1058 GETDSEG error. Returned index of n; too many XDSs this process.	Internal software error; not under user control.	Contact system manager.
1059 GETDSEG error. Returned index of nnn; not enough storage for XDS.	Internal software error; not under user control.	Contact system manager.
1060 GETDSEG error. Returned index of nnn; stack frozen — can't expand.	Internal software error; not under user control.	Contact system manager.
1061 GETDSEG error. GETDSEG returned index of nnn.	Internal software error; not under user control.	Contact system manager.
1062 Can't rename temporary file to xxx (FSERR nnn).	Specified file has not been renamed.	Refer to the file system error number for more information.
1063 xxx already exists as a permanent file.	Specified file has already been created.	Purge file or use another file name.
1064 xxx already exists as a temporary file.	Specified file has already been created.	Purge temporary file, save and rename to permanent file, or use another file name.

ERROR MESSAGE	MEANING	ACTION
1065 Invalid file type n; must be -1, 0 or 1.	File type must be -1 for unknown, 0 for raster, 1 for figure.	Check your program.
1066 Figure width read from figure file is 0; no image to convert.	Error in figure file.	Recreate figure file.
1067 Real number in xxx too small to represent.	ASCII string contains real number that is too small.	Check your program.
1068 Invalid char and real number too small to represent in xxx.	ASCII string contains invalid character and real number that is too small.	Check your program.
1069 Real number in xxx too large to represent.	ASCII string contains real number that is too large.	Check your program.
1070 Invalid char and real number too large to represent in xxx.	ASCII string contains invalid character and real number that is too large.	Check your program.
1071 Invalid character in xxx to convert to a real number.	ASCII string contains invalid character.	Check your program.
1072 No number found in xxx to convert to a real number.	ASCII string must contain a real number.	Check your program.
1073 Error while attempting to convert xxx to a real number.	Internal software error; not under user control.	Contact system manager.
1074 Cannot convert real number to a string; result returned is xxx.	Real number to be converted is probably too large.	Check your program.

ERROR MESSAGE	MEANING	ACTION
1100 AGLERR 501. XMin >=XMax or YMin >=YMax.	Internal software error; not under user control.	Contact system manager.
1101 AGLERR 508. One or more viewport corners outside the NDC space.	Internal software error; not under user control.	Contact system manager.
1102 AGLERR 510. NDC position is outside the current viewport.	Internal software error; not under user control.	Contact system manager.
1103 AGLERR 512. WC position is outside the current viewport.	Internal software error; not under user control.	Contact system manager.
1104 AGLERR 705. The view surface is already initialized.	Internal software error; not under user control.	Contact system manager.
1105 AGLERR 706. No output device is associated with the view surface.	Internal software error; not under user control.	Contact system manager.
1106 AGLERR 708. The specified surface has not been initialized.	Internal software error; not under user control.	Contact system manager.
1107 AGLERR 717. The graphics package has not been initialized.	Internal software error; not under user control.	Contact system manager.
1108 AGLERR 718. The function is not supported by the device.	Internal software error; not under user control.	Contact system manager.
1109 AGLERR 1001. The GSET array is not large enough.	Internal software error; not under user control.	Contact system manager.
1110 AGLERR 1004. The string length is <0.	Internal software error; not under user control.	Contact system manager.

ERROR MESSAGE	MEANING	ACTION
1111 AGLERR 1101. Mapping value is not 0 or 1. Map remains unchanged.	Internal software error; not under user control.	Contact system manager.
1112 AGLERR 1201. FOPEN failure (FSERR nnn).	Internal software error; not under user control.	Note file system error number and contact system manager.
1113 AGLERR 1202. FCLOSE failure (FSERR nnn).	Internal software error; not under user control.	Note file system error number and contact system manager.
1114 AGLERR 1203. Error writing to device (FSERR nnn).	Internal software error; not under user control.	Note file system error number and contact system manager.
1115 AGLERR 1204. Error reading from device (FSERR nnn).	Internal software error; not under user control.	Note file system error number and contact system manager.
1116 AGLERR 1210. Error occurred when calling FCHECK.	Internal software error; not under user control.	Contact system manager.
1117 AGLERR 1215. FFILEINFO failure (FSERR nnn).	Internal software error; not under user control.	Note file system error number and contact system manager.
1118 AGLERR 1216. DMOVOUT failure.	Internal software error; not under user control.	Contact system manager.
1119 AGLERR 1217. DMOVIN failure.	Internal software error; not under user control.	Contact system manager.
1120 AGLERR 1218. GETDSEG failure.	Internal software error; not under user control.	Contact system manager.

ERROR MESSAGE	MEANING	ACTION
1121 AGLERR 1219. FREEDSEG failure.	Internal software error; not under user control.	Contact system manager.
1122 AGL error 1301. Internal error. A call to BINARY failed.	Internal software error; not under user control.	Contact system manager.
1123 AGLERR 1302. Internal error. Floating point result out of range.	Internal software error; not under user control.	Contact system manager.
1124 AGLERR 1502. No figure of that name was found.	The specified figure does not exist.	Check the figure name.
1125 AGLERR 1504. Specified figure file is currently in use.	Figure file is being written to at this time.	Try again later.
1126 AGLERR 1505. No figure file of that name can be found.	The specified figure file does not exist.	Check the figure file name.
1127 AGLERR 1506. Rotation angle out of range.	Internal software error; not under user control.	Contact system manager.
1128 AGLERR 1507. Scale factor out of range.	Internal software error; not under user control.	Contact system manager.
1129 AGLERR 1509. Attempt to open a figure file failed (FSERR nnn).	Internal software error; not under user control.	Note file system error number and contact system manager.
1130 AGLERR 1510. Attempt to close a figure file failed (FSERR nnn).	Internal software error; not under user control.	Note file system error number and contact system manager.
1131 AGLERR 1515. Figure file is opened for both input and output.	Internal software error; not under user control.	Contact system manager.

ERROR MESSAGE	MEANING	ACTION
1132 AGLERR 1516. System crash flag set to true in figure.	Error in figure file.	Recreate figure file.
1133 AGLERR 1518. Invalid figure name.	Incorrectly specified figure name.	Check the figure name.
1134 AGLERR 1800. File system error (FSERR nnn) figure not converted.	Internal software error; not under user control.	Note file system error number and contact system manager.
1135 AGLERR 1801. IFS/3000 error trying to print image (PSPERR nnn).	Internal software error; not under user control.	Refer to the IFS error number in this appendix for more information.
1136 AGLERR 1802. OUT2680A error (FSERR nnn); figure not converted.	PCONVERTFIGURE intrinsic caused error in OUT2680A temporary file.	Refer to the file system error number for more information.
1137 AGLERR 1803. CREATEPROCESS returned error nnn; figure not converted.	Figure has not been converted or printed.	Refer to CREATEPROCESS error number in the MPE Ininsics Reference Manual.
1138 AGLERR 1804. SORT/3000 returned error nnn; figure not converted.	Figure has not been converted or printed.	Refer to the error number in the SORT/3000 Reference Manual.
1139 AGLERR 1805. Conversion process failed; figure not converted.	Internal software error; not under user control.	Contact system manager.
1140 AGLERR 1806. Partitioning process failed; figure not converted.	Internal software error; not under user control.	Contact system manager.
1141 AGLERR 1807. Figure not converted.	Internal software error; not under user control.	Contact system manager.

ERROR MESSAGE	MEANING	ACTION
1142 Warning: AGL warning 1851. Figure may contain errors.	Internal conversion error; not under user control.	Check printed output for errors.
1143 Warning: AGL warning 1852. Figure may contain errors.	Internal partitioning error; not under user control.	Check printed output for errors.
1144 AGLERR 1808. Image would be positioned off page; image not printed.	Internal software error; not under user control.	Contact system manager.
1145 AGLERR 709. Unable to load AGL'2680DDD module; (LOAD ERR nnn).	If LOAD ERR number is 65, there are not enough system CST entries available.	Note LOAD error number and contact system manager.
1198 AGLERR nnn (FS parameter is nnn).	Internal software error; not under user control.	Contact system manager.
1199 Warning: AGL warning nnn. (FS parameter is n).	Internal software error; not under user control.	Contact system manager.
410 Old IDSCHAR file. Format must be converted using IDSCHAR.	IFS encountered a font/logo file made by an earlier release of IDSCHAR.	Bring the font/logo file into the new version of IDSCHAR to update the fil format.
411 Error encountered while converting between devices.	An enviornment file couldn't be completely or accurately converted.	Manually inspect the environment an correct any incorrecly converted or inverted values.
412 nn Warning(s) while converting between devices.	Non-fatal condition(s) while converting between devices.	None (This is just a warning).
413 Reduction is not available on the 2688A.	2:1 and 4:1 environment initialization cannot be done for the 2688A.	Select a full size HP-supplied environment for you initialization.

Merging PCELL Files

The PCELL file merge utility provides the capability to merge two or more separate character fonts into a single character font file. This will enable those users who own both the HP 2680A and the HP 2688A Laser Printers to store character fonts designed for these devices in the same PCELL file. This makes it much easier to recompile the same form for multiple devices since IFS will select the correct font for a device from the character font file specified.

It is suggested that you develop a logical system of merging character fonts within a common file. This will be useful when referencing character fonts which normally reside in a file containing different point sizes of the same character font. The added distinction using the merge utility is that the character fonts are device dependent. In addition, you may not elect to have all point sizes for a character font for both devices.

The IFS PCELL Merge Utility is an interactive session which runs as a sub-task of the IFS/3000 program. It is invoked at the MPE Operating System prompt as follows:

```
:RUN IFS.PUB.SYS, IFSUTIL
```

The utility will then prompt for the names of the PCELL files you wish to merge and a name for the resulting file. If the operation is successful, a message is displayed and you are prompted for another set of files to be merged. This process will be repeated until you press **RETURN** without typing a filename; OR, by typing EXIT.

Example

```
:RUN IFS.PUB.SYS, IFSUTIL
```

```
> Cell files to be merged:  
   FILE 1:  filename1  
   FILE 2:  filename2
```

PCELL UTILITIES

FILE 3: RETURN or EXIT
> Resulting cell file: *newfilename1*

Cell Files merged successfully. New cell file created.

END OF PROGRAM

:

OR, if you need assistance type ? or HELP at the
> prompt at any time as follows:

>?

OR,

>HELP

If any of the input PCELL files specified do not exist, or the file specified is not a PCELL file, or the input *filenames* cannot be opened for any reason, or the *newfilename* cannot be created, an error message is displayed and the > prompt is repeated. If the input *filenames* are opened successfully and the *newfilename* is created, then the files will be merged as follows:

- If the two input *filenames* contain character fonts designed for different devices (i.e. one for the HP 2680A, and one for the HP 2688A), then the resulting *newfilename* file will contain all the sizes for all the devices contained in the two input *filenames*, even though the character font may not be identical. This enables you to merge character fonts which are not the same font. It is suggested that they be similar or a logical substitute in order to maintain an orderly system of font file residence.
- If the two input *filenames* contain character fonts for the same device and there are no common point sizes, the resulting *newfilename* file will then contain the combination of all point sizes merged.
- If the two input *filenames* contain character fonts designed for the same device and there is a common point size, you will then be prompted to make a decision as to which one to put in the resulting *newfilename* file. The two character fonts cannot be combined on a cell by cell basis. The character font chosen will be included intact in the resulting file.

Updating PCELL Files

Environment files created and compiled using IFS/3000 versions prior to A.02.01 can be used in their compiled form with later software. If these older environment files are recompiled on IFS/3000 A.02.01 or later, the environment file will be automatically updated to the new environment file format.

PCELL files created using IDSCHAR/3000 versions prior to A.01.00 must be converted to the new file format before they can be used in the new versions of IDIFORM and IFS.

The easiest way to convert an old PCELL file is:

1. Run latest IDSCHAR.
2. Select the Modify function and enter the name of the PCELL file to be converted.
3. The file will then be converted, and you will get the following message:

The file has been updated to the new PCELL format.

4. Return to the Main Menu or continue working on the file.

Once an old PCELL file is converted to the new format, that file cannot be used with previous versions of IDSCHAR.

If the PCELL file runs out of space when converting, create a new file and copy the old PCELL file to the new file, as shown in the following example:

```
:BUILD filename;REC=64,2,F,BINARY;CODE=PCELL;DISC=40
:RUN FCOPY.PUB.SYS
>From=oldcellfilename;to=filename
>Exit
```

Now you can go back to step 1 to convert the file. Don't forget to rename the newly converted PCELL file to whatever name was originally referenced in any environment files.

THIS PAGE SHOULD BE BLANK

Graphics can be printed on the HP 2608S, HP 2563A, HP 2565A and HP 2566A impact printers using a subset of the PSP intrinsics. The supported intrinsics are:

- PINITDEVICE
- PERRMSG
- PCONVERTFIGURE(A)
- PFIGUREINFO(A)
- PFLASHRASTER(A)

Refer to section 5, Programmatic Interface, for more information about these intrinsics. You can also plot directly to the HP 2608S, HP 2563A HP 2565A and HP 2566A printers in the HP Business Graphics Products - DSG, HPDRAW, and HPEASYCHART.

The resolution for both printers is 70 dots per inch in the horizontal (x) direction and 72 dots per inch

in the vertical (y) direction. Any units specified should be computed using these values. The largest image we recommend you specify to be printed on these printers is 924 dots (13.5 inches or 342.9 mm) in the x direction and 720 dots (10 inches or 254 mm) in the y direction.

Since environments aren't downloaded to the impact printers at print time, the permanent downloadable form, downloadable character font, logical page and related concepts do not apply. Instead, all positioning and rotation is done with respect to the physical page. This is why only a subset of the graphics intrinsics and none of the formatting intrinsics can be used with these printers.

A sample PASCAL program using the PINITDEVICE, PERRMSG, PCONVERTFIGURE and PFLASHRASTER intrinsics to print a figure on the HP 2563A printer is included on the following pages.

IMPACT PRINTER GRAPHICS

```

1.000      0 0      SUSLINIT, STANDARD_LEVEL 'HP3000'$
2.000      0 0
3.000      0 0      program Quarterly_Report (input, output);
4.000      0 0
5.000      0 0      {
6.000      ** 0      { This sample program illustrates the calling sequences for a subset
7.000      ** 0      { of the IFS/3000 graphics intrinsics from Pascal. The program opens
8.000      ** 0      { a SpoolFile specifying the logical device name for the 2563A
9.000      ** 0      { printer.
10.000     ** 0      {
11.000     0 0      { }
12.000     0 0
13.000     0 0      const
14.000     0 0          CCE                = 2;
15.000     0 0          cCR                = chr(13);
16.000     0 0          cStatusOk         = 0;
17.000     0 0
18.000     0 0      type
19.000     0 0          SmallInteger       = -32768..32767;
20.000     0 0          ComAreaRec        = record
21.000     0 0              Status          : SmallInteger;
22.000     0 0              ErrorNum       : SmallInteger;
23.000     0 0              ErrParm1      : SmallInteger;
24.000     0 0              ErrParm2      : SmallInteger;
25.000     0 0              MsgFileNum    : SmallInteger;
26.000     0 0              Reserved     : array [1..2995]
27.000     0 0                  of SmallInteger;
28.000     0 0              end;
29.000     0 0          DevArray          = packed array [1..5] of char;
30.000     0 0
31.000     0 0      const
32.000     0 0          cDevParm            = DevArray ['LP63',cCR];
33.000     0 0
34.000     0 0      type
35.000     0 0          TagType                = (ByteAddr, WordAddr);
36.000     0 0          CharArrayType       = packed array [1..80] of char;
37.000     0 0          LogArrayType        = array [1..40] of SmallInteger;
38.000     0 0          LogicalRec        = record
39.000     0 0              dummybyte : char; {To keep data aligned}
40.000     0 0              case AddrTag : TagType of
41.000     0 0                  ByteAddr : (CharArray : CharArrayType);
42.000     0 0                  WordAddr  : (LogArray  : LogArrayType);
43.000     0 0              end;
44.000     0 0      $PAGE$

```

IMPACT PRINTER GRAPHICS

```

45.000      0 0      var
46.000      0 0      DeviceClass          : DevArray;
47.000      0 0      FormalDesignator     : packed array [1..9] of char;
48.000      0 0      FSerr                 : SmallInteger;
49.000      0 0      ComArea               : ComAreaRec;
50.000      0 0      Language              : SmallInteger;
51.000      0 0      XPosition             : real;
52.000      0 0      YPosition             : real;
53.000      0 0      SpoolFileNum          : SmallInteger;
54.000      0 0      AreaLen               : SmallInteger;
55.000      0 0      OutputDev              : packed array [1..6] of char;
56.000      0 0      FigureName            : packed array [1..16] of char;
57.000      0 0      FigureFileName        : packed array [1..35] of char;
58.000      0 0      RasterFileName        : packed array [1..35] of char;
59.000      0 0      Height                : real;
60.000      0 0      Rotation              : SmallInteger;
61.000      0 0      Units                 : SmallInteger;
62.000      0 0      RasterNumber          : SmallInteger;
63.000      0 0      PositionMode          : SmallInteger;
64.000      0 0
65.000      0 0      function FOPEN : SmallInteger; intrinsic;
66.000      0 0      procedure FCLOSE;      intrinsic;
67.000      0 0      procedure FCHECK;      intrinsic;
68.000      0 0      procedure FWRITE;      intrinsic;
69.000      0 0      procedure TERMINATE;    intrinsic;
70.000      0 0      procedure PINITDEVICE;  intrinsic;
71.000      0 0      procedure PERRMSG;      intrinsic;
72.000      0 0      procedure PCONVERTFIGURE; intrinsic;
73.000      0 0      procedure PFLASHRASTER; intrinsic;
74.000      0 0      procedure PFIGUREINFO;  intrinsic;
75.000      0 0
76.000      0 0      $PAGES

```

IMPACT PRINTER GRAPHICS

```
77.000      0 0      procedure CHECK_FOR_ERROR;
78.000      0 0
79.000      0 0      {
80.000      ** 0      { This routine checks the ComArea's Status to see if an error
81.000      ** 0      { occurred. If so, then PERRMSG is called, the error is reported,
82.000      ** 0      { and the program terminates.
83.000      0 0      { }
84.000      0 0
85.000      0 0      var
86.000      0 0      ErrorMsg          : packed array [1..80] of char;
87.000      0 0      ArrayLen          : SmallInteger;
88.000      0 0      MsgLen            : SmallInteger;
89.000      0 0
90.000      0 0
91.000      0 1      begin {CHECK_FOR_ERROR}
92.000      0 1
93.000      0 1      with ComArea do
94.000      1 1
95.000      1 2      begin {with}
96.000      1 2      if Status <> cStatusOk then
97.000      2 3      begin {Get error/warning message}
98.000      2 3      writeln (output, '(IFS Error ', ErrorNum:1, ')');
99.000      3 3      ArrayLen := 80;
100.000     4 3      PERRMSG (ComArea, ErrorMsg, ArrayLen, MsgLen);
101.000     5 3      writeln (output, ErrorMsg:MsgLen);
102.000     6 3      TERMINATE;
103.000     6 3      end; {Get error/warning message}
104.000     7 2
105.000     6 2      end; {with}
106.000     7 1
107.000     6 1      end; {CHECK_FOR_ERROR}
108.000     7 0
109.000     7 0      $PAGE$
```

```

110.000      0 1  begin   {Quarterly_Report}
111.000      0 1
112.000      0 1      {
113.000     ** 1      { First, Open the Spoolfile.
114.000      0 1      { }
115.000      0 1
116.000      0 1      FormalDesignator := 'SPOOLOUT ';
117.000      1 1      DeviceClass := cDevParm;
118.000      2 1      SpoolFileNum := FOPEN (FormalDesignator,
119.000      3 1                octal('604'), {CTL.Undef,Formal,ASCII,New}
120.000      3 1                1,          {Write only}
121.000      3 1                ,          {Default RecSize}
122.000      3 1                DeviceClass);
123.000      3 1      if ccode <> CCE then
124.000      4 2      begin {FOPEN error}
125.000      4 2          writeln(output, 'Cannot Open Spoolfile');
126.000      5 2          FCHECK (SpoolFileNum, FSErr);
127.000      6 2          writeln(output, '(FSERR ', FSErr:1, ')');
128.000      7 2          TERMINATE;
129.000      7 2      end;   {FOPEN error}
130.000      8 1
131.000      8 1      {
132.000     ** 1      { Next, Initialize the IFS/3000 intrinsics.
133.000      8 1      { }
134.000      8 1
135.000      8 1      AreaLen := 3000;
136.000      9 1      Language := 5;
137.000     10 1      OutputDev := '2563A ';
138.000     11 1      PINITDEVICE (ComArea, AreaLen, Language, SpoolFileNum, OutputDev);
139.000     12 1      CHECK_FOR_ERROR;
140.000     13 1
141.000     13 1      {
142.000     ** 1      { Convert a figure created with DSG/3000 into a raster image.
143.000     13 1      { }
144.000     13 1
145.000     13 1      FigureFileName := 'QTRSALES ';
146.000     14 1      FigureName := 'SECOND QUARTER '; {Terminated with a blank}
147.000     15 1      RasterFileName := 'R2NDQTR '; {Terminated with a blank}
148.000     16 1      Units := 1; {Inches}
149.000     17 1      Height := 8.0; {Height in inches}
150.000     18 1      Rotation := 0; {Could be 0, 90, 180, 270}
151.000     19 1      PCONVERTFIGURE (ComArea, FigureFileName, FigureName, RasterFileName,
152.000     20 1      OutputDev, Height, Units, Rotation);
153.000     20 1      CHECK_FOR_ERROR;
154.000     21 1
155.000     21 1      writeln ('**** Converted figure ');
156.000     22 1
157.000     22 1      {
158.000     ** 1      { Now print the image positioned relative to the upper left
159.000     ** 1      { corner of the physical page.
160.000     22 1      { }
161.000     22 1
162.000     22 1      XPosition := 1.5; {Inches to the right}
163.000     23 1      YPosition := 1.0; {Inches down the page}
164.000     24 1      PositionMode := 0; {Relative}
165.000     25 1      PFLASHRASTER (ComArea, RasterFileName, XPosition, YPosition, Units,
166.000     26 1      PositionMode);

```

IMPACT PRINTER GRAPHICS

```
167.000    26 1    CHECK_FOR_ERROR;
168.000    27 1
169.000    27 1    writeln ('***** printed raster image ');
170.000    28 1    {
171.000    ** 1    { Finally, close the spoolfile.
172.000    28 1    { }
173.000    28 1
174.000    28 1    FCLOSE (SpoolFileNum, 0, 0);
175.000    29 1
176.000    28 1    end.    {Quarterly_Report}

NUMBER OF ERRORS = 0    NUMBER OF WARNINGS = 0
PROCESSOR TIME 0: 0: 7    ELAPSED TIME 0: 0:13
NUMBER OF LINES = 176    LINES/MINUTE = 1508.6
```

INDEX

A

ASCII Character Codes	2-2
ASCII Character Code Field	3-72
ASCII File	B-3
Abbreviations	4-9
Active/Inactive Logical Pages	2-14
Actual Spacing Field	3-51
Another Environment Field (Initialize Menu)	3-25

B

Batch Job	4-8
Bar Codes (info.)	D-36
Bar Codes (samples)	D-35.1,D-54
Byte Arrays	5-6

C

Calling the Intrinsics	5-2
Carriage Control	4-5,E-8
Cell File	2-5
Change Fonts	2-4
Change Forms or VFC	3-44
Change VFC Specification Field	3-55
Character Cell	2-2
Character Code Number	2-2
Character Font File	2-5
Character Font Menu Function Keys	3-65
Character Font Menu Field Discussion	3-66
Character Font Menu Font Number Field	3-67
Character Font Menu Orientation Field	3-68
Character Font Menu Font Name Field	3-69

Character Font Menu Font File Field	3-69
Character Font Menu Match Size Field	3-71
Character Font Menu Units Field	3-72
Character Font Menu ASCII Code Field	3-72
Character Font Menu Protected Information Fields	3-73
Character Font Menu	3-64
Character Font Name Field	3-20
Character Font Number	2-4
Character Font Orientations	2-6
Character Font Sizes	2-6.1
Character Font	2-4
Character Set Keys	3-6
Comarea	4-3
Commands (Figure Maker)	B-3
Commands (LPS Interpreter)	4-3
Comment Character	4-4
Comments	4-4
Communication Area	5-6
Compiling Environments	2-11
Continuation Lines	4-4
Conventions (Menu)	3-2
Conversion (Figure)	2-24
Conversion (Forms)	2-7
Converting Other Graphic Figures with Figure Maker	B-2
Copy Associated Compiled Parts Field	3-80
Copy Menu Copy Associated Compiled Parts	3-80
Copy Menu Environment File to Copy Field	3-76
Copy Menu Field Discussion	3-76
Copy Menu Function Keys	3-75
Copy Menu Method Of Copy Field	3-78
Copy Menu Parts To Copy Fields	3-77
Copy Menu	3-74
CTRL N (SHIFT OUT)	2-4
CTRL O (SHIFT IN)	2-4

INDEX

D

DSG/3000 2-20,B-6
Decision Support Graphics 2-20
Default Environments 2-10
Default Form File Field 3-34
Default Measurement System Field 3-34
Default Parameters (LPS Interpreter) 4-4
Device Field (Main Menu) 3-19
Display Control Keys 3-7

E

EZCHART (HPEasyChart) 2-21
Easy Chart (HPEasyChart) 2-21
Eighth bit 2-4
Ending A Session 4-8
Environment File Menu 3-11
Environment File Menu Function Keys 3-12
Environment File Menu (Field Discussion) 3-13
Environment File 2-10
Environment File to Copy 3-76
Environment File 4-7
Environment Files (updating) H-3
Environment Size 4-3,4-6
Environments (HP Supplied) 1-6
Error Messages (LPS) 4-72
Error Messages (IFS/3000) G-1
Error Number (Programmatic Interface) 5-8
Error Parameters 1 and 2
 (Programmatic Interface) 5-8
Examples (Figure Maker) B-8

F

Features (Laser Printing System) 1-2
Field Discussion (Copy Menu) 3-76
Field Discussion (Environment File Menu) 3-13
Field Discussion (Initialize Menu) 3-23
Field Discussion (Logical Page Menu) 3-43
Field Discussion (Logical Page Forms Menu) 3-54
Field Discussion (Main Menu) 3-15
Field Discussion (Multi-Copy Forms Menu) 3-36
Field Discussion (Physical Page Menu) 3-30
Field Discussion (VFC Menu) 3-61
Field Discussion Character Font Menu 3-66
Field Edit Keys 3-8
Figure (Print) 2-27
Figure File 2-22
Figure Files (updating) B-6
Figure Maker (Run) B-2
Figure Maker Commands B-3
Figure Maker Error Messages B-11
Figure Maker Examples B-8
Figure Maker 2-22
Figure Rotation and Conversion 2-24
Figure 2-22
Font File Field 3-69
Font Name Field 3-69
Font Number Field 3-67
Font/Page Number Field (Main Menu) 3-20
Form 1 Field 3-55
Form File 1 Field 3-55
Form File Field (Multi-Copy Forms Menu) 3-38
Formatting Commands 4-3
Forms File 2-8
Forms 2-7

Function Keys (Character Font Menu) . . . 3-65
 Function Keys (Copy Menu) 3-75
 Function Keys (Environment File Menu) . . 3-12
 Function Keys (Initialize Menu) 3-22
 Function Keys (Logical Page Menu) 3-42
 Function Keys (Logical Page Forms Menu) . 3-53
 Function Keys (Main Menu) 3-15
 Function Keys (Multi-Copy Forms Menu) . . 3-35
 Function Keys (Physical Page Menu) 3-29
 Function Keys (VFC Menu) 3-60
 Function Keys 3-4
 FUPDATE B-6

G

Graphics Software 1-11,B-1
 Graphics 2-19

H

HP 2563A, 2565A, 2566A, 2608S I-1
 HP 2680 Character Fonts Supplied D-1
 HP 2680 Environment Formats C-1
 HP 2688 Character Fonts Supplied D-1
 HP 2688 Environment Formats C-1
 HP 36580 (IFS) vii,5-13
 HP 36581 (IDS) vii
 HP 36583 (Graphics) vii
 HP Defined Environment Field 3-23
 HPDRAW 2-21,B-6
 HPEasyChart 2-21,B-6
 HPENV.SYS (Environment File Location) . . C-1
 HP2680A.SYS (Changed to HPENV.SYS) . . . C-1

I

IDS/3000 (Customization) 1-6
 IDSCHAR (description) 1-7
 IDSCHAR 2-1
 IDSFORM (description) 1-8
 IDSFORM 2-5
 IFS/3000 (Running) 3-1
 IFS/3000 (description) 1-8
 IFS/3000 Menus (Using) 3-8
 IFS/3000 2-10
 Impact Printer Graphics I-1
 Inactive/Active Logical Pages 2-14
 Initialize Menu (HP Defined
 Environment Field) 3-23
 Initialize Menu Another Environment Field . 3-25
 Initialize Menu Field Discussion 3-23
 Initialize Menu Function Keys 3-22
 Initialize Menu Reduction Field 3-25
 Initialize Menu 3-21
 Initially Active Field 3-44
 Interactive Figure Maker B-2
 Interactive Session 4-7
 Intrinsic Calls 4-3
 Intrinsic 5-2
 PACTIVATEPAGE 5-15
 PCELL Merge Utility H-1
 PCONVERTFIGURE[A] 5-17,I-1
 PCONVERTRASTER[A] 5-21
 PDEACTIVATEPAGE 5-27
 PDELETERASTER 5-28
 PERRMSG 5-30
 PFIGUREINFO[A] 5-33,I-1
 PFLASHRASTER[A] 5-40,I-1

INDEX

PFONTINFO	5-44
PFONTNUM	5-47
PINITDEVICE	5-49,I-1
PINITIALIZE	5-53
PLOADRASTER	5-56
PLOGPAGEINFO	5-58
PMOVEPENABS	5-62
PMOVEPENREL	5-63
PNEWFORM	5-65
PNEWPAGE	5-67
PNEWPHYSPAGE	5-69
PNEWSUBFORM	5-71
PPRINTFIGURE[A]	5-73
PPRINTRASTER[A]	5-79
PSELECTPAGE	5-83
PSTATEINFO	5-85
PSTRINGWIDTH	5-88
PUSEFONT	5-91
PWRITEFIELD	5-92

K

Keys (Character Set)	3-6
Keys (Display Control)	3-7
Keys (Field Edit)	3-8
Keys (Terminal Control)	3-5

L

LPS Commands	4-12
.ACTIVATEPAGE	4-12
.COMMENTCHAR	4-14
.CONTINUECHAR	4-15
.CONTROLCHAR	4-15

.CONVERTFIGURE	4-16
.CONVERTRASTER	4-18
.DEACTIVATEPAGE	4-22
.DEFAULTCHAR	4-23
.DELETERSTER	4-24
.ENDWRITEFIELD	4-25
.EXIT	4-27
.FLASHRASTER	4-28
.HELP	4-31
.INCLUDE	4-33
.LOADRASTER	4-35
.MOVEPENABS	4-37
.MOVEPENREL	4-39
.NEWFORM	4-41
.NEWPAGE	4-42
.NEWPHYSPAGE	4-43
.NEWSUBFORM	4-44
.PRINTFIGURE	4-45
.PRINTRASTER	4-51
.SELECTPAGE	4-55
.SHIFTCHAR	4-56
.USEFONT	4-57
.WRITEFIELD	4-59

LPS Example (Batch Job)	4-71
LPS Example (Environment File)	4-62
LPS Example (Environment File)	4-66
LPS Interpreter (Run)	4-6
LPS Interpreter (description)	1-9,4-1
LPS Interpreter Commands	4-9
LPS Terminology	4-8
Laser Printer	1-1
Laser Printing System Features	1-2
Laser Printing System Hardware	1-3
Line Position Fields	3-62
Log-On and Run IFS/3000	3-10
Logical Page Field	

(Multi-Copy Forms Menu) 3-37

Logical Page Forms Menu 3-51

Logical Page Forms Menu Function Keys . . 3-53

Logical Page Forms Menu Field Discussion . 3-54

Logical Page Forms Menu (Change VFC) . . 3-54

Logical Page Forms Menu (Form 1 Field) . . 3-55

Logical Page Forms Menu Form File 1 Field . 3-55

Logical Page Forms Menu Scale Field 3-56

Logical Page Forms Menu
 Positioning Field 3-56

Logical Page Forms Menu Manual
 Positioning 3-57

Logical Page Menu (Actual Spacing Field) . 3-51

Logical Page Menu Change Forms
 or VFC Field 3-44

Logical Page Menu Field Discussion 3-43

Logical Page Menu Function Keys 3-42

Logical Page Menu Initially Active Field . . 3-44

Logical Page Menu Logical Page
 Number Field 3-43

Logical Page Menu Orientation Field 3-44

Logical Page Menu Size and
 Positioning Field 3-45

Logical Page Menu 3-40

Logical Page Number Field 3-43

Logical Page Numbers 2-14

Logical Page Orientations 2-14

Logical Pages (Active/Inactive) 2-14

Logical Pages 2-13

Logo File 2-5

M

MPE Commands A-1

MPE Intrinsic A-4

Main Menu (Character Font Name Field) . . 3-20

Main Menu (Recompile Everything) 3-21

Main Menu Device Field 3-19

Main Menu Field Discussion 3-15

Main Menu Font/Page Number Field 3-20

Main Menu Function Keys 3-15

Main Menu Selection Field 3-15

Main Menu 3-13

Manual Positioning Field 3-57

Manual Usage viii

Match Size Field 3-71

Menu (Character Font) 3-64

Menu (Copy) 3-74

Menu (Environment File) 3-11

Menu (Initialize) 3-21

Menu (Logical Page Forms) 3-51

Menu (Logical Page) 3-40

Menu (Main) 3-13

Menu (Multi-Copy Forms) 3-34

Menu (Physical Page) 3-29

Menu Conventions 3-2

Menus (Using) 3-8

Merge Character Fonts H-1

Message Catalog File Number
 (Programmatic Interface) 5-8

Method Of Copy Field 3-78

Multi-Copy Forms Field
 (Physical Page Menu) 3-31

Multi-Copy Forms Menu 3-34

Multi-Copy Forms Menu Function Keys . . 3-35

Multi-Copy Forms Menu Field Discussion . 3-36

Multi-Copy Forms Menu Logical Page Field 3-37

INDEX

Multi-Copy Forms Menu Forms File Field	3-38
Multi-Copy Forms Menu Scale Field	3-40
Multi-Copy Forms	2-8
Multiple Line Commands	4-4

N

Non-Graphics Software	1-6
Non-Interactive Session	4-8
Number of Copies Field (Physical Page Menu)	3-30
Number of Copies	2-13
Numbers (Logical Page)	2-14

O

Orientation Field (Character Font Menu)	3-68
Orientation Field	3-44
Orientations (Character Font)	2-6
Orientations (Logical Page)	2-14
Output Device	4-7
Output File	4-5
Overview, Software products	vii

P

Parameter Data Types	5-5
Partitioned Raster File	2-23
Partitioned Raster Image	2-23
Parts To Copy Fields	3-77
PCELL File Merge	H-1
PCELL File Update	H-3
Physical Page Copies	2-13

Physical Page Menu (Primary and Secondary Font Fields)	3-33
Physical Page Menu (Default Measurement System Field)	3-34
Physical Page Menu (Default Form File Field)	3-34
Physical Page Menu Function Keys	3-29
Physical Page Menu Field Discussion	3-30
Physical Page Menu Multi-Copy Forms Field	3-31
Physical Page Menu Number of Copies Field	3-30
Physical Page Menu Physical Page Size Fields	3-31
Physical Page Menu	3-29
Physical Page Size Fields (Physical Page Menu)	3-31
Physical Page	2-12
Position On Page Field	3-56
Positioning a Partitioned Raster Image	2-28
Primary Font Field	3-33
Primary and Secondary Font Fields	3-33
Primary/Secondary Fonts, changing	2-4
Printing a Figure	2-27
Printing a Partitioned Raster Image	2-27
Printing on Impact Printers	I-1
Procedure Descriptions (Programmatic Interface)	5-1
Programmatic Interface (description)	1-9
Programmatic Interface Procedure Descriptions	5-1
Programmatic Interface Capabilities	5-4
Protected Information Fields	3-73

R

Raster File	2-23
Raster Image (Positioning)	2-28
Raster Image (Printing)	2-27
Raster Image (Terminology)	2-23
Recompile Everything Field	3-21
Reduction Field (Initialize Menu)	3-25
Reference Documents	ix
Rotation (Figure)	2-24
Running Figure Maker	B-2
Running IFS/3000	3-1
Running LPS Interpreter	4-6

S

Sample Programs	F-1
Scale Field (Logical Page Forms Menu)	3-56
Scale Field (Multi-Copy Forms Menu)	3-40
Secondary Font Field	3-33
Selection Field (Main Menu)	3-15
Session (Interactive)	4-7
Session (Non-Interactive)	4-8
Session End	4-8
Sessions	4-7
Shift Character	4-4
SHIFT IN (CTRL O)	2-4
SHIFT OUT (CTRL N)	2-4
Size Specification and Positioning Field	3-45
Sizes (Character Font)	2-6.1
Software (Graphics)	1-11
Software (Non-Graphics)	1-6
Specifying Environments	2-12
String Parameter in an Error (Programmatic Interface)	5-8

Subforms	2-8
Supplied Environments	1-6

T

Terminal Control Keys	3-5
Terminal Keyboard (Using)	3-3
Terminology (Cell File)	2-5
Terminology (Character Cell)	2-2
Terminology (Character Code Number)	2-2
Terminology (Character Font)	2-4
Terminology (Character Font Number)	2-4
Terminology (Character Font File)	2-5
Terminology (Character Font Orientations)	2-6
Terminology (Character Font Sizes)	2-6.1
Terminology (Compiling Environments)	2-11
Terminology (Default Environments)	2-10
Terminology (Environment File)	2-10
Terminology (Figure Maker)	2-22
Terminology (Figure File)	2-22
Terminology (Figure Rotation and Conversion)	2-24
Terminology (Figure)	2-22
Terminology (Forms File)	2-8
Terminology (Forms)	2-7
Terminology (HPDRAW)	2-21
Terminology (HPEasyChart)	2-21
Terminology (IDSCHAR)	2-1
Terminology (IDSFORM)	2-5
Terminology (LPS)	4-8
Terminology (Logical Pages)	2-13
Terminology (Logical Page Orientations)	2-14
Terminology (Logical Page Numbers)	2-14
Terminology (Logical Pages Active/Inactive)	2-14
Terminology (Logo File)	2-5

INDEX

Terminology (Multi-Copy Forms) 2-8
Terminology (Partitioned Raster Image) . . . 2-23
Terminology (Partitioned Raster File) 2-23
Terminology (Physical Page) 2-12
Terminology (Physical Page Copies) 2-13
Terminology (Positioning a
 Partitioned Raster Image) 2-2
Terminology (Printing a Figure) 2-27
Terminology (Printing a
 Partitioned Raster Image) 2-27
Terminology (Specifying the Environment) . . 2-12
Terminology (Subforms) 2-8
Terminology (VFC) 2-19
Terminology (Vertical Format Control) 2-19
Text Files 4-3
Top Margin Field 3-63
Two-Line Commands 4-4
Type of VFC Field 3-61

U

Units Field 3-72
Updating files B-6,H-3

USEFONT command 2-5,4-58
User Area Status 5-7
Using IFS/3000 Menus 3-8
Using this manual viii

V

VFC File Field 3-63
VFC Menu Field Discussion 3-61
VFC Menu Function Keys 3-60
VFC Menu Line Position Fields 3-62
VFC Menu Top Margin Field 3-63
VFC Menu Type of VFC Field 3-61
VFC Menu VFC File Field 3-63
VFC Specifications E-1
VFC 2-19
Vertical bar (with USEFONT) 2-5
Vertical Format Control 2-19
Vertical Format Control Menu 3-58



MANUAL UPDATE

FK 29 MAR 85

IFS/3000 Reference Guide Update for manual part number 36580-90001

This manual update package corresponds with the software update A.02.03. Changes in the program as well as typographic and informational corrections are included. If your version ID number (displayed on your terminal screen when entering and exiting IFS/3000) is lower than A.02.03, and you wish to update it, contact your nearest HP Sales and Service Office for assistance.

Use the pages in this package to update your Interactive Formatting System Reference Guide. Check each item off as you complete it.

Remove These Old Pages	Insert These New Pages	Changes: (indicated by a vertical bar in page margins)	Done? <input checked="" type="checkbox"/>
i thru xviii	i thru xx	New title page, printing history, list of effective pages, <i>Overview</i> and <i>Contents</i> .	
2-3 thru 2-6	2-3 thru 2-6.2	New topic, <i>Primary and Secondary Character Fonts</i> .	
3-33, 3-34	3-33, 3-34	Additional info on Primary and Secondary Font Fields.	
4-1 thru 4-8	4-1 thru 4-8	New <i>comarea</i> info, new paragraph (<i>Carriage Control</i>), stack area warning info.	
4-19, 4-20	4-19, 4-20	Correction to <i>inrasterfilename</i> description.	
5-1, 5-2	5-1, 5-2	Change in Figure 5-1.	
5-21 thru 5-26	5-21 thru 5-26	<i>PCONVERTASTER</i> Syntax change.	
Apdx B (all)	Appendix B	New Appendix title, new section (<i>Updating Figure Files</i>).	
D-1 thru D-6	D-1 thru D-6	New font sizes (11.2 in HELV and ROM families).	
D-35 thru D-38	D-35 thru D-38	HP 2680 bar code samples and new bar code info.	
D-49 thru D-56	D-49 thru D-56	New HP 2688 character sets and bar code.	
Apdx H (all)	Appendix H	New Appendix title, new section (<i>Updating Files</i>).	
I-1, I-2	I-1, I-2	New impact printers, <i>PCONVERTASTER</i> deleted.	
Index (all)	Index	New index.	

DON'T HAVE ORIGINAL
DATED - REPLACED ANYWAY!

