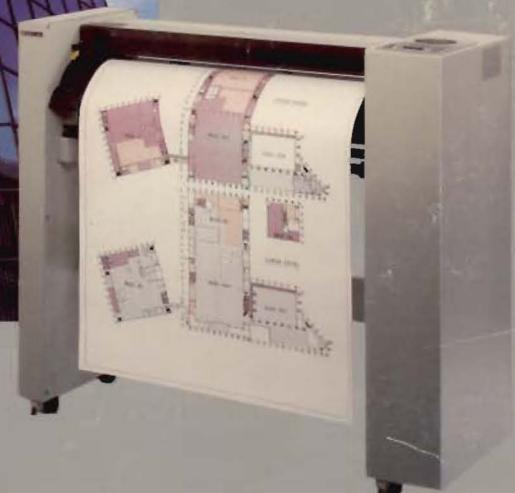


F. VAN GRIEKEN.

HEWLETT-PACKARD

HP DraftMaster Plotter Programmer's Reference



HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

HP DraftMaster Plotter Programmer's Reference



Manual Part Number: 07595-90001

Printed in U.S.A., JANUARY 1987

Copyright © Hewlett-Packard Company 1987
16399 W. Bernardo Drive, San Diego, CA 92127-1899

Part No. 07595-90001

Printed in U.S.A.

First Edition — January 1987

How to Use This Documentation

This *Programmer's Reference* provides you with information about the Hewlett-Packard Graphics Language (HP-GL) and how it is used with your plotter.

You must use a programming language in addition to HP-GL. However, this manual will not teach you how to program your computer. Your method of programming will depend on your computer system, the programming language you use, and your level of expertise. The manual does, however, give details about many plotting concepts, along with digitizing, interfacing and handshaking, and device control.

How to Use the Programmer's Reference

Once you are familiar with the fundamental plotting concepts and HP-GL, use the manual as a detailed reference to specific questions. Use the *Pocket Guide* (described on the next page) for a quick reminder of a specific HP-GL instruction's syntax.

Organization

Part I: Chapters 1–4 discuss general information about programming, plotting, and graphics. They provide an introduction to HP-GL and tell you how to produce simple plots using straight lines and rectangles.

Part II: Chapters 5–10 give more advanced plotting techniques. This section includes using different types of lines and area fills; drawing circles, arcs, and wedges; labeling your plots; creating your own polygons; and changing the size of your plot.

Part III: Chapters 11–13 cover special applications for advanced users. This section includes digitizing, rollfeed and long-axis plotting, and alternate character sets.

Part IV: Chapters 14–16 discuss device control. This section includes obtaining information from the plotter, using device-control instructions, interfacing, and handshaking.

Conventions

Before reading the rest of this manual, you should understand the type styles and number representation used in the text.

SMALL BOLDFACE TYPE denotes buttons or switches located on the plotter or displayed on the front panel.

BOLD CONDENSED TYPE denotes a single, nonprinting ASCII character (control code) to be sent to the plotter.

Numbers are typed using SI (International System of Units) standards. Numbers with more than four digits are placed in groups of three—separated by a space instead of a comma—counting both to the left and right of the decimal point (e.g., 54 321.123 45).

The symbols and type styles used to represent the syntax requirements of HP-GL and device-control instructions are discussed in Chapters 3 and 15, respectively.

All references to RS-232-C interface in this manual apply equally to RS-232-C and CCITT V.24 interfaces. The term RS-232-C is used for simplicity.

Additional Documentation

Following is a list of additional documentation for the plotter.

- **DraftMaster User's Guide** (Part No. 07595-90002). The User's Guide is shipped with the plotter. It contains detailed information about loading paper and pens, using the front panel, determining the best pen/paper combinations, and ordering available accessories. It also describes how to connect the plotter to various computers.
- **DraftMaster Pocket Guide** (Part No. 07595-90003). The Pocket Guide is a convenient reference list of all HP-GL and device-control instructions along with their parameters.

You can order these from the dealer or the HP Sales and Support Office where you purchased your plotter.

Table of Contents

Part I — Fundamental Concepts & Plotting

How to Use this Documentation

How to Use the Programmer's Reference	iii
Organization	iii
Conventions	iv
Additional Documentation	iv

Chapter 1: Programming Concepts

Terms You Should Understand	1-1
HP-GL and Device-Control Instructions	1-2
HP-GL Instructions	1-2
Device-Control Instructions	1-3
Using HP-GL With BASIC, FORTRAN, and Pascal	1-4
Configuration Statements	1-4
Output Statements	1-6
Input Statements	1-6
Number Formats	1-7
Using the Programming Examples	1-7
Programming Statements You May Need to Change	1-8
Hints for the Novice Programmer	1-8
Program Errors	1-9

Chapter 2: Plotting and Graphics Concepts

Plotting with the Coordinate System	2-1
Units of Measure and Scaling	2-3
Plotter Units	2-3
User Units	2-3
Scaling	2-4
Graphic Limits	2-4
Hard-Clip Limits	2-4
Soft-Clip Limits	2-5
Graphics Concepts	2-7
Lines	2-7

Chapter 2: Plotting and Graphics Concepts (Continued)

Rectangles	2-7
Plot Enhancements	2-8
Circles	2-9
Arcs	2-9
Wedges	2-9
Polygons	2-10
Labels	2-10

Chapter 3: Beginning Your HP-GL Program

Instructions Covered	3-1
Terms You Should Understand	3-1
Understanding HP-GL Syntax	3-2
Omitting Optional Parameters	3-3
Parameter Formats	3-3
Notations Used in This Manual for Expressing Syntax	3-4
Setting the Plotter to Standard Values	3-5
Sizing Your Plot	3-6
Hard-Clip Limits	3-6
Media Loaded Vertically and Horizontally	3-7
The Scaling Points P1 and P2	3-7
Plotter Unit Values for Different Media Sizes	3-8
Scaling	3-11
DF, Default	3-14
IN, Initialize	3-17
IP, Input P1 and P2	3-19
SC, Scale	3-21

Chapter 4: Drawing Lines and Rectangles

Instructions Covered	4-1
Terms You Should Understand	4-1
Selecting and Controlling the Pen	4-2
Pen Position and Location	4-2
Absolute versus Relative Movement	4-3
Drawing Lines	4-5
Drawing Rectangles	4-6
EA, Edge Rectangle Absolute	4-7
ER, Edge Rectangle Relative	4-9
PA, Plot Absolute	4-11
PD, Pen Down	4-12
PR, Plot Relative	4-14
PU, Pen Up	4-16
SP, Select Pen	4-17

Part II — Advanced Plotting

Chapter 5: Enhancing Your Plots

Instructions Covered	5-1
Terms You Should Understand	5-1
Using Line Types	5-2
Using Fill Types	5-3
User-Defined Fill Types	5-4
Filling Rectangles	5-5
FT, Fill Type	5-6
LT, Line Type	5-9
PT, Pen Thickness	5-12
RA, Fill Rectangle Absolute	5-14
RR, Fill Rectangle Relative	5-16
SM, Symbol Mode	5-18
TL, Tick Length	5-20
UF, User-Defined Fill Type	5-23
XT, X-Tick	5-28
YT, Y-Tick	5-29

Chapter 6: Drawing Circles, Arcs, and Wedges

Instructions Covered	6-1
Terms You Should Understand	6-1
Chords and Chord Tolerance	6-2
Chord Angle	6-2
Deviation Distance	6-3
Drawing Circles and Arcs	6-4
Drawing Wedges	6-5
AA, Arc Absolute	6-7
AR, Arc Relative	6-9
CI, Circle	6-11
CT, Chord Tolerance	6-15
EW, Edge Wedge	6-17
WG, Fill Wedge	6-21

Chapter 7: Labeling Your Plots

Instructions Covered	7-1
Terms You Should Understand	7-2
Using and Terminating Labels	7-2
Carriage Returns and Line Feeds	7-2
The Carriage Return Point	7-3
Variables in Labels	7-4
Default Label Settings	7-6
Understanding the Character Plot Cell	7-7
Enhancing Labels	7-9

Chapter 7: Labeling Your Plots (Continued)

Character Size and Slant	7-9
Character Spaces and Text Lines	7-9
Label Orientation and Placement	7-9
BL, Buffer Label	7-11
CP, Character Plot	7-13
DI, Direction Absolute	7-16
DR, Relative Direction	7-22
DT, Define Label Terminator	7-27
DV, Direction Vertical	7-29
ES, Extra Space	7-31
LB, Label	7-33
LO, Label Origin	7-35
PB, Print Buffered Label	7-38
SI, Absolute Character Size	7-39
SL, Character Slant	7-42
SR, Relative Character Size	7-45

Chapter 8: Drawing Polygons and Using the Polygon Buffer

Instructions Covered	8-1
Understanding the Plotter's Buffers	8-1
I/O Buffer	8-1
Polygon Buffer	8-2
Downloadable Character Buffer	8-2
Vector Buffer	8-3
Pen Sort Buffer	8-3
Notes for Buffer Allocation	8-4
Understanding and Defining Polygons	8-6
Drawing Polygons and Subpolygons	8-7
Drawing Circles in Polygon Mode	8-7
Using the Polygon Buffer	8-9
Determining the Approximate Polygon Buffer Size	8-9
Determining the Exact Buffer Size	8-12
Changing the Size of the Polygon Buffer	8-14
EP, Edge Polygon	8-15
FP, Fill Polygon	8-17
GM, Graphics Memory	8-19
PM, Polygon Mode	8-21

Chapter 9: Changing Picture Area and Orientation

Instructions Covered	9-1
Terms You Should Understand	9-1
Windowing: Setting Up Soft-Clip Limits	9-2
Enlarging or Reducing a Picture	9-3
Drawing Equal-sized Pictures on One Page	9-4

Chapter 9: Changing Picture Area and Orientation (Continued)

Creating Mirror Images	9-5
Rotating a Picture	9-7
Using the Output Instructions in This Chapter	9-7
IW, Input Window	9-8
OH, Output Hard-Clip Limits	9-11
OP, Output PI and P2	9-12
OW, Output Window	9-13
RO, Rotate Coordinate System	9-14

Chapter 10: Advanced Pen Control and Front-Panel Interaction

Instructions Covered	10-1
Terms You Should Understand	10-1
Using the Advanced Pen Control Features	10-2
Grouping Your Pens for Longer Life	10-2
Protecting Your Program From Front-Panel Override	10-2
Programming the Front-Panel Display and Keys	10-3
AP, Automatic Pen Operations	10-4
AS, Acceleration Select	10-6
CV, Curved Line Generator	10-7
FS, Force Select	10-9
GP, Group Pen	10-11
KY, Define Key	10-13
NR, Not Ready	10-16
OK, Output Key	10-17
SG, Select Group	10-19
VS, Velocity Select	10-20
WD, Write to Display	10-22

Part III — Special Applications

Chapter 11: Digitizing

Using Digitizing Instructions	11-1
Preparing Your Plotter for Use as a Digitizer	11-1
Digitizing with the Plotter	11-2
Manual Digitizing	11-2
Monitoring the Status Byte	11-3
HP-IB Interrupts and Polling	11-5
DC, Digitize Clear	11-6
DP, Digitize Point	11-7
OD, Output Digitized Point and Pen Status	11-8

Chapter 12: Rollfeed Instructions and Long-Axis Plotting	
Instructions Covered	12-1
Terms You Should Understand	12-1
Long-Axis Plotting	12-2
X-Axis Hard-Clip Limits	12-4
Locating a P1 X-Coordinate on a Long Axis Plot	12-5
Media Absorbency	12-5
Stabilization of Paper, Vellum, and Tracing Bond	12-6
Example — Creating a Long Axis Plot	12-7
AF, Advance Full Page	12-9
AH, Advance Half Page	12-10
EC, Enable Cut Line	12-11
FR, Frame Advance	12-12
PG, Page Feed	12-13
PS, Page Size	12-14
Chapter 13: Alternate Character Sets and User-Designed Characters	
Instructions Covered	13-1
Terms You Should Understand	13-1
Using Character Sets	13-2
The Drafting Set	13-5
Differences When Plotting with Variable-Space Fonts	13-5
The Downloadable Set and User-Defined Characters	13-7
Character Selection Modes	13-8
Designating and Selecting Character Sets	13-9
Standard and Alternate Character Sets	13-9
Special Characters	13-9
Choosing Other Character Selection Modes	13-11
The In-Use Code Table	13-12
Control Characters	13-14
Fallback Mode	13-14
HP 7-Bit Compatibility Mode	13-15
HP 8-Bit Mode	13-16
ISO 7-Bit Mode	13-17
Example — Using the DS and IV Instructions in the ISO 7-Bit Mode	13-18
ISO 8-Bit Mode	13-18
Example — Embedding a Single-Shift in a Label String	13-20
CA, Designate Alternate Character Set	13-21
CC, Character Chord Angle	13-23
CM, Character Selection Mode	13-25
CS, Designate Standard Character Set	13-27
DL, Define Downloadable Character	13-28
How to Define a Downloadable Character	13-29
DS, Designate Character Set into Slot	13-32
IV, Invoke Character Slot	13-34

Chapter 13: Alternate Character Sets and User-Designed Characters (Continued)

SA, Select Alternate Character Set	13-36
SS, Select Standard Character Set	13-37
UC, User-Defined Character	13-38

Part IV — Interfacing, Handshaking, Output, and Device Control

Chapter 14: Obtaining Information from the Plotter

Instructions Covered	14-1
Notes for Obtaining Plotter Output	14-2
Hints for the HP-IB (IEEE-488) Configuration	14-2
Hints for the RS-232-C Configuration	14-3
Using Output Instructions	14-3
Identifying the Pen Location and Position	14-3
Identifying the Plotter and Its Functions	14-4
Obtaining Error Information	14-4
Obtaining Status Byte Information	14-5
Summary of Output Responses	14-5
Polling	14-7
Serial Polling	14-7
Parallel Polling	14-7
GC, Group Count	14-9
IM, Input Mask	14-10
OA, Output Actual Pen Status	14-13
OC, Output Commanded Pen Status	14-14
OE, Output Error	14-15
OF, Output Factors	14-17
OG, Output Group Count	14-18
OI, Output Identification	14-19
OL, Output Label Length	14-20
OO, Output Options	14-22
OS, Output Status	14-23
OT, Output Carousel Type	14-25

Chapter 15: Device-Control Instructions

Instructions Covered	15-1
Understanding Device-Control Instructions	15-2
Sending Device-Control Instructions	15-2
Syntax for Device Control Instructions	15-3
Omitting Optional Parameters	15-3

Chapter 15: Device-Control Instructions (Continued)

Using Device-Control Instructions	15-4
Setting the Plotter to Known Conditions.....	15-4
Identifying the Plotter	15-5
Monitoring and Changing the Buffer Sizes	15-5
Verifying Errors or Operating Conditions	15-5
ESC.A, Output Identification	15-7
ESC.B, Output Buffer Space	15-8
ESC.E, Output Extended Error	15-9
ESC.J, Abort Device-Control.....	15-11
ESC.K, Abort Graphics	15-12
ESC.L, Output Buffer Size When Empty	15-13
ESC.O, Output Extended Status	15-14
ESC.Q, Set Monitor Mode.....	15-17
ESC.R, Reset	15-19
ESC.S, Output Configurable Memory Size.....	15-20
ESC.T, Allocate Configurable Memory.....	15-21
ESC.U, End Flush Mode	15-24
ESC.Y or ESC.(, Plotter On	15-25
ESC.Z or ESC.), Plotter Off.....	15-26
ESC.@", Set Plotter Configuration	15-27

Chapter 16: Interfacing and Handshaking

The Hewlett-Packard Interface Bus (HP-IB).....	16-1
Reactions to Bus Commands DCL and SDC.....	16-5
RS-232-C Interfacing and Handshaking	16-5
Choosing a Handshake	16-6
Handshaking Through Device-Control Instructions	16-7
Hardwire Handshake	16-7
Xon-Xoff Handshake	16-9
Enquire/Acknowledge Handshake.....	16-12
Software Checking Handshake.....	16-16
Data Transmission Modes	16-19
Normal Mode	16-19
Block Mode	16-19
Automatic Modem Disconnection Modes.....	16-21
Switched/Datex-line Disconnect Mode	16-21
Leased-Line Disconnect Mode	16-21
ESC.H, Set Handshake Mode 1.....	16-22
ESC.I, Set Handshake Mode 2	16-23
ESC.M, Set Output Mode	16-25
ESC.N, Set Extended Output and Handshake Mode	16-28
ESC.P, Set Handshake Mode	16-29

Appendix A: Error Messages

No Operation (NOP) Instructions	A-2
---------------------------------------	-----

Appendix B: Reference Material

ASCII Character Codes	B-1
ASCII Control Codes	B-2
Character Sets	B-5

Appendix C: Sample Plots and Program Listings

Example 1: Scaling and P1/P2 Locations	C-2
Example 2: Moving a Scaled P1	C-6
Example 3: Using a Window	C-9
Example 4: Rotating a Plot	C-12
Example 5: FORTRAN and Pascal—AR Instruction	C-14
Example 6: FORTRAN and Pascal—DI and LB Instructions	C-16

Appendix D: Using Kanji

Accessing the Kanji Character Set	D-2
Using Kanji Characters in Symbol Mode	D-5
Terminating Kanji Labels	D-5
The Kanji Character Set	D-6

Glossary**Subject Index**

HP-GL Instructions

AA, Arc Absolute	6-7
AF, Advance Full Page	12-9
AH, Advance Half Page	12-10
AP, Automatic Pen Operations	10-4
AR, Arc Relative	6-9
AS, Acceleration Select	10-6
BL, Buffer Label	7-11
CA, Designate Alternate Character Set	13-21
CC, Character Chord Angle	13-23
CI, Circle	6-11
CM, Character Selection Mode	13-25
CP, Character Plot	7-13
CS, Designate Standard Character Set	13-27
CT, Chord Tolerance	6-15
CV, Curved Line Generator	10-7
DC, Digitize Clear	11-6
DF, Default	3-14
DI, Absolute Direction	7-16
DL, Define Downloadable Character	13-28
DP, Digitize Point	11-7
DR, Relative Direction	7-22
DS, Designate Character Set into Slot	13-32
DT, Define Label Terminator	7-27
DV, Direction Vertical	7-29
EA, Edge Rectangle Absolute	4-7
EC, Enable Cut Line	12-11
EP, Edge Polygon	8-15
ER, Edge Rectangle Relative	4-9
ES, Extra Space	7-31
EW, Edge Wedge	6-17
FP, Fill Polygon	8-17
FR, Advance Frame	12-12
FS, Force Select	10-9
FT, Fill Type	5-6
GC, Group Count	14-9
GM, Graphics Memory	8-19
GP, Group Pen	10-11
IM, Input Mask	14-10
IN, Initialize	3-17
IP, Input P1 and P2	3-19
IV, Invoke Character Slot	13-34
IW, Input Window	9-8
KY, Define Key	10-13
LB, Label	7-33

HP-GL Instructions (Continued)

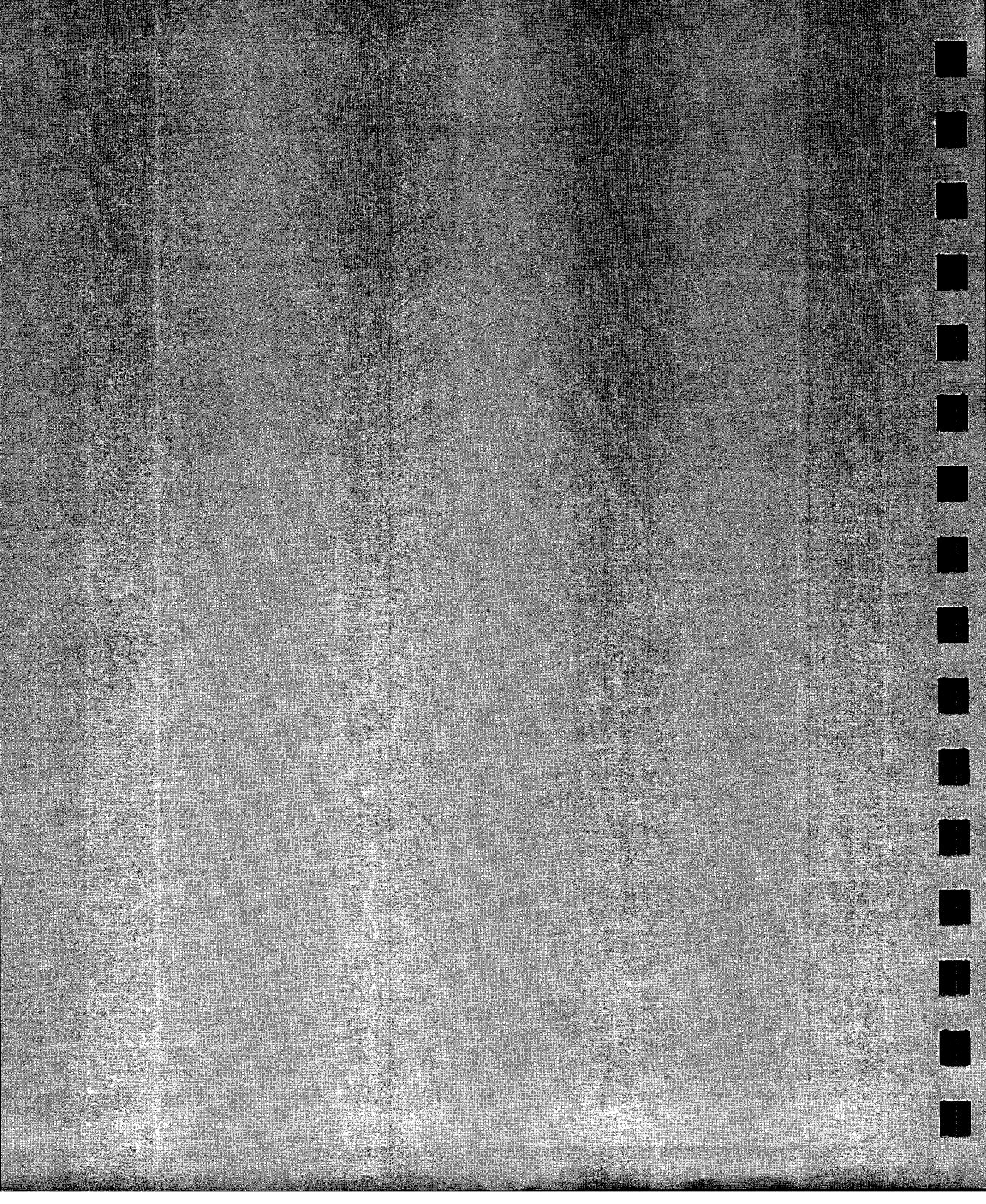
LO, Label Origin	7-35
LT, Line Type	5-9
NR, Not Ready	10-16
OA, Output Actual Pen Status	14-13
OC, Output Commanded Pen Status	14-14
OD, Output Digitized Point and Pen Status	11-8
OE, Output Error	14-15
OF, Output Factors	14-17
OG, Output Group Count	14-18
OH, Output Hard-Clip Limits	9-11
OI, Output Identification	14-19
OK, Output Key	10-17
OL, Output Label Length	14-20
OO, Output Options	14-22
OP, Output P1 and P2	9-12
OS, Output Status	14-23
OT, Output Carousel Type	14-25
OW, Output Window	9-13
PA, Plot Absolute	4-11
PB, Print Buffered Label	7-38
PD, Pen Down	4-12
PG, Page Feed	12-13
PM, Polygon Mode	8-21
PR, Plot Relative	4-14
PS, Page Size	12-14
PT, Pen Thickness	5-12
PU, Pen Up	4-16
RA, Fill Rectangle Absolute	5-14
RO, Rotate Coordinate System	9-14
RR, Fill Rectangle Relative	5-16
SA, Select Alternate Character Set	13-36
SC, Scale	3-21
SG, Select Pen Group	10-19
SI, Absolute Character Size	7-39
SL, Character Slant	7-42
SM, Symbol Mode	5-18
SP, Select Pen	4-17
SR, Relative Character Size	7-45
SS, Select Standard Character Set	13-37
TL, Tick Length	5-20
UC, User-Defined Character	13-38
UF, User-Defined Fill Type	5-23
VS, Velocity Select	10-20
WD, Write To Display	10-22

HP-GL Instructions (Continued)

WG, Fill Wedge	6-21
XT, X-Tick	5-28
YT, Y-Tick	5-29



Notes



Programming Concepts

Before you can write graphics programs for your plotter, you need to be familiar with your computer and a programming language that the computer and plotter understand. Use any ASCII*-based computer and language that outputs literal strings and uses input and output statements to transfer information to and from a peripheral device. BASIC, FORTRAN, and Pascal are three commonly used languages. This manual assumes that you already have some programming experience in the language you select.

This chapter discusses the following fundamental programming concepts necessary to begin plotting.

- HP-GL and Device-Control Instructions
- using HP-GL with BASIC, FORTRAN, and Pascal
- using the programming examples
- hints for the novice programmer

If you are programming in a language other than BASIC, FORTRAN, or Pascal, you will have to make an extra effort to identify input, output, and program statements that perform the same function as the BASIC statements used in this manual. Refer to your computer documentation for information about sending literal strings to a peripheral (in this case, literal strings of HP-GL to the plotter).

Terms You Should Understand

Default	Mnemonic
Input	Output
Literal String	Syntax

*American Standard Code for Information Interchange

HP-GL and Device-Control Instructions

The plotter uses special graphics instructions to draw pictures. These instructions are known as HP-GL (Hewlett-Packard Graphics Language). In addition, the plotter uses device-control instructions (primarily with the RS-232-C interface) to control certain plotter functions.

HP-GL Instructions

Each instruction is a two-letter mnemonic code designed to remind you of its function. For example, IN is the INitialize instruction, SP is the Select Pen instruction, and PD is the Pen Down instruction. Parameters (numbers following some of the HP-GL mnemonics) tell the plotter to complete (execute) the instruction in a particular way. A typical HP-GL instruction looks like this:

SP1;

The mnemonic tells the plotter to select a pen; the parameter specifies pen number one in the carousel; and the semicolon terminates the instruction.

HP-GL instructions can be divided into two categories: graphics instructions and output instructions.

The graphics instructions are the largest category. Using graphics instructions, you can do the following things.

- Define your plotting area and select certain settings—such as default parameters, plot rotation, the pen to use, and the size of the plot.
- Define and draw your plot—including shapes such as lines, arcs, circles, polygons, rectangles, and alphabetic characters (labels). You can draw many shapes with a single instruction. For example, with one instruction you can draw a circle of any radius, an arc of any degree, or any-sized wedge of a circle. Sometimes you have a choice of how you want a particular shape to appear. That is, you can draw a rectangle by filling in a rectangular shape or by outlining it. These shapes and features are inherent in your plotter, ready for you to access.

Output instructions enable you to obtain information from the plotter, such as:

- error messages
- pen location and position
- digitized points
- current size of plotting area
- current positions of P1 and P2
- label length
- plotter identification

Device-Control Instructions

Device-control instructions are used primarily with an RS-232-C interface. The HP-IB interface has limited device control. Each device-control instruction is an escape sequence, such as:

ESC . B

ESC represents the non-printing ASCII character **ESC** and can be accessed using the **ESC** key on your keyboard or a programming function such as the BASIC *CHR\$(27)* string function. Device-control instructions perform advanced operations, such as:

- changing and monitoring buffer sizes
- handshaking
- verifying plotter operating conditions

This manual gives full details on the capabilities of the plotter's HP-GL and device-control instruction set. To communicate with the plotter, you must insert the instructions with their parameters in an appropriate output statement from your computer's language. This is referred to as 'sending' an instruction to the plotter.

Using HP-GL with BASIC, FORTRAN, and Pascal

Before you can send instructions to the plotter, your computer and plotter must communicate; refer to the User's Guide for interconnection instructions.

How you send an instruction to your plotter depends on the programming language you are using. Some languages include some graphics statements, but these are used primarily for the computer's monitor. Three common languages used for plotting are BASIC, FORTRAN, and Pascal.

BASIC (Beginner's All-purpose Symbolic Instruction Code) is one of the most common languages used for plotting. It uses statements resembling English to perform many complex operations. Some graphics statements may be included in your implementation of BASIC but they are usually designed for the CRT, not your plotter.

FORTRAN (FORmula TRANslator) is a problem-oriented language. The programmer can think in terms of the problem and write programs as algebraic expressions and arithmetic statements.

Pascal is a block-structured language that requires structured programming, but still allows many operator and control statements.

Follow these steps to send an instruction in your language.

1. Use whatever opening statements are required to define the computer's output port and establish the plotter as the recipient of an output string. These are called configuration statements and usually must be the first statements in your program. Configuration statements are explained in detail in the next section.
2. Send the string to the plotter using an output statement. Output statements are explained after configuration statements.

Configuration Statements

The configuration statement directs computer input and/or output to the plotter. It will be specific to your computer language and the type of interface (RS-232-C or HP-IB) you are using. You must use a configuration statement at the beginning of every program; for example,

```
10 OPEN "COM1:9600,N,8,1,RS,CS65535,DS,CD" AS #1
```

The BASIC configuration statement shown above is for the HP Vectra (also IBM PC, XT, and AT) RS-232-C interface. It **OPENS** **COM1** (port 1) for output at **9600** baud, **No parity**, **8** data bits, **1** stop bit, suppresses RTS (**RS**), waits 65535 milliseconds before

returning a Device timeout error (**CS65535**), ignores the DSR and Carrier Detect lines (**DS,CD**) and calls the file #1.

Your configuration statement will vary according to the programming language and the hardware configuration of your computer. Some sample configuration statements are listed below. Also refer to Chapter 8 of the User's Guide.

HP 3000 DEC VAX (FORTRAN, RS-232-C)	<pre>WRITE (6,10) 10 FORMAT ("<i>instruction</i>")</pre> <p>(The WRITE statement also functions as a configuration statement. Unless you specify otherwise, the instruction is automatically directed to the standard output device, which is typically your terminal. When your plotter is in an eavesdrop configuration, it executes these instructions as they go to the terminal.)</p>
HP 9000, Series 300 (HP BASIC, HP-IB)	<pre>OUTPUT 205 ; ("<i>instruction</i>")</pre> <p>(In this case the output statement also functions as a configuration statement, directing output to the plotter at select code 7, HP-IB address 5.)</p>
(PASCAL, HP-IB)	<pre>WRITELN ('<i>instruction</i>')</pre> <p>(The WRITELN statement also functions as a configuration statement. Unless you specify otherwise, the instruction is automatically directed to a file called 'output'.)</p>
HP Touchscreen (RS-232-C and HP-IB) (Microsoft® BASIC*)	<pre>10 OPEN **0'',1,**PLT''</pre>
(GW™ BASIC*)	<pre>10 OPEN **LPT3:** FOR OUTPUT AS #1</pre>
HP Vectra IBM PC, XT, AT AT&T PC 6300 (RS-232-C) (Microsoft BASIC)	<pre>10 OPEN **COM1:9600,N,8,1,RS,CS65535,DS,CD** AS #1</pre>

If your computer is not listed, refer to Chapter 7 of the User's Guide to learn how to make your computer and plotter communicate.

*GW™ BASIC and Microsoft® are trademarks of Microsoft Corporation.

Output Statements

After the configuration statement, use output statements to send the instruction to the plotter as a string. Output statements vary with the computer and language. To give you an idea of the variety of statements available, the following table lists a few computers and languages with their associated output statements for the HP-GL instruction (*SP1;*).

Output Statements for SP1;

Computer	Language	Example
HP Touchscreen HP Vectra IBM PC/PC-XT/AT	GW BASIC	PRINT #1, "SP1;"
HP 9000, Series 300 (HP-IB Interface)	PASCAL	WRITELN("SP1;")
	HP BASIC	OUTPUT 205; "SP1;"
HP 3000 DEC VAX	FORTRAN	WRITE(6,10) 10 FORMAT(X,4HSP1;)

Input Statements

The computer language output statements we discussed in the last section output information to the plotter. To get information from the plotter, you must send an HP-GL output statement to cause the plotter to output information back to the computer.

You then must use a computer language input statement to read the plotter's output. The following table lists a few computers and languages with their associated input statements for the HP-GL instruction (*OP;*).

Input Statements for OP;

Computer	Language	Example
HP Touchscreen Computer HP Vectra IBM PC/PC-XT/AT	GW BASIC	INPUT #1, A,B,C,D
HP 9000, Series 300 (HP-IB Interface)	PASCAL	READ(A,B,C,D);
	HP BASIC	ENTER 205; A,B,C,D
HP 3000 DEC VAX	FORTRAN	READ(5,40) 40 FORMAT(4I2)

Number Formats

The plotter accepts numbers from $-8\ 388\ 608$ (-2^{23}) to $8\ 388\ 607$ ($2^{23}-1$). However, the plotter does not understand exponential format (i.e., $6.03E8$). If you are using a computer or language that uses exponential format, you must use integer variables or a formatting technique to round the number to something smaller.

Using the Programming Examples

Examples are presented as complete programs written in a version of GW BASIC for the MS™-DOS* 2.11 (or higher) operating system. This version of BASIC is used by many popular personal computers. The examples show the use of the instruction being explained as well as its interaction with other instructions and the BASIC language.

If you are using GW BASIC, replace line 10 with the proper configuration statement for your computer; then, you can enter and run the program examples as shown.

If you are not using GW BASIC, you may need to change some of the program statements as well as insert the proper configuration statement at the beginning of the program.

Below is a simple example of the way complete programs appear in this manual.

```
10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;PA4000,1000;"
30  PRINT #1, "PD;AA2000,1000,45,25;"
40  PRINT #1, "PU3050,2060;PD;AA2000,1000,-45,10;"
50  PRINT #1, "PU3000,1000;PD;AA2000,1000,45;"
60  PRINT #1, "SP0;"
70  END
```

*MS™-DOS is a trademark of Microsoft, Inc.

Programming Statements You May Need to Change

You might need to change BASIC statements if you are not using the GW BASIC version used here, or if you are programming in another language. Your computer language documentation should tell you how to do this. Here are the statements that you will likely need to change:

- PRINT #1** This statement sends the HP-GL instructions, via an output file, to the plotter. Your computer might use a statement such as WRITE, WRITELN, OUTPUT, LPRINT, or simply PRINT. Here "1" corresponds to the file number in the "OPEN" statement.
- INPUT #1** This statement causes information from the plotter to be read by the computer. Your computer might use a statement such as READ, READLN, or ENTER. Here "1" corresponds to the file number in the "OPEN" statement.
- FOR . . . NEXT X=3.14** FOR . . . NEXT are loop statements. X=3.14 is a variable assignment. Change these statements to whatever your language uses.

Hints for the Novice Programmer

If you are an inexperienced HP-GL programmer, use the following guidelines to ensure success.

- Know your equipment; this includes the computer and plotter. Know how to write, edit, save, and run a program. Determine how your computer sends (outputs) information to and reads (inputs) information from peripheral devices.
- Know your computer programming language. Even if you aren't programming in BASIC, get a BASIC book and look up unfamiliar statements. Their definitions will help you find and use the equivalent statement in your language. Always use the syntax your language requires. Notice the syntax differences between the two versions of BASIC represented in the tables under *Output Statements* and *Input Statements* earlier in this chapter.

- Enter HP-GL and device-control instructions exactly as they appear in the text. In a program, there is often only one "right way" to say something. Every letter, symbol, and number is significant. Common mistakes are substituting the letter oh "O" for the number zero "0" or a lowercase el "l" for the number one "1". Substituting a semicolon for a comma or omitting a quotation mark can make the difference between success and failure.
- Begin every program with the proper configuration statement and an IN or DF instruction.

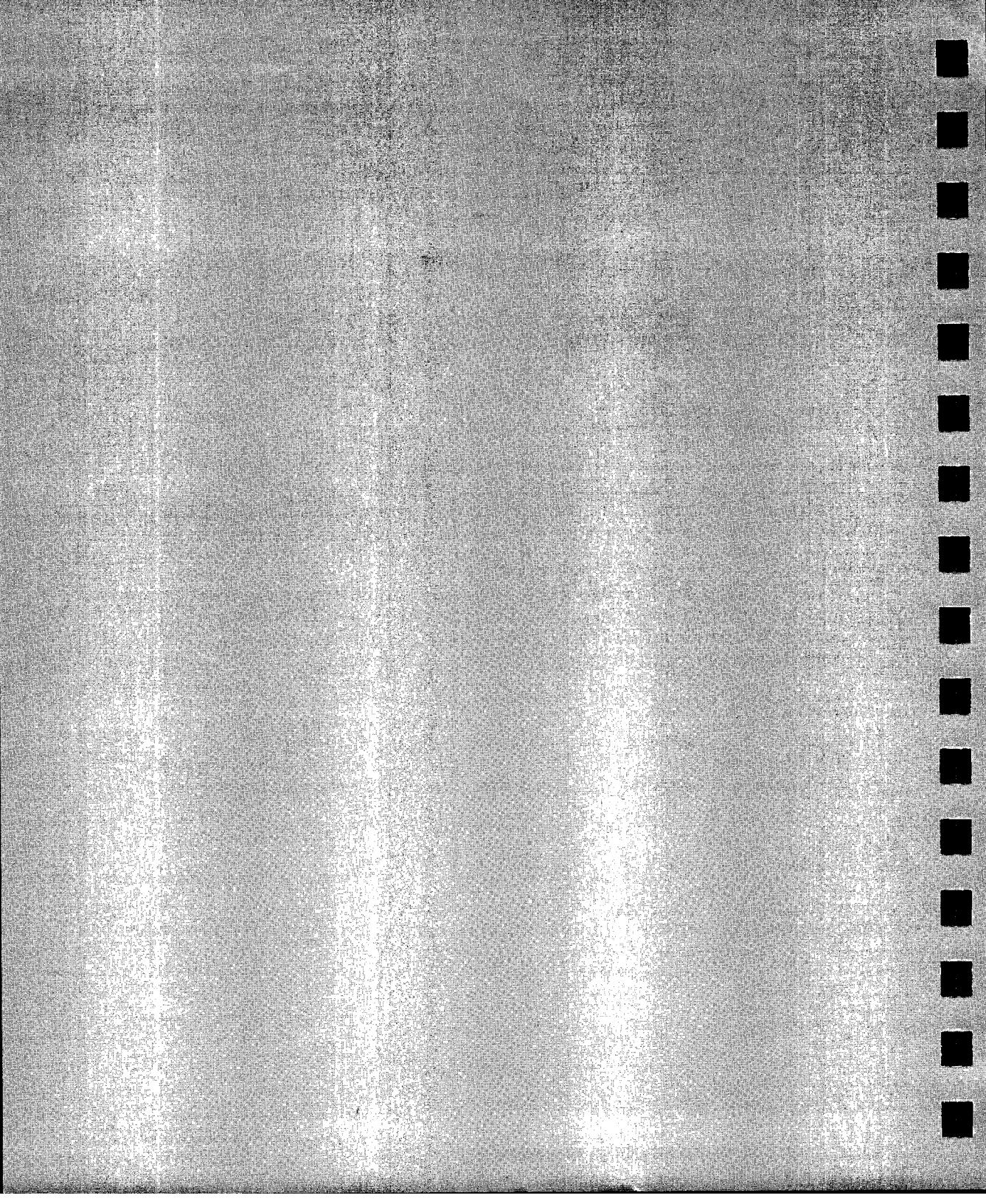
To learn HP-GL, study the basic instructions in Part 1 and run the examples. Then try combining the instructions in simple programs of your own before moving on to Part 2.

Program Errors

If your program contains syntax or parameter errors (e.g., a comma instead of semicolon; number too big or small), the plotter will display an error number and message on the front panel. The message will display until you press **Next Display**, use **Reset**, send the IN or DF instruction, or use the OE instruction to output the error to your computer.

Each HP-GL instruction includes a table of the errors and their causes that typically occur with that instruction. Appendix A also includes a complete table of error numbers and messages. Note that the plotter may or may not complete your plot correctly, depending on the severity of the error.

You can also get unexpected or incorrect plots without receiving an error message. This usually results from inaccurate use of an instruction. Also some instructions work together with another instruction, and you must use both for your plot to be correct. The description of each instruction includes a list of related instructions, which may or may not be necessary to your program. Refer to these lists when troubleshooting a program.



Plotting and Graphics Concepts

In order to use the graphics instructions, you must specify a location on the paper. This chapter gives an overview of some of the graphics capabilities of the plotter and discusses the following fundamental concepts necessary to begin plotting.

- the coordinate system
- units of measure
- graphic limits
- scaling

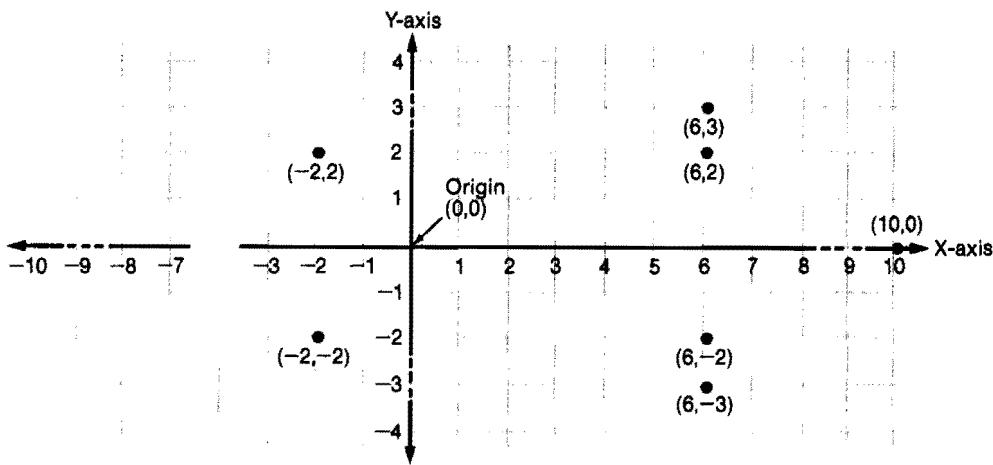
You will find this chapter useful if you have never done any graphics programming or if you have written programs for a graphics terminal but not a plotter. If you already know these concepts, go on to Chapter 3.

Plotting with the Coordinate System

The plotter uses the Cartesian coordinate system: a set of two perpendicular scaled axes, usually called the X and Y axes, in which any point on the XY plane is located by specifying its X and Y coordinates. You can visualize the system as a grid. By convention the X-axis (abscissa) is parallel to the long side of the paper and the Y-axis (ordinate) is parallel to the short side.

Refer to the illustration below. The intersection of the axes is called the *origin* of the system and has a location of $(0, 0)$. Positive X values are plotted to the right of the origin, and positive Y values are plotted above the origin.

To locate any point in the plotting area, move from the origin so many units along the X-axis, then so many units along the Y-axis to your point. Each point, then, is referenced by the combination of its X-coordinate and Y-coordinate, known as an X,Y coordinate pair. To specify a position, you must always specify a complete X,Y coordinate pair, X-coordinate first and Y-coordinate second. The manual shows the coordinate pair in parentheses (X,Y) for clarity. Do not use the parentheses when programming.



Look at the illustration again to locate these points: $(0, 0)$; $(-2, 2)$; $(6, 2)$; $(6, 3)$; $(10, 0)$; $(6, -3)$; $(6, -2)$; $(-2, -2)$; $(0, 0)$. Now draw a straight line between each point in the order listed. (You should have drawn an arrow.) This is how you define a picture on a two-dimensional coordinate system.

Units of Measure and Scaling

You can measure along the axes and express coordinates using two types of units:

- plotter units
- user units

The plotter always moves in plotter units, which are a fixed size. However, when you define and draw your application, you may find that another unit makes more sense. You can define a unit of measure called a user unit. The plotter internally converts user units to plotter units before locating the coordinates on paper.

Plotter Units

A plotter unit is the smallest move you can tell the plotter to make (this is also known as the addressable resolution of the plotter). Note the following measurements pertaining to plotter units.

$$1 \text{ plotter unit} = 0.025 \text{ mm or } 0.00098 \text{ in.}$$

$$40 \text{ plotter units} = 1 \text{ mm}$$

$$1016 \text{ plotter units} = 1 \text{ inch}$$

Because plotter units are the smallest move the plotter can make, it interprets all X,Y coordinates for plotter units as integers (whole numbers).

User Units

User units allow you to customize your coordinate system to your particular needs. For example, you might want to make a plot showing maximum temperatures along the Y-axis and the days of the month along the X-axis. User units can represent months, years, dollars, francs, distances, temperatures, population, or whatever meets your requirements.

In addition, user units are flexible. You can assign the user-unit origin (0,0) to be anywhere on the paper. You can also specify coordinates with fractions, making it easier to directly represent your data. You establish user units using a process known as *scaling*.

Scaling

When you scale a plot, you specify the number of user units in each axis, and the plotter automatically converts one user unit to a suitable number of plotter units. In effect, you are assigning a scale (or ratio) of plotter units to user units. For example, since 400 plotter units equals 1 centimetre, you can establish this scaling to plot in user units of centimetres. This is similar to scaling on maps where one inch (a user unit) represents a specific number of miles.

You can specify units that are the same size on both the X-axis and the Y-axis (isotropic scaling), or you can specify units that are different sizes on each axis (anisotropic scaling). In addition each axis can have a different number of units. When you scale a plot, you can also move the origin to the corner so that all coordinates are positive.

The specifics of scaling are discussed in Chapter 4.

Graphic Limits

When you scale your plot, you set up your 'grid' within the graphics limits of the plotter. The plotter recognizes two types of graphics limits:

- the hard-clip limits
- the soft-clip limits

The word 'clip' indicates that a portion of the paper is selected as the boundary; that is, any drawing that would extend beyond the limits is clipped (and therefore not drawn), as you would clip a newspaper article. *Hard-clip* refers to a physical boundary, beyond which a pen cannot move. *Soft-clip* refers to a programmatically set boundary that restricts pen movement to within its borders. When the plotter is first turned on, the hard-clip and soft-clip limits are identical.

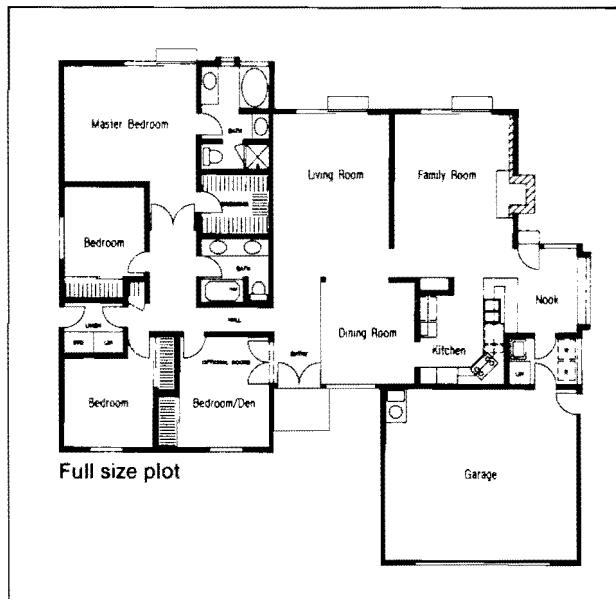
Hard-Clip Limits

The hard-clip limits represent the physical boundary for pen movement. When the plotter senses the paper size, it automatically sets the hard-clip limits a few millimetres inside of the pinch wheels. This allows room for the pinch wheels to move without rolling over plotted lines and possibly smearing ink. The hard-clip limits correspond to the maximum plotting area.

Soft-Clip Limits

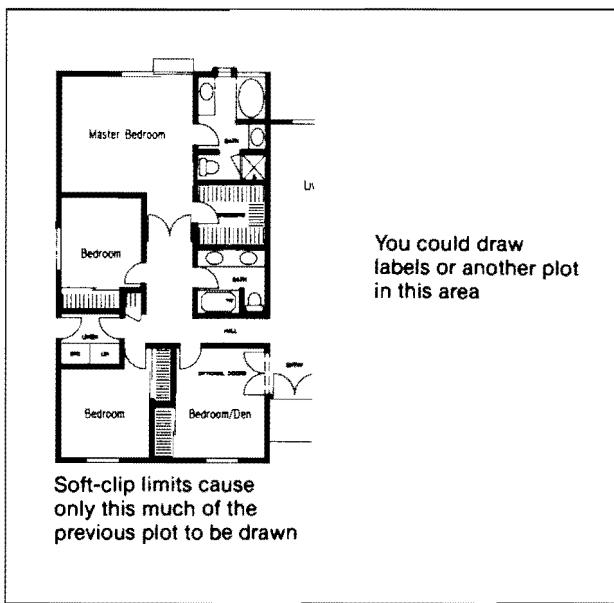
Soft-clip limits temporarily restrict pen movement to a specified area of the page. Because the soft-clip limits allow you to draw attention to a particular set of data they are often referred to as *windows*. Usually, you will use soft-clip limits to ensure that nothing will be drawn beyond that portion of the page.

For example, look at the following floor plan.



After plotting the full floor plan, suppose that you decide to make a new plot showing only the sleeping areas and use the space on the right for text.

To create the new plot, use the program for the full floor plan, adding soft-clip limits to restrict plotting to the left half of the plotting area. Finally, at the end of the program, add program lines to extend the soft-clip limits and plot the text.



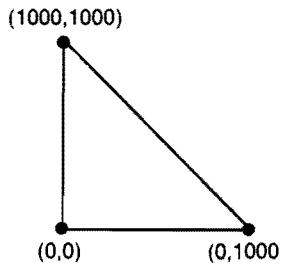
Using soft-clip limits with user-unit scaling provides a great deal of flexibility with respect to where you plot data, how much of your data is plotted, and reducing/enlarging plots. With the above example, you could set the soft-clip limits for a portion of the plot, but then draw that portion on the full plotting area, thus allowing a closer look at the detail. The specifics of using soft-clip limits are in Chapter 9.

Graphics Concepts

This section introduces you to the variety of your plotter's graphics capabilities. If you are unfamiliar with plotter graphics, use this section to stimulate your imagination. Each example refers you to the chapter containing the instructions to create that type of shape.

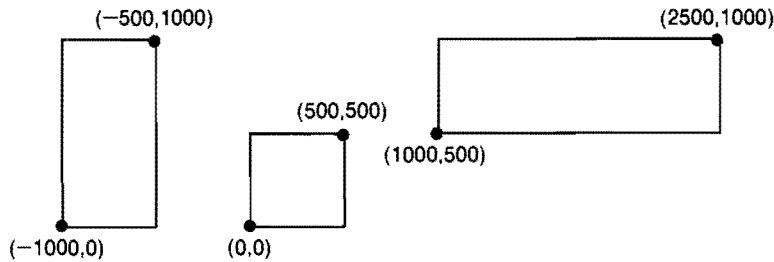
Lines

You can draw lines between any two points (X,Y coordinate pairs). If one or both points falls outside of the plotting area (soft- or hard-clip limits), the plotter will draw only the portion of the line that falls within the plotting area. Refer to Chapter 4.



Rectangles

It is easy to draw rectangles with the plotter. You simply give it the coordinate pairs of opposite corners, and it will draw the encompassing rectangle. Refer to Chapter 4.

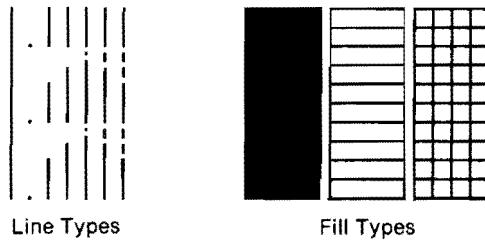


Plot Enhancements

Use plot enhancements to emphasize and clarify different parts of your plot.

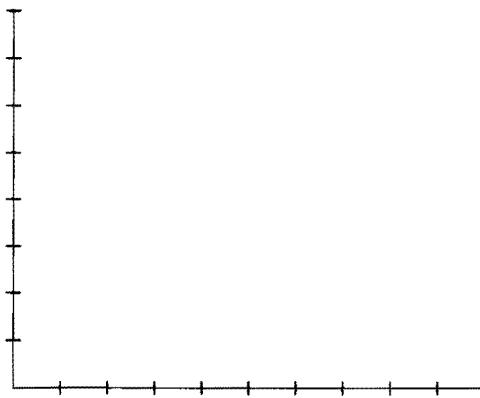
Line and Fill Types

To give variety to your plots, you can use different line and fill types when drawing. The plotter uses line types when drawing lines and fill types when filling in shapes. Refer to Chapter 5.



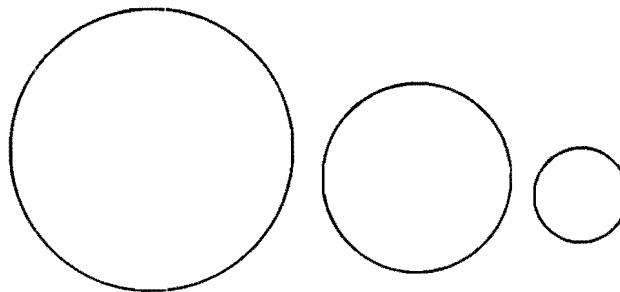
Tick Marks

When you draw graphs, you can have the plotter draw tick marks to show your units. Refer to Chapter 5.



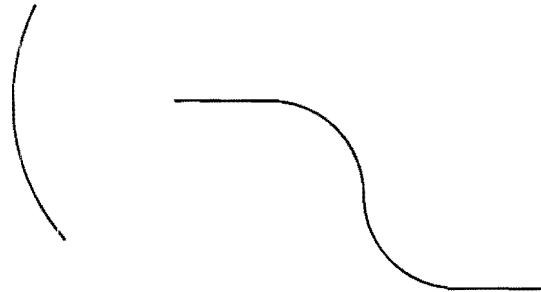
Circles

Circles can be drawn using a single instruction whose parameter is the radius of the circle. Refer to Chapter 6.



Arcs

The plotter draws arcs (or curved lines) as portions of circles. For example, a sine wave consists of a series of arcs and straight lines. Refer to Chapter 6.



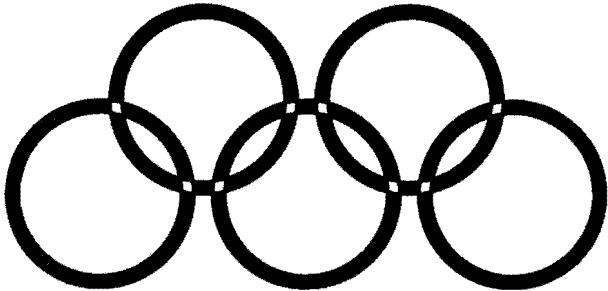
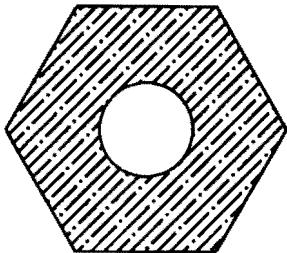
Wedges

Wedges are also portions of circles and can be outlined (edged) or solid (filled). They are frequently used in pie charts. Refer to Chapter 6.



Polygons

Polygons are multi-sided figures. Two of the simplest polygons are circles and rectangles. You can define a polygon to create a more complex figure than those inherent in the plotter. Refer to Chapter 8.



Labels

You will use labels on many of your plots. For example, you can use labels for data points, legends, and text. The plotter provides a number of alternate character sets so you can use mathematical symbols or plot in languages other than English. Refer to Chapters 7 and 13 and Appendices B and D.

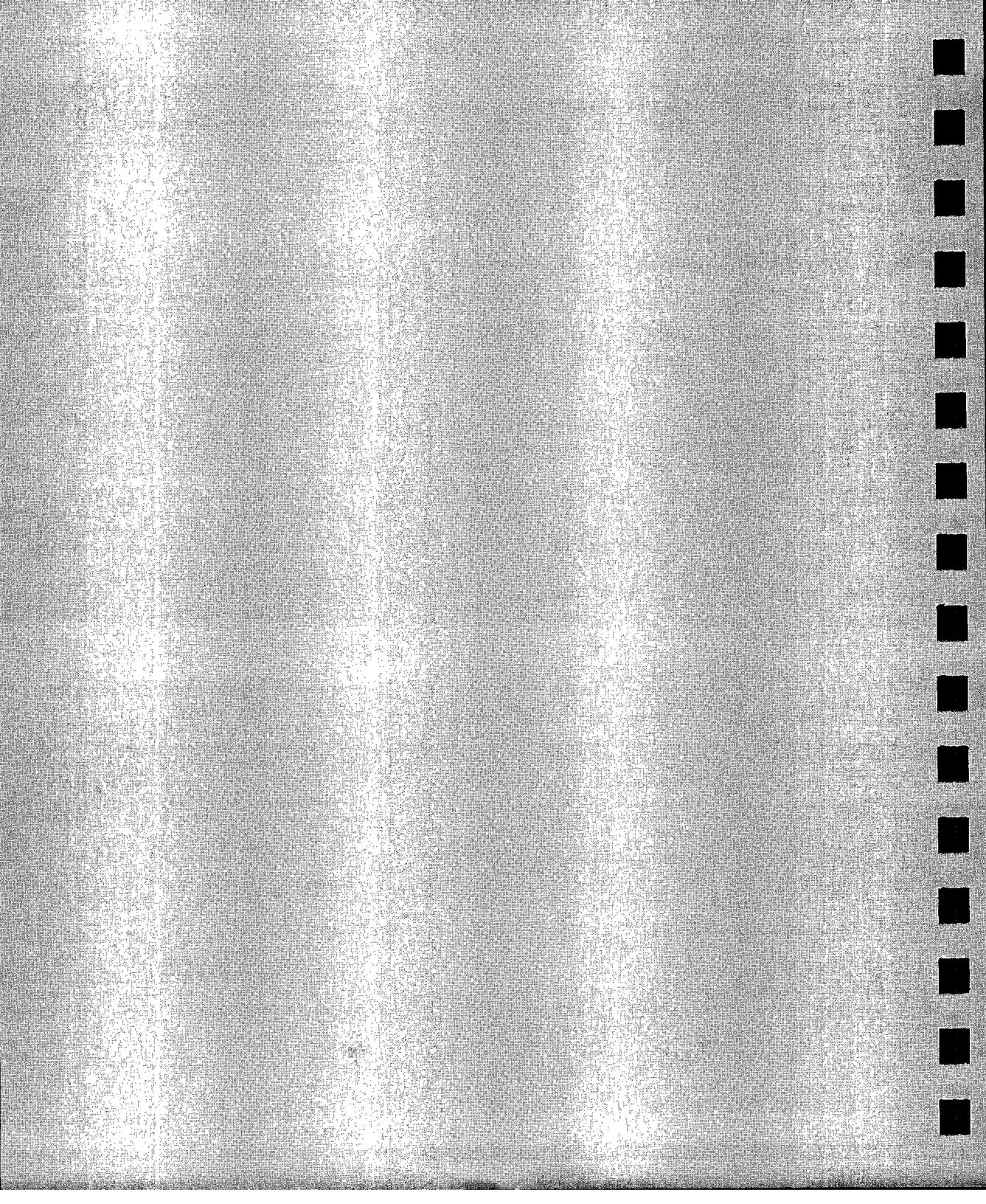
You can label

your plots

in different sizes

E
v
e
n

V
e
r
t
i
c
a
l
l
y



3

3

Beginning Your HP-GL Program

At the beginning of each program, you must set your plotter to known conditions, establish the size of your plot (graphics limits), and decide whether to use plotter units or user units. This chapter contains guidelines and instructions necessary to begin programming.

Instructions Covered

DF, Default

IN, Initialize

IP, Input P1 and P2

SC, Scale

Terms You Should Understand

Default

Mnemonic

Plotter Units

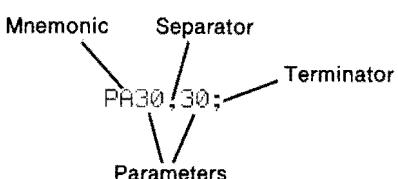
Scaling

Syntax

User Units

Understanding HP-GL Syntax

Each HP-GL instruction consists of a two-letter mnemonic, a parameter(s), and a terminator. The HP-GL instruction begins with a two-letter mnemonic (upper- or lowercase) designed to remind you of its function. Some instructions have no parameters; in others, parameters are optional. If you use parameters, you must separate them from each other with at least one *comma* or *space*, or with a + or - sign, which may be preceded by commas or spaces. All instructions require a terminator. In its basic form, an HP-GL instruction looks like this.



In both HP-IB and RS-232-C interface configurations, HP-GL instructions are terminated by a semicolon or the first letter of the next mnemonic. In the HP-IB configuration, HP-GL instructions are also terminated by a line feed (**LF**).

The label and buffer label instructions (LB and BL) and the write-to-display instruction (WD) are special cases: they must be terminated with the label terminator. The label terminator defaults to the ASCII end-of-text character, **ETX** (decimal code 3), but may be changed from its default value using the define terminator instruction (DT).

Only the terminator and the separators between parameters are required. You can insert separators between the letters of the mnemonic and before parameters, but it isn't good programming practice. Extra separators will make your programs difficult to read and can affect compatibility with other plotters.

The syntax examples in this manual show the *recommended* programming practice. Mnemonics are uppercase and do not have separators between the letters or before the parameters. Also, parameters are separated by commas instead of spaces. (Commas are recommended because some computers eliminate spaces, especially when sending variables. Commas are also necessary for complete compatibility with earlier plotters.)

To illustrate the flexibility of the syntax, the following two-instruction sequence is presented in three ways. Spaces are shown between the mnemonic and the first parameter in example 2. Two types of terminators are used: the semicolon (;) and the first letter of the next mnemonic. Example 1 shows the recommended programming practice.

PD;PU10,20;
/ Recommended PD PU 10 20; PDP10,20;

Omitting Optional Parameters

Some instructions have optional parameters that assume a default value when they are omitted. If you omit a parameter, you must omit **all subsequent parameters in the same instruction**. (The only exception is the pen control parameter in the user-defined character (UC) instruction.)

For example, the fill type (FT) instruction has three optional parameters. One way to use all three is:

FT3,100,45;

If you were to omit the second parameter, you must also omit the third parameter. Send:

FT3;

Do **not** send:

FT3,,45;

(The plotter would interpret 45 as the second parameter.)

Parameter Formats

You must specify parameters in the format (type of units) defined for each respective HP-GL instruction (given in the parameter table accompanying the instruction's definition in this manual.) Three formats can be used:

1. **Integer** — An integer from **-8 388 608** (-2^{23}) to **8 388 607** ($2^{23}-1$). The plotter automatically rounds fractional parameters that must be integers.

If the first digit after the decimal point is ≥ 5 , positive parameters are incremented by 1 and negative parameters are decremented by 1; e.g., 1008.5 is rounded to 1009, while -1008.5 is rounded to -1009.

If the first digit after the decimal point is < 5 , the fraction is dropped; e.g., 1008.4 is rounded to 1008, while -1008.4 is rounded to -1008. If you don't specify a sign, the parameter is assumed to be positive.

Plotter units are always integer.

2. **Real** — A number where the integer portion is from **-8 388 608** (-2^{23}) to **8 388 607** ($2^{23}-1$), and the optional decimal fraction has a maximum of 10 significant digits. The decimal point may be omitted when no decimal fraction is specified. If you don't specify a sign, the parameter is assumed to be positive.

User units are real.

3. **Label** — Any sequence of characters. Refer to the label instruction (LB) for a complete description.

The term *current units* in a parameter table indicates that the format of that parameter depends on whether scaling is on or off.

When scaling is *on*, the format is interpreted as real (user units). Any fractional part of a parameter up through the eighth digit is used.

When scaling is *off*, the format is interpreted as integer (plotter units).

Notations Used in This Manual for Expressing Syntax

- MNemonic — For readability, the mnemonic is shown uppercase and separated from the parameters and/or terminator.
- parameters* — Parameters are shown in *italic*.
- () — Parameters in parentheses are optional.
- label — Any number of labeling characters.
- (,...) — Any number of the previous parameter (you must have an even number of X,Y coordinates).
- ; — Instruction terminator.
- [TERM] — The terminator sent back to your computer by the plotter at the end of the response to an output instruction. The default output terminators are: RS-232-C—a carriage return (**CR**) and HP-IB—carriage return and line feed (**CR LF**). You can change the default terminator using the device-control instruction ESC . M (set output mode instruction). Refer to Chapter 14 for more information on output instructions.

NOTE: Remember that while X,Y coordinates are shown in parentheses in text (e.g., (3 , 4) or (0 , 0)), the parentheses are not part of the syntax. Do not enter parentheses in your instructions. ■

Setting the Plotter to Standard Values

The graphics instructions perform two functions: graphics functions and physical functions. The graphics functions determine the appearance of the plot—which lines are drawn and what they look like. The physical functions determine how the lines are drawn (pen speed, etc.) as well as the location and orientation of the plot on the page.

Many of these functions have standard or ‘default’ values. After you change these values, you can return them to their default values by ‘initializing’ the plotter (returning it to its initial state). You can initialize the plotter by turning on the power switch or by using the front-panel **Reset** (refer to the User’s Guide). Both of these procedures reset both the graphics and the physical functions. You can also reset both the graphics and physical functions using the the initialize (IN) instruction. The default (DF) instruction resets only the graphics functions. We recommend that you always include one of these instructions at the beginning of your program.

- Turn the plotter off and on using the power switch to return the plotter to the settings in continuous memory. Turn it off, then hold down the **FAST** button while turning it on, to return the plotter to *factory default* settings.
- Use the DF instruction to reset graphics functions to their default values.
- Use the front-panel **Reset** or the IN instruction to reset the plotter’s graphics and physical functions to their defaults.

Refer to the IN and DF instructions at the end of this chapter for a complete list of the settings each instruction establishes.



Sizing Your Plot

As you learned in Chapter 2, the smallest move the plotter can make is called a plotter unit. Plotter units are always interpreted as integers.

$$1 \text{ plotter unit} = 0.025 \text{ mm or } 0.00098 \text{ in.}$$

$$40 \text{ plotter units} = 1 \text{ mm}$$

$$1016 \text{ plotter units} = 1 \text{ inch}$$

3

As we discussed, the range for parameters of HP-GL instructions is $-8\ 388\ 608$ to $8\ 388\ 607$ (or -2^{23} to $2^{23}-1$) plotter units. Practically however, the range of plotter units is limited to the media size you are using as these ranges would fit a sheet 209 715 by 209 715 mm (8256 by 8256 in.).

The available range of plotter units for a particular media size is determined by whether front-panel **Expand** is on or off (refer to the User's Guide, Chapter 2), the hard-clip (and soft-clip) limits, and the location of scaling points P1 and P2. You can find a list of plotter unit values for different media sizes under *Plotter Unit Values for Different Media Sizes* later in this section.

Hard-Clip Limits

The hard-clip limits represent the physical boundaries for pen movement. The hard-clip limits are designed to allow room for the pinch wheels to move without rolling over plotted lines and possibly smearing wet ink. When the plotter senses the media size, it automatically sets the hard-clip limits inside the media edge—15 mm on three sides and 39 mm on the fourth.

If **Expand** is on, they are set to 5 mm on three sides and 29 mm on the fourth. When **Expand** is on, the pinch wheels can roll over drawn lines, possibly smearing ink. You must pay careful attention to the order and position of the lines being drawn to prevent ink smearing. You should not use roll media with **Expand** on.

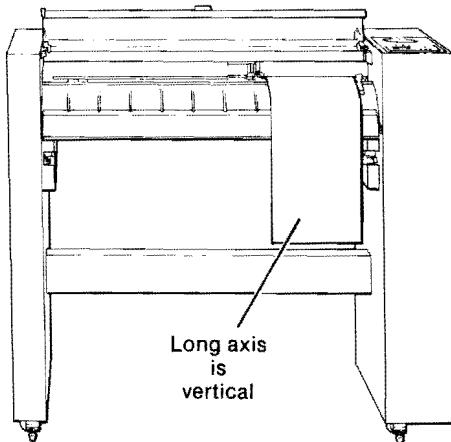
Media Loaded Vertically and Horizontally

This manual refers to media as 'loaded vertically' when the long axis of the page hangs vertically out of the plotter.

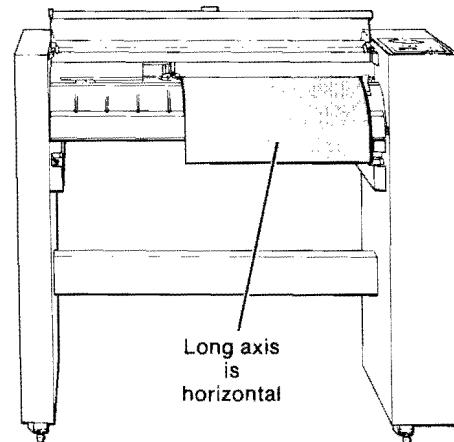
Media is 'loaded horizontally' when the long axis of the page hangs horizontally out of the plotter.

Refer to the illustrations below for examples of each way of loading media.

Media Loaded Vertically



Media Loaded Horizontally



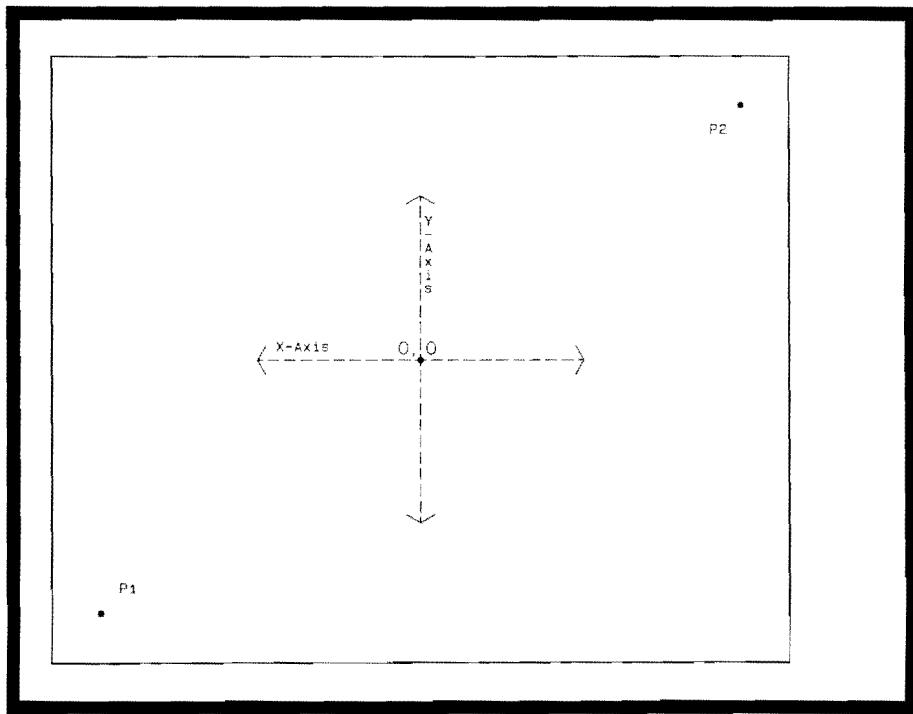
The Scaling Points P1 and P2

These two points are called scaling points because they take on the user unit values that you specify with the scaling instruction (SC). P1 and P2 are automatically set 15 mm within the hard-clip limits whenever the plotter is turned on or initialized. P1 and P2 always represent an absolute plotter-unit location on the media and define opposite corners of a rectangular plotting area. The default location of P1 is in the lower-left corner when the long edge of the media is horizontal.



The plotter unit locations of P1 and P2 represent the usual plotting range of your media. You can change the locations of P1 and P2 using the input P1 and P2 instruction (IP) or using the front panel (refer to the User's Guide). However, the hard-clip limits represent the maximum plotting range.

The illustration below shows the normal hard clip limits, the default locations of P1 and P2, and the origin.



Plotter Unit Values for Different Media Sizes

The following tables list by media size—the default plotter unit locations of P1 and P2, the maximum plotting range of the hard-clip limits, and the maximum plot size—in **Normal** (with **Expand: OFF**) and in **Expand** (with **Expand: ON**) mode.

Normal Plot Size
(Expand: OFF)

Paper Size	Default Scaling Points P1		Hard-Clip Plotting Range		Maximum Plotting Area (X and Y)
	P1 _x , P1 _y	P2 _x , P2 _y	X-axis	Y-axis	
A (horizontal)	-4348,-2598	4348,2598	±4948	±3198	247.4 × 159.9 mm (9.74 × 6.3 in.)
A (vertical)	-3868,-3078	3868,3078	±4468	±3678	223.4 × 183.9 mm (8.8 × 7.24 in.)
B (vertical)	-6916,-4348	6916,4348	±7516	±4948	375.8 × 247.9 mm (14.8 × 9.74 in.)
C (horizontal)	-9936,-6916	9936,6916	±10536	±7516	526.8 × 375.8 mm (20.74 × 14.8 in.)
D (horizontal)	-16032,-9456	16032,9456	±16632	±10056	831.6 × 502.8 mm (32.74 × 19.8 in.)
D (vertical)	-15552,-9936	15552,9936	±16152	±10536	807.6 × 526.8 mm (31.8 × 20.74 in.)
E (vertical)	-20632,-16032	20632,16032	±21232	±16632	1061.6 × 833.1 mm (41.8 × 32.8 in.)
A4 (horizontal)	-4700,-2480	4700,2480	±5300	±3080	265 × 154 mm (10.43 × 6.06 in.)
A4 (vertical)	-4220,-2960	4220,2960	±4820	±3560	241 × 178 mm (9.49 × 7.01 in.)
A3 (vertical)	-6680,-4700	6680,4700	±7280	±5300	364 × 265 mm (14.33 × 10.43 in.)
A2 (horizontal)	-10640,-6680	10640,6680	±11240	±7280	562 × 364 mm (22.13 × 14.33 in.)
A1 (horizontal)	-15580,-10160	16180,10160	±16180	±10760	809 × 538 mm (31.85 × 21.18 in.)
A1 (vertical)	-15100,-10640	15100,10640	±15700	±11240	785 × 562 mm (30.91 × 22.13 in.)
A0 (vertical)	-22060,-15580	22060,15580	±22660	±16180	1133 × 809 mm (44.61 × 31.85 in.)
Architectural ^C (horizontal)	-10952,-7424	10952,7424	±11552	±8024	577.6 × 401.2 mm (22.74 × 15.8 in.)
Architectural ^D (vertical)	-16568,-10952	16568,10952	±17168	±11552	858.4 × 577.6 mm (33.8 × 22.74 in.)
Architectural ^D (horizontal)	-17048,-10472	17048,10472	±17648	±11072	882.4 × 553.6 mm (34.74 × 21.8 in.)
Architectural ^E (vertical)	-22664,-17048	22664,17048	±23264	±17648	1163.2 × 882.4 mm (45.8 × 34.74 in.)

Expanded Plot Size
(Expand: ON)

Paper Size	Default Scaling Points P1 P2		Hard-Clip Plotting Range		Maximum Plotting Area (X and Y)
	P1 _x ,P1 _y	P2 _x ,P2 _y	X-axis	Y-axis	
A (horizontal)	-4788,-3038	4788,3038	±5388	±3638	269.4 × 181.9 mm (10.61 × 7.16 in.)
A (vertical)	-4308,-3518	4308,3518	±4908	±4118	245.4 × 205.9 mm (9.66 × 8.11 in.)
B (vertical)	-7356,-4788	7356,4788	±7956	±5388	397.8 × 269.4 mm (15.66 × 10.61 in.)
C (horizontal)	-10 376,-7356	10 376,7356	±10976	±7956	548.8 × 397.8 mm (21.61 × 15.66 in.)
D (horizontal)	-16 472,-9896	16 472,9896	±17072	±10496	853.6 × 524.8 mm (33.61 × 20.66 in.)
D (vertical)	-15 992,-10 376	15 992,10 376	±16592	±10976	829.6 × 548.8 mm (32.66 × 21.61 in.)
E (vertical)	-21 072,-16 472	21 072,16 472	±21672	±17072	1083.6 × 853.6 mm (42.66 × 33.61 in.)
A4 (horizontal)	-5140,-2920	5140,2920	±5740	±3520	287 × 176 mm (11.3 × 6.93 in.)
A4 (vertical)	-4660,-3400	4660,3400	±5260	±4000	263 × 200 mm (10.35 × 7.87 in.)
A3 (vertical)	-7120,-5140	7120,5140	±7720	±5740	386 × 287 mm (15.2 × 11.3 in.)
A2 (horizontal)	-11 080,-7120	11 080,7120	±11680	±7720	584 × 386 mm (23 × 15.2 in.)
A1 (horizontal)	-16 020,-10 600	16 020,10 600	±16620	±11200	831 × 560 mm (32.72 × 22.05 in.)
A1 (vertical)	-15 540,-11 080	15 540,11 080	±16140	±11680	807 × 584 mm (31.77 × 22.3 in.)
A0 (vertical)	-22 500,-16 020	22 500,16 020	±23100	±16620	1155 × 831 mm (45.47 × 32.72 in.)
Architectural C (horizontal)	-11 392,-7864	11 392,7864	±11992	±8464	599.6 × 423.2 mm (23.61 × 16.66 in.)
Architectural D (horizontal)	-17 488,-10 912	17 488,10 912	±18088	±11512	904.4 × 575.6 mm (35.61 × 22.66 in.)
Architectural D (vertical)	-17 008,-11 392	17 008,11 392	±17608	±11992	880.4 × 599.6 mm (34.66 × 22.03 in.)
Architectural E (vertical)	-23 104,-17 488	23 104,17 488	±23704	±18088	1185.2 × 904.4 mm (46.66 × 36.61 in.)

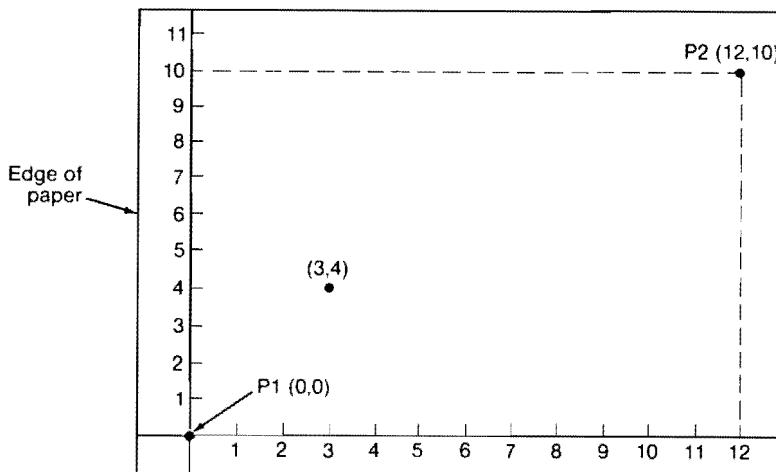
Scaling

Besides establishing user units, scaling relies on the relationship between P1 and P2.

The scaling instruction (SC) specifies the minimum and maximum user-unit coordinates. When you issue an SC instruction, P1 and P2 take on the X,Y coordinate values you specify. The entire plotting area is then effectively divided into an imaginary grid based on the new user units. This is sometimes referred to as mapping; the user-unit coordinates are ‘mapped onto’ P1 and P2. P1 and P2 do not represent a graphic limit; therefore, the new user-unit coordinate system extends across the entire range of the plotter-unit coordinate system. Thus, you can plot to a point beyond P1 or P2, as long as you are within the hard-clip limits. The actual size of the units depends on the locations of P1 and P2 and the range of user units established by the SC instruction.

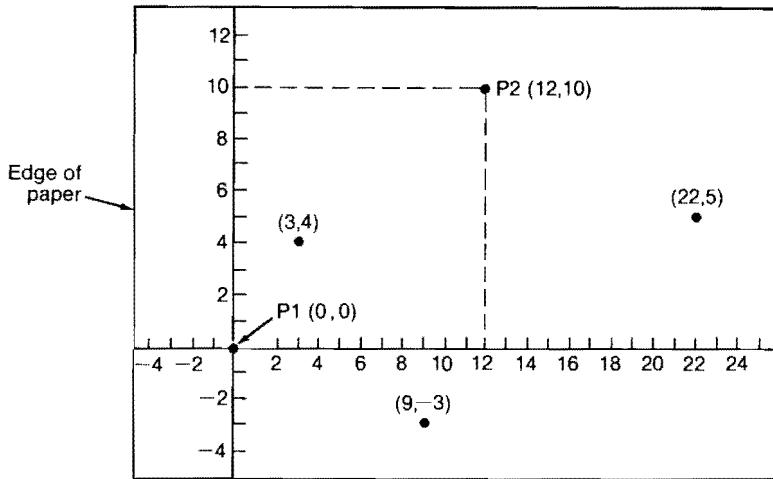
NOTE: Scaling does not change the locations of P1 and P2, only their coordinate values. You can change the plotter-unit locations of P1 and P2 with the input P1 and P2 instruction (IP) or with the front panel (refer to your User’s Guide). ■

The following example will help you to understand the scaling instruction. To divide the X-axis into 12 units representing months, and the Y-axis into 10 units representing thousands of dollars, specify the X-axis to scale from 0 to 12, and the Y-axis to scale from 0 to 10. P1 becomes the origin with user-unit coordinate (0, 0) and P2 becomes (12, 10). The entire plotting area is now divided into the desired units as shown below. Subsequent plotting instructions will use these units. If you tell the plotter to move to the point (3, 4), the plotter will move to the location equivalent to (3, 4) user units (*not* (3, 4) plotter units).



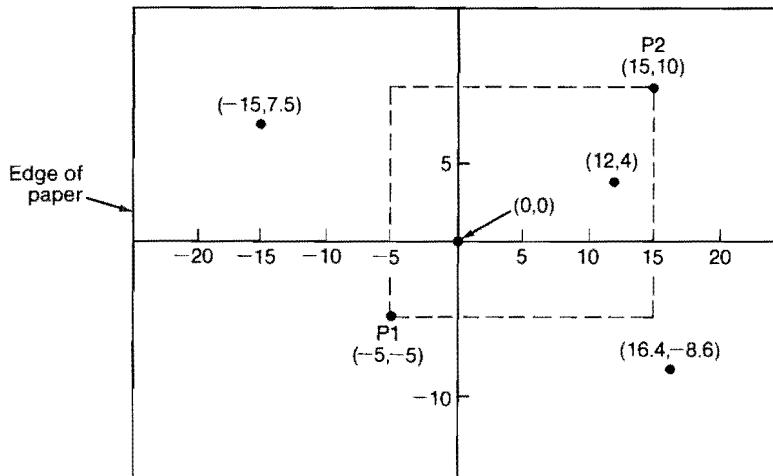
User-Unit Scaling with Default P1 and P2

If you move the locations of P1 and P2, the size of user units will change. The previous illustration showed P1 and P2 in their default locations. Following are the same minimum/maximum user units, only with different locations of P1 and P2. Note that the size of the user units have decreased.



Same User-Unit Scaling with New P1 and P2

To further illustrate the flexibility of user-unit scaling, the following example shows the P1 and P2 locations with the user-unit values of $(-5, -5)$ and $(15, 10)$ respectively. Remember, because the origin in this illustration is at the physical center of the hard-clip limits, you can plot anywhere on the plotting area using negative or positive values.



New P1 and P2 and User-Unit Scaling with Negative Values

An obvious benefit of user-unit scaling is that you can change the P1 and P2 locations with front-panel buttons so that your plot occupies more or less space on the page. Contrast this flexibility with plotting in plotter units; since plotter units are absolute with relation to a fixed origin, plots always occupy the same space on any sized media.

DF, Default

USE: Sets certain graphics functions to their predefined default settings. Use this instruction to return the plotter to a known state while maintaining the current locations of P1 and P2. When you use DF at the beginning of a program, unwanted graphics parameters such as character size, slant, or scaling are not inherited from another program.

SYNTAX: DF;

REMARKS: No parameters are used. DF resets the plotter to the following conditions.

Function	Equivalent Instruction	Default Condition
Label Buffer	BL CHR\$(3)	Cleared.
Alternate Character Set	CA0;	Character set 0.
Character Chord Angle	CC;	Chord angle of 5.
Character Selection Mode	CM;	HP 7-bit mode.
Standard Character Set	CS0;	Character set 0.
Chord Tolerance	CT;	Chord angle of 5.
Digitize Clear	DC;	Clear DP instruction and return to current display.
Downloadable Character Buffer	DL;	Cleared.
Direction Absolute	DII,0	Horizontal characters.
Label Terminator	DT;	CHR\$(3)/decimal code 3/ ETX .
Direction Vertical	DV;	Horizontal characters.
Extra Space	ES0,0;	No extra space.
Fill Type	FT;	—Type 1, solid bidirectional fill. —Spacing determined by PT Instruction. —0 degrees angle.
Input Mask	IM;	Recognizes all defined errors.
Input Window	IW;	Set to hard-clip limits.

(Table continues)

Function	Equivalent Instruction	Default Condition
Label Origin	L01;	Standard labeling starting at current location.
Line Type	LT;	Solid line.
Plotting Mode	PA;	Absolute plotting.
Polygon Mode	PM0; PM2;	Polygon buffer cleared.
Page Size	PS;	Uses normal hard-clip limits.
Pen Thickness	PT;	0.3 mm
Scaling	SC;	User-unit scaling is off.
Character Size Absolute	SI;	Size as follows: —Width = 0.285 cm —Height = 0.375 cm
Character Slant	SL;	No slant.
Symbol Mode	SM;	Off.
Select Standard Set	SS;	Standard set selected.
Tick Length	TL;	$tp=tn=0.5\% \text{ of } P2_x - P1_x \text{ for X-tick}$ $\text{and } 0.5\% \text{ of } P2_y - P1_y \text{ for Y-tick}$
User-Defined Fill	UF;	Solid bidirectional fill.

In addition, the carriage-return point for labeling instructions is updated to the current pen position.

The following plotter conditions are *not* affected by a DF instruction:

- automatic pen operations
- locations of P1 and P2
- current pen, its location, and up/down position
- current pen group and its definition
- pen speed, force, and acceleration
- 90-degree rotation
- axis alignment
- enable cut line instruction status
- function key definitions established by KY and WD instructions
- generated errors (not cleared)

**RELATED
INSTRUCTIONS:** IN, Initialize

ERRORS:

Condition	Error	Plotter Response
any parameter used	2	executes instruction

IN, Initialize

USE: Resets most plotter functions to their default settings. Use this instruction to return the plotter to a known state and to cancel settings that may have been changed by a previous program.

NOTE: Buffer sizes established by GM or ESC.T are **not** affected by the IN instruction. If you change buffer sizes, always reestablish the default buffer sizes by using a GM or ESC.T without parameters at the end of the program. ■

In this manual, all program examples begin with (IN;) to clear unwanted conditions from the previous program.

The IN instruction has no effect on handshake protocol in any RS-232-C environment, but does clear any existing I/O or HP-GL error condition.

SYNTAX: IN;
 or
IN -I;

REMARKS:

IN; — Sets the plotter to the same conditions as the DF instruction, plus these *additional* conditions:

- raises the pen (PU;)
- cancels programmatic 90-degree rotation (RO;)
- sets P1, P2, and the axis-align point to the X,Y coordinate values set when the hard-clip limits were established (IP;)
- turns on automatic pen operations (AP;)
- sets default pen speed, force, and acceleration values for the carousel installed in the plotter (VS;FS;AS;)
- clears any HP-GL error condition
- clears the display and removes any function key definitions established by the KY and WD instructions (KY;WD CHR\$(3))
- sets the group count to 0 (GC;)

IN-1; — Establishes a partial initialization that resets all of the same conditions as *(IN;)* **except** the following.

- default pen speed, force, and acceleration values for the carousel installed in the plotter (*VS;FS;AS;*)
- automatic pen operations (*AP;*)
- programmatic 90-degree rotation (*RO;*)

Changing any of the following front-panel settings from their defaults also establishes a partial initialization.

P1 P2 Speed

Force Rotate

Group Sort

If the user changes any of these front-panel settings, none of them will be restored to their default values by *(IN;)*. You can override these settings by implementing the individual instructions for those settings you don't want under user control.

Use the front-panel **Reset** to restore these settings to their default values.

RELATED

INSTRUCTIONS: DF, Default

OA, Output Actual Location and Pen Status

OC, Output Commanded Location and Pen Status

OS, Output Status

ERRORS:

Condition	Error	Plotter Response
more than 1 parameter	2	executes IN;
parameter ≥ 0	2	executes IN;

IP, Input P1 and P2

USE: Allows you to establish new or default locations for the scaling points P1 and P2. P1 and P2 are used by the scaling instruction (SC) to establish user-unit scaling. The IP instruction is often used to ensure that a plot is always the same size, regardless of where P1 and P2 might have been set from the front panel or the size of media loaded in the plotter. This instruction can also be used in advanced techniques such as plotting mirror images, enlarging/reducing plots, and enlarging/reducing relative character size or direction (refer to Chapters 7 and 9).

SYNTAX: IP $P1_X, P1_Y, P2_X, P2_Y$;
or
IP;

Parameter	Format	Range	Default
X,Y coordinates	integer	-8 388 608 to 8 388 607* plotter units	depends on media size**

*Range for sheet media, refer to *Long-axis Plotting* in Chapter 12 for uses beyond this range with roll media.

**Refer to the table of plotter unit ranges for various media sizes earlier in this chapter.

REMARKS:

- **No Parameters** — Sets P1 and P2 to their default locations for the media size, adjusted by any current axis rotation or alignment.

NOTE: If an IP instruction without parameters is executed after the axes have been rotated, the locations of P1 and P2 are changed to reflect the rotation. Additionally, the X,Y coordinates for each scaling point are reversed. For example, (IP;) after an axis rotation on A-size media sets P1 to (-2648, -4383) and P2 to (2648, 4383).

The front-panel **Invert** does not affect the plotter unit values of P1 and P2. ■

- **X,Y Coordinates** — Specify the location of P1 and P2 in plotter units. You don't have to specify P2; however, if you do not, P2 tracks P1 and its coordinates change so that the X- and Y-distances between it and P1 stay the same. This tracking process can cause P2 to end up located outside the hard clip limits. Used carefully, this tracking function can be useful for preparing more than one equal-sized plot on one page. For an example, refer to *Drawing Equal-Sized Pictures on One Page* in Chapter 9.

The IP instruction remains in effect until another IP instruction is executed, P1 and P2 are changed from the front panel, the plotter is initialized, or a different paper size is loaded.

RELATED

INSTRUCTIONS: SC, Scale
RO, Rotate

ERRORS:

Condition	Error	Plotter Response
1 or 3 parameters	2	ignores instruction
more than 4 parameters	2	uses first 4 parameters
number out of range	3	ignores instruction
P2 tracking P1 puts P2 _X or P2 _Y out of range	3	sets coordinate within range

SC, Scale

USE: Establishes a user-unit coordinate system by mapping user-defined coordinate values onto the scaling points P1 and P2. Thus, you can plot in units convenient to your application. In addition, you can use this instruction to establish automatic isotropic scaling or to relocate the origin and set a specific ratio of plotter units to user units. For a discussion of the basic concept of scaling, refer to *Scaling* in Chapter 2.

SYNTAX: SC $X_{min}, X_{max}, Y_{min}, Y_{max} (, type(, left, bottom));$
 or
 SC $X_{min}, X_{factor}, Y_{min}, Y_{factor}, type;$
 or
 SC;

Parameter	Format	Range	Default
Xmin	real	-8 388 608 to 8 388 607	none
Xmax	real	-8 388 608 to 8 388 607	none
Ymin	real	-8 388 608 to 8 388 607	none
Ymax	real	-8 388 608 to 8 388 607	none
type	integer	0 to 2	0
left	real	0 to 100%	50%
bottom	real	0 to 100%	50%
Xfactor	real	-8 388 608 to 8 388 607*	none
Yfactor	real	-8 388 608 to 8 388 607*	none

*Excluding zero (0) and values approaching zero

REMARKS: Has two forms. The first sets up standard user-unit scaling and allows you to specify automatic isotropic scaling. The second form allows you to move the origin while maintaining a specific ratio of plotter units to user units.

- **No Parameters** — Turns off both forms of scaling.

FORM 1: SC X_{min} , X_{max} , Y_{min} , Y_{max} (*, type(, left, bottom)*);

- **X_{min} , X_{max} , Y_{min} , Y_{max}** — Represent the user unit X- and Y-axis ranges, respectively. As a result, the first and third parameters (X_{min} and Y_{min}) are the coordinate pair that is mapped onto P1; the second and fourth parameters (X_{max} and Y_{max}) are the coordinate pair that is mapped onto P2. (This is different from the IP instruction, where the parameters are expressed as X,Y coordinate pairs rather than as ranges.) For example, (SC0, 15, 0, 10;) specifies P1 as (0,0) and P2 as (15, 10).

X_{min} cannot be set equal to X_{max} , and Y_{min} cannot be set equal to Y_{max} .

As their names suggest, you will normally want to specify X_{min} smaller than X_{max} , and Y_{min} smaller than Y_{max} . If you specify X_{min} larger than X_{max} and Y_{min} larger than Y_{max} , your plot could be drawn as a mirror image—reversed and/or upside down—depending on the relative positions of P1 and P2.

The parameters of the SC instruction are always mapped onto the current P1 and P2 locations. P1 and P2 retain these new values until scaling is turned off or another SC instruction redefines the user-unit values. Thus, the size of a user unit could change if any change is made in the relative position and distance between P1 and P2 *after* an SC instruction is executed.

- **Type** — allows you to specify anisotropic or isotropic scaling.
 - 0 Anisotropic scaling. Allows a user unit to be different sizes along the X-axis and the Y-axis. If your plot includes letters or shapes such as circles, they will be distorted if your scaling is anisotropic. For example, a circle will be drawn as an ellipse (egg-shaped) rather than a circle.
 - 1 Isotropic scaling. The user unit is the same size on the X- and Y-axis. When the plotter encounters a CI (circle), PM (polygon), SR (relative character size), DR (relative label direction), LT (pattern length), or TL (tick length) instruction—it will automatically create the largest square (isotropic) plotting area that will fit into the current plotting area and use the isotropic area for plotting.

NOTE: You can set isotropic scaling manually by setting P1 and P2 so that they define a square area. The easiest way, however, is to set it automatically using the type parameter. ■

- **Left, Bottom** — Allows you to position the isotropic plotting area in the hard-clip limits.

Normally, the isotropic area is centered with the blank space equally divided between left and right or top and bottom; e.g., the defaults of the left and bottom parameters of 50%.

You can however, move the isotropic area around in the P1/P2 plotting area using the left and bottom parameters. The left parameter indicates the percentage of the blank space on the left of the isotropic area; the bottom parameter indicates the blank space below the isotropic area. For example, (*SC0, 15, 0, 10, 1, 0, 0;*) specifies P1 as (0, 0), P2 as (15, 10), isotropic scaling, and locates the isotropic plotting area in the bottom lefthand corner of the P1/P2 plotting area.

FORM 2: *SC X_{min}, X_{factor}, Y_{min}, Y_{factor}, type;*

The second form of the scaling instruction allows you to shift the user-unit origin (0, 0) relative to P1 while maintaining a specific ratio of plotter units to user units.

- **X_{min}, X_{factor}, Y_{min}, Y_{factor}** — Establish the user unit coordinates of P1 and the ratio of plotter to user units. X_{min} and Y_{min} are the user unit coordinates of P1. X_{factor} sets the number of plotter units per user unit on the X-axis; Y_{factor} sets the number of plotter units per user unit on the Y-axis.
- **Type** — Must be 2 for this type of scaling.

For example, (*SC0, 1, 0, 1, 2;*) moves the origin to P1 and establishes a one-to-one ratio of plotter to user units. Plotting with this scaling instruction would be just like plotting in plotter units except that you could do everything in positive numbers.

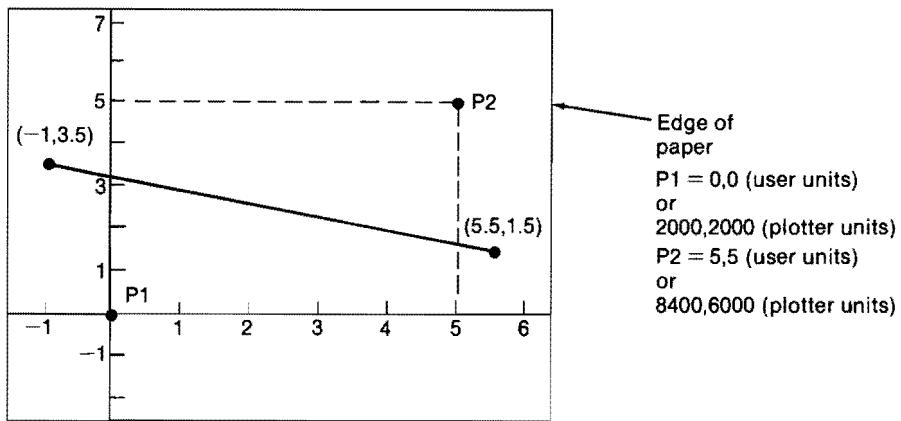
(*SC0, 40, 0, 40, 2;*) allows scaling in millimetres since 1 millimetre = 40 plotter units. Each user unit will be 1 millimetre square (thus, your scaling is also isotropic).

(*SC0, 1.016, 0, 1.016, 2;*) allows scaling in thousandths of an inch since 1 inch = 1016 plotter units. Each user unit will be 1000 of an inch square (also isotropic).

(*SC600, 1, 600, 1, 2;*) shifts the plotter-unit origin to the lower left corner of the plotting area while maintaining user units the same size as plotter units. This allows you to plot to the lower left corner (hard-clip limits) without using negative coordinates.

While scaling is on (i.e., after either form of the SC instruction has been executed), only those plotting instructions that can be issued in 'current units' are interpreted as user units; all instructions that can only be issued in plotter units are still interpreted as plotter units. (The table in the SYNTAX section of each instruction tells you what kind of units each parameter requires.)

Remember that the SC parameters are mapped onto the current locations of P1 and P2. P1 and P2 do not represent a graphic limit; therefore, the new user-unit coordinate system extends across the entire range of the plotter-unit coordinate system. Thus, you can plot to a point beyond P1 or P2, as long as you are within the hard-clip limits. For example, you can plot from the point (-1, 3.5) to the point (5.5, 1.5) as illustrated below.



RELATED

INSTRUCTIONS: IP, Input P1 and P2

ERRORS:

Condition	Error	Plotter Response
no parameters	none	turns scaling off
more than 7 parameters	2	executes first 4 parameters
6 parameters or less than 4 parameters	2	ignores instruction
$X_{\min} = X_{\max}$ or $Y_{\min} = Y_{\max}$ or number out of range	3	ignores instruction

Notes



4

Drawing Lines and Rectangles

Now that you understand the coordinate system, how to reset your plotter to standard values, and how to scale your plot, you are ready to begin drawing plots. In this chapter, you will learn how to select, change, raise, and lower the pen. Additionally, you'll learn how to use absolute or relative coordinates to draw lines and rectangles.

4

Instructions Covered

EA, Edge Rectangle Absolute
ER, Edge Rectangle Relative
PA, Plot Absolute
PD, Pen Down
PR, Plot Relative
PU, Pen Up
SP, Select Pen

Terms You Should Understand

Coordinate	Initialize
Current Units	Point
Default	Scaling

Selecting and Controlling the Pen

To draw any plot, you must load a pen from the carousel into the pen holder. While you can load pens from the front panel (refer to the User's Guide), the most efficient way is by using the select pen (SP) instruction in your program. Use the SP instruction to load a pen when the pen holder is empty, to change to a different pen (for a new color and/or thickness), or to return the pen to the carousel at the end of the program.

NOTE: If a pen is inactive in the pen holder for a period of time, the plotter returns it to the carousel to keep it from drying out. The period of time depends on the pen type. Refer to the automatic pen (AP) instruction in Chapter 10 for further details. ■

Pen Position and Location

4

When you initialize the plotter, two of the default conditions set are the pen position and location.

- **Pen Position** — Up (off the media) or Down (on the media). The pen will be up after you initialize the plotter.

Once a pen is loaded in the pen holder, you must raise and lower the pen in order to draw. Raise the pen off the media using the pen up (PU) instruction. When you have moved the pen to its proper location, lower the pen onto the media using the pen down (PD) instruction. You must be aware of the pen's position (up or down) to avoid drawing stray lines between figures.

Every time you use the PU or PD instructions or change pens the plotter updates the pen position information. Most HP-GL instructions plot according to the 'current pen position'. Some instructions have an automatic PD instruction and return the pen to its previous position (up or down) after execution. The definition of the instruction will tell you whether it has an automatic PD instruction. If part of your plot isn't drawn, make sure your program uses the PD instruction before the affected instructions.

- **Pen Location** — The X,Y coordinates of the current pen location.

Whenever a plotting instruction is completed, the pen location is updated to the current point. The next instruction will begin at the new pen location.

Using the front-panel to move pens or pressing **P1** or **P2** also updates the current pen location.

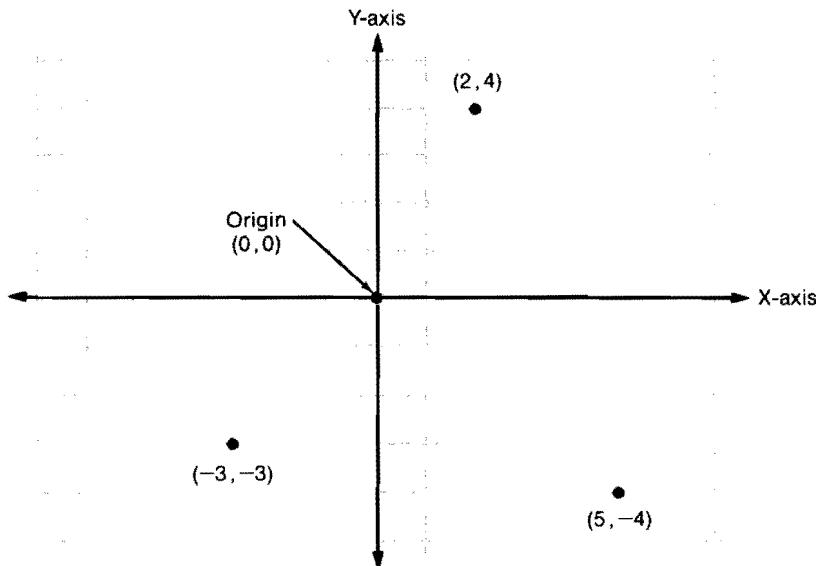
Some instructions do not update the current pen location. For example, after the circle (CI) instruction finishes drawing a circle, it returns to the previous pen location. The definition of each instruction will tell you whether the current pen location is updated or restored.

Initializing the plotter does not reset the current pen location. The DF and IN instructions and the front-panel **Reset** do not reset the current pen location. You must specify your beginning pen location for each plot.

Absolute versus Relative Movement

The plot absolute (PA) and plot relative (PR) instructions allow you to determine whether you want the pen to move using X,Y *coordinates* or X,Y *increments*.

Absolute movement uses X,Y coordinates to specify an exact, fixed point relative to the origin (0, 0). In the following illustration the coordinates (-3, -3); (2, 4); and (5, -4) are always in the same place with respect to the origin, no matter where the pen is when the coordinates are issued.

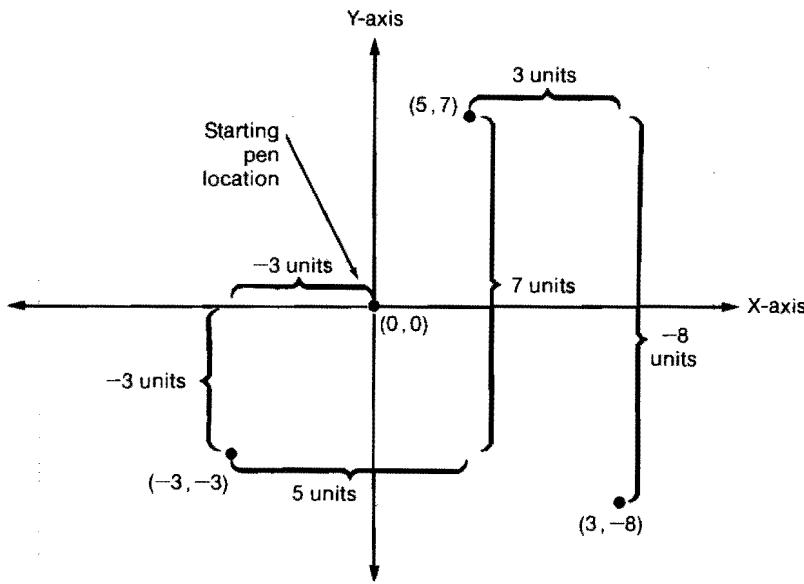


Absolute Coordinates

Note that all X-coordinates to the right of the Y-axis are positive values, those to the left are negative. Similarly, all Y-coordinates above the X-axis are positive, those below are negative. By changing the sign of the X- and/or Y-coordinate, you change the location of the pen to a new quadrant.

Relative coordinates are more accurately called increments, because their values represent the number of units the pen moves from its current pen location. As with absolute coordinates, the units can be user units or plotter units.

For example, assume that the pen is currently at the origin. To arrive at each location marked from left to right in the previous illustration, count 3 units to the left and 3 units down from the current pen location; these are both negative directions with respect to the origin. This is the relative location $(-3, -3)$. Now move 5 positive X-units and 7 positive Y-units from this location to the upper point; this is the relative location $(5, 7)$. From this location, move to the lower right point by moving 3 positive X-units and 8 negative Y-units $(3, -8)$.



Relative Coordinates

Relative movement is very useful in many applications where you know the dimensions of the shape you want, but don't want to calculate the absolute coordinates. Using the previous illustration, if you knew you wanted a box 4 X-units by 8 Y-units, you could use the edge relative rectangle (ER) instruction (which draws to the increments given from the current pen location) to draw the box without having to calculate the absolute coordinates of the opposite corner.

All instructions that use relative increments include 'relative' in the title and note that information in their definitions.

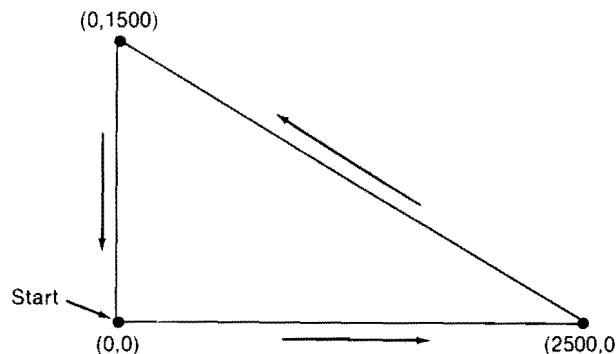
NOTE: Relative increments add to the current pen location. The plotter automatically converts the new relative position to absolute coordinates and updates the current pen location. ■

Drawing Lines

You can draw lines between any two points (X,Y coordinate pairs) using the PD instruction with a series of absolute and/or relative coordinate pairs. (If one (or both) coordinate pair falls outside the plotting area (soft- or hard-clip limits), the plotter will draw only the portion of the line that falls within the plotting area.)

In the following example, note that the PA instruction establishes absolute plotting, and the coordinate pair (0,0) designates the beginning pen location.

```
10  'Insert configuration statement here  
20  PRINT #1, "IN;SP1;PA0,0;PD2500,0,0,1500,0,0;SP0;"  
30  END
```



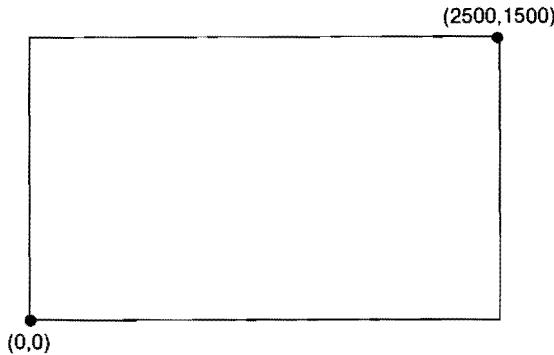
Drawing Rectangles

All rectangle instructions start at the current pen location and require that you specify the diagonally opposite corner location. For example, from a coordinate pair defining the lower left corner, you need specify the upper-right corner location only.

You can define a rectangle using absolute (EA) or relative (ER) coordinates. These instructions draw the rectangle by edging the defined area.

The following simple program uses EA to draw a rectangle.

```
10 'Insert configuration statement here
20 PRINT #1, "IN;SP1;"
30 PRINT #1, "PA0,0;EA2500,1500;"
40 PRINT #1, "SP0;"
50 END
```



EA, Edge Rectangle Absolute

USE: Defines and outlines a rectangle using absolute coordinates. Use the EA instruction to create charts that require rectangles; for example, bar charts, flow charts, and organization charts.

SYNTAX: EA X,Y;

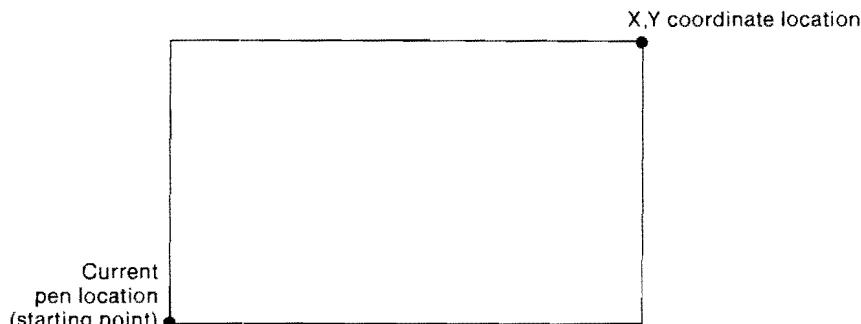


Parameter	Format	Range	Default
X,Y increments	current units	-8 388 608 to 8 388 607	none

REMARKS: Defines and edges a rectangle using the current pen, a solid line, and absolute plotting. The EA instruction includes an automatic pen down. When the rectangle is complete, the pen returns to the starting point (restores current pen location) and restores the pen position (up/down).

- **X,Y Coordinates** — Specify the opposite corner of the rectangle from the current pen location. The current pen location is the starting point of the rectangle. Coordinates are interpreted in current units: as user units when scaling is on, as plotter units when scaling is off.

NOTE: The illustration shows the current pen location in the lower-left corner and the instruction's X,Y coordinates in the upper-right corner. However, they can be in any two diagonally opposite corners. ■



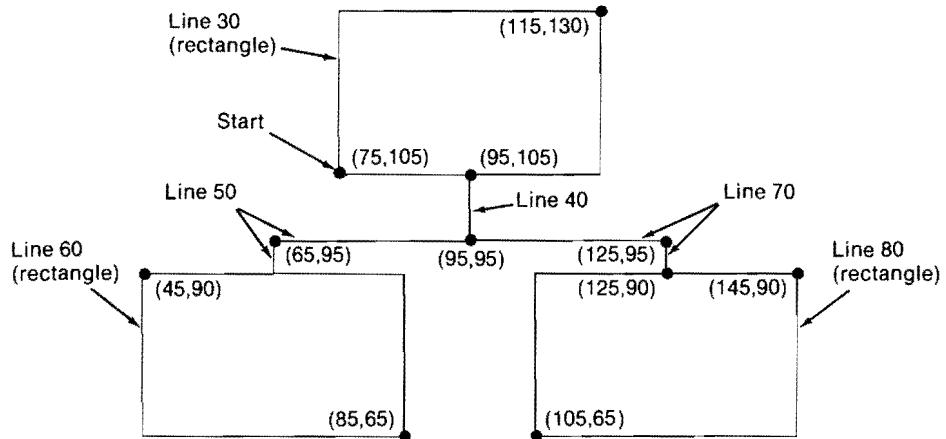
The EA instruction clears the polygon buffer and then uses it to define the rectangle before drawing. A rectangle requires enough buffer space to hold five points. The default buffer size is sufficient. Refer to *Understanding the Plotter's Buffers* in Chapter 8 for more information on buffers.

EXAMPLE:

```

10  'Insert configuration statement here
20  PRINT #1, "IN;500,200,0,200,1;SP1;"
30  PRINT #1, "PA75,105;EA115,130;"
40  PRINT #1, "PA95,105;PD95,95;"
50  PRINT #1, "PD65,95,65,90;"
60  PRINT #1, "PU45,90;EA85,65;"
70  PRINT #1, "PU95,95;PD125,95,125,90;"
80  PRINT #1, "PU145,90;EA105,65;"
90  PRINT #1, "SP0;"
100 END

```

**RELATED****INSTRUCTIONS:** RA, Fill Rectangle Absolute

ER, Edge Rectangle Relative

RR, Fill Rectangle Relative

ERRORS:

Condition	Error	Plotter Response
only 1 coordinate	2	ignores instruction
more than 2 coordinates	2	uses first 2 coordinates
number out of range	3	ignores instruction
polygon buffer overflow	7	edges contents of buffer

ER, Edge Rectangle Relative

USE: Defines and outlines a rectangle using relative coordinates. Use ER to create charts that require rectangles; for example, bar charts, flow charts, and organization charts.

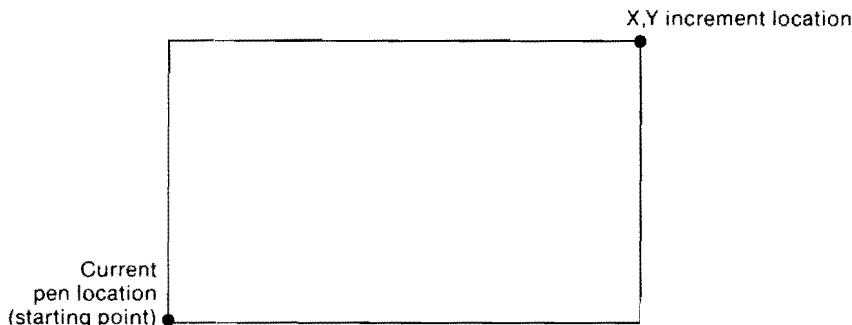
SYNTAX: ER X,Y;

Parameter	Format	Range	Default
X,Y increments	current units	-8 388 608 to 8 388 607	none

REMARKS: Defines and edges a rectangle using the current pen, a solid line, and relative coordinates. The ER instruction includes an automatic pen down. When the rectangle is complete, the pen returns to the starting point (restores current pen location) and restores the pen position (up/down).

- **X,Y Coordinates** — Specify the opposite corner of the rectangle from the current pen location. The current pen location is the starting point of the rectangle. Increments are interpreted in current units: as user units when scaling is on, as plotter units when scaling is off.

NOTE: The illustration below shows the current pen location in the lower-left corner and the instruction's X,Y increments in the upper-right corner. However, they can be in any two diagonally opposite corners. ■



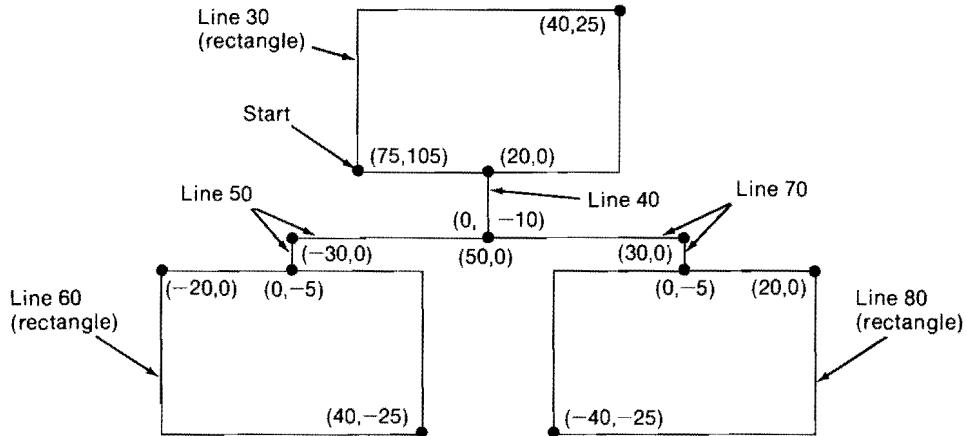
NOTE: The ER instruction clears the polygon buffer and then uses it to define the rectangle before drawing. A rectangle requires enough buffer space to hold five points. The default buffer size is sufficient. Refer to *Understanding the Plotter's Buffers* in Chapter 8 for more information on buffers. ■

EXAMPLE: This example uses relative coordinates to draw the same plot shown with the EA instruction. Compare this program with the EA example program to understand the differences between the coordinates used.

```

10  'Insert configuration statement here
20  PRINT #1, "IN;SC0,200,0,200,1;SP1;"
30  PRINT #1, "PA75,105;ER40,25;"
40  PRINT #1, "PR20,0;PD0,-10;"
50  PRINT #1, "PD-30,0,0,-5;"
60  PRINT #1, "PU-20,0;ER40,-25;"
70  PRINT #1, "PUS0,5;PD30,0,0,-5;"
80  PRINT #1, "PU20,0;ER-40,-25;"
90  PRINT #1, "SP0;"
100 END

```



RELATED

INSTRUCTIONS: RA, Fill Rectangle Absolute
EA, Edge Rectangle Absolute
RR, Fill Rectangle Relative

ERRORS:

Condition	Error	Plotter Response
only 1 increment	2	ignores instruction
more than 2 increments	2	uses first 2 increments
number out of range	3	ignores instruction
polygon buffer overflow	7	edges contents of buffer

PA, Plot Absolute

USE: Establishes absolute plotting and moves the pen to the specified absolute coordinates using the current pen position.

SYNTAX: PA $X,Y(\dots)$;
or
PA;

Parameter	Format	Range	Default
X,Y coordinates	current units	-8 388 608 to 8 388 607	none

REMARKS: Use PA when you want to establish absolute plotting.

- **No Parameters** — Establishes absolute plotting mode for subsequent instructions.
- **X,Y Coordinates** — Specify the absolute location to which the pen will move. Coordinates are interpreted in current units: as user units when scaling is on, as plotter units when scaling is off.

You can specify as many X,Y coordinate pairs as you want. When you include more than one coordinate pair, the pen moves to each point in the order given, using the current pen position (up or down). If the pen is up, PA moves the pen to the point; if the pen is down, PA draws a line to the point.

NOTE: If you are using an HP-IB interface, the number of coordinate pairs you can specify may be limited. This is dependent on the ability of your controller to output long strings without a line feed. ■

RELATED

INSTRUCTIONS: PR, Plot Relative

ERRORS:

Condition	Error	Plotter Response
odd number of coordinates	2	ignores last coordinate
number out of range	3	ignores wrong coordinate and any subsequent coordinates

PD, Pen Down

USE: Lowers the pen onto the writing surface for drawing.

SYNTAX: PD $X,Y(\dots)$;
or
PD;

Parameter	Format	Range	Default
X,Y coordinates/increments	current units	-8 388 608 to 8 388 607	none

REMARKS: Use PD to draw lines on the media.

- **No Parameters** — Lowers the pen without moving it to a new location.
- **X,Y Coordinates/Increments** — Lowers the pen and draws (in current units) to the point specified. You can specify as many X,Y coordinate pairs as you want. When you include more than one coordinate pair, the pen moves to each point in the order given.

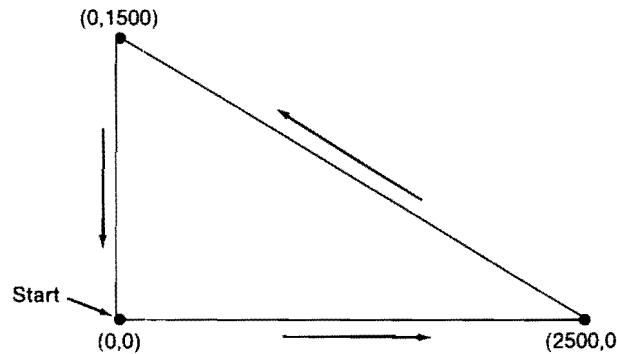
The pen will lower only if it is within the current graphics limits (window) and is not in the 'up' portion of a dashed line type (refer to the LT instruction).

Whether the PD instruction uses coordinates or increments depends on the most recently executed PA or PR instruction. If you have not issued a PA or PR instruction, absolute plotting (PA) is assumed.

NOTE: When using an HP-IB interface, the number of coordinate pairs you can specify may be limited. This is dependent on the ability of your controller to output long strings without a line feed. ■

EXAMPLE:

```
10 'Insert configuration statement here  
20 PRINT #1, "IN:SP1:PA0,0;PD2500,0,0,1500,0,0;SP0;"  
30 END
```

**RELATED**

INSTRUCTIONS: PA, Plot Absolute
PR, Plot Relative
PU, Pen Up

ERRORS:

Condition	Error	Plotter Response
odd number of coordinates	2	ignores last coordinate
number out of range	3	ignores that coordinate pair and any subsequent coordinates

PR, Plot Relative

USE: Establishes relative plotting and moves the pen to specified points, each successive move relative to the current pen location.

SYNTAX: PR $X, Y(\dots)$;
or
PR ;

Parameter	Format	Range	Default
X,Y increments	current units	-8 388 608 to 8 388 607	none

REMARKS: Use PR to establish relative plotting.

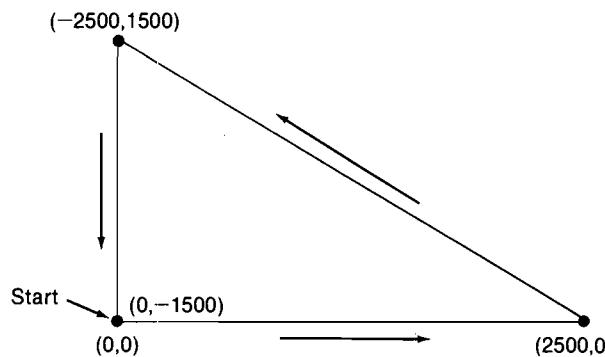
- **No Parameters** — Establishes relative plotting mode for subsequent instructions.
- **X,Y Increments** — Specify incremental moves relative to the current pen location. The pen will move to the new location, the next X,Y coordinate pair will indicate a move relative to this new location, and so on. Coordinates are interpreted in current units: as user units when scaling is on, as plotter units when scaling is off.

You can specify as many X,Y increment pairs as you want. When you include more than one increment pair, the pen moves to each point in the order given, using the current pen position (up or down). If the pen is up, PR moves the pen to the point; if the pen is down, PR draws a line to the point.

NOTE: If you are using an HP-IB interface, the number of coordinate pairs you can specify may be limited. This is dependent on the ability of your controller to output long strings without a line feed. ■

EXAMPLE:

```
10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;"
30  PRINT #1, "PA0,0;PD;PR2500,0,-2500,1500,0,-1500;""
40  PRINT #1, "SP0;""
50  END
```

**RELATED****INSTRUCTIONS:** PA, Plot Absolute**ERRORS:**

Condition	Error	Plotter Response
odd number of coordinates	2	ignores last coordinate
number out of range	3	ignores wrong coordinate pair and any subsequent coordinates

PU, Pen Up

USE: Raises the pen from the plotting surface. Use this instruction to move the pen to the beginning of the next line.

SYNTAX: PU $X, Y(\dots)$;
or
PU;

Parameter	Format	Range	Default
X,Y coordinates/increments	current units	-8 388 608 to 8 388 607	none

4

REMARKS: Use PU to raise the pen to prevent stray lines on the media.

- **No Parameters** — Raises the pen without moving the pen to a new location.
- **X,Y Coordinates/Increments** — Raise the pen and move to the point(s) specified. You can specify as many X,Y coordinate pairs as you want. When you include more than one coordinate pair, the pen moves to each point in the order given.

Coordinates are interpreted in current units: as user units when scaling is on, as plotter units when scaling is off.

Coordinates or increments depends on the most recently executed PA or PR instruction. If you have not issued a PA or PR instruction, absolute plotting (PA) is assumed.

RELATED

INSTRUCTIONS: PA, Plot Absolute
PD, Pen Down
PR, Plot Relative

ERRORS:

Condition	Error	Plotter Response
odd number of coordinates	2	ignores last coordinate
number out of range	3	ignores wrong coordinate pair and any subsequent coordinates

SP, Select Pen

USE: Loads specified pen into the pen holder or returns the current pen to the carousel. Use the SP instruction to change pen colors or widths during a plot. At the end of every program, use SP to return the pen to the carousel.

SYNTAX: SP *pen number*;
or
SP;

Parameter	Format	Range	Default
pen number	integer	0 to 8	0

REMARKS:

- **No Parameter** — Returns the pen currently in the pen holder to the carousel stall from which it came, if it is vacant. If the stall is occupied, the pen is placed in the vacant stall with the lowest number.
- **Pen Number** — Corresponds to the stall numbers marked on the pen carousel.
 - 0 Same as *no parameter*. Returns the pen currently in the pen holder to the carousel. Should be used at the end of every program.
 - 1-8 Loads the pen from the specified stall into the pen holder. If a pen currently is in the holder, that pen is stored before the new pen is selected. After selecting the new pen, the pen holder returns to the current pen location.

An SP instruction remains in effect until a new pen is selected or the current pen is returned to the carousel.

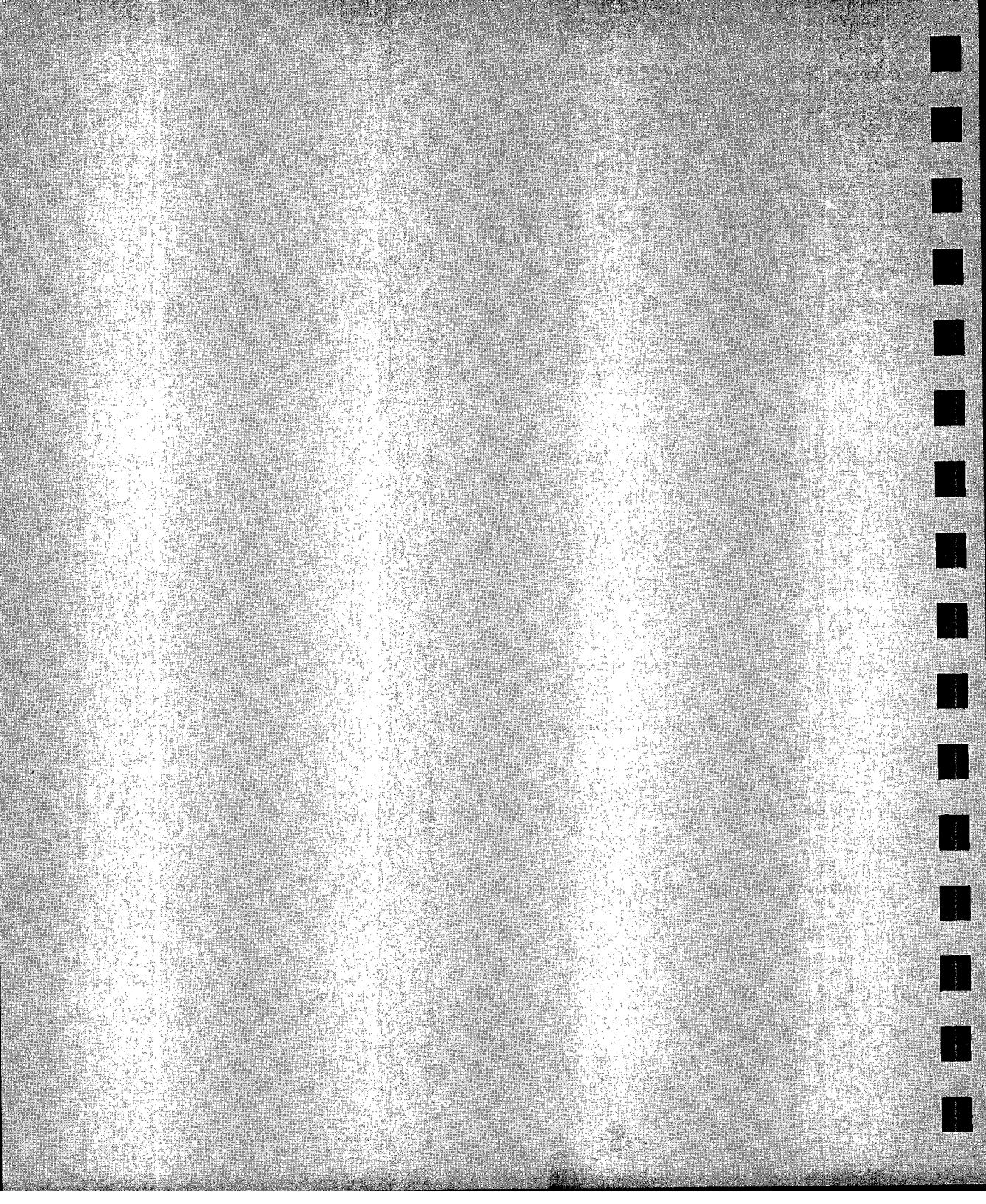
NOTE: If a pen is inactive in the pen holder for a period of time, the plotter returns it to the carousel to keep it from drying out. The period of time depends on the pen type. Refer to the automatic pen (AP) instruction in Chapter 10 for further details. ■

RELATED

INSTRUCTIONS: SG, Select Pen Group

ERRORS:

Condition	Error	Plotter Response
number greater than 8	none	ignores instruction
number less than 0	3	ignores instruction



Enhancing Your Plots

You can increase the visual effectiveness of your plots in several ways. You can use dotted or dashed line patterns, differentiate solid areas with patterned and solid fills, draw tick marks, and use symbols (on data points).

Instructions Covered

FT, Fill Type
LT, Line Type
PT, Pen Thickness
RA, Fill Rectangle Absolute
RR, Fill Rectangle Relative
SM, Symbol Mode
TL, Tick Length
UF, User-Defined Fill Type
XT, X-Tick
YT, Y-Tick

Terms You Should Understand

Anisotropic
Cross-Hatch
Grid

Hatch
Tick

Using Line Types

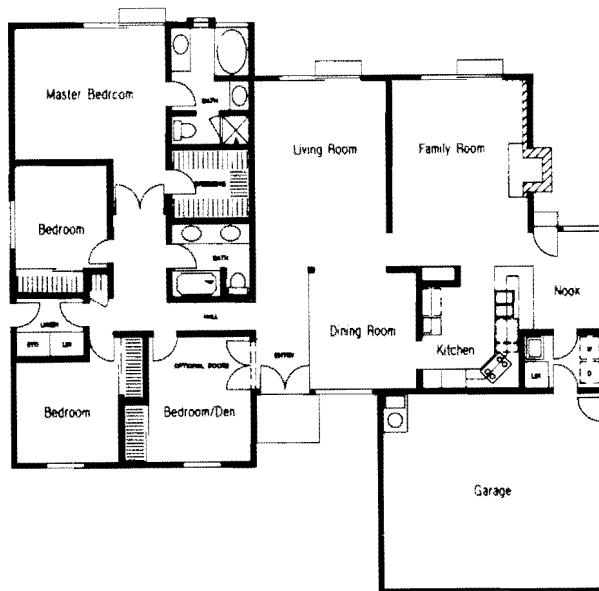
Line types are repeated patterns of dots and/or dashes (including solid lines). You can establish different line types using the line type (LT) instruction. Following are some possible uses of line types.

- Differentiate curves on plots such as line charts or measurement tracings, especially if these plots might later be reproduced in fewer colors.
- Show the lines that wouldn't be visible in a solid object.
- Indicate missing data on a curve.
- Indicate optional items.

Once you establish a line type, all lines drawn by the following instructions will be drawn using the specified line type.

- Arc Absolute and Relative (AA and AR)
- Circle (CI)
- Pen Down (PD, whether absolute or relative)
- Edge Polygon (EP)
- Fill Relative and Absolute Rectangles (RA and RR)
- Fill Wedge (WG)
- Fill Polygon (FP)
- Fill Type (FT)
- User-Defined Fill Type (UF)

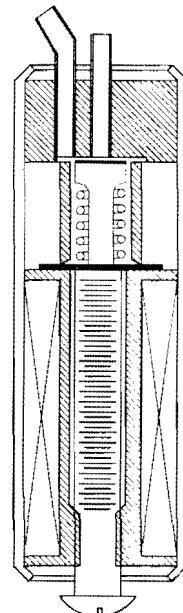
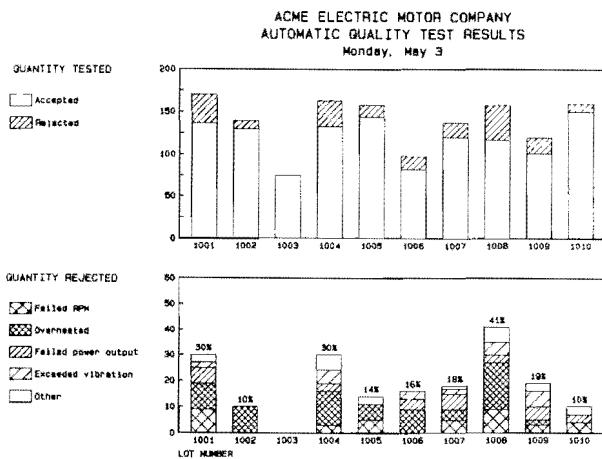
Fill patterns and their interactions with line types are discussed in the next section.



Using Fill Types

Use fill types or area fill to shade 'solid' areas on a plot. Fill types can be solid, parallel lines, cross-hatched or user-defined (refer to the next section). Use the fill type (FT) instruction to select fill types. If you use a parallel line, cross-hatched, or user-defined fill type; the fill type is drawn using the currently designated line type. That is, if you have selected a dashed line type and a parallel lines fill type, your figure will be filled with dashed, parallel lines. Use fill types:

- to differentiate solid areas of bar and pie charts, especially if these plots might later be reproduced in fewer colors.
- to show solid areas of cross-sections.
- to create wider 'lines' than a pen can draw when plotting greatly enlarged detail.



User-Defined Fill Types

Use the user-defined fill type (UF) instruction to define a fill pattern composed of gaps between parallel lines. You can create patterns such as a semilog or even a candy-stripe.



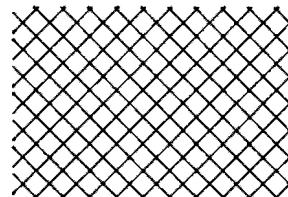
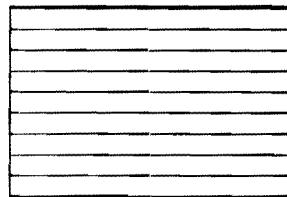
Filling Rectangles

Use the fill absolute and relative rectangle instructions (RA and RR) to fill rectangles. Like the edge rectangle instructions, the fill absolute and relative rectangle instructions start at the current pen location and require that you specify the diagonally opposite corner. The rectangular shape is formed by the fill pattern.

If you use an open fill type, you may want to edge (or outline) the rectangle also for a cleaner edge and to reduce the illusion of unequal size.

The following program draws two filled rectangles: one edged and one not.

```
10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;PA0,0;FT3;RR1500,1000;ER1500,1000;"
30  PRINT #1, "PR2000,0;FT4,100,45;RR1500,1000;SP0;"
40  END
```



FT, Fill Type

USE: Selects the shading pattern used to fill polygons (FP), rectangles (RA or RR), or wedges (WG). Use this instruction to enhance plots with solid fill, parallel lines (hatching), cross-hatching, or a fill pattern you designed using the user-defined fill type (UF) instruction.

SYNTAX: FT *type* (, *spacing* (, *angle*));
or
FT;

Parameter	Format	Range	Default
type	integer	1 to 6	1
spacing	current units	0 to 8 388 607	depends on type
angle	real	0 to 90 degrees*	0

*Practical range; actual range is -8388608 to 8 388 607.

REMARKS: The plotter maintains the spacing and angle parameters of the FT instruction. When you issue subsequent FT instructions, you can omit either the angle or the spacing and angle parameters if you want to use the same ones. The plotter will use the corresponding values from the previous FT instruction in place of the missing parameters. If you omit all parameters (FT,) the plotter uses the default values.

For example, if you use two FT instructions and the first FT was (FT3,100,45,), and the one you just issued is (FT4,), the plotter will use the spacing and angle from the first instruction and implement (FT4,100,45,). If (FT4,) is what you want to implement, you must send (FT;FT4,) to the plotter. If the first instruction had been (FT3,), it wouldn't affect the current (FT4,).

- **No Parameters** — Produces solid, bidirectional filling. Same as Type 1.
- **Type** — Selects the shading pattern you want. The six parameter values and corresponding fill types are listed below. ‘Bidirectional’ means the pen draws back and forth; ‘unidirectional’ means the pen draws in one direction only. For the highest quality solid fill on transparency film, use a unidirectional type (2 or 6).
 - 1 **Solid bidirectional.** Lines with spacing as defined in the pen thickness (PT) instruction discussed later in this chapter. (The fill type spacing parameter will be ignored.)
 - 2 **Solid unidirectional.** Lines with spacing as defined in the PT instruction. (The fill type spacing parameter will be ignored.)

- 3 **Parallel lines.** Uses the current line type (LT). (Always bidirectional for solid line type; unidirectional for dotted or dashed line types.)
- 4 **Cross-hatch.** Uses the current line type. (Always bidirectional for solid line type; unidirectional for dotted or dashed line types.)
- 5 **User-defined bidirectional.** Defined by the user-defined fill (UF) instruction described in this chapter. If you have not issued a UF instruction, the plotter will use type 1, solid fill.
- 6 **User-defined unidirectional.** Defined by the UF instruction described in this chapter. If you have not issued a UF instruction, the plotter will use type 1, solid fill.

Parameter Value	Fill Type	Pattern
1	bidirectional solid	
2	unidirectional solid	
3	hatch	
4	cross-hatch	
5	user-defined bidirectional	refer to the
6	user-defined unidirectional	UF instruction

- **Spacing —** For **solid-fill types 1 and 2**, the spacing parameter is ignored; their spacing is determined by the PT instruction.

For **fill types 3 and 4**, the spacing parameter is the distance between the parallel lines in the fill area.

For **fill types 5 and 6**, the spacing parameter is used by the UF instruction as the pattern repeat length.

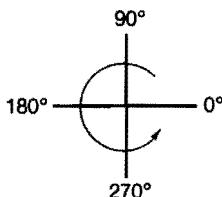
Spacing is interpreted in current units: as user units when scaling is on, as plotter units when scaling is off.

When scaling is on and is anisotropic, the spacing parameter is measured using units along the X-axis.

The default spacing is 1% of the diagonal distance between P1 and P2. Subsequent changes in the P1/P2 locations affect this distance and therefore affect spacing.

If scaling is on and is isotropic, the default distance will be 1% of the diagonal distance between the two points established by the plotter when it sets up the square, isotropic plotting area. (Refer to the scaling (SC) instruction for further details.)

- **Angle** — Referenced counterclockwise from the positive X-axis as shown below (0 and 180 are horizontal; 90 and 270 are vertical). The angle applies to all fill types. For cross-hatching (FT4), the first set of lines is drawn at the specified angle. The cross-hatched lines are then drawn at that angle plus 90 degrees.



EXAMPLE:



FT1;



FT2;



FT3;



FT4;



FT5;



FT6;



FT3, 40;



FT4, 80, 45;

RELATED

INSTRUCTIONS:

LT, Line Type

PT, Pen Thickness

UF, User-Defined Fill Type

ERRORS:

Condition	Error	Plotter Response
more than 3 parameters	2	uses first 3 parameters
number out of range	3	uses parameter of previous FT instruction, or default if no previous FT instruction

LT, Line Type

USE: Specifies the line pattern to be used when drawing lines and nonsolid fill types. Use LT to emphasize or de-emphasize plotted lines and shapes.

SYNTAX: LT *pattern number* (,*pattern length*);
or
LT;

Parameter	Format	Range	Default
pattern number	integer	-6 to +6	no parameter, solid line
pattern length	real	0 to 100* percentage	4% of the distance between P1 and P2

*Practical range; allowable range is -8 388 608 to 8 388 607.

REMARKS: Once you issue an LT instruction with a *pattern length* parameter, the pattern length remains in effect until you change it with another LT instruction. When you issue subsequent LT instructions, you can omit the pattern length parameter if you want to use the same one. If you omit all parameters (LT;) the plotter uses the default values.

For example, the previous LT was (LT3,10,) and the one you just issued is (LT-5,). The plotter will use the pattern length from the first instruction and implement (LT-5,10,). If (LT-5,) is what you want to implement, you must send (LT;LT-5,).

- **No Parameters** — Draws a solid line. This is *not* the same as sending (LT0,).
- **Pattern Number** — Produces the corresponding line pattern. Line patterns can be of fixed or adaptive type, as explained below.

Positive pattern numbers (1-6) are fixed line types and use the actual specified pattern length to draw lines. Any unused part of the pattern is carried over into the next line, unless you have issued an PU instruction. The PU instruction clears the carry-over portion of the pattern segment; the next line will be drawn from the beginning of the pattern.

A **zero (0) pattern number** draws dots at the PA and/or PR coordinate points only.

Negative pattern numbers (-1 - -6) are adaptive line types. The pattern length is automatically adjusted so that each line contains one or more complete patterns.

NOTE: Do not use an adaptive line type for drawing circles, arcs, wedges, or polygons using these shapes. The plotter will attempt to draw the complete pattern in every chord. (There are 72 chords in a circle using the default chord angle.) ■

- **Pattern Length** — Specifies the length of one complete line pattern as a percentage of the diagonal distance between the scaling points P1 and P2. If you don't specify a length, the plotter uses the last value specified. If you haven't previously specified a pattern length, the plotter uses the default of 4%.

8 _____
5 _____
4 _____
3 _____
2 _____
1 .
0 . **specifies dots only at plotted points** .
-1 .
-2 _____
-3 _____
-4 _____
-5 _____
-6 _____
No parameter (Default Value) _____

EXAMPLE:

Fixed LT6;



Adaptive LT-6;

Dots LT1;

Dots LT0;

RELATED**INSTRUCTIONS:** AA, Arc Absolute
AR, Arc Relative
CI, Circle
EP, Edge Polygon
FP, Fill Polygon
FT, Fill Type
PA, Plot Absolute
PD, Pen Down
PR, Plot Relative
RA, Fill Rectangle Absolute
RR, Fill Rectangle Relative
WG, Fill Wedge
UF, User-Defined Fill Type

ERRORS:

Condition	Error	Plotter Response
more than 2 parameters	2	uses first 2 parameters
number out of range	3	ignores instruction

PT, Pen Thickness

USE: Determines the spacing between the parallel lines in solid fill patterns, according to the pen tip thickness.

SYNTAX: PT *pen thickness*;
or
PT;

Parameter	Format	Range	Default
pen thickness	real	0.1 to 5.0 millimetres	0.3 millimetres

REMARKS: Pen tip width is an average width. Use the PT instruction to compensate for slight differences in the width of pen tips.

The PT instruction pertains only to the currently selected pen. You must issue the SP instruction before the PT instruction.

- **No Parameter** — Sets the default thickness of 0.3 mm.
- **Pen Thickness** — Represents the physical pen tip width in millimetres. The tip width is stamped on the end of each pen (e.g., 0.3 or 0.7).

The pen thickness determines the spacing of lines needed to produce a solid fill. A wider pen thickness allows wider gaps between the lines to fill an area (and thus, fewer strokes). If your solid fill has gaps showing between the lines, specify a smaller pen thickness parameter than is true for your pen. If you want to plot faster but with a less dense solid fill, use a larger pen thickness than your pen requires.

EXAMPLE: The following rectangles were drawn by the same pen. From left to right they show a pen thickness of 0.3 (the actual thickness of the pen), a pen thickness of 0.7 (more than twice the thickness of the pen), and a pen thickness of 0.9.



PT.3;



PT.7;



PT.9;

RELATED

INSTRUCTIONS: FP, Fill Polygon

FT, Fill Type

RA, Fill Rectangle Absolute

RR, Fill Rectangle Relative

WG, Wedge Fill

ERRORS:

Condition	Error	Plotter Response
more than 1 parameter	2	uses first parameter
number out of range	3	ignores instruction

RA, Fill Rectangle Absolute

USE: Defines and fills a rectangle using absolute coordinates. Use this instruction to fill rectangular shapes in bar charts, logos, and other plots. To outline a rectangle using absolute coordinates, use the EA instruction.

SYNTAX: RA X,Y;

Parameter	Format	Range	Default
X,Y coordinates	current units	-8 388 608 to 8 388 607	none

REMARKS: Defines and fills a rectangle using the current pen, the current line and fill types, and absolute plotting. The RA instruction includes an automatic pen down. When the rectangle is complete, the pen returns to the starting point (restores current pen location) and restores the pen position (up/down).

- **X,Y Coordinates** — Specify the corner of the rectangle that is diagonally opposite from the current pen location, which is the starting point of the rectangle. Coordinates are interpreted in current units: as user units when scaling is on, as plotter units when scaling is off.



NOTE: The illustration shows the current pen location in the lower-left corner and the instruction's X,Y coordinates in the upper-right corner. However, both positions can be in any corner as long as the X,Y coordinates are in the corner diagonally opposite the current pen location. ■

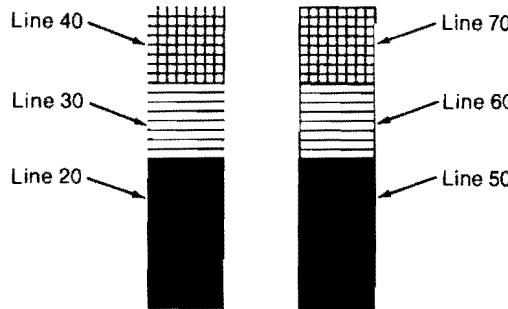
The RA instruction clears the polygon buffer and then uses it to define the rectangle before drawing. A rectangle requires enough buffer space to hold five points. The default buffer size is sufficient. Refer to Chapter 8 for more information on buffers.

EXAMPLE: This program uses RA with three different fill types (refer to the FT instruction) to create rectangles such as those you might use in a bar chart. The rectangles in the right bar are edged using the EA instruction.

```

10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;PA400,400;RA800,1200;" 
30  PRINT #1, "PA400,1200;FT3,50;RA800,1600;" 
40  PRINT #1, "PA400,1600;FT4;RA800,2000;" 
50  PRINT #1, "PA1200,400;FT;RA1600,1200;EA1600,1200;" 
60  PRINT #1, "PA1200,1200;FT3,50;RA1600,1600;EA1600,1600;" 
70  PRINT #1, "PA1200,1600;FT4;RA1600,2000;EA1600,2000;" 
80  PRINT #1, "SP0;" 
90  END

```



RELATED

INSTRUCTIONS: EA, Edge Rectangle Absolute

ER, Edge Rectangle Relative

RR, Fill Rectangle Relative

ERRORS:

Condition	Error	Plotter Response
1 parameter	2	ignores instruction
more than 2 parameters	2	uses first 2 parameters only
number out of range	3	ignores instruction
polygon buffer overflow	7	ignores excess data

RR, Fill Rectangle Relative

USE: Defines and fills a rectangle using relative coordinates. Use this instruction to fill rectangular shapes in bar charts, logos, and other plots. To outline a rectangle using relative coordinates, use the ER instruction.

SYNTAX: RR $X,Y;$

Parameter	Format	Range	Default
X,Y increments	current units	-8 388 608 to 8 388 607	none

REMARKS: Defines and fills a rectangle using the current pen, the current line and fill types, and absolute plotting. The RR instruction includes an automatic pen down. When the rectangle is complete, the pen returns to the starting point (restores current pen location) and restores the pen position (up/down).

- 5
- **X,Y Increments —** Specify the corner of the rectangle that is diagonally opposite from the current pen location, which is the starting point of the rectangle. Coordinates are interpreted in current units: as user units when scaling is on, as plotter units when scaling is off.



NOTE: The illustration shows the current pen location in the lower-left corner and the instruction's X,Y increments in the upper-right corner. However, both positions can be in any corner as long as the X,Y increments are in the corner diagonally opposite the current pen location. ■

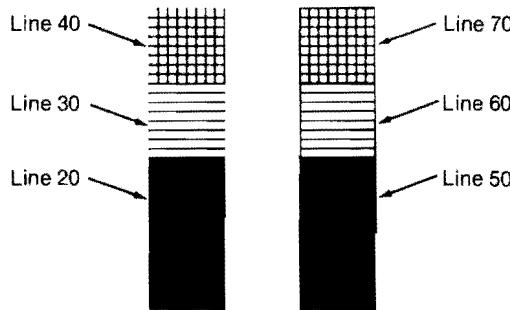
The RR instruction clears the polygon buffer and then uses it to define the rectangle before drawing. A rectangle requires enough buffer space to hold five points. The default buffer size is sufficient. Refer to Chapter 8 for more information on buffers.

EXAMPLE: This program uses RR with three different fill types (refer to the FT instruction) to create rectangles such as those you might use in a bar chart. The rectangles in the right bar are edged using the ER instruction.

```

10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;PA400,400;RR400,800;""
30  PRINT #1, "PR0,800;FT3,50;RR400,800;""
40  PRINT #1, "PR0,400;FT4;RR400,400;""
50  PRINT #1, "PA1200,400;FT;RR400,800;ER400,800;""
60  PRINT #1, "PR0,800;FT3,50;RR400,400;ER400,400;""
70  PRINT #1, "PR0,400;FT4;RR400,400;ER400,400;""
80  PRINT #1, "SP0;""
90  END

```



RELATED

INSTRUCTIONS: ER, Edge Rectangle Relative
RA, Fill Rectangle Absolute
EA, Edge Rectangle Absolute

ERRORS:

Condition	Error	Plotter Response
1 parameter	2	ignores instruction
more than 2 parameters	2	uses first 2 parameters only
number out of range	3	ignores instruction
polygon buffer overflow	7	ignores excess data

SM, Symbol Mode

USE: Draws the specified symbol at each X,Y coordinate point. Use symbol mode to create scattergrams, indicate points on geometric drawings, and differentiate data points on multi-line graphs.

SYNTAX: SM *character* (,*character**);

or

SM ;

Parameter	Format	Range	Default
character	label	most printing characters (decimal codes 33-58 and 60-126)**	none

**Decimal code 59 (the semicolon) is an HP-GL terminator and cannot be used as a symbol *in any character set*. Use it only to cancel symbol mode (e.g., SM;).

REMARKS: When you use the PA, PD, PR, and PU instructions, SM draws the specified symbol at each X,Y coordinate point. The SM instruction includes an automatic pen down; after the symbol is drawn, the pen position is restored.

- **No Parameter** — Terminates symbol mode.
- **Character** — Draws the specified character at each subsequent X,Y coordinate, centered over the point. The symbol is drawn in addition to the usual function of each plotting instruction.

The character is drawn in the character set selected at the time the SM instruction is executed. The character does not change even if you subsequently select a new character set. If you have defined downloadable characters in set -1, you can use these in symbol mode plotting. (Refer to the downloadable character (DL) instruction.) Also, the size (SI and SR), slant (SL), and direction (DI and DR) instructions affect how the character is drawn.

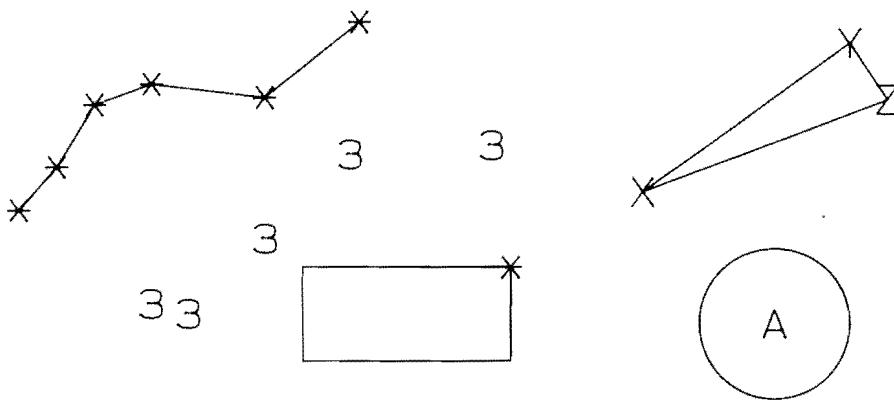
An SM instruction remains in effect until another valid SM instruction is executed or the plotter is initialized or set to default conditions.

EXAMPLE: The following program shows several uses of symbol mode: with the pen down for a line graph, with the pen up for a scattergram, and with the pen down for geometric drawings.

NOTE: Symbol mode works only with the PA/PR instructions. Notice that the circle and rectangle have symbols only for the PA instruction coordinate point. ■

*Used only with Kanji characters. Refer to Appendix D for details about Kanji characters.

```
10  *Insert configuration statement here
20  PRINT #1, "IN;SP1;SC0,1,0,1,2;"
30  PRINT #1, "SM*:PA0,1000;PD200,1230,400,1560;"
40  PRINT #1, "PD700,1670,1300,1800,2000;PU;"
50  PRINT #1, "SM3;PA700,500,900,450,1300,850;"
60  PRINT #1, "PA1750,1300,2500,1350;PU;SM;"
70  PRINT #1, "PA3300,1100;PD;SMY;PA4400,1890;SMZ;"
80  PRINT #1, "PA4600,1590;SMX;PA3300,1100;PU;"
90  PRINT #1, "SMA;PA4000,400;CI400;"
100 PRINT #1, "SM*:PA2600,700;EA1500,200;"
110 PRINT #1, "SP0;"
120 END
```



RELATED

INSTRUCTIONS: PA, Plot Absolute
PD, Pen Down
PR, Plot Relative
PU, Pen Up

ERRORS:

Condition	Error	Plotter Response
number out of range	3	ignores instruction

TL, Tick Length

USE: Specifies the length of the tick marks produced by the tick instructions (XT and YT). Use this instruction to adjust both the positive and negative portions of tick marks, and to establish a tick length for drawing grids.

SYNTAX: TL *positive tick (,negative tick);*

or

TL;

Parameter	Format	Range	Default
positive tick length	real	-100 to 100* percentage	0.5% of $ P2_x - P1_x $ and $ P2_y - P1_y $
negative tick length	real	-100 to 100* percentage	0.5% of $ P2_x - P1_x $ and $ P2_y - P1_y $

*Practical range; allowable range is -8 388 608 to 8 388 607.

REMARKS: Allows you to specify the length of tick marks and whether or not they are drawn on both sides of the axis. The TL instruction does not itself draw ticks. You must use the YT or XT instructions to actually draw ticks.

You can draw grids easily using the XT and YT instructions. To draw grid lines, set the positive tick length to 100 (i.e., *TL100* ;).

- **No Parameters** — Reestablishes the default values.
- **Positive Tick** — Specifies as a percentage of the P1/P2 distance, the length of the tick mark drawn to the *positive* side of the X- and Y-axes. When used with the X-tick (XT) instruction, the length is a percentage of the absolute value of the vertical scale ($|P2_y - P1_y|$). When used with the Y-tick (YT) instruction, the length is a percentage of the absolute value of the horizontal scale ($|P2_x - P1_x|$).

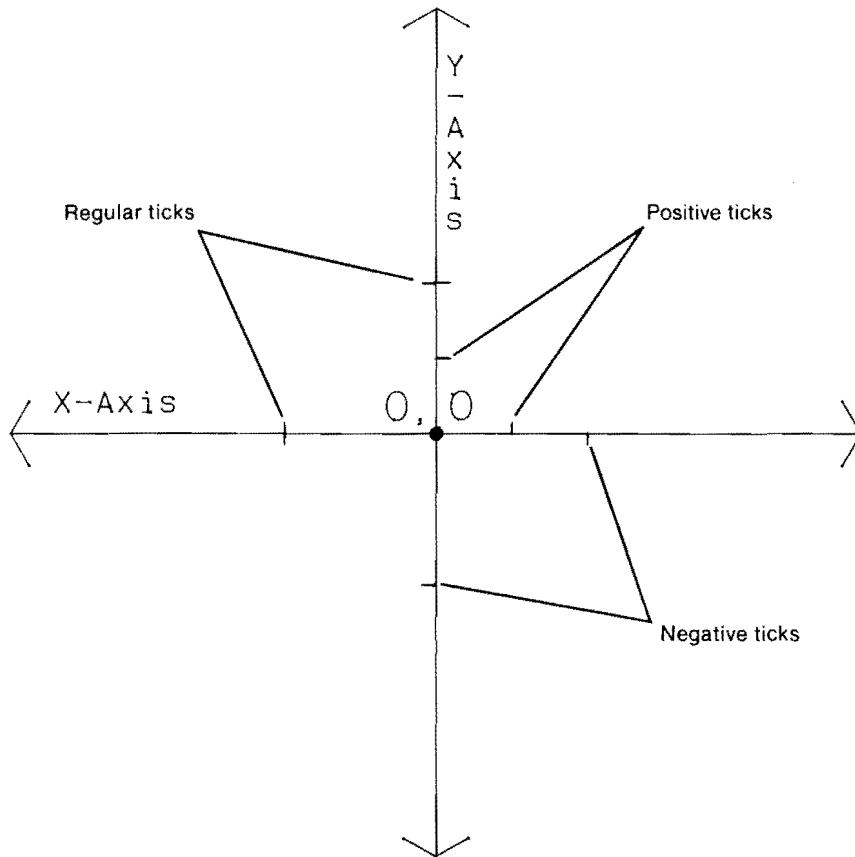
If you specify only the positive tick length, the tick will only be drawn on the positive side of the axis.

- **Negative Tick** — Specifies as a percentage of the P1/P2 distance, the length of the tick mark drawn to the *negative* side of the X- and Y-axes. When used with the X-tick (XT) instruction, the length is a percentage of the absolute value of the vertical scale ($|P2_y - P1_y|$). When used with the Y-tick (YT) instruction, the length is a percentage of the absolute value of the horizontal scale ($|P2_x - P1_x|$).

To draw negative ticks only, you must specify zero (0) as the positive tick length.

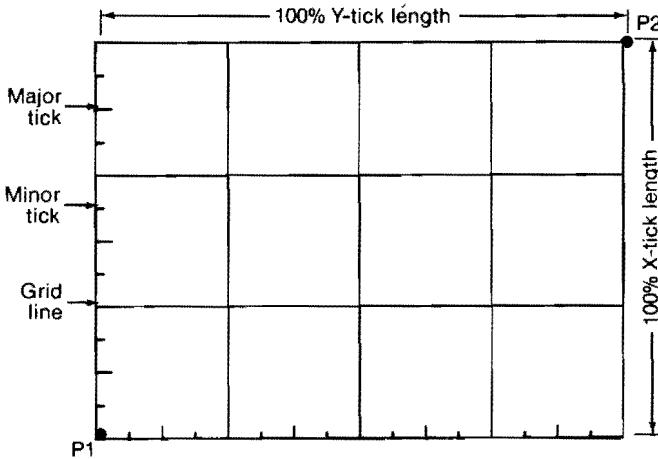
Because tick length is a percentage of P1/P2 distance, X-ticks and Y-ticks will be the same length only when scaling is isotropic. A TL instruction remains in effect until another TL instruction is issued, or the plotter is initialized or set to default conditions.

The following illustration shows regular ticks, positive ticks, and negative ticks on both the X- and Y-axis.



EXAMPLE: This program shows grid lines, X-ticks and Y-ticks. Vary the length of the ticks to indicate major and minor divisions.

```
10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;IP200,1000,3000,3100;""
30  PRINT #1, "SC0,40,0,30;""
35  PRINT #1, "PA0,30;EA40,0;""
40  FOR I=1 TO 3
50  PRINT #1, "PD;PR0,-2.5;TL1.5;YT;PR0,-2.5;""
60  PRINT #1, "TL3;YT;PR0,-2.5;TL1.5;YT;""
70  PRINT #1, "PR0,-2.5;TL100;YT;""
80  NEXT I
90  FOR J=1 TO 4
100 PRINT #1, "TL100;XT;PR2.5,0;TL1.5;XT;""
110 PRINT #1, "PR2.5,0;TL3;XT;PR2.5,0;""
120 PRINT #1, "TL1.5,0;XT;PR2.5,0;""
130 NEXT J
140 PRINT #1, "SP0;""
150 END
```



RELATED

INSTRUCTIONS: SC, Scale
XT, X-Tick
YT, Y-Tick

ERRORS:

Condition	Error	Plotter Response
more than 2 parameters	2	uses first 2 parameters only
number out of range	3	ignores instruction

UF, User-Defined Fill Type

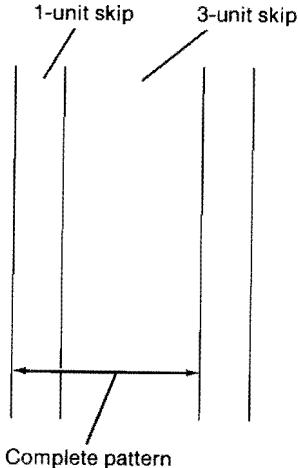
USE: Defines a fill pattern composed of 'gaps' between parallel lines such as a semilog or candy-stripe effect. All fill instructions (FP, RA, RR, and WG) can use this fill type.

SYNTAX: UF *gap1, gap2, ..., gap20*;
or
UF;

Parameter	Format	Range	Default
gap	real	0 to 8 388 607	none

REMARKS: Allows you to define the gaps between lines of the area fill. The UF instruction does not itself select a fill type. To access this fill pattern, use the fill type (FT) instruction with a parameter of 5 or 6.

- **No Parameter** — Establishes solid fill for the user-defined fill type.
- **Gap** — Represents the number of units to the next fill line. That is, if you issue (UF1, 3;), the plotter will draw 1 fill line, skip 1 unit, draw 1 fill line, and skip 3 units.



The size of the units is determined by the number of gaps you specify in the UF instruction combined with the spacing parameter of the FT instruction. The plotter takes the total number of gaps you specify in the UF instruction and fits them into the spacing defined in the FT instruction. So the spacing parameter in the FT instruction becomes the pattern repeat length for the UF instruction.

The pattern repeat length always begins at the plotter-unit origin (0, 0), no matter how your plot is scaled. Depending on where in the plotting area your shape begins, your fill pattern may or may not begin with the same gap parameter.

The following illustration shows the same essential fill pattern in 2 pattern lengths. Both rectangles are filled using the default spacing parameter of the FT instruction. The one on the left has a UF instruction of (UF3,5;) while the one on the right is (UF3,5,3,5;).



Since the pattern for the left box is 3,5—the plotter fits 8 units ($3+5=8$) into the pattern repeat length. On the other hand, the pattern for the right box is 3,5,3,5—so the plotter fits 16 ($3+5+3+5=16$) units into the same pattern repeat length.

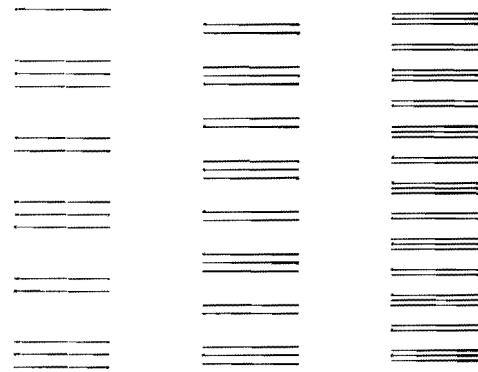
NOTE: On the DraftMaster II, the FR instruction maintains registration of fill patterns across frame boundaries. This is done by referencing the fill pattern with the plotter-unit origin in the first frame drawn, which will have a different plotter-unit value (but the same current-unit value if scaling is on) after the FR is executed. ■

You can specify up to 20 gap parameters in one UF instruction. Each parameter must be positive. You may include parameters equal to zero (0), but the sum of the parameters must be greater than zero.

Lines are drawn using the current pen and line type. Because of the way gaps are defined, positive parameters in the LT instruction do not always produce the alternating fill pattern described under the FT instruction. Therefore, you may find that negative line-type parameters produce more satisfactory results, and solid lines look even better with user-defined fill type. Lines are either bidirectional or unidirectional, depending on whether you specify parameter 5 or 6 for the FT instruction.

EXAMPLE: The following shows the difference in the appearance of a UF instruction created by decreasing the spacing parameter in the FT instruction to produce shorter pattern repeat lengths.

```
10  *Insert configuration statement here
20  PRINT #1, "IN;500,1,0,1,2;SP1;PA1600,0;""
30  PRINT #1, "UF1,4,1,4,1;""
40  PRINT #1, "FT5,750,0;RR500,2000;""
50  PRINT #1, "FT5,500,0;PR1000,0;RR500,2000;""
60  PRINT #1, "FT5,300,0;PR1000,0;RR500,2000;""
70  PRINT #1, "SP0;""
80  END
```



If the pattern repeat length is small, so that spaces between clusters of lines are much greater than the spaces between lines in a cluster, you can create the appearance of different line widths.

```
10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;SC0,1,0,1,2;PA0,0;""
30  PRINT #1, "UF1,5,1,1,5,1,1,1,5,1,1,1,5;""
40  PRINT #1, "FT5,300,0;""
50  PRINT #1, "RA400,1200;""
60  PRINT #1, "SP0;""
70  END
```

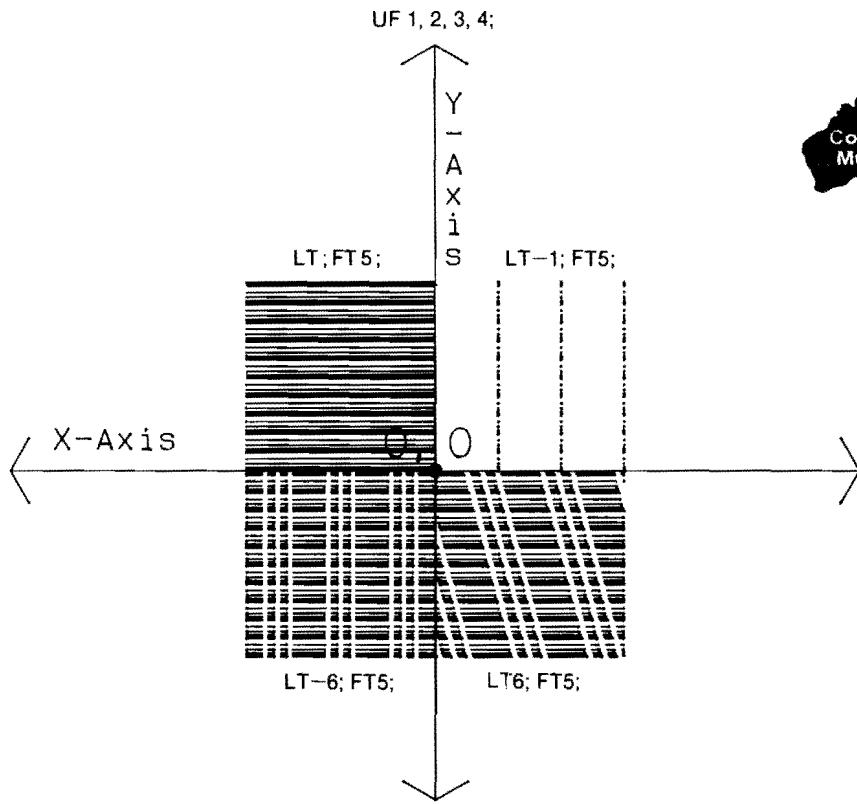


Or, you might want to create a semilog pattern, as illustrated below.

```
10  'Insert configuration statement here
20  PRINT #1, "IN;SC0,1,0,1,2;SP1;PA600,0;""
30  PRINT #1, "UF9,8,7,6,5,4,3,2,1;""
40  PRINT #1, "FT5,500,0;""
50  PRINT #1, "RA1000,1200;""
60  PRINT #1, "SP0;""
70  END
```



This illustration shows the effects of different line types on user-defined fill types.



5

RELATED

INSTRUCTIONS: FT, Fill Type
LT, Line Type

ERRORS:

Condition	Error	Plotter Response
more than 20 parameters	2	ignores extra parameters
number out of range or sum of parameters $\leqslant 0$	3	ignores instruction; executes previous UF instruction

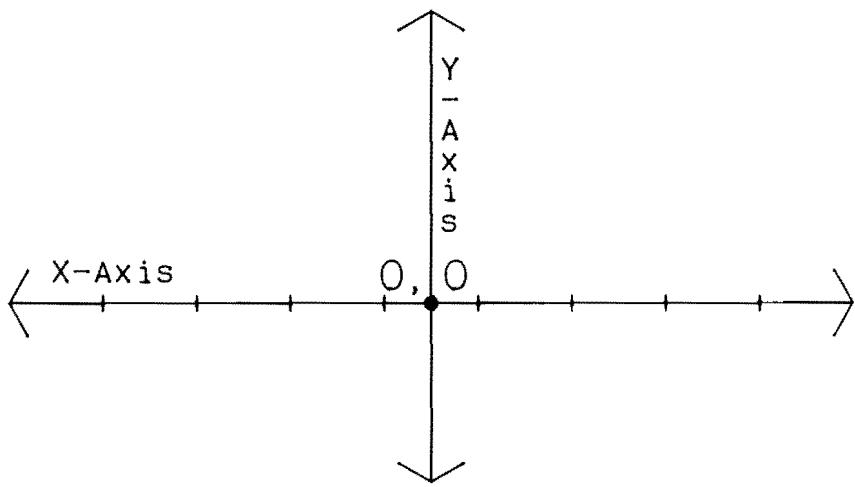
XT, X-Tick

USE: Draws a vertical (parallel to the Y-axis) tick at the current pen location. Use this instruction to draw vertical tick marks on axes, grid lines, or lines centered on or starting at the current pen location.

SYNTAX: XT;

REMARKS: The XT instruction includes an automatic pen down. After drawing the tick, the previous pen position is restored. The default length of the tick mark is 0.5% of $|P2_y - P1_y|$ for each positive and negative tick. Change the length of the positive and negative tick marks using the TL instruction.

EXAMPLE:



RELATED

INSTRUCTIONS: TL, Tick Length
YT, Y-Tick

ERRORS:

Condition	Error	Plotter Response
1 or more parameters	2	executes instruction

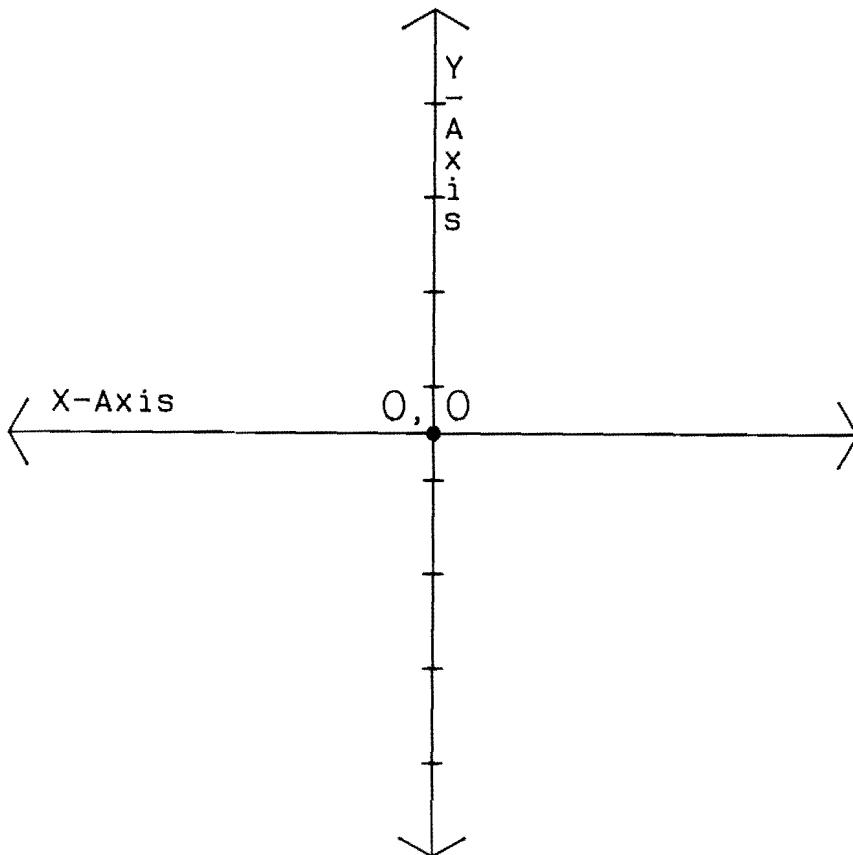
YT, Y-Tick

USE: Draws a horizontal (parallel to the X-axis) tick at the current pen location. Use this instruction to draw horizontal tick marks on axes, grid lines, or lines centered on or starting at the current pen location.

SYNTAX: YT;

REMARKS: The YT instruction includes an automatic pen down. After drawing the tick, the previous pen position is restored. The default length of the tick mark is 0.5% of $|P2_x - P1_x|$ for each positive and negative tick. Change the length of the positive and negative tick marks using the TL instruction.

EXAMPLE:



RELATED**INSTRUCTIONS:** TL, Tick Length
XT, X-Tick**ERRORS:**

Condition	Error	Plotter Response
1 or more parameters	2	executes instruction

Notes

5



6

Drawing Circles, Arcs, and Wedges

This chapter discusses some individual instructions that draw complete shapes such as circles, wedges, and arcs. Additionally it discusses instructions that control the smoothness of a circle or arc.

Instructions Covered

AA, Arc Absolute
AR, Arc Relative
CI, Circle
CT, Chord Tolerance
EW, Edge Wedge
WG, Fill Wedge

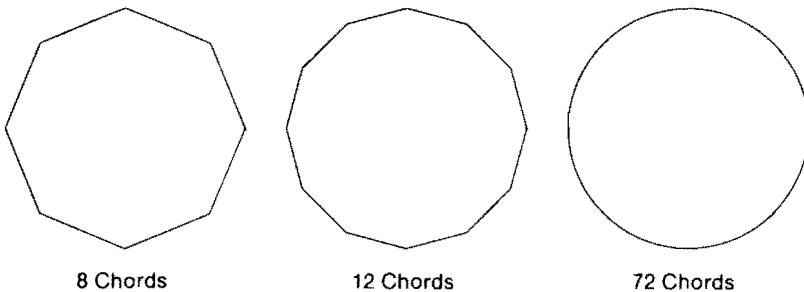
6

Terms You Should Understand

Anisotropic Scaling	Fill Type
Arc	Isotropic Scaling
Edge	Wedge
Fill	

Chords and Chord Tolerance

To draw curves the plotter draws a series of straight lines called chords to represent each arc segment. The apparent smoothness of the curve depends on the number of chords used to draw it; the more chords, the smoother the shape appears. The following illustration shows circles drawn with different numbers of chords.

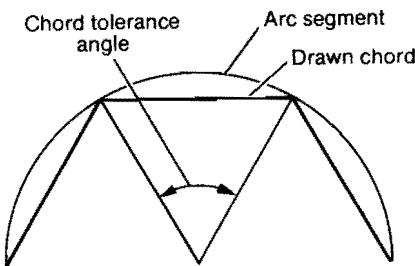


The number of chords allowed in any given curved shape is called the chord tolerance (i.e., the allowable deviation from a smooth curve). Many instructions allow you to establish the chord tolerance you want. You can use two methods to establish chord tolerance: chord angle and deviation distance. Use the chord tolerance (CT) instruction to select the method you want to use.

6

Chord Angle

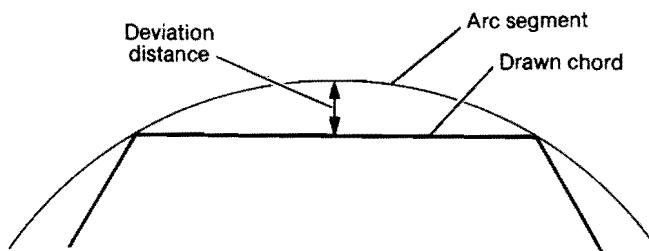
The chord angle method specifies, in degrees, the maximum angle created when lines from each end of the chord intersect the center point of the circle.



When you specify the chord tolerance as a chord angle, the curved shape will always have the same number of chords, regardless of its size. For example, a circle drawn with the default chord angle of 5 degrees will always have 72 chords. One result of this is that a large circle will be less smooth than a smaller circle with the same chord angle.

Deviation Distance

Deviation distance specifies, in current units, the maximum distance between the chord and the arc segment it represents.



When you specify a deviation distance, the plotter will adjust the number of chords in the curved shape to maintain the deviation distance. A small circle drawn with the deviation distance of 2 current units will have fewer chords than a large circle drawn with the same deviation distance.

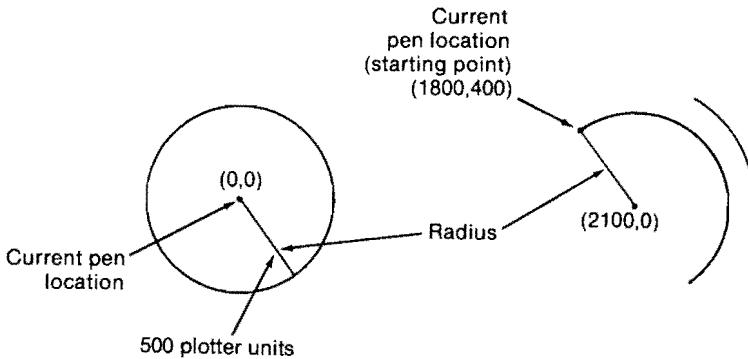
Drawing Circles and Arcs

Circles use the current pen location as their center point. You specify the radius and, optionally, the chord tolerance. Circles are defined by the circle (CI) instruction, which outlines a circle. (To draw filled circles, refer to the fill wedge (WG) instruction in this chapter or the polygon mode (PM) instruction in Chapter 8.)

Arcs use the current pen location as one end of the arc. You specify the center point (thus establishing the radius), the number of degrees through which you want the arc drawn, and the direction you want the arc drawn. You can draw arcs using both the arc absolute (AA) and the arc relative (AR) instruction.

The following illustration shows a simple program using CI and AA to draw a circle and an arc.

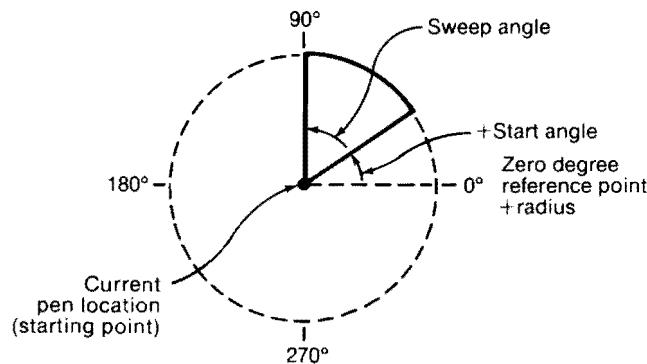
```
10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;PA0,0;""
30  PRINT #1, "CI500;PA1800,400;PD;AA2100.0,-180;""
40  PRINT #1, "SP0;""
50  END
```



Drawing Wedges

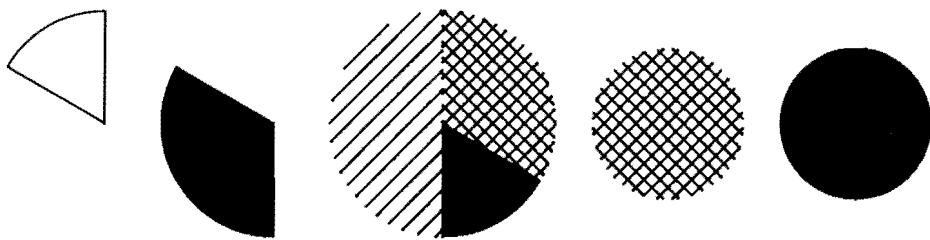
A wedge is a section of a circle. Wedges are commonly used to draw pie charts. You can draw filled or edged wedges using the fill wedge (WG) and edge wedge (EW) instructions.

Like the CI instruction, wedges use the current pen location as the center point. You specify the length of the radii, in current units, the start angle, and the sweep angle. The start angle is the number of degrees from the zero-degree reference point where you want to draw the first radius. The sweep angle is the number of degrees through which you want to draw the arc before drawing the second radius. You can also specify the arc's chord tolerance.



The following program draws an edged and a filled wedge. The program uses different fill types with wedges and the WG instruction with a 360-degree sweep angle to draw filled circles.

```
10  *Insert configuration statement here
20  PRINT #1, "IN;SP1;PA-3000,0;"
30  PRINT #1, "EW600,90,60;"
40  PRINT #1, "PA-2100,0;WG600,150,120;"
50  PRINT #1, "PA-1200,0;FT3,75,45;WG600,90,180;"
60  PRINT #1, "FT1,0,0;WG600,270,60;"
70  PRINT #1, "FT4,60,45;WG600,330,120;"
80  PRINT #1, "PA0,0;WG400,0,360;"
90  PRINT #1, "PA1000,0;FT;WG400,0,360;"
100 PRINT #1, "SP0;"
110 END
```



AA, Arc Absolute

USE: Draws an arc, using absolute coordinates, that starts at the current pen location and uses the specified center point.

SYNTAX: AA *X,Y,arc angle (, chord angle);*

Parameter	Format	Range	Default
X,Y coordinates	current units	-8 388 608 to 8 388 607	none
arc angle	real	-360 to 360 degrees**	none
chord tolerance chord angle*	real	0.36 to 180 degrees**	5 degrees
chord deviation	current units	-8 388 608 to 8 388 607	5 degrees***

*Chord angle is the default interpretation of chord tolerance.

**Practical range; allowable range is -8 388 608 to 8 388 607.

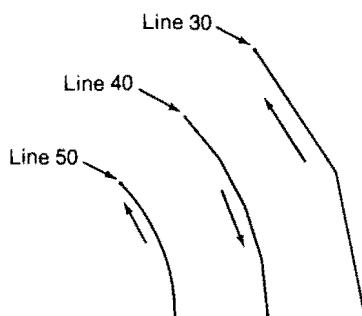
***If no deviation distance is specified, the chord tolerance defaults to a chord *angle* of 5 degrees.

REMARKS: Draws the arc starting at the current pen location using the current pen position (up or down) and line type. After drawing the arc, the pen location remains at the end of the arc.

- **X,Y Coordinates** — Specify the absolute location of the center of the circle that would be drawn if the arc were 360 degrees.
- **Arc Angle** — Specifies (in degrees) the angle through which the arc is drawn. A positive angle draws counterclockwise from the current pen location, and a negative angle draws clockwise.
- **Chord Tolerance** — Specifies the chord tolerance used to draw the arc. The default is a chord angle of 5 degrees. Refer to the *Chords and Chord Tolerance* discussion at the beginning of this chapter or the chord tolerance (CT) instruction for information on setting and determining chord tolerance.

EXAMPLE:

```
10 'Insert configuration statement here
20 PRINT #1, "IN:SP1;PA4000,1000;"
30 PRINT #1, "PD:AA2000,1000,45,25;"
40 PRINT #1, "PU3050,2060;PD:AA2000,1000,-45,10;"
50 PRINT #1, "PU3000,1000;PD:AA2000,1000,45;"
60 PRINT #1, "SP0;"
70 END
```



6

RELATED

INSTRUCTIONS: AR, Arc Relative

CI, Circle

CT, Chord Tolerance

LT, Line Type

ERRORS:

Condition	Error	Plotter Response
1 or 2 parameters	2	ignores instruction
more than 4 parameters	2	uses first 4 parameters
number out of range	3	ignores instruction

AR, Arc Relative

USE: Draws an arc, using relative coordinates, that starts at the current pen location and uses the specified center point.

SYNTAX: AR *X,Y,arc angle (,chord angle);*

Parameter	Format	Range	Default
X,Y increments	current units	-8 388 608 to 8 388 607	none
arc angle	real	-360 to 360 degrees**	none
chord tolerance chord angle*	real	0.36 to 180 degrees**	5 degrees
chord deviation	current units	-8 388 608 to 8 388 607	5 degrees***

*Chord angle is the default interpretation of chord tolerance.

**Practical range: allowable range is -8 388 608 to 8 388 607.

***If no deviation distance is specified, the chord tolerance defaults to a chord angle of 5 degrees.

REMARKS: Draws the arc starting at the current pen location using the current pen position (up or down) and line type. After drawing the arc, the pen location remains at the end of the arc.

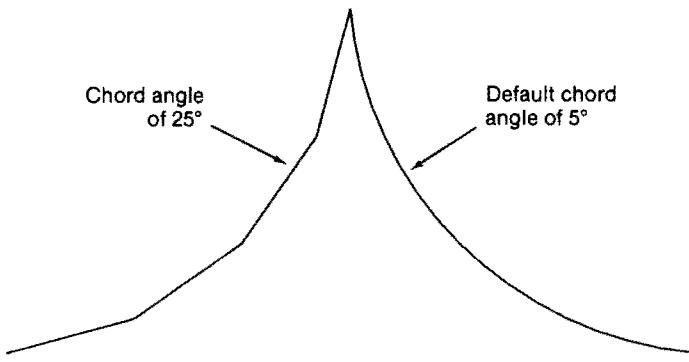
- **X,Y Increments** — Specify, relative to the current location, the center of the circle that would be drawn if the arc were 360 degrees.
- **Arc Angle** — Specifies (in degrees) the angle through which the arc is drawn. A positive angle draws counterclockwise from the current pen location, and a negative angle draws clockwise.
- **Chord Tolerance** — Specifies the chord tolerance used to draw the arc. The default is a chord angle of 5 degrees. Refer to the *Chords and Chord Tolerance* discussion at the beginning of this chapter or the chord tolerance (CT) instruction for information on setting and determining chord tolerance.

EXAMPLE:

```

10 'Insert configuration statement here
20 PRINT #1, "IN;SP1;PA10,10;PD;"
30 PRINT #1, "AR0,2000,80,25;AR2000,0,80;"
40 PRINT #1, "SP0;"
50 END

```

**RELATED****INSTRUCTIONS:** AA, Arc Absolute

CI, Circle

CT, Chord Tolerance

LT, Line Type

ERRORS:

Condition	Error	Plotter Response
1 or 2 parameters	2	ignores instruction
more than 4 parameters	2	uses first 4 parameters
number out of range	3	ignores instruction

CI, Circle

USE: Draws a circle using the specified radius and chord tolerance. If you want a filled circle, refer to the WG or PM instruction.

SYNTAX: CI *radius(, chord tolerance);*

Parameter	Format	Range	Default
radius	current units	-8 388 608 to 8388607	none
chord tolerance	real	0.36 to 180 degrees**	5 degrees
chord angle*	real	0.36 to 180 degrees**	5 degrees***
chord deviation	current units	-8 388 608 to 8 388 607	5 degrees***

*Chord angle is the default interpretation of chord tolerance.

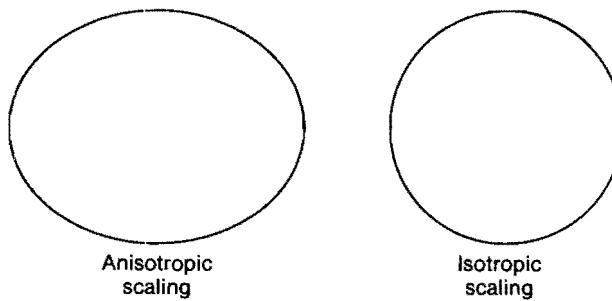
**Practical range: allowable range is -8 388 608 to 8 388 607.

***If no deviation distance is specified, the chord tolerance defaults to a chord angle of 5 degrees.

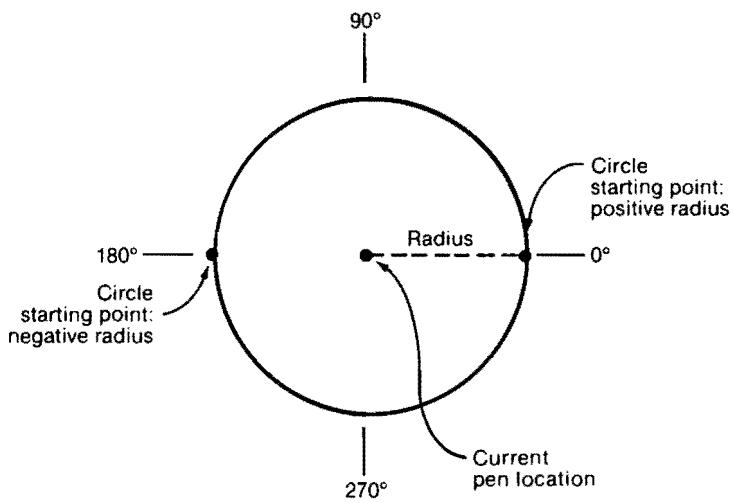
REMARKS: The CI instruction includes an automatic pen down. When a CI instruction is received, the pen lifts, moves from the center of the circle (the current pen location) to the starting point on the circumference, lowers the pen, draws the circle, then returns with the pen up to the center of the circle. After the circle is drawn, the previous pen position (up or down) is restored. To avoid leaving dots in the center of the circle, move to and from the circle's center with the pen up.

Each chord of the circle is drawn using the currently defined line type. Do not use an adaptive (negative) line type to draw a circle as the plotter will attempt to draw a complete pattern for **every** chord (72 in the default chord angle mode).

Always use isotropic scaling in plots that draw circles. Refer to the discussion of scaling in Chapter 3 for more information.



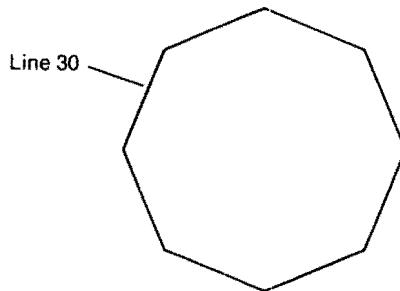
- **Radius** — Measured in current units from the current pen location. The sign (+ or -) defines the starting point of the circle: a positive radius begins at 0 degrees; a negative radius begins at 180 degrees.



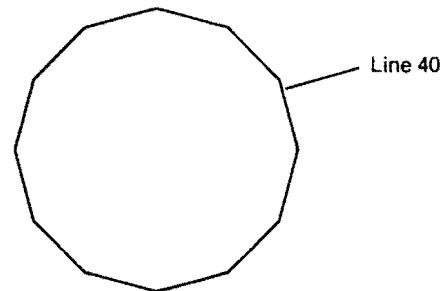
- **Chord Tolerance** — Specifies the chord tolerance used to draw the arc. The default is a chord angle of 5 degrees. Refer to the *Chords and Chord Tolerance* discussion at the beginning of this chapter or the chord tolerance (CT) instruction for information on setting and determining chord tolerance.

EXAMPLE: Effects of Chord Angle on Circle Smoothness

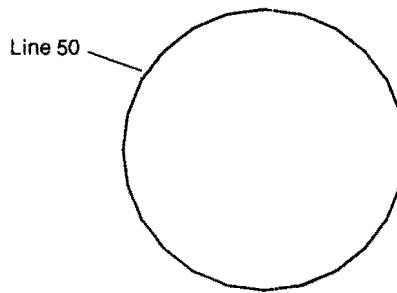
```
10 'Insert configuration statement here
20 PRINT #1, "IN;SP1;SC-3906,3906,-3131,3131,1;"
30 PRINT #1, "PA-1700,2000;CI750,45;"
40 PRINT #1, "PA300,2000;CI750,30;"
50 PRINT #1, "PA-1700,-200;CI750,15;"
60 PRINT #1, "PA300,-200;CI750;"
70 PRINT #1, "SP0;"
80 END
```



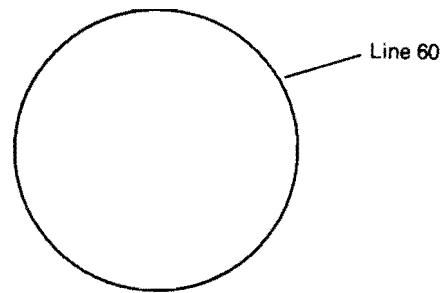
45-Degree chord angle



30-Degree chord angle



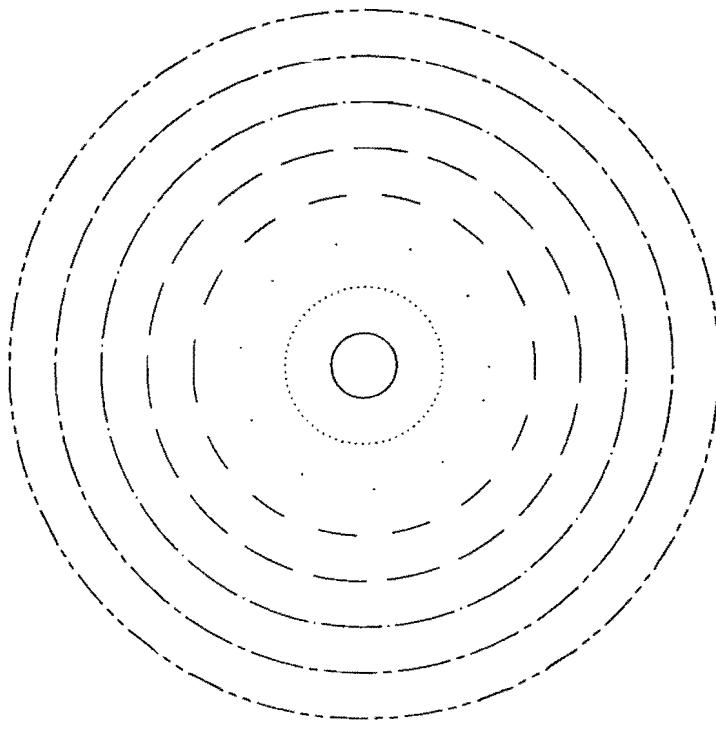
15-Degree chord angle



5-Degree chord angle

Drawing Circles with Different Radii and Line Types

```
10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;SC-100,100,-100,100,1;""
30  PRINT #1, "PA0,0;LT;CI5;LT0;CI-12;LT1;CI19;""
40  PRINT #1, "LT2;CI-26;LT3;CI33;LT4;CI-40;""
50  PRINT #1, "LT5;CI47;LT6;CI54;""
60  PRINT #1, "SP0;"
```



RELATED

INSTRUCTIONS: CT, Chord Tolerance
LT, Line Type
SC, Scale
WG, Fill Wedge

ERRORS:

Condition	Error	Plotter Response
more than 2 parameters	2	uses first 2 parameters
number out of range	3	ignores instruction

CT, Chord Tolerance

USE: Determines whether the chord tolerance parameter of the CI, AA, AR, EW, WG instructions is interpreted as a chord angle in degrees or as a deviation distance in current units.

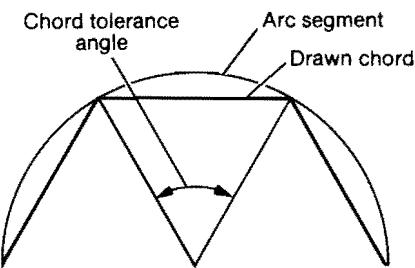
SYNTAX: CT *n*;
or
CT;

Parameter	Format	Range	Default
<i>n</i>	integer	0 or 1	0

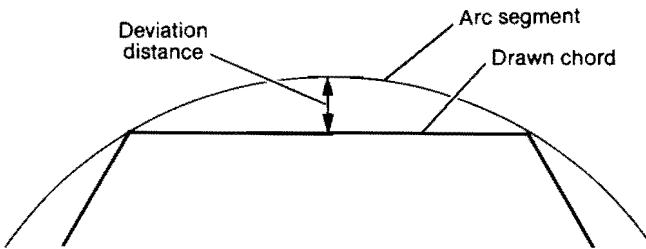
REMARKS: A plotted circle or arc actually consists of a series of straight line segments, or chords that represent arc segments. Increasing the number of chords increases the smoothness of the circle. Chord tolerance is defined as the acceptable deviation from a smooth circle and can be established as either a chord angle or a deviation distance.

- **No Parameter —** Equivalent to (CT0,) and resets chord tolerance to chord angle.

- **n** — Specifies the type of chord tolerance.
- 0 Sets the chord tolerance to chord angle. The chord angle specifies, in degrees, the maximum angle created when lines from each end of the chord intersect the center point of the circle. When chord tolerance is specified as chord angle, the circle will always have the same number of chords, irrespective of its size.



- 1 Sets chord tolerance to deviation distance. Deviation distance specifies, in current units, the maximum distance between the chord and the arc segment it represents. When you specify a deviation distance, the number of chords in the circle will vary with its size.



RELATED

INSTRUCTIONS:

CI, Circle
AA, Arc Absolute
AR, Arc Relative
EW, Edge Wedge
WG, Fill Wedge

ERRORS:

Condition	Error	Plotter Response
more than 1 parameter	2	executes first parameter
number not 0 or 1	3	ignores instruction

EW, Edge Wedge

USE: Outlines any wedge. Use this instruction to draw sectors of pie charts.

SYNTAX: EW *radius, start angle, sweep angle (, chord angle);*

Parameter	Format	Range	Default
radius	current units	-8 388 608 to 8 388 607	none
start angle	real	0 to 360 degrees*	none
sweep angle	real	0 to 360 degrees*	none
chord tolerance chord angle**	real	0.36 to 50 degrees**	5 degrees
chord deviation	current units	-8 388 608 to 8 388 607	5 degrees***

*Chord angle is the default interpretation of chord tolerance.

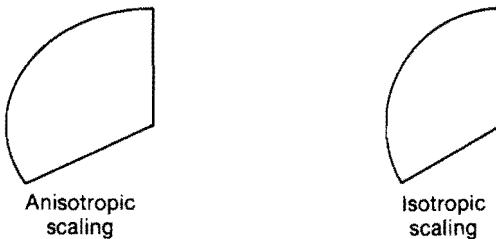
**Practical range: allowable range is -8 388 608 to 8 388 607.

***If no deviation distance is specified, the chord tolerance defaults to a chord angle of 5 degrees.

REMARKS: The EW instruction defines and edges a wedge using the current pen and a solid line. The EW instruction includes an automatic pen down. After the wedge is drawn, the pen returns to the previous pen location and the pen position is restored.

The only difference between the EW instruction and the WG instruction is that the EW instruction produces an outlined wedge and the WG, a filled one. It is good programming practice to outline filled areas *after* they have been filled; this creates a clean edge.

Always use isotropic scaling in plots that draw wedges. Refer to the discussion of scaling in Chapter 3 for more information.



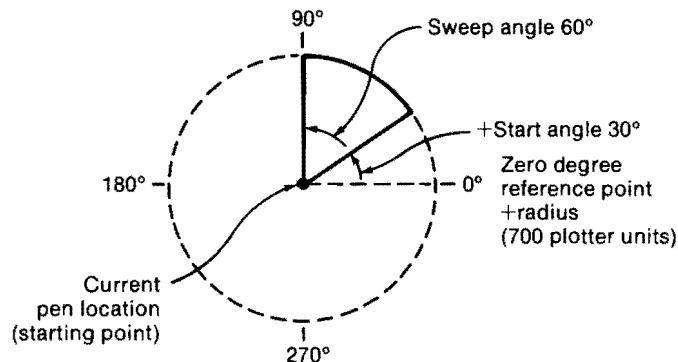
- **Radius** — Specifies the distance from the current pen location to the start of the wedge's arc. Since the wedge is a portion of a circle, this parameter is the radius of the circle. It specifies the distance from the current pen location (which becomes the center of the circle), to any point on the circumference of the circle.

The radius is interpreted in current units: as user units when scaling is on, as plotter units when scaling is off. The sign of the radius determines the location of the zero-degree reference point. The illustration following the parameter descriptions shows the location of the zero-degree reference point for a positive and a negative radius.

- **Start Angle** — Specifies the beginning point for the arc as the number of degrees from the zero-degree reference point. A positive start angle positions the radius counterclockwise from the zero-degree reference point; a negative start angle positions the radius clockwise from the zero-degree reference point.
- **Sweep Angle** — Specifies the number of degrees through which the arc is drawn. A positive sweep angle draws the arc counterclockwise; a negative sweep angle draws the arc clockwise. If you specify a sweep angle greater than 360 degrees, a 360 degree angle is used.
- **Chord Tolerance** — Specifies the chord tolerance used to draw the arc. The default is a chord angle of 5 degrees. Refer to the *Chords and Chord Tolerance* discussion at the beginning of this chapter or the chord tolerance (CT) instruction for information on setting and determining chord tolerance.

NOTE: The number of chords per arc is limited to ~200 if you use the default size of the polygon buffer. To use a larger number of chords, use the GM or **ESC . T** instruction (explained in Chapters 8 and 15) to increase the size of the polygon buffer. ■

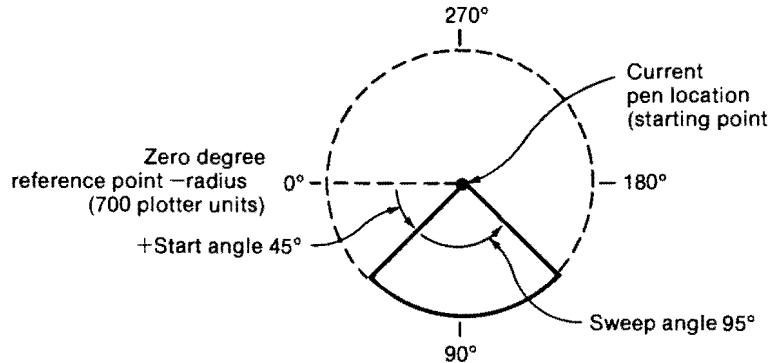
EW 700, 30, 60;



Positive Radius



EW -700, 45, 95;



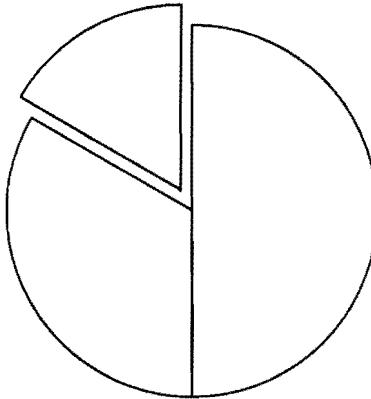
Negative Radius

EXAMPLE:

```

10 'Insert configuration statement here
20 PRINT #1, "IN:SP1;SC-3906,3906,-3131,3131,1;"
30 PRINT #1, "PA0,0;"
40 PRINT #1, "EW-1000,90,180;"
50 PRINT #1, "EW-1000,330,120;"
60 PRINT #1, "PR-60,110;"
70 PRINT #1, "EW-1000,270,60;"
80 PRINT #1, "SP0;"
90 END

```

**RELATED**

INSTRUCTIONS: CT, Chord Tolerance
 GM, Graphics Memory
 SC, Scale
 WG, Fill Wedge

ERRORS:

Condition	Error	Plotter Response
fewer than 3 parameters	2	ignores instruction
more than 4 parameters	2	uses first 4 parameters
number out of range	3	ignores instruction
polygon buffer overflow	7	edges buffer contents

WG, Fill Wedge

USE: Defines and fills any wedge. Use this instruction to draw filled sectors of a pie chart.

SYNTAX: WG *radius, start angle, sweep angle (, chord angle);*

Parameter	Format	Range	Default
radius	current units	-8 388 608 to 8 388 607	none
start angle	real	0 to 360 degrees*	none
sweep angle	real	0 to 360 degrees*	none
chord tolerance chord angle**	real	0.36 to 50 degrees**	5 degrees
chord deviation	current units	-8 388 608 to 8 388 607	5 degrees***

*Chord angle is the default interpretation of chord tolerance.

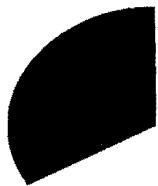
**Practical range: allowable range is -8 388 608 to 8 388 607.

***If no deviation distance is specified, the chord tolerance defaults to a chord *angle* of 5 degrees.

REMARKS: The WG instruction defines and fills a wedge using the current pen, fill type, and line types. The WG instruction includes an automatic pen down. After the wedge is drawn, the pen returns to the previous pen location and the pen position is restored.

The only difference between the WG instruction and the EW instruction is that the WG instruction produces a filled wedge and the EW, an outlined one. It is good programming practice to outline filled areas *after* they have been filled; this creates a clean edge.

Always use isotropic scaling in any plot that draws wedges. Refer to the discussion of scaling in Chapter 3 for more information.



Anisotropic scaling



Isotropic scaling

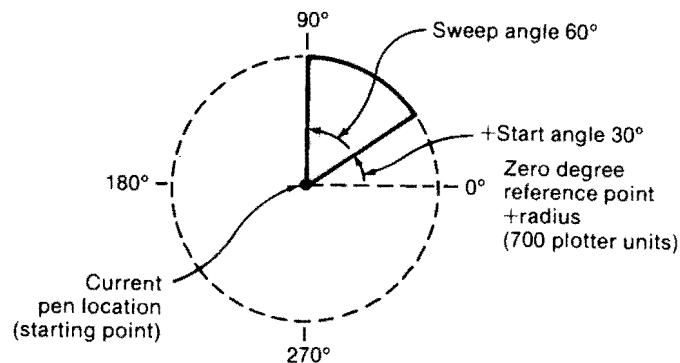
- **Radius** — Specifies the distance from the current pen location to the start of the wedge's arc. Since the wedge is a portion of a circle, this parameter is the radius of the circle. It specifies the distance from the current pen location (which becomes the center of the circle), to any point on the circumference of the circle.

The radius is interpreted in current units: as user units when scaling is on, as plotter units when scaling is off. The sign of the radius determines the location of the zero-degree reference point. The illustration following the parameter descriptions shows the location of the zero-degree reference point for a positive and a negative radius.

- **Start Angle** — Specifies the beginning point of the arc as the number of degrees from the zero-degree reference point. A positive start angle positions the radius counterclockwise from the zero-degree reference point; a negative start angle positions the radius clockwise from the zero-degree reference point.
- **Sweep Angle** — Specifies the number of degrees through which the arc is drawn. A positive sweep angle draws the arc counterclockwise; a negative sweep angle draws the arc clockwise. If you specify a sweep angle greater than 360 degrees, a 360 degree angle is used.
- **Chord Tolerance** — Specifies the chord tolerance used to draw the arc. The default is a chord angle of 5 degrees. Refer to the *Chords and Chord Tolerance* discussion at the beginning of this chapter or the chord tolerance (CT) instruction for information on setting and determining chord tolerance.

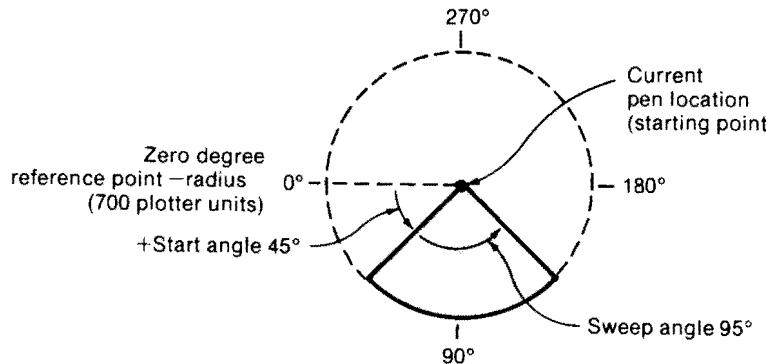
NOTE: The number of chords per arc is limited to ~200 if you use the default size of the polygon buffer. To use a larger number of chords, use the GM or **ESC .T** instruction (explained in Chapters 8 and 15) to increase the size of the polygon buffer. ■

WG 700, 30, 60;



Positive Radius

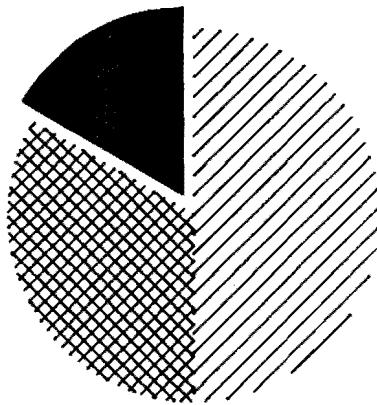
WG -700, 45, 95;



Negative Radius

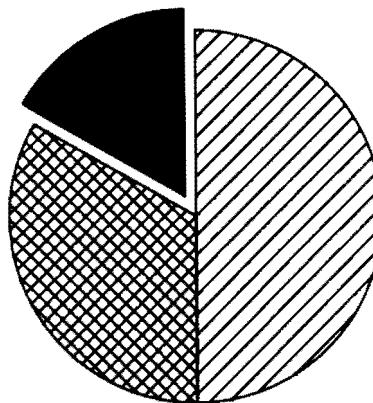
EXAMPLE:

```
10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;SC-3906,3906,-3131,3131,1;""
30  PRINT #1, "PA0,0;FT3,75,45;WG-1000,90,180;""
40  PRINT #1, "FT4,60,45;WG-1000,330,120;""
50  PRINT #1, "PR-60,110;FT1,0,0;""
60  PRINT #1, "WG-1000,220,60;""
70  PRINT #1, "SP0;""
80  END
```



The same pie chart with each sector edged.

```
10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;SC-3906,3906,-3131.3131,1;""
30  PRINT #1, "PA0,0;FT3,75,45;WG-1000,90,180;""
35  PRINT #1, "EW-1000,90,180;""
40  PRINT #1, "FT4,60,45;WG-1000,330,120;""
45  PRINT #1, "EW-1000,330,120;""
50  PRINT #1, "PR-60,110;FT1,0,0;""
60  PRINT #1, "WG-1000,270,60;""
65  PRINT #1, "EW-1000,270,60;""
70  PRINT #1, "SP0;""
80  END
```



RELATED

INSTRUCTIONS: EP, Edge Polygon

EW, Edge Wedge

FT, Fill Type

GM, Graphics Memory

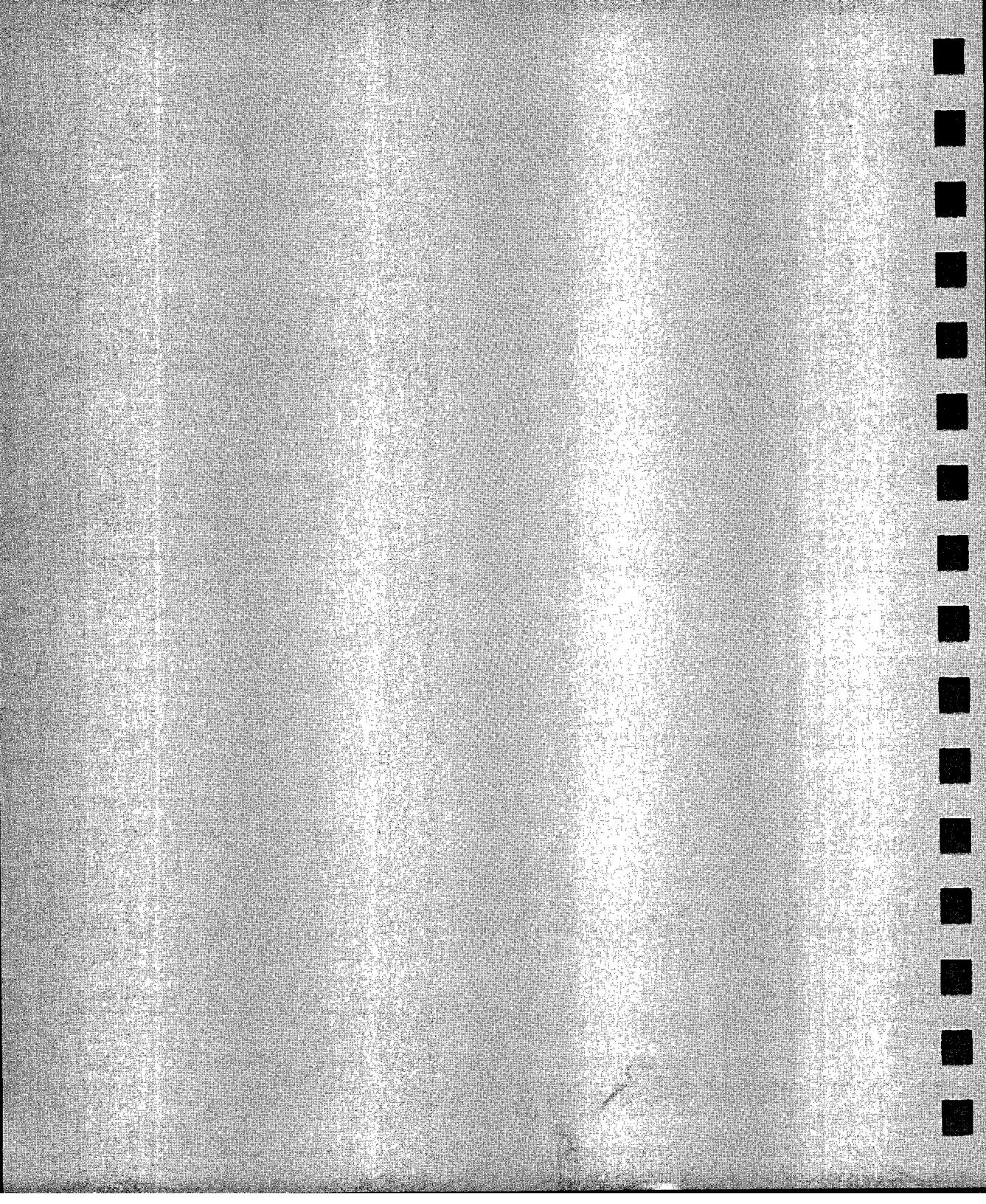
LT, Line Type

SC, Scale

ERRORS:

Condition	Error	Plotter Response
less than 3 parameters	2	ignores instruction
more than 4 parameters	2	uses first 4 parameters
number out of range	3	ignores instruction
polygon buffer overflow	7	ignores instruction

Notes



7

Labeling Your Plots

Labels are an important part of graphic communication. This chapter presents basic labeling techniques of:

- drawing labels
- positioning labels
- changing the label size, slant, and direction

These techniques are illustrated using the plotter's default character set (ANSI ASCII English). All of the instructions in this chapter are valid with the plotter's other character sets and fonts. Chapter 13 discusses the plotter's other character sets and fonts.

Instructions Covered

BL, Buffer Label
CP, Character Plot
DI, Absolute Direction
DR, Relative Direction
DT, Define Label Terminator
DV, Direction Vertical
ES, Extra Space
LB, Label
LO, Label Origin
PB, Print Buffered Label
SI, Absolute Character Size
SL, Character Slant
SR, Relative Character Size

Terms You Should Understand

Chord Angle
Current Units

Using and Terminating Labels

You can use labels to create text charts or to emphasize areas of a graphics chart or plot that need special attention or explanation. You can control virtually all facets of the label's appearance; for example, its position, size, slant, and direction. You can also use the label buffer to repeatedly draw a label.

The label instructions (LB and BL) tell the plotter to draw everything following the instruction instead of trying to interpret the characters as graphics instructions. Since the semicolon (;) is normally a text character, you must use a special 'label terminator' to tell the plotter to return to interpreting characters as graphics instructions.

As we discussed in Chapter 3, the label terminator is the nonprinting ASCII end-of-text character **ETX** (decimal code 3). In the examples, the BASIC character string function **CHR\$(3)** is used to terminate labels. You must use the label terminator, or the plotter will draw the rest of your program on your plot. You can change the label terminator using the define label terminator instruction (DT). Read the next section for more information about inserting ASCII control characters in your labels.

Carriage Returns and Line Feeds

7
You can also insert carriage returns and line feeds in your labels. Carriage returns and line feeds are also nonprinting ASCII control characters. You can insert these formatting characters using a character string function like **CHR\$** or by producing them directly from the keyboard. The examples in this manual use the BASIC **CHR\$** function.

When you use a string function such as **CHR\$**, you must separate it from the label string to prevent the plotter from drawing it. To do this, you close the quotation marks, then insert the string function. To ensure that the character produced by the string function is sent immediately after the label string without any extra space, you may need to link the label and the string function with a concatenation symbol such as '+', '&', ',', or ';'. For example, in **BASIC—...label"+CHR\$(3)**. Use the symbol your system and language requires.

Check the table in Appendix A for the complete list of the plotter's responses to ASCII control characters. Some common **CHR\$** equivalents are:

CHR\$(3)	produces ETX (end-of-text)
CHR\$(10)	produces LF (line feed)
CHR\$(13)	produces CR (carriage return)

On some computers, you can produce these ASCII control characters from the keyboard by pressing two keys simultaneously. Check your computer documentation for the availability of this option and for the appropriate combination of keys on your system. Some common combinations and their equivalents are:

CONTROL and C	produces ETX
CONTROL and J	produces LF
CONTROL and M	produces CR

If you enter the character directly from the keyboard, you can enter the character within the label; you do not need to close quotes or use a concatenation symbol as you do with the CHR\$ function.

Instructions Using String Function

```
"LBLLabel"+CHR$(3)  
or  
"LBLLabel"+CHR$(13)+CHR$(10)+"return"+CHR$(3)
```

Equivalent Instructions Using Keyboard

```
"LBLLabel\x"  
or  
"LBLLabel\x\return\x"
```

Both methods produce the following labels:

Label

Label
return

The Carriage Return Point

The current pen location when the label (LB) instruction is issued becomes the carriage return point. If the plotter encounters a carriage return (**CR**, CHR\$(13), or **CP;**), the pen moves to the carriage return point, adjusted up or down by any line feeds or inverse line feeds. (Note that (**CP;**) includes a line feed.)

The carriage return point is updated to the current pen location by these instructions: AA, AR, DI, DR, DV, IN, PA, PR, RO. In addition, moving the pen with the front panel buttons or using the front-panel **CLEAR** or **RESET** updates the carriage return point.

The LB instruction does **not** update the carriage return point, just the current pen location. This feature allows you to issue several label instructions that write one long label and still use a carriage return to get to the beginning of the entire label.

Variables in Labels

You may want to use numeric or string variables in your label instructions instead of writing the numbers or text in each instruction. The principles for sending variables are similar to those we discussed in Chapter 2 for sending variables in other graphics instructions. However, the punctuation mark (delimiter) you use to separate the variable from the instruction is especially important.

As you remember, many computer languages use the quotation mark (single or double) to define the literal (actual) characters to be sent. The graphics instructions are enclosed in quotation marks because the computer program itself does nothing with them; since they're in quotation marks, it just sends them along to the plotter.

Variables are not enclosed in quotation marks because the computer program works with them—it substitutes the variable definition for the variable when it issues the instruction. Further, the variables are separated from the quotation marks by another punctuation mark, typically a comma or semicolon, called a delimiter.

The delimiter determines the format that the number is printed in and can have a major effect on the appearance of your label.

- **Commas** typically cause numbers and text to be right-justified in a specific character field width. (The field width may be different for numbers and text.) For example, if the character field width is 15 characters, a 5-digit number will be indented 10 spaces to be printed in spaces 11 through 15.
- **Semicolons** typically suppress the extra blank spaces and are recommended for use in labels. However, in many computer languages, numbers will still be printed with a blank space in front for the sign and one after for separation. (Consult the control character chart in Appendix A for the backspacing control characters.) Text, however, will be printed with no spaces between the variable strings.

Any blank spaces that you need within a label should be sent within the quotation marks as part of the character string.

The following illustrations show the use of variables with a comma, a semicolon, and spaces within the quotation marks (hyphens are used for visual clarity). The vertical line indicates the current pen location when the label instruction was issued.

Numeric Variables

```
10  *Insert configuration statement here
20  PRINT #1, "IN;SP1;SI.187,.269;PA-3000,1000;"  
30  X=50
40  PRINT #1, "LB",X,X+1,X+2,+CHR$(3)
50  PRINT #1, "PA-3000,500;"  
60  Y=50
70  PRINT #1, "LB";Y;Y+1;Y+2;+CHR$(3)
80  PRINT #1, "PA-3000,0;"  
90  Z=50
100 PRINT #1, "LB";Z;"----";Z+1;"----";Z+2;+CHR$(3)
110 PRINT #1, "PA-3000,1000;WG20,0,360;"  
120 PRINT #1, "PD-3000,0;WG20,0,360;"  
130 PRINT #1, "SP0;"  
140 END
```

7

•	50	51	52		
•	50	51	52		
•	50	----	51	----	52

String Variables

NOTE: This version of GW-BASIC uses such a wide character field width with commas that the 'AND STILL MORE' is plotted off the right-hand side of the page. ■

```
10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;SI.187,.269;PA-3000,1000;"  
30  X$="VARIABLE"  
40  Y$="MORE TEXT"  
50  Z$="AND STILL MORE"  
60  PRINT #1, "LB",X$,Y$,Z$,+CHR$(3)  
70  PRINT #1, "PA-3000,500;"  
80  PRINT #1, "LB";X$;Y$;Z$;+CHR$(3)  
90  PRINT #1, "PA-3000,0;"  
100 PRINT #1, "LB";X$;"----";Y$;"----";Z$;+CHR$(3)  
110 PRINT #1, "PA-3000,1000;WB20,0,360;"  
120 PRINT #1, "PD-3000,0;WB20,0,360;"  
130 PRINT #1, "SP0;"  
140 END
```

• VARIABLE MORE TEXT AND S
• VARIABLEMORE TEXTAND STILL MORE
• VARIABLE----MORE TEXT----AND STILL MORE

Default Label Settings

Whenever the plotter is initialized, the following label default settings are established. To change these settings, refer to the appropriate chapter or instruction.

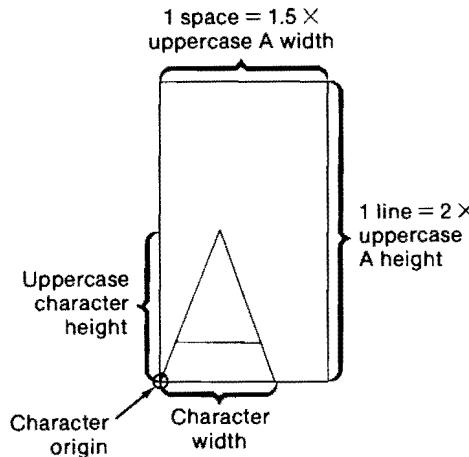
- **Character Set** — ANSI ASCII (English) fixed-space vector font (character set 0). Refer to Chapter 13 for information about the various plotter character sets.
- **Label terminator** — ASCII end-of-text character **ETX** (decimal code 3). Refer to the **DT** instruction.

- **Label starting point** — Current pen location. Refer to the LO instruction.
- **Character size** — Width = 0.285 cm; height = 0.375 cm. Refer to SI and SR instructions.
- **Label direction** — Horizontal. Refer to the DI, DR, and DV instructions.
- **Space between characters and lines** — Normal. Refer to the ES instruction.
- **Character slant** — None (vertical). Refer to the SL instruction.

Understanding the Character Plot Cell

The character plot (CP) cell is the basis for almost all positioning and movement of labels. Think of the CP cell as a rectangular area around a character that includes blank areas above and to the right of the character. In a fixed-space font (such as the default font being discussed in this chapter), the CP cell is the same size for every character. (Refer to Chapter 13 for information on the CP cell in variable-space fonts.)

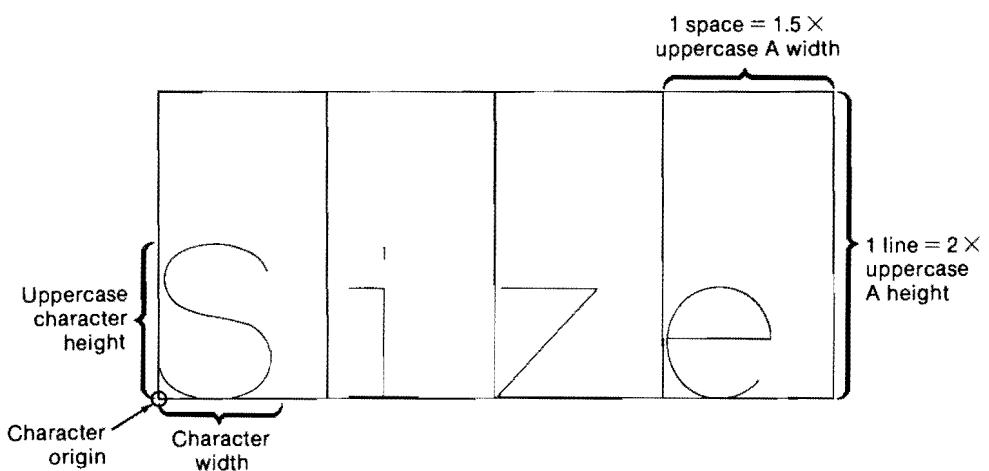
The CP cell is based on the uppercase 'A'. The CP cell height corresponds to a text line and is twice the height of the A. The CP cell width corresponds to a character space and is one and one-half times the width of the A. If you use the SI or SR instructions to change the size of the characters or use the ES instruction to add extra space around them, you will alter the size of the CP cell.



The current pen location when the label (LB) instruction is issued becomes the 'character origin' of the first CP cell as well as the carriage return point. (Unless you

have used the label origin (LO) instruction to alter the origin.) After drawing the first character, the pen moves to the character origin of the next CP cell and draws the next character. This continues until the end of the label unless an embedded control character (e.g., a carriage return) is encountered.

The character origin is the lower left corner of the CP cell. It is *not*, necessarily, the lower left corner of the character. Since the CP cell is based on an uppercase A, some of the lowercase characters begin slightly to the right of the left edge of the CP cell in order to maintain a reasonable appearance. Observe the location of the lowercase 'i' in the word 'Size' in the illustration below.



Enhancing Labels

You can enhance your labels by changing such aspects as the character size and slant, the space between characters and lines, and the orientation and/or placement of the label on the page. To use many of these enhancements, you must understand the character plot (CP) cell described previously.

Character Size and Slant

You can change the size of the characters using the absolute character size and the relative character size (SI and SR) instructions. The absolute character size (SI) instruction establishes the size of the uppercase A in centimetres and maintains this character size independently of the location of P1 and P2 or the media size. The relative character size (SR) establishes the size of the uppercase A as a percentage of the distance between P1 and P2. Subsequent changes in the location of P1 and P2 will cause the character size to change. Changing the character size changes the size of the CP cell.

You can use the character slant (SL) instruction to slant the characters at a specified angle in either direction from the left vertical side of the CP cell. The CP cell is not altered.

Character Spaces and Text Lines

You can use the extra space (ES) instruction to automatically put extra spaces or lines between all characters or lines. For example, you could use ES to skip a space between every character (for example, M E M O R A N D U M) or to skip a line between every line of text, double-spacing your text. You can also decrease the spacing between characters and lines.

You can use the character plot (CP) instruction to move the pen a specific number of lines or spaces (CP cells) from the current pen location. One use for the CP instruction would be indenting a label a certain number of spaces.

Label Orientation and Placement

You can place your labels anywhere on the page in almost any orientation you desire. The direction absolute (DI) instruction specifies the angle at which you want the characters drawn, independent of the location of P1 and P2. The direction relative (DR) instruction specifies the angle at which you want the characters drawn as a function of the P1 and P2 distance; thus, when P1 and P2 change, the angle changes to maintain the same orientation.

DI and DR allow you to plot labels at any angle with the letters in their normal side-by-side orientation. Direction vertical (DV) plots vertically with the letters stacked on top of each other (a carriage return and line feed places the next 'column' to the left of the previous one). DV was designed for use with Kanji (refer to Appendix D) but can be used to create interesting effects with any character set.

*DIRECTION
*DIRECTION
*DIRECTION
*DIRECTION
*DIRECTION
*DIRECTION
*DIRECTION
*DIRECTION

LABEL

D	I	V
R	E	E
C	T	T
I	I	C
O	A	N
N	L	L

Using the label origin (LO) instruction greatly simplifies placing labels on a plot. Normally the first character origin is the current pen location when the label instruction is issued. The LO instruction allows you to specify that the label be offset slightly (in any direction you desire), offset distinctly, centered, and right- or left-justified from the current pen location. For example, the following illustration shows four centered lines of text.

7 Lines of any length
can easily be
centered

without cumbersome calculations.

This plot was programmed with one X,Y coordinate pair, one LO instruction to center labels, and a carriage return and line feed after each line. An alternative method would involve calculating the length of the line in CP cells, dividing by two, and using the CP instruction to 'backspace' the required number of cells; and that's just the first line.

The LO instruction also works well for offsetting a label from a data point so that the point is not obscured. Many of the plotted examples in this manual use the LO instruction to label X,Y coordinate points and to offset the label slightly from a line.

BL, Buffer Label

USE: Stores a label in the label buffer. You can then use the output length (OL) instruction to determine its space requirement prior to drawing it. Or, you can use the plot buffered label (PB) instruction to repeatedly plot this label.

SYNTAX: BL *c...c CHR\$(3)*
or
BL *CHR\$(3)*

Parameter	Format	Range	Default
<i>c...c</i>	label	1 to 150 ASCII characters	none

REMARKS: The BL instruction does not draw labels. Once the label is stored in the buffer, you can print it using the PB instruction or size it using the OL instruction.

- **No Parameter** — Clears the label buffer. The buffer can also be cleared by a front-panel **RESET**, DF or IN instruction, or a LB instruction without a parameter.
- **c...c** — Can be up to 150 ASCII characters **including** control characters and the label terminator.
- **CHR\$(3)** — Consists of the default label terminator **ETX** (decimal 3) or the one you defined using the DT instruction.

A new BL or LB instruction will overwrite anything in the buffer. It is not necessary to clear it first.

NOTE: You cannot change the size of the label buffer. ■

The label origin (LO) instruction affects labels with embedded carriage returns that are stored with BL differently than the same labels plotted directly with the label (LB) instruction. The example below illustrates the difference.

EXAMPLE: These examples illustrate the difference between the BL and the LB instruction in the effect of the LO instruction on a label with embedded carriage returns. The examples both use label origin 14 (*LO14;*), which centers the label slightly above the current pen location.

The first example shows the LB instruction, where the LO instruction centers each segment of the label.

```

10  'Insert configuration statement here
20  PRINT #1, "IN:SP1;PA0,0;" 
30  PRINT #1, "LO14;LBEMBEDDED"+CHR$(13)+CHR$(10)
     +"CARRIAGE-RETURN"+CHR$(13)+CHR$(10)
     +"CHARACTERS"+CHR$(3)
40  PRINT #1, "SP0;" 
50  END

```

EMBEDDED
CARRIAGE-RETURN
CHARACTERS

Current pen location

This example shows the use of the BL instruction on the same label. The plotter positions the entire label over the current pen location as if it was enclosed in a rectangular box the length of the longest text line. In addition, each line within the box is left-justified.

```

10  'Insert configuration statement here
20  PRINT #1, "IN:SP1;PA0,0;" 
30  PRINT #1, "LO14;BLEMBEDDED"+CHR$(13)+CHR$(10)
     +"CARRIAGE-RETURN"+CHR$(13)+CHR$(10)
     +"CHARACTERS"+CHR$(3)
40  PRINT #1, "PB:SP0;" 
50  END

```

EMBEDDED-----
| CARRIAGE-RETURN |
| CHARACTERS |-----

Current pen location

RELATED

INSTRUCTIONS: DT, Define Label Terminator
LB, Label
LO, Label Origin
OL, Output Label Length
PB, Print Buffered Label

CP, Character Plot

USE: Moves the pen the specified number of character plot cells from the current pen location (e.g., to indent or center a label).

SYNTAX: CP *spaces, lines;*
or
CP;

Parameter	Format	Range	Default
spaces	real	-8 388 608 to 8 388 607	none
lines	real	-8 388 608 to 8 388 607	none

REMARKS: Use the CP instruction to position a label for indenting, centering, etc. For more information on spaces, lines, and the character plot cell, refer to *Understanding the Character Plot Cell* earlier in this chapter.

- **No Parameters** — Performs a carriage return and line feed, moving one line down and returning to the carriage return point.
- **Spaces** — Specifies the number of spaces (character plot cell widths) the pen will move relative to the current pen location.

Positive values specify the number of spaces the pen will move to the right of the current pen position; negative values specify the number of spaces the pen will move to the left. Right and left are relative to current label direction.

Note that the line parameter is **not** optional. If you only want to move a certain number of spaces (e.g., to backspace for an underscore), you must specify a zero (0) lines parameter. Likewise, you must specify a zero (0) space parameter if you only want to move lines but no spaces.

NOTE: If you are using a variable-space font, the average character plot cell will be used. Refer to Chapter 13 for information about variable-space fonts. ■

- **Lines** — Specifies the number of lines (character plot cell heights) the pen will move relative to the current pen location.

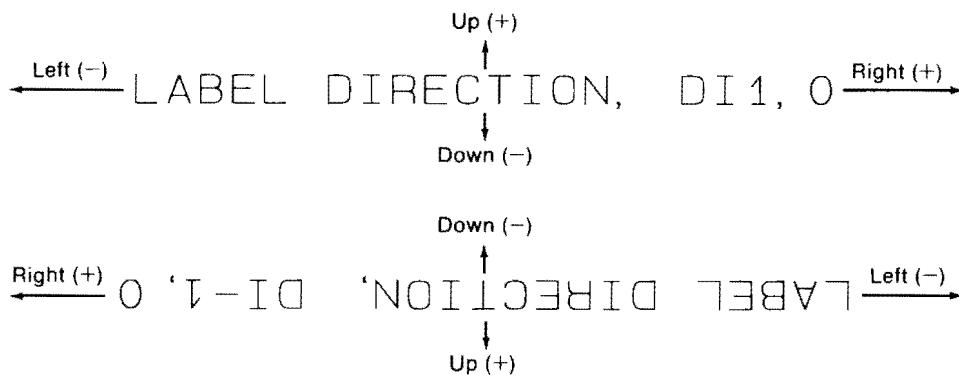
Positive values specify the number of lines the pen will move up from the current pen position; negative values specify the number of lines the pen will move down. Up and down are relative to the current label direction.

Note that the line parameter is **not** optional. If you only want to move a certain number of spaces (e.g., to backspace for an underscore), you must specify a zero (0) lines parameter. Likewise, you must specify a zero (0) space parameter if you only want to move lines but no spaces.

When you move the pen up or down a specific number of lines, the carriage return point will shift up or down accordingly.

NOTE: If you are using a variable-space font, the average character plot cell will be used. Refer to Chapter 13 for information about variable-space fonts. ■

The illustration below shows the effect of label direction on the positive and negative parameters.



CP moves the pen using the current pen position (up/down) and all moves are made with respect to the current character origin. CP *only affects the next label*, you must issue new CP instructions for subsequent labels.

EXAMPLE: This example uses the CP instruction to produce lettering along a line (but not directly on top of it) and to align labels along a left margin.

The CP instruction in line 30 moves the next label 0.35 lines up so it will be drawn just above the line. The PA instruction in line 40 establishes a new carriage return point (the current pen location when the LB instruction is issued); the X-tick marks the new

carriage return point. Notice that the CP instruction without parameters (CP;) in line 60 performs the same function as the carriage return and line feed (CHR\$(13)+CHR\$(10)) in line 50.

```

10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;PA1500,3000;PD-2000,3000;PU;""
30  PRINT #1, "CP5,.35;LBABOVE THE LINE"+CHR$(3)
40  PRINT #1, "PA-1000,3000;XT;""
50  PRINT #1, "CP0,-.95;LBBELOW THE LINE"+CHR$(13)
     +CHR$(10)+"WITH A NEAT"+CHR$(3)
60  PRINT #1, "CP;LBMARGIN"+CHR$(3)
70  PRINT #1, "SP0;""
80  END

```

ABOVE THE LINE

BELOW THE LINE

WITH A NEAT
MARGIN

RELATED

INSTRUCTIONS: DI, Absolute Direction
 DR, Relative Direction
 ES, Extra Space
 LO, Label Origin
 SI, Absolute Character Size
 SR, Relative Character Size

ERRORS:

Condition	Error	Plotter Response
1 parameter	2	ignores instruction
more than 2 parameters	2	uses first 2 parameters
number out of range	3	ignores instruction

DI, Direction Absolute

USE: Specifies the direction in which labels are drawn, independent of P1 and P2 settings. Use this instruction to change labeling direction when you are labeling curves in line charts, schematic drawings, blueprints, and survey boundaries.

SYNTAX: DI *run, rise;*
or
DI ;

Parameter	Format	Range	Default
run (or $\cos \theta$)	real	-8 388 608 to 8 388 607	1
rise (or $\sin \theta$)	real	-8 388 608 to 8 388 607	0

REMARKS: The P1 and P2 settings have no effect on the label direction. However, the direction vertical (DV) instruction interacts with the DI (and DR) instructions. The DV instruction places labels in either horizontal (normal) mode or vertical mode, where the letters are stacked on top of each other in a column.

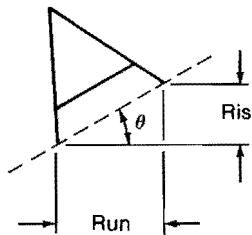
You can express the parameters in measured units as run and rise, or using the trigonometric functions cosine and sine according to the following relationships:

where: run and rise = number of measured units

θ = the angle measured in degrees

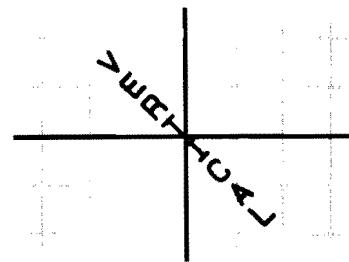
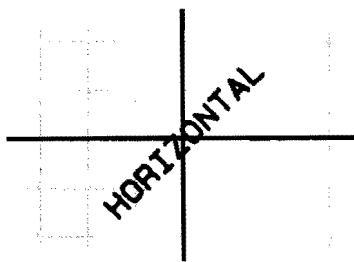
$$\frac{\sin\theta}{\cos\theta} = \frac{\text{rise}}{\text{run}}$$

$$\theta = \tan^{-1} \left(\frac{\text{rise}}{\text{run}} \right) \quad \text{and} \quad \tan \theta = \frac{\sin\theta}{\cos\theta}$$

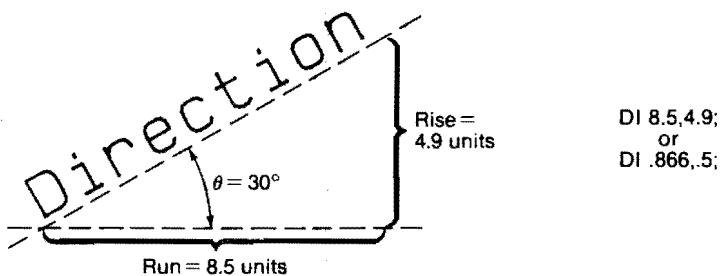


Note that the run and rise determine the slope or angle of an imaginary line under the base of *each* character in the label. Refer to the illustration above. When you are plotting in horizontal mode (you haven't used the DV instruction or have turned off vertical mode), the run and rise appear to determine the slope of the entire label. However, if you are plotting in vertical mode (*DVI_v*) where the letters are stacked vertically, even though the base of each letter is plotted on the same slope, the label will appear to slant in the opposite direction.

The horizontal mode and vertical mode illustrations below were plotted with the same run and rise parameters.



Suppose you want your label to be plotted in the direction shown in the following graph. You can do this in either of two ways: by measuring the run and rise or by measuring the angle.



To measure the run and rise, first draw a grid with the lines parallel to the X- and Y-axis. The grid units should be the same size on all sides, but their actual size is irrelevant. Then draw a line parallel to the label and one parallel to the X-axis. The lines should intersect to form an angle.

Select a point on the open end of your angle (where another line would create a triangle). On the line parallel to the X-axis, count the number of grid units from the intersection of the two lines to your selected point. This is the run. In the illustration above, the run is 8.5. Now, count the number of units from your selected point along a perpendicular line that intersects the line along the label. This is the rise. In the illustration above, the rise is 4.9.

Your DI instruction using the run and rise would be (*DI8.5,4.9;*).

If you know the angle (θ), you can use the trigonometric functions sine (sin) and cosine (cos). In this example, $\theta = 30$, $\cos 30 = 0.866$, and $\sin 30 = 0.5$.

Your DI instruction using the sin and cos would be (*DI.866,.5;*).

Whichever set of parameters you use, the label will be drawn in the same direction as shown in the illustration above.

A DI instruction remains in effect until another DI or DR instruction is executed, the plotter is initialized or set to default conditions.

When you know the angle and want to specify the cosine and sine values, you can use the table below, or you can use the SIN and COS functions available in most programming languages. (The example at the end of this section shows both methods.)

NOTE: The plotter does not understand exponential notation. If you compute the SIN and COS, you must use integer variables or a formatting technique to force rounding of the numbers. Refer to Number Formats in the *Using HP-GL with BASIC, FORTRAN, and Pascal* section of Chapter 1. ■

θ	Cosine	Sine
0	1	0
-30	0.87	-0.50
-45	0.71	-0.71
-60	0.50	-0.87
-90	0	-1
-120	-0.50	-0.87
-135	-0.71	-0.71
-150	-0.87	-0.50
-180	-1	0
-210	-0.87	0.50
-225	-0.71	0.71
-240	-0.50	0.87
-270	0	1
-300	0.50	0.87
-315	0.71	0.71
-330	0.87	0.50
-360	1	0

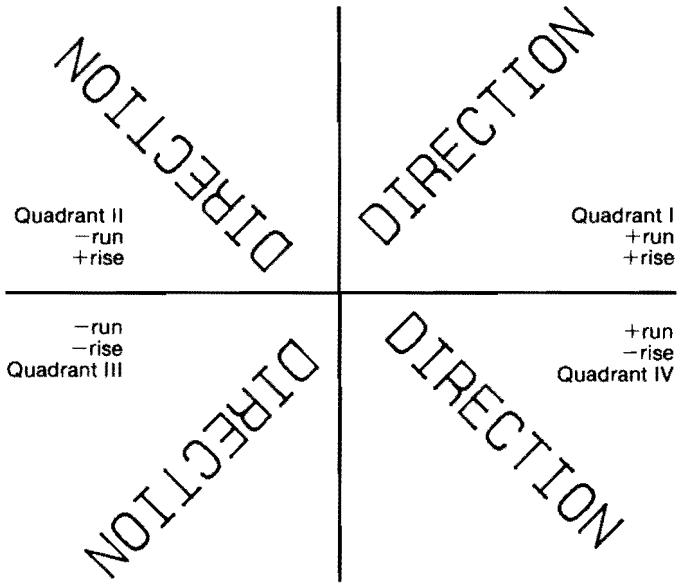
θ	Cosine	Sine
0	1	0
30	0.87	0.50
45	0.71	0.71
60	0.50	0.87
90	0	1
120	-0.50	0.87
135	-0.71	0.71
150	-0.87	0.50
180	-1	0
210	-0.87	-0.50
225	-0.71	-0.71
240	-0.50	-0.87
270	0	-1
300	0.50	-0.87
315	0.71	-0.71
330	0.87	-0.50
360	1	0

At least one parameter must be nonzero. The ratio of the parameters to each other is more important than the actual numbers. The table below lists three common label angles produced by using 1's and 0's.

DI Instruction	Label Direction
DI 1, 0	horizontal
DI 0, 1	vertical
DI 1, 1 or DI 0.7, 0.7 (any parameters equal to each other)	45-degree angle

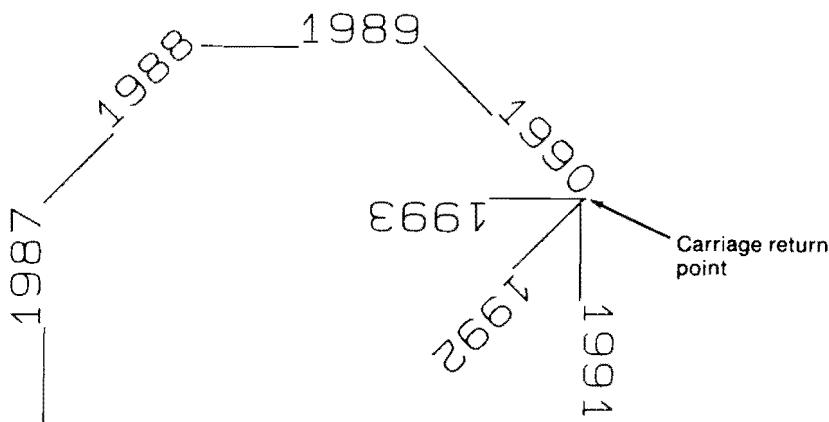
The relative size and sign of the two parameters determine the amount of rotation. If you imagine the current pen location to be the origin of a coordinate system for the label, you can see that the signs of the parameters determine which quadrant the label will be in.

```
10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;PA0,0;"
30  PRINT #1, "DI1,1;LB  DIRECTION"+CHR$(13)+CHR$(3)
40  PRINT #1, "DI1,-1;LB  DIRECTION"+CHR$(13)+CHR$(3)
50  PRINT #1, "DI-1,-1;LB  DIRECTION"+CHR$(13)+CHR$(3)
60  PRINT #1, "DI-1,1;LB  DIRECTION"+CHR$(13)+CHR$(3)
70  PRINT #1, "SP0;""
80  END
```



EXAMPLE: This example illustrates the use of positive and negative parameters, the use of the BASIC COS and SIN functions, how the LB instruction updates the current pen location, and how DI updates the carriage return point.

```
10  'Insert configuration statement here
20 PRINT #1, "IN:SP1;"
30 PRINT #1, "PA0,0;" 
40 PRINT #1, "DI0,1;LB____1987"+CHR$(3)
50 PRINT #1, "DI1,1;LB____1988"+CHR$(3)
60 PRINT #1, "DI1,0;LB____1989"+CHR$(3)
70 'Angle = 135 degrees. Must convert degrees to
80   ' radians. Radians=Degrees*PI/180
90   ' Variables are integer.
100 PI=3.14593
110 A%==COS(315*(PI/180))
120 B%==SIN(315*(PI/180))
130 PRINT #1, "DI";A%;";B%;";LB____1990"+CHR$(3)
140 C%==COS(270*(PI/180))
150 D%==SIN(270*(PI/180))
160 PRINT #1, "DI";C%;";D%;";LB____1991"+CHR$(13)+CHR$(3)
170 E%==COS(225*(PI/180))
180 F%==SIN(225*(PI/180))
190 PRINT #1, "DI";E%;";F%;";LB____1992"+CHR$(13)+CHR$(3)
200 G%==COS(-180*(PI/180))
210 H%==SIN(-180*(PI/180))
220 PRINT #1, "DI";G%;";H%;";LB____1993"+CHR$(13)+CHR$(3)
230 PRINT #1, "SP0;""
240 END
```



RELATED

INSTRUCTIONS: DR, Relative Direction
LB, Label

ERRORS:

Condition	Error	Plotter Response
1 parameter	2	ignores instruction
more than 2 parameters	2	uses first 2 parameters
both parameters = 0 or number out of range	3	ignores instruction

DR, Relative Direction

USE: Specifies the direction in which labels are drawn, relative to the scaling points P1 and P2. *Label direction is adjusted when P1 and P2 change so that labels maintain the same relationship to the plotted data.* Use DI if you want label direction to be independent of P1 and P2.

SYNTAX: DR *run, rise;*
or
DR ;

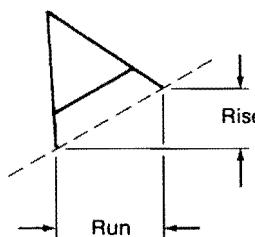
Parameter	Format	Range	Default
run	real	-8 388 608 to 8 388 607	1% of $P2_x - P1_x$
rise	real	-8 388 608 to 8 388 607	0% of $P2_y - P1_y$

REMARKS: When you include parameters, the actual angle at which the label is plotted will vary with the settings of P1 and P2. At least one parameter must be nonzero. The DR instruction is affected by the use of the direction vertical (DV) instruction. Refer to the DI instruction earlier in this chapter for an explanation of this interaction.

- **No Parameters** — Establishes relative direction and sets the label direction to horizontal (parallel to the X-axis). (Same as (DRI, 0,).)
- **Run** — Specifies a percentage of the distance between $P2_x$ and $P1_x$.

- **Rise** — Specifies a percentage of the distance between $P2_y$ and $P1_y$.

You can define the parameters of run and rise as the following.



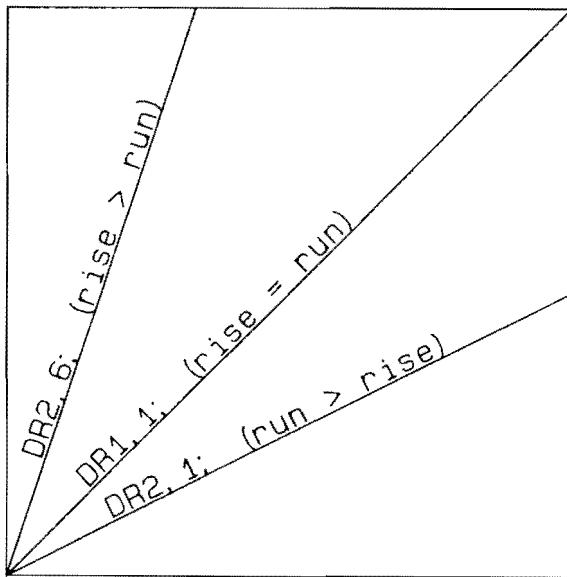
With the DR instruction, the reaction of the plotter to run and rise is somewhat different than with the DI instruction. Refer to the illustration on the next page. Think of the directional line on which the label rests as being parallel to a line beginning at the lower-left scaling point (usually P1) and intersecting the opposite side or the top edge of the graphics limits established by P1 and P2.

To calculate the angle of the label, use the run and rise parameters to form a fraction that is ≤ 1 . It doesn't matter whether run or rise is the numerator. If run = 4 and rise = 6, the fraction is 4/6. If run = 6 and rise = 4, the fraction is still 4/6.

The larger of the two terms determines whether the directional line intersects the top or the side of the graphics limits as follows:

- If run = rise, the fraction will equal 1 and the directional line will extend from corner to corner at a 45-degree angle. (The exact corner is determined by the sign of the parameters.) Refer to the illustration that follows.
- If run > rise, the directional line will intersect the **side** of the graphics limits. The fraction determines the exact intersection point moving from the bottom up to the top scaling point. For example, if P1 is in the lower-left corner, run > rise, and the fraction is 1/2, the directional line will intersect the side 1/2 of the way up toward P2. Refer to the illustration that follows.

- If rise > run, the directional line will intersect the **top** of the graphics limits. The fraction determines the exact intersection point moving from the left toward the right scaling point. For example, if P1 is in the lower-left corner, rise is greater than run, and the fraction is 1/3 ($2/6 = 1/3$), the directional line will intersect the top 1/3 of the way toward P2. Refer to the illustration that follows.



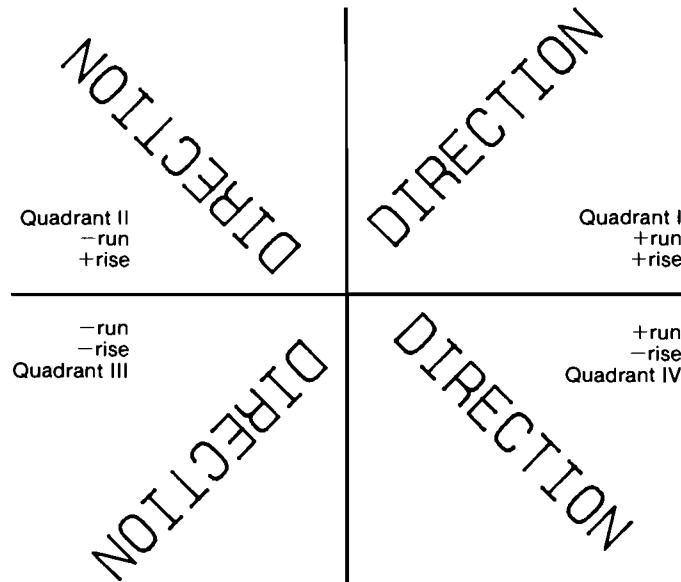
Remember that labels begin at the current pen location and thus will be drawn parallel to the directional line, not necessarily on it.

At least one parameter must be nonzero. The ratio of the parameters to each other is more important than the actual numbers. The table below lists three common label angles produced by using 1's and 0's.

DR Instruction	Label Direction
DR1, 0	horizontal
DR0, 1	vertical
DR1, 1 or D 0.7, 0.7 (any parameters equal to each other)	45-degree angle

A DR instruction remains in effect until another DR or DI instruction is executed, the plotter is initialized or set to default conditions.

The relative size and sign of the two parameters determine the amount of rotation. If you imagine the current pen location to be the origin of a coordinate system for the label, you can see that the signs of the parameters determine which quadrant the label will be in.



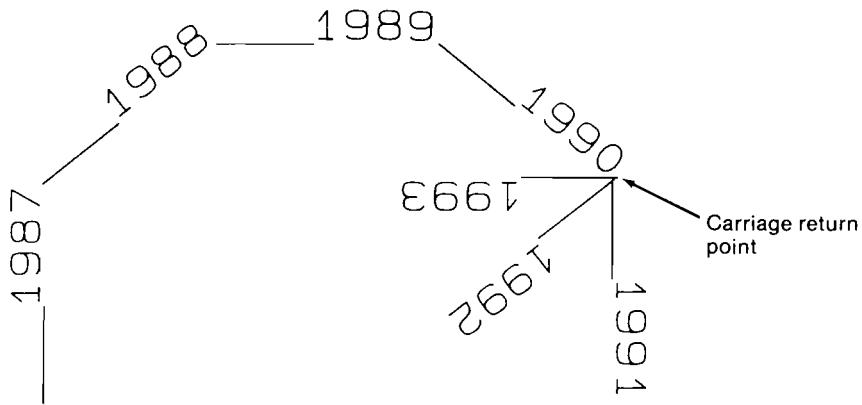
EXAMPLE: This example illustrates the use of positive and negative parameters, how the LB instruction updates the current pen location, and how DR updates the carriage return point.

Notice that this is the same example as shown with the DI instruction. Changing the DI to DR and using the 1:0 ratio instead of COS and SIN were the only changes. However, if you measure the two plots, you'll discover that they are slightly different in size. The size difference results from the DR instruction's use of the percentage of the P2/P1 distance (both examples were plotted on A-size paper).

```

10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;"
30  PRINT #1, "PA0,0;"
40  PRINT #1, "DR0,1;LB____1987"+CHR$(3)
50  PRINT #1, "DR1,1;LB____1988"+CHR$(3)
60  PRINT #1, "DR1,0;LB____1989"+CHR$(3)
70  PRINT #1, "DR1,-1;LB____1990"+CHR$(3)
80  PRINT #1, "DR0,-1;LB____1991"+CHR$(13)+CHR$(3)
90  PRINT #1, "DR-1,-1;LB____1992"+CHR$(13)+CHR$(3)
100 PRINT #1, "DR-1,0;LB____1993"+CHR$(13)+CHR$(3)
110 PRINT #1, "SP0;"
120 END

```



7

RELATED

INSTRUCTIONS: DI, Absolute Direction
LB, Label

ERRORS:

Condition	Error	Plotter Response
1 parameter	2	ignores instruction
more than 2 parameters	2	uses first 2 parameters
both parameters = 0 or number out of range	3	ignores instruction

DT, Define Label Terminator

USE: Specifies the character to be used as the label terminator. Use this instruction to define a new label terminator if your computer cannot use the default label terminator (**ETX**, decimal code 3).

SYNTAX: DT *label terminator*;

or

DT;

Parameter	Format	Range	Default
label terminator	label	any character except NULL , LF , ESC , ENQ , and ; (decimal codes 0, 5, 10, 27, and 59 respectively)	ETX (decimal code 3)

REMARKS: The character immediately following DT is interpreted to be the new label terminator. The label terminator can be a printing character or a nonprinting ASCII control character. Labeling instructions react to the different types of characters as follows:

printing character	terminates the label instruction and prints the character.
nonprinting character (control character)	terminates the label instruction and performs the function specified by the character.

NOTE: A DT instruction with no parameter (DT;) does not establish the semicolon as the terminator; it reestablishes **ETX** as the default terminator. ■

The DT instruction remains in effect until another DT instruction is executed or the plotter is initialized or set to default conditions.

EXAMPLE: The following program shows how to change the label terminator, as well as what happens to plotted labels with different terminators.

NOTE: Although some program lines are shown on two lines to fit on the page, you should write them on just one line. ■

```
10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;SC0.5000,0,5000;" 
30  PRINT #1, "SI.187..269;PA0,4500;" 
40  PRINT #1, "LBDefault control character ETX"+CHR$(13)
     +CHR$(10)+CHR$(3)
50  PRINT #1, "LBterminates by performing end- "+CHR$(13)
     +CHR$(10)+CHR$(3)
60  PRINT #1, "LBof-text function."+CHR$(3)
70  PRINT #1, "CP;CP;DT#;" 
80  PRINT #1, "LBprinting characters terminate,"+CHR$(13)
     +CHR$(10)+"#"
90  PRINT #1, "LBbut are also printed.#"
100 PRINT #1, "CP;CP;DT"+CHR$(13)+";" 
110 PRINT #1, "LBControl characters terminate"+CHR$(10)
     +CHR$(13)
120 PRINT #1, "LBand perform their function."+CHR$(13)
130 PRINT #1, "SP0;" 
140 END
```

Default control character ETX
terminates by performing end-
of-text function.

Printing characters terminate,
#but are also printed.#

Control characters terminate
and perform their function.

7

RELATED

INSTRUCTIONS: BL, Buffer Label

LB, Label

WD, Write To Display

ERRORS:

Condition	Error	Plotter Response
more than 1 parameter	1 or 2	uses first character

DV, Direction Vertical

USE: Specifies vertical mode as the direction for subsequent labels. Use this instruction to 'stack' horizontal characters in a column. This is especially useful when using the Kanji character set (refer to Appendix D).

SYNTAX: DV *n*;

or

DV;

Parameter	Format	Range	Default
<i>n</i>	integer	0 or 1	0 (horizontal)

REMARKS: The plotter interprets the parameters as follows.

- **No Parameter** — Sets the direction for labels to horizontal (unstacked).
- **0** — Sets the label direction to horizontal (unstacked).
- **1** — Sets the label direction to vertical (stacked).

The P1 and P2 settings have no effect on the label direction.

The DV instruction updates the carriage-return point to the current pen location. Line feeds cause the next column of characters to be labeled to the left of the previous column. Also, the plotter rotates the label origins initiated using the LO instruction. The following show some of the LO instruction label origins; the remaining label origins follow the same pattern as these.

7

L	L	L	0	0	0	L	L	L
0	0	0	4	5	6	0	0	0
1	2	3				7	8	9

EXAMPLE: The following illustrates how line feeds and carriage returns affect vertical labels. Horizontal labels are shown for comparison. All terminators, line feeds, and carriage returns are sent using their ASCII decimal code equivalents with the CHR\$ function. These are as follows.

CHR\$(3)	sends the ETX (end-of-text) character
CHR\$(10)	sends the LF (line feed) character
CHR\$(13)	sends the CR (carriage return) character

```
10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;PA1000,3000;DV1;" 
30  'vertical
40  PRINT #1, "LBABC"+CHR$(10)+CHR$(13)+CHR$(3)
50  PRINT #1, "LBDEF"+CHR$(10)+CHR$(3)
60  PRINT #1, "LBGHI"+CHR$(3)
70  'horizontal
80  PRINT #1, "PA3000,3000;DV0;" 
90  PRINT #1, "LBABC"+CHR$(10)+CHR$(13)+CHR$(3)
100 PRINT #1, "LBDEF"+CHR$(10)+CHR$(3)
110 PRINT #1, "LBGHI"+CHR$(3)
120 PRINT #1, "SP0;" 
130 END
```

D	A	ABC
E	B	DEF
F	C	GHI
G		
H		
I		

RELATED

INSTRUCTIONS: DI, Absolute Direction

DR, Relative Direction

LO, Label Origin

ERRORS:

Condition	Error	Plotter Response
more than 1 parameter	2	uses first parameter
number out of range	3	ignores instruction

ES, Extra Space

USE: Adjusts space between characters and lines of labels without affecting character size.

SYNTAX: ES *spaces (, lines)*;
or
ES ;

Parameter	Format	Range	Default
spaces	real	-0.5 to 1* character plot cells	0
lines	real	-0.5 to 2* character plot cells	0

*Practical range; allowable range is -8 388 608 to 8 388 607.

REMARKS:

- **No Parameters** — Sets the spaces and lines between characters to the dimensions of the character plot cell, as set by the most recent SI or SR instruction.
- **Spaces** — Adds (positive number) or subtracts (negative number) spaces between characters. The space is determined by the current character plot cell (average character plot cell for variable-space fonts). You can specify fractional spaces; for maximum legibility, do not specify more than 1 extra space or subtract more than 1/2 space.
- **Lines** — Adds (positive number) or subtracts (negative number) lines between character lines. The line is determined by the current character plot cell (average character plot cell for variable-space fonts). You can specify fractional lines; for maximum legibility, do not specify more than 2 extra lines or subtract more than 1/2 line.

The ES instruction remains in effect until another ES instruction is executed or the plotter is initialized or set to default conditions.

EXAMPLE:

```
10  'Insert configuration statement here
20  PRINT #1, "IN:SP1;PA-4000,2300;SI.187,.269;" 
30  PRINT #1, "ES:LBES; CAUSES"+CHR$(3)
40  PRINT #1, "CP;LBTHIS SPACING."+CHR$(3)
50  PRINT #1, "PA-4000, 1600;
60  PRINT #1, "ES-.1,-.25;LBES-.1,-.25; CAUSES"+CHR$(3)
70  PRINT #1, "CP;LBTHIS SPACING."+CHR$(3)
80  PRINT #1, "PA-4000,900;
90  PRINT #1, "ES.2,.25;LBES.2,.25; CAUSES"+CHR$(3)
100 PRINT #1, "CP;LBTHIS SPACING."+CHR$(3)
110 PRINT #1, "SP0;" 
120 END
```

ES; CAUSES
THIS SPACING.

ES-.1,-.25; CAUSES
THIS SPACING.

ES.2,.25; CAUSES
THIS SPACING.

RELATED

INSTRUCTIONS: CP, Character Plot
LB, Label

ERRORS:

Condition	Error	Plotter Response
more than 2 parameters	2	uses first 2 parameters
number out of range	3	ignores instruction

LB, Label

USE: Plots text using the currently defined character set.

SYNTAX: LB c...c CHR\$(3)

Parameter	Format	Range	Default
c...c	label	any ASCII character	none

REMARKS: Use this instruction to annotate drawings or create text-only charts.

- c...c — Includes up to 150 ASCII characters. Printing characters are drawn using the currently selected character set. (Refer to Chapter 13 for information on selecting alternate character sets.)

You can include nonprinting ASCII characters such as the carriage return (**CR**, decimal code 13) and line feed (**LF**, decimal code 10). These characters invoke the specified function, but are not drawn. Refer to Appendix B for a list of ASCII characters.

The label begins at the current pen location. After each character is drawn, the pen location is updated to be the next character origin (refer to *Understanding the Character Plot Cell* earlier in the chapter.) Therefore, if you need to plot more than 150 characters of text, you can simply put the rest into another label instruction, which will begin labeling at the first character origin after the last character of the previous label.

- **CHR\$(3)** — Terminates the LB instruction. You *must* use a special label terminator instead of the usual terminator (:). The terminator tells the plotter to exit from the label mode. (If you don't use the label terminator, everything following the LB mnemonic will be printed in the label, including other instructions.) The default label terminator is the nonprinting end-of-text character **ETX** (decimal code 3). In BASIC, CHR\$(3) accesses **ETX**. If you wish, you can define a different terminator with the DT instruction.

EXAMPLE:

```
10 'Insert configuration statement here  
20 PRINT #1, "IN;SP1;PA0,0;"  
30 PRINT #1, "LBThis is a label."+CHR$(3)  
40 PRINT #1, "SP0;"  
50 END
```

This is a label.

RELATED

INSTRUCTIONS: CA, Designate Alternate Character Set
CP, Character Plot
CS, Designate Standard Character Set
SA, Select Alternate Character Set
SS, Select Standard Character Set
DT, Define Label Terminator
DI, Absolute Direction
DR, Relative Direction
DV, Direction Vertical
LO, Label Origin
SI, Absolute Character Size
SR, Relative Character Size
SL, Character Slant

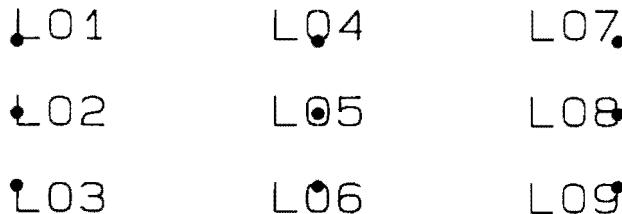
LO, Label Origin

USE: Positions labels relative to current pen location. Use LO to center, left justify, or right justify labels. The label can be drawn above or below the current pen location and can also be offset by an amount equal to 1/2 the character's width and height.

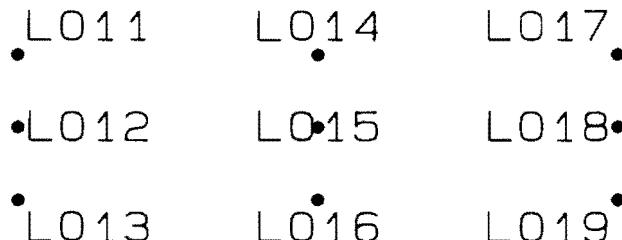
SYNTAX: LO *position number*;
or
LO;

Parameter	Format	Range	Default
position number	integer	1 to 9 or 11 to 19	1

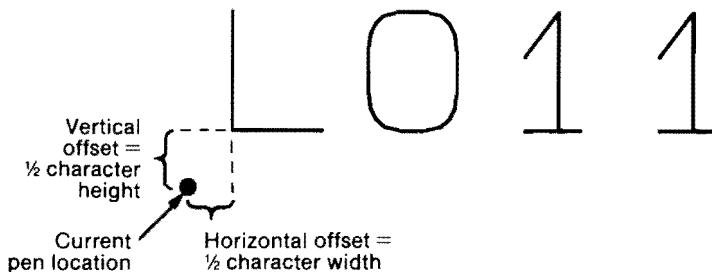
REMARKS: The position numbers are graphically illustrated below. Each dot represents the current pen location.



The label positions LO 11 through LO 19 differ from LO 1 through LO 9 only in that the labels are offset from the current pen location.



The amount of offset is equal to 1/2 the character's width and 1/2 of the character's height as specified by the most recent SI or SR instruction. The offset is shown below.



The current pen location (but not the carriage return point) is updated after each character is drawn and the pen automatically moves to the next character origin. If you want to return a pen to its previous location prior to the next label instruction, you can send a carriage return, CHR\$(13), after the label but before the label terminator.

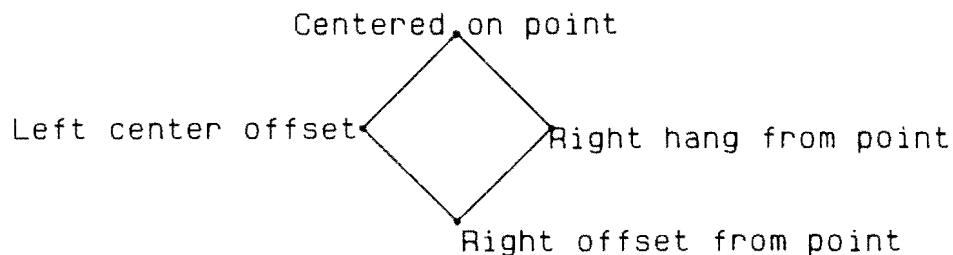
When you embed carriage return characters in a label, each portion of the label is positioned according to the label origin, just as if they were written as separate label instructions.

The LO instruction remains in effect until another LO instruction is executed or the plotter is initialized or set to default conditions.

NOTE: Buffered labels with embedded carriage returns react differently to the LO instruction. The plotter positions the entire label as if it was enclosed in a rectangular box the length of the longest text line. In addition, each line within the box is left-justified. Refer to the BL instruction for an example of this effect. ■

EXAMPLE:

```
10  'Insert configuration statement here
20  PRINT #1, "IN;SI.17,.26;SP1;PA0,500;""
30  PRINT #1, "PD-500,0,0,-500,500,0,0,500;""
40  PRINT #1, "L04;LBCentered on point"+CHR$(3)
50  PRINT #1, "PU-500,0;L018;""
60  PRINT #1, "LBLeft center offset"+CHR$(3)
70  PRINT #1, "PU0,-500;L013;""
80  PRINT #1, "LBRight offset from point"+CHR$(3)
90  PRINT #1, "PA500,0;L03;""
100 PRINT #1, "LBRight hang from point"+CHR$(3)
110 PRINT #1, "SP0;""
120 END
```



RELATED

INSTRUCTIONS: BL, Buffer Label
LB, Label

ERRORS:

Condition	Error	Plotter Response
more than 1 parameter	2	executes first parameter
number out of range	3	ignores instruction

PB, Print Buffered Label

USE: Prints the contents of the label buffer.

SYNTAX: PB;

REMARKS: Enter labels into the label buffer using the BL or LB instructions. Use the PB instruction to plot the contents of the last label at the current pen location.

You can change spacing, direction, position, slant, and size between repeated printings of a label.

EXAMPLE:

```
10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;PA0,2000;SI.17,.29;""
30  PRINT #1, "LBLLabel strings are buffered."+CHR$(3)
40  PRINT #1, "CP;LBLLast string is repeatable."+CHR$(3)
50  PRINT #1, "PA500,1300;PB;""
60  PRINT #1, "SI.25,.35;CP;DR4,-1;SL.36;PB;""
70  PRINT #1, "SP0;""
80  END
```

Label strings are buffered.

Last string is repeatable.

Last string is repeatable.

Last string is repeatable.

RELATED

INSTRUCTIONS: BL, Buffer Label

ES, Extra Space

DI, Absolute Direction

DR, Relative Direction

DV, Direction Vertical

LB, Label

LO, Label Origin

SI, Absolute Character Size

SL, Character Slant

SR, Relative Character Size

ERRORS:

Condition	Error	Plotter Response
1 or more parameters	2	executes instruction

SI, Absolute Character Size

USE: The SI instruction specifies the size of labeling characters in centimetres. Use this instruction to establish character sizing that is not dependent on the settings of P1 and P2.

SYNTAX: SI *width, height*;
or
SI;

Parameter	Format	Range	Default
width	real	-110 to 110*	0.285 cm
height	real	-110 to 110*	0.375 cm

*Excluding zero and values approaching zero. Practical range; allowable range is -8 388 608 to 8 388 607.

REMARKS: You should note that in most languages the width of a letter is typically less than the height. If you set your characters to have a different 'aspect ratio', they may look odd to your readers.

Negative parameters produce mirror images of labels.

- **No Parameters** — Establishes the character size at the default values of 0.285 cm wide and 0.375 cm high.
- **Width** — Specifies the width in centimetres of the capital A on which the character plot cell is based. A negative width parameter mirrors labels in the right-to-left direction.
- **Height** — Specifies the height in centimetres of the capital A on which the character plot cell is based. A negative height parameter mirrors labels in the top-to-bottom direction.

Absolute character size remains in effect until another SI instruction is executed, an SR instruction is executed, or the plotter is initialized or set to default conditions.

EXAMPLE: The following draws characters in two different sizes: first, in the default size, and then 1-cm wide and 1.5-cm high.

```
10  'Insert configuration statement here
20  PRINT #1, "IN:SP1;PA0,0;"
30  PRINT #1, "LBPlot"+CHR$(3)
40  PRINT #1, "PA0,-1000;SI1,1.5;LBPlot"+CHR$(3)
50  PRINT #1, "SP0;"
60  END
```

Plot

P l o t

Negative parameters produce mirror images of labels. A negative width parameter mirrors labels in the right-to-left direction.

"SI-.3,.45;LBPlot"+CHR\$(3)

to [] Current pen location

A negative height parameter mirrors labels in the top-to-bottom direction.

"SI.3,-.45;LBPlot"+CHR\$(3)

Current pen location → b] 0 f

Negative width and height parameters together mirror labels in both directions, causing the label to appear to be rotated 180 degrees.

"SI-.3,-.45;LBPlot"+CHR\$(3)

? 0 [d → Current pen location

RELATED

INSTRUCTIONS: SR, Relative Character Size
DI, Absolute Direction
DR, Relative Direction

ERRORS:

Condition	Error	Plotter Response
1 parameter	2	ignores instruction
more than 2 parameters	2	uses first 2 parameters
number out of range	3	ignores instruction

SL, Character Slant

USE: The SL instruction specifies the slant at which labels are drawn. Use this instruction to create slanted text for emphasis, or to reestablish upright labeling after an SL instruction with parameters has been in effect.

SYNTAX: SL *tangent*;

or

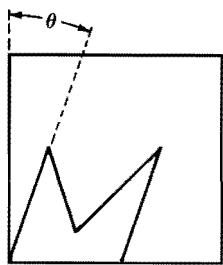
SL;

Parameter	Format	Range	Default
tangent	real	-3.5 to 3.5*	0 (no slant)

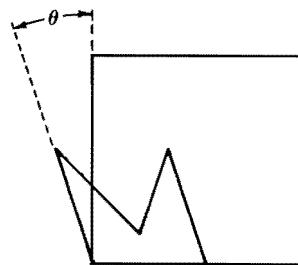
*Practical range; allowable range is -8 388 608 to 8 388 607.

REMARKS:

- **No Parameters** — Establishes the default slant of zero (no slant). This is the same as (SL0;).
- **Tangent** — Interpreted as from an angle ± 90 degrees from vertical. The base of the character always stays on the horizontal as shown in the illustration below.



Positive Slant



Negative Slant

The SL instruction only affects each character relative to an imaginary line under the label. The direction or placement of the label on the plot does not affect the SL instruction.

You can specify the actual tangent value, or you can use the TAN function available on most computers. A table of tangent values for selected angles follows.

θ	Tangent
0	0
-10	-0.18
-20	-0.36
-30	-0.58
-40	-0.84
-45	-1.00
-50	-1.19
-60	-1.73
-70	-2.75
-80	-5.67
-90	infinity

θ	Tangent
0	0
10	0.18
20	0.36
30	0.58
40	0.84
45	1.00
50	1.19
60	1.73
70	2.75
80	5.67
90	infinity

The settings of P1 and P2 do not affect the slant. The DI and DR instructions do affect the slant direction, since the base of a character always stays on the baseline of the label.

An SL instruction remains in effect until another SL instruction is executed, or the plotter is initialized or set to default conditions.

EXAMPLE: The following shows you two methods for specifying the slant parameter. The first label uses a variable generated by the TAN function. The second label uses a tangent value given in the previous table.

```
10 'Insert configuration statement here
20 PRINT #1, "IN;SP1;SI.2,1;PA1000,1000;""
30 PI=3.141593
40 A=TAN(20*(PI/180))
50 PRINT #1, "SL";A;"LBSlant"+CHR$(3)
60 PRINT #1, "PA1000,300;SL-.36;LBSlant"+CHR$(3)
70 PRINT #1, "SP0;""
80 END
```

Slant
Slant

NOTE: If you want to use the TAN function on your computer, check your computer documentation to see how your computer interprets angles. This version of GW-BASIC interprets angles as radians, so line 40 in this program converts radians to degrees. ■

RELATED

INSTRUCTIONS: DI, Absolute Direction
DR, Relative Direction
LB, Label

ERRORS:

Condition	Error	Plotter Response
more than 1 parameter	2	ignores additional parameter
number out of range	3	ignores the instruction

SR, Relative Character Size

USE: The SR instruction specifies the size of characters as a percentage of the distance between P1 and P2. Use this instruction to establish relative character sizes so that if the P1/P2 distance changes, the character sizes adjust to occupy the same relative amount of space.

SYNTAX: SR *width, height*;
or
SR ;

Parameter	Format	Range	Default
width	real	-100 to 100 percent* of $P2_x - P1_x$	0.285 cm
height	real	-100 to 100 percent* of $P2_y - P1_y$	0.375 cm

*Practical range: allowable range is -8 388 608 to 8 388 607.

REMARKS: Use the SR instruction when you want the size of the characters to vary with changes in P1 and P2.

- **No Parameters** — Sets the character size to default values, which are the same size characters (in centimetres) as listed previously under the SI instruction. However, with the SR instruction, the character size adjusts (expands or contracts) with subsequent changes in the locations of P1 and P2.
- **Width** — Sets the character width to the specified percentage of the distance in plotter units between the X coordinates of P1 and P2. A negative width parameter mirrors labels in the right-to-left direction.
- **Height** — Sets the character height to the specified percentage of the distance in plotter units between the Y coordinates of P1 and P2. A negative height parameter mirrors labels in the top-to-bottom direction.

For example, the formula the plotter uses is:

$$\begin{aligned} \text{width} &= \text{width}/100 \times (P2_x - P1_x) \\ \text{height} &= \text{height}/100 \times (P2_y - P1_y) \end{aligned}$$

If the P1 and P2 locations are (-6956, -4388) and (6956, 4388) respectively and you establish a size relative width of 2 and a height of 3.5, the plotter will determine the actual character size as follows.

$$\text{width} = 2/100 \times (6956 - (-6956)) = 278 \text{ plotter units or } 0.695 \text{ cm}$$

$$\text{height} = 3.5/100 \times (4388 - (-4388)) = 307 \text{ plotter units or } 0.768 \text{ cm}$$

If you changed P1 and P2 settings to (100, 100) and (5000, 5000), but didn't change the SR parameters, the character size would change as follows.

$$\text{width} = 2/100 \times (5000 - 100) = 98 \text{ plotter units or } 0.245 \text{ cm}$$

$$\text{height} = 3.5/100 \times (5000 - 100) = 171.5 \text{ plotter units or } 0.429 \text{ cm}$$

An SR instruction remains in effect until another SR (or SI) instruction is executed, or the plotter is initialized or set to default conditions.

NOTE: Either negative SR parameters or switching the relative position of P1 and P2 will produce mirror images of labels. When P1 is in the lower left and P2 is in the upper right, the SR instruction gives the same mirroring results as the SI instruction. However, if you move P1 to the right of P2, characters are mirrored right-to-left; when you move P1 above P2, characters are mirrored top-to-bottom. When *both* of these situations occur—using negative parameters in the SR instruction with an unusual P1/P2 position—double mirroring may result in either direction, in which case *the two inversions cancel*, and lettering appears normal. ■

EXAMPLE: The following shows the default character size after an SR instruction is executed. Next, the locations of P1 and P2 are changed; then, the character size percentages are specified. Notice that the new character size has equal parameters of 2.5; because the P1/P2 area is square, the resulting characters are square.

```
7  
10  'Insert configuration statement here  
20  PRINT #1, "IN;SP1;IP-6956,-4388,6956,4388;"  
30  PRINT #1, "SR;PA0,2700;LBDEFAULT SIZE"+CHR$(3)  
40  '  
50  PRINT #1, "IP0,0,5500,5500;PA0,2000;"  
60  PRINT #1, "LBNEW P1 AND P2 CHANGE LABEL SIZE"  
    +CHR$(3)  
70  '  
80  PRINT #1, "PA0,1000;SR2.5,2.5;"  
90  PRINT #1, "LBNEW SR INSTRUCTION"+CHR$(3)+"CP;"  
100 PRINT #1, "LBCHANGES LABEL SIZE"+CHR$(3)  
110 PRINT #1, "SP0;"  
120 END
```

DEFAULT SIZE

NEW P1 AND P2 CHANGE LABEL SIZE



NEW SR INSTRUCTION
CHANGES LABEL SIZE

RELATED

INSTRUCTIONS: SI, Absolute Character Size
DI, Absolute Direction
DR, Relative Direction

ERRORS:

Condition	Error	Plotter Response
1 parameter	2	ignores instruction
more than 2 parameters	2	ignores additional parameters
number out of range	3	ignores instruction



8

Drawing Polygons and Using the Polygon Buffer

This chapter discusses the plotter's buffers, how to place the plotter in polygon mode, define and draw polygons within the polygon buffer, and change the allocation of polygon buffer space, when necessary.

Instructions Covered

EP, Edge Polygon

FP, Fill Polygon

GM, Graphics Memory

PM, Polygon Mode

Understanding the Plotter's Buffers

The plotter has five configurable buffers: input/output (I/O), polygon, downloadable character, vector, and pen sort. A buffer is a temporary storage area for information. The following paragraphs describe the buffers, their default and configurable sizes, and where to find information on configuring the buffers to meet your needs.

I/O Buffer

The plotter uses the physical I/O buffer for processing instructions. It also uses a logical portion of the buffer for limiting plotter functions such as the number of graphics instructions waiting to be parsed and threshold levels in certain handshaking methods. The logical I/O buffer is not a separate buffer; instead, it is considered an operational subset of the physical I/O buffer.

Because the logical I/O buffer is not a separate buffer, but an operational subset of the physical I/O buffer, the logical I/O buffer cannot be larger than the physical I/O buffer. When you turn on the plotter, the physical and logical I/O buffer sizes are identical. You can change the size of the physical I/O buffer using the ESC . T device-control instruction. If you decrease the size of the physical I/O buffer, the logical I/O buffer automatically decreases to the same size. If you increase the size of the physical I/O buffer, however, the size of the logical I/O buffer does not change. Change the size of the logical I/O buffer using the ESC . @ device-control instruction. Refer to Chapter 15 for complete descriptions of these device-control instructions.

Note that only graphics (HP-GL) instructions are stored and processed in the I/O buffer; device-control instructions bypass this buffer and are executed immediately. If pen sorting is disabled, graphics instructions are executed on a first-in, first-out basis; if pen sorting is enabled, graphics instructions in the I/O buffer are sorted according to pen number, then placed in the pen sort buffer. Refer to the pen sort buffer discussion later in this chapter.

Polygon Buffer

The polygon buffer collects the instructions and coordinates that define a polygon. The polygon remains in the buffer until replaced by another polygon, or the buffer is cleared by initialization or memory allocation. The following instructions use the polygon buffer.

- EA, Edge Rectangle Absolute
- ER, Edge Rectangle Relative
- EW, Edge Wedge
- PM, Polygon Mode
- RA, Fill Rectangle Absolute
- RR, Fill Rectangle Relative
- WG, Wedge Fill

Refer to *Using the Polygon Buffer* later in this chapter for more information on using the polygon buffer.

Downloadable Character Buffer

If your program defines downloadable characters with the DL instruction, you must allocate enough bytes to store all of the downloadable characters that are defined. A formula for determining the number of bytes you need is provided in the section *The Downloadable Buffer* in Chapter 13.

Vector Buffer

The endpoints of the vectors to be plotted are calculated in the vector buffer. Think of a vector as a line segment. Everything that is plotted, even circles and characters, is actually composed of a series of vectors, whose endpoints the plotter calculates internally. The capacity of the buffer depends on whether the curved line generator is on or off.

If the curved line generator is off, the pen stops at the endpoints of each vector as they become available from the vector buffer. Because of this you can sometimes see a difference in line density at the endpoints, particularly with curved lines such as arcs and circles.

If the curved line generator is on (default), the vectors are gathered in the vector buffer before being released for plotting as a group. The pen does not stop at each endpoint in this group of vectors, so the curve appears smoother.

Pen Sort Buffer

When pen sorting is enabled, graphics instructions in the I/O buffer are sorted according to pen number and placed in the pen sort buffer. The pen sort buffer begins drawing vectors (lines) as soon as they are received; the buffer does not need to be full before the plotter begins drawing lines. Sorting vectors by pen number can shorten plotting time by reducing the number of times the plotter switches pens.

You should not use output instructions when using the pen sort buffer. When pen sorting is enabled, the order in which vectors are drawn is not necessarily the order in which the plotter received them. For this reason, you should not use output instructions that output the pen's location. The pen may not be where you expected when the plotter executes the output instruction. You can get unexpected results if your program bases subsequent vectors on the output response.

You can turn on or off pen sorting using either the automatic pen operations (AP) instruction or the front-panel **Sort** function.

Notes for Buffer Allocation

The total configurable plotter memory is 25 600 bytes. The following table shows the default, minimum, and maximum sizes for the plotter's buffers.

Parameter	Format	Range	Default
physical I/O buffer	integer	2 to 25 518	1024
polygon buffer	integer	4 to 25 520 bytes	3072
downloadable character buffer	integer	0 to 25 516 bytes	0
reserved buffer	integer	0	0
vector buffer	integer	66 to 25 582	3000
pen sort buffer	integer	12 to 25 528	18 504

You can change these buffer sizes using the graphics memory (GM) instruction (described later in this chapter) or the ESC . T instruction (described in Chapter 15). The GM instruction is easier to use and is recommended unless you need to change the physical I/O buffer, which it cannot do. Use the ESC . T instruction if you need to change the physical I/O buffer.

If your program does not use a buffer, you may want to maximize the effectiveness of the plotter by allocating that buffer's memory to one or both of the remaining buffers.

For example, if pen sorting is enabled and your program does not include any instructions that make use of the polygon buffer, you may want to set the polygon buffer to 0 and allocate that buffer space to either the I/O or pen sort buffers, or divide it between the two. If pen sorting is disabled and your program does not include any polygon instructions, you can reduce those two buffers to their minimum allocations and increase the size of the physical I/O buffer. (Remember to increase the logical I/O buffer as well.) The larger the I/O buffer, the more data the computer can send to the plotter at a given time.

If You Allocated Too Much Buffer Space

If you allocated more than 25 600 bytes of buffer space, the plotter uses the algorithm described next to reduce the sum to 25 600.

First the plotter determines the number of excess requested bytes, the maximum requested of any buffer, the number of requests at the maximum size, and the second largest amount requested of any buffer. If there is only one maximum request, it is

reduced by the difference between the maximum and second largest amount. If there are 2 or more maximum requests, they are reduced by the number of excess requested bytes divided by the number of maximum requests. This sequence is repeated until no excess remains.

For example, **ESC . T3000 ; 4000 ; 1000 ; 0 ; 20 000 ; 30 000**:

3 000		
4 000		
1 000		
0		
20 000	second largest request	
<u>30 000</u>	maximum request	
58 000	total requested	
<u>-25 600</u>	total allowed	
32 400	excess requested	

The plotter reduces the maximum request by the difference; the result is: **ESC . T3000 ; 4000 ; 1000 ; 0 ; 20 000 ; 20 000**:

3 000		
4 000	second maximum request	
1 000		
0		
20 000	maximum request	
<u>20 000</u>	maximum request	
48 000	total requested	
<u>-25 600</u>	total allowed	
22 400	excess requested	

Reduce maximum requests by $22\ 400 / 2 = 11\ 200$. The result is:
ESC . T3000 ; 4000 ; 1000 ; 0 ; 13 800 ; 13 800:

3 000		
4 000		
1 000		
0		
8 800		
<u>8 800</u>		
25 600	equals maximum buffer size	

No excess request, these are the final buffer allocations.

Understanding and Defining Polygons

A polygon is a closed shape made up of straight lines and/or arcs that the plotter draws by filling with a shaded pattern and/or outlining. With the exception of circles, rectangles, and wedges (all three are examples of polygons), you must place the plotter in polygon mode to define shapes before drawing them. Polygon mode lets you actively use polygon buffer space to define a shape before drawing it. You can also use a series of polygons, called subpolygons, when the shape you need requires it. For example, the block letter **C** is one complete polygon. However the block letter **D** consists of two subpolygons; the outline and the ‘hole’. You can use polygon mode to create numerous shapes for such uses as logos.

When defining polygons, you can use the following graphics instructions.

Appropriate Polygon Mode Instructions

PM	Polygon mode instruction
PA/PR	Plotting instructions
PU/PD	Pen instructions
AA/AR	Arc instructions
CI	Circle instruction
CT	Chord Tolerance

These instructions are stored in the polygon buffer until they are replaced with another polygon, the plotter is initialized, or plotter memory is reallocated. You can also use all of the output instructions while in polygon mode.* The plotter does not store the output instructions in the polygon buffer, they are executed immediately. If you initialize the plotter (issue the IN instruction) while in polygon mode, the plotter exits polygon mode and immediately begins executing subsequent instructions. You must exit polygon mode to execute other graphics instructions.

*The output instructions are OA, OC, OD, OE, OF, OG, OH, OI, OK, OL, OO, OP, OS, OT, and OW. (Refer to Chapters 9, 10, 11, and 14.)

Drawing Polygons and Subpolygons

Knowing and understanding the proper sequence for defining polygons and subpolygons lets you use the polygon buffer space effectively. Using the proper sequence can help prevent dividing one polygon into multiple polygons with stray lines, or overflowing the polygon buffer with too many points or vertices. The following steps define and draw polygons.

1. Move your pen to the point that will be the starting location for your polygon. Then use the polygon mode (PM) instruction to enter polygon definition mode.
2. Define the shape of your polygon using any combination of the previously listed instructions. (The vectors that define your polygon are stored in the polygon buffer until you exit polygon mode.)
3. Close the current polygon and either begin a subpolygon or exit polygon mode.
4. Fill the polygon using the fill polygon (FP) instruction, and/or outline it using the edge polygon (EP) instruction.

Refer to the polygon mode (PM) instruction later in this chapter for a full description of the parameters that open and close polygons and subpolygons.

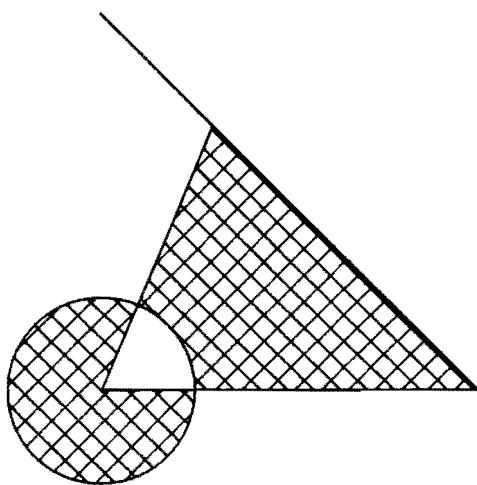
You can define points with the pen up or down. However, the EP instruction only draws between points that are defined when the pen is down. The FP instruction fills all vertices, regardless of whether the pen is up or down when defined.

Drawing Circles in Polygon Mode

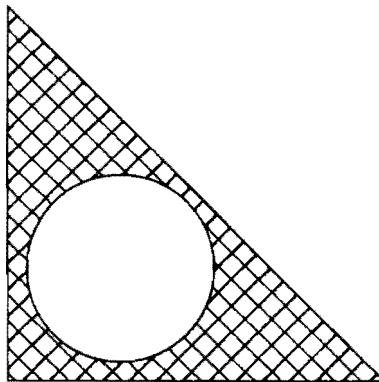
Polygon mode interprets the circle (CI) instruction differently from the other graphics instructions. The plotter treats a circle as a complete subpolygon; that is, the plotter automatically closes the first polygon (if any) before starting the circle, and the first coordinates (if any) after the circle is drawn to start a new subpolygon.

If you have not completely closed your first polygon when you issue the CI instruction, the plotter automatically closes it for you by adding a point. This can change your current pen location and the placement of the circle in your polygon resulting in an inaccurate polygon.

In the following illustration, only two sides of the triangle were drawn before the circle instruction was issued as a subpolygon. This forced the closure of the triangle changing the location of the circle.



The following illustration shows the polygon as it should appear when the first polygon is closed properly. (The edge polygon (EP) instruction example gives the correct program to draw this polygon.)



NOTE: In polygon mode, the smaller a circle's chord tolerance, the more chords will be stored in the polygon buffer to draw it. Circles with small chord tolerances, then, can easily overflow the polygon buffer and the circle will be incomplete. ■

Using the Polygon Buffer

The polygon buffer is a temporary storage area for the instructions that create individual polygons. Polygons refer to those shapes created in polygon mode and by the rectangle and wedge instructions (see Chapters 5 and 6). Once a polygon is stored, it remains in the polygon buffer until it is replaced by a subsequent polygon, or until the buffer is cleared during initialization or memory allocation.

The polygon buffer must be large enough to store the largest polygon in your plot. If the buffer is too small, it overflows and the polygon is incompletely drawn.

When the polygon buffer overflows, the plotter generates an error. You can verify the size of all buffers by using the ESC . S instruction (see Chapter 15). Enlarge the polygon buffer using either the GM (Graphics Memory) instruction described later in this chapter or the ESC . T instruction described in Chapter 15.

Before you change the polygon buffer allocation, it is most important that you thoroughly understand the PM instruction and its parameters, along with the instructions (and parameters) with which you define polygons. The following discussion assumes that you are familiar with these instructions. The PM instruction is discussed later in this chapter.

Determining the Approximate Polygon Buffer Size

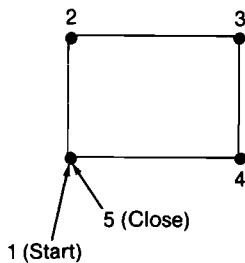
To determine whether or not the polygon buffer has enough room for your polygon, you need to convert the number of points in your polygon into bytes. The following formula is approximate, but it will suit most situations. For a more accurate formula, see *Determining the Exact Buffer Size* later in this chapter.

$$\# \text{ of bytes in buffer} = \# \text{ of points in polygon} \times 14$$

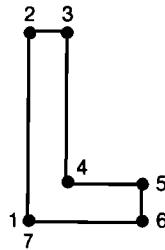
The default size of the polygon buffer is 3072 bytes. This size lets the polygon buffer store any polygon up to 219 points ($3072/14$); this is sufficient for most polygons.

Counting the Points in Your Polygon

The starting pen location and each subsequent point define a polygon. As shown in the following illustration, a rectangle is defined by 5 points, not 4. This is because the starting location is counted again as the ending location.



The following shape has 7 points.



Determining the Number of Points in an Arc or Circle

When an arc or circle defines a polygon, the number of points depends on the number of chords in the arc. When you are using chord angles to determine chord tolerance, use the following formula to determine the number of points used to draw an arc or circle.

Chord Angle Formula

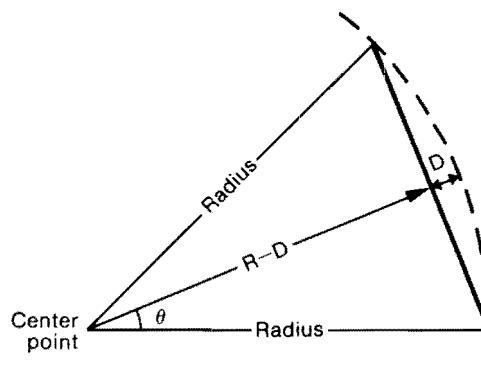
$$\# \text{ of points} = \frac{\text{arc size (degrees)}}{\text{chord angle (degrees)}} + 1$$

Using this formula, a full circle with the default chord angle of 5 degrees consists of 73 points ($360/5 + 1 = 73$) and a 45-degree arc with a chord angle of 3 consists of 16 points ($45/3 + 1 = 16$).

NOTE: If the chord angle does not divide evenly into the arc, round up to the next integer before adding one: $45/2 + 1 = 23 + 1 = 24$. ■

When using deviation distance rather than degrees, the calculation changes. Use the following procedures to convert your deviation distance information to a chord angle, then use the formula above.

When you draw a circle or arc using a specific deviation distance, two things are known, the radius (R) and the deviation distance (D). When you bisect a chord, you divide the radius into the deviation distance and the radius minus the deviation distance ($R - D$). Refer to the following illustration.



The angle θ is determined by the following equation:

$$\theta = \text{ARCCOS } ((R-D)/R)$$

Because you want the angle that created the whole chord, not the bisected chord, the chord angle, then, is twice θ . When you have the chord angle, use the chord angle formula to determine the number of points in the circle.

For example, consider a circle when the radius is 1000 and the deviation distance is 20, the angle θ equals $\text{ARCCOS } ((1000-20)/1000)$ or 11.5 degrees. The chord angle that created the whole chord is 23 degrees (11.5×2). Using the chord angle formula above the number of points is then $(360/23) + 1 = 16 + 1 = 17$.

Determining the Exact Buffer Size

In most situations, the methods previously described will provide the best approximation. The following formula is more precise if you want to allocate unused bytes to the I/O and/or pen sort buffers.

$$\text{number of bytes} = 2 + [(p_1 + f_1) + (p_2 + f_2) + \dots]$$

Each segment of the equation is described as follows:

2 is the number of bytes required for overhead.

p is 1 byte required by a PU, PD, PM1, or PM2 instruction.

f is defined by the following formula.

$$f = (12 \times \text{number of points}) + (\left\lceil \frac{\text{total number of points}}{128} \right\rceil \times 2)$$

(The ceiling symbol $\lceil \rceil$ indicates that the result of the division is to be rounded up to the next integer. Thus, for 1 point, $f = (12 \times 1) + (\lceil 1/128 \rceil \times 2) = 12 + 2 = 14$ because $1/128 = 1$, not 0.078.)

To better understand the equation, let's try an example.

```
“PA0,0;”
“PM0;PD0,10,10,16;”
“PD20,20,30,14,40,18,50,16;”
“PD60,22,60,0,0,0;”
“PM1;PU4,4;PD4,8,16,8,16,4,4,4;”
“PU;PM2;”
```

This polygon has 15 points. The following shows you how the number of bytes is determined for this polygon.

Overhead	2
Current pen location before PM0 (0,0) counts as first point; solve f using 1 point	14
PD requires 1 byte	1
2 points follow PD; solve f using 2 points	26
PD requires 1 byte	1
4 points follow PD; solve f using 4 points	50
PD requires 1 byte	1
3 points follow; solve f using 3 points	38
PM1 uses 1 byte *	1
PU requires 1 byte	1
1 point follows; solve f using 1 point	14
PD requires 1 byte	1
4 points follow PD; solve f using 4 points	50
PU requires 1 byte	1
<u>PM2*</u> requires 1 byte	1
Total number of bytes required for this polygon is	202

Contrast this number with the previous formula ($14 \times$ number of points) for an approximate buffer size. The polygon has 15 points, so the approximate size of the polygon buffer is 210 bytes (15×14). You save 8 bytes for one of the other buffers by using the more precise equation. Of course, if you sent longer strings, thus eliminating the extra PD instructions, you would save a few more bytes.

*In this example, the last point before PM1 closed the subpolygon, and was included in the previous point count when solving for f . However, if you do not include a point that closes the subpolygon, the plotter will automatically add this point, so you would need to add 14 (solve f for one point) after the PM1. This also holds true for PM2 if you do not include a point that closes the polygon.

Changing the Size of the Polygon Buffer

You can use one of two instructions to change memory allocation. These are the graphics memory (GM) instruction and the ESC.T instruction, which change configurable memory allocations. The GM instruction described later in this chapter, allocates memory to all except the physical I/O buffer. The ESC.T instruction is a device-control instruction that allocates memory to all buffers including the physical I/O buffer. (ESC.T is described in Chapter 15).

The following guidelines will help you decide which instruction to use.

- Use the GM instruction when you don't need to change the size of the I/O buffer. The GM instruction is easiest to use. Unlike the ESC.T instruction, GM enters the I/O buffer and is executed in sequence with other graphics instructions. It does not purge the contents of the I/O buffer before allocating memory.
- Use the ESC.T instruction when you also need to change the size of the physical I/O buffer. ESC.T first clears all buffers, then allocates memory. Since it is a device-control instruction, it does not enter the I/O buffer but is executed immediately. When you use ESC.T, precede it with an ESC.O to ensure that the I/O and pen sort buffers are empty before memory is allocated.

Refer to *Understanding the Plotter's Buffers* earlier in this chapter for an overview of each of the plotter's buffers.

EP, Edge Polygon

USE: Outlines the polygon currently stored in the polygon buffer. Use this instruction to edge polygons that you defined in polygon mode and with the rectangle and wedge instructions (RA, RR, and WG).

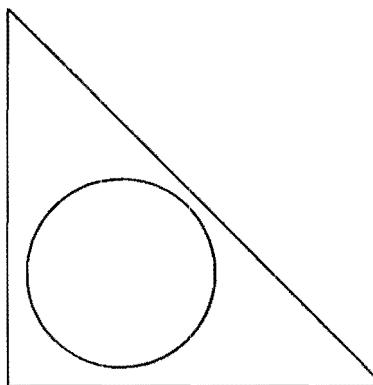
SYNTAX: EP;

REMARKS: The EP instruction outlines any polygon that is currently in the polygon buffer. This includes wedges and rectangles defined using the WG, RA, and RR instructions. EP accesses the data in the polygon buffer, but does *not* clear the buffer or change the data in any way.

EP only edges between points that were defined with the pen down. These are edged using the current pen and line type; the edge rectangle instructions (EA and ER) and edge wedge instruction (EW) outline using a solid line only. When finished the pen returns to its original location and position (up/down).

EXAMPLE: The following creates a shape in polygon mode, then uses EP to outline it.

```
10 'Insert configuration statement here
20 PRINT #1, "IN;SP1;PA4000,0;"'
30 PRINT #1, "PM0;P02000,0,0,2000,0,0:PM1;"'
40 PRINT #1, "PU600,600;CI500;PM2;"'
50 PRINT #1, "EP;"'
60 PRINT #1, "SP0;"'
70 END
```



RELATED

INSTRUCTIONS: EA, Edge Absolute Rectangle
ER, Edge Relative Rectangle
EW, Edge Wedge
PM, Polygon Mode

ERRORS:

Condition	Error	Plotter Response
use of a parameter	2	executes instruction

FP, Fill Polygon

USE: Fills the polygon currently in the polygon buffer. Use FP to fill polygons defined in polygon mode and by the edge rectangle and wedge instructions (EA, ER, and EW).

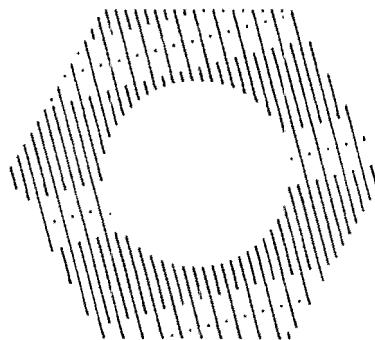
SYNTAX: FP;

REMARKS: The FP instruction fills a polygon that has been previously placed in the polygon buffer, but does not clear the buffer or in any way change the data.

The polygon is filled using the current pen, fill type, and line type (if the fill type is not solid). On completion of the FP instruction, the plotter restores the original location and pen up/down status.

EXAMPLE: The following creates a polygon composed of two subpolygons. In this case, the FP instruction fills alternating areas, beginning with the outside area. After the polygon is filled, you may want to outline it (using the edge polygon (EP) instruction) to give the polygon a smooth, finished appearance.

```
10  *Insert configuration statement here
20  PRINT #1, "IN;SP1;PA2000,2000;""
30  PRINT #1, "PM0;CI1000,60;CI500;PM2;""
40  PRINT #1, "LT4;FT3,S0,45;""
50  PRINT #1, "FP;""
60  PRINT #1, "SP0;""
70  END
```



RELATED

INSTRUCTIONS: EA, Edge Absolute Rectangle
EP, Edge Polygon
ER, Edge Relative Rectangle
EW, Edge Wedge
FT, Fill Type
LT, Line Type
PM, Polygon Mode

ERRORS:

Condition	Error	Plotter Response
one or more parameters	2	executes instruction
previous PM, RA, RR, or WG instruction overflowed the polygon buffer	7*	ignores instruction

*Error actually occurs when buffer overflows. That is, it is possible to define and edge a polygon, but not be able to fill it.

GM, Graphics Memory

USE: Allocates memory to four of the five buffers in the configurable graphics memory.

SYNTAX: GM (*polygon buffer*)*(, downloadable character buffer)*
 *(, reserved buffer)**(, vector buffer)**(, pen sort buffer)*;
 or
 GM ;

Parameter	Format	Range	Default
polygon buffer	integer	4 to 25 520 bytes	3072 bytes
downloadable character buffer	integer	0 to 25 516 bytes	0
reserved buffer*	integer	0	0
vector buffer	integer	66 to 25 582	3000 bytes
pen sort buffer	integer	12 to 25 528	18 504 bytes

*This buffer is unused.

REMARKS: The plotter has a total of 25 600 bytes of available memory in five buffers: the physical I/O buffer, polygon buffer, downloadable character buffer, vector buffer and pen sort buffer. The GM instruction does not change the size of the physical I/O buffer. Use the ESC.T instruction (Chapter 15) if you need to change the physical I/O buffer.

You must allocate 0 to the reserved buffer in order to enter values for the vector and pen sort buffers. If you specify fewer than five parameters with the GM instruction, the plotter allocates the buffers in order and sets any subsequent, unspecified buffer to its default allocation. For example, (GM512;) sets the polygon buffer to 512 and defaults the vector buffer to 3000 and the pen sort buffer to 18 504.

The GM instruction clears the polygon, downloadable character, vector, and pen sort buffers before allocating memory; it does not affect the I/O buffer.

If you increase any buffer above its default size, you must decrease the size of another buffer by an equivalent amount. Refer to *If You Allocated Too Much Buffer Space* under *Notes for Buffer Allocation* earlier in this chapter for the algorithm the plotter follows if you increase a buffer without decreasing another one.

The GM instruction is executed like other graphics instructions; that is, on a first in/first out basis. ESC . T is a device-control instruction and is executed immediately. The advantage of the GM instruction is that it lets you alter the size of the polygon buffer, for example, for a large and/or intricate polygon at the moment that you need the extra buffer space, rather than at the beginning of the program. As soon as you are through drawing the polygon, you can immediately reset the buffers to more reasonable sizes with no loss of data.

RELATED

INSTRUCTIONS: **ESC . S**, Output Configurable Memory Size
ESC . T, Allocate Configurable Memory

ERRORS:

Condition	Error	Plotter Response
too many parameters	2	ignores extra parameters
sum of parameters (and I/O buffer) exceeds 25 600 bytes	3	performs algorithm and reduces parameters accordingly

PM, Polygon Mode Instruction

USE: Enters polygon mode for defining shapes such as block letters, logos, surface charts, or any unique or intricate area, and exits for subsequent filling and/or edging. Fill polygons using the fill polygon (FP) instruction and/or outline them using the edge polygon (EP) instruction.

SYNTAX: PM *n*;
or
PM;

Parameter	Format	Range	Default
<i>n</i>	integer	0, 1, and 2	0

REMARKS: In polygon mode, you define the area of the polygon(s) using graphics instructions. These instructions (and associated X,Y coordinates) are stored in the polygon buffer. The polygon is not plotted until you exit polygon mode and fill and/or outline the area.

The PM instruction accepts only three parameters:

- 0 — clears the polygon buffer and enters polygon mode. This is the same as no parameter (*PM*);.
- 1 — closes the current polygon (or subpolygon).
- 2 — closes current polygon (or subpolygon) and exits polygon mode.

The following paragraphs explain how to use each parameter. The order in which you use these instructions is very important.

PM0; or PM;

Use (PM0;) to clear the polygon buffer and enter polygon mode. While in polygon mode, you can use certain instructions to define your polygon. The following table lists these instructions.

Appropriate Polygon Mode Instructions

PM	Polygon mode
PA/PR	Plotting modes
PU/PD	Pen positions
AA/AR	Arc instructions
CI	Circle instruction
CT	Chord Tolerance instruction

The polygon buffer stores the lines (vectors) that define your polygon. These vectors are accessed later when you exit polygon mode and fill and/or edge the polygon.

NOTE: While in polygon mode, the CI instruction is interpreted differently than other graphics instructions. Refer to *Drawing Circles in Polygon Mode*, earlier in this chapter for more details. ■

You can also use the IN (initialize) instruction, which exits polygon mode and begins executing subsequent instructions immediately. Output instructions can also be used; they are not stored in the buffer, but are executed immediately. You must exit polygon mode to execute other graphics instructions.

When you define polygons, the pen location before (PM0;) is the first point (vertex) of the polygon, and the first point stored in the polygon buffer. For example, if you execute the instructions (PA0, 1750; PM0;) the absolute coordinates (0, 1750) are the first point of your polygon. Each subsequent pair of coordinates defines a point, or vertex, of the polygon.

You can define points with the pen up or down. However, the EP instruction only draws between points that are defined when the pen is down. The FP instruction fills the area(s) between all vertices, regardless of whether the pen is up or down when defined.

NOTE: The default buffer size is sufficient for a polygon with approximately 219 vertices. Refer to *Using the Polygon Buffer* earlier in this chapter if you think you may need to change the size of your polygon buffer. ■

It is good programming practice to ‘close’ the polygon before exiting polygon mode. Closing a polygon means adding the final vertex that defines a continuous shape; the last coordinates or increments represent the same location as the first. If you have not closed the polygon, executing (*PM1;*) or (*PM2;*) forces closure by adding a point to close the polygon. However, you can end up with an unexpected polygon by not closing it yourself.

PM1;

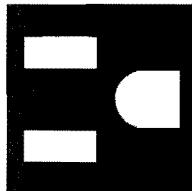
Use (*PM1;*) to close the current polygon (or subpolygon) and remain in polygon mode. When you use (*PM1;*), the point after (*PM1;*) becomes the first point of the next subpolygon. When drawing the polygon, the pen will always move to this point in the up position, regardless of the current pen status. Each subsequent coordinate pair after (*PM1;*) defines a point of the subpolygon.

PM2;

Use (*PM2;*) to close the current polygon (or subpolygon) and exit polygon mode. Remember, if you have not closed your polygon, executing (*PM2;*) adds a point to close the polygon. Unless you exit polygon mode, the plotter will ignore all graphics instructions *except* IN and all graphics output instructions.

EXAMPLE: The following draws the surface area of a 3-prong receptacle as a series of subpolygons, fills and edges it using the FP and EP instructions, respectively.

```
10  'Insert configuration statement here
20  PRINT #1, "IN:SP1;PA2000,2000;" 
30  PRINT #1, "PM0;" 
40  PRINT #1, "PD3000,2000,3000,3000;" 
50  PRINT #1, "FD2000,3000,2000,2000;" 
60  PRINT #1, "PM1;" 
70  PRINT #1, "PD2080,2160,2480,2160,2480,2340,2080,2340,
     2080,2160;" 
80  PRINT #1, "PM1;" 
90  PRINT #1, "PD2080,2660,2480,2660,2480,2840,2080,2840,
     2080,2660;" 
100 PRINT #1, "PM1;" 
110 PRINT #1, "PD2920,2340,2920,2660,2720,2660;" 
120 PRINT #1, "AA2720,2500,180;PD2920,2340;" 
130 PRINT #1, "PM2;FP;EP;SP0;" 
140 END
```



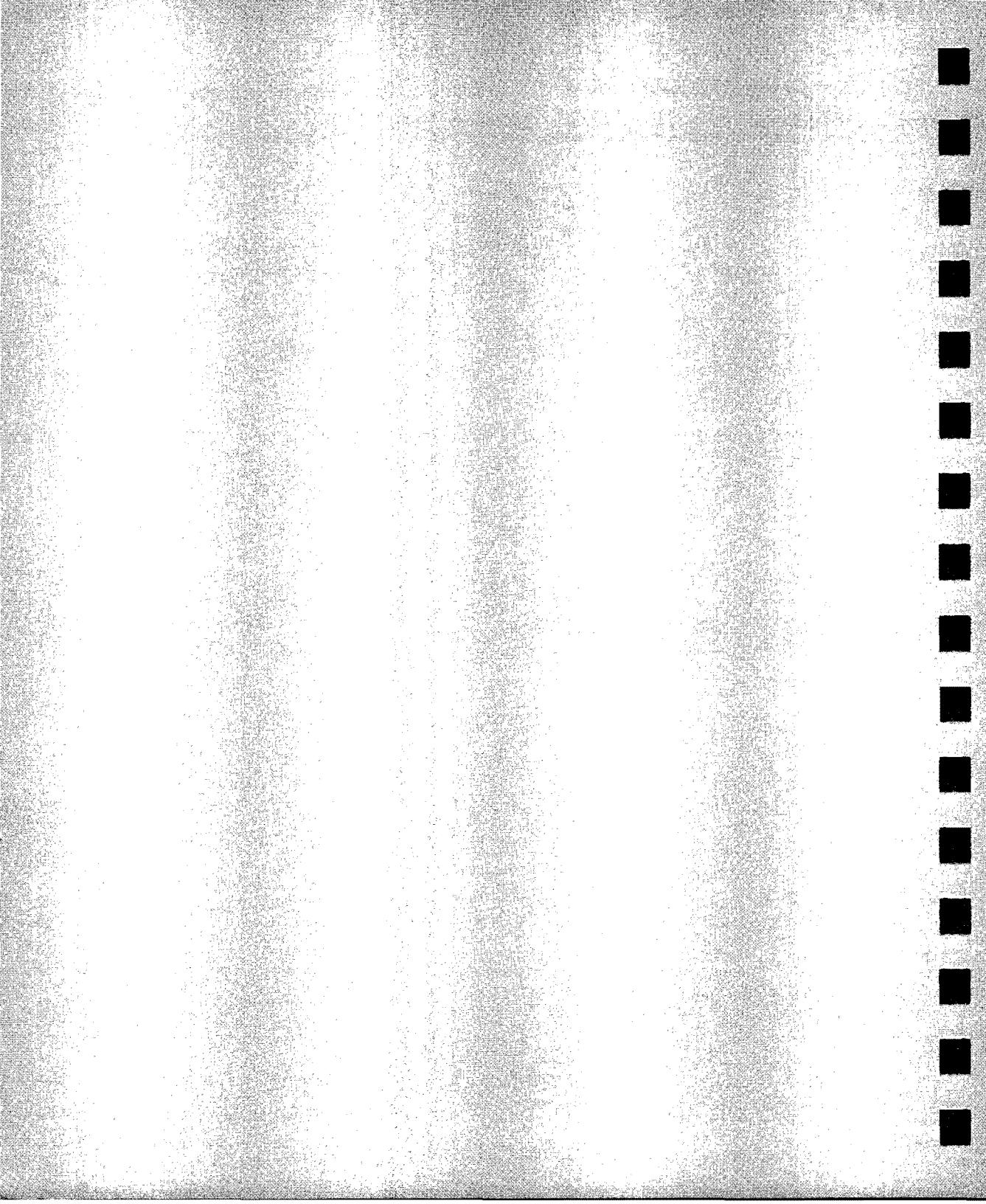
RELATED

INSTRUCTIONS: EP, Edge Polygon
FP, Fill Polygon

ERRORS:

Condition	Error	Plotter Response
invalid instruction used in polygon mode	1	ignores illegal instruction
number out of range	3	ignores instruction
buffer overflow	7	ignores overflowing points

Notes



9

Changing Picture Area and Orientation

For most plotting situations, the plotter will probably draw as you expect it to. However, proper movement depends on your graphics limits and the X,Y coordinates you specify. This chapter discusses how to manipulate the graphics limits to your specific needs; that is, establish windows (soft-clip limits) to restrict plotting to a specific area, and rotate the coordinate system. This chapter also introduces you to some output instructions that you can use effectively with the other instructions presented here.

Instructions Covered

IW, Input Window
OH, Output Hard-Clip Limits
OP, Output P1 and P2
OW, Output Window
RO, Rotate Coordinate System

Terms You Should Understand

Graphics Limits
Hard-clip Limits
Scaling
Soft-clip limits
Window

Windowing: Setting Up Soft-Clip Limits

Soft-clip limits temporarily restrict pen movement to a rectangular area or window. When you turn on, initialize the plotter, or set it to its default conditions, the soft-clip limits are the same as the hard-clip limits. You establish the soft-clip window using the input window (IW) instruction. The window has the same effect as the hard-clip limits; the plotter does not draw vectors or labels outside the window.

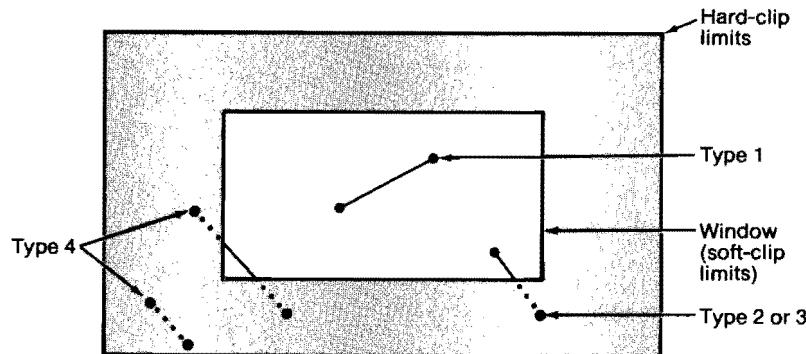
The following illustration shows the four types of line segments that you can specify from one point to another.

Last Point

1. Inside window area
2. Inside window area
3. Outside window area
4. Outside window area

New Point

- | | |
|----|---------------------|
| to | inside window area |
| to | outside window area |
| to | inside window area |
| to | outside window area |



The IW instruction enables you to control the size of the plotting area so that you can draw a particular portion of a plot. You can use the remaining area for labels, or another plot. Refer to *Soft-Clip Limits* in Chapter 2 and the IW instruction description later in this chapter.

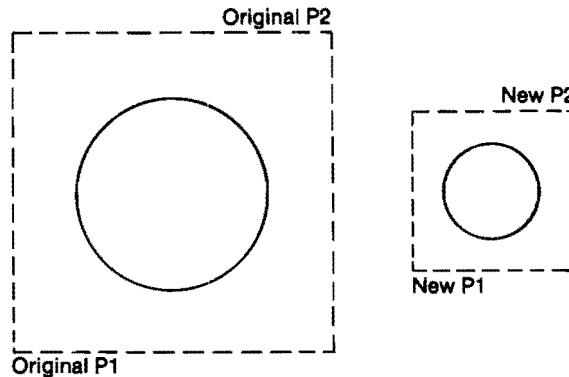
Enlarging or Reducing a Picture

The basic technique for changing a picture's size is to scale the plotting area defined by P1 and P2, then move the locations of P1 and P2 so they define a smaller or larger area. (Only scaled plots are affected by the changes in the P1/P2 locations.) This is especially useful when you want to be able to plot the picture on any portion of the page.

To maintain the proportions of scaled plots, set P1 and P2 so they define an area with the same aspect ratio as the original scaling rectangle. For example, if the area defined by P1 and P2 is 3000 X 2000 plotter units in its X- and Y-axes, respectfully, its aspect ratio is 3:2. To enlarge the plot, set P1 and P2 to define a larger area that maintains a 3:2 ratio.

The following illustrates this using a square (isotropic) P1/P2 scaling rectangle (the ratio is 1:1) with a scale of 0 to 10 in both axes. After drawing a circle within the scaled area, the locations of P1 and P2 move to form a new rectangular area that maintains the 1:1 ratio. Note that the circle plotted in the new area is smaller but is proportionally identical.

```
10  'Insert configuration statement here
20  PRINT #1, "IN;IP2000,2000,6000,6000;""
30  PRINT #1, "SC0,10,0,10;SP1;""
40  PRINT #1, "PA5,5;CI3;""
50  PRINT #1, "IP7000,2000,9000,4000;""
60  PRINT #1, "PA5,5;CI3;SP0;""
70  END
```

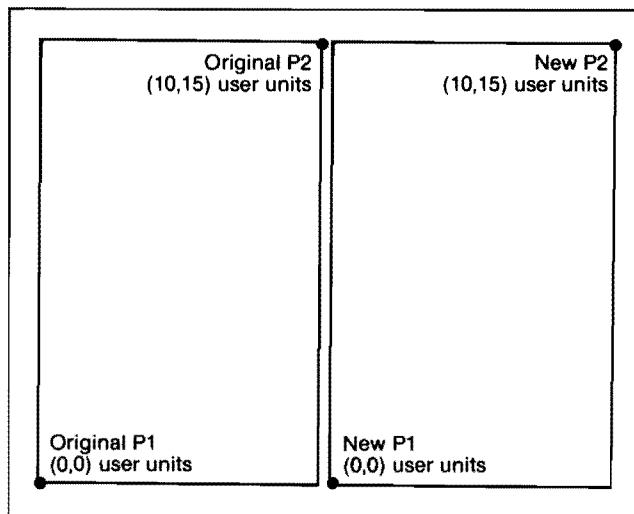


Drawing Equal-Sized Pictures on One Page

You may occasionally want to plot more than one drawing on the same page for a side-by-side comparison. This could be useful for comparing parts, assemblies, layouts, or other similar information. The easiest way to draw equal-sized pictures on one piece of paper is to take advantage of the fact that P2 follows P1 whenever you change the location of P1.

For example, the following locates P1 and P2 on the left side of the paper and scales the area. After drawing a boundary using the scaling parameters, only the P1 location is moved to the right side of the paper; P2 automatically tracks P1 so the plotting area retains the same dimensions as the first. The plotted rectangle around the second area shows P2 in its new location.

```
10 *Insert configuration statement here
20 PRINT #1, "IN:IP-10000,-7000,-100,7000;"
30 PRINT #1, "SC0.10.0.15;"
40 PRINT #1, "SP1;PA0.0;PD10.0,10,15,0,15,0,0;PU;"
50 PRINT #1, "IP100,-7000;"
60 PRINT #1, "PA0.0;PD10.0,10,15,0,15,0,0;"
70 PRINT #1, "SP0;"
80 END
```



NOTE: These P1/P2 frames are not windows or graphics limits; the pen can plot anywhere within the hard-clip limits. The new P1 and P2 retain their scaled values. This allows you to use the same coordinates on both halves of the page. If you do not assign a scale to P1 and P2, you must calculate the new plotter unit coordinates for the plot on the second half of the page. ■

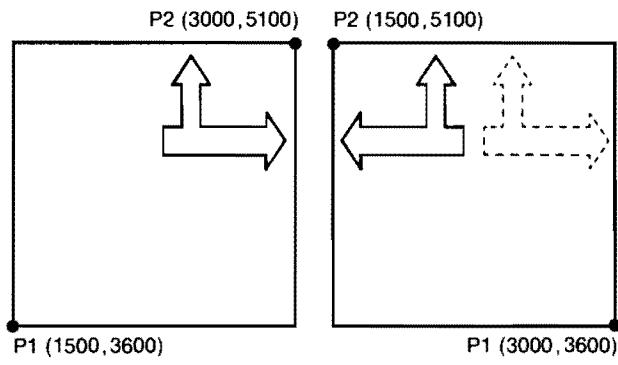
Creating Mirror Images

For most plots, you will probably set P1 and P2 so that P1 is in the lower-left corner and P2 is in the upper-right corner of the scaling area. However, you can change the relationship of P1 and P2. When you do, mirror imaging can occur (scaling must be on).

You can mirror image any scaled plot by changing the relative locations of P1 and P2. You can mirror image labels using the direction absolute and direction relative (DI and DR) instructions or the size relative (SR) instruction. The DI, DR, and SR instructions are discussed in Chapter 7, *Labeling Your Plots*.

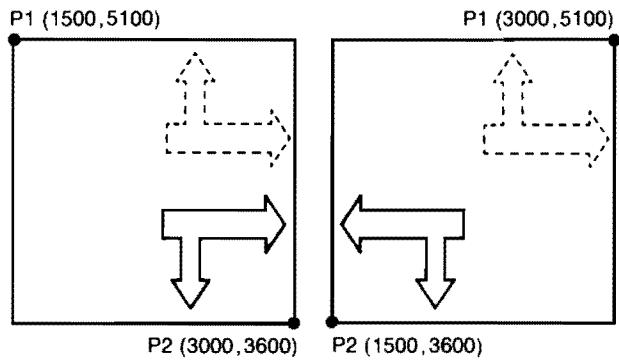
The following program uses a subroutine to draw the exact same picture (an arrow) four times. Refer to the illustration on the following page. Because the program changes the relative locations of P1 and P2, the direction of the arrow is different in each of the four drawings. The program sets P1 and P2, draws the plot, then returns to reset P1 and P2 (using the IP instruction). This continues until all four possible mirror images are plotted. (The original plot is shown in each picture so you can compare the orientation of the mirror image.)

```
10  "Insert configuration statement here
20  PRINT #1, "IN;SP1;IP1500,3600,3000.5100;""
30  PRINT #1, "SC-15,15,-10,10;""
40  GOSUB 130
50  PRINT #1, "IP3000,3600,1500,5100;""
60  GOSUB 130
70  PRINT #1, "IP1500,5100,3000,3600;""
80  GOSUB 130
90  PRINT #1, "IP3000,5100,1500,3600;""
100 GOSUB 130
110 PRINT #1, "SP0;""
120 END
130 "draws the plot
140 PRINT #1, "PA1,2;P01,4,3,4,3,2,2,2,4,9,6,7,5,7;""
150 PRINT #1, "P05,4,12,4,12,5,14,3,12,1;PU;""
160 PRINT #1, "P012,2,1,2;PU;""
170 RETURN
```



*Normal
(Lines 20-40)*

*Reversed
(Lines 50-60)*



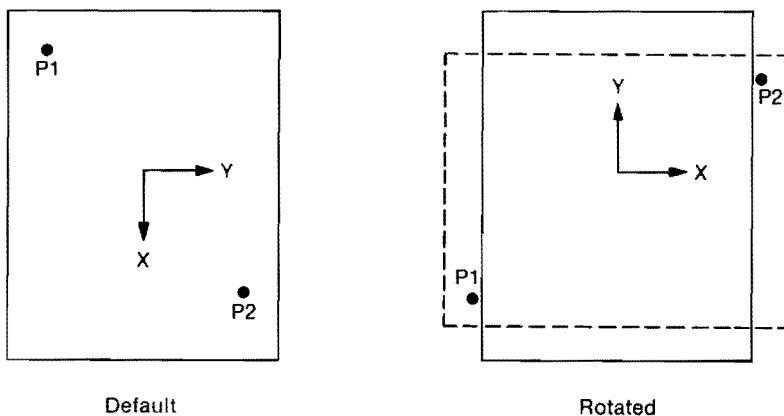
*Upside Down
(Lines 70-80)*

*Upside Down and Reversed
(Lines 90-100)*

Rotating a Picture

The plotter always sets the X-axis parallel to the longest edge of your media with P1 in the lower-left corner. However, you can change this orientation using the rotate (RO) instruction to rotate the coordinate system counterclockwise 90 degrees. Note that you can rotate the orientation only once, and then rotate it back again. Rotations are not cumulative.

The following shows the default and rotated orientation of the axes and locations of P1 and P2.



Note that the locations of P1 and P2 are now off the page. This occurs because the X,Y coordinates of P1 and P2 do not change. (The front-panel **Rotate** rotates the coordinate system and places P1 and P2 within the hard-clip limits; refer to the User's Guide.) You can use the IP instruction after the RO instruction to set the locations of P1 and P2 within the hard-clip limits. When you reset your coordinate system to its default orientation, remember to reset P1 and P2 (using the IP instruction again).

Using the Output Instructions in This Chapter

When changing a plot's size or orientation, you often need to know the current soft-clip limits (window), hard-clip limits, and the locations of P1 and P2. You can get this information from the plotter by using the output window (OW), output hard-clip limits (OH), and output P1 and P2 (OP) instructions described later in this chapter.

Output instructions are a special type of graphics instructions. They perform no plotting. They send information from the plotter to the computer. Read *Notes for Obtaining Plotter Output* in Chapter 14 before using the output instructions in this chapter.

IW, Input Window

USE: Defines a rectangular area, or window, that establishes soft-clip limits. Subsequent programmed pen motion will be restricted to this area. Use this instruction when you want to be sure that your plot falls within a specific plotting area.

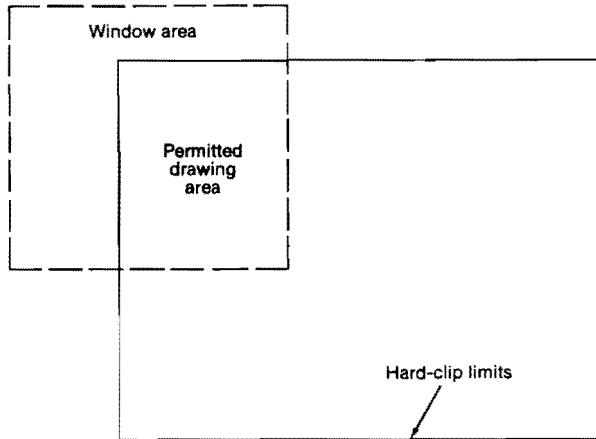
SYNTAX: IW $X1, Y1, X2, Y2;$
or
IW;

Parameter	Format	Range	Default
X,Y coordinates	integer current units*	-8 388 608 to 8 388 607	current hard-clip limits

*Regardless of scaling, the IW parameters must be integer. They will be interpreted as current units.

REMARKS: The four parameters of the IW instruction specify the X,Y coordinates of opposite, diagonal corners of the window area, usually the lower-left and upper-right corners.

You can define a window that extends beyond the hard-clip limits, however, the plotter cannot move the pen beyond the hard-clip limits.



If the window falls entirely outside of the hard-clip limits, no plot will be drawn. This can happen when you define a window that is normally within the hard-clip limits, then a subsequent rotate (RO) instruction moves the window outside of the hard-clip limits.

When you turn the plotter on, the window is automatically set to the hard-clip limits. You can define a window anywhere within the hard-clip limits. All programmed pen

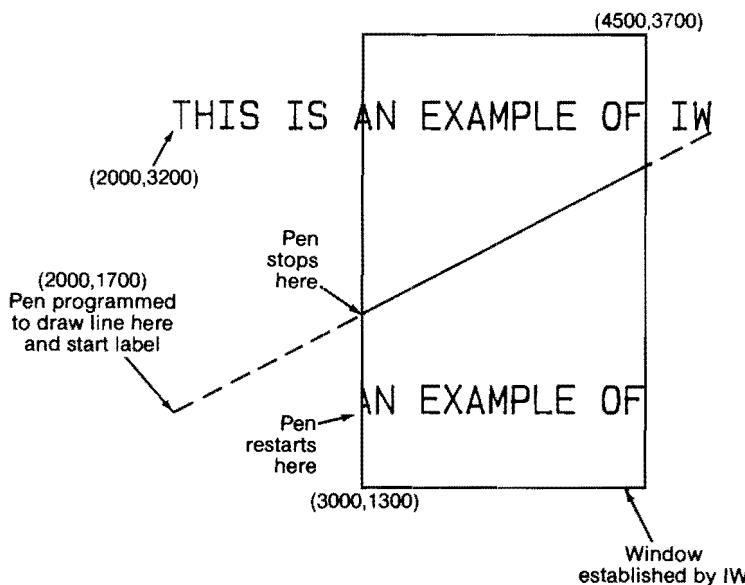
motion is then restricted to this area. For more information, refer to *Windowing: Setting Up Soft-Clip Limits* at the beginning of this chapter.

NOTE: Pen movement directed by the front-panel **Cursor Control** buttons is not restricted by a window. ■

The IW instruction remains in effect until another IW instruction is executed, or the plotter is initialized or set to default conditions.

EXAMPLE: The following draws a label, then establishes a window and draws the label again along with a line. Notice how the line and label are clipped after the window has been established, but not before.

```
10  *Insert configuration statement
20  PRINT #1, "IN;SP1;"
30  PRINT #1, "SI.Z..35;PA2000,3200;"
40  PRINT #1, "LBTHIS IS AN EXAMPLE OF IW"+CHR$(3)
50  PRINT #1, "IW3000,1300,4500,3700;"
60  PRINT #1, "PD2000,1700;"
70  PRINT #1, "LBTHIS IS AN EXAMPLE OF IW"+CHR$(3)
80  PRINT #1, "PU3000,1300;PD4500,1300,4500,3700;"
90  PRINT #1, "PD3000,3700,3000,1300;PU;"
100 PRINT #1, "SP0;"
110 END
```



RELATED**INSTRUCTIONS:** OW, Output Window**ERRORS:**

Condition	Error	Plotter Response
more than 4 parameters	2	uses first 4 parameters
1, 2, or 3 parameters	2	ignores instruction
number out of range	3	ignores instruction

OH, Output Hard-Clip Limits

USE: Outputs the X,Y coordinates of the current hard-clip limits. Use this instruction to determine the plotter unit dimensions of the area in which plotting can occur.

SYNTAX: OH;

Parameter	Response	Format	Range
none	X _{LL} , Y _{LL} , X _{UR} , Y _{UR}	integer	current hard-clip limits

REMARKS: The coordinates are always expressed in plotter units and represent the lower-left and upper-right corners of the hard-clip limits. After sending the OH instruction, have your program immediately read the plotter's output response.

RELATED

INSTRUCTIONS: PS, Page Size

ERRORS:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

OP, Output P1 and P2

USE: Outputs the X,Y coordinates (in plotter units) of the current scaling points P1 and P2. Use this instruction to determine the numeric coordinates of P1 and P2 when they have been set manually, and to help compute the number of plotter units per user unit when scaling is on. This instruction can also be used with the input (IW) instruction to programmatically set the window to P1 and P2.

SYNTAX: OP;

Parameter	Response	Format	Range
none	P1 _X , P1 _Y , P2 _X , P2 _Y	integer	-8 388 608 to 8 388 607*

*Except that P2 tracks P1 and may be outside this range.

REMARKS: The P1/P2 coordinates are output as plotter units. After sending the OP instruction, have your program immediately read the plotter's output response.

Upon completion of output, bit position 1 of the status word is cleared (refer to the output status (OS) instruction).

EXAMPLE: Note that your computer may use different statements and format to read input from a peripheral. Use whatever is required by your computer. The following reads output from the plotter and prints the information on the computer's screen.

```
10 'Insert configuration statement
20 PRINT #1, "IN;OP;"
30 INPUT #1, A,B,Y,Z
40 PRINT A,B,Y,Z
50 END
```

RELATED

INSTRUCTIONS: IP, Input P1 and P2
OS, Output Status

ERRORS:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

OW, Output Window

USE: Outputs the X,Y coordinates of the lower-left and upper-right corners of the window area in which plotting can occur. This instruction is especially useful when the window area (defined by IW) extends beyond the hard-clip limits.

SYNTAX: OW;

Parameter	Response	Format	Range
none	X _{LL} , Y _{LL} , X _{UR} , Y _{UR}	integer	current hard-clip limits

REMARKS: When scaling is on, the coordinates are expressed in user units; otherwise, they are in plotter units. When the window defined by IW extends beyond the hard-clip limits, the plotter outputs the coordinates of the intersection of the window and hard-clip limits. Have your computer read the output immediately.

Note that you may not be able to draw the window output by OW when scaling is on. Because the plotter rounds user unit values before output, the window coordinates may be outside the actual window.

EXAMPLE: Note that your computer may use different statements and format to read input from a peripheral. Use whatever is required by your computer. The following reads output from the plotter and prints the information on the computer's screen.

```
10 *Insert configuration statement
20 PRINT #1, "IN:OW;"
30 INPUT #1, L,F,U,R
40 PRINT L,F,U,R
50 END
```

RELATED

INSTRUCTIONS: IW, Input Window

ERRORS:

Condition	Error	Plotter Response
I or more parameters	2	ignores parameter(s)

RO, Rotate Coordinate System

USE: Rotates the plotter's coordinate system 90 degrees about the plotter-unit coordinate origin. This instruction allows you to orient your plot vertically or horizontally.

SYNTAX: RO *n*;
or
RO;

Parameter	Format	Range	Default
<i>n</i>	integer	0 or 90 degrees	0 degrees

REMARKS: The plotter interprets the values for the parameter *n* as follows.

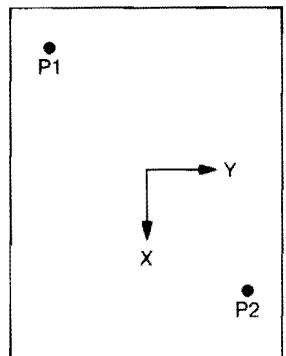
- **0** — Sets the orientation to horizontal. This is the same as sending no parameter (RO;).
- **90** — Rotates the coordinate system 90 degrees about the plotter-unit coordinate origin.

Rotations are not cumulative; you can toggle the rotate function on and off only. Scaling points P1 and P2 rotate with the coordinate system. However, they maintain the same X,Y coordinate values as before the rotation. This means that P1 and P2 can be located outside of the hard-clip limits. Follow (RO90;) with (IP;) to relocate points P1 and P2 within the hard-clip limits.

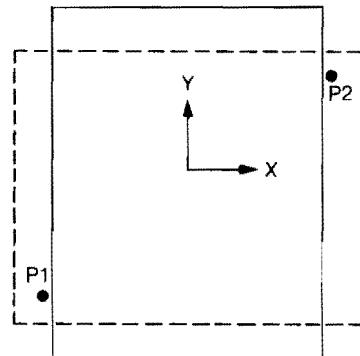
Note that the physical location of the pen does not change when you rotate the coordinate system. Instead, the plotter updates the pen's X,Y coordinate location to reflect the new orientation. Obtain the coordinates of the new pen location by sending an output actual pen location (OA) or output commanded pen location (OC) instruction after the rotation (refer to Chapter 14).

The RO instruction remains in effect until the rotation is changed by another RO instruction or you use the front-panel **Reset** or turn the plotter off and on.

EXAMPLE: The following illustration shows the default orientation and the result of rotating the orientation without relocating P1 and P2.

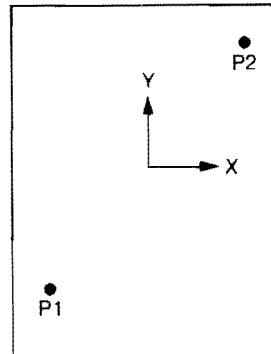


Default



RO 90 ;

The next illustration shows the locations of P1 and P2 when you follow the rotation with the IP instruction. (This is the same rotation you would get with the front-panel **Reset**.)

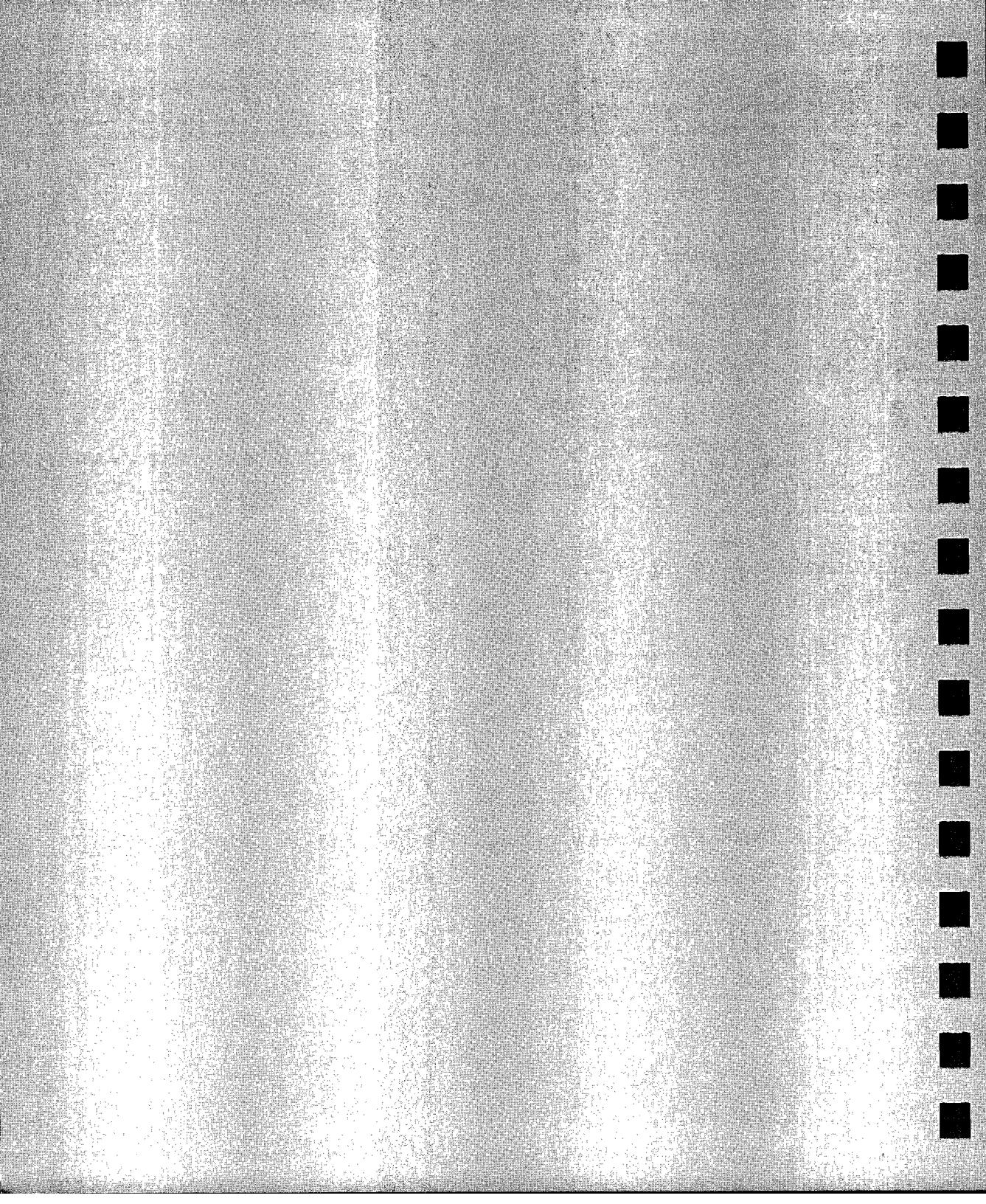


RO90 ; IP ;

RELATED**INSTRUCTIONS:** IP, Input P1 and P2**ERRORS:**

Condition	Error	Plotter Response
more than 1 parameter	2	uses first parameter only
number out of range	3	ignores instruction

Notes



10

Advanced Pen Control and Front-Panel Interaction

Instructions Covered

AP, Automatic Pen Operations

AS, Acceleration Select

CV, Curved Line Generator

FS, Force Select

GP, Group Pen

KY, Define Key

NR, Not Ready

OK, Output Key

SG, Select Group

VS, Velocity Select

WD, Write to Display

Terms You Should Understand

Front-panel Display

Function Keys

Using the Advanced Pen Control Features

The plotter automatically determines pen speed, pen force, pen acceleration, and the automatic pen operations from the carousel type. If you want to ensure that the right settings are used regardless of the carousel type, you can programmatically establish or change any of these settings using the velocity select (VS), force select (FS), acceleration select (AS), and automatic pen operations (AP) instructions.

Changing the settings allows you to optimize pen life and line quality. In addition, you can set the pen stalls to different settings if your application requires it. The automatic pen operations also allow you to control bidirectional plotting and pen sorting.

Grouping Your Pens for Longer Life

When your plot uses the same color over a large area, whether the plot contains a lot of solid fill or a lot of intricate plotting (e.g., text such as Kanji) where line quality is crucial, you can set your plotter to use a series of pens of the same color for the plot.

By grouping the pens in this manner then selecting the group you want to use instead of the specific pen, you extend the writing distance beyond the limits of one pen. Use the group pen (GP) and select group (SG) instructions to set up and select pen groups.

Protecting Your Program From Front-Panel Override

The initialize instruction (IN) **cannot** override the following front-panel settings. This feature gives the user more flexibility when running the program. However, if the user changes **any** of the listed front-panel settings, **none** of the listed program instructions will be restored to their defaults by the initialize (IN) instructions.

Front-Panel Settings	Program Instruction
P1 and P2	IP, Input P1 and P2
Speed and Force	VS and FS, Velocity and Force Select
Rotate	RO, Rotate Coordinate System
Group	GP and SG, Group Pen and Select Group
Sort	AP, Automatic Pen Operations

If it is crucial that the user not change a particular setting, you must use the program instructions to establish it; even if you want to use the default parameters.

Programming the Front-Panel Display and Keys

For some applications you may need an ‘interactive program’ where you have the user perform certain activities at a given point. You can design interactive plotting programs by sending messages to the front-panel display and/or redefining front-panel function keys. For example, you might want the user to change pens at a certain point in the program. The program could pause and send a message to the display; such as, ‘*Change Pens and Press a Key*’. Then, when a key has been pressed, your program could continue.

The instructions that allow you to design interactive programs are:

- WD, Write To Display
- KY, Define Key
- OK, Output Key
- GC, Group Count
- OG, Output Group Count

You can use two primary methods to design interactive programs. One was described in the example above and involves sending a message that requires a function key to be pressed, then polling the plotter to determine if it has been. When the response is positive, you have a subroutine in your program perform the needed function. To implement this method, you would use the WD instruction to send the message to the display, then use the OK instruction to poll the plotter for a pressed key. This method is illustrated later, under the description of the OK instruction.

The other method involves sending a message to the front-panel display and redefining one or more of the function keys to perform any of 12 specific functions. A typical example is to define a function key to perform an ‘escape’ when it is pressed. That way if users discover they have loaded the wrong paper or pens, they won’t have to wait for the plot to finish to correct the error. The program would poll the plotter for the escape key being pressed. When the program detected the escape function, it would begin ‘flush mode,’ which causes all subsequent graphics instructions to be discarded until flush mode is terminated. Then the program could automatically start over, or it could terminate until the operator started it again.

For this type of interactive program, you would use the KY instruction to assign the new function to one of the four function keys. Then you would use the WD instruction to send a message to the display (such as the word ‘ESCAPE’ to label which key will perform the escape function). Finally, you would send the OG instruction to determine whether the user has activated the escape function, and to enable flush mode if he has. This method is illustrated under the description of the KY instruction.

AP, Automatic Pen Operations

USE: Controls automatic pen operations such as returning a pen to the carousel if it has been in the holder without drawing for a certain time.

SYNTAX: AP n;
or
AP;

Parameter	Format	Range	Default
n	integer	1 to 255	95

REMARKS: To turn on any subset of the pen operations, add the value of the parameters. For example, (AP3;) will turn on only operations 1 and 2. (AP7;) turns on operations 1, 2, and 4. (AP95;) turns on all of the automatic pen operations, except 128, converting SP instructions to SG instructions.

- **No Parameters —** Turns on all the automatic pen operations except for converting SP instructions to SG instructions. This is the default set.
- **n —** To turn on any subset of the automatic pen operations, add the value of the parameters.
 - 0 Disables automatic pen operations. Pens not lifted until commanded by PU instruction or front panel. Pens not stored until commanded by SP instruction or front panel. Pens retrieved immediately when selected by SP instruction.
 - 1 Lifts the pen if it has been down for the allotted time.*
 - 2 Stores the pen if it has not moved in the allotted time.* If unable to put the pen away, lifts the pen (if down).
 - 4 Does not retrieve a pen selected by the SP instruction until the new pen is required to draw.
 - 8 Merges consecutive pen up moves.

*The ‘allotted time’ depends on the carousel type installed in the plotter. If the carousel is for paper fiber-tip pens or for roller-ball pens (or the carousel has been removed)—the allotted time is 65 seconds. If the carousel is for transparency fiber-tip pens or drafting pens, the allotted time is 15 seconds.

- 16** Sorts the instructions in the pen sorting buffer according to pen number. The plotter then plots all the instructions for one pen before switching to another one. When pen sorting is off, plots are drawn in the order the instructions are received. Turn pen sorting off to debug a program. If ink smears or bleeds when shapes are outlined, turn pen sorting off.
- 32** Reserved.
- 64** Allows bidirectional plotting.
- 128** Automatically converts all SP instructions to SG instructions so you can group pens with software that does not use the SG instruction.

RELATED

INSTRUCTIONS: GP, Group Pens
SG, Select Group

ERRORS:

Condition	Error	Plotter Response
more than 1 parameter	2	uses first parameter
number is not a valid sum or number out of range	3	ignores instruction

AS, Acceleration Select

USE: Sets pen acceleration for one or all pens. The default acceleration is suitable for all recommended pen and media combinations. Slowing the acceleration may improve line quality if you are using heavier than recommended media.

SYNTAX: AS *pen acceleration* (,*pen number*);
or
AS;

Parameter	Format	Range	Default
pen acceleration	integer	1 to 4 g's	4
pen number	integer	1 to 8	all pens

REMARKS: Selects the pen acceleration in g's and the pen(s) to which the acceleration applies.

- **No Parameter** — Sets pen acceleration to the default value of 4 for all pens.
- **Pen Acceleration** — Specifies the actual acceleration in g's, represented by integers 1 to 4. The default value is 4. If you specify a value greater than 4, then 4 is used.
- **Pen Number** — Specifies the particular pen to which the acceleration instruction applies. If omitted, acceleration applies to all pens; if greater than 8, the pen number parameter is ignored and the acceleration will apply to all pens.

RELATED

INSTRUCTIONS: CV, Curved Line Generator
FS, Force Select

ERRORS:

Condition	Error	Plotter Response
more than 2 parameters	2	uses first 2 parameters
number out of range	3	ignores instruction

CV, Curved Line Generator

USE: Collects vectors (line segments) in the vector buffer so that they can be plotted as a group. This allows the plotter to plot in a continuous motion, rather than stopping and starting at each vector endpoint. As a result, curves appear smoother.

SYNTAX: CV *n*;
or
CV;

Parameter	Format	Range	Default
<i>n</i>	integer	0 or 1	1
input delay	integer	0 to 8 388 607 msec	100

REMARKS: Use this instruction to increase the line quality of curves. Or, you could use it when your program must perform computations for each vector, which would ordinarily cause the pen to pause between each vector.

- **No Parameter** — Turns on the curved line generator. This is the default value; same as (CV1;) and (CV;).
- **n** — Turns on and off the curved line generator.
 - 1 Turns on the curved line generator. Default setting. Same as (CV;).
 - 0 Turns off the curved line generator.
- **Input Delay** — The time that can elapse from the last vector received before the curved line generator will plot the collected vectors.

The input delay should be large enough to accommodate the longest time that will elapse between vectors being received by the vector buffer.

When the curved line generator is on, the plotter gathers vectors in the vector buffer until it receives a PU or PD instruction, the vector buffer is full, a vector has not been received in the amount of time established by the input delay, or an OA instruction is received.

Once one of these conditions has been met, the vectors in the buffer are released and are plotted sequentially without stopping. Pen acceleration and pen speed are reduced to increase smoothness of curves.

The CV instruction remains in effect until a new CV instruction is executed, or the plotter is initialized.

RELATED

INSTRUCTIONS: AS, Acceleration Select
FS, Force Select

ERRORS:

Condition	Error	Plotter Response
more than 2 parameters	2	uses first 2 parameters
number out of range	3	ignores instruction

FS, Force Select

USE: Sets pen pressure to the paper for one or all pens. Use this instruction to optimize pen life and line quality for each pen and paper combination.

SYNTAX: FS *pen force (, pen number);*

or

FS;

Parameter	Format	Range	Default
pen force	integer	1 to 8	varies with carousel
pen number	integer	1 to 8	all pens

Carousel Type	Default Pen Force
paper	24
transparency	24
roller ball	51
drafting	30

REMARKS: You can use the FS instruction to extend pen life by reducing the force. Also use FS when you want to use different pen types in the same carousel type.

The FS instruction remains in effect until a new FS instruction is received, the plotter is initialized, or set to default conditions. Also if the carousel type is changed, all pens default to the new force.

- **No Parameters** — Sets force to the default for the carousel type for all pens.
- **Pen Force** — Specifies force of pen tip on paper in grams according to the formula:

1 = 15 grams	5 = 45 grams
2 = 24 grams	6 = 51 grams
3 = 30 grams	7 = 57 grams
4 = 36 grams	8 = 66 grams

- **Pen Number** — If omitted, force applies to all pens; specifies pen to which force applies.

RELATED

INSTRUCTIONS: AS, Acceleration Select
CV, Curved Line Generator

ERRORS:

Condition	Error	Plotter Response
more than 2 parameters	2	uses first 2 parameters
number out of range	3	ignores instruction

GP, Group Pen

USE: Assigns pens of the same type/color to a group in order to extend the effective writing distance beyond the life of one pen.

SYNTAX: GP *group number* (, *pen number* (, *number of pens* (, *length*)));
or
GP;

Parameter	Format	Range	Default
group number	integer	1 to 8	all groups
pen number	integer	1 to 8*	specified group number
number of pens	integer	1 to 8*	1
length	integer	1 to 50 000 metres	100 metres

*Pen number + number of pens must be < 9.

REMARKS:

- **No Parameters** — Defines no pen groups; i.e., 8 groups of 1 pen.
- **Group Number** — Designates the group being defined. This is the number you use with the SG instruction.
- **Pen Number** — Is the pen number of the *first* pen in the group. If you omit the pen number, a group will consist of one pen whose number is the same as the group number.
- **Number of Pens** — Is the total number of pens in the group. The group begins with the pen specified by the pen number parameter and increases sequentially until the proper number of pens is reached.

The pen sequence will stop with pen 8, even if the number of pens in the sequence is less than the number of pens specified. If you omit the number of pens, the group will have one pen.

- **Length** — Is the length in metres at which the pens will switch. Numbers larger than 50 000 will be reduced to 50 000.

The GP instruction remains in effect until the plotter is initialized, reset to default conditions, or another GP instruction is received.

RELATED**INSTRUCTIONS:** SG, Select Group**ERRORS:**

Condition	Error	Plotter Response
more than 2 parameters	2	uses first 2 parameters
number out of range	3	ignores instruction

KY, Define Key

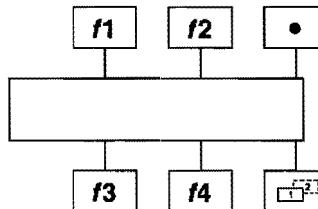
USE: Assigns a predefined function to one of the front-panel function keys. Use this instruction in conjunction with the WD instruction when designing interactive programs.

SYNTAX: KY *key*(,*function*);
or
KY;

Parameter	Format	Range	Default
key	integer	1 to 4	none
function	integer	1 to 12	none

REMARKS: The key definitions are only enabled while the plotter is in keyboard mode. In other words, if you send a KY instruction followed by a WD instruction, pressing a redefined function key will cause the new function to be performed. However, if you send a KY instruction but do not send a WD instruction, then pressing a 'redefined' function key will cause the normal, displayed function to be performed instead of the redefined function.

- **No Parameters** — Cancels all special definitions.
- **Key** — Specifies which of the four function keys will take on a new definition. The function keys are numbered as shown in the following illustration.



- **Function** — Defines which function will be assigned to the specified key. You can choose among the 12 functions listed below. Each function operates in the same manner as it does when in normal operating mode. For example, if you specify the 'pen up' function for key 4 (*KY4, 2;*), the plotter will lift the pen each time you press key 4, just as it does when you press the normal **Pen Up** key.

0	cancel key	7	rotate
1	view	8	force
2	pen up	9	align
3	pen down	10	reset
4	P1	11	clear
5	P2	12	escape
6	speed		

To cancel the special function, specify the key parameter and then omit the function parameter, or specify 0 (zero) for the function parameter (*KY4, 0;*). To cancel all keys, send the KY instruction without parameters (*KY;*).

The KY instruction only redefines one key at a time. To redefine all four keys, you must send the KY instruction four times, once for each key.

NOTE: While a function key is redefined by the KY instruction, the OK instruction cannot detect whether that key has been pressed. The ESC.O instruction can still detect that a function key has been pressed, but it cannot confirm *which* key was pressed. ■

The KY instruction remains in effect until another KY instruction redefines or cancels the *same* function key, or the plotter is initialized.

EXAMPLE: The following program is very simple and only shows the concept of how you might redefine function keys in an interactive program.

First, the program redefines f1 to be P1, f2 to be P2, and f3 to be **Escape**. Next the WD instruction displays labels for each key (blanks are included so that each label will be centered under the correct key). Finally, a loop is set up with the OG instruction for the escape function being activated (by f3 being pressed). When OG detects the function, it begins flush mode, which discards all incoming graphics instructions.

The OG instruction provides two pieces of information (refer to line 50)—the group number (C) and a number that indicates whether the escape function has been activated (E). (The group count number is most useful in RS-232-C spooling applications, so it is ignored in this program.) The escape function number (E) is checked in this program. If it is -1 (E=-1), the escape function has been activated so the program sends the end flush mode instruction (ESC.U), cancels all function key definitions, and displays the message **PROGRAM TERMINATED** on the front-panel display. Otherwise the loop continues by sending another OG instruction. (Refer to the OG instruction later in this chapter and the ESC.U instruction in Chapter 15.)

```
10  "Insert configuration statement here
20  PRINT #1, "IN;KY1,4;KY2,5;KY3,12;""
30  PRINT #1, "WD    P1    P2      ESC"+CHR$(3)
40  PRINT #1, "OG;""
50  INPUT #1, C,E
60  IF E=-1 THEN PRINT #1, CHR$(27)+"^U;KY;WD      PROG
                           RAM      TERMINATED"+CHR$(3) ELSE
                           GOTO40
70  END
```

NOTE: Although this program does not do any plotting, paper must be loaded for it to run. ■

RELATED

INSTRUCTIONS: ESC.U, End Flush Mode
OG, Output Group Count
WD, Write To Display

ERRORS:

Condition	Error	Plotter Response
more than 2 parameters	2	uses first 2 parameters
number out of range	3	ignores instruction

NR, Not Ready

USE: Programmatically simulates pressing the front-panel **View** button. However, you cannot take the plotter out of the **View** state with the NR instruction.

SYNTAX: NR;

REMARKS: After an NR instruction is executed, the plotter moves the currently loaded media forward and enters the **View** state.

You must press the **View** button on the plotter before you can do any more plotting. Normally, you will have loaded new paper before pressing the **View** button.

OK, Output Key

USE: Outputs a number that indicates which, if any, of the front-panel function keys has been pressed. Use this instruction in conjunction with the WD instruction when designing interactive programs.

SYNTAX: OK;

Parameter	Response	Format	Range
none	key pressed	integer	0 to 4

REMARKS: After receiving the OK instruction, the plotter outputs an integer between 0 and 4, followed by the output terminator. The number indicates which key has been pressed, as follows.

- **Key Pressed** — Defines which key has been pressed.
 - 0 No key has been pressed, the plotter is not in keyboard mode, or the key has been redefined.
 - 1 Function key 1 has been pressed.
 - 2 Function key 2 has been pressed.
 - 3 Function key 3 has been pressed.
 - 4 Function key 4 has been pressed.

(Refer to the define key (KY) instruction for an illustration of the function keys and their associated numbers.)

When a function key is pressed, bit 9 in the extended status word is set and the corresponding key number is saved by the plotter. The plotter does not sense any subsequent keys being pressed, since the bit has already been set. Therefore, when the plotter receives the OK instruction, it outputs only the first key pressed. Upon completion of the OK instruction, bit 9 is cleared.

After you send the OK instruction, your program should read the plotter's output response. Refer to *Hints for Obtaining Plotter Output* in Chapter 14.

NOTE: If a function key has been redefined by the KY instruction, the OK instruction cannot detect when that key has been pressed. If you redefine the function keys, you can instead use the ESC.O instruction to detect when a key has been pressed. However, ESC.O does not determine which key has been pressed. ■

EXAMPLE: The following program uses the WD (write to display) and OK instructions together in an interactive program. Typically, you would follow line 80 with instructions for drawing a plot. However, you can also run this program as shown. You must load paper in the plotter before running the program, even though nothing will be drawn on the paper.

First, this program uses WD to display a prompt; blank spaces are included so that the prompt will be centered. Then a loop in the program repeatedly sends the OK instruction to determine when a function key has been pressed. (In this program, it does not matter which key has been pressed.) When a key has been pressed (the plotter outputs a number between 1 and 4 inclusively), the program could continue with its plot. To show when this occurs, this program displays the phrase **PENS ARE LOADED** and halts. The extra blank spaces are included in this message (line 80) in order to erase the characters from the previous message.

```
10  'Insert configuration statement here
20  PRINT #1, "IN;""
30  PRINT #1, "WD      LOAD PENS      AND PRESS KEY "+CHR$(3)
40  PRINT #1, "OK;""
50  INPUT #1, K
60  IF K<1 THEN 40
70  IF K>4 THEN 40
80  PRINT #1, "NDPENS ARE LOADED"           "+CHR$(3)
90  END
```

RELATED

INSTRUCTIONS: KY, Define Key
WD, Write To Display

ERRORS:

Condition	Error	Plotter Response
no parameters	none	outputs information
plotter not in keyboard mode (WD not in effect)	none	outputs 0
function key(s) redefined (KY in effect)	none	outputs 0
1 or more parameters	2	outputs information

SG, Select Pen Group

USE: Allows the plotter to select a predesignated group of pens. Use this instruction with the GP instruction to extend the effective writing distance beyond the limits of one pen.

SYNTAX: SG *pen number*;
or
SG ;

Parameter	Format	Range	Default
pen number	integer	0 to 8	none

REMARKS: The parameter applies to a pen group previously designated with the GP instruction. The SG instruction selects the pen or pens which belong to this group. When the group contains more than one pen, the plotter will sequentially select pens according to the length parameter of the GP instruction.

When no groups have been previously designated with a GP instruction, the SG instruction is equivalent to the SP (select pen) instruction. Accordingly, (SG0;) and (SG;) are equivalent to (SP0;)—any pen in the pen holder will be returned to the carousel.

RELATED

INSTRUCTIONS: AP, Automatic Pen Operations
GP, Group Pen
SP, Select Pen

ERRORS:

Condition	Error	Plotter Response
number out of range	3	ignores instruction

VS, Velocity Select

USE: Specifies pen speed. Use the instruction to optimize pen life and line quality for each pen and media combination. Create a slightly thicker line on any media by slowing the pen speed.

SYNTAX: VS *pen velocity (, pen number);*

or

VS ;

Parameter	Format	Range	Default
pen velocity	integer	1 to 60	varies with carousel type
pen number	integer	1 to 8	all pens

Carousel Type	Default Pen Speed
paper	50
transparency	10
roller ball	60
drafting	30

REMARKS: Use the VS instruction to increase the pen life and line quality produced by your pens. Slowing pen speed increases line quality.

The VS instruction remains in effect until another VS instruction is received, the plotter is initialized, or set to default conditions.

- **No Parameters** — Sets the speed for all pens to the default value.
- **Pen Velocity** — Specifies the pen speed in centimetres per second (cm/s). The selected velocity applies only when the pen is down. Pen movement with the pen up is executed at 60 cm/s.
- **Pen Number** — Applies the pen speed to a particular pen. When this parameter is omitted, the velocity applies to all pens.

Note that you can also set pen speed with the front-panel **Speed** button. However, the pen speed you specify using the front-panel buttons applies to all pens. Refer to the User's Guide for more information.

RELATED

INSTRUCTIONS: AS, Acceleration Select
FS, Force Select

ERRORS:

Condition	Error	Plotter Response
more than 2 parameters	2	extra parameter ignored
number out of range	3	instruction ignored



WD, Write to Display

USE: Causes the specified message to be displayed on the front panel of the plotter and establishes keyboard mode. Use this instruction when designing interactive programs.

SYNTAX: WD *c...c* CHR\$(3)
or
WD CHR\$(3)

Parameter	Format	Range	Default
<i>c...c</i> (up to 32 characters)	label	any character except NULL , ETX , ENQ , ESC , and DEL (decimal codes 0, 3, 5, 27, and 127 respectively)	none

REMARKS: When you send a WD instruction with characters, the plotter displays the specified characters and establishes keyboard mode (described below). The plotter displays up to 32 characters (2 lines of 16 characters each); any excess characters are ignored. The characters are displayed using ANSI ASCII English characters. Therefore, the current character set has no effect on the WD instruction. You can send the actual character or use a function such as CHR\$ to specify the decimal code of a character.

NOTE: The WD instruction is similar to other labeling instructions in that it requires a label terminator. The default label terminator is **ETX** (CHR\$(3)), but may be redefined by the define label terminator (DT) instruction. Refer to the label (LB) instruction and the DT instruction for examples using label terminators. ■

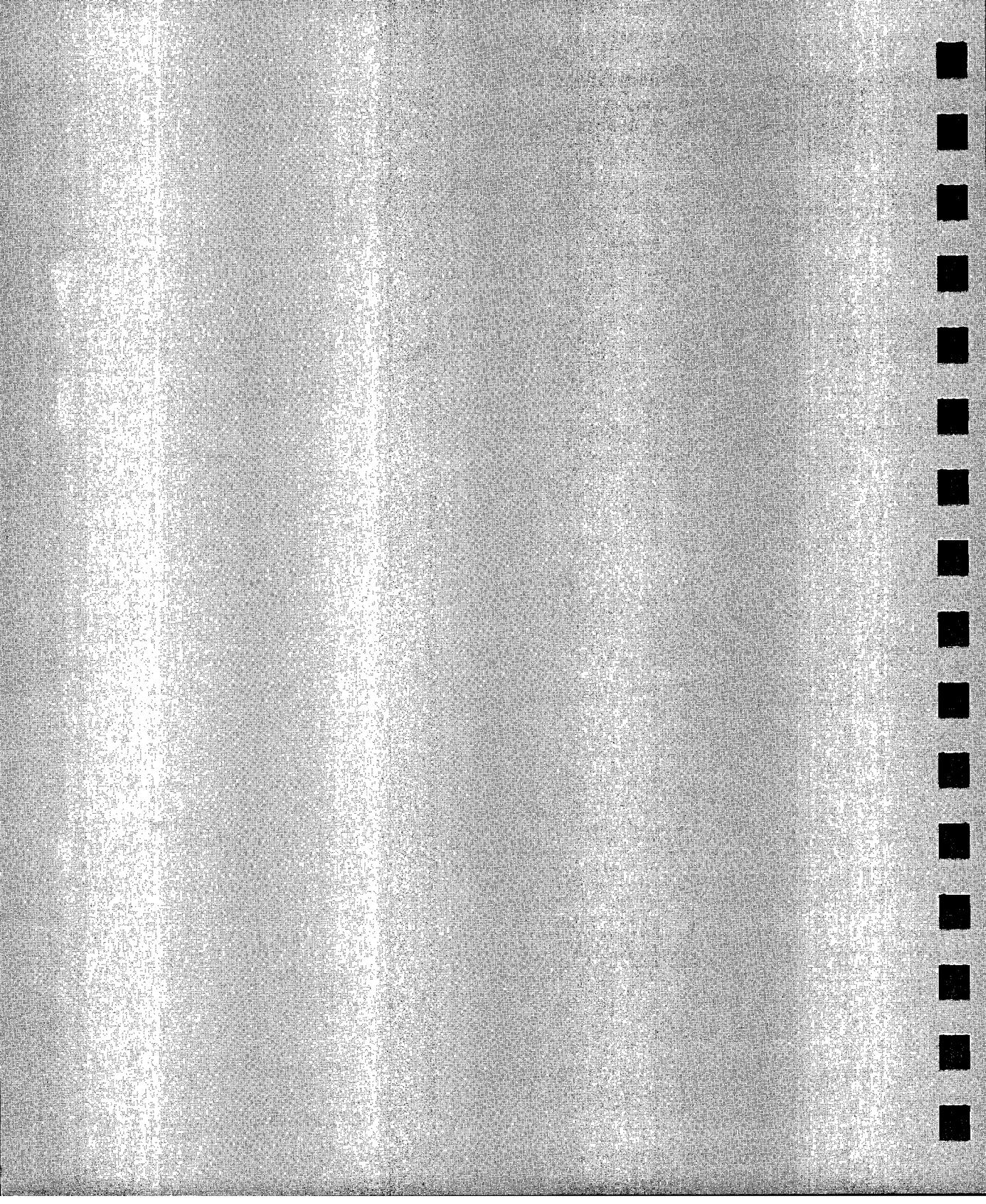
While in keyboard mode, the message is displayed and the function keys are available for use in a program. This means that you can use the OK or ESC.O instructions to detect whether a function key has been pressed, or you can use the KY instruction to redefine the function keys.

When a function key is pressed while the plotter is in keyboard mode, bit 9 of the extended status word is set. You can detect which key has been pressed with the OK instruction. Or, you can use the ESC.O device-control instruction to detect only that a function key has been pressed (but not which one). Thus, you can set up your program to poll for a key being pressed, and then include programming statements that respond by performing a certain function when a key has been pressed. For an example, refer to the OK, Output Key, instruction.

Alternatively, you can redefine the function keys using the KY instruction. In this case, the plotter will perform the redefined function when the associated function key is pressed. Refer to the KY, Define Key, instruction for an example.

If an error occurs while the plotter is in keyboard mode, the error overwrites the WD message and temporarily disables keyboard mode. To return to the keyboard mode, press any front-panel function key, or clear the error by sending an OE instruction.

The WD instruction remains in effect until another WD instruction overwrites the message, a WD instruction without characters clears the display, or the plotter is initialized.



11

Digitizing

You can use your plotter to digitize as well as plot graphics. Digitizing means moving the pen or digitizing sight to a point on the plotting surface, entering the point, and sending the X,Y coordinates of that point to the computer. The chapter discusses the instructions used in digitizing, along with the methods and procedures for digitizing and verifying the entry of a point.

You must already be familiar with how your computer reads information from the plotter. If not, refer to Chapter 14, *Obtaining Information From The Plotter*.

Using Digitizing instructions

The digitizing instructions, in the order used, are:

- DP, Digitize Point
- DC, Digitize Clear
- OD, Output Digitize Point

Use the digitize point (DP) instruction to enter digitize mode. After entering the point, use the output digitized point (OD) instruction to send the X,Y coordinates of the point and the pen status (up/down) back to your computer. The digitize clear (DC) instruction clears and exits the digitize mode.

Preparing Your Plotter for Use as a Digitizer

Although you can use a pen for digitizing, we recommend you use a digitizing sight. Refer to the User's Guide for information on using a digitizing sight. Digitize with the sight in the pen down position for the highest accuracy.

Digitizing with the Plotter

Familiarize yourself with the digitizing instructions later in this chapter before reviewing the digitizing methods here. When digitizing, you must make sure that a point has been entered before attempting to retrieve that point. The following paragraphs show you three methods for digitizing and retrieving points.

Manual Digitizing

The manual method is the easiest digitizing method to understand. It is not efficient, however, when you want to enter many points, or where human intervention during program execution is not possible. The following steps detail a typical program using the manual method.

1. Put the plotter in digitizing mode by sending a DP instruction to the plotter. The message **DIGITIZE POINT** will display on the front panel.
2. Have the program display a message on the computer screen prompting you to enter a point.
3. Cause the program to pause until instructed to continue. Using the BASIC INPUT statement and entering an empty string when you are ready to continue will work on some systems. Some versions of BASIC use statements such as STOP, WAIT, or PAUSE.
4. Move the digitizing sight (or pen) to the desired point using the front-panel cursor control keys. Complete final positioning with the sight (or pen) down. Press the **ENTER** button on the plotter's front panel.
5. Cause the program to resume. The way you resume program execution depends on the statement you used to halt the program. If you use an INPUT statement in step 3, press the RETURN key on the computer.
6. Output the digitized information to the computer using the output digitized point (OD) instruction. Have your computer read the information (the X,Y coordinates and the pen status). Then take the necessary steps to process the digitized data.

Using this method, you do not need to monitor the status byte because the program does not proceed to the OD (Output Digitized Point) instruction until you enter a point and cause the program to resume.

Example — Digitizing Using the Manual Method

The following program digitizes a single point and displays the coordinates and pen status.

NOTE: Your computer may read input from the plotter differently than shown here. Refer to your computer documentation. ■

```
10  'Insert configuration statement here
20  PRINT #1, "DP;""
30  PRINT "Press the plotter's ENTER button to digitize your
     point."
40  PRINT:PRINT "Press your computer's RETURN key to continue."
50  INPUT N$"
60  PRINT #1, "OD;""
70  INPUT #1, X,Y,P
80  PRINT X,Y,P
90  END
```

Monitoring the Status Byte

The second digitizing method monitors bit position 2 of the plotter's status byte, which is set when a digitized point is available. Refer to the *OS, Output Status Instruction* description, in Chapter 14 for more information.

There are a variety of ways to monitor bit position 2, depending on the instructions available in the computer you are using. If there are instructions in your programming language to check bits directly, the third least significant bit (lsb) should be checked for the occurrence of a 1. If no bit operations are available, the status byte can be operated on arithmetically to check for the availability of a digitized point. Executing successive divisions by a power of two and checking the answer for an odd or even integer is a common way of monitoring bits without converting the number to binary form. The following steps detail a program using this method.

1. Send a DP instruction to the plotter.
2. Have the program display or print a message on the computer screen prompting you to enter a point.

3. Send an OS instruction followed by a loop dividing the status value and checking for a final odd or even integer.

When you press the ENTER button on the front panel, the X,Y coordinates and the pen status is stored and bit position 2 is set. The status value increments by the value of bit 2; your division yields the odd integer allowing the program to continue.

4. Send an OD instruction. Read and display the X,Y coordinates and pen status. Then take the necessary steps to process the digitized data.

Example — Digitizing by Monitoring the Status Byte

The following sequence of BASIC instructions checks the proper bit of the status byte. In line 60, use your computer's BASIC read statement to read the status byte into a variable called STATUS. In line 70 and 80 you must use an integer statement (e.g., INT) that truncates, not rounds.

```
10  'Insert configuration statement here
20  PRINT #1, "DP;"
30  PRINT "Press the plotter's ENTER button to digitize your
     point."
40  PRINT #1, "OS;"
50  INPUT #1, STATUS
60  STATUS = INT(STATUS/4)
70  IF STATUS = INT(STATUS/2)*2 THEN 40
80  PRINT #1, "OD;"
90  INPUT #1, X,Y,P
100 PRINT X,Y,P
110 END
```

Note that your computer system may require different BASIC statements than those presented here (refer to Chapter 1 and to your computer documentation).

Example — Digitizing Many Points

In many applications, you may need to digitize a large number of points. When the computer is used to monitor bit position 2, the data points may or may not be processed immediately. Generally, you need to allocate space for the total number of points to be digitized. Then, you can establish a loop to process the total number of points, calling a subroutine each time to check that a point has been entered.

A complete program example follows. When prompted to enter a point, use the cursor keys to move the digitizing sight to the desired location. Then press the **ENTER** button on the plotter. Continue for all 25 points. After all 25 points are entered, their coordinates will display on the computer's screen.

```
10  *Insert configuration statement here
20  DIM X(25),Y(25),P(25)
30  FOR C = 1 TO 25
40    PRINT #1, "DP;"
50    PRINT "Enter point ";C
60    GOSUB 140
70    PRINT #1, "OD;"
80    INPUT #1, X(C),Y(C),P(C)
90  NEXT C
100 FOR C = 1 TO 25
110   PRINT X(C),Y(C),P(C)
120 NEXT C
130 END
140 *Check bit 2 for digitized point
150 PRINT #1, "OS;"
160 INPUT #1, STATUS
170 STATUS = INT(STATUS/4)
180 IF STATUS = INT(STATUS/2)*2 THEN 150
190 RETURN
```

HP-IB Interrupts and Polling

This third digitizing method is for advanced programmers, who are thoroughly familiar with the HP-IB interface, polling techniques, and interrupts. Use this technique if your computer can perform useful tasks while waiting for the digitized point to be entered.

This method involves setting a value of 4 in the S-mask of the IM (Input Mask) instruction to cause the plotter to generate a service request when a digitized point is available. With an interrupt routine enabled for service requests, the computer can send a DP (Digitize Point) instruction to initiate digitizing, and then proceed with some other task until the digitized point is entered. When the point is available, the computer is interrupted by the service request, and program execution branches to the routine to process the digitized data.

This routine could send an OD (Output Digitized Point) instruction and read the digitized point, or it could perform bit checking of the plotter status byte if multiple S-mask values have been specified to generate the service request. The status byte can be obtained by serial polling or by sending an OS (Output Status) instruction. Because interrupts and polling are machine-dependent and beyond the scope of this manual, no examples are given. For information regarding interfacing and handshaking, refer to Chapter 6.

DC, Digitize Clear

USE: Terminates digitize mode. For example, if you are using an interrupt routine in a digitizing program to branch to another plotting function, use DC to clear the digitize mode immediately after branching.

SYNTAX: DC;

REMARKS: When the plotter receives the DC instruction, it terminates digitize mode and reactivates automatic pen lift.

RELATED

INSTRUCTIONS: DP, Digitize Point

ERRORS:

Condition	Error	Plotter Response
use of a parameter	2	executes instruction anyway

DP, Digitize Point

USE: Returns the X,Y coordinates of a selected point on a plot to the computer for later use. Use this instruction to input data for a graphics program or to obtain the coordinates of a point or points on a plot.

SYNTAX: DP;

REMARKS: This instruction suppresses automatic pen storage and lift as set by the AP instruction. That is, you have full control of the pen holder while the plotter is in digitize mode.

Use the front-panel **CURSOR CONTROL** buttons to move the digitizing sight to the desired location, lower the sight, then press the front-panel **ENTER** button. Pressing **ENTER** sets bit position 2 of the status byte, indicating a digitized point is available for output. Use the OD instruction to *retrieve* the X,Y coordinates of the point and the pen up/down position. You can display them on the computer screen or write them to a file.

EXAMPLE: Refer to *Digitizing With the Plotter* earlier in this chapter.

RELATED

INSTRUCTIONS: OD, Output Digitized Point
DC, Digitize Clear
OS, Output Status

ERRORS:

Condition	Error	Plotter Response
using a parameter	2	executes instruction

OD, Output Digitized Point and Pen Status

USE: Outputs the X,Y coordinates and up/down pen position associated with the last digitized point. Use this instruction after the DP instruction to return the coordinates of the digitized point to your computer.

SYNTAX: OD;

Parameter	Response	Format	Range
none	X,Y Pen Status	integer, current units	current hard-clip limits

REMARKS: The ranges of the X,Y coordinates are the hard-clip limits of the plotter. The pen position is either 0 (up) or 1 (down).

After sending the OD instruction, have your program read the plotter's output response. The timing of output depends on the interface you are using.

Executing an OD instruction clears bit position 2 of the status byte. Refer to the OS, Output Status, instruction.

EXAMPLE: Refer to *Digitizing With the Plotter* earlier in this chapter.

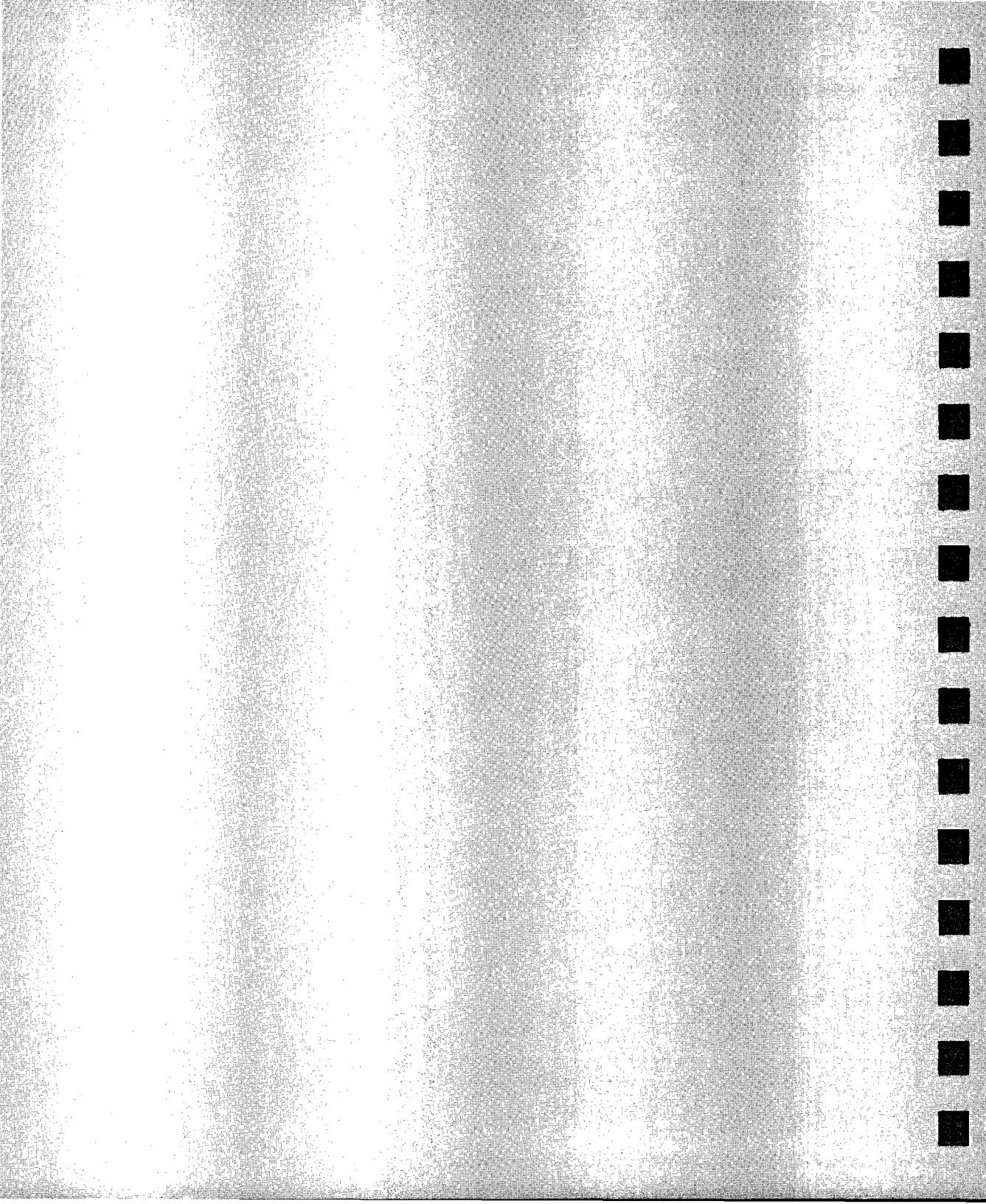
RELATED

INSTRUCTIONS: DP, Digitize Point
DC, Digitize Clear
OS, Output Status

ERRORS:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

Notes



Rollfeed Instructions and Long-Axis Plotting

Instructions Covered

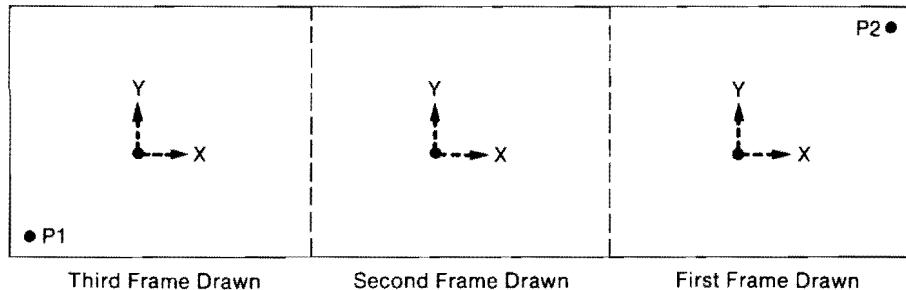
AF, Advance Full Page
AH, Advance Half Page
EC, Enable Cut Line
FR, Frame Advance
PG, Advance Full Page
PS, Page Size

Terms You Should Understand

Hard-Clip Limits	X,Y Coordinates
P1/P2	Origin
Window	Scaling

Long-Axis Plotting

You can use the advance frame instruction (FR) to align adjacently drawn frames to form the equivalent of a long axis plot. The following diagram shows a typical location of P1 and P2 in a long axis plot.



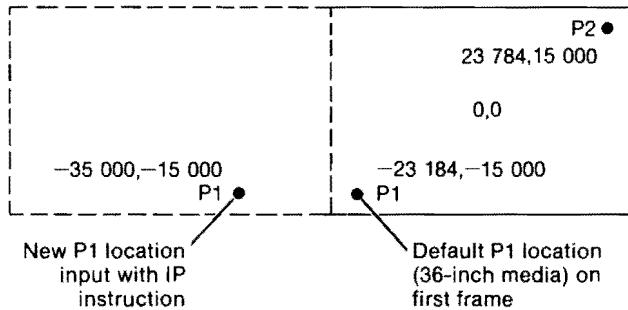
Use the IP instruction to change the X-coordinate of P1 to a large negative value. The frame advance normally occurs in the negative X-direction, as pictured above. Therefore the right-most frame is the first frame drawn. When the plot is rotated (90 degrees), frame advance will occur in the positive Y-direction and the bottom-most frame is the first frame drawn.

The plotter treats each frame as a separate window and plots only the data which falls within that frame. During the initial program sequence, the plotter draws only the plot data in the rightmost frame. After an execution of the FR instruction, the paper is advanced, all data is sent again, and the next frame is drawn.

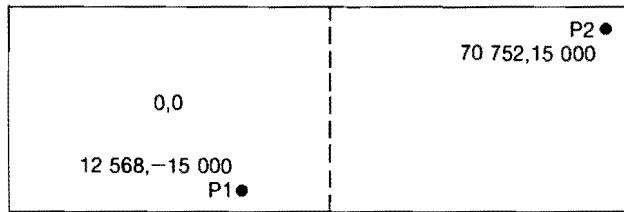
It is suggested that you use scaling for long axis plotting. Scaling is performed either in the plotter with the SC instruction, or in the controller, which uses P1/P2 values obtained from an OP instruction. If you define user units in the plotter with an SC instruction, it is not necessary to redefine coordinates after an execution of an FR instruction. When scaling is performed in the controller, you must use an OP instruction after each FR instruction in order to obtain the P1/P2 location with respect to the new coordinate origin.

With each frame advance, the plotter-unit origin relocates to the center of the current frame. The plotter adjusts P1 and P2 values to reflect this change. The *physical* location of P1 and P2 are retained, but the *logical* location relative to the current origin changes. You can see this in the following illustration.

Input P1, P2 Coordinates
 (The plotter-unit origin is on the first frame drawn.)



New Coordinate System Established After FR
 (The plotter-unit origin relocates on the next frame.)



The P1/P2 values are input as ($IP - 35\ 000, -15\ 000, 23\ 184, 15\ 000$). (The next section explains how to calculate a P1 X-coordinate and how to determine the number of frames your plot will require.)

After execution of FR, the logical values of P1 and P2 on the new frame become $12\ 568, -15\ 000, 70\ 752, 15\ 000$. Note that the span of the X-axis between P1 and P2 is 58 184 plotter units in each case. The plotter performs these computations internally so that, when scaling is in effect, the currently scaled window is mapped onto the current frame.

The plotter computes these values by drawing registration marks (directly on the X-boundary and 7.5 mm within the Y-boundary) to indicate the hard-clip limits of each frame. After a frame advance, the plotter digitizes the marks in order to properly align the next frame. These marks are drawn by a black .3 mm fiber-tip or drafting pen that **must** be installed in pen stall 8. If pen stall 8 is empty, no registration marks are drawn, and the proper frame alignment cannot be guaranteed.

The plotting area available after an FR instruction is the same as after a PG instruction, but the actual advance length of an FR instruction is shorter. The margins between plotting areas are deleted so that the frames share a common edge.

For optimum results, plot within the default P1/P2 location. This insures a minimal 30 mm margin between the media's edge and plotting area. Failure to maintain a 30 mm margin may cause the following:

- Plotting lines and registration marks are too close together for proper digitization of the marks, resulting in possible inaccuracy.
- The pinch wheels may track over the plotting area. Inherent rotation in the plotter's coordinate axis relative to the media's edges can cause subsequent frames to wander off the media's surface. The longer the plot, the greater the margin that you must have.

X-Axis Hard-Clip Limits

The plotter determines the advance length of a physical page according to the width of the paper loaded. The plotter also uses the width of the paper to automatically determine the discreet (in that they are not visibly apparent on roll media) X-axis hard-clip limits for one frame. Knowing these limits makes P1 and P2 predictable; you can calculate the length of a frame and thus determine the number of frames to set up in your plot. This feature is especially useful in a spooled environment when you cannot use an OP instruction to interrogate the plotter for hard-clip limit values.

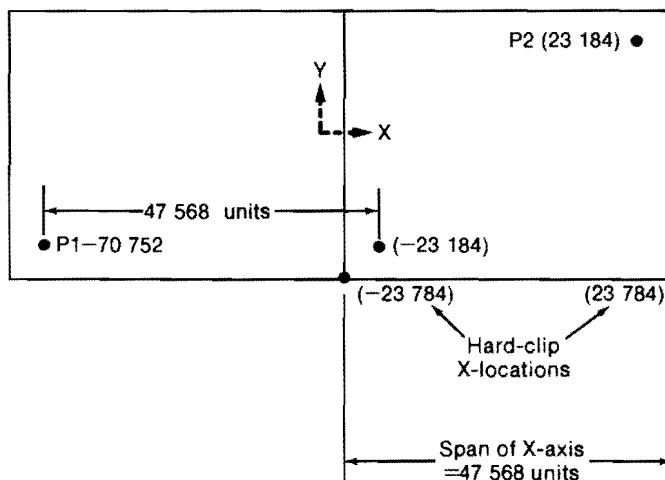
The hard-clip limits for the Y-axis will vary slightly according to the exact location of the pinch wheel. The X-axis hard-clip limits are fixed at default. The following table lists the default settings for one frame, based on paper width. P1 and P2 remain 15 mm (600 plotter units) inside of the hard-clip limits.

Roll Media Width	Default Hard-Clip X-Coordinates		Default P1, P2 X-Coordinates	
	X_{min}	X_{max}	X_{min}	X_{max}
914.4 mm/36 in.	-23 784	23 784	-23 184	23 184
609.6 mm/24 in.	-17 688	17 688	-17 088	17 088
279.4 mm/11 in.	- 8 036	8 036	- 7 436	7 436

Locating a P1 X-Coordinate on a Long Axis Plot

One simple formula for calculation of a P1 location is to multiply the number of frames added to the original frame times the difference in the X-axis hard-clip limits for one frame. (This gives you the hard-clip span of the X-axis.) Then subtract this amount from the default P1 location for the first frame. You can use the preceding table to determine the default P1 location.

Following are calculations for a two-frame plot on E-size paper. Only the X-coordinate locations are shown in the diagram.



$$P1 = \text{default P1 on first frame} - [\# \text{ additional frames} \times (X_{\max} - X_{\min})]$$

$$P1 = -23\ 184 - [1 \times (23\ 784 - (-23\ 784))]$$

$$P1 = -23\ 184 - [1 \times (47\ 568)]$$

$$P1 = -23\ 184 - [47\ 568]$$

$$P1 = -70\ 752$$

Media Absorbency

Liquid ink on vellum or polyester film will not dry fast enough to allow immediate spooling on the take-up roll after a plot is finished. In order to have sufficient drying time, you must program a delay before the page-advance instruction. The time required will vary according to relative temperature and humidity. At 50% relative humidity, 5 to 10 minutes drying time is sufficient with HP-recommended media and ink.

Stabilization of Paper, Vellum, and Tracing Bond

All non-plastic media (paper, vellum, and tracing bond) will expand or contract when exposed to changes in relative humidity. Any change in relative humidity can result in a change of up to $\pm 1\%$ in media size.

To ensure maximum plotting accuracy and repeatability, sheet and roll media should be exposed to the air and permitted to stabilize before plotting.

Inner layers of roll media will not stabilize until fully exposed to the air. Therefore, roll media cannot stabilize properly if it is unwound from the supply roll and plotted on immediately. It will expand or contract during plotting, resulting in an inaccurate plot.

To allow stabilization of roll media, the following programming practice is recommended before each plot.

1. Advance the page with a PG or AF instruction. This unwinds the media from the supply roll and exposes it to the air as it hangs in queue in the service loop.
2. Program a five-minute wait so that the media can stabilize.

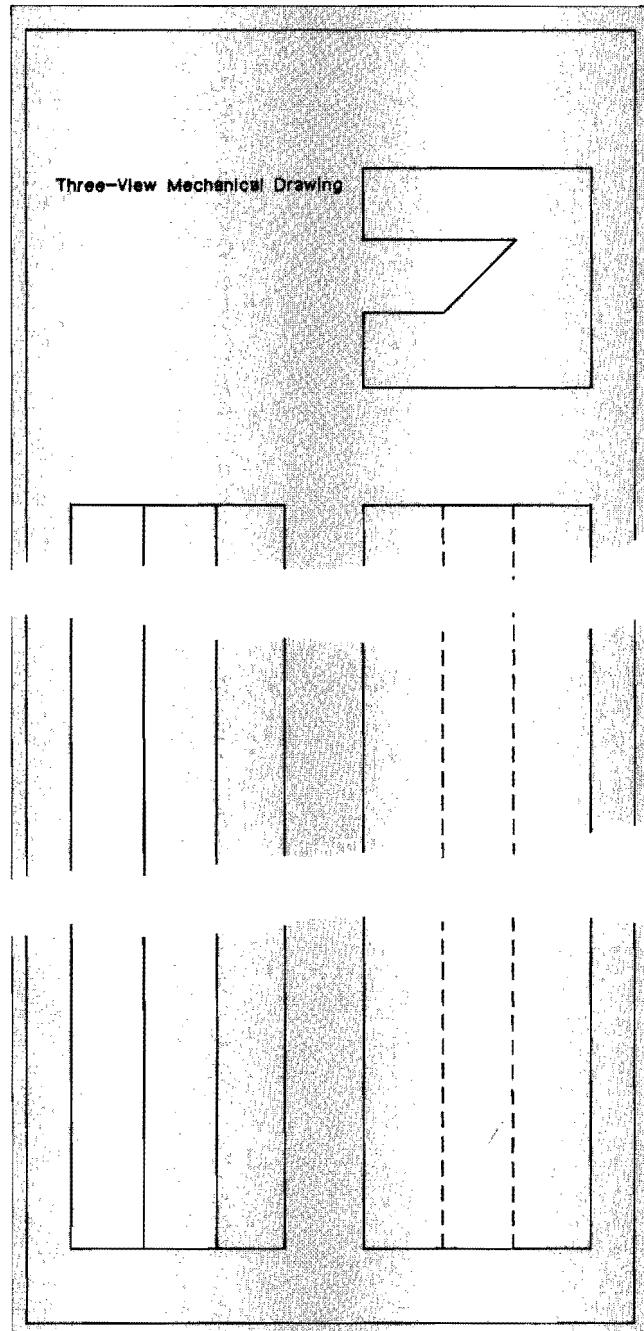
NOTE: After any five-minute wait without plotting, the plotter automatically makes two forward-backward passes of the media to develop grit tracks that will improve registration. ■

3. Send the data for the plot.

Example — Creating a Long Axis Plot

On the next page is a simple example of a three-view mechanical drawing plotted on E-size media. This program uses a subroutine to resend the block of data after the frame advance.

```
10  'Insert configuration statement here
20  PRINT #1, "IN;IP-70752,-16899,23184,16899;"
30  PRINT #1, "SC0,2789,0,1000;LT:L013;DI0,1;"
40  PRINT #1, "SI0.855,1.125;"
50  GOSUB 120
60  PRINT #1, "FR;"
70  GOSUB 120
80  PRINT #1, "SP0;"
90  END
100 '
110 '
120 PRINT #1, "SP1;PU;PA0,0;PD2780,0,2780,1000;"
130 PRINT #1, "PD0,1000,0,0;SP2;PU2502,50;PD914,50;"
140 PRINT #1, "PD914,408,2502,408,2502,50;"
150 PRINT #1, "PU2502,169;PD914,169;PU914,289;"
160 PRINT #1, "PD2502,289;PU2502,592;"
170 PRINT #1, "PD914,592,914,950,2502,950,2502,592;"
180 PRINT #1, "PU2502,711;LT3,1;PD914,711;PU914,831;"
190 PRINT #1, "PD2502,831;LT;PU636,592,517,592;"
200 PRINT #1, "PD517,711,397,831,397,592,278,592;"
210 PRINT #1, "PD278,950,636,950,636,592;PU278,50;"
220 PRINT #1, "LBThree-View Mechanical Drawing"+CHR$(3)
230 RETURN
```



12-8 Rollfeed Instructions and Long-Axis Plotting

AF, Advance Full Page

USE: Advances roll media one full page length and establishes the origin at the center of the new page.

SYNTAX: AF;

REMARKS: Page length is determined when the plotter senses page width. Media which is 11, 24, or 36 inches wide will have a page length of 17, 36, or 48 inches respectively.

The values of P1 and P2 and the plot rotation will be unchanged.

- If Automatic Take-Up is on (i.e., the front take-up spool is attached) the finished plot will be wound onto the take-up roll as the next page is pulled off the rear supply roll.
- If Automatic Take-Up is off (i.e., front take-up spool not attached) the plotter will advance the media and pause for you to cut the media and remove the plot.

RELATED

INSTRUCTIONS: AH, Advance Half Page
EC, Enable Cut Line
PG, Page Feed
PS, Page Size

AH, Advance Half Page

SYNTAX: AH;

USE: Advances roll media one half page length and establishes the origin at the center of the new page.

REMARKS: The plotter determines page length when it senses page width. Media which is 11, 24, or 36 inches wide will have a half page length of 8.5, 18, or 24 inches respectively.

The values of P1 and P2 and the plot rotation will be unchanged.

- If Automatic Take-Up is on the finished plot will be wound onto the take-up roll as the next page is pulled off the rear supply roll.
- If Automatic Take-Up is off (i.e., front take-up spool not attached) the plotter will advance the media and pause for you to cut the media and remove the plot.

RELATED

INSTRUCTIONS: AF, Advance Full Page

EC, Enable Cut Line

PG, Page Advance

EC, Enable Cut Line

USE: Draws a dashed cut line between ‘pages’ on roll media to indicate where to cut the media. Used with AF, AH, and PG instructions.

SYNTAX: EC;

or
EC *n*;

Parameter	Format	Range	Default
<i>n</i>	integer	-8 388 608 to 8 388 607	none

REMARKS:

- **No Parameter** — draws cut line after every AF, AH, and PG instruction.
- ***n* (any number)** — turns off the EC function.

RELATED

INSTRUCTIONS: AF, Advance Full Page
AH, Advance Half Page
PG, Page Feed

FR, Frame Advance

USE: Advances media to the next plot frame and calculates a relative coordinate system for that frame. Use FR to do multi-frame long-axis plotting.

SYNTAX: FR ;

REMARKS: Performs a long-axis frame advance and alignment. The plotter draws registration marks on the present frame, digitizes the marks, advances the media, and digitizes the registration marks again. In this way it calculates a new coordinate system and aligns that system to the old one.

The values of P1 and P2 change by the advance length of the frame in order to keep the same relative location on the media. The length advanced with an FR instruction is shorter than that of an AF instruction; the margins between the plotting areas are deleted so that the frames share a common edge.

The FR instruction will align a full-size plot area (the area advanced with a PG or AF instruction), to the present plot area, which can be either full or half size. When executed, FR clears the polygon buffer.

EXAMPLE: Refer to *Creating a Long-Axis Plot* earlier in this chapter.

PG, Page Feed

USE: Advances roll media one page length and establishes the plotter-unit origin at the center of the new page.

SYNTAX: PG *n*;
or
PG ;

Parameter	Format	Range	Default
n	integer	-8 388 608 to 8 388 607	none

REMARKS: Interpret the parameters as follows.

- **n** — Any integer (within the plotter's range) included as a parameter in the PG instruction forces a page feed whether or not you have plotted on the media. Executing a (PG *n*;) instruction is equivalent to pressing **Page Advance** on the front panel.
- **No Parameters** — The PG instruction issues a page feed only if you have plotted on the current page.

RELATED

INSTRUCTIONS: AF, Advance Full Page
AH, Advance Half Page

ERRORS:

Condition	Error	Plotter Response
number out of range	3	ignores instruction

PS, Page Size

USE: Changes the size of the hard clip limits while keeping the origin in the center of the plot. This allows for variable length page advances with the AF, AF, FR, and PG instructions.

SYNTAX: PS *length(, width);*
or
PS;

Parameter	Format	Range	Default
length	integer	0 to 8 388 607 plotter units	none
width	integer	0 to 8 388 607 plotter units	none

REMARKS: The plotter unit origin is maintained in the center of the plot.

- **Length** — Establishes the new length, in plotter units, of the hardclip limits. The length is *always* in the rollfeed direction of the media.
- **Width** — Establishes the new width, in plotter units, of the hardclip limits. The width is *always* the horizontal direction (between the pinch wheels).

The plotter unit axes will be oriented so that the X-axis is in the direction of the longer side of the plot, regardless of the size of the media itself. P1 and P2 will be defaulted to their usual locations inside the hardclip limits.

RELATED

INSTRUCTIONS: OH, Output Hardclip Limits

ERRORS:

Condition	Error	Plotter Response
number is ≤ 0	3	ignores instruction

Notes

12



Alternate Character Sets and User-Designed Characters

This chapter discusses how to designate and select any of 66 character sets in three fonts (fixed-space vector, variable-space arc, and fixed-space arc). Once you have selected a character set, you can use the labeling instructions presented in Chapter 7 to plot with the new characters. The plotter can label in four character selection modes; two HP modes and two ISO (International Standards Organization) modes. Usually you can use the default HP mode. However, if you label in languages other than English, you might find that one of the ISO modes is most flexible. Finally, you will learn how to design your own character, or even your own character set.

Instructions Covered

CA, Designate Alternate Character Set
CC, Character Chord Angle
CM, Character Selection Mode
CS, Designate Standard Character Set
DL, Define Downloadable Character
DS, Designate Character Set into Slot
IV, Invoke Character Slot
SA, Select Alternate Character Set
SS, Select Standard Character Set
UC, User-Defined Character

Terms You Should Understand

Carriage-Return Point
Character Origin
Character Plot Cell

Using Character Sets

A total of 66 character sets are available on the plotter. The plotter has 21 character sets available on each of three fonts. In addition to these 63 character sets, the plotter contains the Drafting set (set 99), the Downloadable set (set -1), and, if purchased, the optional Kanji set (sets 100 and 101). (Refer to Appendix D for information on the optional Kanji character set.)

Note that 'font' simply denotes a style of lettering, which is described thusly.

Fixed-space vector font: Characters all occupy an equal horizontal space and are always drawn using a fixed number of vectors.

Variable-space arc font: Characters are proportionately spaced; the amount of horizontal space occupied by each character varies with the character. The characters also have a contour smoothness which is programmable. Refer to the character chord angle (CC) instruction in this chapter.

Fixed-space arc font: Characters all occupy an equal horizontal space and contain the same programmable contour and smoothness as an arc-drawn character.

In the following table, each of the three columns represents a font. The character sets on the same line have the same characters in the same positions, differing only in the font.

Character sets 0, 10, and 20 from the three respective fonts are shown after the table. The difference in the total space required to plot the sets emphasizes the difference between fixed-space fonts and variable space fonts. Compare also the variation in the smoothness of the characters between the arc fonts and the vector font. All of the character sets (except Kanji) are shown in Appendix B. (Refer to Appendix D for the Kanji set.)

Fixed-Space Vector Font	Variable-Space Arc Font	Fixed-Space Arc Font	Character Set	ISO Registration Number
0	10	20	ANSI ASCII	006
1	11	21	9825 Character Set	—
2	12	22	French/German	—
3	13	23	Scandinavian	—
4	14	24	Spanish/Latin American	—
5	15	25	Special Symbols	—
6	16	26	JIS ASCII	014
7	17	27	Roman Extensions	—
8	18	28	Katakana	013
9	19	29	ISO IRV (International Reference Version)	002
30	40	50	ISO Swedish	010
31	41	51	ISO Swedish for Names	011
32	42	52	ISO Norway, Version 1	060
33	43	53	ISO German	021
34	44	54	ISO French	025
35	45	55	ISO United Kingdom	004
36	46	56	ISO Italian	015
37	47	57	ISO Spanish	017
38	48	58	ISO Portuguese	016
39	49	59	ISO Norway, Version 2	061
60	70	80	ISO French	069
99	—	—	Drafting	—

CHARACTER SET 0

```
! "#$%&' ()*+, -./0123456789: ; <=>?@  
ABCDEFGHIJKLMNOPQRSTUVWXYZ [\]^_`  
abcdefghijklmnopqrstuvwxyz {|}~
```

CHARACTER SET 10

```
!"#$%&'0*+,-./0123456789:;<=>?@  
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`  
abcdefghijklmnopqrstuvwxyz{|}~
```

CHARACTER SET 20

```
!"#$%&' ()*+, -./0123456789: ; <=>?@  
ABCDEFGHIJKLMNOPQRSTUVWXYZ [\]^_`  
abcdefghijklmnopqrstuvwxyz {|}~
```

All sets except Special Symbols (sets 5, 15, and 25), Roman Extensions (sets 7, 17, 27), and Katakana (sets 8, 18, and 28) draw identical upper- and lowercase letters and numbers. With these exceptions, the sets differ only in the additional characters that are needed in a certain language, for example, the *ç* in German set 2 and the *€* in Scandinavian set 3.

The Drafting Set

Set 99, the Drafting set, is a fixed-space, vector font designed to provide reliable character recognition in situations where photo reduction may cause image degradation and loss of resolution. The characters are drawn in a way to avoid confusion between lines and figures such as:

B 8 S
V U
5 6 S

The set also contains symbols used in drafting, such as ∞ or Δ . Refer to Appendix B to see the entire set. The following table describes some characters in more detail.

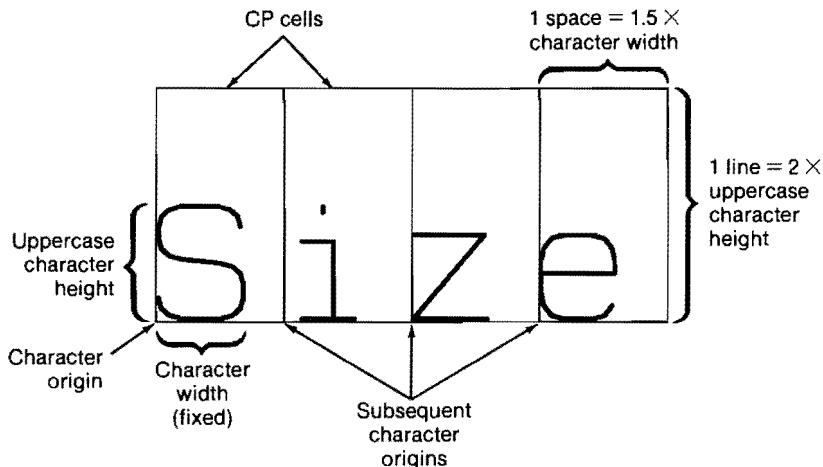
-B-	Bottom wider than top.
-6-	Large body, stem curved and open.
-8-	Lower part larger than upper, full and round to avoid blur.
-9-	Large body, stem curved but open.

Differences When Plotting with Variable-Space Fonts

By definition, the variable-space fonts don't use the same amount of space for each letter. This variance produces some differences in the way some of the labeling instructions work with variable-space fonts.

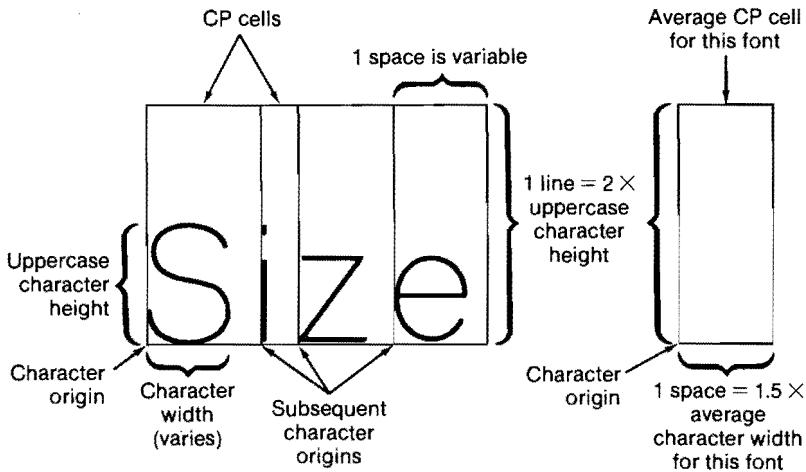
The Character Plot Cell

As you recall from Chapter 7, the character plot cell is defined as being 2 times the height of the uppercase A and 1.5 times the width. The example for fixed-space fonts is below.



Fixed-Space Font

In variable-space fonts, however, the width of a character varies; therefore, while the height of the CP cell remains 2 times the height of the uppercase A, the width is defined as 1.5 times the 'average character width'. Refer to the following illustration.



Variable-Space Font

Differences with Character Plot (CP) Instruction

The CP instruction (*CP spaces, lines;*) uses the ‘average character plot cell’ in computing lines and spaces. Otherwise, the instruction behaves the same for fixed- and variable-space fonts.

Differences with Extra Space (ES) Instruction

The ES instruction (*ES spaces, lines;*) uses the ‘average character plot cell’ in computing lines and spaces. Otherwise, the instruction works the same for fixed- and variable-space fonts. However as illustrated below, using ES with variable-space fonts tends to make them look as though they were fixed-space.

ES; CAUSES
THIS SPACING.

ES-.1,-.25; CAUSES
THIS SPACING.

ES.2,.25; CAUSES
THIS SPACING.

Character Set 10
(Variable Space)

Differences with Character Size (SI and SR) Instructions

Both the SI and SR instructions (*SI width, height;*) and (*SR width, height;*) use the ‘average character plot cell’ in calculating character size. Otherwise, both instructions behave the same as they do with fixed-space fonts.

The Downloadable Set and User-Defined Characters

If you need special label character(s) or symbol(s) that are not included in any of the character sets, you can design your own character or even an entire character set using the downloadable character (DL) and user-defined character (UC) instructions.

The UC instruction allows you to define a single character or design using X,Y increments on a grid superimposed on a character plot cell. The character plot cell used by the UC instruction varies with whether a fixed-space or variable-space font is currently selected.

The DL instruction allows you to define up to 94 characters that are stored in a buffer for repeated plotting. The character plot cell used by the DL instruction is always a fixed-space character cell, regardless of the font currently selected.

Character Selection Modes

The plotter can generate characters in any of four different modes. The default is HP 7-bit compatibility mode. With this mode, you can designate a standard set and an alternate set using the designate standard character set (CS) and designate alternate character set (CA) instructions, respectively. You can then label with both character sets either by selecting the desired set with the select standard character set (SS) and select alternate character set (SA) instructions, or by using a technique called shift-in/shift-out. When the plotter is turned on, initialized, or set to default conditions, it automatically selects set 0 (ANSI ASCII English) as both the standard and alternate character sets.

You can also use any of three other character selection modes: HP 8-bit, ISO 7-bit, and ISO 8-bit. In each of these modes, you can label in standard and alternate sets as described above. However, you have further options for accessing up to four character sets (instead of two) at a given time. To determine whether you want to use any of these alternate modes, refer to *Choosing Other Character Selection Modes* later in this chapter.

Designating and Selecting Character Sets

If you always intend to label with the default fixed-space set (ANSI ASCII English), you do not need to use the CS or CA instructions for designating standard and alternate character sets. However, if you intend to use a different set, you must use these instructions to designate sets before you can select those sets (using either SA or SS) for labeling. Refer to the descriptions for each of these instructions at the end of this chapter.

The methods described in this section are valid for any character selection mode. However, if you intend to use a mode other than the default HP 7-bit mode, refer also to *Choosing Other Character Selection Modes* and to the character selection mode (CM), designate into slot (DS), and invoke slot (IV) instructions.

Standard and Alternate Character Sets

The following outlines some of the principles to use when labeling with different character sets.

- Designate the standard and alternate character sets using the CS and/or CA instructions *before* labeling. If you are using set 0 as your standard set, you need specify only your alternate set.
- Select either the *designated* standard set or alternate set using either the SS or SA instruction before labeling.

Note that labeling always begins with the standard set. If you want to start with the alternate set first, use the SA instruction before you label.

- Switch from the standard character set to the alternate character set either using SS and SA or the shift-in/shift-out method. If you are changing sets within a label string, the shift-in/shift-out method is usually more efficient. Switch from the standard set to the alternate set using the shift-out **S0** ASCII control character (decimal code 14). Switch from the alternate set to the standard set using the shift-in **S1** ASCII control character (decimal code 15).

Special Characters

There are four ways to access special characters in any set.

- Use the equivalent ANSI ASCII English character on your keyboard in the label string. (Refer to the Character Sets and ASCII Codes table in Appendix B.) For example, to draw the character '1/2' in set 7, you can use the 'x' from an English keyboard.

`"CS7;LBx"+CHR$(3) 1/2`

- Use a computer language dependent function such as CHR\$ to enter the decimal code. For example, to draw the character '1/2' in set 7, use CHR\$(120).

`"CS7;LB"+CHR$(120)+CHR$(3) 1/2`

- Use the SS and SA instructions to shift sets to use a character from another set. Some characters, such as the underscore in set 4, automatically backspace before drawing the character. To use the backspacing characters, include them in the label string *after* the character to be underscored or accented.

```

10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;"
30  PRINT #1, "PA1000,1000;CS0;CA4;"
40  PRINT #1, "SS;LBS-E-T-0"+CHR$(3)
50  PRINT #1, "SA;LB S_E_T_4_" +CHR$(3)
60  PRINT #1, "CP-14,-2;LB#su compañia"+CHR$(124)+" ia?""
    +CHR$(3)
70  PRINT #1, "SP0:"
90  END

```

S_E_T_0_ SET4

¿su compañía?

- If you need to use a special character from another set in the middle of a label string, using SS and SA to toggle between sets can be inefficient. Instead, use the control characters shift-in (**SI**, decimal code 15) and shift-out (**SO**, decimal code 14) to toggle between the sets. You can use either the keyboard method or the CHR\$ method to shift-in/shift-out. The following example shows both methods.

```
10 'Insert configuration statement here
20 PRINT #1, "IN;SP1;"
30 PRINT #1, "PA300,300;CS30;CA7;SI.4.,6;SS;"
40 PRINT #1, "LB3"+CHR$(14)+"x"+CHR$(15)+"-5 ]R"+CHR$(3)
50 PRINT #1, "CP2.0;"
60 PRINT #1, "LB3"+CHR$(14)+CHR$(120)+CHR$(15)+"-5 "
    +CHR$(93)+"R"+CHR$(3)
70 PRINT #1, "SP0;"
80 END
```

3½-5 ÅR 3½-5 ÅR

Choosing Other Character Selection Modes

The default character mode on the plotter is the HP 7-bit compatibility mode. In this mode, you can access any character set using the CS, SS, CA, and SA instructions. You can also use the shift-in and shift-out control characters in a label string to access the designated standard and alternate sets, as described earlier.

The plotter also offers three additional character modes: the HP 8-bit mode, the ISO 7-bit mode, and the ISO 8-bit mode. The HP 8-bit mode is a tool for converting European software to HP systems software and will be of use primarily if you are using new HP systems software in a foreign language environment. The ISO modes permit more flexibility when you frequently want to access more character sets, yet conserve buffer space.

NOTE: If the default HP 7-bit mode is adequate for your labeling needs, you do not need to read this section or the explanations of the CM, DS, and IV instructions that follow. ■

The first three portions of this section discuss background information common to the character modes, specifically—the in-use code table, control characters, and fallback mode. The remaining portions of this section discuss each of the character selection modes. These discussions assume that you understand the basic principles of designating and selecting character sets with the CS, CA, SS, and SA instructions (refer to *Designating and Selecting Character Sets* earlier in this chapter). These discussions also assume that you understand the sequence of instructions to use when

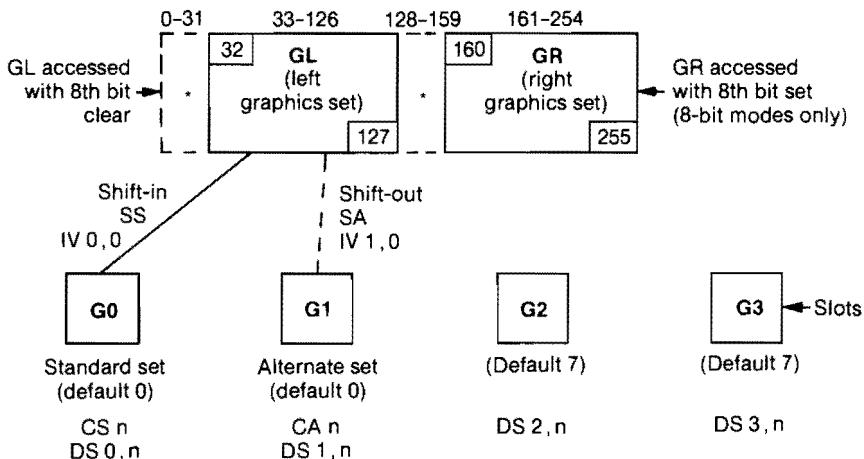
labeling in the alternative character modes. The steps for selecting a character mode and then labeling follow:

1. First execute the character selection mode (CM) instruction to establish one of the modes.
2. Then execute the designate character set into slot (DS) instruction to designate character sets (similar to the CS and CA instructions).
3. Finally, execute the invoke character slot (IV) instruction to select the set to be used for labeling (similar to the SS and SA instructions).
4. Now you can execute labeling instructions, and change set selections or designations as required.

The specific details of the syntax of the CM, DS, and IV instructions are presented following the discussions of the individual character selection modes.

The In-Use Code Table

Each character selection mode makes use of the 'in-use code table,' shown on the next page. This table presents the general concept of how the plotter selects character sets. In other words, this code table illustrates the underlying architecture for the selection of any character set in any mode. The descriptions of each mode later in this section present specific illustrations of how this table is interpreted for that particular mode.



*Codes 0-31 are control characters for all character sets, whether in GL or GR. Codes 128-159 are control characters for character sets in GR. For a list of recognized and ignored codes, refer to the section titled *Control Characters*.

G0 through G3 are the graphics slots. These slots designate which of the plotter's character sets may be selected (invoked into GL and GR) for labeling. When the plotter is initialized or set to default conditions, both G0 and G1 designate set 0, and both G2 and G3 designate set 7. Use the DS instruction to designate different character sets into these slots. (The CS and CA instructions are a subset of the DS instruction. If you are only interested in designating sets into slots G0 and G1, you can use the CS and CA instructions, respectively.)

GL and GR are the left graphics set and right graphics set, respectively. They contain the currently selected (invoked) character sets from the slots; all labeling instructions use the sets that currently reside in GL and GR. To place a character set into GL or GR, you would first use the DS instruction to designate character sets into the slots as described above. Then you would invoke one of the slots into GL or GR with the IV instruction. You can change the sets that reside in GL and GR by invoking different slots with new IV instructions. (The SS and SA instructions are a subset of the IV instruction. If you are only interested in invoking slots G0 and G1 into GL, you can use the SS and SA instructions, respectively. Depending on which mode you are using, you can also invoke slots by embedding shift-in, shift-out, and locking single shifts within label strings. These are discussed separately later with each mode.)

There are some restrictions on when the character sets residing in GL and GR can be accessed in labeling instructions. These restrictions depend on which character selection mode is in effect: both 7-bit modes can only access character sets in GL, whereas both 8-bit modes can access character sets in both GL and GR. The following paragraphs describe how character sets are 'stored' in GL and GR, depending on the mode.

- Both 7-bit modes can access a 128-character set that is contained in GL. Technically, only the printing characters (33–126) reside in GL. This is because the first 32 characters (0–31) of any set, regardless of mode, are nonprinting control characters. Characters 32 and 127 are also considered to be part of GL, although they are both nonprinting characters (32 is a space and 127 is ignored).

To print a character residing in GL, either use the character on your keyboard that represents the associated decimal code, or use the CHR\$ function (or equivalent for your computer) to specify the actual decimal code. For example, to print the character [(decimal 91) in set 32, you could enter [in the label string. Or, you could use the CHR\$(91) function. For more examples, refer to the section earlier in this chapter, titled *Labeling with Standard and Alternate Character Sets*. Appendix B contains a character set table with the decimal codes of all characters.

- Both 8-bit modes can access two 128-character sets that are contained in GL and GR. In effect, you are actually accessing one linked 256-character set. GL still contains characters 33–126, and you can print characters from GL in the manner described above. However, GR contains characters 161–254. To print a character from GR, you must use the CHR\$ function (or equivalent for your computer) and add 128 to the character's decimal code (as listed in Appendix B). Thus, if set 32 is

in GR and you want to print the character € , you would use the CHR\$(219) function ($128 + 91 = 219$).

In the 8-bit modes, characters 0–31 and 128–159 are control characters that do not technically reside in GL or GR. In fact, you can use characters in the range 0–31 regardless of whether you are currently labeling from GL or GR. However, you can only use characters in the range 128–159 if you are currently labeling from GR. Control characters are further explained in the next section.

13

Control Characters

The plotter recognizes only the following control characters:

- backspace, decimal code 8
- half backspace, decimal code 9
- line feed, decimal code 10
- inverse line feed, decimal code 11
- carriage return, decimal code 13
- shift-out, decimal code 14
- shift-in, decimal code 15

In addition, in the ISO 8-bit mode, the plotter recognizes characters 142 (single-shift from slot G2) and 143 (single-shift from slot G3).

All other control characters in the ranges 0–31 and 128–159 are ignored.

NOTE: The full range of invocations described in ISO 2022.2, which uses two-character escape sequences, is *not* operative in this plotter's version of the ISO modes. The escape character (decimal code 27) is trapped by the plotter's escape sequence parser, which implements the immediately executed device-control (ESC .) instructions. Therefore, if the escape character is placed in a label instruction, any subsequent character that is not part of the escape sequence will appear as part of the printed label. ■

Fallback Mode

The fallback mode determines the action taken when the plotter receives undefined printing characters in a label string. “Undefined printing characters” refers only to undefined characters within the ranges 33–126 and 161–254. Examples of undefined characters are those areas that are blank in the Katakana set (see the character set

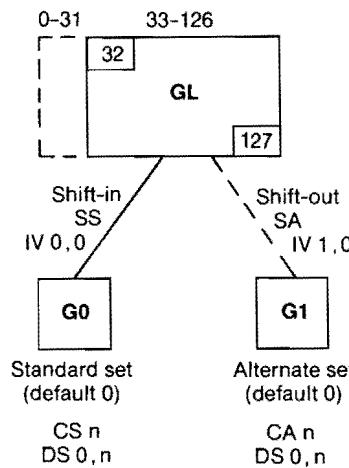
table in Appendix B). The downloadable character set also consists entirely of undefined characters until you use DL to define one or more characters.

The plotter's default response for the fallback mode is to ignore all undefined characters. However, you can use the CM instruction to specify that the plotter draw a box (□) as a fallback character in place of an undefined character. Some graphics standards mandate that character codes in the range 33–126 (sometimes 127) which do not have graphic representation be executed with a special printing character. The fallback mode supports this mandate. The fallback mode is also useful for debugging when the disappearance of characters from a label string might be confusing.

HP 7-Bit Compatibility Mode

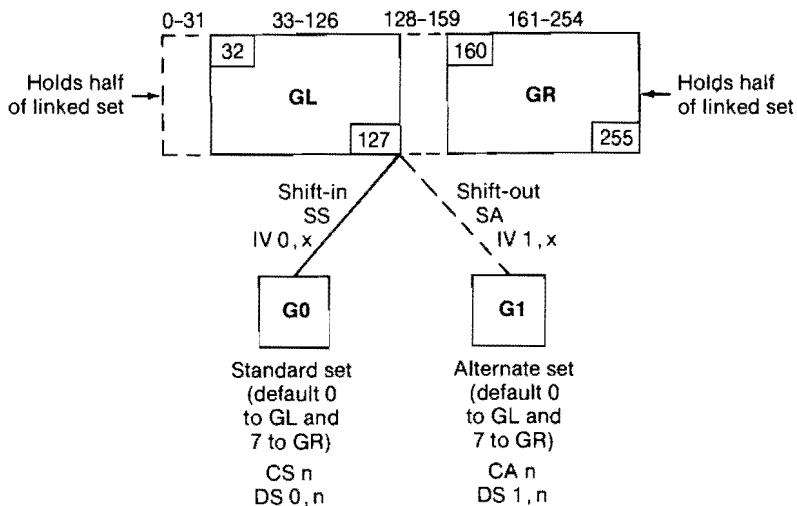
The HP 7-bit compatibility mode is the default mode implemented on all HP plotters. To use the HP 7-bit mode, you do not need to designate sets or invoke slots with the DS and IV instructions. Instead, you can simply access any of the plotter's character sets with the CS, SS, CA, and SA instructions. You can also change character sets within a label string by using the control characters shift-out (decimal code 14) to access the alternate set, and shift-in (decimal code 15) to access the standard set. Refer to the CS, SS, CA, and SA instructions, for syntax details and examples of using different character sets in this mode.

In the HP 7-bit compatibility mode, the eighth bit is always cleared and ignored. Thus, character sets can reside only in GL. All character sets are composed of 128 characters. The in-use code table for the HP 7-bit compatibility mode follows.



HP 8-Bit Mode

In the HP 8-bit mode, the plotter links two 128-character sets so that they form one 256-character set. Characters 0–127 reside in GL, and characters 128–255 reside in GR. Only slots G0 and G1 are used in this mode. The in-use code table follows.



There are only two full linked sets for the HP 8-bit mode: Roman8 and Katakana8. Roman8 is composed of ANSI ASCII and Roman Extensions, and Katakana8 is composed of JIS ASCII and Katakana.

To obtain the Roman8 linked set, designate ANSI ASCII as either the standard or alternate set. Characters 33–126 will be from the ANSI ASCII set, and characters 161–254 will be automatically lifted from Roman Extensions. Similarly, to obtain the Katakana8 linked set, designate JIS ASCII as either the standard or alternate set. Characters 33–126 will be from the JIS ASCII set, and characters 161–254 will be automatically lifted from Katakana. These paired relationships, shown in the table below, hold true regardless of whether fixed-space or variable-space fonts are designated.

Character Set Designated with CS, CA, or DS	GL Characters 33-126	GR Characters 161-254
ANSI ASCII (set 0 or 10)	ANSI ASCII (set 0 or 10)	Roman Extensions (set 7 or 17)
JIS ASCII (set 6 or 16)	JIS ASCII (set 6 or 16)	Katakana (set 8 or 18)

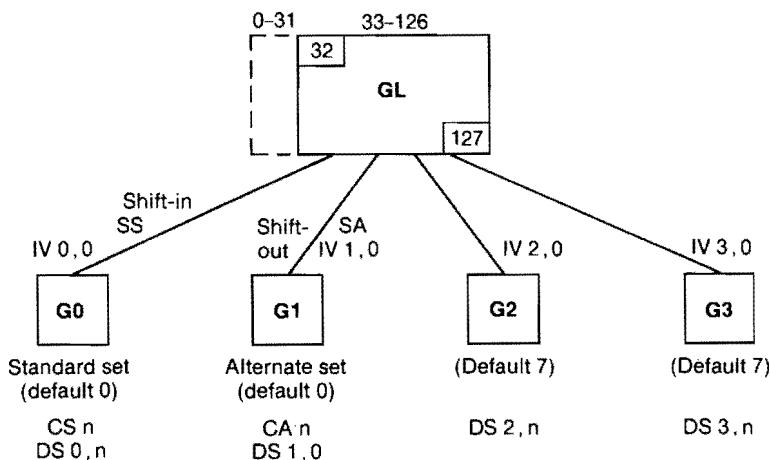
When you designate any set other than ANSI ASCII or JIS ASCII, that set is used for characters 33–126. However, in this case characters 161–254 are considered to be undefined characters and are treated according to the current fallback mode. The only way to set the eighth bit and therefore access characters in GR is to designate ANSI ASCII or JIS ASCII, as described above.

As with the HP 7-bit mode, you can change character sets within a label string by using shift-out (decimal code 14) and shift-in (decimal code 15). All invocations other than shift-in, shift-out, SS, SA, IV0,x;, and IV1,x; are ignored. In this mode, slots G2 and G3 cannot be designated with the DS instruction nor invoked with the IV instruction. However, G0 and G1 can be designated either with the DS instruction or the CS and CA instructions, and invoked either with the IV instruction or the SS and SA instructions.

NOTE: In order to access a character in GR, you must use a function such as CHR\$ (or equivalent for your computer). To determine the decimal code to use in the CHR\$ function, simply add 128 to the decimal code of the desired character. Thus, to label the character c in Roman8 Extensions, you would use the CHR\$(191) function (128 + 63). You can find the decimal codes of characters in Appendix B. ■

ISO 7-Bit Mode

In the ISO 7-bit mode, the eighth bit is always cleared and ignored. Thus, character sets can reside only in GL. All character sets are composed of 128 characters. When the plotter is initialized or default conditions are established, character set 0 resides in G0 and G1 and character set 7 resides in G2 and G3. The in-use code table for the ISO 7-bit mode follows.



The in-use code table shows which instructions may be used for designating character sets into slots, and for invoking slots into GL and GR. All invocations with the IV instruction are locking, meaning that the invoked set remains resident in GL until it is overridden with a new IV instruction or the plotter is initialized or set to default conditions. If you invoke a slot into GR, the plotter will automatically invoke the slot into GL instead. (For example, IV 3,1; which normally invokes slot 3 into GR will instead invoke slot 3 into GL.)

13

Example — Using the DS and IV Instructions in the ISO 7-Bit Mode

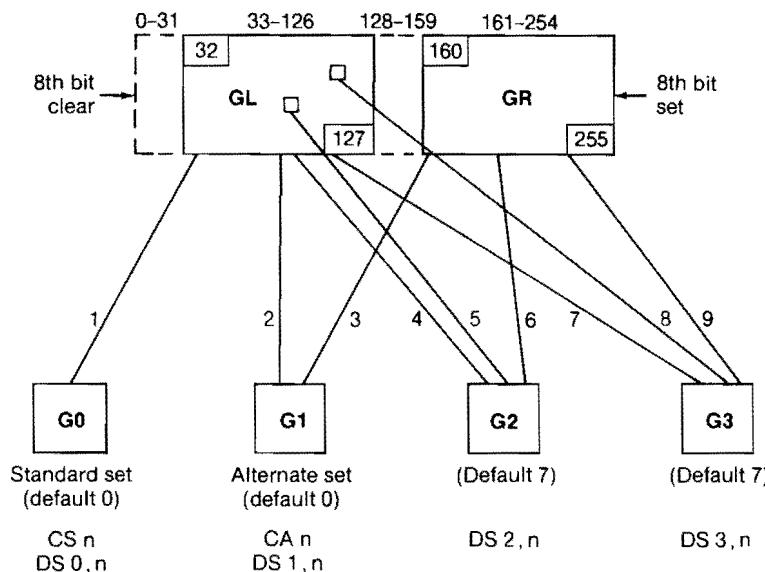
The following program illustrates the procedure for designating and invoking character sets when using the ISO 7-bit mode.

```
10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;SI.3,.4;PA100,1000;""
30  PRINT #1, "CM2;DS1,2;DS2,37;""
40  PRINT #1, "LBEnglish  "+CHR$(14)+"Fran"+CHR$(92)
     +"'ais"+CHR$(3)
50  PRINT #1, "IV2;LB  Espa"+CHR$(124)+"ol"+CHR$(3)
60  PRINT #1, "SP0;""
70  END
```

English Français Español

ISO 8-Bit Mode

The ISO 8-bit mode enables you to access two character sets at once without having to use invocations and designations. You can also access four character sets in any label string with the same economy. When the plotter is initialized or default conditions are established, character set 0 resides in G0 and G1 and character set 7 resides in G2 and G3. The in-use code table for the ISO 8-bit mode follows. The numbered items refer to the possible invocations of the slots into GL or GR. The possible designations of the slots are also shown.



- 1 use G0 for GL (shift-in, SS, or IV 0, 0)
- 2 use G1 for GL (shift-out, SA, or IV 1, 0)
- 3 use G1 for GR (IV 1, x)
- 4 use G2 for GL (IV 2, 0)
- 5 use G2 for next GL character only (single-shift G2)
- 6 use G2 for GR (IV 2, x)
- 7 use G3 for GL (IV 3, 0)
- 8 use G3 for next GL character only (single-shift G3)
- 9 use G3 for GR (IV 3, x)

You can invoke slots into GL using the IV, SS, or SA instructions, or by embedding shift-in (decimal code 15), shift-out (decimal code 14), and single-shift (decimal codes 142 and 143) control characters. However, the only way to invoke slots into GR is with the IV instruction.

Invocations can be locking or single. All invocations with the IV instruction are locking: the invoked set remains resident in GL or GR until it is overridden with a new IV, SS, or SA instruction, or until the plotter is initialized or set to default conditions.

A single invocation is in effect for one character only. Single-shift G2 (decimal code 142) invokes slot G2, and single-shift G3 (decimal code 143) invokes slot G3. After you make either single invocation, the plotter prints the next character from the single-invoked set, then returns to the previous character set.

NOTE: In order to access the single-shift control codes or a character in GR, you must use a function such as CHR\$ (or equivalent for your computer). To determine the decimal code to use in the CHR\$ function, simply add 128 to the decimal code of the desired character. Thus, to label an A when set 0 is in GR, you would use the CHR\$(193) function (128 + 65). You can find the decimal codes of characters in Appendix B. ■

Example — Embedding a Single-Shift in a Label String

The following example shows how to embed a single-shift character in a label string. The variable-space fonts of four character sets are used in this example: the ANSI ASCII English set (10), the French/German set (12), the Special Symbols set (15), and the Roman Extensions set (17).

```

10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;SI.3..4;PA100,2000;""
30  PRINT #1, "CM3;DS0,10;DS1,12;DS2,15;DS3,17;""
40  PRINT #1, "LBtrigonometric formula"+CHR$(3)
50  PRINT #1, "CP;LB"+CHR$(14)+“formule trigonomé
      rique”+CHR$(3)
60  PRINT #1, "CP;LBsec(""+CHR$(142)+“n-”+CHR$(143)+
      “v”)+CHR$(3)
70  PRINT #1, "LB = "+CHR$(142)+“psec(”+CHR$(143)+
      “v)”)+CHR$(3)
80  PRINT #1, "SP0;""
90  END

```

trigonometric formula
 formule trigonométrique
 $\sec(\pi - \theta) = \pm \sec(\theta)$

CA, Designate Alternate Character Set

USE: Designates a character set as the alternate character set to be used in labeling instructions. Use this instruction to provide an additional character set that you can easily access in a program.

SYNTAX: CA *set*;
or
CA;

Parameter	Format	Range	Default
set	integer	-1, 0-59, 60, 70, 80, 90, 100, 101	0

REMARKS: Specifies the character set to be used when the alternate set is selected by the SA instruction or by the control character shift-out (**\$0**, decimal 14) in a label string. Refer to *Standard and Alternate Character Sets* earlier in the chapter.

- **No Parameter** — Designates the default character set of 0 (ANSI ASCII English). Initializing or setting the plotter to default conditions will also do this.
- **Set** — Used for all labeling operations when the alternate set is selected by the SA instruction or by the control character shift-out (decimal 14) in a label string.
 - 1 Designates the downloadable character set you created using the DL instruction.
 - 0-9, 30-39, and 60** Designate fixed-space vector fonts.
 - 10-19, 40-49, and 70** Designate variable-space arc fonts.
 - 20-29, 50-59, and 80** Designate fixed-space arc fonts.
 - 99** Designates the fixed-space vector drafting font.
 - 100 and 101** Designate the two halves of the Kanji character set.

The CA instruction does not select the character set for current use. When you have designated a character set as the alternate, use the SA instruction to select it for current use. Once you have used the SA instruction to select the alternate character set, a new CA instruction will cause all subsequent labeling to be done with the new set.

However, if you execute a CA instruction while the standard set is selected (an SS instruction has been executed), subsequent labeling will not be done with the new alternate set until it is selected with an SA instruction.

EXAMPLE:

```

10 'Insert configuration statement here
20 PRINT #1, "IN;SP1;PA-1400,-1400;""
30 PRINT #1, "CS0;CA4;SR;""
40 PRINT #1, "SS;LBEEnglish"+CHR$(3)+"CP;""
50 PRINT #1, "LBor Spanish--"+CHR$(3)+"SA;LB#su compan"
      "+CHR$(124)+"ia?"+CHR$(3)
60 PRINT #1, "CP;CA33;LBor German--1{äßb sich anhand"
      "+CHR$(3)
65 PRINT #1, "SP0;""
70 END

```

English

or Spanish--¿su compañía?

or German--läßb sich anhand

RELATED

INSTRUCTIONS: CS, Designate Standard Character Set

SS, Select Standard Character Set

SA, Select Alternate Character Set

ERRORS:

Condition	Error	Plotter Response
more than 1 parameter	2	uses first parameter
number out of range	3	ignores instruction
invalid character set	5	ignores instruction

CC, Character Chord Angle

USE: Sets the chord angle that determines the smoothness of characters drawn when you select one of the arc-font character sets for labeling.

SYNTAX: CC *chord angle*;
or
CC;

Parameter	Format	Range	Default
chord angle	real	0.36 to 45 degrees*	5

*Practical range; allowable range is -8 388 608 to 8 388 607.

REMARKS: Each character in an arc-font character set is composed of arc segments with varying radii combined with straight lines. Use this instruction to decrease the number of chords in the arc segments, which reduces drawing time for rough or preliminary plots, or to increase the number of chords for arc-font sets when you are drawing very large letters, which increases smoothness.

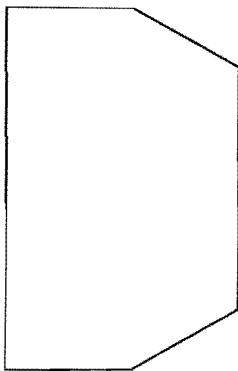
- **No Parameter** — Resets the chord angle to 5 degrees. The chord angle can also be reset to 5 degrees by a front-panel **RESET**, a DF or an IN instruction.
- **Chord Angle** — Specifies (in degrees) the angle of the chords drawn to create an arc segment in a character. Any chord angle greater than 45 degrees will be changed to 45 degrees. Decreasing the chord angle increases the number of chords and produces smoother arcs; however, this also increases the time required to draw the characters.

If you specify an angle larger than ± 45 , the plotter uses +45 degrees for the chord angle, since large chord angles would result in very poor character quality. If you specify an angle of 0 degrees, the plotter uses 0.36 instead.

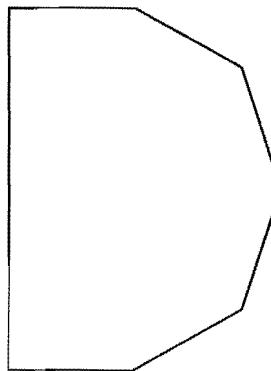
The CC instruction remains in effect until another CC instruction is executed, or the plotter is initialized or set to default conditions.

EXAMPLE:

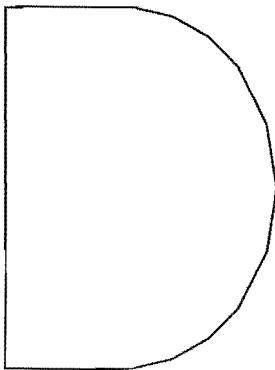
```
10 'Insert configuration statement here
20 PRINT #1, "IN;SP1;"
30 PRINT #1, "CS10;SI3.6,4.8;PA-3000,1000;"
40 PRINT #1, "CC45;LBD"+CHR$(3)+"CC30;LBD"+CHR$(3)
50 PRINT #1, "CP-2,-.75;CC15;LBD"+CHR$(3)
60 PRINT #1, "CC5;LBD"+CHR$(3)
70 PRINT #1, "SP0;""
80 END
```



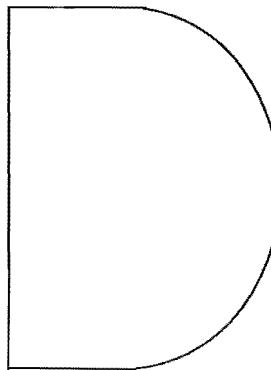
45-Degree chord angle



30-Degree chord angle



15-Degree chord angle



5-Degree chord angle

RELATED

INSTRUCTIONS: CA, Designate Alternate Character Set
 CS, Designate Standard Character Set
 LB, Label
 SA, Select Alternate Character Set
 SS, Select Standard Character Set

ERRORS:

Condition	Error	Plotter Response
more than 1 parameter	2	uses first parameter
number out of range	3	ignores instruction

CM, Character Selection Mode

USE: Specifies mode of character set selection and usage. Read *Choosing Other Character Selection Modes* earlier in this chapter to determine if you need this instruction.

SYNTAX: CM *switch mode (,fallback mode)*;
 or
 CM ;

Parameter	Format	Range	Default
switch mode	integer	0 to 3	0
fallback mode	integer	0 or 1	0

REMARKS: Use CM to access one of the four different character set selection modes: the HP 7-bit mode, the HP 8-bit mode, the ISO 7-bit mode, or the ISO 8-bit mode. The character set selection mode determines the implementation of the DS and IV instructions.

- **No Parameters** — Sets switch and fallback modes to their defaults of 0. You can also reestablish the default modes with the DF or IN instructions.

- **Switch Mode** — Specifies the character selection mode.
 - 0 HP 7-bit compatibility mode (default)
 - 1 HP 8-bit mode
 - 2 ISO 7-bit mode
 - 3 ISO 8-bit mode
- **Fallback Mode** — Specifies the plotter's action when it encounters undefined printing characters.
 - 0 Ignores undefined characters.
 - 1 Draws a box (□) in place of the undefined character.

Executing a CM instruction clears the label buffer. If you change to a different mode, the plotter sets the slot designations and invocations to their default values (refer to the DS and IV instructions).

EXAMPLE: Refer to *Choosing Other Character Selection Modes* earlier in this chapter for an example using CM.

RELATED

INSTRUCTIONS: DS, Designate Character Set into Slot
IV, Invoke Character Slot

ERRORS:

Condition	Error	Plotter Response
more than 2 parameters	2	executes first 2 parameters
number out of range	3	ignores instruction

CS, Designate Standard Character Set

USE: Designates a character set as the standard character set for labeling instructions. Use this instruction to change the default ANSI ASCII English set to one with characters appropriate to your application. This instruction is particularly useful if you plot most of your labels in a language other than English.

SYNTAX: CS *set*;
or
CS;

Parameter	Format	Range	Default
set	integer	-1, 0-59, 60, 70, 80, 90, 100, 101	0

REMARKS: Specifies the character set to be used when the standard set is selected by the SS instruction or by the control character shift-in (**\$I**, decimal 15) in a label string. Refer to *Special Characters* earlier in this chapter.

- **No Parameter** — Designates the default character set of 0 (ANSI ASCII English). Initializing or setting the plotter to default conditions will also do this.
- **Set** — Used for all labeling operations when the alternate set is selected by the SA instruction or by the control character shift-out (decimal 14) in a label string.
 - 1 Designates the downloadable character set you created using the DL instruction.
 - 0-9, 30-39, and 50 Designate fixed-space vector fonts.
 - 10-19, 40-49, and 60 Designate variable-space arc fonts.
 - 20-29, 50-59, and 70 Designate fixed-space arc fonts.
 - 99 Designates the fixed-space vector drafting font.
 - 100 and 101 Designate the two halves of the Kanji character set.

If you execute a CS instruction while the standard set is selected (e.g., an SS instruction has been executed), subsequent labeling will be done with the set designated by the new CS instruction. However, if you execute a CS instruction while the alternate set is selected (e.g., an SA instruction has been executed), subsequent labeling will not be done with the new standard set until it is selected with an SS instruction.

EXAMPLE: Refer to the CA instruction for an example using CS and CA.

RELATED

INSTRUCTIONS: CA, Designate Alternate Character Set

DL, Define Downloadable Characters

SS, Select Standard Character Set

SA, Select Alternate Character Set

ERRORS:

13

Condition	Error	Plotter Response
more than 1 parameter	2	executes first parameter
number out of range	3	ignores instruction
unsupported set	5	ignores instruction

DL, Define Downloadable Character

USE: Allows you to design characters and stores them in a buffer for repeated use.

SYNTAX: DL *character number (, pen control),
X-coordinate, Y-coordinate (,...,
(, pen control)(,...);*
or
DL *character number ;*
or
DL ;

Parameter	Format	Range	Default
character number	integer	33 to 126	none
pen control	integer	-128	none
X,Y coordinates	integer	-127 to 127 primitive grid units	none

REMARKS: Use this instruction whenever you want to create characters or symbols not included in the plotter's character sets.

- **No Parameters —** Clears the downloadable character buffer.

- **Character Number** — The decimal value (33–126) of the character. If the character number has been previously defined, the new definition overwrites the old one.
- **Pen Control (–128)** X When used, this parameter indicates that the next coordinate pair defines a move with the pen up. It is unnecessary to use the pen control directly after the character number as the first move is always made with the pen up.
- **X,Y Coordinates** — Absolute coordinates ranging from –127 to 127; drawn on a 32×32 unit grid. After the first pair, which always define a pen up move, all coordinates define moves with the pen down (unless preceded with pen control).

The downloadable characters are contained in character set –1 and must be designated and selected in the same manner as any other character set before you can use them.

Once you have defined downloadable characters and selected set –1, you can use them in LB and BL instructions. They can be downloaded in any order and redefined as needed.

How to Define a Downloadable Character

Define the character in absolute units on a 32×32 unit grid in a 48×64 unit cell. The origin (0, 0) is in the lower-left corner. This is the same grid used for the fixed-vector character sets in the plotter. The area occupied by a 32×32 unit grid approximates the size of an uppercase 'A'. The downloaded character may extend outside this grid to ± 127 units on each axis.

1. Allocate space in the downloadable character buffer with the GM or ESC .T instruction.

Each character requires:

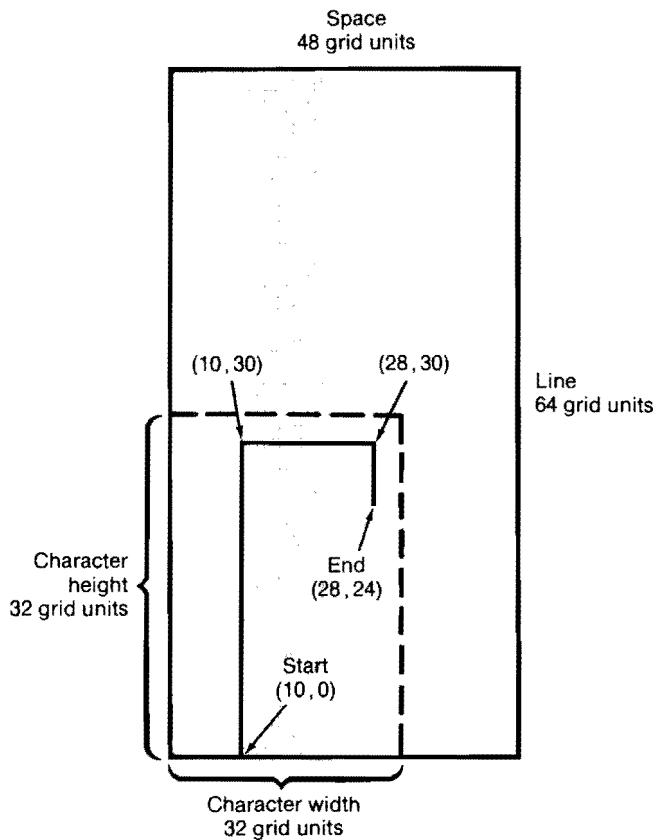
- one byte for each pen control parameter
- one byte for each X-coordinate
- one byte for each Y-coordinate
- one byte for the byte count

The minimum space required is 454 bytes for overhead, plus whatever you need according to the above formula for each character. For the gamma defined in the following example, you would need to allocate a minimum of 464 bytes: 454 for overhead, 1 for the implied pen down, 8 for the coordinates, and 1 for the byte count.

2. Assign a character number (decimal code) to the downloadable character.

- 13
3. Designate the starting point with the first X,Y coordinate pair (this will always be a pen up move.)
 4. Specify the vectors of the character using absolute X,Y coordinates.

The diagram below shows the grid for a gamma symbol and is followed by the program that draws it.



EXAMPLE:

```
10 'Insert configuration statement here
20 PRINT #1, "IN;GM4,464;SP1;CA-1;SI.3..4;"
30 PRINT #1, "DL65,10,0,10,30,28,30.28,24;"
40 PRINT #1, "PA0,0;"
50 PRINT #1, "LBThe symbol for gamma "+CHR$(3)
60 PRINT #1, "LBis "+CHR$(14)+"A"+CHR$(3)
70 PRINT #1, "SP0;"
80 END
```

The symbol for gamma is Ⓛ

RELATED

INSTRUCTIONS: BL, Buffer Label
CS, Select Alternate Character Set
LB, Label
SS, Select Standard Character Set

ERRORS:

Condition	Error	Plotter Response
more than 255 parameters	2	uses first 255 parameters
number out of range, odd no. of coordinates, or pen control with no coordinates	3	ignores instruction
downloadable character buffer overflow	7	erases both the new and previous definitions

DS, Designate Character Set into Slot

USE: Designates up to four character sets to be immediately available for plotting. Used with ISO character sets and modes.

SYNTAX: DS *slot, set;*
or
DS ;

Parameter	Format	Range	Default
slot	integer	0 to 1 (HP modes) 0 to 3 (ISO modes)	0
set	integer	-1, 0-59, 60, 70, 80, 100, and 101	0

REMARKS: A DS instruction without parameters (DS,) establishes default conditions. The following paragraphs list further details of each parameter.

- **Slot** — Represents the part of the in-use code table where character sets can be designated. Once it is designated, you can invoke any slot into GL or GR for use in a labeling instruction. For more details, refer to *The In-Use Code Table* and the character selection mode descriptions earlier in this chapter.
 - 0 Slot G0
 - 1 Slot G1
 - 2 Slot G2
 - 3 Slot G3
- **Set** — Is the number of the character set to be designated into the slot. The character set numbers are listed under *Using Character Sets* at the beginning of this chapter and in Appendix B.

The character selection mode (CM) instruction determines the implementation of the DS instruction, as follows. Refer to the separate descriptions of each character selection mode earlier in this chapter for examples and more details of how DS is implemented.

Character Selection Mode	Implementation of DS
HP 7-bit (default, CM 0)	DS 2, n and DS 3, n are ignored.
HP 8-bit (CM 1)	DS 2, n and DS 3, n are ignored.
	If DS 0, n or DS 1, n and n = 0 or 10, then set 7 or 17, respectively, is placed in G2 and G3.
	If DS 0, n or DS 1, n and n = 6 or 16, then set 8 or 18, respectively, is placed in G2 and G3.
	If DS 0, n or DS 1, n and n = -1, 1-5, 7-9, 11-15, 17-19, or 30-49, then null set is placed in G2 and G3; therefore G2 and G3 contain undefined characters.
ISO 7-bit (CM 2)	All forms of DS are accepted.
ISO 8-bit (CM 3)	All forms of DS are accepted.

Under default conditions, slots G0 and G1 each contain character set 0, and slots G2 and G3 each contain character set 7 (ignored in the HP modes). If you want to designate character sets into more than one slot, you must execute the DS instruction once for each slot. Remember that you can use the CS and CA instructions for designating character sets into slots G0 and G1, respectively.

The designation for a particular slot remains in effect until another DS (or CS or CA) instruction is executed for that slot, or the plotter is initialized or set to default conditions.

RELATED

INSTRUCTIONS: CM, Character Selection Mode
IV, Invoke Character Slot

ERRORS:

Condition	Error	Plotter Response
1 or more than 2 parameters	2	ignores instruction
slot number out of range	3	ignores instruction
set number out of range	5	ignores instruction

IV, Invoke Character Slot

USE: Invokes a character slot into either the right or left half of the in-use code table. Primarily used with ISO modes of character selection.

SYNTAX: IV *(slot, (left))*;
or
IV;

Parameter	Format	Range	Default
slot	integer	0 to 1 (HP modes) 0 to 3 (ISO modes)	0
left	integer	0 to 1	0

REMARKS: An IV instruction without parameters (IV;) establishes default conditions as follows: for all modes, G0 is invoked into GL; for ISO 8-bit mode, G1 is invoked into GR. The following paragraphs list further details of each parameter.

- **Slot** — Is the number of the slot to be invoked into the left or right half of the in-use code table (GL or GR, respectively). For more details, refer to *The In-Use Code Table* and to the descriptions of the character selection modes earlier in this chapter. The values for the slot parameters follow.
 - 0 Slot G0
 - 1 Slot G1
 - 2 Slot G2
 - 3 Slot G3
- **Left** — Specifies which side of the in-use code table will receive the character set, as follows. If this parameter is omitted, the slot is invoked into GL.
 - 0 Slot invoked into GL
 - 1 Slot invoked into GR

The character selection mode (CM) instruction determines the implementation of the IV instruction, as follows. Refer to the descriptions of each character selection mode earlier in this chapter for examples and more details of how IV is implemented.

Character Selection Mode	Implementation of IV
HP 7-bit (default, CM 0)	Slots G2 and G3 cannot be invoked. Also GR cannot be accessed explicitly. Therefore, IV 0,x (or IV 2,x) invokes slot G0 into GL; IV 1,x (or IV 3,x) invokes slot G1 into GL. Nothing is invoked into GR.
HP 8-bit (CM 1)	Same as above, except that GL and GR are linked; i.e., either the null set or the right half of a linked set is automatically invoked into GR, depending on which character set has been designated.
ISO 7-bit (CM 2)	All four slots can be invoked; however, they will automatically be invoked into GL, regardless of whether you specify GL or GR.
ISO 8-bit (CM 3)	All forms of the IV instruction are executed as specified.

NOTE: The instruction (IV0,0,) or (IV0,) is the same as (SS,) or a shift-in (**SI** decimal code 15) character in the label string. All invoke the standard set from slot G0 to GL. Similarly, (IV1,0,) or (IV1,) is the same as (SA,) or a shift-out (**SO** decimal code 14) character in the label string. All invoke the alternate set from slot G1 into GL.

All invocations with the IV instruction are locking. The invoked set remains in GL or GR until overwritten with a new instruction IV, SS, or SA), or until the plotter is initialized or set to default conditions.

RELATED

INSTRUCTIONS: CM, Character Selection Mode
DS, Designate Set into Slot

ERRORS:

Condition	Error	Plotter Response
more than 2 parameters	2	uses first 2 parameters
number out of range	3	ignores instruction

SA, Select Alternate Character Set

USE: The SA instruction selects the alternate character set (already designated by the CA instruction) for subsequent labeling. Use this instruction to shift from the currently selected standard set to the designated alternate set.

SYNTAX: *SA*;

REMARKS: The SA instruction tells the plotter to draw subsequent labeling instructions using characters from the alternate character set previously designated by the CA instruction. The SA instruction is equivalent to using the control character 'shift-out' (**\$0**, decimal 14) within a label string.

The default designated alternate character set is set 0. The alternate character set remains in effect until an SS instruction is executed, a 'shift-in' character (**\$1**, decimal 15) is encountered, or the plotter is initialized or set to default conditions.

RELATED

INSTRUCTIONS: CA, Designate Alternate Character Set
CS, Designate Standard Character Set
SS, Select Standard Character Set

ERRORS:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameters

SS, Select Standard Character Set

USE: The SS instruction selects the standard set (already designated by the CS, Designate Standard Character Set, instruction) for subsequent labeling. Use this instruction to shift from the currently selected alternate set to the designated standard set.

SYNTAX: SS;

REMARKS: The SS instruction tells the plotter to draw subsequent labeling instructions using characters from the standard character set designated by the CS instruction. The SS instruction is equivalent to using the control character 'shift-in' (**SI**, decimal 15) within a label string.

The default designated standard character set is set 0. Character set 0 is in effect when the plotter is initialized or set to default conditions. The SS instruction remains in effect until an SA instruction is executed.

RELATED

INSTRUCTIONS: SA, Select Alternate Character Set
CS, Designate Standard Character Set
CA, Designate Alternate Character Set



ERRORS:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameters

UC, User-defined Character

USE: The UC instruction draws characters of your own design. Use this instruction to create characters or symbols not included in your plotter's character sets or to draw logos.

SYNTAX: UC *(pen control,) X-increment, Y-increment (,...) (,pen control) (,...);*
or
UC;

13

Parameter	Format	Range	Default
pen control	integer	-9999 to -8 388 608 = pen up +9999 to +8 388 607 = pen down	pen up
X-, Y-increments	integer	-9998 to +9998	none

REMARKS: The characters are drawn using a primitive grid that is superimposed on the character plot (CP) cell.

- **Pen Control —** Integers $\geq +9999$ interpreted as pen down

Integers ≤ -9999 interpreted as pen up

- **X,Y Increments —** Integers between ± 9998 interpreted as primitive grid units

The primitive grid character plot cell is as follows:

1 space = 48 grid units

1 line = 64 grid units

Characters in a 32×32 grid will be the same size as characters drawn with LB or BL and PB instructions.

The primitive grid character plot cell for a *variable-space font* is as follows.

1 space = 42 grid units

1 line = 72 grid units

Characters in a 28×36 grid will be the same size as characters drawn with LB or BL and PB instructions.

The UC instruction initially raises the pen (regardless of the current pen status). In order to draw a character, then, you must include at least one pen down parameter. The pen will remain down for subsequent X,Y increments until you specify a pen up

parameter or terminate the UC instruction. You can include as many pen control parameters as you need to draw your character.

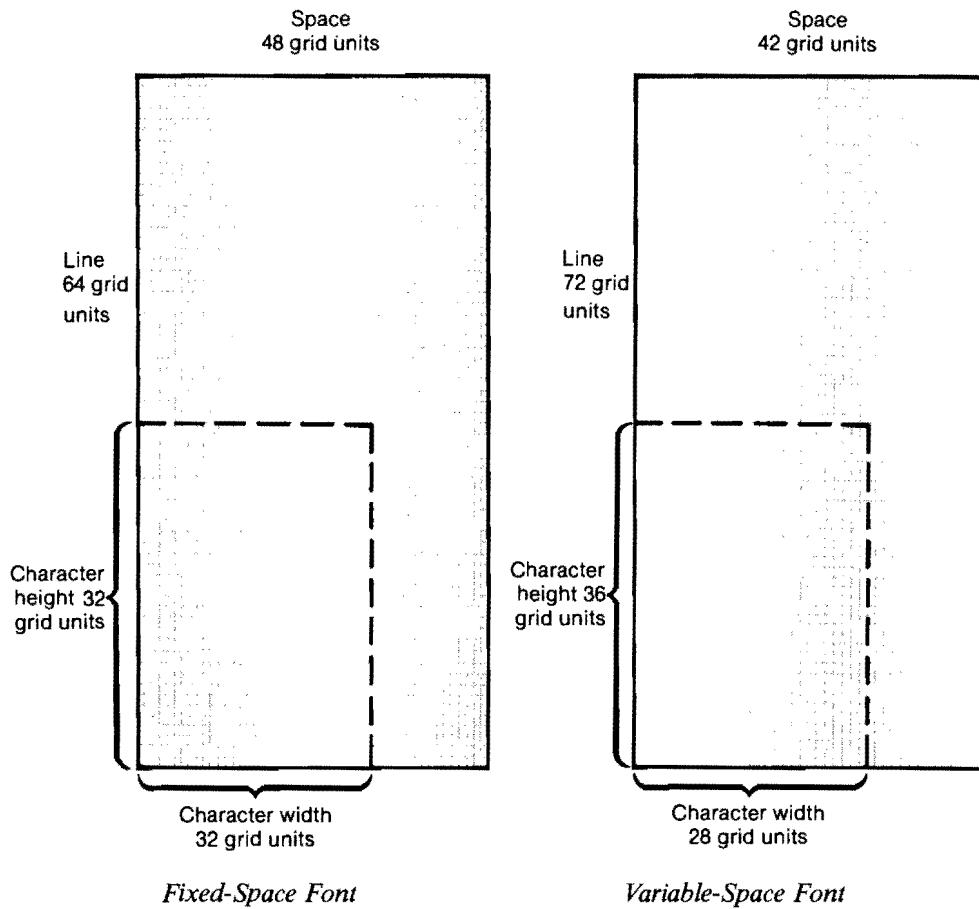
The X,Y increment pairs represent relative positions on a grid that is superimposed on the character plot cell. You can specify as many X,Y increments as you need to draw your character.

The UC instruction establishes your current pen location as the lower-left corner of the grid; this is the character plot origin (0,0). Using the pen control parameters and X,Y increments, move the pen to the relative points that define your character.

On completion of the character, the pen raises and moves one character plot cell to the right of the current character's origin, including any adjustment to spacing set by the ES (extra space) instruction. The current pen up/down position is then restored to that of the most recent PU or PD instruction.

A UC instruction with no parameters (UC;) causes the pen to return to the carriage return point.

Because the primitive grids are different for fixed- and variable-spaced fonts, user-defined characters will be drawn in different sizes and with different proportions, depending on the plotter's mode and the character font selected. However, the use of the UC instruction is similar for each primitive grid resolution that is superimposed on the CP cell. The size of the CP cell is independent of the character font selected. Its size will always be 1.5 times the width and 2 times the height of an uppercase 'A' as established by the most recent SI or SR instruction.



Confining your user-defined character to the width and height specified above ensures that your character will be the same size as characters in any character set. However, you can extend your character into the area normally reserved for the space around a character, and even into the next character plot cell. After drawing a character that extends into the next character plot cell, the pen moves one CP cell from the user-defined character's origin. That is, the next character drawn would partially overwrite the user-defined character. Use a PA, PR, or CP instruction to move the pen beyond the limits of the user-defined character before drawing the next character.

User-defined characters are drawn using the current direction (DI or DR), size (SI or SR), and slant (SL). The user-defined character is also subject to the vertical positioning component of the current LO instruction, but not to horizontal positioning component.

EXAMPLE: The following program draws a series of E's and Σ's. It demonstrates that the SI instruction has the same effect on user-defined characters as with normal characters. The CP instruction (line 70) creates spacing between the characters. It is necessary because the sigma extends into the next character plot cell.

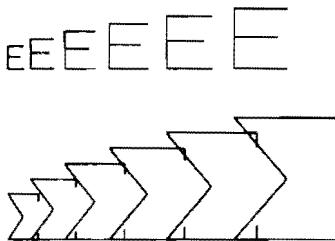
NOTE: Although line 60 is printed on two lines to fit on this page, you should send it to the plotter as one string. ■

```

10  'Insert configuration statement here
20  PRINT #1, "IN;SP1;IP1000,1000,5500,5500;"
30  PRINT #1, "SC0,10,0,10;PA1,4;"
40  FOR A=.2 TO .8 STEP .1
50  PRINT #1, "SI";A;".";A+.1
60  PRINT #1, "UC64,56,9999,0,B,-64,0,32,-32,-32,
      64,0,0,B;"
70  PRINT #1, "CP1,0;"
80  NEXT A
90  PRINT #1, "PA1,6;"
100 FOR B=.2 TO .8 STEP .1
110 PRINT #1, "SI";B;".";B+.1
120 PRINT #1, "LBE"+CHR$(3)
130 NEXT B
140 PRINT #1, "SP0;"
150 END

```

If line 70 were deleted, the plotter would draw the following.

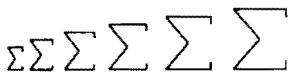


13

Notice that when the CP instruction is omitted, the symbols are partially superimposed. The (sigma) is designed in two CP cells and is twice as large as a standard uppercase character, but the pen position on completion of a UC instruction is always only one CP cell to the right of the character origin point.

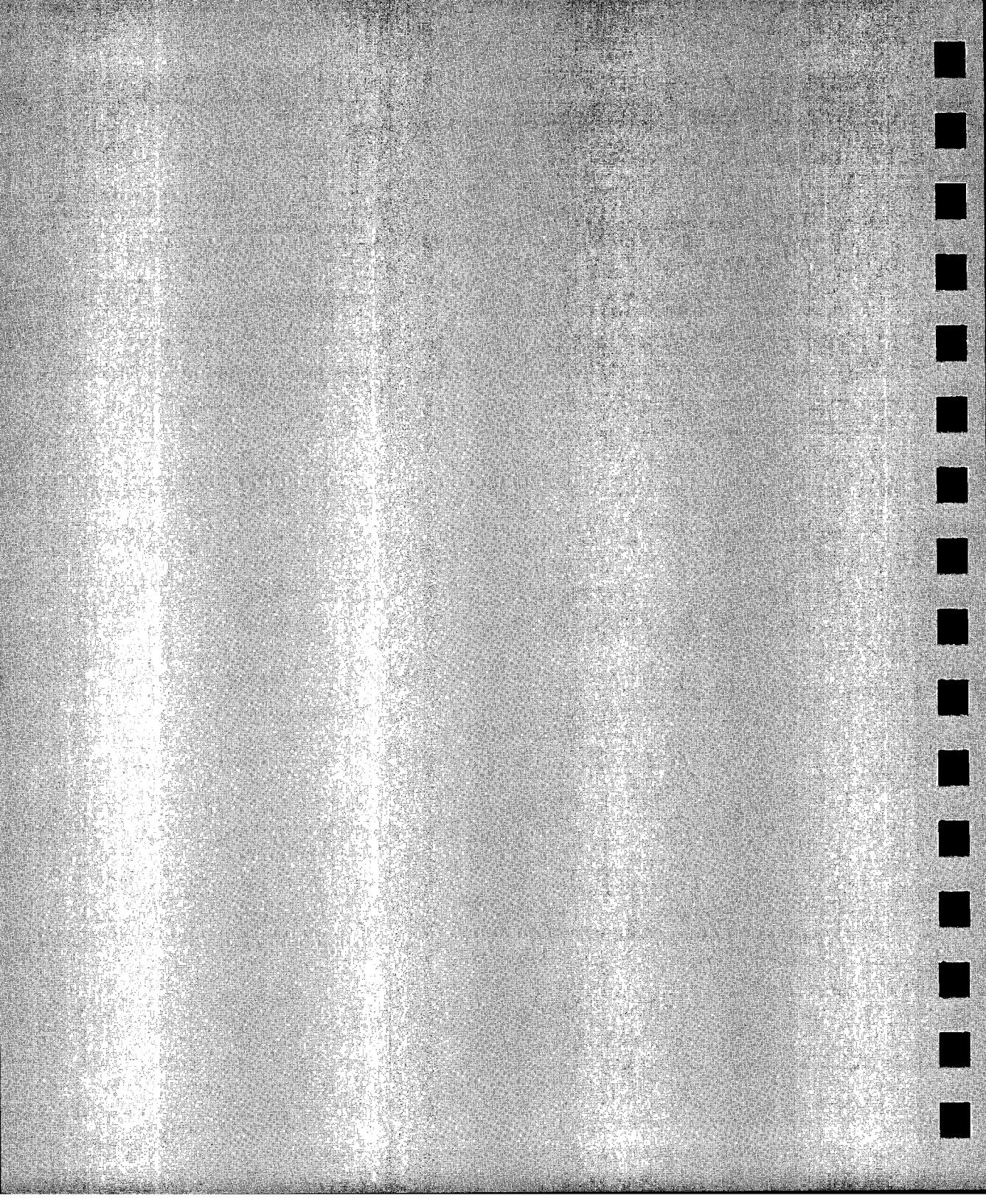
You can change the size of user-defined characters using the SI (Size Absolute) or SR (Size Relative) instructions, and by dividing or multiplying the X,Y increments of the UC instruction. Divide the X,Y increments in line 60 of the previous example by 2. Now each symbol is only half as large as before, and therefore, is now the same size as the E's.

```
60 PRINT #1, "UC32,28,9999,0,4,-32,0,16,-16,-16,-16,  
32,0,0,4;"
```



RELATED**INSTRUCTIONS:** DL, Define Downloadable Character**ERRORS:**

Condition	Error	Plotter Response
no parameter	none	moves to carriage return point
no pen down parameter	none	no character drawn
odd number of increments	2	ignores odd increment
number out of range	3	terminates instruction with out of range parameter



14

Obtaining Information from the Plotter

Output instructions send information from the plotter to your computer. This chapter discusses how to obtain plotter information concerning the following features.

- pen location and position
- plotter ID and features
- HP-GL errors

This chapter also includes a summary of output responses generated by the HP-GL output instructions, and a description of HP-IB serial and parallel polling. Refer also to chapters 9, 10, and 11 for descriptions of additional output instructions.

Instructions Covered

GC, Group Count
IM, Input Mask
OA, Output Actual Pen Status
OC, Output Commanded Pen Status
OE, Output Error
OF, Output Factors
OG, Output Group Count
OI, Output Identification
OL, Output Label Length
OO, Output Options
OS, Output Status
OT, Output Carousel Type

Notes for Obtaining Plotter Output

When the plotter receives an output instruction, it responds by making the information available in the form of an output response. If you want to retrieve this information, your computer must read the output response.

Most computers use an input statement such as ENTER, INPUT#, READ, or READLN to read the output response. When you read the output response, be sure to specify the correct number and type of variable(s) the computer will require to store the output response. For example, many computers require that a character string variable be in the form *A\$*, while numeric variables will be in the form *A*. The examples in this chapter use these conventions.

Refer to your computer documentation for the correct input statement to use and the correct format for numeric and character string variables.

When an output response includes more than one piece of information, read each piece of information whether or not you need it. This ensures that the response is cleared from the I/O buffer. Read each piece of information into a separate variable. For example, the OO (Output Options) instruction outputs eight integers; your input statement might resemble the following:

```
INPUT #1, A,B,C,D,E,F,G,H
```

Hints for HP-IB Configuration

When using an HP-IB configuration, the response to an output instruction is sent after the output instruction is completely processed *and* the I/O buffer is completely empty. For example, the following sends the OE (Output Error) instruction and then other HP-GL instructions before reading the output response.

```
"OE;PM0;CI500;PM2;FP;"
```

In this example, the plotter does not respond to the output instruction (OE) until the circle is completely filled.

If you are sending device-control instructions along with HP-GL output instructions, note that you can get an overwrite error if the outputs from both are simultaneous. Generally, you can prevent this by reading the output response for each output instruction or device-control instruction immediately, before sending any other instructions.

The plotter signals the end of its output response with output response terminator, noted in this manual by [TERM]. For HP-IB users, the output response terminator is a carriage return followed by a line feed (**CR LF**).

NOTE: Be sure the plotter's HP-IB address is not set to **LISTEN ONLY**. Otherwise, the plotter will not send an output response and your program may halt. Refer to Chapter 16 in this manual and Appendix A in your User's Guide for more information regarding HP-IB addressing. ■

Hints for the RS-232-C Configuration

The plotter outputs information according to the handshake protocol established by the ESC .P, ESC .M, and ESC .N instructions. Use these instructions to specify turnaround delays and intercharacter delays as necessary to prepare your computer to receive the output response. Your computer documentation should specify whether or not such delays are required.

The plotter signals the end of its output response with an output response terminator, noted in the manual by [TERM]. When using the RS-232-C interface, the default output response terminator is a carriage return (**CR**).

Using Output Instructions

Use the following procedures for sending output instructions.

1. Send the output instruction to the plotter as you do other HP-GL instructions. Note that none of the output instructions use parameters.
2. Read the plotter's output response immediately using a input statement appropriate to your language, keeping in mind the number and type of variable(s).

Don't send multiple output instructions and then try to read the responses sequentially. This often leads to intermittent timing problems that are dependent on what the computer and plotter are currently doing.

Identifying the Pen Location and Position

Two output instructions help you determine the current pen location and up/down position: the output actual pen status (OA) instruction and the output commanded pen status (OC) instruction.

Use the OA instruction to determine the pen's current location and position in plotter units. Use the OC instruction to determine the pen's commanded location and position in current units. (This is especially useful if you have been making relative moves and want to know your current absolute location.)

Identifying the Plotter and Its Functions

When you have more than one peripheral (such as plotters and printers) simultaneously connected to your computer, it may be necessary to have the plotter output its identification so that you know it is on-line. The output identification (OI) instruction outputs the plotter ID to the computer. The output options (OO) instruction outputs the capabilities of the plotter.

Obtaining Error Information

Two HP-GL instructions are useful for error retrieval; these are the output error (OE) instruction and the input mask (IM) instruction.

The OE instruction outputs the error number that corresponds to the first HP-GL error the plotter receives. Use this instruction to identify errors by number when debugging a program. The first parameter of the IM instruction controls what errors are noted by the plotter. Only those errors included in the error-mask (E-mask) parameter are output in response to the OE instruction. In addition, these are the only errors that display on the front-panel and set the error bit of the status byte and response to the output status (OS) instruction. (Status byte information follows this section.) Use the E-mask parameter of IM when you want the plotter to be selective about the errors it reports.

You can also control which errors generate an HP-IB service request with serial polling using the S-mask (service mask) parameter of the IM instruction. For example, if you want the errors included in the E-mask to generate a service request, you must also set the error bit of the S-mask.

The same is true to generate a positive response to a parallel poll using the P-mask (parallel poll mask). If you want the errors included in the E-mask to generate a positive response to a parallel poll, you must also set the error bit of the P-mask.

Both the S-mask and P-mask parameters of the IM instruction are valid in an HP-IB configuration only.

NOTE: The IM instruction is not an output instruction, it only defines the errors output by the OE instruction and errors which cause the error bit of the status byte (and OS response) to be set. If the HP-IB interface is active, IM also determines the plotter conditions that generate the service request message to be sent. ■

Obtaining Status Byte Information

The eight-bit status byte stores information about plotter operating conditions. You can use the OS (Output Status) instruction to learn the status of the plotter's current operating conditions. Each condition is assigned a bit number (from 0 to 7) and a corresponding decimal value. (The conditions, bit numbers, and corresponding decimal values are shown in the description for the OS instruction.)

Only the E-mask parameter has any effect on the status byte. Since the E-mask parameter determines which errors are recognized by the plotter, only those errors can set the error bit of the status byte. The S-mask determines which status conditions will generate a service request. The P-mask determines which status conditions will generate a positive response to a parallel poll. The S-mask and P-mask have no effect on the status byte.

You can obtain the value of the status byte by reading the response to the OS instruction or executing an HP-IB serial poll of the plotter.

Summary of Output Responses

The following table summarizes the output responses generated by HP-GL output instructions. Use this table when programming in languages that require you to specify the variable type and maximum number of digits to be stored as variables (as in FORTRAN).

Note in the following table that numeric ranges do not include the sign of the response. For example, if a five-digit response is a negative value, a minus sign precedes the five digits. The minus sign does not replace a digit.

Output Responses Types

Instruction	Parameters Returned*	Type and Range
OA	X,Y P	Integer, ≤ 5 digits (plotter units) Integer, 1 digit
OC	X,Y	User units: real number, ≤ 11 digits (including sign and decimal point)
	P	Plotter units: integers, ≤ 5 digits Integer, 1 digit
OD	X,Y P	Integer, ≤ 5 digits (plotter units) Integer, 1 digit
OE	Error number	Integer, 1 digit
OF	X,Y	Integer, 2 digits (plotter units/mm)
OH	X _{LL} , Y _{LL} X _{UR} , Y _{UR}	Integer, ≤ 5 digits (plotter units)
OI	Model number	5-character string
OO	Options	8 one-digit integers
OP	P1 _X , P1 _Y , P2 _X , P2 _Y	Integer, ≤ 5 digits
OS	Status	Integer, ≤ 3 digits
OT	Type Map	Integer, 1 digit Integer, ≤ 3 digits
OW	X _{LL} , Y _{LL} X _{UR} , Y _{UR}	Integer, ≤ 5 digits (plotter units)

*In addition to these parameters, the output terminator [TERM] is always sent at the end of output, and commas are sent to separate parameters.

Polling

Serial and parallel polling are the processes used by the computer to determine what device (such as a plotter or printer) on the HP-IB bus has initiated a service request. The conditions that initiate a service request are defined by the parameters of the IM instruction.

Refer to your computer documentation to determine if your system can conduct a serial and/or parallel poll. If so, identify the necessary enable/disable commands, then read the following sections describing serial and parallel polling.

Serial Polling

A serial poll enables the computer to learn the status or condition of devices on the HP-IB bus. It is commonly used to determine which device requires service. The serial poll is so named because the computer polls the devices one at a time rather than all at once. The plotter responds to a serial poll by sending the current decimal value of the status byte conditions.

NOTE: The default S-mask parameter of the IM instruction is 0, which will not send a service request. Refer to the IM instruction for setting the S-mask parameter. ■

The plotter will send a service request to the computer whenever any of the S-mask conditions are true. Each service request must be followed by a serial poll. Use an interrupt routine to halt the program when a service request is received. Then, send a serial poll to determine the device on the bus requiring service. Sending the serial poll clears bit 6 of the serial poll status byte. A service request won't be sent again until bit 6 of the serial poll status byte has been cleared (set to a logical '0'), and some S-mask condition occurs again.

A computer must issue special commands to begin and end a serial poll. During a serial poll, the plotter must be instructed to talk and the computer to listen. Therefore, you can't execute a serial poll with your plotter in listen-only mode.

Parallel Polling

The parallel poll is the fastest way to determine which device requires service, since all devices are polled at once. Each device responds with only one bit of status information, but if that information is affirmative, a serial poll can then be conducted to obtain the device's specific status.

Use the P-mask parameter of the IM instruction to specify which conditions, when true, will cause the plotter to respond positively to a parallel poll. After receiving a positive response, use the OS instruction to obtain specific status information.

NOTE: The default P-mask parameter of the IM instruction is 0, which will not respond positively to a parallel poll. Refer to the IM instruction for setting the P-mask parameter. ■

If the plotter's address is 0 through 7, refer to the following table to see which HP-IB data line will be used for responding to a parallel poll. If the plotter's address is greater than 7, your computer must tell the plotter which line to use for responding.

Address	Parallel Poll Bit		HP-IB Data Line Number
	Position	Value	
Preset	0	7	128
	1	6	64
	2	5	32
	3	4	16
	4	3	8
	5	2	4
	6	1	2
	7	0	1

GC, Group Count

USE: Assigns a number known as the ‘group count,’ which is the number that will be output by the OG instruction. The group count is an arbitrary number that you assign at the beginning of a data block in spooling applications (typically in an RS-232-C configuration with a mainframe computer). Use this instruction in conjunction with the OG instruction to monitor the successful transfer of data blocks.

SYNTAX: GC *count number*;
or
GC;

Parameter	Format	Range	Default
count number	integer	-8 388 608 to 8 388 607	0

REMARKS: A GC instruction without parameters (GC;) sets a default group count of 0. Any other parameter sets the group count to the specified number.

The GC instruction remains in effect until another GC instruction is received, or the plotter is initialized or set to default conditions.

RELATED

INSTRUCTIONS: OG, Output Group Count

ERRORS:

Condition	Error	Plotter Response
more than 1 parameter	2	uses first parameter
number out of range	3	ignores instruction

IM, Input Mask

USE: Controls which HP-GL errors are reported. If you are using an HP-IB interface, you can also use IM to control the conditions that cause an HP-IB service request or a positive response to a parallel poll.

SYNTAX: IM *E-mask value* (, *S-mask value* (, *P-mask value*));
or
IM ;

Parameter	Format	Range	Default
E-mask value	integer	0 to 255	223
S-mask value	integer	0 to 255	0
P-mask value	integer	0 to 255	0

REMARKS: The parameters of the IM instruction determine the HP-GL errors the plotter will recognize, the status byte conditions that generate a service request, and the status byte conditions that generate a positive response to a parallel poll, respectively.

If you are using an RS-232-C interface, only the E-mask parameter is valid; the plotter will ignore the S- and P-mask parameters, if present. If you are using an HP-IB interface, you can use all three parameters.

- **E-mask** — Allows the reporting of HP-GL errors. Without the E-mask value, no errors are reported. Look up the error(s) you want to report in the following table. Then use the decimal value of the associated bit as the E-mask parameter value. To report more than one error, add the decimal values of those errors and use the sum as the parameter. For example, to report errors 3 and 5, you would use the sum of their bit decimal values ($4 + 16 = 20$) as the E-mask parameter (i.e., *IM20*.).

The default value of 233 causes the plotter to recognize errors 1, 2, 3, 5, and 7. By default, position overflow is not reported.

When an HP-GL error occurs, the error number is compared with the E-mask bits specified. If the two match, the error bit (bit 5) of the status byte is set, and the error message will display on the front panel. Use the output error instruction, OE (described later in this chapter) to read the exact error number to your computer.

E-Mask Bits

Error No.	Meaning	Bit No.	Decimal Value
1	Instruction not recognized	0	1
2	Wrong number of parameters	1	2
3	Parameter out of range	2	4
4	Not used	3	8
5	Unknown character set	4	16
6	Position overflow	5	32
7	Polygon buffer overflow	6	64
8	Not used	7	128

- **S-mask** — Allows the plotter to send an HP-IB service request to the computer. Look up the conditions in the following table that you want to generate a request. Then use the decimal value of the associated status bit as the S-mask parameter value. To specify more than one condition, add the decimal values of those conditions and use the sum as the parameter. For example, to have pen down and digitized point available generate a service request, you would use the sum of their bit decimal values ($1 + 4 = 5$) as the S-mask parameter (i.e., *IM20, 5;*).

When one of the conditions occurs, a bit of the status byte changes value. The status byte is then compared bit by bit with the S-mask to determine if the service request is to be sent, and if bit 6 of the status byte is to be set. You must do a serial poll to clear bit 6 of the status byte. It is not set again until one of the conditions specified by the S-mask value occurs.

NOTE: The S-mask parameter is valid only in an HP-IB configuration. ■

S-Mask Bits

Status Bit No.	Meaning	Decimal Value
0	Pen down	1
1	P1 or P2 changed	2
2	Digitized point available	4
3	Initialized	8
4	Ready for data (buffer empty)	16
5	Error bit of status byte set*	32
6	Not used	64
7	Not used	128

*The E-mask controls which errors set the error bit of the status byte.

- **P-mask** — Specifies which conditions will generate a positive response to an HP-IB parallel poll. Look up the conditions in the following table. Then use the decimal value of the associated status bit as the P-mask parameter value. To specify more than one condition, add the decimal values of those errors and use the sum as the

parameter. For example, to have pen down and ready for data generate a positive response to a parallel poll, use the sum ($1 + 16 = 17$) as the parameter (i.e., $IM20, 5, 17,$).

Note that specifying a P-mask does not cause either a service request message to be sent or the service request bit (bit 6) of the status byte to be set. These occur based on the conditions set by the S-mask value.

P-Mask Bits

Status Bit No.	Meaning	Decimal Value
0	Pen down	1
1	P1 or P2 changed	2
2	Digitized point available	4
3	Initialized	8
4	Ready for data (buffer empty)	16
5	Error bit of status byte set*	32
6	Not used	64
7	Not used	128

*The E-mask controls which errors set the error bit of the status byte.

RELATED

INSTRUCTIONS: OE, Output Error

OS, Output Status

Serial and Parallel Polling

ERRORS:

Condition	Error	Plotter Response
more than 3 parameters	2	uses first 3 parameters
number out of range	3	ignores instruction

OA, Output Actual Pen Status

USE: Outputs the current pen location (in plotter units) and up/down position. You can use this information to position a label or figure, to determine the parameters of some desired window, or to determine the pen's current location if it has been moved using front-panel **Cursor Control** buttons.

NOTE: When in digitize mode, use the OD (Output Digitized Point) to determine the above information. Refer to Chapter 11. ■

SYNTAX: OA;

Parameter	Response	Format	Range
none	X,Y Pen status	integer	current hard-clip limits

REMARKS: The pen location and position are output to the computer as integers, separated by commas. The X,Y coordinates are always output in plotter units, regardless of whether scaling is on or off. The plotter outputs a minus sign (−) for negative numbers; positive signs (+) are suppressed. The pen position is either 0 (up) or 1 (down).

EXAMPLE: The following outputs the X,Y coordinates and the pen position, which are read by the computer and printed on the screen. Note that your computer system may require different BASIC statements than those presented here (refer to Chapter 1, *Programming Concepts* for more information).

```

10  'Insert configuration statement here
20  PRINT #1, "IN;OA;"
30  INPUT #1, X,Y,P
40  PRINT X,Y,P
50  END

```

If your current pen location was (10 515, −7576) (in plotter units) and the pen was in the down position, the plotter would send the following information to the computer.

10 515 -7576 1 [TERM]

NOTE: The word [TERM] indicates only that the plotter sends the output terminator; '[TERM]' will not display on your terminal. ■

RELATED

INSTRUCTIONS: OC, Output Commanded Pen Status

ERRORS:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

OC, Output Commanded Pen Status

USE: Outputs the location and up/down position of the last commanded pen move. Use this instruction to position a label or determine the parameters of an instruction that tried to move the pen beyond the limits of some window. You can also use this instruction when you want to know the pen's location in user units.

SYNTAX: OC;

Parameter	Response	Format	Range
none	X,Y Pen status	current units	-8 388 608 to 8 388 607

REMARKS: When scaling is off, the X,Y coordinates are in plotter units; integers. When scaling is on, coordinates are interpreted as user units. The plotter outputs a minus sign (-) for negative numbers; positive signs (+) are suppressed. The pen position is either 0 (up) or 1 (down).

Note that any pen motion caused by the front-panel **Cursor Control** buttons causes OC to output the current pen location rather than the last commanded location.

EXAMPLE: The OC instruction is similar to the output actual status (OA) instruction. Refer to the OA instruction example, changing the OA to OC.

RELATED

INSTRUCTIONS: OA, Output Actual Status

ERRORS:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

OE, Output Error

USE: Outputs a number corresponding to the type of HP-GL error (if any) received by the plotter after the most recent IN instruction, front-panel reset, or OE instruction. Use this instruction for debugging programs.

SYNTAX: OE;

Parameter	Response	Format	Range
none	error number	integer	0 to 7

REMARKS: The OE instruction outputs an integer (within the range 0 to 7) corresponding to first HP-GL error (if any) that occurred. The plotter outputs only the first error. If you suspect more than 1 error, place the instruction in as many locations in your program as necessary. The following table defines the error numbers.

HP-GL Error Number	Meaning
0	No error
1	Instruction not recognized
2	Wrong number of parameters
3	Out-of-range parameter, or illegal character
4	Not used
5	Unknown character set
6	Position overflow (not reported with default E-mask value)
7	Buffer overflow for polygons

Executing an OE instruction clears bit position 5 of the status byte (if previously set) and the error message from the front-panel display (unless there is an I/O error which has not been cleared by an ESC .E instruction).

EXAMPLE: In the following program, line 20 contains two errors.

```
10 'Insert configuration statement here  
20 PRINT #1, "IN;SP1;PA1000,1000,20;ED;DE;"  
30 INPUT A  
40 PRINT A  
50 END  
  
20 PRINT #1, "IN;SP1;PA1000,1000,20;ED;DE;"
```

first error (wrong number of parameters) _____

second error (instruction not recognized) _____

Plotter response: 2 [TERM]

By referring to the table, we know that error 2 indicates the wrong number of parameters. Once the first error is corrected, run the program again to find the other HP-GL error.

Note that your computer system may require different BASIC statements than those presented here (refer to Chapter 1, *Programming Concepts* for more information).

ERRORS:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

OF, Output Factors

USE: Outputs the number of plotter units per millimetre in each axis. This instruction lets you use the plotter with software that needs to know the size of a plotter unit.

SYNTAX: OF;

Parameter	Response	Format	Range
none	40,40	integer	none

REMARKS: The plotter response indicates there are 40 plotter units per millimetre each in the X- and Y-axes (0.025 mm per plotter unit).

ERRORS:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

14 OG, Output Group Count

USE: Outputs the data block number of the current group count and whether the escape function has been activated. Use this instruction at the end of a data block in spooling applications, where it is important to know the current data block number and whether the data block has been transferred.

SYNTAX: OG;

Parameter	Response	Format	Range
none	count number, escape status	integer	none

REMARKS: After you send the OG instruction, your program should read the plotter's output response. Refer to *Hints for Obtaining Plotter Output* earlier in this chapter.

- **Number** — Is the number defined in the most recent GC instruction (or 0 is no number is defined).
- **Escape Status** — Provides information regarding whether the escape function has been activated. For the function to be activated, a function key redefined by the KY instruction to perform the escape function must be pressed.
 - 0 The escape function has not been activated since the last OG or IN instruction.
 - 1 The escape function has been activated.

When the escape function has been activated, the OG instruction causes the plotter to enter flush mode. In flush mode, the I/O buffer is cleared and all incoming graphics instructions are ignored by the plotter. The plotter remains in flush mode until it receives an ESC.U instruction.

RELATED

INSTRUCTIONS: ESC.U, End Flush Mode
GC, Group Count
KY, Define Key

OI, Output Identification

USE: Outputs the plotter's identifying model number. This information is useful in a remote operating configuration (where several plotters are connected to the computer) to determine which plotter model is on-line, or when software needs the plotter's model number.

SYNTAX: OI;

Parameter	Response	Format	Range
none	7595A or 7596A*	character string	none

*If Emulate is ON, the response will be 7585B or 7586B.

REMARKS: The plotter always outputs its model number and letter as a 5-character string.

EXAMPLE: The following program outputs the plotter identification.

```

10  "Insert configuration statement here
20  PRINT #1, "IN;OI;"
30  INPUT #1, A$
40  PRINT A$
50  END

```

Note that your computer system may require different BASIC statements than those presented here (refer to Chapter 1, *Programming Concepts* for more information).

ERRORS:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

OL, Output Label Length

USE: Output information about the label contained in the label buffer.

SYNTAX: OL;

Parameter	Response	Format
none	length characters line feeds	real integer integer

REMARKS: If the buffer is empty, three zeros are output. Otherwise the information is output as follows:

- **Length —** Contains the length of the longest line in the buffered label in character plot cell spaces.
- **Characters —** Contains the number of printing characters and spaces in the longest line of the buffered label. A backspace counts as -1; a character with automatic backspace (e.g., some accent characters) counts as zero.
- **Line Feeds —** Contains the net number of line feeds that will occur when the buffered label is drawn. An inverse line feed (VT, CHR\$(11)) counts as -1, and a line feed (LF, CHR\$(10)) counts as +1. If the buffered label contains the same number of both types of line feeds, zero is output.

The OL instruction is normally used with the BL instruction, but it outputs information on the current contents of the buffer whether it was filled with the LB or BL instruction.

EXAMPLE: This example uses OL to determine the correct length of a label so that the entire label can be underlined.

```
10 'Insert configuration statement here
20 PRINT #1, "IN;SP1;PA5000,3000;""
30 PRINT #1, "BLUnderline this label"+CHR$(13)+CHR$(3)
40 PRINT #1, "OL;""
50 PRINT #1, A, B, C
60 PRINT #1, "PB;""
70 PRINT #1, "CP0,-.25;PD:CP";A;"",0;""
80 PRINT#1, "SP0;""
90 END
```

Underline this label

RELATED:**INSTRUCTIONS:** BL, Buffer Label

LB, Label

PB, Plot Buffered Label

ERRORS:

Condition	Error	Plotter Response
1 or more parameters	2	executes instruction

OO, Output Options

USE: Outputs eight option parameters indicating the features implemented on the plotter. Some software packages use this feature to determine which plotter capabilities exist.

SYNTAX: OO;

Parameter	Response	Format	Range
none	c,1,0,0,1,1,0,1	integer	0 or 1

REMARKS: The plotter outputs eight ASCII integers as follows, all separated by commas.

C , 1 , 0 , 0 , 1 , 1 , 0 , 1 [TERM]

C	1	0	0	1	1	0	1	[TERM]
Indicates plotter has configurable buffers	Indicates plotter has polygon fill capability	Indicates plotter has arc and circle instructions	Indicates plotter has pen select capability	Indicates status of plotter paper check bit				

- C Takes on a value of 0 to 3.
- 0 Sheet media is loaded and unmarked.
- 1 Roll media is loaded and unmarked.
- 2 Sheet media is loaded and plotted on.
- 3 Roll media is loaded and plotted on.

The value is set to zero or one when you load new paper. It can not be cleared by the IN or DF instructions, or by a front-panel **Reset**.

EXAMPLE: The following outputs the options of your plotter as described above.

```

10  'Insert configuration statement here
20  PRINT #1, "IN;00;"
30  INPUT #1, A,B,C,D,E,F,G,H
40  PRINT A,B,C,D,E,F,G,H
50  END

```

Note that your computer system may require different BASIC statements than those presented here (refer to Chapter 1, *Programming Concepts* for more information).

ERRORS:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

OS, Output Status

USE: Outputs the decimal value of the status byte. Use this instruction in debugging operations and in digitizing applications.

SYNTAX: OS;

Parameter	Response	Format	Range
none	status number	integer	0 to 255

REMARKS: On execution of the OS instruction, the internal 8-bit status byte is converted to an ASCII integer between 0 and 255 and output to your computer.

Instruct your computer to read the output response then refer to the table below and find the **largest decimal value** that can be subtracted from the output response. The condition corresponding to the decimal value has been met.

Continue subtracting the largest possible decimal value from the remainder of the output response. Each time you subtract a decimal value, the corresponding condition has been met. Continue this process until the remainder is zero.

Decimal Value	Meaning	Bit No.
1	Pen down	0
2	P1 or P2 newly established; cleared by OP	1
4	Digitized point available; cleared by OD	2
8	Initialized; cleared by OS	3
16	Ready for data (buffer empty)	4
32	Error, cleared by OE	5
64	Request service (always 0 for OS; or 1 for HP-IB serial poll)	6
128	Not used (always set to "0")	7

On power-up, the status byte is 26, the sum of 16 (ready for data), 8 (Initialized), and 2 (P1/P2 newly established). On execution of OS, bit position 3 is cleared and the status byte is 18.

EXAMPLE: The following outputs the numeric representation of the status byte. Note that your computer system may require different BASIC statements than those presented here (refer to Chapter 1, *Programming Concepts* for more information).

```

10  'Insert configuration statement here
20  PRINT #1, "IN;OS;" 
30  INPUT #1, S
40  PRINT S
50  END

```

RELATED

INSTRUCTIONS: IM, Input Mask
 OD, Output Digitized Point
 OE, Output Error
 OP, Output P1 and P2
 PD, Pen Down

ERRORS:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

OT, Output Carousel Type

USE: The OT instruction outputs information on the type of carousel loaded and the stalls occupied.

SYNTAX: OT;

Parameter	Response	Format	Range
none	type map	integer integer	-1 to 5 0 to 255

REMARKS: The current carousel type and its pen map are output as follows.

- **Type** — Defines whether a carousel is installed and, if so, which type.
 - 1 Carousel of unknown type is installed.
 - 0 Carousel is not installed.
 - 1 Carousel for paper fiber-tip pens is installed.
 - 2 Carousel for roller-ball pens is installed.
 - 3 Carousel for refillable drafting pens is installed.
 - 4 Carousel for transparency fiber-tip pens is installed.
 - 5 Carousel for disposable drafting pens is installed.
- **Map** — Defined as the sum (0 through 255) of any combination of the decimal values shown below. For example, a map value of 15 (1+2+4+8) indicates pens are installed only in stalls 1, 2, 3, and 4 of the carousel. A map value of 0 means either all stalls are empty, or no carousel is installed.
 - 1 Stall 1 is occupied.
 - 2 Stall 2 is occupied.
 - 4 Stall 3 is occupied.
 - 8 Stall 4 is occupied.
 - 16 Stall 5 is occupied.

32 Stall 6 is occupied.

64 Stall 7 is occupied.

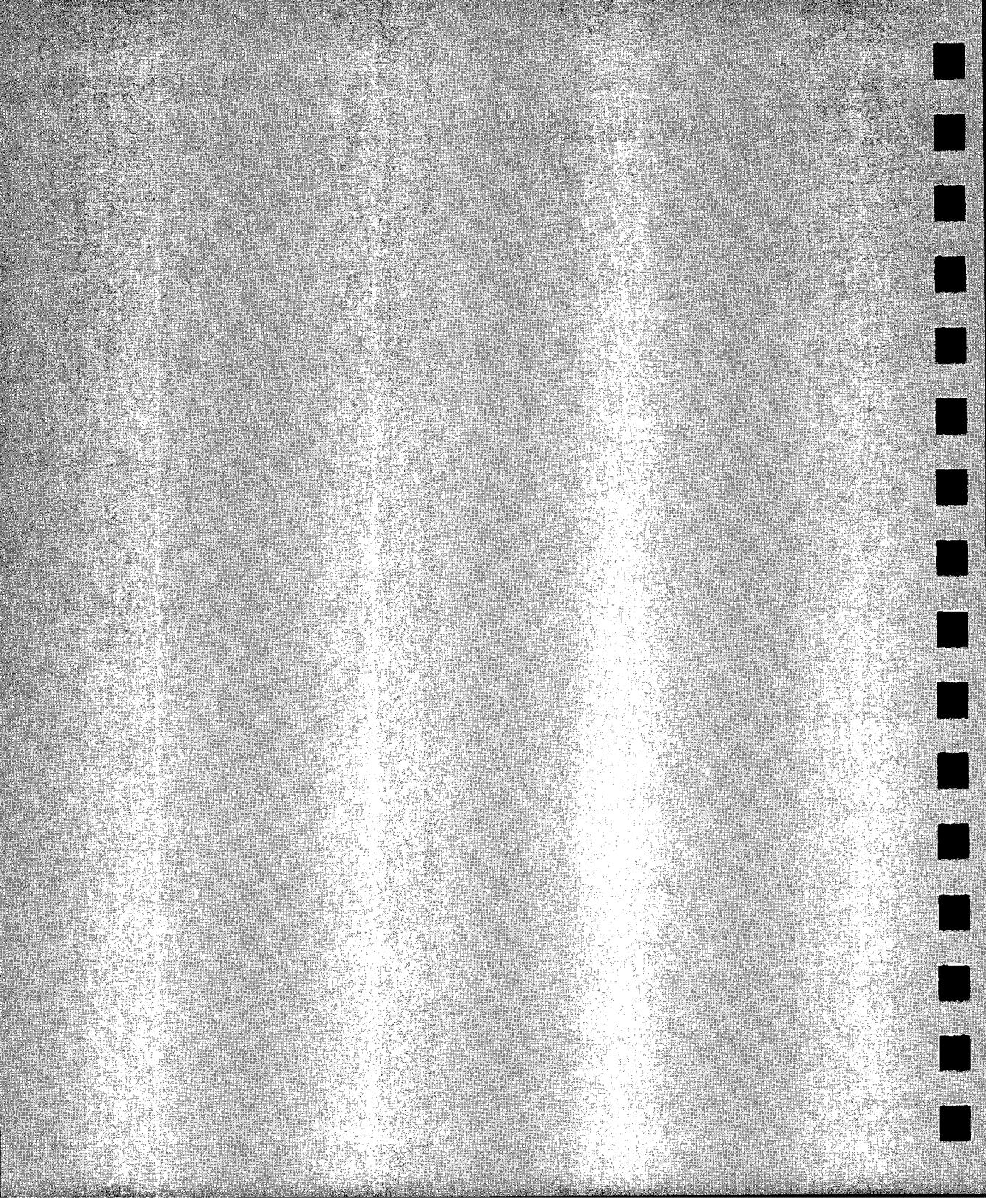
128 Stall 8 is occupied.

ERRORS:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

Notes

14



Device-Control Instructions

Device-control instructions differ from HP-GL (graphics) instructions in that they control internal plotter functions such as the configurable graphics memory, input/output, and data flow. This chapter discusses the majority of device-control instructions implemented by the plotter, including syntax and conventions. All device-control instructions discussed in this chapter are valid for RS-232-C and HP-IB configurations except where noted. Chapter 16 discusses the device-control instructions used in establishing interfacing and handshaking conditions.

Instructions Covered

- ESC . A, Output Identification
- ESC . B, Output Buffer Space
- ESC . E, Output Extended Error
- ESC . J, Abort Device Control
- ESC . K, Abort Graphics
- ESC . L, Output Buffer Size When Empty
- ESC . O, Output Extended Status
- ESC . Q, Set Monitor Mode
- ESC . R, Reset
- ESC . S, Output Configurable Memory Size
- ESC . T, Allocate Configurable Memory
- ESC . U, End Flush Mode
- ESC . Y or ESC .(, Plotter On
- ESC . Z or ESC .), Plotter Off
- ESC .@, Set Plotter Configuration

Understanding Device-Control Instructions

Device-control instructions serve three basic purposes: to control memory allocation (buffer sizes), provide information about certain internal plotter conditions, and to control data transfer between the computer and plotter. Device-control instructions differ from graphics instructions in the following ways.

- Device-control instructions are not represented by mnemonics; they consist of the single ASCII character **ESC** (decimal code 27) followed by a period (.) and a character that represents the instruction's unique function.
- Device-control instructions follow specific syntax conventions that are different from the graphics instructions.
- Device-control instructions don't enter the plotter's buffer but are processed immediately. (Graphics instructions enter the buffer and are processed in the order they are received.)

Sending Device-Control Instructions

The principles for sending device-control instructions to the plotter are the same as for graphics instructions; send them as a literal string in an output statement such as PRINT or OUTPUT.

Send the **ESC** character the same way you send other ASCII characters; either by using a character string function such as CHR\$, or by producing the character directly from the keyboard. Send the period and the final character as a literal string. You may need to link the CHR\$ function with the literal string using one of the following symbols: +, &, or ;. Use the symbol your system or language requires.

The examples in this chapter and Chapter 16 use the CHR\$ function with the concatenation symbol +. The following shows the ESC.L instruction.

```
PRINT #1, CHR$(27)+"."L"
```

Syntax for Device-Control Instructions

A complete device-control instruction includes the three character sequence consisting of **ESC** and . followed by the character or uppercase letter indicating its function (e.g., @, A, or Z). Some instructions also include parameters and a terminator according to the following syntax conventions.

- ESC** — the single ASCII character, escape (decimal code 27) is shown in bold, condensed print.
- () — Parameters in parentheses are optional.
- ;
— Separates parameters. A semicolon without a parameter sets the parameter to its default value.
- :
— Terminates any instruction that uses parameters, whether or not the parameters are used. Instructions that do not have parameters need no terminator.
- [TERM] — The terminator sent back to your computer by the plotter at the end of the response to an output instruction. The default output terminators are: RS-232-C—a carriage return (**CR**) and HP-IB—carriage return and line feed (**CR LF**). You can change the default terminator using the device-control instruction ESC.M (set output mode instruction).

Omitting Optional Parameters

Unlike the graphics instructions, you can set a parameter of a device control instruction to its default value by entering only the semicolon. Thus, you *can* send an instruction such as (ESC. T;;2000;;4000;15504.). The omitted parameters are set to their default values.

Using Device-Control Instructions

All device-control instructions with the word 'output' in their title cause the plotter to send information (called an output response) back to your computer. Immediately after sending a device-control instruction, have your program read the output response into a variable. Refer to *Notes for Obtaining Plotter Output* in Chapter 14.

Setting the Plotter to Known Conditions

You should already be acquainted with the following methods for establishing known plotter conditions.

- turning the power switch off then on
- using the front-panel **Reset**
- sending the DF or IN instructions to your plotter

The first two methods require your presence at the plotter. The last method is not immediate; the DF and IN instructions are processed in the I/O buffer with other graphics instructions in the order they are received. The following three device-control instructions can establish specific default conditions and/or immediately clear the buffers of any HP-GL instructions.

- ESC . J, Abort Device-Control, aborts any device-control instruction partially decoded or executed.
- ESC . K, Abort Graphics, aborts any partially sent graphics instruction and clears the buffers of all remaining graphics instructions.
- ESC . R, Reset, resets certain I/O conditions to power-up default states, including clearing the buffers and setting them to their default size allocations. (In an RS-232-C interface, this instruction sets the handshake protocol to default conditions.)

Note that you can also use the ESC . M (Output Mode) instruction (in an RS-232-C interface) to change the output terminator from its default value (carriage return). This instruction, however, is primarily used in establishing certain handshaking conditions and is discussed in detail in Chapter 16.

Identifying the Plotter

When using the plotter in a remote location (in an Eavesdrop configuration), you may need to confirm that the plotter is connected to your computer, and that it is turned on. Use the ESC . A instruction to immediately output the plotter's model number and firmware revision level.



Monitoring and Changing the Buffer Sizes

You can use the following device-control instructions to monitor the sizes of the plotter's buffers.

- ESC . B, Output Buffer Space, outputs the number of bytes currently available in the logical I/O buffer.
- ESC . L, Output Buffer Size When Empty, outputs the size of the logical I/O buffer when empty.
- ESC . S, Output Configurable Memory Size, outputs the size of any individual or all buffers (depending on parameter).

You can use the following device-control instructions to change the size of the plotter's buffers.

- ESC . T, Allocate Configurable Memory, changes the size of the physical I/O buffer, the polygon buffer, downloadable character buffer, vector buffer, and the pen sort buffer.
- ESC . @, Set Plotter Configuration, changes the size of the logical I/O buffer.

Verifying Errors or Operating Conditions

The following device-control instructions are useful in debugging operations.

- ESC . E, Output Extended Error, outputs any I/O error related to device-control instructions, and is used in RS-232-C block I/O error checking.
- ESC . O, Output Extended Status, outputs information about the plotter's current operating status.
- ESC . Q, Set Monitor Mode, sets either parse or receive monitor mode so you can view program instructions on your screen while the plotter is drawing.

- ESC.Y or ESC.(, Programmed-On, causes the plotter to react to information from the computer in an Eavesdrop environment.
- ESC.Z or ESC.), Programmed-Off, (in Eavesdrop only) disables the plotter so that it does not react to any information until returned to the programmed-on state.

ESC . A, Output Identification

USE: Outputs the plotter's model number and firmware revision level.

SYNTAX: **ESC . A**

Parameter	Response	Format	Range
none	model number, firmware revision level*	character string integer	7595A or 7596A* 1 to 32767

*If Emulate is on, the response will be 7585B or 7586B and no firmware revision level will be returned.

RELATED

INSTRUCTIONS: OI, Output Identification

ESC . B, Output Buffer Space

USE: Outputs the plotter's currently available logical I/O buffer space.

SYNTAX: **ESC . B**

Parameter	Response	Format	Range
none	available logical I/O buffer space	integer	2 to 25 518

REMARKS: The plotter outputs the number of unused bytes in the logical I/O buffer. If your program continually interrogates the plotter until the response indicates a specific amount of available space, use a pause routine to allow the plotter to execute other instructions; this increases the amount of available space.

RELATED

INSTRUCTIONS: ESC . L, Output Buffer Size When Empty

ESC.E, Output Extended Error

USE: Outputs the error number for any I/O error related to device-control instructions and clears the error message from the front-panel display. Use this instruction when debugging a program to determine which errors have occurred (if any). Additionally, if you are using the RS-232-C interface, you can use ESC.E with ESC.@ to do block I/O error checking.

SYNTAX: **ESC.E**

Parameter	Response	Format	Range
none	error number	integer	0, 10 to 18

REMARKS: The plotter's response is one of the following error numbers. For RS-232-C users, all numbers listed are valid. For HP-IB users, the valid error numbers are 0 and 11 through 14.

Error No.	Meaning
0	No I/O error has occurred.
10	Output instruction received while another output instruction is executing. The original instruction will continue normally; the second instruction received will be ignored.
11	Invalid byte received after first two characters, ESC. , in a device-control instruction.
12	Invalid byte received while parsing a device-control instruction. The parameter containing the invalid byte and all following parameters are defaulted.
13	Parameter out of range.
14	Too many parameters received. Additional parameters beyond the proper number are ignored; parsing of the instruction ends when a colon (normal exit) or the first byte of another instruction is received (abnormal exit).
15	A framing error, parity error, or overrun error has been detected.
16	The input buffer has overflowed. As a result, one or more bytes of data have been lost and therefore, an HP-GL error will probably occur.
17	Baud rate mismatch or, full-duplex communication is selected and conditions for data transmission are not met, i.e., cabling is configured for three-wire communications.
18	I/O error of indeterminate cause.

RS-232-C Users Only: In addition to checking I/O errors, you can use the ESC.E instruction to start and end a data block for block I/O checking. If block I/O error checking has been activated (bit 4 of the second parameter of the ESC.@ instruction is set to 1), the ESC.E instruction will terminate a block.

If there are no I/O errors (response to ESC.E is 0), the plotter executes the data normally. If response to ESC.E indicates an I/O error, the plotter discards the entire data block. You can then retransmit the entire block of data and prevent errors in the plot.

RELATED

INSTRUCTIONS: OE, Output Error
ESC.@", Set Plotter Configuration

ESC . J, Abort Device-Control

USE: Aborts any device-control instruction that may be partially decoded or executed. Use this instruction in an initialization sequence when you first access the plotter.

SYNTAX: **ESC . J**

REMARKS: Unspecified parameters of partial instructions are defaulted. All pending or partially transmitted output requests, from either HP-GL or device-control instructions, are immediately terminated, including handshake outputs. Intermediate output operations, such as turnaround delay and echo suppression, are aborted, and buffer input is enabled. Only the specified execution of an output operation is aborted. The handshake and output mode parameters remain as specified.

RELATED

INSTRUCTIONS: ESC . K, Abort Graphics

ESC.K, Abort Graphics

USE: Aborts any partially decoded HP-GL instruction and discards remaining instructions in the I/O, pen sort, bidirectional, and vector buffers. Use this instruction as part of an initialization sequence when starting a new program or to terminate plotting of HP-GL instructions in the buffer.

SYNTAX: **ESC.K**

REMARKS: This instruction allows the instruction or vector being executed to finish, aborts any partially decoded instruction, and clears the I/O, pen sort, bidirectional, and vector buffers of all remaining graphic instructions. The plotter finishes executing its current vector, but all remaining vectors are aborted. (This is particularly evident when the plotter is in the process of completing a label, arc, circle, polygon, or line type instruction that contains multiple vector moves, or if a VS instruction specified a slow velocity.) Any data entered since block I/O error checking was enabled (with ESC.@@) is aborted.

RELATED

INSTRUCTIONS: ESC.J, Abort Device Control

ESC . L, Output Buffer Size When Empty

USE: Outputs the size (in bytes) of the logical I/O buffer. The response is not transmitted by the plotter until the buffer is empty.

SYNTAX: **ESC . L**

Parameter	Response	Format	Range
none	available logical I/O buffer space	integer	2 to 25518 bytes

REMARKS: The plotter outputs an ASCII integer equal to the number of bytes allocated to the logical I/O buffer. The response is not transmitted by the plotter until the buffer is empty.

NOTE: If your plotter is in block mode, using ESC . L could disrupt communication. If you use ESC . L while in block mode, send it immediately after reading the ESC . E response. ■

RELATED

INSTRUCTIONS: **ESC . B, Output Buffer Space**

ESC.O, Output Extended Status

USE: Outputs the plotter's extended status. Use this instruction to obtain information about the current operating status of the plotter.

SYNTAX: **ESC.O**

Parameter	Response	Format	Range
none	operating status	integer	0 to 4095

REMARKS: When used with an RS-232-C interface, the instruction is subject to any turnaround or intercharacter delays specified by ESC.M and ESC.N.

The operating status is the decimal equivalent of a 16-bit extended status word. The status word bits are defined in the following table.

Bit Position	Logic State	Decimal Value	Meaning
0	0	0	Single sheet loaded.
	1	1	Roll paper loaded.
1	0	0	'Clean' paper loaded. (Reset after paper is sensed.)
	1	2	Current page is not clean. Set after executing a pen down or at power-up when no paper is loaded, setting bit 5.
2	0	0	No paper advance (no AF, AH, FR, or PG instruction executed) since last ESC.O instruction. The ESC.O instruction resets this bit to 0.
	1	4	Paper advance instruction (AF, AH, FR, or PG) executed since last ESC.O instruction.
2	0	0	Not used.
3	0	0	I/O and pen sort buffer not empty.
	1	8	I/O and pen sort buffers are empty and ready for data.
4	0	0	Ready state. Processing HP-GL instructions.
5	0	0	
4	1	16	View state. Paper loaded but graphics suspended.
5	0	0	
4	0	0	Not ready. Paper not loaded, graphics suspended.
5	1	32	
6	0	0	Cover is lowered.
	1	64	Cover is open.
7	0	0	Emulate mode is OFF.
	1	128	Emulate mode is ON.
8	0	0	Expand mode is OFF.
	1	256	Expand mode is ON.
9	0	0	ESC.O, ESC.U, WD, or OK instruction has been executed. Execution of each of these instructions resets this bit to 0.
	1	512	Function key has been pressed while plotter is in keyboard mode.
10	0	0	Servo is functioning.
	1	1024	Servo is not functioning.
11	0	0	Invert Plot is ON.
	1	2048	Invert Plot is OFF.
12-15	*	*	Not used.

*Undefined.

Possible outputs and their meanings are described in the following table.

	Decimal Value Output to the Computer											
	0	2	8	10	16	18	24	26	32	34	40	42
I/O and/or pen sort buffer is not empty.	X	X			X	X			X	X		
I/O and pen sort buffers are empty.			X	X			X	X			X	X
Ready state. Processing HP-GL Instructions.	X	X	X	X								
View state. Paper is loaded. Graphics suspended.					X	X	X	X				
Not ready state. Paper not loaded. Graphics suspended.									X	X	X	X
Paper marked on.		X		X		X		X		X		X

RELATED

INSTRUCTIONS: OS, Output Status

ESC.Q, Set Monitor Mode

USE: Enables or disables either monitor mode 1 (parse) or monitor mode 2 (receive). Use this instruction as a debugging aid in program development. This instruction is valid only when you have set **Monitor Mode: ON** from the front panel (refer to the User's Guide for more information).

SYNTAX: **ESC.Q n:**

Parameter	Format	Range	Default
n	integer	0, 1, or 2	0

REMARKS: The plotter interprets the parameter as follows.

- **0 or no parameter** — Disables selected monitor mode without changing selection.
- **1** — Activates monitor mode 1, parse mode. Monitor mode 1 causes HP-GL instructions to be retransmitted to the terminal as they are parsed from the plotter's buffer. The terminal displays whatever the plotter is currently doing. The plotter sends HP-GL responses to both the terminal and the computer if **Duplex** is **HALF**.
- **2** — Activates monitor mode 2, receive mode. Monitor mode 2 causes both HP-GL and device-control instructions to be retransmitted to the terminal as they are received by the plotter. The terminal displays what the plotter will be doing later when the received data is parsed.

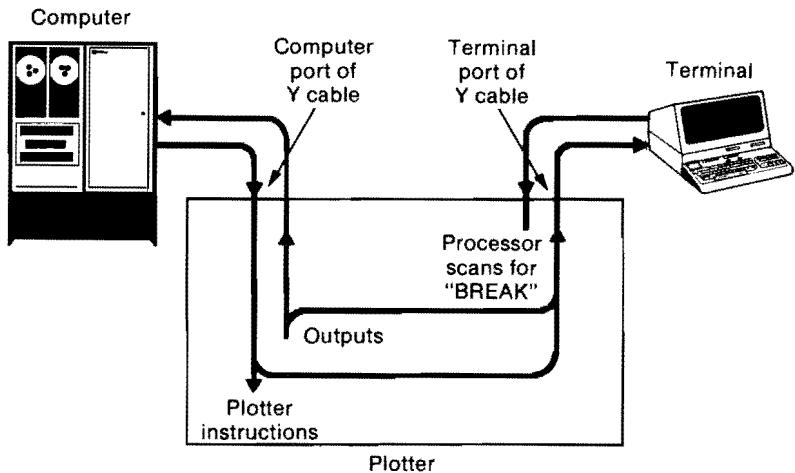
At power-on in Eavesdrop mode, the plotter is in the programmed-off state. Monitor mode is available only when the plotter is programmed-on; refer to the ESC.Y or ESC.(instructions later in this chapter. When the plotter is programmed-off, data is transferred between the computer and terminal in a transparent manner.

In either monitor mode, the communications channel between the terminal and computer is disconnected to assure that the terminal does not take part in any handshake process between the plotter and the computer. With the exception of the break signal, the plotter ignores all terminal-generated data. The plotter interprets the break signal as an instruction to flush the I/O buffer and return the plotter to the programmed-off mode.

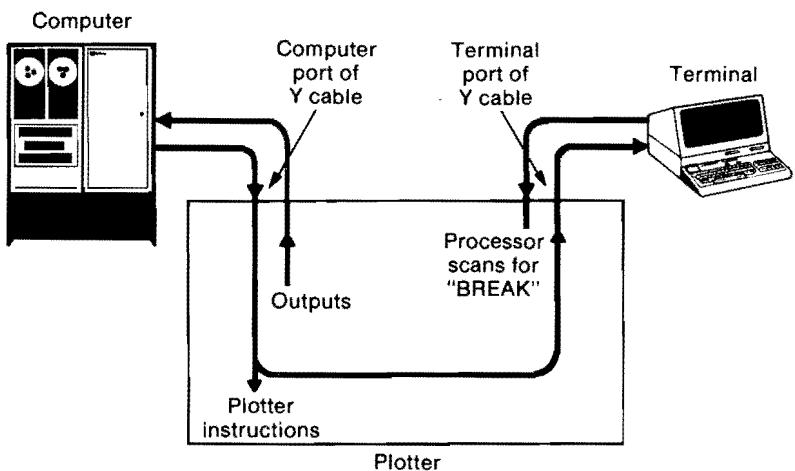
If **Duplex** is set to **FULL**, the plotter output responses are sent to the computer, but not to the terminal. The computer is working in an echo-plex environment, such that plotter responses to the computer are echoed to the terminal; if the plotter explicitly sent output responses to the terminal, the terminal would receive the responses twice (the explicit response and the echoed response).

If **Duplex** is set to **HALF**, the plotter output responses are sent to the terminal and the computer.

The following illustrates data flow for the two monitor modes.



15



RELATED

INSTRUCTION: ESC . @, Set Plotter Configuration

ESC . R, Reset

USE: Resets certain I/O conditions to power-up default states. Use this instruction to establish known conditions when starting a new plot.

SYNTAX: **ESC . R**

REMARKS: The ESC . R instruction aborts any device-control instruction currently in use, aborts any partially parsed HP-GL instruction, resets the parser, clears all buffers, and resets all buffers to their default sizes.

ESC . R is equivalent to sending the following four instructions:

ESC . J

ESC . K

ESC . P: (without parameters)

ESC . T: (without parameters)

(Refer to the detailed descriptions of ESC . J, ESC . K, and ESC . T elsewhere in this chapter. Refer to Chapter 16 for a detailed description of the ESC . P instruction.)

After sending ESC . R, send the ESC . L instruction to ensure that all conditions have been reset before any subsequent instructions are parsed. In this application, the ESC . L response is not important. However, to avoid potential errors, read the output response before sending more data. Refer to the ESC . L instruction for additional information.

ESC.S, Output Configurable Memory Size

USE: Outputs the total memory size of user-definable RAM, or the memory space available in one of its five buffers: the physical I/O buffer, polygon buffer, downloadable character buffer, vector buffer, and pen sort buffer. Use this instruction to determine how much memory is currently allocated to each buffer or to confirm the allocation performed by GM, ESC.T, or ESC.R.

SYNTAX: **ESC.S** *n*:

Parameter	Format	Range	Default
<i>n</i>	integer	0 to 6*	0 (total memory)

*Parameter 4 is an unused buffer.

RESPONSE:

Parameter	Response	Format	Range
none	memory size	integer	0 to 25 600

REMARKS: The following table shows each parameter value with its corresponding memory designation.

Parameter Value	Memory Specification
0	Total configurable memory
1	Physical I/O buffer
2	Polygon buffer
3	Downloadable character buffer
4	Reserved buffer
5	Vector buffer
6	Pen sort buffer

RELATED

INSTRUCTION: GM, Graphics Memory
ESC.R, Reset
ESC.T, Allocate Configurable Memory

ESC . T, Allocate Configurable Memory

USE: Allocates memory in user-definable RAM, which consists of five buffers: the physical I/O buffer, polygon buffer, downloadable character buffer, vector buffer, and pen sort buffer. Use this instruction to change the sizes of these buffers as needed.

SYNTAX: **ESC . T**(physical I/O buffer);(polygon buffer);(downloadable character buffer);0;(vector buffer);(pen sort buffer);

Parameter	Format	Range	Default
physical I/O buffer size	integer	2 to 25 518 bytes	1024
polygon buffer	integer	4 to 25 520 bytes	3072
downloadable character buffer	integer	0 to 25 516 bytes	0
reserved			0
vector buffer	integer	66 to 25 582 bytes	3000
pen sort buffer	integer	12 to 24 528 bytes	18 504

REMARKS: You can divide the plotter's configurable memory between the buffers. If you do not need to change the I/O buffer, you should consider using the GM instruction to change the sizes of the other buffers.

When ESC . T is executed, it performs the following operations (in sequence).

- Clears the I/O buffer, polygon, downloadable character, vector and pen sort buffers.
- Allocates memory as specified in its parameters.
- Resets the parser.

The following discusses each of the parameters in order.

- **Physical I/O buffer** — Affects the speed of data transmission. Reducing the size of the I/O buffer slows down data transmission. The larger the physical I/O buffer, the more data the computer can send to the plotter at a given time.

NOTE: For practical purposes, the HP-IB interface only uses the logical buffer. If you use ESC.T to increase the physical I/O buffer size from a previously defined value, you must also use the ESC.@ instruction to make the larger buffer space available on HP-IB. The ESC.B and ESC.L instructions, along with serial and parallel polls all give information about the logical buffer, so it is especially important to reset the logical buffer size using ESC.@ after enlarging the physical size with ESC.T.

When you set the physical buffer size to less than the current logical buffer size established by ESC.@", the logical size is automatically set to the value of the new physical size. ■

- **Polygon buffer** — Must be large enough to contain your largest polygon. Refer to *Using the Polygon Buffer* in Chapter 8 to determine the polygon buffer size.
- **Downloadable character buffer** — Must be large enough to contain your downloadable character(s). Refer to *Designing a Downloadable Character Set* in Chapter 13 to determine the downloadable character buffer size.
- **Reserved** — Not used. This parameter must always be zero (0).
- **Vector** — Allows the plotter to collect vectors and then plot them in one motion, thus creating smoother curves.
- **Pen sort buffer** — Allows the plotter to sort the vectors according to the pen required for drawing. This reduces both the number of times the plotter switches pens and amount of time required to draw the plot.

When allocating buffer space with the ESC.T instruction, be aware of the following recommendations.

- Allocate buffer sizes at the beginning of a program when the buffers are empty, before you send any HP-GL instructions. The ESC.T instruction is executed immediately, so HP-GL instructions in any of the buffers are cleared.
- To allocate buffer space during a program, send the ESC.O instruction before the ESC.T instruction. Check bit 3 of the response. When this bit indicates that the I/O and pen sort buffers are empty, send ESC.T to allocate your buffer space. After sending ESC.T, send ESC.L to allow the plotter enough time to clear and reset the buffers before resuming execution. The numeric value of ESC.L is not important in this application, but we recommend you read the value to avoid potential errors.

RELATED

INSTRUCTIONS: ESC.@", Set Plotter Configuration
ESC.S, Output Configurable Memory Size
GM, Graphics Memory

ERRORS:

Condition	Error	Plotter Response
sum of parameters exceeds 25 600 bytes	13	sets all buffers to their default sizes

ESC . U, End Flush Mode

USE: Ends flush mode. Use this instruction in spooling applications to end flush mode, thus allowing the plotter to begin parsing graphics instructions again.

SYNTAX: **ESC . U**

REMARKS: Flush mode is enabled when one of the front-panel function keys has been defined to be Escape and is pressed. Flush mode, however, is not triggered until the OG instruction has been executed. Once triggered, flush mode causes all incoming HP-GL data to be discarded. This is a useful feature in spooling applications where it is important for the plotter and application program to be able to respond to a user-initiated 'escape.'

The ESC . U instruction causes the plotter to stop discarding HP-GL data, and resets bit 9 of the extended status word (refer to the ESC . O instruction). After execution of ESC . U, HP-GL instructions can again be parsed.

RELATED

INSTRUCTIONS: ESC . O, Output Extended Status
KY, Define Key
OG, Output Group Count

ESC.Y or ESC.(, Plotter On

USE: Enables the plotter to accept data and interpret it as graphics or device-control instructions. Use this instruction in Eavesdrop (RS-232-C interface only) to establish programmed-on operation.

SYNTAX: **ESC.Y** or **ESC.(**

REMARKS: With **Dataflow** set to **EAVESDROP**, (refer to the User's Guide), the plotter is automatically programmed-off at power-on. You must send this instruction from the computer to the plotter before it can respond to any other instructions. After the plotter receives this instruction, it interprets all subsequent data from the selected source as plotter instructions. If the plotter is already programmed-on, it ignores this instruction. Program the plotter off to use the terminal to communicate with the computer.

RELATED

INSTRUCTIONS: **ESC.Z** or **ESC.)**, Plotter Off

ESC.Z or ESC.), Plotter Off

USE: Disables the plotter so that it accepts only a plotter-on instruction. Use this instruction in Eavesdrop (RS-232-C interface only) to establish programmed-off operation.

SYNTAX: **ESC.Z** or **ESC.)**

REMARKS: After receiving a plotter-off instruction, any remaining HP-GL instructions in the buffer are executed; no additional instructions are accepted until the plotter receives a plotter-on instruction.

RELATED

INSTRUCTIONS: **ESC.Y** or **ESC.(**, Plotter On

ESC .@, Set Plotter Configuration

USE: *For RS-232-C users*, this instruction sets an effective logical I/O buffer size and controls hardwire handshake, communications protocol, monitor modes 1 and 2, and block I/O error checking.

For HP-IB users, sets an effective logical I/O buffer size. Use the instruction to enlarge the logical I/O buffer.

SYNTAX: **ESC .@(*logical I/O buffer size*);(*I/O conditions*)**:

Parameter	Format	Range	Default
logical I/O buffer size	integer	0 to 25 518 bytes*	1024 bytes
I/O conditions	integer	0 to 31**	3

*Practical range; actual range is 0 to 65 535.

**Practical range; actual range is 0 to 127.

REMARKS: The physical I/O buffer is the actual portion of the configurable graphics memory where storage and parsing of HP-GL instructions takes place. The logical I/O buffer can be thought of as the operational subset of the physical I/O buffer; it is not a separate buffer. It is the size of the logical I/O buffer that limits plotter I/O functions such as the quantity of HP-GL instructions waiting to be parsed, and threshold levels in certain handshaking methods.

The logical I/O buffer cannot be larger than the physical I/O buffer. At power-on, the physical I/O buffer and the logical I/O buffer are the same size. If you decrease the size of the physical I/O buffer, the logical I/O buffer automatically decreases to the same size. However, enlarging the physical I/O buffer does not affect the logical I/O buffer.

Use ESC.T to change the physical I/O buffer size; ESC.@ to change the logical I/O buffer size. The plotter interprets the parameters as follows.

- **Logical I/O buffer size** — Sets the size of the logical I/O buffer. If you do not specify this parameter, the plotter sets the logical I/O buffer to the same size as the current physical I/O buffer. Always increase the physical I/O buffer size (using ESC.T) before you enlarge the logical I/O buffer size.
- **I/O conditions** — Specifies an integer equivalent value that controls the states of bits 0 through 4 of the configuration byte. When using an RS-232-C interface, these bits control hardwire handshake, communications protocol, monitor modes 1 and 2, and block I/O error checking. When using an HP-IB interface, this parameter is ignored. Refer to the ESC.Q instruction for information concerning monitor mode.

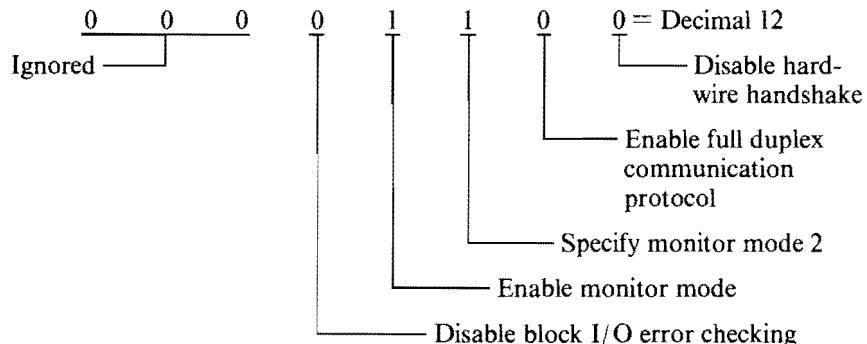
At power-on, bits 0 and 1 are set to 1; bits 2 through 4 are set to zero. When hardwire handshake is enabled, the plotter uses the Data Terminal Ready (DTR) line in the same manner that buffer threshold indicators are used in the Xon-Xoff handshake. DTR is set high at power-on and is not set low until available buffer space is less than the data block size specified by either ESC.H or ESC.I (default is 80 bytes). DTR remains low until the buffer space is greater than twice the data block size available. The condition occurring first causes DTR to be set high. When full duplex communication is enabled, the plotter can only transmit data to the computer when both DSR and CTS (pins 5 and 6 on the RS-232-C connector) are high.

The following table shows the logic states of the significant bits 0 through 4.

Bit Number	Logic State	Description
0	0	Disable hardwire handshake mode (set and hold pin 20 of RS-232-C port connector high).
	1	Enable hardwire handshake mode.
1	0	Enable full duplex data communication.
	1	Enable three-wire data communication.
2	0	Specify monitor mode 1 (only HP-GL instructions are displayed as they are parsed from the buffer). Refer to Monitor Mode in this chapter.
	1	Specify monitor mode 2 (all bytes are displayed as they are received by the plotter).
3	0	Disable monitor mode.
	1	Enable the monitor mode (RS-232-C only. Bit 2 specifies one of two monitor modes.)
4	0	Disable block I/O error checking.
	1	Enable block I/O checking.

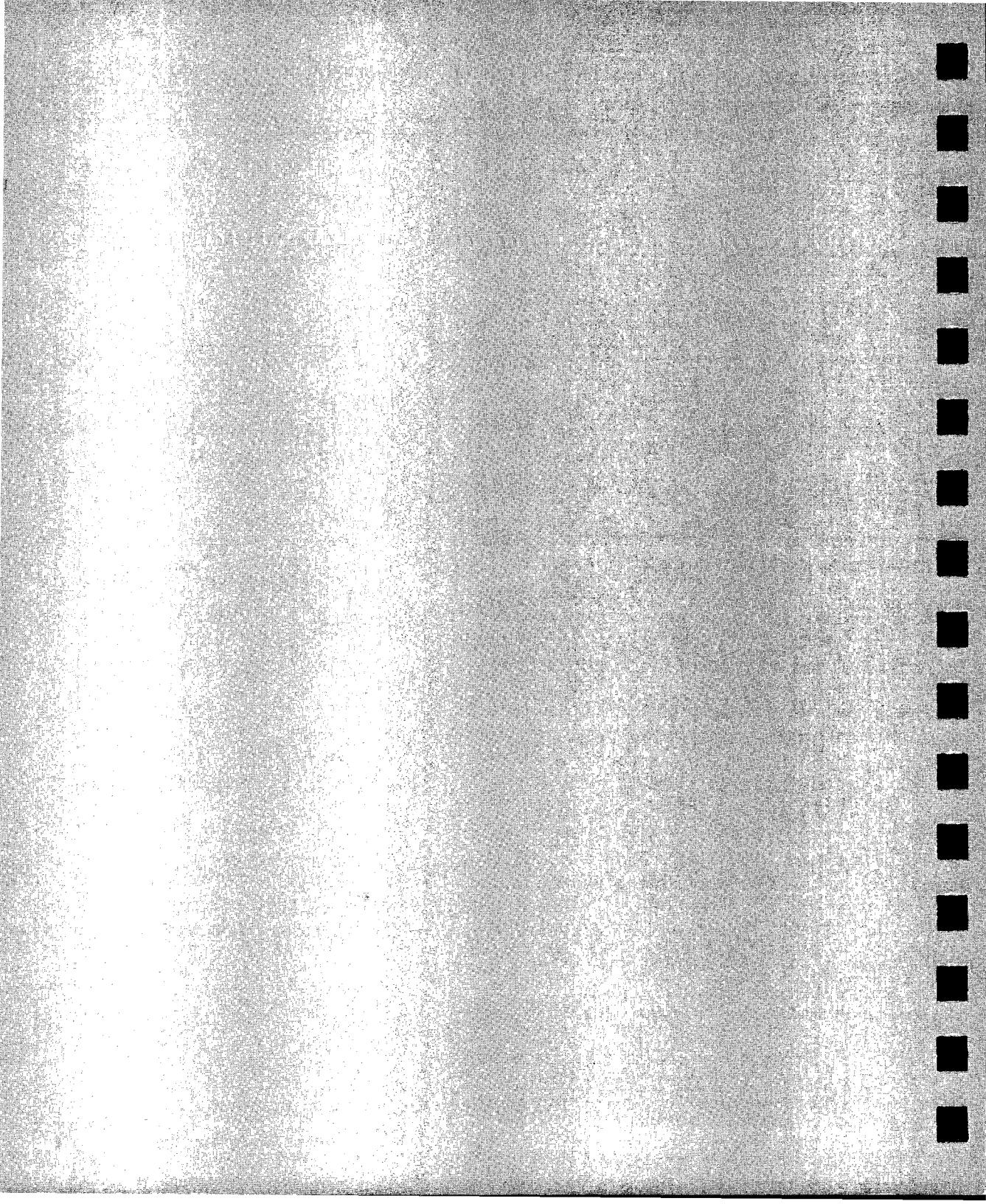
EXAMPLE: Using an RS-232-C interface, the following specifies a logical I/O buffer size of 5000 bytes. Decimal value 27 sends the ASCII **ESC** character. The decimal value 12 specifies the corresponding binary value to set the logic state of bits 0 through 3 as shown.

CHR\$(22)+“.@5000;12;”



Notes

15



16

Interfacing and Handshaking

This chapter discusses how to programmatically establish communication compatibility between the plotter and the computer. The User's Guide explains the steps necessary to connect your plotter to a computer and provides some program listings to establish critical interface and handshake conditions. The first part of this chapter discusses the HP-IB (IEEE-488 compatible) interface. The second part of this chapter explains the different handshakes, the device-control instructions necessary to establish communication on the RS-232-C interface. Read the section that applies to the interface you are using. If your computer is not listed in the User's Guide, this chapter can help you establish the appropriate handshake conditions for your interface.

16

The Hewlett-Packard Interface Bus (HP-IB)

The HP Interface Bus (HP-IB) transfers data and commands between the computer and plotter on 16 signal lines. Eight data I/O lines are reserved for the transfer of data and other messages in a byte-serial, bit parallel manner. Data and message transfer is asynchronous, coordinated by three handshake lines. The remaining five lines are for management of bus activity.

Devices connected to the bus can be talkers, listeners, or controllers. The controller dictates the role of each of the other devices by setting the attention (ATN) line true and sending talk or listen addresses on the data lines. Addresses are set into each device at the time of system configuration either by switches built into the device or by jumpers on an internal board. While the ATN line is true, all devices must listen to the data lines. When the ATN line is false, only devices that are addressed actively send or receive data.

This plotter's HP-IB adheres to the ANSI/IEEE-488 standard, and implements the following interface functions.

HP-IB Interface Functions

Interface Function	Capability Identification
Source Handshake	SH1
Acceptor Handshake	AH1
Talker	T6
Listener	L3
Service Request	SR1
Device Clear	DC1
Remote Local	RL0
Device Trigger	DT0
Controller	C0
Parallel Poll	PP0, PP1, or PP2*

*PP0 if listen only mode; PP1 if address ≥ 8 ; PP2 if address < 8 . Refer to *Parallel Polling* in Chapter 14 for the correspondence between the address and data line used in parallel poll.

For most applications, you probably only need to understand how to address your plotter. Refer to the User's Guide for instruction on setting the plotter's address.

When using the HP-IB, the plotter responds to 12 device-control instructions. The following lists the device-control instructions that are valid when using the HP-IB. (These instructions are not restricted to HP-IB; they can be used in an RS-232-C interface also.)

ESC . A	ESC . O
ESC . B	ESC . R
ESC . E	ESC . S
ESC . J	ESC . T
ESC . K	ESC . @
ESC . L	

All of the instructions listed above are described in Chapter 15. The plotter recognizes but ignores eight additional device-control instructions. Five of these eight additional device-control instructions are valid only for RS-232-C interfacing and handshaking and are described at the end of this chapter.

Controlling Addressing Sequences

When you are programmatically controlling the HP-IB, one of the first things you must consider is addressing. The following table lists the listen and talk characters for specific addresses.

Plotter Address	Address Characters	
	Listen	Talk
0	(space)	@
1	!	A
2	"	B
3	#	C
4	\$	D
5	%	E
6	&	F
7	,	G
8	(H
9)	I
10	*	J
11	+	K
12	,	L
13	-	M
14	.	N
15	/	O
16	0	P
17	1	Q
18	2	R
19	3	S
20	4	T
21	5	U
22	6	V
23	7	W
24	8	X
25	9	Y
26	:	Z
27	;	[
28	<	\
29	=]
30	>	^
31	?	—

An addressing sequence is made up of three major parts.

<Unlisten Command> <Talk Address> <Listen Addresses>

The purpose of these parts is as follows.

- The unlisten command is a universal bus command; its character is ? (ASCII decimal code 63). It unaddresses all listeners. After transmitting the unlisten command, no active listeners remain on the bus.
- The talk address indicates the device that is to talk, or send data. A new talk address automatically unaddresses the previous talker.
- The listen addresses indicate one or more devices that are to listen, or receive data. A listen address adds the designated device as listener along with other addressed listeners.

This addressing sequence directs who talks to whom. You can implement the commands (unlisten, talk, listen) by putting data on the bus and setting the proper control line true. The unlisten command (?) plays a vital role in this sequence. It is important that a device receive only the data that is intended for it.

When a new talk address is transmitted in the addressing sequence, the previous talker is unaddressed. Therefore, only the new talker can send data on the bus and you don't need to use an untalk command in the same manner as the unlisten command.

For example, to tell a computer at address 21 to talk and a plotter at address 05 to listen, the controller (usually the computer) sets the proper control line true and sends the following sequence:

? U %

where ? — tells all devices on the bus to unlisten,

U — designates the device at address 21 as the talker,

% — designates the device at address 05 as the listener.

To have the plotter talk and the computer listen, you would set the control lines and set the following.

? E 5

Reactions to Bus Commands DCL and SDC

The computer can set all devices on the HP-IB system to a predefined or initialized state by sending the device clear command, DCL. The computer can also set selected devices to a predefined or initialized state by sending a selected device clear command, SDC, along with the addresses of the devices. The basic difference is that devices obey SDC only if they are addressed to listen, whereas DCL clears all devices on the bus. You can override all bus operations and return the bus to an inactive state by sending the interface clear command, IFC, from the computer. IFC does not affect data already received by the plotter.

When the plotter receives a DCL or SDC command, it allows the instruction or vector being executed to finish, aborts any partially decoded instruction, and clears the I/O, pen sort, bidirectional, and vector buffers of all remaining graphics instructions. The plotter finishes executing its current vector but all remaining vectors are aborted. (The DCL and SDC commands do not reset any parameters in the plotter to default values. They are not the same as the HP-GL instructions DF or IN.) Partially parsed HP-GL instructions and/or parameters are lost.

RS-232-C Interfacing and Handshaking

Interfacing establishes communication by matching a set of conditions between the computer and the plotter. Your system's requirements dictate the interface conditions you must set on your plotter. To establish compatible interface conditions, your computer and plotter must agree on the following.

- **Number of data bits** — The plotter uses standard 7-bit ASCII code; it is not compatible with 6-bit or 12-bit ASCII devices. (If data from the computer is not in this format, you need a protocol converter.)
- **Parity** — ASCII characters are coded in seven bits, with an eighth bit for parity, or error-checking. Refer to the User's Guide to set **Parity** to the same parity as your computer.
- **Baud rate** — The rate at which transmitted data is sent (approximately equal to bits per second). You must set the plotter's baud rate to match the baud rate of the computer; otherwise, the plotter will not be able to understand the data. Refer to the User's Guide to set the **Baud Rate** to the same as the computer.
- **Number of stop bits** — On the plotter, baud rates 75 and 110 use 2 stop bits while baud rates of 200 and faster use 1 stop bit.

If your computer is not listed in the User's Guide, check your computer system documentation to determine the values for the above information.

Choosing a Handshake

Once you establish the interface conditions, you must select a handshake method. The plotter can implement any of the following four handshakes.

- hardwire (factory default)
- Xon-Xoff
- enquire/acknowledge (ENQ/ACK)
- software checking

Handshaking establishes the manner in which your plotter and computer transmit data once the interface is established. Since computers can send data faster than a plotter can process it, a handshake method is required to prevent data from being lost or misinterpreted.

For information on which handshake to use, consult your system's documentation or the installation manual for your computer and/or graphics software package. Most graphics software packages designed for use with RS-232-C plotters contain the instructions necessary to set up a handshake. You may need to provide your software with parameters suitable for your system. Instructions for setting the parameters can be found in the software installation guide.

The following paragraphs describe each of the handshake methods listed above. Use these descriptions along with your computer documentation to determine the handshake you should use. You must know your computer's requirements to have your computer and plotter communicate efficiently.

Hardwire Handshake

The hardwire handshake is the plotter's factory default handshake. You can turn **Hardwire: ON** or **OFF** from the front panel. This handshake uses the plotter's Data Terminal Ready (DTR) control line (pin 20) to control handshaking. You can use this handshake if your computer can monitor pin 20.

The hardwire handshake requires that the plotter be connected directly to the computer; there can be no intermediary devices (such as modems).

Xon-Xoff Handshake

The Xon-Xoff handshake is controlled by the plotter. You can use this handshake **only** if your computer supports an Xon-Xoff protocol. When using the Xon-Xoff handshake, the plotter transmits a control character to the computer when the plotter's I/O buffer is full, and another character when the buffer is ready to receive more data.

Enquire/Acknowledge Handshake

The enquire/acknowledge (ENQ/ACK) handshake is controlled by the computer. The computer sends an enquire character that prompts the plotter to respond when there is enough room in the buffer for more data. When sufficient I/O buffer space is available, the plotter sends an acknowledgment string that signals the computer to send data. The ENQ/ACK handshake is not as efficient as the Xon-Xoff handshake when there is a wide range of record lengths.

Software Checking Handshake

Probably the least efficient of the handshakes, you can implement this method on almost any computer system. You *must* use it, however, if your computer system can not implement any of the other handshaking methods. This handshake is managed by the programmer. To use this handshake, you must include device-control instructions in your program to repeatedly check the availability of I/O buffer space.

Handshaking Through Device-Control Instructions

You can establish each handshake programmatically using device-control instructions. This enables you to switch from one handshake to another for different applications.

The following list of device-control instructions directly affect your handshake and handshake capabilities. Use these instructions in combination with the device-control instructions in Chapter 15 to enhance the communication between your computer and plotter.

- ESC . H, Set Handshake Mode 1
- ESC . I, Set Handshake Mode 2
- ESC . M, Set Output Mode
- ESC . N, Set Extended Output and Handshake
- ESC . P, Define Handshake

Hardwire Handshake

The hardwire handshake takes place in the hardware rather than the firmware or software. To use hardwire handshake, you must connect the plotter directly to the computer; there can be no intermediate hardware (such as modems) between the plotter and computer. Hardwire handshake is the factory default handshake. You can turn **Hardwire: ON** or **OFF** from the front panel. The software can also turn it on or off

using the ESC .@ instruction. Most personal computers can implement a hardware handshake; refer to your computer's documentation.

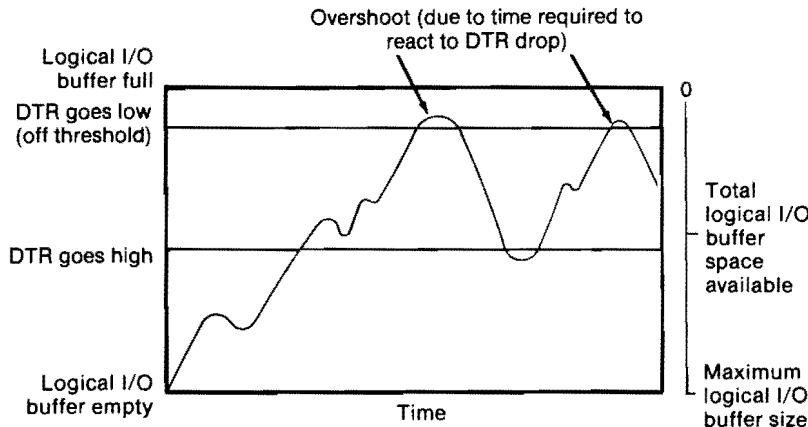
In a hardware handshake, the computer monitors one of the interface lines from the plotter. The hardware handshake works as follows.

- When the plotter has room for data in its I/O buffer, it signals the computer to send data by setting the DTR line high (turning it on). The DTR line is pin 20 on the plotter's RS-232-C connector. The DTR line is high at power-on.

The plotter raises the DTR line when the I/O buffer space is twice the data block size (default data block size is 80 bytes).

- The plotter has a threshold level that determines when the I/O buffer is in danger of overflowing and losing data. When the threshold level (the data block size specified by the ESC .H or ESC .I instructions) is reached, the plotter sets the DTR line low (off).

The computer continually checks the status of the line: if the line is high, it sends data; if the line is low, the computer waits until the line is high again before sending more data. This prevents the computer from over-filling the plotter's I/O buffer.



Use either of the following device-control instructions to initiate a hardware handshake in your program.

- ESC .P
- ESC .@

Using ESC.P To Initiate a Hardwire Handshake

The ESC.P device-control instruction with a parameter of 3 (ESC.P3;) sets several parameters for you. It is the simplest method that initiates a hardwire handshake. Be aware that these parameter values may not be the best for your computer system. If your system requires other values, consider the following device-control instructions.

- ESC.M to change the output terminator
- ESC.I to change the 'off' threshold level

To override parameter values set by the ESC.P instruction, place the appropriate instruction *after* ESC.P.

Using ESC.@ To Initiate a Hardwire Handshake

If a hardwire handshake has been disabled by a previous program, use ESC.@ to enable it. As with the ESC.P instruction, you can add the ESC.I instruction to set the threshold level. Since the other parameters of ESC.I are unnecessary for a hardwire handshake, they need not be included. The following shows an example using the ESC.@ and ESC.I instructions. ASCII decimal code 27 sends the **ESC** character. The first parameter of the ESC.@ instruction is set to its default value by using just the semicolon.

```
CHR$(27)+";@;3:"  
CHR$(27)+".I10:"
```

The following defines the parameter values indicated by the above example.

ESC. @

I/O Buffer size = 1024 bytes (default)
Hardwire handshake = 3-wire data communication and hardwire
handshake (enabled)

ESC. I

'Off' Threshold level = 10 bytes

Xon-Xoff Handshake

Refer to your computer system's documentation to determine whether or not you can use the Xon-Xoff handshake.

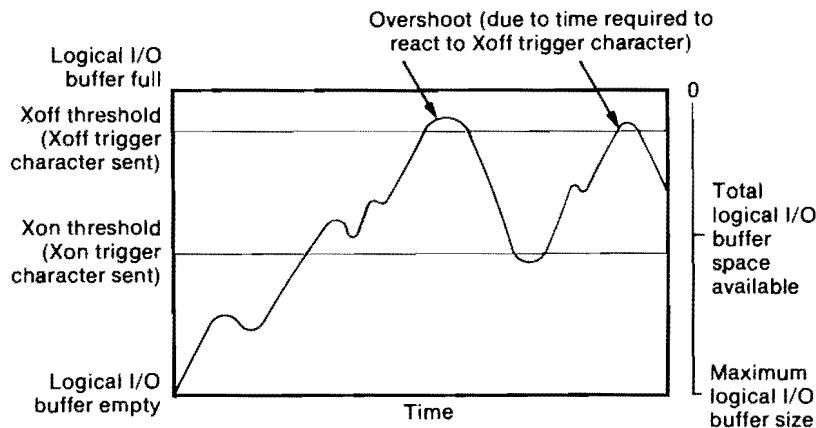
When using the Xon-Xoff handshake method, the plotter controls the data exchange sequence by signaling the computer when it has sufficient room in its I/O buffer for



data and when to stop sending data. The plotter uses buffer threshold indicators (an Xon trigger character and an Xoff trigger character) to prevent buffer overflow.

1. Data enters the plotter's buffer faster than it can be processed, and the buffer starts to fill. When the data in the I/O buffer reaches the Xoff threshold level (twice the data block size), the plotter sends the Xoff trigger character to the computer.
2. The plotter's buffer empties as data is processed. When the Xon threshold level is reached (the data block size specified by the ESC . H or ESC . I instructions), the plotter sends the Xon trigger character to the computer, restarting the flow of data.

This process is repeated until all data has been sent. The following is a graphic representation of this process.



NOTE: There is a delay between the time the signal is sent from the plotter and the time the computer stops sending data. Allow extra buffer room to avoid losing data. ■

Use either of the following device-control instructions to initiate a Xon-Xoff handshake in your program.

- **ESC . P**
- **ESC . I** and **ESC . N**

Using ESC.P To Initiate a Xon-Xoff Handshake

Using ESC.P with a parameter of 1 (ESC.P1:) sets several Xon-Xoff parameters for you; it is the simplest way to initiate an Xon-Xoff handshake with a device-control instruction. Be aware that these parameter values may not be the best for your computer. If your computer requires other values, consider the following instructions.

ESC.I to change the Xoff threshold level and Xon trigger character

ESC.N to change the intercharacter delay and Xoff trigger character

ESC.M to change the turnaround delay, output trigger character, echo terminate character, and output terminator

To override parameter values set by the ESC.P instruction, place the appropriate instruction *after* the ESC.P instruction.

Using ESC.I and ESC.N To Initiate a Xon-Xoff Handshake

To initiate an Xon-Xoff handshake using the ESC.I instruction, you must omit its second parameter (enquiry character). Also, send the ESC.N instruction to establish an intercharacter delay and the Xoff trigger character. The following shows an example using the ESC.N and ESC.I instructions. The ASCII decimal code 27 sends the **ESC** character.

```
CHR$(27)+“.I10;;17:”  
CHR$(27)+“.N0;19:”
```

The parameter values defined by the above example are as follows.

ESC.I

Xoff threshold level = 10 bytes

Enquiry character = omitted (indicates Xon-Xoff)

Xon trigger character = 17 (ASCII **DC1**)

ESC.N

Intercharacter delay = 0 (no delay)

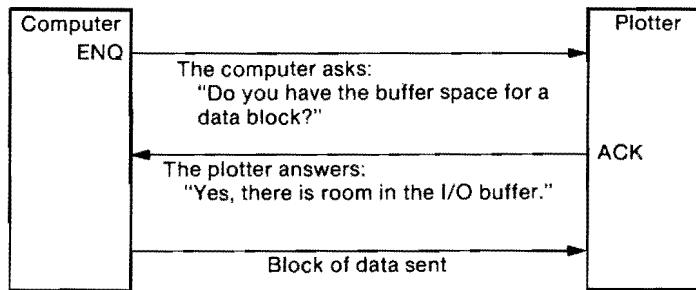
Xoff trigger character = 19 (ASCII **DC3**)

These are commonly used Xon-Xoff trigger character values. Check your system documentation to determine your system's requirements.

Enquire/Acknowledge Handshake

This handshake method derives its name from the two ASCII characters, **ENQ** and **ACK**, used on some systems as the enquiry character and acknowledgment string. The enquire/acknowledge handshake prevents the computer from sending the plotter more data than its buffer can accommodate. The computer sends the enquiry character to the plotter, asking if there is enough room in its I/O buffer for a block of data. The computer waits until it receives an acknowledgment string from the plotter signaling that there is sufficient room in the I/O buffer for a block of data. Only then does the computer send a block of data. In this way, the computer does not send the plotter more data than its buffer can accommodate.

The diagram below illustrates how an enquire/acknowledge handshake works.



Use any of the following device-control instructions to initiate an enquire/acknowledge handshake in your program.

- **ESC . P**
- **ESC . I**
- **ESC . H**

The method you use depends on your computer system's requirements. If your computer documentation lists the enquire/acknowledge handshake, you should consider either ESC . P or ESC . I. You can use the ESC . H instruction in systems where the output trigger character, echo terminator, and output terminator must be used with the enquire and acknowledge exchange in addition to being used with plotter output responses.

Using ESC.P to Initiate Enquire/Acknowledge Handshake

Using ESC.P with a parameter of 2 (ESC.P2:) sets several ENQ/ACK parameters for you. Be aware that these parameter values may not be the best for your system. If your computer requires other values, consider the following instructions.

ESC.I to change the block size, enquiry character, and acknowledgment string.

ESC.M to change the turnaround delay, output trigger, echo terminate, and output terminate characters.

ESC.N to change the intercharacter delay and immediate response string.

To override the parameter values set by the ESC.P instruction, place the appropriate instruction *after* ESC.P.

Using ESC.I to Initiate Enquire/Acknowledge Handshake

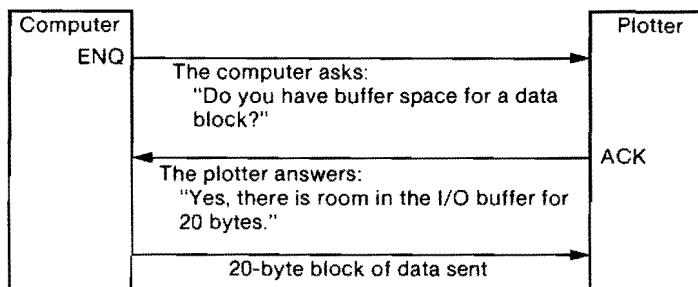
The ESC.I instruction also lets you establish an enquire/acknowledge handshake. If your computer does not *require* that you use the output trigger character, echo terminator, or output initiator with the handshake characters, use ESC.I to initiate the enquire/acknowledge handshake.

`CHR$(27)+“.I20;5;6:”`

The following lists the parameter values this instruction specifies.

Block size	= 20 bytes
Enquiry character	= 5 (ASCII ENQ)
Acknowledgment string	= 6 (ASCII ACK)

The following illustrates the data exchange this handshake would implement.



Note that you can also use the ESC.M instruction to specify the turnaround delay and the ESC.N instruction to change the intercharacter delay and immediate response string.

Using ESC.H to Initiate Enquire/Acknowledge Handshake

If your computer cannot implement a true enquire/acknowledge handshake, you can use the ESC.H to initiate a form of a software checking enquire/acknowledge handshake that can be implemented by your software and used on all computers. This method, however, is time consuming and requires that you specify ESC.M and ESC.N parameters. The following shows an example of this generic enquire/acknowledge handshake. The ASCII decimal code 27 sends the **ESC** character.

```
CHR$(27)+“.M100;17;0;13:”  
CHR$(27)+“.N50;21:”  
CHR$(27)+“.H20;5;6:”
```

The following lists the parameter values these instructions specify.

ESC.M

Turnaround delay	= 100 milliseconds
Output trigger character	= 17 (ASCII DC1)
Echo terminate character	= none
Output terminator	= 13 (ASCII CR)

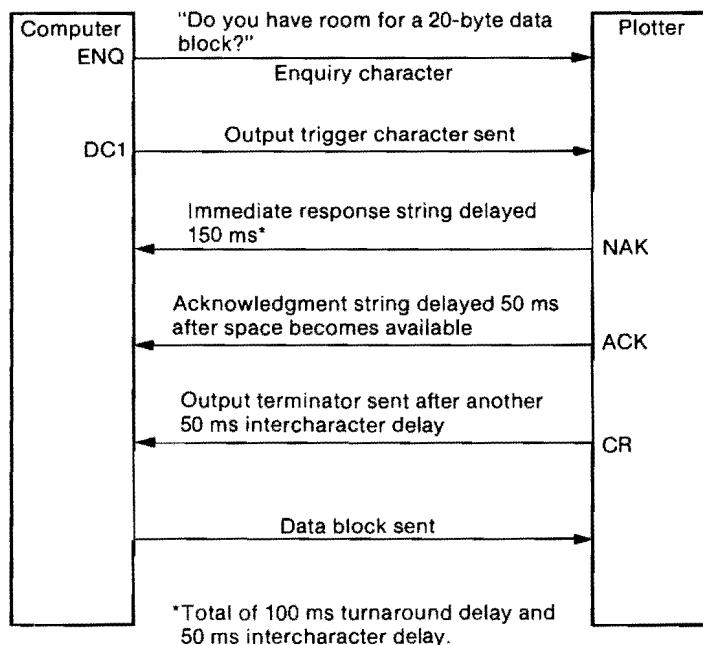
ESC.N

Intercharacter delay	= 50 milliseconds
Immediate response string	= 21 (ASCII NAK)

ESC.H

Data block size	= 20 bytes
Enquiry character	= 5 (ASCII ENQ)
Acknowledgment string	= 6 (ASCII ACK)

The following illustrates the data exchange this handshake would implement.



The following table summarizes the maximum specifications (instructions and parameters) needed to establish an enquire/acknowledge handshake using the ESC.I or ESC.H instructions. In particular, it shows which parameters of the ESC.M instruction are used with ENQ/ACK characters depending on the device-control instruction that initiates the handshake. All specified ESC.M parameters are used with plotter output responses regardless of the device-control instruction establishing the handshake.

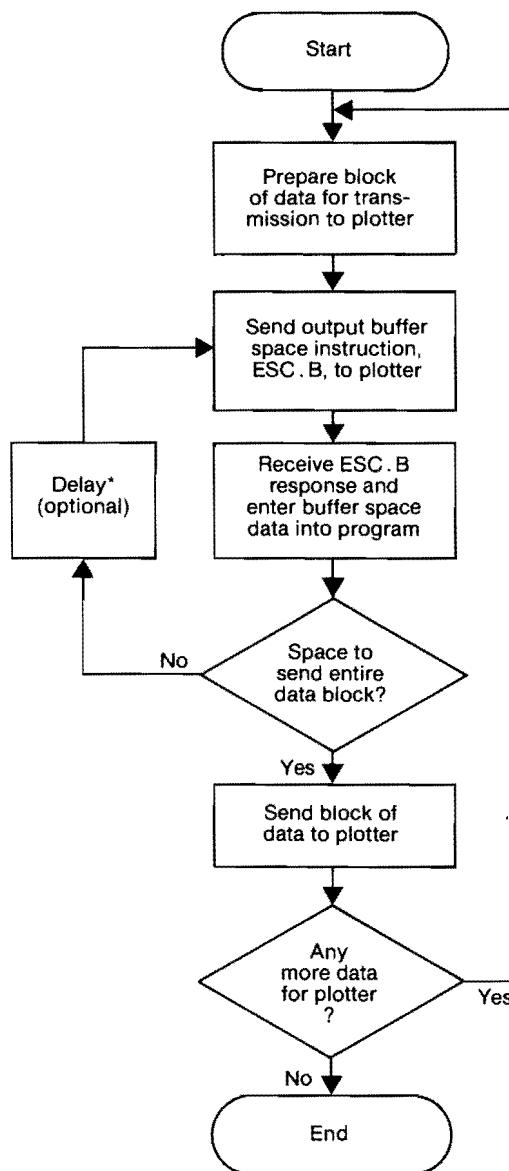
Enquire/Acknowledge Instructions and Parameters Used in Handshake Response

ESC . H block size enquiry character acknowledgment string	ESC . I block size enquiry character acknowledgment string
ESC . M turnaround delay output trigger character echo terminate character output terminator output initiator	ESC . M turnaround delay
ESC . N intercharacter delay immediate response string	ESC . N intercharacter delay immediate response string

Software Checking Handshake

When your computer system cannot use any of the previously described handshakes, you must incorporate some form of software checking handshake to prevent data from being lost when transferred between the computer and plotter. The ESC . B instruction (Output Buffer Space) is an effective way to monitor the plotter's I/O buffer. After you send the ESC . B instruction, the plotter outputs the number of empty bytes in the I/O buffer. Program a pause after successive ESC . B instructions to allow the plotter more time to process the instructions it already has and thereby improve efficiency. Send the block of data when there is sufficient room in the plotter's I/O buffer. Repeat this process until all data has been sent.

The following chart illustrates how a typical software checking handshake works within a program.



*Insert computer instructions that will temporarily halt the program, such as a WAIT statement or a FOR...NEXT loop.

Software Checking Handshake

This method can use large amounts of computer and plotter I/O processing time and is therefore inefficient in any environment, it is especially slow in time-shared environments. To reduce the inquiries about the availability of I/O buffer space, consider the following techniques.

1. Count the number of bytes to send to the plotter, then fill the I/O buffer before you send the initial inquiry about available I/O buffer space.
2. After filling the I/O buffer, or receiving a negative reply concerning available buffer space, wait a short time before sending another buffer space inquiry.

Use the following instructions to match the requirements of your computer system. Check your computer's documentation to determine which of the following parameters are required.

Device-Control Instruction	Parameter
ESC . M	Turnaround delay Output trigger character Echo terminate character Output terminator Output initiator
ESC . N	Intercharacter delay

Using ESC . B to Initiate a Software Checking Handshake

The enquire/acknowledge handshake includes an example that helps set up a type of software checking handshake. A second (though less efficient) method of software handshake uses the ESC . B, Output Buffer Space, instruction. (Refer to Chapter 15 for a complete description of this device-control instruction.)

In addition to the ESC . B instruction, this example also uses the ESC . M and ESC . N instructions to establish a turnaround delay and an intercharacter delay. Since all other parameters are omitted, they assume their default values. Use the ESC . B instruction to send a block of data *after* ESC . M and ESC . N. The ASCII decimal code 27 sends the **ESC** character.

```
CHR$(27)+“.M250:”  
CHR$(27)+“.N50:”  
CHR$(27)+“.B”
```

The parameter values for the above example are as follows.

ESC . M

Turnaround delay = 250 milliseconds
Output terminator = ASCII code 13 (**CR**; default value)

ESC . N

Intercharacter delay = 50 milliseconds

Data Transmission Modes

When you are using the plotter in an RS-232-C environment, you can use two modes of data transmission: normal mode and block mode. Normal mode is the default mode. Use the ESC .@ instruction (Set Plotter Configuration) to switch from one mode to the other. Refer to Chapter 15 for a complete description of the ESC .@ instruction.

Normal Mode

The plotter is in normal mode when you turn it on. In normal mode, all HP-GL instructions are stored in an I/O buffer where they are parsed in the order in which they are received, then they are either executed or moved to the pen sort buffer (if pen sorting is enabled). Device-control instructions do not enter the buffer, but are executed immediately.

Block Mode

Block mode lets you monitor the transmission of a block of data for any errors that may reach the plotter. This allows you to retransmit the block of data again so that the plotter receives it correctly, thus preventing errors in the plot.

After sending the ESC .@ instruction to turn on block mode, you define a block by sending some data followed by the ESC .E instruction (Output Extended Error). The ESC .E instruction serves as a block separator by terminating the first block of data and signaling the beginning of the subsequent block (if any). Refer to the ESC .E instruction in Chapter 15 for more information.

In block mode, all HP-GL instructions are stored in the I/O buffer until you send an ESC .E instruction to determine whether or not errors occurred; device-control and handshake instructions are not stored. You should always send the ESC .E instruction before you turn off block mode (using the ESC .@ instruction). If you turn off block

mode before sending the final ESC . E instruction, the I/O buffer will contain unparsed HP-GL instructions; they will not be executed.

Block mode has no effect on the type of handshake used or on the handshaking parameters defined. For example, if the number of characters in the block exceeds the logical buffer size, your handshake should prevent a buffer overflow error (error 16) from occurring.

NOTE: Do not use the ESC . L instruction in block mode as it may disrupt communication. If you must use the ESC . L instruction in block mode, send it immediately after an ESC . E instruction. Send the ESC . E instruction, read the response, send the ESC . L instruction, read the response, and then send the additional HP-GL instructions. ■

The following shows an example of a block mode transmission process.

Block I/O Error Checking

Computer	Plotter	Comments
ESC . @; 16: → Data block A →		Enable block I/O checking. Send a block of data. [Assume a character gets garbled (e.g., bad parity).]
ESC . E →	← 15 [TERM]	Any I/O errors? Parity error. At this point, the plotter discards the block because an error occurred.
Data block A → ESC . E →	← 0 [TERM]	Retransmit the block. Any I/O errors?
Data block B →		No errors. Plotter executes block.
ESC . E →	← 16 [TERM]	Send a block of data. [Assume a handshake character gets lost, and buffer overflows.]
Data block B → ESC . E →	← 0 [TERM]	Any I/O errors? Buffer overflow. Plotter discards block because an error occurred.
		Retransmit the block. Any I/O errors?
		No errors. Plotter executes block.

Automatic Modem Disconnect Modes

Two modes are available for automatic disconnection at the end of an RS-232-C session conducted over phone lines: switched/datex-line disconnection and leased-line disconnection. Hardwire handshake cannot be used when either disconnect mode is active.

Auto-Disconnect can be set to **OFF**, **SWITCHED/DATEX**, or **LEASED LINE** from the front panel. Refer to the User's Guide for details.

Switched/Datex-line Disconnect Mode

In this mode, the CTS (Clear To Send) and DSR (Data Set Ready) lines control the DTR (Data Terminal Ready) line. As long as the CTS and DSR lines are high, the DTR line is high. When either the CTS or DSR line goes low, the DTR line goes low; this causes a disconnection.

Leased-Line Disconnect Mode

In this mode, the CTS, DSR, and DCD (Data Carrier Direct) lines control the DTR line. As long as these three lines are high, the DTR line is high. If any of the three lines goes low, the DTR line goes low; this causes a disconnection.

ESC . H, Set Handshake Mode 1

USE: Configures the plotter for enquire/acknowledge handshake when the computer requires the parameters of ESC . M and ESC . N be used during the handshaking sequence.

SYNTAX: **ESC . H** (*data block size*);(*enquiry character*);(*acknowledgment string*):
or
ESC . H:

Parameter	Format	Range	Default
data block size	integer	0 to 25 518 bytes*	80 bytes
enquiry character	ASCII value	0 to 26, 28 to 31**	0 (no character)
acknowledgment string	ASCII value	0 to 126	0 (no character)

*Practical range; allowable range is 0 to 65 535.

**Practical range; printable characters (ASCII codes 32 to 126) can be used but should be avoided as they are required to send the HP-GL instructions.

REMARKS: When you send an ESC . H instruction without parameters (**ESC . H:**), the enquire/acknowledge handshake is disabled, data block size is 80 bytes, and there is no enquiry character or acknowledgment string. If however, the computer is configured to send an ENQ character anytime it is ready to send data to the plotter, the plotter automatically responds with ACK when it receives ENQ. This 'dummy' handshake is not dependent on available buffer space and does not protect against buffer overflow.

The plotter interprets the parameters as follows.

- **Data Block Size** — Specifies the maximum size of each block of data the plotter will receive from the computer. This must be less than the logical I/O buffer size.
- **Enquiry Character** — Prompts the plotter to acknowledge when there is room in the I/O buffer for a block of data. Any value other than zero enables the enquire/acknowledge handshake. Decimal code 5 (**ENQ**) is generally used as the enquiry character.
- **Acknowledgment String** — Signals the computer that the plotter has space available in the I/O buffer for a block of data. This parameter can be a string of up to 10 ASCII character codes, each subsequent character code separated from the previous one by a semicolon. Zero is not transmitted and terminates the string. Decimal code 6 (**ACK**) is generally used as the acknowledgment string. Avoid using characters that have a special meaning to the computer.

RELATED

INSTRUCTIONS: ESC . I

ESC . I, Set Handshake Mode 2

USE: Configures the plotter for either the Xon-Xoff or enquire/acknowledge handshakes when the computer does not expect the parameters of the ESC . M and ESC . N instructions to be used during the handshaking sequence. This is often true when the handshake protocol is part of the computer's operating system.

SYNTAX: Depends on the handshake being configured.

Xon-Xoff: **ESC . I** (*Xoff threshold level*); (*omitted*); (*Xon trigger character(s)*):

Parameter	Format	Range	Default
Xoff threshold level	integer	0 to logical I/O buffer size -1*	80 bytes
omitted	integer	0**	none
Xon trigger	ASCII value	0 to 126 1 to 10 decimal codes	0 (no character)

*Practical range; actual range is 0 to 65 535; however any value greater than the logical buffer size is changed to one byte less than the logical I/O buffer size.

**You can designate the omitted parameter by entering a 0 or by putting the semicolon without a parameter.

Enquire/Acknowledge: **ESC . I** (*data block size*); (*enquiry character*); (*acknowledgment string*):

Parameter	Format	Range	Default
data block size	integer	0 to 25 518 bytes*	80 bytes
enquiry character	ASCII value	0 to 26, 28 to 126** decimal code	0 (no character)
acknowledgment string	ASCII value	0 to 126 1 to 10 decimal codes	0 (no character)

*Practical range; allowable range is 0 to 65 535.

**Practical range; printable characters (ASCII codes 32 to 126) can be used but should be avoided as they are required to send the HP-GL instructions.

REMARKS: When you send an ESC.I instruction without parameters (**ESC . I:**), neither the Xon-Xoff nor enquire/acknowledge handshake is enabled, the data block size is 80 bytes, and there is no enquiry character or acknowledgment string. If, however, the computer is configured to send an **ENQ** character anytime it is ready to send data to the plotter, the plotter automatically responds with **ACK** when it receives **ENQ**. This 'dummy' handshake is not dependent on available buffer space and does not protect against buffer overflow.

For an Xon-Xoff handshake, the plotter interprets the parameters as follows.

- **Xoff Threshold Level** — Specifies the number of available bytes in the I/O buffer when the Xoff trigger character is to be sent. If you specify the Xoff threshold level greater twice the data block size, the plotter automatically resets the Xon threshold level so that the Xon character is sent when one byte more than the Xoff level is available.
- **Omitted Parameter** — Sets an Xon-Xoff handshake when you omit this parameter by entering only the semicolon, or the value zero followed by the semicolon. To enable the Xon-Xoff handshake, you must specify the next parameter.
- **Xon Trigger Character** — Specifies the character or characters the plotter sends the computer when there is sufficient space available in the I/O buffer. Although the **DC1** character (decimal code 17) is often used as the Xon trigger character, you can specify up to 10 character codes separated by semicolons.

For an enquire/acknowledge handshake, interpret the parameters as follows.

- **Data Block Size** — Specifies the maximum size of each block of data the plotter will receive from the computer.
- **Enquiry Character** — Prompts the plotter to when there is room in the I/O buffer for a block of Any value other than zero enables the enquire/acknowledge Decimal code 5 (**ENQ**) is generally used as the enquiry character.
- **Acknowledgment String** — Signals the computer that plotter has space available in the I/O buffer for a block of This parameter can be a string of up to 10 ASCII character codes, each subsequent character separated from the previous one by a semicolon. Zero is not transmitted and terminates the string. Decimal code 6 (**ACK**) is generally used as the acknowledgment string. Avoid using characters that have a special meaning to the computer.

EXAMPLE: For the Xon-Xoff handshake:

```
CHR$(27)+“.I81;;17:”
```

The ASCII decimal code 27 sends the **ESC** character. The parameters set the Xoff threshold level to 81 (the Xoff character is sent when 81 bytes remain in the plotter's I/O buffer) and the Xon trigger character to **DC1**. Note that the second parameter is omitted (as required by this handshake) by entering the semicolon only. Set the Xoff trigger character using the ESC.N instruction.

For the enquire/acknowledge handshake:

```
CHR$(27)+".I;5;6;"
```

The ASCII decimal code 27 sends the **ESC** character. The parameters set the block size to its default value of 80 bytes, the ASCII character **ENQ** as the enquiry character, and the single ASCII character **ACK** as the acknowledgment string. No output initiator will precede it, even if one is defined, and no output terminator will follow it.

For more information concerning these handshakes, refer to *Xon-Xoff Handshake* and *Enquire/Acknowledge Handshake* earlier in this chapter.

RELATED

INSTRUCTIONS: ESC.H, Set Handshake Mode 1
ESC.N, Set Extended Output and Handshake Mode

ESC.M, Set Output Mode

USE: Establishes parameters for the plotter's communication format. Use the instruction to establish a turnaround delay, an output trigger character, an echo terminator, and an output initiator character. Also use it to change the output terminator from its default value, ASCII decimal code 13 (carriage return).

SYNTAX: **ESC.M** (*turnaround delay*);(*output trigger*);(*echo terminator*);(*output terminator*);(*output initiator*):

Parameter	Format	Range	Default
turnaround delay	integer	0 to 9999 msecs	0
output trigger	ASCII value	0 to 4, 6 to 26, 28 to 126 decimal code	0 (no character)
echo terminator	ASCII value	0 to 4, 6 to 26, 28 to 126 decimal code	0 (no character)
output terminator	ASCII value	0 to 4, 6 to 26, 28 to 126 1 or 2 decimal codes	13 (carriage return)
output initiator	ASCII value	0 to 126 decimal code	0 (no character)

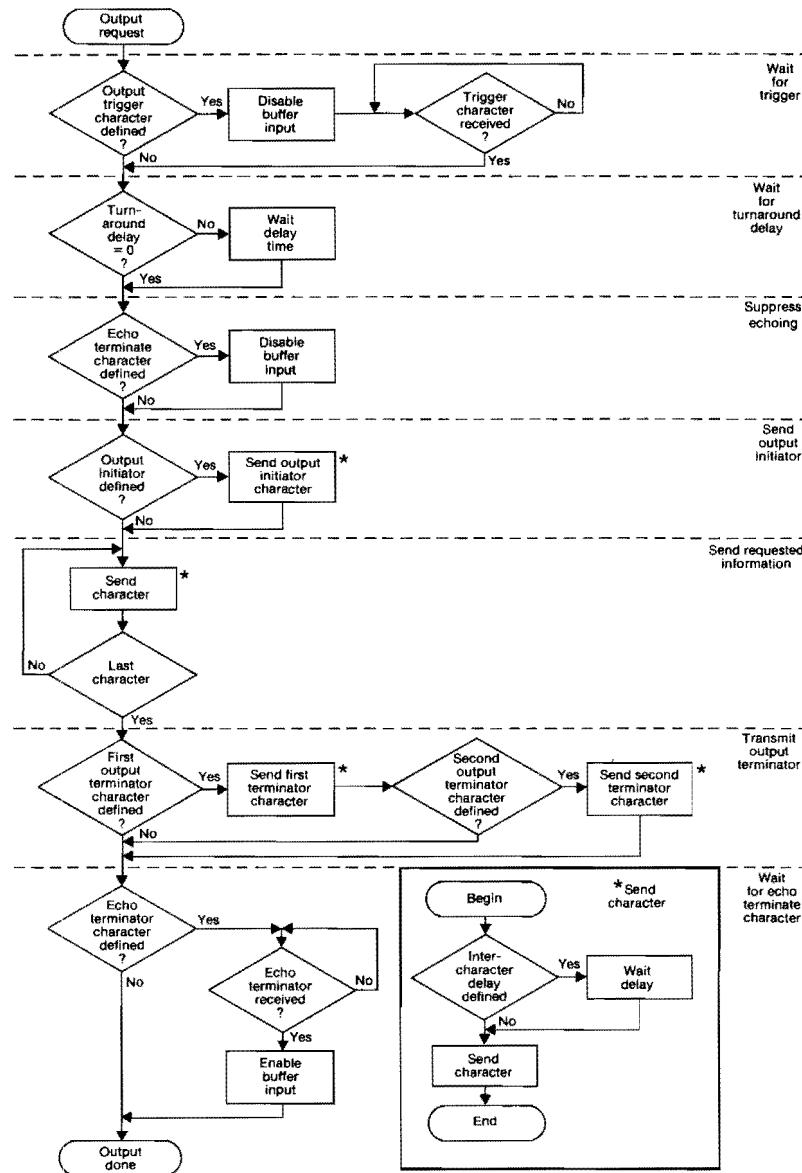
REMARKS: The plotter interprets the parameters as follows.

- **Turnaround Delay** — Prevents the plotter from sending data until the computer is ready to receive and process it. (An intercharacter delay set by ESC.N increments the turnaround delay.)
- **Output Trigger Character** — Is the last character output by the computer when making a request of the plotter. This signals the plotter to respond to the request. Most computers use the **DG1** character (decimal code 19) or the **LF** character (decimal code 10) the output trigger.
- **Echo Terminator** — Closes the plotter's I/O buffer while the computer echoes the plotter's responses. The I/O buffer remains closed until the echo response terminator is received. Refer to your computer documentation to determine if your computer echoes data. If your computer does not echo data, specify this parameter as zero or omit it.

The **LF** character (decimal code 10) is often used for the echo terminate character. If the computer echoes the plotter's response without a terminating character, then use the plotter's output terminator (next parameter) as the echo terminator.

- **Output Terminator** — Indicates the end of plotter response to the computer. This can be a one- or two-character terminator. If the next parameter (Output Initiator) is to be specified, the output terminator must consist of two characters, or the second character must be set to zero or omitted (by entering a semicolon). Most computers use the **CR** character (decimal code 13).
- **Output Initiator** — Indicates the beginning of a plotter response to the computer. To specify an output initiator, the output terminator must consist of two characters (see previous parameter description). Most computers do not use an output initiator. Many of those who do use **STX** for the output initiator.

The flowchart on the next page illustrates a plotter output request.



Output Request Flowchart

ESC . N, Set Extended Output and Handshake Mode

USE: Establishes parameters for the plotter's communication format. Use this instruction to specify an intercharacter delay in all handshake modes and either the immediate response string for the enquire/acknowledge handshake or the Xoff trigger character(s) for the Xon-Xoff handshake.

SYNTAX: **ESC . N** (*intercharacter delay*);(*handshake dependent parameter*):

Parameter	Format	Range	Default
intercharacter delay	integer	0 to 9999 msecs	0
handshake dependent parameter Xon-Xoff: Xoff trigger character(s)	ASCII value	0 to 126 up to 10 decimal codes	0 (no character)
Enquire/Acknowledge: immediate response string	ASCII value	0 to 126 up to 10 decimal codes	0 (no character)

REMARKS: The plotter interprets the parameters as follows.

- **Intercharacter Delay** — Specifies the length of transmission delay between each character output by the plotter. This allows extra time for computers with limited I/O port buffering capability to process data. The intercharacter delay is added to the turnaround delay (if one has been specified using the ESC . M instruction) before the plotter sends the first character, and is also inserted before each subsequent character in a string being sent to the computer.
- **Handshake Dependent Parameter** — Depends on the handshake method implemented on your plotter. It is valid in either an Xon-Xoff or enquire/acknowledge handshake environment and is interpreted as follows.
 - **Xoff Trigger Character** — (Xon-Xoff handshake environment only) signals the computer to temporarily stop sending data while the plotter processes what it has already received. The **DC3** character (decimal code 19) is often used for the Xoff trigger character.

— **Immediate Response String** — (Enquire/Acknowledge handshake environment only) indicates to the computer that the plotter is acknowledging receipt of an enquiry character and is checking for available buffer space. The **NAK** character (decimal code 21) is often used for the immediate response string.

In either handshake environment, the second parameter can list up to 10 ASCII characters, each separated from another by a semicolon.

RELATED

INSTRUCTIONS: ESC . I, Set Handshake Mode 2
ESC . M, Set Output Mode

ESC . P, Set Handshake Mode



USE: Sets one of three standard handshakes.

SYNTAX: **ESC . P** (*handshake*):

Parameter	Format	Range	Default
handshake	integer	0 (none) 1 (Xon-Xoff) 2 (ENQ-ACK) 3 (hardwire)	0

REMARKS: You can use this instruction to select among the four standard handshakes. (Use ESC . @, H, I, M, and N to enhance, or select nonstandard handshakes.) The following table shows the parameters and the handshakes they implement.

Parameter Value and Handshake Method				
0 None	1 Xon- Xoff*	2 Enq/ Ack*	3 Hardwire (default)	Predefined Handshake Parameters Established
0	50	0	0	Turnaround Delay
0	0	17	0	Output Trigger Character
0	10	10	0	Echo Terminate Character
13	13	13	13	Output Terminator
80	N/A	80	N/A	Block (Record) Size
1024	1024	1024	1024	Logical I/O Buffer Size
0	10	0	0	Intercharacter Delay
0	N/A	0	0	Immediate Response String
N/A	17	N/A	N/A	Xon Trigger Character
N/A	19	N/A	N/A	Xoff Trigger Character
0	0	5	0	Equity Character
6†	N/A	6	6†	Acknowledgment String
N/A	80	N/A	80	Threshold Level

*This handshake method is established as though you had used the ESC.I instruction.

†This is a "dummy" acknowledge and does not prevent buffer overflow.

RELATED

INSTRUCTIONS: ESC.@", Set Plotter Configuration

ESC.H, Set Handshake Mode 1

ESC.I, Set Handshake Mode 2

ESC.M, Set Output Mode

ESC.N, Set Extended Output and Handshake Mode

ERRORS:

Condition	Error	Plotter Response
parameter greater than 3	13	ignores the instruction

A

Error Messages

There are two types of errors: those related to HP-GL instructions, and those related to device-control instructions. To read an HP-GL error number in your program, use the OE instruction (presented in Chapter 11). To read a device-control error number in your program, use the ESC.E instruction (presented in Chapter 15). Note that for both HP-GL and device-control instructions error number 0 indicates that no error has occurred.

The following table lists the HP-GL error numbers, their meanings, and the probable cause of the error (if any).

HP-GL Errors

Error Number	Meaning	Possible Cause
1	unrecognized command	A mnemonic is incorrect or missing; an alphabetic character was specified in a parameter when a numeric character was expected.
2	wrong number of parameters	Too few or too many parameters; an incomplete X,Y coordinate pair.
3	bad parameter	A parameter is out of range.
5	unknown character set	A set other than -1, 0-59, 60, 70, 80, 99, 100, or 101 has been designated or invoked.
6	position overflow	A single label is so long that it exceeds the numeric range.
7	buffer overflow	One of the graphics memory buffers does not have enough space allocated.

The following table lists the device-control error messages, their meanings, and the probable cause of the error (if any).

Device-Control Errors

Error Number	Meaning	Possible Cause
10	bad output request	(<i>Serial only</i>). New output was requested before previous output was finished being transmitted. The previous output will continue normally and the new output will be ignored (thus causing the error).
11	bad byte after ESC .	Invalid character received after first two characters (ESC .) in a device-control instruction.
12	bad byte in I/O control	Invalid character received while parsing a device-control instruction. The parameter containing the invalid character and all following parameters are defaulted.
13	bad parameter	One or more parameters are out of range.
14	too many parameters	Too many parameters received. Additional parameters beyond the proper number are ignored; parsing of the instruction ends when a colon (normal termination) or the next ESC character (abnormal termination) is received.
15	bad transmission	(<i>Serial only</i>). A parity error has been detected.
16	buffer overflow	(<i>Serial only</i>). The physical I/O buffer has overflowed. As a result, one or more characters have been lost; therefore, an HP-GL error will probably occur.
17	transmit underrun	(<i>Serial only</i>). Transmit underrun can be caused by a baud rate mismatch between devices, or by excessive I/O activity in receive monitor mode.
18	indeterminate error	I/O error of indeterminate cause.

No Operation (NOP) Instructions

In order to maintain software compatibility with other HP plotters, this plotter recognizes the following HP-GL instructions as no operation (NOP) instructions. They are ignored and no error is generated.

BF, Buffer Plot

VA, Adaptive Velocity

VN, Normal Velocity

B

B

Reference Material

One of the most commonly used computer codes is ASCII. The plotter uses seven-bit ASCII encoding with an eighth bit for parity. ASCII is used by the plotter for I/O operations. The parity bit here is set to 0. Characters in ASCII are either nonprinting control characters or graphic printing characters (used, for example, in labeling). The following information discusses ASCII character and control codes and the character sets implemented by the plotter.

ASCII Character Codes

Often, it is convenient to refer to ASCII characters using their decimal codes (the decimal equivalents of their binary codes). The following table shows three ASCII characters, their binary codes, and their decimal codes. (The graphic representation of the decimal code is dependent on the currently selected character set.)

Character	ASCII Binary Code	ASCII Decimal Code
A	0100 0001	65
B	0100 0010	66
?	0011 1111	63

ASCII Control Codes

The plotter's reactions in label mode to nonprinting ASCII control codes (decimal codes 0 to 32) are shown in the following table. These reactions are true regardless of the character set currently used for labeling.

Reaction to Nonprinting Control Characters

Decimal Code	ASCII Character	All Sets
0	NULL	No Operation (NOP)
1	SOH	NOP
2	STX	NOP
3	ETX	Terminate Label Instruction
4	ETO	NOP
5*	ENQ	NOP
6	ACK	NOP
7	BEL	NOP
8	BS	Backspace
9**	HT	Horizontal Tab (½ Backspace)
10	LF	Line Feed
11	VT	Inverse Line Feed
12	FF	NOP
13	CR	Carriage Return
14	SO	Shift-Out (Select Alternate Character Set)
15	SI	Shift-In (Select Alternate Character Set)
16	DLE	NOP

*With an RS-232-C interface, unless 5 has been established as an enquiry character, the plotter will respond to an ENQ with an ACK.

**Using control character horizontal tab (decimal code 9) inside a label string moves the pen one-half character space back (equivalent to a *CP0.5,0,;*).

(Table continued)

Reaction to Nonprinting Control Characters (Continued)

Decimal Code	ASCII Character	All Sets
17	DC1	NOP
18	DC2	NOP
19	DC3	NOP
20	DC4	NOP
21	NAK	NOP
22	SYN	NOP
23	ETB	NOP
24	CAN	NOP
25	EM	NOP
26	SUB	NOP
27	ESC	NOP
28	FS	NOP
29	GS	NOP
30	RS	NOP
31	US	NOP
32	SP	Space

The plotter has 66 character sets. The following table lists these character sets.

Fixed-Space Vector Font	Variable-Space Arc Font	Fixed-Space Arc Font	Character Set Name	ISO Registration Number
0	10	20	ANSI ASCII	006
1	11	21	9825 Character Set	—
2	12	22	French/German	—
3	13	23	Scandinavian	—
4	14	24	Spanish/Latin American	—
5	15	25	Special Symbols	—
6	16	26	JIS ASCII	014
7	17	27	Roman Extensions	—
8	18	28	Katakana	—
9	19	29	ISO IRV (International Reference Version)	002
30	40	50	ISO Swedish	010
31	41	51	ISO Swedish for Names	011
32	42	52	ISO Norway, Version 1	060
33	43	53	ISO German	021
34	44	54	ISO French	025
35	45	55	ISO United Kingdom	004
36	46	56	ISO Italian	015
37	47	57	ISO Spanish	017
38	48	58	ISO Portuguese	016
39	49	59	ISO Norway, Version 2	061
60	70	80	ISO French	069
99	—	—	Drafting Set	—
100 & 101	—	—	Kanji	—

Character Sets

The following pages show the plotter's character sets. (The Kanji character set, character sets 100 and 101, is shown in Appendix D.) All printing ASCII characters and their decimal codes (33–126) are listed for each character set.

NOTE: Each of the shaded symbols is automatically backspaced one character before it is drawn. Therefore, when you need to accent or underscore a letter, you should label the letter first, then the accent or underscore. In addition, the special centered symbols in character set 5, 15, and 25 (decimal codes 65–79) are designed for use in symbol mode with the SM instruction. When used in a label instruction, spacing will be irregular and may produce undesirable results. ■

Fixed-Space Vector Font

Decimal Value	SET																			
	0	1	2	3	4	5	6	7	8	9	30	31	32	33	34	35	36	37	38	39
33	!	!	!	!	!	!	!	!	!	À	.	!	!	!	!	!	!	!	!	!
34	“	”	”	”	”	”	”	”	”	Â	”	”	”	”	”	”	”	”	”	”
35	#	#	£	£	£	£	#	#	£	£	£	£	£	£	£	£	£	£	£	\$
36	\$	\$	\$	\$	\$	\$	\$	\$	£	,	□	□	□	\$	\$	\$	\$	\$	\$	\$
37	%	%	%	%	%	%	%	%	£	·	%	%	%	%	%	%	%	%	%	%
38	&	&	&	&	&	&	&	&	†	‡	&	&	&	&	&	&	&	&	&	&
39	‘	’	’	’	’	’	’	’	’	’	’	’	’	’	’	’	’	’	’	’
40	(((((((()	‘	((((((((((
41)))))))))	’))))))))))
42	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
43	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
44	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~
45	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
46	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
47	/	/	/	/	/	/	/	/	/	£	/	/	/	/	/	/	/	/	/	/
48	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
49	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
50	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
51	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
52	4	4	4	4	4	4	4	4	ç	I	4	4	4	4	4	4	4	4	4	4
53	5	5	5	5	5	5	5	5	ç	†	5	5	5	5	5	5	5	5	5	5
54	6	6	6	6	6	6	6	6	Ñ	†	6	6	6	6	6	6	6	6	6	6
55	7	7	7	7	7	7	7	7	ñ	†	7	7	7	7	7	7	7	7	7	7
56	8	8	8	8	8	8	8	8	‡	8	8	8	8	8	8	8	8	8	8	8
57	9	9	9	9	9	9	9	9	‡	9	9	9	9	9	9	9	9	9	9	9
58	:	:	:	:	:	:	:	:	¤	¤	:	:	:	:	:	:	:	:	:	
59	:	:	:	:	:	:	:	:	£	†	:	:	:	:	:	:	:	:	:	
60	<	<	<	<	<	<	<	<	¥	§	<	<	<	<	<	<	<	<	<	<
61	=	=	=	=	=	=	=	=	§	‡	=	=	=	=	=	=	=	=	=	=
62	>	>	>	>	>	>	>	>	f	‡	>	>	>	>	>	>	>	>	>	>
63	?	?	?	?	?	?	?	?	¢	‡	?	?	?	?	?	?	?	?	?	?
64	€	€	€	€	€	€	€	€	â	‡	€	€	€	€	€	€	€	€	€	€

Fixed-Space Vector Font (Continued)

Decimal Value	SET																		
	0	1	2	3	4	5	6	7	8	9	30	31	32	33	34	35	36	37	38
65	A	A	A	A	A	ä	A	é	ñ	A	A	A	A	A	A	A	A	A	A
66	B	B	B	B	B	ö	B	ö	ü	B	B	B	B	B	B	B	B	B	B
67	C	C	C	C	C	å	C	ó	ñ	C	C	C	C	C	C	C	C	C	C
68	D	D	D	D	D	+	D	á	ł	D	D	D	D	D	D	D	D	D	D
69	E	E	E	E	E	x	E	é	ł	E	E	E	E	E	E	E	E	E	E
70	F	F	F	F	F	*	F	ö	ł	F	F	F	F	F	F	F	F	F	F
71	G	G	G	G	G	*	G	ú	ł	G	G	G	G	G	G	G	G	G	G
72	H	H	H	H	H	x	H	à	ł	H	H	H	H	H	H	H	H	H	H
73	I	I	I	I	I	z	I	é	ł	I	I	I	I	I	I	I	I	I	I
74	J	J	J	J	J	y	J	ö	ł	J	J	J	J	J	J	J	J	J	J
75	K	K	K	K	K	x	K	ù	ł	K	K	K	K	K	K	K	K	K	K
76	L	L	L	L	L	*	L	ä	ł	L	L	L	L	L	L	L	L	L	L
77	M	M	M	M	M	x	M	ë	ł	M	M	M	M	M	M	M	M	M	M
78	N	N	N	N	N	i	N	ö	ł	N	N	N	N	N	N	N	N	N	N
79	O	O	O	O	O	*	O	ö	ł	O	O	O	O	O	O	O	O	O	O
80	P	P	P	P	P	-	P	å	ł	P	P	P	P	P	P	P	P	P	P
81	Q	Q	Q	Q	Q	i	Q	í	ł	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
82	R	R	R	R	R	r	R	ø	ł	R	R	R	R	R	R	R	R	R	R
83	S	S	S	S	S	s	E	ë	ł	S	S	S	S	S	S	S	S	S	S
84	T	T	T	T	T	t	ä	ł	ł	T	T	T	T	T	T	T	T	T	T
85	U	U	U	U	U	U	í	ł	ł	U	U	U	U	U	U	U	U	U	U
86	V	V	V	V	V	v	ø	ł	ł	V	V	V	V	V	V	V	V	V	V
87	W	W	W	W	W	w	æ	ł	ł	W	W	W	W	W	W	W	W	W	W
88	X	X	X	X	X	ä	ł	ł	ł	X	X	X	X	X	X	X	X	X	X
89	Y	Y	Y	Y	Y	y	í	ł	ł	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
90	Z	Z	Z	Z	Z	z	ö	ł	ł	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z
91	[[[0	[[0	0	0	[ä	ä	ä	ä	ä	ä	ä	ä	ä
92	\	ç	é	i	\	¥	é	ł	ł	\	ö	ö	ö	ö	ö	ö	ö	ö	ö
93]]]	0]]	i	ł	ł]	ä	ä	ä	ä	ä	ä	ä	ä	ä
94	^	^	^	^	^	^	8	"	"	^	ö	ö	ö	ö	ö	ö	ö	ö	ö
95	—	—	—	—	—	—	ö	ö	ö	—	—	—	—	—	—	—	—	—	—
96	~	~	~	~	~	~	á	á	á	~	é	é	é	é	é	é	é	é	é

B

Fixed-Space Vector Font

Decimal Value	SET																			
	0	1	2	3	4	5	6	7	8	9	30	31	32	33	34	35	36	37	38	39
97	a	a	a	a	a	ñ	a	Á	a	a	a	a	a	a	a	a	a	a	a	a
98	b	b	b	b	b	ó	b	ã	b	b	b	b	b	b	b	b	b	b	b	b
99	c	c	c	c	c	c	c	c	c	c	c	c	c	c	c	c	c	c	c	c
100	d	d	d	d	d	ú	d	ð	d	d	d	d	d	d	d	d	d	d	d	d
101	e	e	e	e	e	-	e	í	e	e	e	e	e	e	e	e	e	e	e	e
102	f	f	f	f	f	=	f	í	f	f	f	f	f	f	f	f	f	f	f	f
103	g	g	g	g	g	≈	g	ó	g	g	g	g	g	g	g	g	g	g	g	g
104	h	h	h	h	h	≈	h	ò	h	h	h	h	h	h	h	h	h	h	h	h
105	i	i	i	i	i	~	i	ö	i	i	i	i	i	i	i	i	i	i	i	i
106	j	j	j	j	j	≤	j	ð	j	j	j	j	j	j	j	j	j	j	j	j
107	k	k	k	k	k	≥	k	š	k	k	k	k	k	k	k	k	k	k	k	k
108	l	l	l	l	l	*	l	š	l	l	l	l	l	l	l	l	l	l	l	l
109	m	m	m	m	m	Δ	m	ú	m	m	m	m	m	m	m	m	m	m	m	m
110	n	n	n	n	n	ñ	n	ý	n	n	n	n	n	n	n	n	n	n	n	n
111	o	o	o	o	o	Σ	o	ÿ	o	o	o	o	o	o	o	o	o	o	o	o
112	p	p	p	p	p	±	p	þ	p	p	p	p	p	p	p	p	p	p	p	p
113	q	q	q	q	q	≠	q	þ	q	q	q	q	q	q	q	q	q	q	q	q
114	r	r	r	r	r	+	r	-	r	r	r	r	r	r	r	r	r	r	r	r
115	s	s	s	s	s	↑	s	μ	s	s	s	s	s	s	s	s	s	s	s	s
116	t	t	t	t	t	-	t	¶	t	t	t	t	t	t	t	t	t	t	t	t
117	u	u	u	u	u	↓	u	¾	u	u	u	u	u	u	u	u	u	u	u	u
118	v	v	v	v	v	ʃ	v	-	v	v	v	v	v	v	v	v	v	v	v	v
119	w	w	w	w	w	÷	w	¼	w	w	w	w	w	w	w	w	w	w	w	w
120	x	x	x	x	x	*	x	½	x	x	x	x	x	x	x	x	x	x	x	x
121	y	y	y	y	y	÷	y	á	y	y	y	y	y	y	y	y	y	y	y	y
122	z	z	z	z	z	·	z	º	z	z	z	z	z	z	z	z	z	z	z	z
123	{	π	~	~	~	~	~	~	{	{	≤	~	~	~	~	~	~	~	~	~
124		†	·	~	~	~			□		ö	ö	ø	ø	ù		ò	ñ	ç	ø
125)	+	~	~	~	~))	>)	à	à	à	ü	è)	è	ç	ð	à
126	~	~	~	~	~	~	~	~	±	-	ú	-	ú	-	ú	-	í	~	·	~

Variable-Space Arc Font

B

Decimal Value	SET																			
	10	11	12	13	14	15	16	17	18	19	40	41	42	43	44	45	46	47	48	49
33	!	!	!	!	!	!	!	À	.	!	!	!	!	!	!	!	!	!	!	!
34	„	„	„	„	„	„	„	„	„	„	„	„	„	„	„	„	„	„	„	„
35	#	#	£	£	£	£	#	#	È	Ј	#	#	#	#	£	£	£	£	£	§
36	\$	\$	\$	\$	\$	\$	\$	\$	Ê	·	¤	¤	¤	\$	\$	\$	\$	\$	\$	\$
37	%	%	%	%	%	%	%	%	Œ	·	%	%	%	%	%	%	%	%	%	%
38	&	&	&	&	&	&	&	&	†	ঃ	&	&	&	&	&	&	&	&	&	&
39	,	,	,	,	,	,	,	,	ঁ	ঁ	,	,	,	,	,	,	,	,	,	
40	(((((((()	।	((((((((((
41))))))))	ঁ	ঁ))))))))))
42	*	*	*	*	*	*	*	*	*	ই	*	*	*	*	*	*	*	*	*	*
43	+	+	+	+	+	+	+	+	+	ঁ	+	+	+	+	+	+	+	+	+	+
44	,	,	,	,	,	,	,	,	,	ঁ	,	,	,	,	,	,	,	,	,	
45	-	-	-	-	-	-	-	-	ঁ	ঁ	-	-	-	-	-	-	-	-	-	-
46	০	ঁ	
47	/	/	/	/	/	/	/	£	ঁ	/	/	/	/	/	/	/	/	/	/	/
48	0	0	0	0	0	0	0	—	-	0	0	0	0	0	0	0	0	0	0	0
49	1	1	1	1	1	1	1	Ý	ঁ	1	1	1	1	1	1	1	1	1	1	1
50	2	2	2	2	2	2	2	2	ঁ	১	2	2	2	2	2	2	2	2	2	2
51	3	3	3	3	3	3	3	3	•	ঁ	3	3	3	3	3	3	3	3	3	3
52	4	4	4	4	4	4	4	4	¢	I	4	4	4	4	4	4	4	4	4	4
53	5	5	5	5	5	5	5	5	¢	ঁ	5	5	5	5	5	5	5	5	5	5
54	6	6	6	6	6	6	6	6	Ñ	ঁ	6	6	6	6	6	6	6	6	6	6
55	7	7	7	7	7	7	7	7	ନ	ନ	7	7	7	7	7	7	7	7	7	7
56	8	8	8	8	8	8	8	8	ନ	ନ	8	8	8	8	8	8	8	8	8	8
57	9	9	9	9	9	9	9	9	ନ	ନ	9	9	9	9	9	9	9	9	9	9
58	:	:	:	:	:	:	:	:	¤	¤	:	:	:	:	:	:	:	:	:	
59	:	:	:	:	:	:	:	£	ନ	:	:	:	:	:	:	:	:	:	:	
60	<	<	<	<	<	<	<	¥	ନ	<	<	<	<	<	<	<	<	<	<	<
61	=	=	=	=	=	=	=	§	ନ	=	=	=	=	=	=	=	=	=	=	=
62	>	>	>	>	>	>	>	ନ	ନ	>	>	>	>	>	>	>	>	>	>	>
63	?	?	?	?	?	?	?	¤	ନ	?	?	?	?	?	?	?	?	?	?	?
64	@	@	@	@	@	@	@	ନ	ନ	@	@	É	@	§	ନ	@	§	§	§	@

Variable-Space Arc Font (Continued)

Decimal Value	SET																			
	10	11	12	13	14	15	16	17	18	19	40	41	42	43	44	45	46	47	48	49
65	A	A	A	A	A	ø	A	ê	ñ	A	A	A	A	A	A	A	A	A	A	A
66	B	B	B	B	ø	B	ð	ý	B	B	B	B	B	B	B	B	B	B	B	B
67	C	C	C	C	C	å	C	û	ñ	C	C	C	C	C	C	C	C	C	C	C
68	D	D	D	D	D	þ	D	á	þ	D	D	D	D	D	D	D	D	D	D	D
69	E	E	E	E	E	x	E	é	†	E	E	E	E	E	E	E	E	E	E	E
70	F	F	F	F	F	ø	F	ö	‡	F	F	F	F	F	F	F	F	F	F	F
71	G	G	G	G	G	þ	G	ú	þ	G	G	G	G	G	G	G	G	G	G	G
72	H	H	H	H	H	x	H	à	ž	H	H	H	H	H	H	H	H	H	H	H
73	I	I	I	I	I	z	I	è	ž	I	I	I	I	I	I	I	I	I	I	I
74	J	J	J	J	J	y	J	ð	ñ	J	J	J	J	J	J	J	J	J	J	J
75	K	K	K	K	K	x	K	ù	þ	K	K	K	K	K	K	K	K	K	K	K
76	L	L	L	L	L	*	L	ä	þ	L	L	L	L	L	L	L	L	L	L	L
77	M	M	M	M	M	ø	M	ë	^	M	M	M	M	M	M	M	M	M	M	M
78	N	N	N	N	N	i	N	ö	†	N	N	N	N	N	N	N	N	N	N	N
79	O	O	O	O	O	*	O	ø	?	O	O	O	O	O	O	O	O	O	O	O
80	P	P	P	P	P	-	P	å	Ξ	P	P	P	P	P	P	P	P	P	P	P
81	Q	Q	Q	Q	Q	i	Q	†	6	Q	’	Q	Q	Q	Q	Q	Q	Q	Q	Q
82	R	R	R	R	R	R	R	ø	×	R	R	R	R	R	R	R	R	R	R	R
83	S	S	S	S	S	S	S	€	€	S	S	S	S	S	S	S	S	S	S	S
84	T	T	T	T	T	T	T	ä	þ	T	T	T	T	T	T	T	T	T	T	T
85	U	U	U	U	U	U	U	f	l	U	U	U	U	U	U	U	U	U	U	U
86	V	V	V	V	V	V	V	ø	3	V	V	V	V	V	V	V	V	V	V	V
87	W	W	W	W	W	W	w	æ	5	W	W	W	W	W	W	W	W	W	W	W
88	X	X	X	X	X	X	Ä	ý	X	X	X	X	X	X	X	X	X	X	X	X
89	Y	Y	Y	Y	Y	Y	Y	í	ñ	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
90	Z	Z	Z	Z	Z	Z	ö	ü	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z
91	[[[[[[ø	o	[å	ä	æ	ä	’	[’	i	ä	æ	
92	\	†	ç	æ	i	\	¥	É	þ	\	ö	ö	ø	ö	ç	\	ç	ñ	ç	ø
93]]]]]]	t	þ]	å	å	å	å	ø	§]	é	ø	å	
94	^	↑	^	^	^	^	b	’	^	^	o	^	^	^	^	^	^	^	^	^
95	—	—	—	—	—	—	ö	’	—	—	—	—	—	—	—	—	—	—	—	—
96	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~

Variable-Space Arc Font (Continued)

Decimal Value	SET																		
	10	11	12	13	14	15	16	17	18	19	40	41	42	43	44	45	46	47	48
97	a	a	a	a	a	ñ	a	Á	a	a	a	a	a	a	a	a	a	a	a
98	b	b	b	b	b	ð	b	á	b	b	b	b	b	b	b	b	b	b	b
99	c	c	c	c	c	C	c	Ð	c	c	c	c	c	c	c	c	c	c	c
100	d	d	d	d	d	U	d	ð	d	d	d	d	d	d	d	d	d	d	d
101	e	e	e	e	e	—	e	í	e	e	e	e	e	e	e	e	e	e	e
102	f	f	f	f	f	=	f)	f	f	f	f	f	f	f	f	f	f	f
103	g	g	g	g	g	≈	g	ó	g	g	g	g	g	g	g	g	g	g	g
104	h	h	h	h	h	~	h	ò	h	h	h	h	h	h	h	h	h	h	h
105	i	i	i	i	i	~	i	õ	i	i	i	i	i	i	i	i	i	i	i
106	j	j	j	j	j	≤	j	ö	j	j	j	j	j	j	j	j	j	j	j
107	k	k	k	k	k	≥	k	š	k	k	k	k	k	k	k	k	k	k	k
108	l	l	l	l	l	≠	l	š	l	l	l	l	l	l	l	l	l	l	l
109	m	m	m	m	m	Δ	m	ú	m	m	m	m	m	m	m	m	m	m	m
110	n	n	n	n	n	π	n	ÿ	n	n	n	n	n	n	n	n	n	n	n
111	o	o	o	o	o	Σ	o	ÿ	o	o	o	o	o	o	o	o	o	o	o
112	p	p	p	p	p	±	p	þ	p	p	p	p	p	p	p	p	p	p	p
113	q	q	q	q	q	≠	q	þ	q	q	q	q	q	q	q	q	q	q	q
114	r	r	r	r	r	→	r	·	r	r	r	r	r	r	r	r	r	r	r
115	s	s	s	s	s	↑	s	μ	s	s	s	s	s	s	s	s	s	s	s
116	t	t	t	t	t	↔	t	¶	t	t	t	t	t	t	t	t	t	t	t
117	u	u	u	u	u	↓	u	¾	u	u	u	u	u	u	u	u	u	u	u
118	v	v	v	v	v	ʃ	v	—	v	v	v	v	v	v	v	v	v	v	v
119	w	w	w	w	w	÷	w	¼	w	w	w	w	w	w	w	w	w	w	w
120	x	x	x	x	x	*	x	½	x	x	x	x	x	x	x	x	x	x	x
121	y	y	y	y	y	◊	y	á	y	y	y	y	y	y	y	y	y	y	y
122	z	z	z	z	z	·	z	º	z	z	z	z	z	z	z	z	z	z	z
123	{	π	“	”	~	{	{	«	{	ä	ä	æ	ä	é	{	à	·	ä	æ
124		F	·	~			□		ö	ö	ø	ö	ù		ò	ñ	ç	ø	ø
125)	→	“	”	~))	»)	å	å	å	ü	è)	è	ç	ö	å
126	~	~	·	~	~	—	—	±	—	ü	—	ü	—	ü	—	ü	—	ü	—

B

Fixed-Space Arc Font

Decimal Value	SET																		
	20	21	22	23	24	25	26	27	28	29	50	51	52	53	54	55	56	57	58
33	!	!	!	!	!	!	!	!	!	À	.	!	!	!	!	!	!	!	!
34	„	„	„	„	„	„	„	„	„	À	„	„	„	„	„	„	„	„	„
35	#	#	£	£	£	£	#	#	È	»	#	#	#	#	£	£	£	£	#
36	\$	\$	\$	\$	\$	\$	\$	\$	È	„	\$	\$	\$	\$	\$	\$	\$	\$	\$
37	%	%	%	%	%	%	%	%	È	•	%	%	%	%	%	%	%	%	%
38	&	&	&	&	&	&	&	&	Ì	„	&	&	&	&	&	&	&	&	&
39	,	,	,	,	,	,	,	,	Ì	„	,	,	,	,	,	,	,	,	,
40	((((((((‘	Í	(((((((((
41))))))))	‘	Ó)))))))))
42	*	*	*	*	*	*	*	*	^	È	*	*	*	*	*	*	*	*	*
43	+	+	+	+	+	+	+	+	“	ò	+	+	+	+	+	+	+	+	+
44	,	,	,	,	,	,	,	,	~	†	,	,	,	,	,	,	,	,	,
45	-	-	-	-	-	-	-	-	Ù	ó	-	-	-	-	-	-	-	-	-
46	,	,	,	,	,	,	,	,	Ù	ó	,	,	,	,	,	,	,	,	,
47	/	/	/	/	/	/	/	/	æ	ó	/	/	/	/	/	/	/	/	/
48	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0
49	1	1	1	1	1	1	1	1	Ý	Þ	1	1	1	1	1	1	1	1	1
50	2	2	2	2	2	2	2	2	ý	Í	2	2	2	2	2	2	2	2	2
51	3	3	3	3	3	3	3	3	·	ó	3	3	3	3	3	3	3	3	3
52	4	4	4	4	4	4	4	4	ç	I	4	4	4	4	4	4	4	4	4
53	5	5	5	5	5	5	5	5	ç	õ	5	5	5	5	5	5	5	5	5
54	6	6	6	6	6	6	6	6	Ñ	ñ	6	6	6	6	6	6	6	6	6
55	7	7	7	7	7	7	7	7	ñ	†	7	7	7	7	7	7	7	7	7
56	8	8	8	8	8	8	8	8	í	‡	8	8	8	8	8	8	8	8	8
57	9	9	9	9	9	9	9	9	í	ñ	9	9	9	9	9	9	9	9	9
58	:	:	:	:	:	:	:	:	¤	õ	:	:	:	:	:	:	:	:	
59	;	;	;	;	;	;	;	;	£	†	;	;	;	;	;	;	;	;	
60	<	<	<	<	<	<	<	<	¥	ó	<	<	<	<	<	<	<	<	<
61	=	=	=	=	=	=	=	=	§	λ	=	=	=	=	=	=	=	=	=
62	>	>	>	>	>	>	>	>	f	†	>	>	>	>	>	>	>	>	>
63	?	?	?	?	?	?	?	?	¢	ó	?	?	?	?	?	?	?	?	?
64	ø	ø	ø	ø	ø	ø	ø	ø	à	ß	ø	ø	ø	ø	É	ø	§	à	ø

Fixed-Space Arc Font (Continued)

B

Decimal Value	SET																			
	20	21	22	23	24	25	26	27	28	29	50	51	52	53	54	55	56	57	58	59
65	A	A	A	A	A	ä	A	é	ñ	A	A	A	A	A	A	A	A	A	A	
66	B	B	B	B	B	ö	B	ô	ÿ	B	B	B	B	B	B	B	B	B	B	
67	C	C	C	C	C	ç	C	ó	ñ	C	C	C	C	C	C	C	C	C	C	
68	D	D	D	D	D	+	D	á	ł	D	D	D	D	D	D	D	D	D	D	
69	E	E	E	E	E	x	E	é	ł	E	E	E	E	E	E	E	E	E	E	
70	F	F	F	F	F	*	F	ó	ł	F	F	F	F	F	F	F	F	F	F	
71	G	G	G	G	G	+	G	ú	ł	G	G	G	G	G	G	G	G	G	G	
72	H	H	H	H	H	x	H	à	ł	H	H	H	H	H	H	H	H	H	H	
73	I	I	I	I	I	z	I	é	ł	I	I	I	I	I	I	I	I	I	I	
74	J	J	J	J	J	y	J	ò	ł	J	J	J	J	J	J	J	J	J	J	
75	K	K	K	K	K	x	K	ù	ł	K	K	K	K	K	K	K	K	K	K	
76	L	L	L	L	L	*	L	ä	ł	L	L	L	L	L	L	L	L	L	L	
77	M	M	M	M	M	z	M	ë	ł	M	M	M	M	M	M	M	M	M	M	
78	N	N	N	N	N	,	N	ö	ł	N	N	N	N	N	N	N	N	N	N	
79	0	0	0	0	0	*	0	ü	ł	0	0	0	0	0	0	0	0	0	0	
80	P	P	P	P	P	-	P	ä	ł	P	P	P	P	P	P	P	P	P	P	
81	Q	Q	Q	Q	Q	:	Q	í	ł	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	
82	R	R	R	R	R	R	R	ø	ł	R	R	R	R	R	R	R	R	R	R	
83	S	S	S	S	S	S	S	é	ł	S	S	S	S	S	S	S	S	S	S	
84	T	T	T	T	T	T	T	ä	ł	T	T	T	T	T	T	T	T	T	T	
85	U	U	U	U	U	U	U	í	ł	U	U	U	U	U	U	U	U	U	U	
86	V	V	V	V	V	V	V	ø	ł	V	V	V	V	V	V	V	V	V	V	
87	W	W	W	W	W	W	W	æ	ł	W	W	W	W	W	W	W	W	W	W	
88	X	X	X	X	X	X	X	ä	ł	X	X	X	X	X	X	X	X	X	X	
89	Y	Y	Y	Y	Y	Y	Y	í	ł	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	
90	Z	Z	Z	Z	Z	Z	Z	ö	ł	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	
91	[[[Ø	[[[0	0	[ä	ä	æ	ä	·	[·	i	ä	æ
92	\	√	ç	€	i	\	*	É	ó	\	ö	ö	ø	ç	\	ç	ñ	ç	ø	
93]]]	ø]]]	í	ł]	A	A	A	ø	\$]	é	ł	ø	A
94	^	↑	^	æ	^	^	^	B	^	^	0	^	^	^	^	^	^	^	^	
95	-	-	-	-	-	-	-	Ø	^	-	-	-	-	-	-	-	-	-	-	
96	~	~	~	~	~	~	~	Á	~	~	é	~	~	~	~	~	ú	~	~	

Fixed-Space Arc Font (Continued)

Decimal Value	SET																			
	20	21	22	23	24	25	26	27	28	29	50	51	52	53	54	55	56	57	58	59
97	a	a	a	a	a	n	a	Ã		a	a	a	a	a	a	a	a	a	a	
98	b	b	b	b	b	þ	b	ã		b	b	b	b	b	b	b	b	b	b	
99	c	c	c	c	c	c	c	Ð		c	c	c	c	c	c	c	c	c	c	
100	d	d	d	d	d	u	d	ð		d	d	d	d	d	d	d	d	d	d	
101	e	e	e	e	e	-	e	ƒ		e	e	e	e	e	e	e	e	e	e	
102	f	f	f	f	f	=	f	l		f	f	f	f	f	f	f	f	f	f	
103	g	g	g	g	g	≈	g	ø		g	g	g	g	g	g	g	g	g	g	
104	h	h	h	h	h	≈	h	ð		h	h	h	h	h	h	h	h	h	h	
105	i	i	i	i	i	~	i	ð		i	i	i	i	i	i	i	i	i	i	
106	j	j	j	j	j	≤	j	ð		j	j	j	j	j	j	j	j	j	j	
107	k	k	k	k	k	z	k	š		k	k	k	k	k	k	k	k	k	k	
108	l	l	l	l	l	*	l	š		l	l	l	l	l	l	l	l	l	l	
109	m	m	m	m	m	Δ	m	Ú		m	m	m	m	m	m	m	m	m	m	
110	n	n	n	n	n	Π	n	Ý		n	n	n	n	n	n	n	n	n	n	
111	o	o	o	o	o	Σ	o	ÿ		o	o	o	o	o	o	o	o	o	o	
112	p	p	p	p	p	±	p	þ		p	p	p	p	p	p	p	p	p	p	
113	q	q	q	q	q	†	q	þ		q	q	q	q	q	q	q	q	q	q	
114	r	r	r	r	r	→	r	·		r	r	r	r	r	r	r	r	r	r	
115	s	s	s	s	s	↑	s	μ		s	s	s	s	s	s	s	s	s	s	
116	t	t	t	t	t	+	t	¶		t	t	t	t	t	t	t	t	t	t	
117	u	u	u	u	u	↓	u	¾		u	u	u	u	u	u	u	u	u	u	
118	v	v	v	v	v	f	v	-		v	v	v	v	v	v	v	v	v	v	
119	w	w	w	w	w	÷	w	¼		w	w	w	w	w	w	w	w	w	w	
120	x	x	x	x	x	*	x	½		x	x	x	x	x	x	x	x	x	x	
121	y	y	y	y	y	◊	y	‡		y	y	y	y	y	y	y	y	y	y	
122	z	z	z	z	z	'	z	º		z	z	z	z	z	z	z	z	z	z	
123	{	π	"	"	~	{	{	«		{	ä	ä	æ	ä	é	{	à	'	ä	æ
124		†	"	"	~			□			ö	ö	ø	ö	ù		ò	ñ	ç	ø
125)	→	"	"	~))	»)	å	å	å	ü	è)	è	ç	ö	å
126	~	~	"	"	-	~	-	±		-	-	ü	-	þ	"	-	í	~	'	

Fixed-Space Vector, Variable-Space Arc, and Fixed-Space Arc Font

Decimal Value	SET		
	60	70	80
33	!	!	!
34	"	"	"
35	£	£	£
36	\$	\$	\$
37	%	%	%
38	&	&	&
39	'	'	'
40	(((
41)))
42	*	*	*
43	+	+	+
44	,	,	,
45	-	-	-
46	.	.	.
47	/	/	/
48	0	0	0
49	1	1	1
50	2	2	2
51	3	3	3
52	4	4	4
53	5	5	5
54	6	6	6
55	7	7	7
56	8	8	8
57	9	9	9
58	:	:	:
59	:	;	;
60	<	<	<
61	=	=	=
62	>	>	>
63	?	?	?
64	à	à	à

B

Fixed-Space Vector, Variable-Space Arc, and Fixed-Space Arc Font (Continued)

Decimal value	SET		
	60	70	80
65	A A A		
66	B B B		
67	C C C		
68	D D D		
69	E E E		
70	F F F		
71	G G G		
72	H H H		
73	I I I		
74	J J J		
75	K K K		
76	L L L		
77	M M M		
78	N N N		
79	O O O		
80	P P P		
81	Q Q Q		
82	R R R		
83	S S S		
84	T T T		
85	U U U		
86	V V V		
87	W W W		
88	X X X		
89	Y Y Y		
90	Z Z Z		
91	· · ·		
92	¢ ¢ ¢		
93	§ § §		
94	^ ^ ^		
95	— — —		
96	µ µ µ		

Fixed-Space Vector, Variable-Space Arc, and Fixed-Space Arc Font (Continued)

Decimal Value	SET
60	
70	
80	
97	a a a
98	b b b
99	c c c
100	d d d
101	e e e
102	f f f
103	g g g
104	h h h
105	i i i
106	j j J
107	k k k
108	l l l
109	m m m
110	n n n
111	o o o
112	p p p
113	q q q
114	r r r
115	s s s
116	t t t
117	u u u
118	v v v
119	w w w
120	x x x
121	y y y
122	z z z
123	é é é
124	ù ù ù
125	è è è
126	“ “ ”

Notes

B

C

Sample Plots and Program Listings

The first four examples in this appendix are designed to give you a better understanding of some of the fundamental HP-GL concepts, such as scaling, rotation, and using windows. The four programs are written in GW BASIC.

The explanatory text for each program listing describes the combination of HP-GL concepts illustrated by the program. As you enter each program, concentrate on seeing how these concepts interact with the rest of the program to bring about the desired result.

The last two examples are to assist users of FORTRAN and Pascal with converting the program examples to their language. They replicate in both FORTRAN and Pascal two examples given in earlier chapters.

The plot for each sample program is shown after the listing it represents. Also, the first four examples are written to fit on C-size paper. The others are written for A-size paper.

NOTE: In the listings, a single program line may be printed on two lines to fit on the page. You should enter such a line as though it were unbroken on the page. Otherwise, you may receive unexpected results. ■

Example 1: Scaling and P1/P2 Locations

This program illustrates the effect of P1 and P2 on scaling and uses relative plotting to draw a series of figures at any location on the paper. The program is divided into four sections.

- Section 1 sets P1 and P2 to near-default locations and uses no scaling when drawing the figures.
- Section 2 adds manual isotropic scaling to the current P1/P2 settings.
- Section 3 turns off scaling and dramatically changes the locations of P1 and P2. (Note how similar in size and shape these figures are to Section 1. When scaling is off, the locations of P1 and P2 have no affect on how a figure is drawn.)
- Section 4 isotropically scales these P1/P2 locations.

Each section is numbered and labeled on the plot. The numbers are labeled using size relative characters; that is, the size of the numbers is dependent on the relative locations of P1 and P2.

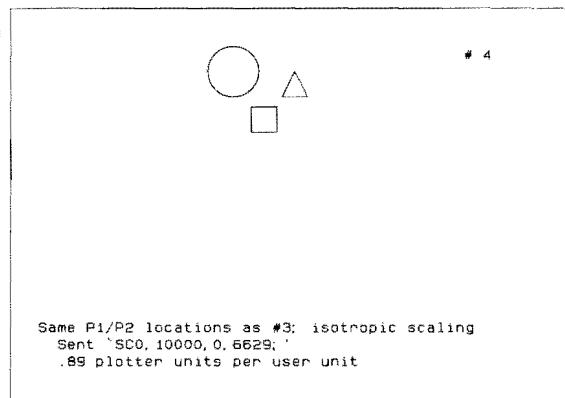
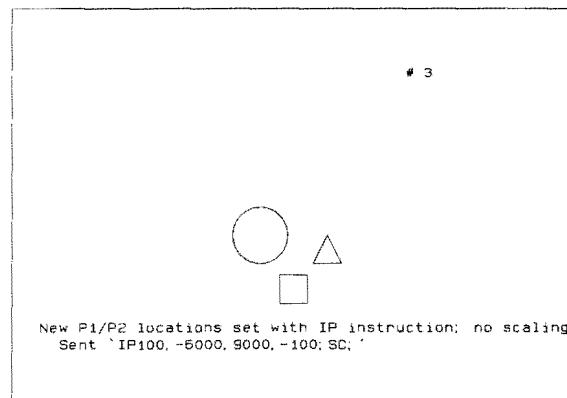
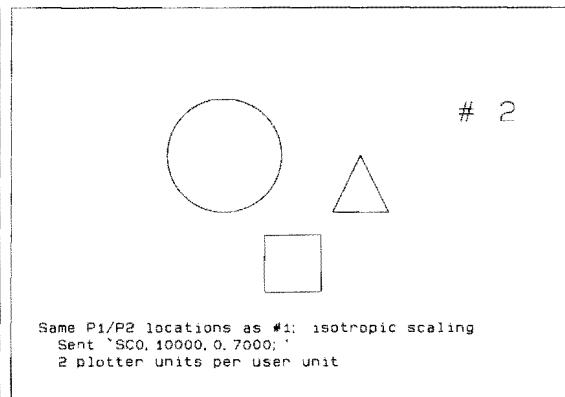
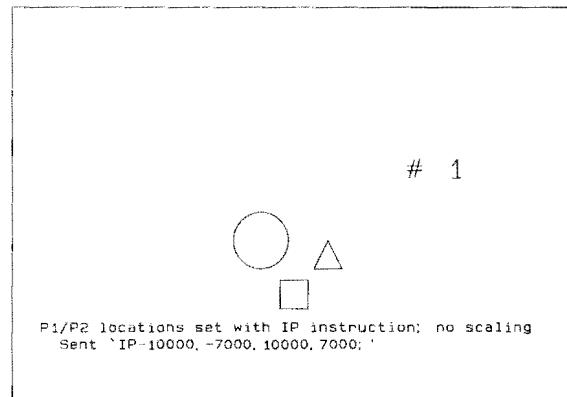
Additionally, the program draws each section using a different pen. We encourage you to use four noticeably different colors so that you can easily see the different concepts involved.

```
10  'Insert configuration statement here
15  '
25  ' Section 1 - IP sets P1/P2; no scaling
30  PRINT #1, "IN;IP-10000,-2000,10000,2000;"'
35  '
45  'draws box around the quadrant
50  PRINT #1, "SP1;PU-10000,2000;"'
60  PRINT #1, "PD-100,2000,-100,100,-10000,100,-10000,2000;"'
65  '
75  'draws figures
80  PRINT #1, "PU-5000,3500;"'
90  GOSUB 600
105 '
115 'numbers quadrant
120 PRINT #1, "SR1.2,2.4;PU-3000,4000;LB# 1"+CHR$(3)
130 PRINT #1, "SI;"'
135 '
145 'labels quadrant
150 PRINT #1, "PU-10000,100;"'
160 PRINT #1, "CP3,4;PR0,0;PA;"'
```

```
120 PRINT #1, "LBP1/P2 locations set with IP instruction;  
           no scaling"+CHR$(13)+CHR$(10)+CHR$(3)  
130 PRINT #1, "LB Sent 'IP-10000,-7000,10000,7000;'"  
           +CHR$(3)  
135 '  
145 'Section 2 - Same P1/P2 locations; isotropic scaling  
150 PRINT #1, "SC0,10000,0,7000;"  
155 '  
160 PRINT #1, "SP2;PU5050,3550;PD5050,2000;"  
165 PRINT #1, "PD10000,7000,10000,3550,5050,3550;"  
170 '  
175 PRINT #1, "PU9000,6000;SR1.2,2.4;LB# 2"+CHR$(3)  
180 '  
185 PRINT #1, "PA7525,6300;"  
190 GOSUB 600  
195 '  
200 PRINT #1, "PU5050,3550;SI;CP3,4;PR0,0;PA;"  
205 PRINT #1, "LBSame P1/P2 locations as #1; isotropic  
           scaling"+CHR$(13)+CHR$(10)+CHR$(3)  
210 PRINT #1, "LB Sent 'SC0,10000,0,7000;'" +CHR$(13)  
           +CHR$(10)+CHR$(3)  
215 PRINT #1, "LB 2 plotter units per user unit"+CHR$(3)  
220 '  
225 'Section 3 - Turn off scaling, move P1/P2  
230 PRINT #1, "SR1.2,2.4;IP100,-6000,9000,-100;SC;"  
235 PRINT #1, "SP3;PU-10000,-7000;"  
240 PRINT #1, "PD-100,-7000,-100,-100,-10000,-100,-10000,  
           -7000;"  
245 '  
250 PRINT #1, "PU-5000,-3500;"  
255 GOSUB 600  
260 '  
265 PRINT #1, "PU-3000,-1300;LB# 3"+CHR$(3)  
270 PRINT #1, "PU-10000,-7000;SI;CP3,4;PR0,0;PA;"  
275 PRINT #1, "LBNew P1/P2 locations set with IP; no  
           scaling"+CHR$(13)+CHR$(10)+CHR$(3)  
280 PRINT #1, "LB Sent 'IP100,-6000,9000,-100;SC;'" +CHR$(3)  
285 '  
290 'Section 4 - New P1/P2, isotropic scaling  
295 PRINT #1, "IP;SR1.2,2.4;IP100,-6000,9000,-100;"  
300 PRINT #1, "SC0,10000,0,6629;PU0,-1123;SP4;"  
305 PRINT #1, "PD11123,-1123,11123,6629,0,6629,0,-1123;"  
310 '  
315 PRINT #1, "PU5000,5979;"  
320 GOSUB 600  
325 '  
330
```

C

```
480 PRINT #1, "PU9000,5629;"  
470 PRINT #1, "LB# 4"+CHR$(3)  
480 PRINT #1, "PU0,-1123;SI;CP3,4;PR0,0;PA;"  
490 PRINT #1, "LBSame P1/P2 locations as #3; isotropic  
scaling"+CHR$(13)+CHR$(10)+CHR$(3)  
500 PRINT #1, "LB Sent 'SC0,10000,0,6629;'" +CHR$(13)  
      +CHR$(10)+CHR$(3)  
510 PRINT #1, "LB .89 plotter units per user unit" +CHR$(3)  
515 '  
520 PRINT #1, "SP0;"  
530 END  
535 '  
600 'draws a circle, an equilateral triangle,  
605 'and a square  
620 PRINT #1, "PR;PU-600,-600;CI500;PU1200,0;"  
630 PRINT #1, "PD250,-500,-500,0,250,500;"  
640 PRINT #1, "PU-350,-700;"  
650 PRINT #1, "PD-500,0,0,-500,500,0,0,500;PU;"  
660 PRINT #1, "PA;"  
670 RETURN
```



Example 2: Moving a Scaled P1

This program illustrates that scaling moves with subsequent changes in the location of P1. This allows you to plot a figure using absolute coordinates anywhere on the page by changing the location of the scaled P1 point. The figure is drawn at the same scaled location each time it is drawn; only P1 moves. The scaling is isotropic.

```
10  'Insert configuration statement here
20  PRINT #1, "IN;IP-4200,-6900,4200,6900;""
30  PRINT #1, "SC0,5000,0,8214;""
40  '
50  'label from scaling origin (0,0)
60  PRINT #1, "SP1;PA0,0;L02;LBOriginal P1: -4200,
      -6900"+CHR$(3)
70  '
80  'draw figure
90  GOSUB 400
100 '
110 'change P1 only; P2 follows
120 PRINT #1, "IP-4200,-2300;""
130 PRINT #1, "PA0,0;LBP1 set to: -4200,-2300"+CHR$(3)
150 GOSUB 400
160 '
170 'change P1, label, draw figure
180 PRINT #1, "IP-4200,2300;""
190 PRINT #1, "PA0,0;LBP1 set to: -4200,2300"+CHR$(3)
200 GOSUB 400
210 '
220 'change P1, label, draw figure
230 PRINT #1, "IP-2300,0;""
240 PRINT #1, "PA0,0;LBP1 set to: -2300,0"+CHR$(3)
250 GOSUB 400
260 '
270 'last P1 change
280 PRINT #1, "IP2520,-4140;""
290 PRINT #1, "PA0,0;LBP1 set to: 2520,-4140"+CHR$(3)
300 GOSUB 400
310 '
320 'return P1 to original values; label scaling
340 PRINT #1, "IP-4200,-6900;""
350 PRINT #1, "SC;PU0,5000;L05;SI.45,.65;CP0,-5;""
360 PRINT #1, "LBDrawn with scale SC0,5000,0,8214"+CHR$(3)
370 PRINT #1, "SP0;""
380 END
```

```
390 '
400 'draws a house using absolute coordinates
410 'main structure, roof, chimney, door,
420 'window frame, panes, address
430 PRINT #1, "PU500,500;PD;EA1500,100;"
440 PRINT #1, "PD1000,800,1500,500;"
450 PRINT #1, "PU1300,620;PD1300,710,1200,710,1200,680;"
460 PRINT #1, "PU700,100;PD700,400,900,400,900,100;"
470 PRINT #1, "PU1200,300;PD;EA1400,200;"
480 PRINT #1, "PU1300,200;PD1300,300;PU1200,250;"
490 PRINT #1, "PD1400,250;PU;SI.15,.25;L04;"
500 PRINT #1, "PU800,420;LB2721"+CHR$(3)
510 'restore program defaults
520 PRINT #1, "SI;L02;"
530 RETURN
```



Drawn with scale SC0, 5000, 0, 8214

C



P1 set to: -4200, 2300



P1 set to: -2300, 0



P1 set to: -4200, -2300



P1 set to: 2520, -4140



Original P1: -4200, -6900

Example 3: Using a Window

This program illustrates how a window restricts plotting to a specific area. The program first sets P1 and P2 to define an area on the left side of the paper, and then scales the area isotropically. After drawing the figures, the program moves only P1 to the right side of the paper (P2 tracks). The program then establishes a window to restrict plotting to the top of the paper so that one of the figures is within the window area and the other is outside of the window. After drawing only what is within the window, the program removes the window and labels the area.

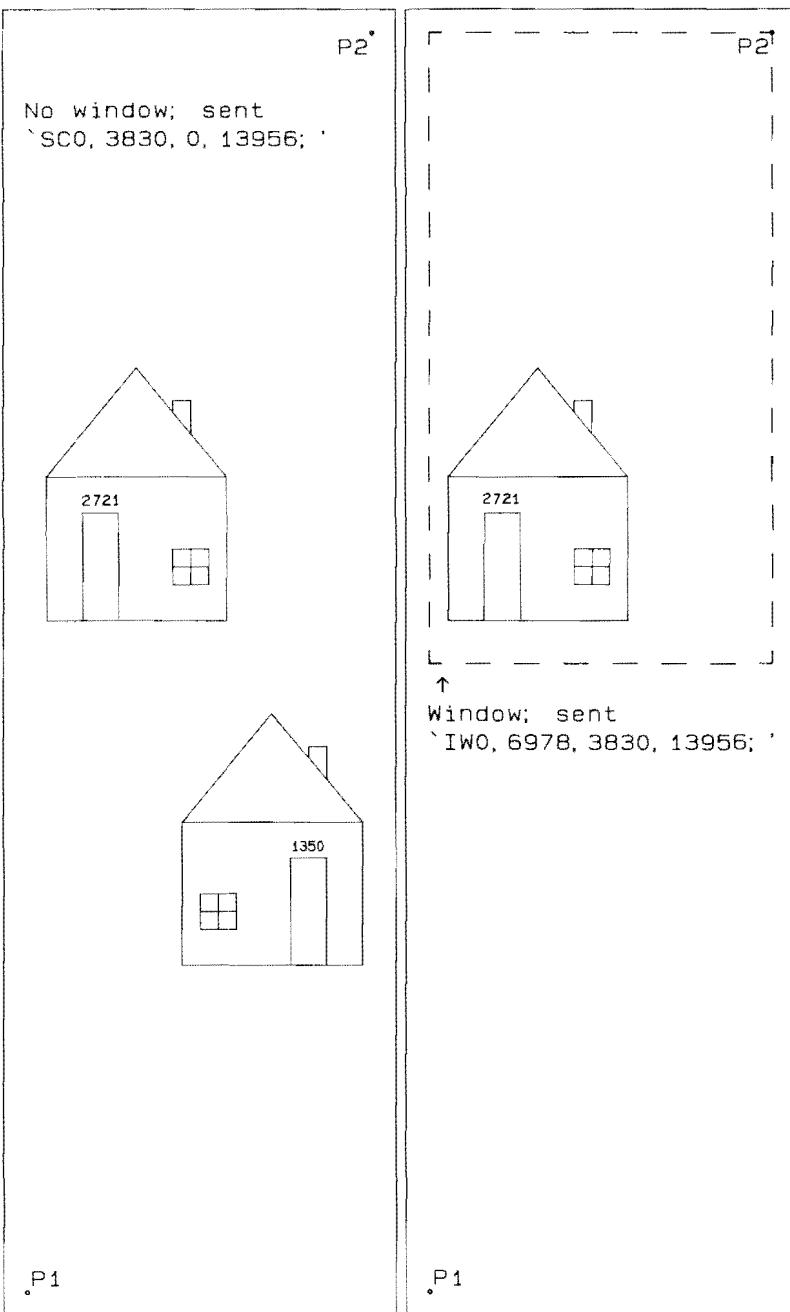
```
10  'Insert configuration statement here
15  'sets P1 and P2 to left side of paper
20  PRINT #1, "IN;IP-3950,-6650,-300,6650;""
25  '
35  'frames P1/P2 area;labels
40  PRINT #1, "SP1;PU-4200,-6900;EA-50,6900;""
50  PRINT #1, "PA-3950,-6650;CI20;L011;LBP1"+CHR$(3)
60  PRINT #1, "PA-300,6650;CI20;L019;LBP2"+CHR$(3)
70  PRINT #1, "PU-3950,6650;CP0,-3;L01;""
80  PRINT #1, "LBNo window; sent"+CHR$(3)
90  PRINT #1, "CP;LB*SC0,3830,0,13956;"+CHR$(3)
105 '
115 'scales area and draws figures
120 PRINT #1, "SC0,3830,0,13956;""
130 PRINT #1, "PA1915,7350;""
140 GOSUB 405
145 '
205 'resets P1 (P2 tracks), labels, frames area
210 PRINT #1, "IP300,-6650;SC;""
220 PRINT #1, "PU50,-6900;EA4200,6900;""
225 '
230 PRINT #1, "PU300,-6650;CI20;L011;LBP1"+CHR$(3)
240 PRINT #1, "PU3950,6650;CI20;L019;LBP2"+CHR$(3)
245 'sets window and draws figures
250 PRINT #1, "SC0,3830,0,13956;""
260 PRINT #1, "IW0,6978,3830,13956;""
270 PRINT #1, "PU1915,7350;""
280 GOSUB 405
285 '
295 'outlines window area
300 PRINT #1, "LT-2;PU0,6978;PD3830,6978,3830,13956;""
310 PRINT #1, "PD0,13956,0,6978;PU;LT;IW;"
```

```

C
315   '
325   'indicate the window
330 PRINT #1, "SC;PU300,0;CP.5,-1;CA5;SA;
      LBs "+CHR$(15)+CHR$(3)
340 PRINT #1, "CP;LBWindow; sent"+CHR$(3)
350 PRINT #1, "CP;LB;IW0,6978,3830,13956;'" +CHR$(3)
360 PRINT #1, "SP0;"'
370 END
380   '
390   'draws two houses
400   'drawing the left house
410 PRINT #1, "SP2;PR;PU-1700,1700;ER2000,-1600;"'
420 PRINT #1, "PD1000,1200,1000,-1200;"'
430 PRINT #1, "PU-400,480;PD0,360,-200,0,0,-120;"'
440 PRINT #1, "PU-1000,-2320;PD0,1200,400,0,0,-120;"'
450 PRINT #1, "PU600,800;ER400,-400;"'
460 PRINT #1, "PU200,-400;PD0,400;PU-200,-200;PD400,0;PU;"'
470   '
480   'label address and return to carriage return point
490 PRINT #1, "SI.15,.25;L04;PU-1200,680;
      LB2721"+CHR$(13)+CHR$(3)
505   '
515   'drawing the bottom house
520 PRINT #1, "PU900,-3500;PD;ER2000,-1600;"'
530 PRINT #1, "PD1000,1200,1000,-1200;"'
540 PRINT #1, "PU-400,480;PD0,360,-200,0,0,-120;"'
550 PRINT #1, "PU200,-2320;PD0,1200,-400,0,0,-1200;"'
560 PRINT #1, "PU-600,800;ER-400,-400;"'
570 PRINT #1, "PU-200,0;PD0,-400;PU-200,200;PD400,0;PU;"'
580 PRINT #1, "SI.15,.25;L04;PU800,680;LB1350"+CHR$(3)
585   '
595   'restore defaults
600 PRINT #1, "SI;L01;PA;SP1;"'
610 RETURN

```

C



C Example 4: Rotating a Plot

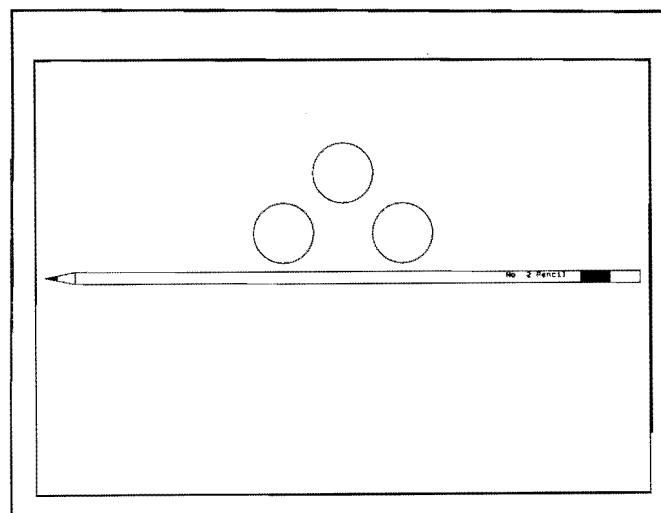
This sample plot is especially effective when drawn on C-size paper. The following program illustrates the effects of rotation on a plot drawn using absolute coordinate values. It also shows the effects of anisotropic scaling on circles.

The first figure, a pencil, fits along the X-axis when drawn unrotated. When the front-panel **ROTATE** button is pressed, the ends of the pencil are clipped. (You would observe the same effect if you sent (*RO90;IP*) to the plotter before drawing the pencil.)

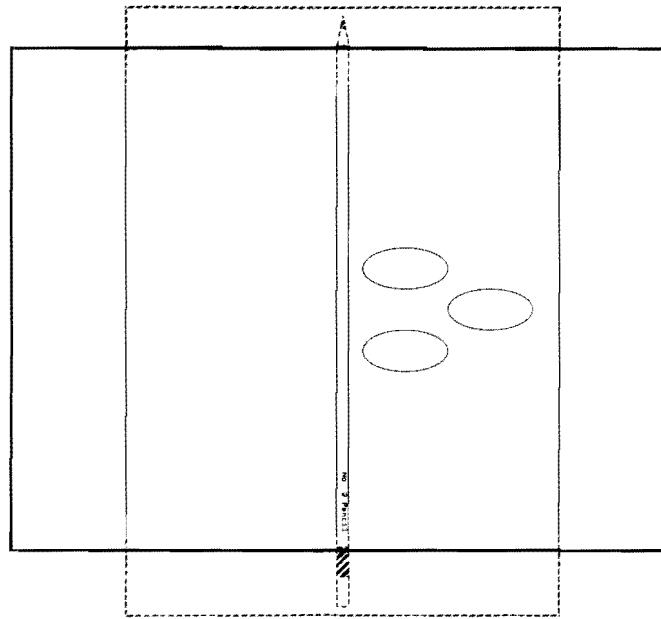
After drawing the pencil, the plotting area is scaled isotropically for unrotated P1 and P2 values. Since pressing the front-panel **ROTATE** button sets P1 and P2 within the hard-clip limits (the same effect as sending (*RO90;IP*)), a rotated plot with the same scaling is anisotropic. Anisotropic scaling will distort geometric shapes such as circles so that they are drawn as ellipses.

```
10  'Insert configuration statement here
20  PRINT #1, "IN;"
30  '
40  'display message to computer monitor
50  PRINT "If you want a rotated plot, press plotter's ROTATE
      button."
60  INPUT "Press the ENTER key on the computer to continue.",A$
70  '
80  'draw the pencil
90  PRINT #1, "SP1;PU-9800,0;""
100 PRINT #1, "PD-8800,200,9800,200,9800,-200,-8800,
      -200,-9800,0;""
110 PRINT #1, "PU;PM0;PD-9400,80,-9400,-80,-9800,0;PM2;FT;""
120 PRINT #1, "FP;PU-8800,200;PD-8800,-200;PU7800,200;""
130 PRINT #1, "PM0;PD8800,200,8800,-200,7800,-200,
      7800,100200;PU;PM2;""
140 PRINT #1, "FT3,40,0;FP;FT;""
150 PRINT #1, "SP2;PU5300,0;LBN0. 2 Pencil"+CHR$(3)
160 '
170 'draw a frame around plot
180 PRINT #1, "SP3;PU-10100,-7200;""
190 PRINT #1, "EA10100,7200;""
200 '
210 'isometric scale for unrotated P1/P2 values
220 PRINT #1, "SC0,9800,0,6900;SP4;""
230 '
240 'draw the circles
250 PRINT #1, "PU4900,5175;CI500;""
260 PRINT #1, "PU3900,4175;CI500;""
270 PRINT #1, "PU5900,4175;CI500;"
```

```
280
290 PRINT #1, "SP0;"
300 END
```



C



Example 5: FORTRAN and Pascal — AR Instruction

The following programs produce the example plot shown for the arc relative (AR) instruction. The GW BASIC program is shown first, followed by the FORTRAN and the Pascal programs. All three programs produce the same plot.

Use this example to help you get started with converting the program examples to FORTRAN or Pascal.

C

GW BASIC

```
10 OPEN "COM1:9600,N,8,1,RS,CS65535,DS,CD" AS #1
20 PRINT #1, "IN;SP1;PA10,10;PD;"
30 PRINT #1, "AR0,2000,80,25;AR2000,0,80;"
40 PRINT #1, "SP0;"
50 END
```

FORTRAN

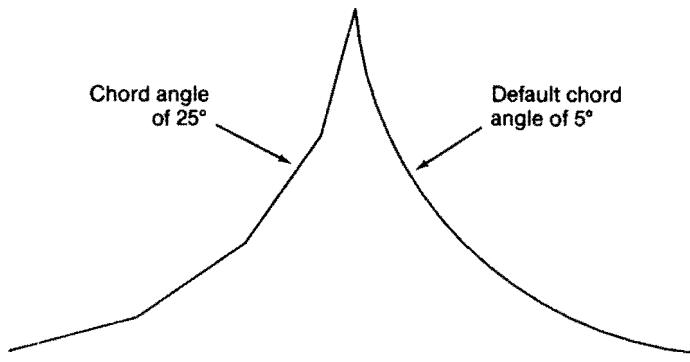
```
PROGRAM AR
INTEGER ESCAPE
ESCAPE = 27
WRITE (6,10) ESCAPE,ESCAPE
10 FORMAT (1X,1R1,".(“.P2:IN;SP1;PA10,10;PD;AR0,2000,
     80,25;”)
WRITE (6,20) ESCAPE
20 FORMAT (1X, “AR2000,0,80,SP0;”,1R1,“.Z”)
STOP
END
```

Pascal

```
PROGRAM AR (OUTPUT);
CONST
  esc = CHR(27);

  PROCEDURE Arcs;
  BEGIN
    Writeln (esc,'.('',esc,'.P2:IN;SP1;PA10,10;PD;');
    Writeln ('AR0,2000,80,25;AR2000,0,80;');
    Writeln ('SP0;',esc,'.Z');
  END; (* Arcs *)

  BEGIN
    Arcs;
    Writeln;
  END (* AR *).
```



Example 6: FORTRAN and Pascal — DI and LB Instruction

The following programs produce the example plot shown for the direction absolute (DI) instruction. The GW BASIC program is shown first, followed by the FORTRAN and the Pascal programs. All three programs produce the same plot.

Use these programs to help you convert program examples that use the label (LB) instruction to FORTRAN or Pascal.

C

GW BASIC

```
10 OPEN "COM1:9600,N,8,1,RS,CS65535,DS,CD" AS #1
20 PRINT #1, "IN;SP1;PA0,0;""
30 PRINT #1, "DI1,1;LB DIRECTION"+CHR$(13)+CHR$(3)
40 PRINT #1, "DI1,-1;LB DIRECTION"+CHR$(13)+CHR$(3)
50 PRINT #1, "DI-1,-1;LB DIRECTION"+CHR$(13)+CHR$(3)
60 PRINT #1, "DI-1,1;LB DIRECTION"+CHR$(13)+CHR$(3)
70 PRINT #1, "SP0;""
80 END
```

FORTRAN

```
PROGRAM DI
INTEGER ESCAPE,CR,ETX
ESCAPE = 27
CR = 13
ETX = 3
WRITE (6,10) ESCAPE,ESCAPE
10 FORMAT (1X,1R1,".(",1R1,".P2:IN;SP1;SI.3,.5;PA0,0;")
WRITE (6,20) CR,ETX
20 FORMAT (1X,"DI1,1;LB DIRECTION",1R1,1R1)
WRITE (6,30) CR,ETX
30 FORMAT (1X,"DI1,-1;LB DIRECTION",1R1,1R1)
WRITE (6,40) CR,ETX
40 FORMAT (1X,"DI-1,-1;LB DIRECTION",1R1,1R1)
WRITE (6,60) ESCAPE
60 FORMAT (1X,"SP0:",1R1,".Z")
STOP
END
```

Pascal

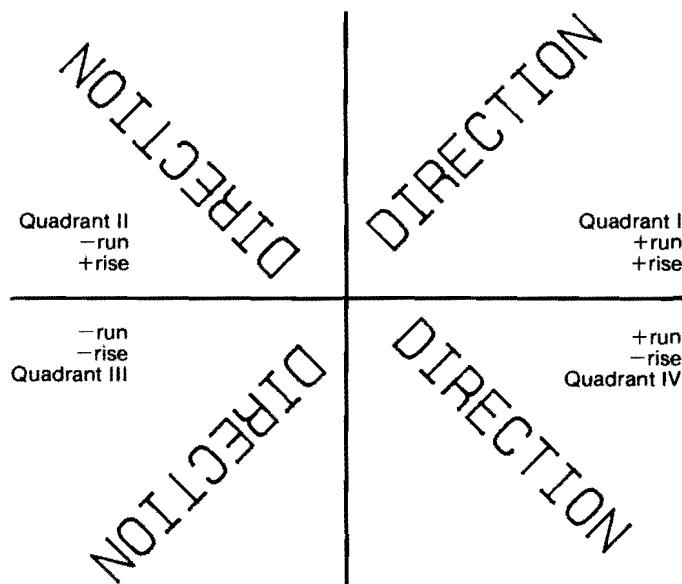
```

PROGRAM DI (OUTPUT)
CONST
  esc := CHR(27);
  cr := CHR(13);
  etx := CHR(3);

PROCEDURE LabelDirection;
BEGIN
  Writeln (esc,'.(,esc,'.P2:IN;SP1;SI.3,.5;PA0,0;');
  Writeln ('DI1,1;LB DIRECTION',cr,etx);
  Writeln ('DI1,-1;LB DIRECTION',cr,etx);
  Writeln ('DI-1,-1;LB DIRECTION',cr,etx);
  Writeln ('DI-1,1;LB DIRECTION',cr,etx);
END ; (*LabelDirection*)

BEGIN
  LabelDirection;
  Writeln;
END (* DI *).

```



Notes

C

D

Using Kanji

The optional Kanji character set includes sets 100 and 101. These sets are independent of each other. Set 100 contains only nonprinting control characters (ASCII codes 0 through 32); this set does not contain any printing characters. Set 101 contains nonprinting control characters (ASCII codes 0 through 32) and 3418 printing characters. The printing characters are arranged in eight distinct printing character groups. The character groups are as follows.

- **General graphic characters** — punctuation marks, arithmetic and logical symbols, and miscellaneous graphics characters.
- **Digits** — Arabic numerals from zero to nine.
- **Latin alphabet** — upper- and lowercase letters from A through Z.
- **Hiragana characters** — 83 Hiragana characters.
- **Katakana characters** — 86 Katakana characters.
- **Greek alphabet** — upper- and lowercase letters from Alpha through Omega.
- **Cyrillic alphabet** — upper- and lowercase letters from A through Ya.
- **Kanji characters** — 2965 Kanji characters.

In both sets 100 and 101, the Kanji control characters are the same as in the other character sets.

Accessing the Kanji Character Set

You designate either Kanji set for use as the standard or alternate character set just as you do other character sets: with the CS (Designate Standard Character Set) or CA (Designate Alternate Character Set) instructions. Of the two sets, only set 101 contains printing characters. For this reason, you will probably only want to use set 101 as either the standard or alternate character set.

You can access and use the nonprinting control characters in set 101 similarly as when labeling with other character sets. However, because of the quantity of printing characters in set 101, you must supply two pieces of information (i.e., two bytes) to access and draw any printing character. To draw printing characters, both bytes must be within the ASCII decimal value range 33 to 126. The first byte accesses the character group, the second byte accesses the printing character within the group. The following table lists the valid first byte values and the character group they access. Since only some of the character groups use all of the available 94 ASCII codes (33–126), the table also lists the valid second-byte ranges for the printing characters in that group.

First Byte Value	Group	Second Byte Value Range
33	General Graphics	33–126
34		33–46
35	Numbers	48–57
35	Latin alphabet	65–90, 97–122
36	Hiragana	33–115
37	Katakana	33–118
38	Greek alphabet	33–56, 65–88
39	Cyrillic alphabet	33–65, 81–113
48–78	Kanji	33–126
79		33–83

For example, the first printing character in the general graphics character group would be specified by its group character byte then the first byte of the valid printing character range; in this case (33, 33). The first printing character of the Cyrillic alphabet is (39, 33). The last printing character of the Kanji group is (79, 83).

If you specify two bytes within the 33–126 range for which there is no printing character, the plotter draws a substitution character. The substitution character is four dots set in a square configuration. All of the printing characters for set 101 are shown at the end of this appendix.

You can specify each byte using the BASIC CHR\$ function, or by using the corresponding keyboard character for that byte value. For example, when character set 101 is selected, the following examples access the same Kanji character.

```
PRINT #1, "LB";CHR$(65);CHR$(42);CHR$(3)
```

```
PRINT #1, "LBA*";CHR$(3)
```

 選

When labeling with Kanji characters, you can use the DV (Direction Vertical) instruction to label the characters vertically. In vertical mode, a carriage return moves the pen up to the carriage return point, instead of moving left to the carriage return point. A line feed, on the other hand, causes the pen to move to the left instead of down. Labeling in vertical mode, then, is from top to bottom, right to left.

The following illustrates carriage returns and line feeds in vertical mode. Note that the listing uses both methods accessing Kanji characters.

```
PRINT #1, "CS0;CA101;SA;DV1;"  
PRINT #1, "LBJ8;z"+CHR$(13)+CHR$(10)+CHR$(3)  
PRINT #1, "LB":CHR$(37)+CHR$(59):CHR$(37)+CHR$(67):  
    CHR$(37)+CHR$(72):CHR$(10)+CHR$(3)  
PRINT #1, "SS;LB101"+CHR$(3)
```

文
字
ト

1
0
1

Line feed only

Line feed carriage return

In the first column (at the far right), the first character is accessed using the characters “J8”; the second character is accessed by the characters “;z”. The characters in the second column are accessed using ASCII decimal values.

Using Kanji Characters in Symbol Mode

The SM (Symbol Mode) instruction has been modified for use with set 101. Usually the SM instruction lets you specify only a single character for use in symbol mode. When you are using character set 101 you specify two characters to properly define your character. Symbol mode allows you to define a single printing character to use as a symbol in geometric drawings or graphs. When you are not using character set 101, you can specify only one character byte for the symbol. If you specify more than one byte or character, only the first character byte is used.

In symbol mode, both of the character bytes you specify must be within the printing character range. However, note that printing character (33, 33) is a space, which terminates symbol mode. If either the first or second character is a printing character but the other character is a nonprinting control character (has a value of 32 or less), then symbol mode is terminated. If it is the second character that is a control character, then the plotter also generates an 'out of range' error.

Note that you cannot specify one character and then use a semicolon to terminate the SM instruction. Symbol mode will use the semicolon as the second byte that defines your symbol character.

Terminating Kanji Labels

The DT (Define Label Terminator) instruction has *not* been modified for character set 101. There is no two-byte form of the DT instruction. The default label terminator is the **ETX** (end of text) character (ASCII decimal code 3). Only ASCII codes 1–9, 11–26, 28–31, or the space character (ASCII decimal code 32) can be used as label terminators. You cannot use the **LF** character (decimal code 10) or the **ESC** character (decimal code 27). Nor can you use the semicolon (decimal code 59). If the DT instruction defines a printing character as a label terminator, that character terminates labels for all character sets but set 101; the designated terminator for set 101 is not altered.

The Kanji Character Set

The following pages show the characters in set 101. Note that the value of the first byte (which specifies a particular group of characters) is listed vertically along the side of the page. The second byte (which specifies a particular character in the group) is listed horizontally across the top of the page. The illustration below shows how the characters are arranged on the following pages. The number in the box is the page number, while the numbers along the top and the left side show the values listed on that page.

	33-44	45-56	57-68	69-80	81-92	93-104	105-116	117-126
33								
39	D-7	D-8	D-9	D-10	D-11	D-12	D-13	D-14
48								
53								
54	D-15	D-16	D-17	D-18	D-19	D-20	D-21	D-22
66								
67	D-23	D-24	D-25	D-26	D-27	D-28	D-29	D-30
79								

Decimal Value	SECOND BYTE												
	33	34	35	36	37	38	39	40	41	42	43	44	
FIRST BYTE	33	、	。	,	.	‘	:	；	？	！	”	‘	
	34	◆	□	■	△	▲	▽	▼	※	〒	→	←	↑
	35	．	．	．	．	．	．	．	．	．	．	．	
	36	あ	お	い	い	う	う	え	え	お	お	か	が
	37	ア	ア	イ	イ	ウ	ウ	エ	エ	オ	オ	カ	ガ
	38	А	В	Г	Д	Е	З	Н	Ө	І	К	Л	М
	39	А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К
	48	亜	啞	娃	阿	哀	愛	挨	始	逢	葵	茜	穂
	49	院	陰	隱	韻	时	右	宇	鳥	羽	迂	雨	卯
	50	押	旺	横	欧	殴	王	翁	懊	鳶	鷗	黄	岡
	51	魁	海	械	海	灰	界	皆	繪	芥	蟹	開	階
	52	粥	刈	茹	瓦	乾	侃	冠	寒	刊	勘	勸	卷
	53	機	帰	毅	氣	汽	畿	祈	季	稀	紀	徽	規

Decimal Value	SECOND BYTE											
	45	46	47	48	49	50	51	52	53	54	55	56
FIRST BYTE	33	‘	‘	”	^	—	、	、	、	、	、	全
	34	↓	=									
	35	.	.	.	0	1	2	3	4	5	6	7
	36	き	ぎ	く	ぐ	け	げ	こ	ご	さ	ざ	し
	37	ヰ	ヰ	ク	ヰ	ケ	ヰ	コ	ヰ	サ	ヰ	シ
	38	N	ニ	〇	ニ	P	Σ	T	Τ	Φ	X	Ψ
	39	ル	M	H	〇	ル	P	C	T	Υ	Φ	Χ
	48	悪	握	渥	旭	葦	芦	醪	梓	圧	斡	扱
	49	鶴	観	丑	碓	臼	渦	嘘	唄	蔚	饅	姥
	50	沖	荻	億	屋	憶	臆	桶	牡	乙	掩	卸
	51	貝	凱	効	外	亥	害	崖	慨	概	涯	得
	52	喚	堪	姦	完	官	寛	干	幹	患	慣	憾
	53	記	貴	起	軌	輝	飢	騎	鬼	龜	偽	儀

Decimal Value	SECOND BYTE											
	57	58	59	60	61	62	63	64	65	66	67	68
FIRST BYTE	33	タ	ノ	ー	ー	ー	/	\	~	।	।	…
	34
	35	9	A	B	C	D
	36	す	ず	せ	ぜ	そ	ぞ	た	だ	ち	ぢ	つ
	37	ス	ズ	セ	ゼ	リ	ゾ	タ	ダ	チ	ヂ	ツ
	38	α	β	γ	δ
	39	四	山	山	乙	巳	巳	零	Я	.	.	.
	48	姐	虹	飴	絢	綾	鮎	或	栗	拾	安	庵
	49	鰐	浦	瓜	閨	噂	云	運	雲	莊	餉	嘗
	50	溫	穩	音	下	化	板	何	伽	価	佳	加
	51	街	該	鎧	鼓	涅	馨	蛙	垣	柿	鷄	鈎
	52	換	敢	柑	桓	棺	款	歡	汗	漢	潤	灌
	53	宜	戲	技	攬	欺	犧	疑	祇	義	儀	誼

Decimal Value	SECOND BYTE											
	69	70	71	72	73	74	75	76	77	78	79	80
33	..	'	"	"	()	()	[]	{	
34	
35	E	F	G	H	J	J	K	L	M	N	O	P
36	づ	て	で	と	ど	な	に	ぬ	ね	の	は	ば
37	ヅ	テ	デ	ト	ド	ナ	ニ	ヌ	ネ	ノ	ハ	バ
FIRST BYTE	38	ε	ç	η	θ	ı	k	ʌ	μ	v	ξ	o
	39	
	48	暗	案	闇	鞍	杏	以	伊	位	依	偉	囂
	49	嬰	影	映	曳	榮	永	泳	洩	瑛	盈	頴
	50	嘉	夏	嫁	家	寡	科	暇	果	架	歌	火
	51	麻	各	廓	括	攬	格	核	殼	獲	確	穫
	52	甘	監	看	竿	管	簡	緩	缶	輸	肝	艦
	53	掬	菊	鞠	吉	屹	喫	桔	橘	詰	砧	杵

Decimal Value	SECOND BYTE											
	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12
33)	<)	{)	[]	[]	[]	+
34
35	Q	R	S	T	U	V	W	X	Y	Z	.	.
36	ぱ	ひ	び	ぴ	ふ	ぶ	ぶ	へ	べ	べ	ほ	ば
37	ぱ	ヒ	ビ	ピ	フ	ブ	プ	ヘ	ベ	ベ	ホ	ボ
FIRST BYTE	ρ	σ	τ	υ	ϕ	χ	ψ	ω
	а	б	в	г	д	е	ё	ж	з	и	й	к
39	委	威	尉	惟	意	慰	易	椅	為	畏	異	移
48	莫	衛	詠	銳	液	疫	益	顯	悅	謁	越	閱
49	珂	禍	禾	稼	箇	花	苟	茄	荷	華	菓	蝦
50	角	赫	較	郭	閣	隔	革	学	岳	樂	額	顎
51	觀	諫	貫	還	鑑	間	閑	閨	陷	韓	館	館
52	却	客	脚	虐	逆	丘	久	仇	休	及	吸	宮
53												

D

Decimal Value	SECOND BYTE											
	93	94	95	96	97	98	99	100	101	102	103	104
FIRST BYTE	33	-	±	X	÷	=	≠	<	>	≤	≥	∞
	34
	35	a	b	c	d	e	f	g
	36	ぼ	ま	み	む	め	も	や	や	ゆ	よ	よ
	37	ボ	マ	ミ	ム	メ	モ	ヤ	ヤ	ユ	ヨ	ヨ
	38
	39	ル	M	H	0	ル	p	c	T	y	中	x
	48	維	緯	胃	萎	衣	謂	違	遺	医	井	亥
	49	櫻	厭	円	園	壇	奄	宴	延	怨	掩	援
	50	課	暉	貨	迦	過	霞	蚊	俄	峨	我	牙
	51	掛	笠	櫻	樞	棍	鰐	漏	割	喝	恰	括
	52	丸	含	岸	巖	玩	癌	眼	岩	翫	匱	頑
	53	弓	急	救	朽	求	汲	泣	灸	球	究	窮

Decimal Value	SECOND BYTE											
	105	106	107	108	109	110	111	112	113	114	115	116
33	♂ ♀ ‘ ’ “ ” ℃ ¥ \$ ¢ £ % #	．	．	．	．	．	．	．	．	．	．	
34	．	．	．	．	．	．	．	．	．	．	．	
35	ิ จ ក ล ມ ນ ອ ປ ດ ຟ ສ ຕ	．	．	．	．	．	．	．	．	．	．	
36	້ ລ ຮ ລ ຮ ວ ວ ວ ວ ວ ວ ວ ວ	້	ໍ	່	໌	໌	໌	໌	໌	໌	໌	
37	໌ ລ ຮ ແ ລ ວ ວ ວ ວ ວ ວ ວ	໌	ໍ	່	໌	໌	໌	໌	໌	໌	໌	
FIRST BYTE	38	．	．	．	．	．	．	．	．	．	．	
	39	ҹ ҹ ҹ ҹ ҹ ҹ ҹ ҹ ҹ ҹ ҹ ҹ	ҹ	ҹ	ҹ	ҹ	ҹ	ҹ	ҹ	ҹ	ҹ	
	40	育 郁 磨 一 吉 溢 逸 稻 芦 蘭 分	育	郁	磨	一	吉	溢	逸	稻	芦	蘭
	41	演 炎 烟 燕 猿 缘 艳 范 菜 遠 钝	演	炎	烟	燕	猿	缘	艳	范	菜	远
	42	臥 苞 蛾 賀 雅 餓 駕 介 会 解 回 塊	臥	苞	蛾	賀	雅	餓	駕	介	会	解
	43	渴 渑 葛 褐 輜 且 鰯 叶 桃 樺 薄 株	渴	渙	葛	褐	𦵹	且	鰯	叶	桃	桦
	44	顔 願 企 伎 危 喜 器 基 奇 嬉 寄 嫩	顔	願	企	伎	危	喜	器	基	奇	嬉
	45	級 紛 紿 紿 日 牛 去 居 巨 拒 売 桂	級	紛	紿	紿	日	牛	去	居	巨	拒
	46	渠	渠	渠	渠	渠	渠	渠	渠	渠	渠	

D

Decimal Value	SECOND BYTE									
	117	118	119	120	121	122	123	124	125	126
33	&	*	@	§	☆	★	○	◊	○	◊
34
35	U	V	W	X	y	Z
36
37	力	ㄎ
FIRST BYTE

48	印	匱	員	因	姻	引	飲	淫	削	蔭
49	鶯	塙	於	汚	甥	凹	央	奧	往	忄
50	壞	廻	快	怪	悔	恢	懷	戒	拐	改
51	兜	寵	蒲	釜	鑊	惱	鴨	栢	茅	葦
52	希	幾	忌	揮	机	旗	既	期	棋	棄
53	虛	許	距	鋸	漁	禦	魚	亨	享	京

Decimal Value	SECOND BYTE											
	33	34	35	36	37	38	39	40	41	42	43	44
54	供	俠	喬	児	競	共	凶	協	匡	卿	叫	喬
55	掘	窟	沓	靴	纏	窪	熊	隈	采	栗	繅	桑
56	檢	權	牽	犬	獻	研	覩	絹	県	肩	見	謙
57	后	喉	坑	垢	好	孔	孝	宏	工	巧	巷	幸
58	此	頃	今	困	坤	墾	婚	恨	貌	昏	昆	根
FIRST BYTE	察	搜	撮	擦	札	殺	薩	雜	臯	鰐	捌	鋸
	次	滋	治	爾	璽	痔	磁	示	而	耳	自	蒔
	宗	就	州	修	愁	拾	洲	秀	秋	終	縕	畧
	勝	匠	升	召	哨	商	唱	嘗	獎	妾	姐	窮
	拭	植	殖	燭	織	職	色	触	食	蝕	辱	尻
	澄	招	寸	世	灘	畝	是	淒	制	勢	姓	征
	纖	羨	腺	舛	船	薦	詮	賤	踐	選	遷	錢
	藏	藏	贈	造	促	側	則	即	息	捉	束	測

Decimal Value	SECOND BYTE											
	45	46	47	48	49	50	51	52	53	54	55	56
54	境	峽	強	彊	怯	恐	恭	挾	教	橋	況	狂
55	鍼	勲	君	薰	訓	群	軍	郡	卦	袈	祁	係
56	賢	軒	遺	鍵	險	顛	驗	檢	元	原	敵	幻
57	広	庚	康	弘	桓	慌	抗	拘	控	攻	昂	晃
58	樞	混	痕	紺	艮	魂	些	佐	叉	唆	嵯	左
FIRST BYTE	敏	皿	迺	三	傘	參	山	慘	撒	散	棧	燦
	辞	沵	鹿	式	讖	鴟	竺	軸	穴	秉	七	叱
	臭	舟	蒐	衆	叢	讐	蹴	輯	週	眚	酬	集
	將	小	少	尚	庄	床	廁	彰	承	抄	招	掌
	伸	信	侵	唇	娠	寢	審	心	慎	振	新	晋
	性	成	政	整	星	晴	棲	栖	正	清	牲	生
	銑	閃	鮮	前	善	漸	然	全	禪	繕	膳	糧
	足	速	俗	属	賊	族	続	卒	袖	其	擗	存

Decimal Value	SECOND BYTE											
	57	58	59	60	61	62	63	64	65	66	67	68
FIRST BYTE	54	狹	矯	胸	脅	興	蕃	鄉	鏡	響	饗	驚
	55	傾	刑	兄	啓	圭	珪	型	契	形	徑	惠
	56	弦	減	源	玄	現	絃	舷	言	謬	限	乎
	57	更	杭	校	梗	構	江	洪	浩	港	溝	甲
	58	差	查	沙	瑳	砂	詐	鎖	裟	坐	座	挫
	59	珊	產	算	纂	蚕	讚	贊	酸	餐	斬	暫
	60	鞠	失	嫉	室	悉	濕	漆	疾	質	寔	篠
	61	醜	什	住	充	十	從	戎	柔	汁	沒	獸
	62	捷	昇	昌	昭	晶	松	梢	樟	樵	沼	消
	63	森	棟	浸	深	申	疹	真	神	秦	紳	臣
	64	盛	精	聖	声	製	西	誠	誓	請	逝	醒
	65	增	塑	姐	措	曾	楚	俎	疏	疎	礎	祖
	66	孫	尊	損	村	遜	他	多	太	汰	訖	墮

FIRST BYTE	SECOND BYTE												
	69	70	71	72	73	74	75	76	77	78	79	80	
	54	凝	堯	曉	業	局	曲	極	玉	桐	糸	僅	勤
	55	慧	憩	揭	携	敬	景	桂	溪	畦	稽	系	經
	56	古	呼	固	姑	孤	己	庫	弧	戶	故	枯	湖
	57	硬	稿	糠	紅	纖	紋	綱	耕	考	肯	肱	腔
	58	准	再	最	哉	塞	妻	宰	彩	才	採	裁	歲
	59	社	仔	同	使	刺	司	史	嗣	四	士	始	姊
	60	患	柴	芝	屢	藥	縞	舍	写	射	捨	赦	斜
	61	重	銚	叔	夙	宿	漱	祝	縮	肅	塾	熟	出
	62	湘	燒	焦	照	症	省	硝	礁	祥	称	章	笑
	63	薪	親	診	身	辛	進	針	震	人	仁	刃	塵
	64	靜	吝	稅	貽	隻	席	惜	戚	斥	昔	析	石
	65	租	粗	素	組	蘇	訴	阻	邇	鼠	僧	創	双
	66	妥	惰	打	杞	耽	精	陀	默	驛	体	堆	对

Decimal Value	SECOND BYTE											
	81	82	83	84	85	86	87	88	89	90	91	92
54	均	巾	錦	斤	欣	欽	琴	禁	禽	筋	累	芹
55	継	繫	罪	莖	荆	熒	計	詔	警	輕	頸	鵝
56	狐	糊	袴	股	胡	蘋	虎	誘	跨	鉤	雇	顧
57	膏	航	荒	行	衡	講	貢	購	郊	酵	鉛	礪
58	濟	災	采	犀	碎	砦	祭	齋	細	菜	裁	載
FIRST BYTE	姿	子	屍	市	師	志	思	指	支	救	斯	施
59	煮	社	紗	者	謝	車	遮	蛇	邪	借	勺	尺
60	術	述	俊	峻	春	瞬	竣	舜	駿	准	循	旬
61	粧	紹	肖	菖	蔣	蕉	衝	裳	訟	証	詔	詳
62	壬	尋	甚	尗	腎	訊	迅	陣	鞞	筭	諷	須
63	積	籍	績	脊	責	赤	跡	蹟	碩	切	拙	接
64	叢	倉	喪	壯	奏	爽	宋	層	迺	惄	想	搜
65	耐	岱	岱	帶	待	怠	態	戴	替	泰	帶	胎
66												

Decimal Value	SECOND BYTE											
	93	94	95	96	97	98	99	100	101	102	103	104
54	菌	汾	襟	謹	近	金	吟	銀	九	俱	句	区
55	芸	迎	鯨	劇	戟	擊	激	隙	桁	傑	欠	決
56	鼓	五	互	伍	午	吳	吾	娛	後	御	悟	梧
57	鋼	閻	降	頃	香	高	鴻	剛	劫	号	合	壕
58	際	剝	在	材	罪	財	汎	扳	阪	堺	榦	肴
FIRST BYTE	旨	枝	止	死	氏	獅	祉	私	糸	紙	紫	肢
	杓	灼	爵	酌	釀	錫	若	寂	弱	惹	主	取
	楯	殉	淳	準	潤	盾	純	巡	遵	醇	順	廸
	象	嘗	醬	鉢	鍾	鐘	障	鞞	上	丈	永	秉
	醉	罔	厨	逗	吹	垂	帥	推	水	炊	睡	粧
	搃	折	設	窃	節	說	雪	絕	舌	蟬	仙	先
	掃	插	搔	操	早	曹	巢	槍	槽	漕	燥	爭
	苔	袋	貸	退	逮	隊	黨	觸	代	台	大	第

Decimal Value	SECOND BYTE											
	105	106	107	108	109	110	111	112	113	114	115	116
54	狗	珮	矩	苦	軀	驅	駢	駟	具	愚	虞	瞷
55	潔	穴	結	血	訣	月	件	僕	倦	健	兼	券
56	檜	瑚	碁	語	誤	護	醡	乞	鯉	交	校	侯
57	拷	濛	豪	轟	翫	克	刻	告	國	穀	酷	鵠
58	咲	崎	崎	琦	鷺	作	削	昨	搾	昨	朔	柵
FIRST BYTE	脂	至	視	詞	詩	試	誌	諮	資	賜	雌	餉
	守	手	朱	殊	狩	珠	種	腫	趣	酒	首	儒
	初	所	暑	曙	渚	庶	緒	署	書	薯	諸	
	冗	剝	城	場	巖	嫗	常	情	擾	条	杖	淨
	翠	衰	遂	醉	錐	錘	隨	瑞	覩	崇	嵩	數
	干	占	宣	專	尖	川	戰	扇	撰	栓	梅	泉
	瘦	相	窓	糟	縑	綜	驗	草	莊	葬	蒼	藻
	醍	題	鷹	滄	瀧	早	隙	宅	托	抿	拓	沵

D

Decimal Value	SECOND BYTE									
	117	118	119	120	121	122	123	124	125	126
FIRST BYTE	空	偶	寓	遇	隠	串	櫛	剣	肩	屈
	劍	喧	圈	堅	嫌	建	憲	懸	拳	捲
	候	倅	光	公	功	効	勾	厚	口	向
	黒	獄	癡	腰	飢	忽	惚	邑	狹	込
	窄	策	索	錯	櫻	鮭	笪	匙	冊	刷
	齒	事	似	侍	兒	字	寺	慈	持	時
	受	呪	寿	授	樹	綬	需	囚	収	周
	助	叙	文	序	徐	恕	鋤	除	傷	嘗
	狀	畠	穢	蒸	讓	釀	銳	ழ	植	節
	极	趨	難	据	杉	柵	菅	頗	雀	裾
	浅	洗	染	潜	煎	煽	旛	穿	箭	線
	装	走	送	遭	鎗	霜	駿	像	增	憎
	濯	琢	託	鐸	潤	諾	苴	廁	鶴	只

Decimal Value	SECOND BYTE											
	33	34	35	36	37	38	39	40	41	42	43	44
67	卯	但	達	辰	奪	脫	巽	豎	廸	櫛	谷	狸
68	帖	帳	庁	弔	張	彫	徵	懲	挑	暢	朝	潮
69	邸	鄭	釘	鼎	泥	摘	跼	敵	滴	的	笛	適
70	董	蕩	藤	討	瞻	豆	踏	逃	透	鎧	陶	頭
71	如	尿	菲	任	妊	忍	謂	濡	穢	祢	寧	葱
FIRST BYTE	72	函	箱	俗	箸	肇	筭	櫨	幡	肌	畠	八
	73	鼻	柊	稗	匹	疋	鬚	彥	膝	菱	肘	弼
	74	福	腹	複	覆	淵	弗	払	沸	仏	物	勑
	75	法	泡	烹	砲	縫	胞	芳	萌	蓬	蜂	褒
	76	漫	蔓	味	未	魅	日	箕	岬	密	蜜	湊
	77	諭	輸	唯	佑	優	勇	友	宥	幽	悠	憂
	78	癆	裏	裡	里	離	陸	律	率	立	葎	掠
	79	蓮	連	鍊	呂	魯	榦	炉	路	路	露	勞
												妻

Decimal Value	SECOND BYTE											
	45	46	47	48	49	50	51	52	53	54	55	56
67	鷗	樽	誰	丹	单	謨	迢	扱	探	日	歎	淡
68	牒	町	眺	聰	賑	腸	蝶	調	謀	超	跳	銚
69	鎧	湯	哲	徹	撤	轍	迭	鉄	典	填	天	展
70	騰	鬪	鬪	動	同	堂	導	橦	撞	洞	墮	童
71	猫	熱	年	念	捨	燃	粘	乃	廻	之	埜	塙
FIRST BYTE	鉢	發	發	駿	髮	伐	罰	拔	筏	閥	鳩	嘶
	畢	筆	逼	檜	姬	媛	紐	百	謬	俵	彪	標
	叻	曠	填	幘	扮	焚	奮	粉	糞	紛	雰	文
	豈	邦	鋒	飽	鳳	鶴	乏	亡	傍	剖	坊	妨
	稔	脈	妙	耗	民	眠	務	夢	無	牟	矛	霧
	有	柚	湧	涌	猶	猷	由	祐	裕	誘	遊	邑
	劉	流	溜	琉	留	硫	粒	隆	竜	龍	侐	慮
	鹿	弄	朗	樓	榔	浪	漏	牢	狼	籠	老	鼈

Decimal Value	SECOND BYTE											
	57	58	59	60	61	62	63	64	65	66	67	68
67	湛	巖	短	端	簾	綻	耽	胆	蛋	誕	鍛	団
68	長	頂	鳥	勅	彷	直	暎	沈	珍	賃	鎮	陳
69	店	添	纏	甜	貼	転	顛	点	云	殿	殿	田
70	洞	萄	道	銅	峠	鴉	匿	得	德	遺	特	督
71	囊	惄	濃	納	能	腦	曇	農	覩	蚤	巴	把
FIRST BYTE	72	搞	蛤	隼	伴	判	半	反	叛	帆	搬	斑
	73	冰	漂	飄	票	表	詳	豹	廟	描	病	秒
	74	聞	內	併	兵	屏	幣	平	弊	柄	並	蔽
	75	帽	忘	忙	房	暴	望	某	棒	冒	紡	肪
	76	鶴	惊	婿	娘	冥	名	命	明	盟	迷	銘
	77	郵	雄	融	夕	予	余	与	誓	輿	預	傭
	78	旅	虜	了	亮	僚	面	凌	寮	料	染	涼
	79	纏	郎	六	蠶	祿	助	錄	論	倭	和	話
											歪	

Decimal Value	SECOND BYTE											
	69	70	71	72	73	74	75	76	77	78	79	80
67	壇	彈	斷	暖	檀	段	男	談	值	知	地	弛
68	建	墜	椎	椎	追	鎗	痛	通	塚	相	撻	楓
69	電	兔	吐	堵	塗	妬	屠	徒	斗	杜	渡	登
70	禿	篤	毒	独	謁	栎	橡	凸	突	櫛	届	鳶
71	播	霸	杷	波	派	琶	破	婆	罵	芭	馬	俳
FIRST BYTE	汎	汎	版	犯	班	畔	繁	般	藩	販	範	采
72	鋸	鋟	蒜	蛭	鰐	品	彬	斌	浜	瀨	賓	貧
73	陛	米	貞	僻	壁	癖	碧	別	鬯	蔑	匱	偏
74	謀	貌	貿	鋒	防	狀	頰	北	僕	卜	墨	撲
75	姪	牝	漱	免	棉	綿	纏	面	麵	摸	模	茂
76	妖	容	庸	揚	搖	擁	曜	楊	樣	洋	溶	熔
77	療	瞭	稜	糧	良	諒	遼	量	陵	領	力	綠
78	眞	脇	惑	柙	鷺	瓦	亘	鷄	託	藁	厥	槐
79												

Decimal Value	SECOND BYTE											
	81	82	83	84	85	86	87	88	89	90	91	92
67	聰	智	池	痴	稚	置	致	嫋	遲	馳	築	畜
68	佃	漬	柘	辻	葛	綴	鍔	椿	潰	坪	壘	墉
69	菟	賭	途	都	鍛	砥	礪	努	度	土	奴	怒
70	苦	寅	酉	灝	頤	屯	惇	敦	油	豚	遁	頓
71	𡇠	𢂑	排	敗	杯	盃	牌	背	肺	輩	配	倍
FIRST BYTE	72	煩	頒	飯	挽	晚	番	盤	磐	蕃	蛮	匪
	73	頻	敏	瓶	不	付	埠	夫	婦	富	富	布
	74	变	片	篇	編	迥	返	遍	便	勉	免	弁
	75	朴	牧	陸	穆	釦	勃	沒	殆	堦	幌	奔
	76	妄	孟	毛	猛	盲	網	耗	蒙	儲	木	默
	77	用	窯	羊	耀	葉	蓉	要	譖	踊	遙	陽
	78	倫	厘	林	淋	熒	琳	臨	輸	隣	鱗	璫
	79	灣	碗									

Decimal Value	SECOND BYTE											
	93	94	95	96	97	98	99	100	101	102	103	104
67	竹	筭	蓄	逐	秩	窒	荼	嫡	着	中	仲	宙
68	紬	爪	吊	釣	鶴	亭	低	停	煩	剃	貞	呈
69	倒	党	冬	凍	刀	唐	塔	塘	套	宕	島	鳩
70	呑	雲	鈍	奈	那	內	乍	𠂇	難	謎	灘	捺
71	培	媒	梅	模	煤	狽	買	壳	賠	陪	這	蠅
FIRST BYTE	72	否	妃	庇	彼	悲	扉	批	披	斐	比	洮
	73	怖	扶	敷	斧	普	浮	父	符	腐	膚	芙
	74	保	舖	鋪	圃	捕	步	甫	補	輔	穗	募
	75	翻	凡	盆	摩	磨	魔	麻	埋	妹	昧	枚
	76	杰	勿	餅	尤	戾	糲	貴	問	闕	紋	門
	77	慾	抑	欲	沃	浴	翌	翼	淀	羅	螺	來
	78	墨	淚	累	類	令	伶	冽	冷	励	嶺	恰
	79											

Decimal Value	SECOND BYTE											
	105	106	107	108	109	110	111	112	113	114	115	116
67	忠	抽	昌	柱	注	虫	衷	註	酌	鑄	駐	櫓
68	堤	定	帝	底	庭	廷	弟	悌	抵	挺	提	梯
69	棹	投	搭	東	桃	櫓	棟	盜	淘	湯	濤	灯
70	銅	檣	馴	繩	駁	南	楠	軟	難	汝	二	尼
71	秤	矧	萩	伯	剝	博	拍	柏	泊	白	箔	粕
FIRST BYTE	皮	碑	秘	緋	罷	肥	被	誹	費	避	非	飛
72	負	賦	赴	阜	附	侮	撫	武	舞	葡	蕪	部
73	慕	戌	暮	母	簿	菩	倣	俸	包	呆	報	奉
74	哩	檳	幕	膜	枕	鮑	柵	鱠	槲	亦	侯	又
75	也	冶	夜	爺	耶	野	弥	矢	厄	役	約	藥
76	萊	賴	雷	洛	絡	落	酩	亂	卵	嵐	欄	鑑
77	礼	芥	鈴	隸	零	靈	麗	齡	曆	歷	列	劣
78												
79												

Decimal Value	SECOND BYTE									
	117	118	119	120	121	122	123	124	125	126
67	猪	猪	苧	著	貯	丁	兆	凋	牒	寵
68	汀	碇	禎	程	綺	艇	訂	諦	蹄	遙
69	燈	当	痘	禱	等	答	筍	糖	統	到
70	式	邇	勾	賑	肉	虹	廿	日	乳	入
71	舶	薄	迫	曝	漠	爆	縛	莫	駁	麦
FIRST BYTE	樞	簸	備	尾	微	枇	毘	毘	眉	美
	封	楓	風	葺	路	伏	副	復	幅	服
74	宝	峰	峯	崩	危	抱	捧	放	方	朋
75	抹	末	沫	迄	盡	蘭	磨	万	慢	滿
76	訛	躍	靖	柳	藪	鎗	愉	愈	油	癒
77	藍	蘭	覽	利	吏	履	李	梨	理	璃
78	烈	烈	廉	恋	憐	漣	煉	簾	練	聯
79										

Glossary

Absolute Coordinates	A pair of values that specify the absolute location of a point with respect to the coordinate system origin (0 , 0).
Absolute Plotting	Plotting to a point whose location is specified with respect to the coordinate system origin (0 , 0).
Acceleration	The rate at which the pen attains its maximum velocity (measured in g's or metres per second squared).
Address	The address specifies the plotter's location on the HP-IB (IEEE-488) interface cable (bus).
Alternate Character Set	A character set other than a device's default (or standard) character set. Alternate character sets are built into a device and can be accessed by using the CS, CA, SS, or SA instructions.
Arc	A portion of the circumference of a circle.
Arc Font	A character font with a programmable contour smoothness.
ASCII	American Standard Code for Information Interchange. A 7-bit code representing character data such as letters, punctuation, symbols, and control characters; includes an eighth bit which can be used for parity. Used by many computers and peripheral devices.
Baud Rate	For an RS-232-C interface, the data transmission rate between the computer and the plotter.
Bit	Binary digit; a bit represents an 'on' or 'off' electrical condition and is the smallest piece of information a computer can handle.

Block I/O Error Checking	A data verification technique used with RS-232-C. It uses the ESC. @ and ESC. E instructions to check for transmission errors in a block of data.
Buffer	A part (or parts) of the computer or plotter's memory where data is held until it can be processed. Usually refers to an area reserved for I/O operations.
Bus	Short for HP-IB (IEEE-488) interface.
Bypass	RS-232-C only. An operating mode that controls when a peripheral can receive and process instructions. In Bypass ON , all instructions received are ignored except the programmed-on instruction. Bypass ON is equivalent to programmed off. In Bypass OFF , all instructions are received and processed. Bypass OFF is equivalent to programmed on.
Byte	Eight bits; the size of a computer word. Used by ASCII binary code to represent alphanumeric characters.
Carriage Return	A non-printing ASCII character (CR) that moves the plotter pen back to the carriage-return point (usually the beginning of the line) while in label mode.
Carriage-Return Point	The point (usually the beginning of the line) the pen moves to when the plotter receives a carriage return (while in label mode).
Character	A letter, digit, or some other symbol used in organizing, controlling, or representing data.
Character Origin	The lower-left corner of a character (i.e., the lower-left corner of the character plot cell).
Character Plot Cell	The area in which characters are drawn. The character plot cell is one space wide by one line high. (Variable-space fonts use an 'average' character plot cell.) The character occupies the lower-left portion of the cell, so that there is a blank area to the right and above the character.
Character Set	A group of characters, each of which is defined by a unique ASCII decimal code. Typically, a character set contains related characters, such as a character set composed of math symbols, or characters in a foreign language.

Character String	A sequence of ASCII characters, such as this sentence. <i>See also</i> literal string.
Chord	A straight line joining two points on an arc or on the circumference of a circle.
Chord Angle	The allowable deviation from a perfectly smooth circle or arc. The chord angle determines the number of chords (and thus the smoothness) used to draw a circle or an arc.
Clipping	Restricting plotting to a rectangular portion of the plotting area.
Communication	Data exchange between two or more devices.
Configuration	The way in which computer equipment is interconnected and set up to operate as a system.
Constant	A number with a fixed value. The coordinates of the absolute point (10, 20) are constants.
Control Character	A character that starts, modifies, or stops computer or peripheral operation instead of printing a character or symbol. Sometimes referred to as nonprinting control characters.
Cross-Hatch	A fill type that consists of one set of parallel lines drawn at a 90-degree angle to another set of parallel lines.
Current Units	Plotter units (if scaling is off) or user units (if scaling is on).
Debug	To find and correct mistakes in a computer program.
Decimal Code	The decimal equivalent value of an ASCII character. For example, the character 'A' has the ASCII binary code value 01000001 and the decimal code value 65.
Default	A value or condition that is assumed if no other value or condition is specified.
Digitize	A process of converting a physical position (defined by X,Y coordinates) to digital information a computer understands.

Eavesdrop	In an eavesdrop configuration, a peripheral is connected between the computer and a terminal. Therefore, data moves through the peripheral between the computer and terminal. The peripheral monitors this data, but passes it on until it recognizes information meant for it. This allows both the terminal and peripheral to use the same computer line.
Edge	The outline of a polygon.
Execute	To carry out an instruction or perform a routine. For example, when a plotter executes the select pen instruction, it gets a pen from the carousel.
File	A set of characters, numbers, and punctuation treated by the computer as a single unit.
Fill Type	The shading pattern used to fill a polygon.
Fixed Space Font	A type font in which each character occupies the same amount of space. The space between characters is uniform.
Font	A style used when lettering. With reference to plotters, the spacing between characters (variable or fixed-space) and the method of drawing characters (arc or vector).
Graphics Limits	<i>See Hardclip Limits and Softclip Limits</i>
Grid	A network of uniformly spaced horizontal and perpendicular lines.
Handshake	Communication between the computer and plotter about the availability of I/O buffer space in the plotter. The purpose of handshaking is to ensure correct and complete data transfer. The plotter can use the following handshakes: hardwire, Xon-Xoff, enquire-acknowledge, and software checking.
Hard-Clip Limits	That part of the plotting area beyond which the pen physically cannot move; the mechanical limits of the plotter.
Hatch	A fill type made of parallel lines.
HP-GL	Hewlett-Packard Graphics Language; the graphics instruction set Hewlett-Packard plotters understand.

HP-IB	Abbreviation for Hewlett-Packard's Interface Bus. Hewlett-Packard's version of IEEE Standard 488-1978 for interfacing programmable devices (e.g., computers, plotters, and printers).
IEEE 488-1978 Interface	A parallel interface standardized by EIA* Standard 488-1978.
Initialize	To set plotter conditions to known default values.
Input/Output (I/O)	Relating to the equipment or method used for transmitting information between (in and out of) devices.
Interface	Anything (a cable, for example) used to join components of a computer system so they function in a compatible and coordinated fashion. Also, standards that allow systems to connect to each other, i.e., RS-232-C, HP-IB (IEEE-488).
I/O Error	An error that occurs in the transmission process between a computer and peripheral. Examples of I/O errors are baud rate or parity mismatch, and incorrect syntax associated with device-control instructions.
Label Mode	The HP-GL label instruction, LB, causes the plotter to print text until it receives a special label terminator. This is known as label mode.
Label Terminator	The final character in every label string. This character terminates label mode, so that the plotter interprets subsequent characters as HP-GL instructions.
Line	The height of the character plot cell; the line includes both the character and the blank area above it. The default size of a line is 2 times the height of a capital A for fixed-space fonts. For variable-space fonts, the line is 2 times the height of the average capital letter.
Line Feed	A non-printing ASCII character (LF) that moves the plotter pen down one line (when in label mode).
Line Type	The line pattern (dashed and/or dotted) with which the plotter draws lines.

*Electronics Industry Association.

Literal String	When using BASIC, any sequence of letters, numbers, and symbols enclosed in quotation marks. Literal characters are taken literally to represent themselves.
Mnemonic	An easy to remember abbreviation. Each HP-GL instruction is a two-letter mnemonic designed to remind you of the instruction's function. For example, SP (select pen) and LT (line type) are HP-GL mnemonics.
Modem	Modulator-demodulator. A device which links a computer to another device, commonly used with telephones and telephone transmission lines. A modem acts as a data translator between devices.
Monitor Mode	A functional state in which the plotter echoes instructions it receives back to a terminal or computer. PARSE MODE echoes instructions after they have been parsed. RECEIVE MODE echoes instructions and escape sequences as soon as they are received.
Operating System	The computer software or firmware that controls the execution of programs.
Output Terminator	The character(s) sent by the plotter at the end of the response to an output instruction.
Overflow	To exceed the capacity of a buffer.
Overhead	Computer operations which are necessary to maintain control of a situation. In a program, those parts which set up the program (e.g., assign variables, reserve memory), as opposed to performing input and output operations.
Parallel Interface	An interface type in which a separate line is used for each data bit in a byte or word and all bits are transferred simultaneously. Centronics and HP-IB (IEEE-488) are examples of parallel interfaces.
Parallel Polling	A method used to poll all devices on an HP-IB bus to find out which device is requesting service. The P-mask parameter of the input mask (IM) instruction specifies what conditions cause a positive response to a parallel poll.

Parameter	One or more characters following an HP-GL mnemonic. The parameters govern how the instruction is executed by the plotter. For example, using a parameter of 2 with the select pen instruction (<i>SP2;</i>) directs the plotter to select pen number 2; if you used a parameter of 1, the the plotter would select pen number 1.
Parity	An error-checking method for information transfer between a computer and a peripheral device. Parity is used to check the accuracy of binary data.
Parse	To subdivide an instruction into components that the plotter can more easily understand and use. For example, an HP-GL instruction is divided into a mnemonic, parameters, separators, and a terminator.
Peripheral	A device separate from, but used with a computer; a disc drive, printer, or plotter, for example.
Plotter Units	The fixed units the plotter understands. Each plotter unit is 0.025 mm. There are 40 plotter units per millimetre.
Point	A location in the plotting area defined by an X,Y coordinate pair.
Poll (HP-IB only)	A computer process that determines the status or condition of devices on an HP-IB bus. Polling finds out which device has requested service. The input mask (IM) instruction specifies which conditions generate a service request.
Polygon	A closed shape. Polygons can be simple shapes such as circles, rectangles, and wedges, or more complex shapes such as block letters.
Polygon Buffer	A portion of the plotter's buffer that stores the points of a polygon that is being defined.
Polygon Mode	A mode established by the polygon mode (PM) instruction. In this mode, the points used to define a polygon are temporarily stored in the plotter's polygon buffer. These points cannot be used to draw the polygon until polygon mode is exited, and the points are retrieved by a fill polygon (FP) or edge polygon (EP) instruction.

Relative Plotting	Plotting to a point whose location is specified <i>relative</i> to the current pen position.
Repeatability	When referring to plotters, a measure of how closely a plotter can return a pen to the previously plotted point.
Resolution	When referring to plotters, addressable resolution is the smallest move you can make programmatically. Mechanical resolution is the smallest move the device can physically make—usually more precise than addressable resolution.
RS-232-C Interface	A serial interface standardized by EIA* Standard RS-232-C.
Scaling	Dividing the plotting area into units convenient for your application.
Scaling Points	Points that are assigned the user-unit values specified in the scale instruction, SC. These points, known as P1 and P2, define opposite corners of a rectangular area.
Separator	A symbol that separates the parameters of an instruction. For example, the comma in this string is a separator: (PA10, 20,).
Serial Interface	A serial interface uses a single data line to transfer data bits sequentially between devices. RS-232-C is an example of a serial interface.
Soft-Clip Limits	That part of the plotting area defined by the JW instruction, beyond which no programmed plotting can occur.
Space	The width of the character plot cell; the space includes both the character and the blank area to the right. The default size of the space depends on the size of the character; the average space is 1.5 times the character width.
Spooling	A technique by which output is temporarily placed in storage for future transmission, allowing for more efficient use of the system.

*Electronics Industry Association.

Standard Character Set	The plotter's primary (default) character set. By using the designate standard character set (CS), you can define any character set as the standard set.
Standby Mode	RS-232-C only. A functional state activated manually from the front panel. In STANDBY mode, the plotter remains in the programmed-off state, either passing data through when set to EAVESDROP , or ignoring all data in the STANDALONE configuration. The standby mode is useful in debugging (when in EAVESDROP and devices are having difficulty communicating through the plotter) or when transmitting binary data between the computer/modem and the terminal.
Stop Bit	In an RS-232-C configuration, a bit (or bits) following a character that notifies the receiving device that the character is complete.
String	In BASIC, any sequence of letters, numbers, and symbols.
Subpolygon	A polygon defined as part of a larger polygon. For example, the block letter O is a polygon that consists of two subpolygons: the outside circle and the inside circle.
Syntax	The rules governing the structure of a language. In HP-GL, the syntax governs the sequence of mnemonics and parameters, the separators between mnemonics and parameters, and terminators at the end of a string.
Syntax Error	An error in an instruction due to a misspelled or missing character, or bad punctuation.
Terminator	A character that signals the end of an HP-GL or device-control instruction.
Throughput	Relates to productivity; throughput is the total time required to complete a drawing.
Tick	A small mark that is often used to indicate a certain number of units along an axis. Major ticks are often used to mark every 5th or 10th unit, whereas minor ticks often mark single units.
Timeshare	The sharing of a central computer by several terminals. In a timeshare environment, each user is typically at a remote location.

Truncate	To discard the decimal portion of a number. For example, if you truncate 2.9 the result is 2.
User Units	The units you use to suit your application. Specify them with the scale instruction, SC. <i>See also</i> Plotter Units.
Variable	A value that can be changed; usually represented by a letter or group of letters.
Variable Space (Proportional) Font	A font in which all characters do not occupy the same amount of space. The space between characters is determined by the width of individual characters.
Vector Font	A font in which the smoothness of the curved letters is not programmable.
Window	The part of the plotting area in which plotting can occur. Also referred to as soft-clip limits.

Subject Index

A

AA instruction 6-7-6-8
AF instruction 12-9
AH instruction 12-10
AP instruction 10-4-10-5
AR instruction 6-9-6-10
AS instruction 10-6
Abort device-control, ESC.J 15-11
Abort graphics, ESC.K 15-12
Absolute
 arc, AA 6-7-6-8
 character size, SI 7-39-7-41
 direction, DI 7-16-7-22
 plotting, PA 4-11
 rectangle, RA 5-14-5-15
Acceleration select, AS 10-6
Addressing protocol (HP-IB) 16-3-16-4
Advance full page, AF 12-9
Advance half page, AH 12-10
Allocate configurable memory,
 ESC.T 15-21-15-23
Alternate character set 13-21-13-22
Arc absolute, AA 6-9-6-10
Arc fonts *see also* Variable Space Fonts
 general 13-2
 tables B-9-B-17
Arc relative, AR 6-9-6-10
ASCII characters
 for each character set B-6-B-17
 in terminating labels 7-2
 list of control characters 13-14
 using CHR\$ function 7-2
 using keyboard 7-3
Automatic modem disconnect modes 16-21
Automatic pen operations, AP 10-4-10-5

B

BL instruction 7-11-7-13
BASIC
 in program examples 1-4-1-7
Block size, *see* Data block size
Buffer label, BL 7-11-7-13
Buffered labels
 interaction with LO 7-12
Buffers, *see also* Downloadable character
 buffer, I/O buffer, Logical I/O buffer,
 Physical I/O buffer, Polygon buffer,
 Pen sort buffer, and Vector buffer
allocating size using ESC.T 15-21-15-23
allocating size using GM 8-19-8-20
general 8-1-8-5

C

CA instruction 13-21-13-22
CC instruction 13-23-13-25
CI instruction 6-11-6-15
CM instruction 13-25-13-26
CP instruction 7-13-7-15
CS instruction 13-27-13-28
CT instruction 6-15-6-17
CV instruction 10-7-10-8
Carriage-return point 7-3-7-4
Character chord angle, CC 13-23-13-25
Character plot cell 7-7-7-8, 13-6
Character plot, CP 7-13-7-15, 13-7
Character selection mode, CM 13-25-13-26
Character selection modes
 definition 13-11
 fallback mode 13-14-13-15
 HP 7-bit mode 13-15
 HP 8-bit mode 13-16-13-17
 how to choose 13-11-13-12
 ISO 7-bit mode 13-17-13-18

C (Continued)

ISO 8-bit mode 13-18-13-20
in-use code table 13-12-13-14
response to control characters 13-14
Character sets *see also* Character selection
 modes
 alternate 13-21-13-22
 general 13-2-13-8
 Kanji D-7-D-30
 list of 13-3
 shift-in and shift-out 13-9-13-10
 standard 13-27-13-28
 tables B-5-B-8
Character size
 absolute, SI 7-39-7-41
 relative, SR 7-45-7-47
Character slant, SL 7-42-7-44
Characters, user-defined, *see also* Character
 plot cell and Downloadable character
 general 13-28-13-31, 13-38-13-43
Chord tolerance
 chord angle 6-2
 chord calculation 8-11
 deviation distance 6-3
Chord tolerance, CT 6-15-6-17
Circle, CI 6-11-6-15
Configurable memory, *see* Buffers
Configuration statement 1-4
Coordinate system 2-1-2-2
Creating mirror images of plots 9-5-9-6
Current units 3-4
Curved line generator, CV 10-7-10-8

D

DC instruction 11-5
DF instruction 3-14-3-16
DI instruction 7-16-7-22
DL instruction 13-28-13-31
DP instruction 11-7
DR instruction 7-22-7-26
DS instruction 13-32-13-33
DT instruction 7-27-7-28
DV instruction 7-29-7-30
Data transmission modes 16-19-16-20
Default, DF 3-14-3-16
Default conditions 3-5, 3-14-3-18
Default P1 and P2 coordinates and media
 sizes 3-8-3-10

Define downloadable character, DL
 13-28-13-31
Define key, KY 10-13-10-15
Define label terminator, DT 7-27-7-28
Designate alternate character set, CA
 13-21-13-22
Designate character set into slot, DS
 13-32-13-33
Designate standard character set, CS
 13-27-13-28
Deviation distance 6-3
Device-control instructions 1-3, 15-1-15-6
Digitize clear, DC 11-6
Digitize point, DP 11-7
Digitizing
 general 11-1
 HP-IB interrupts and polling 11-5
 instructions 11-6-11-8
 manual 11-2-11-3
 monitoring the status byte 11-3-11-5
Direction absolute, DI 7-16-7-22
Direction mode, *see* DV instruction
Direction relative, DR 7-22-7-26
Direction vertical, DV 7-29-7-30
Disconnect modes 16-21

E

EA instruction 4-7-4-8
EC instruction 12-11
EP instruction 8-15-8-16
ER instruction 4-9-4-10
ES instruction 7-31-7-32
EW instruction 6-17-6-20
ESC.(15-25
ESC.) 15-25
ESC.@ 15-26
ESC.A 15-7
ESC.B 15-8
ESC.E 15-9-15-10
ESC.H 16-22
ESC.I 16-23-16-25
ESC.J 15-11
ESC.K 15-12
ESC.L 15-13
ESC.M 16-25-16-27
ESC.N 16-28-16-29
ESC.O 15-14-15-16
ESC.P 16-29-16-30
ESC.Q 15-17-15-18
ESC.R 15-19

E (Continued)

ESC.S 15-20
ESC.T 15-21-15-23
ESC.U 15-24
ESC.Y 15-25
ESC.Z 15-25
E-mask 14-10
Edge absolute rectangle, EA 4-7-4-8
Edge polygon, EP 8-15-8-16
Edge relative rectangle, ER 4-9-4-10
Edge wedge, EW 6-17-6-20
Enable cut line, EC 12-11
End flush mode, ESC.U 15-24
Enlarging a scaled picture 9-3
Equal-sized pictures on one page,
 drawing 9-4
Enquire/Acknowledge handshake
 general 16-7
 using device-control instructions
 16-12-16-16
Errors
 identifying 14-4
 HP-GL A-1
 device-control A-2
Escape function and flush mode 10-3,
 10-14-10-15
Examples
 complete program listings C-1-C-10
 configuration statements 1-5
Extra space, ES 7-31-7-32

F

FP instruction 8-17-8-18
FR instruction 12-12
FS instruction 10-9-10-10
FT instruction 5-6-5-9
Fill types 5-3
Fill rectangle absolute, RA 5-14-5-15
Fill rectangle relative, RR 5-16-5-17
Fill polygon, FP 8-17-8-18
Fill wedge, WG 6-21-6-26
Flush mode 10-14-10-15
Force, default by carousel type 10-9
Force select, FS 10-9-10-10
FORTRAN 1-4-1-7
Frame advance, FR 12-12
Front-panel display, writing messages
 on 10-3
Function keys, redefining 10-3, 10-13-10-15

G

GC instruction 14-9
GM instruction 8-19-8-20
GP instruction 10-11-10-12
Graphics, abort 15-12
Graphics limits *see* Soft-clip limits and
 Hard-clip limits
Graphics memory, GM 8-19-8-20
Graphics Sets GL and GR 13-12-13-20
Graphics Slots G0-G3 13-12-13-20
Group count, GC 14-9
Group pen, GP 10-11-10-12

H

Handshaking, *see also* Enquire/Acknowledge
 handshake, Hardwire handshake,
 Software checking handshake, Xon-Xoff
 handshake
 general 16-4
 choosing a handshake 16-5-16-7
 using device-control instructions
 16-7-16-19

Hard-clip limits 2-4, 3-6

Hardwire handshake
 general 16-7-16-8
 using device-control instructions
 16-9-16-10

HP-GL
 description 1-2, 3-2-3-4
 errors A-1
 NOP (No Operation) A-3

HP-IB
 addressing 16-3-16-4
 general 16-1-16-5
 reactions to DCL and SDC 16-5
 valid device-control instructions 16-2

I

IM instruction 14-10-14-12
IN instruction 3-17-3-18
IP instruction 3-19-3-20
IV instruction 13-34-13-35
IW instruction 9-8-9-10
Initialize, IN 3-17-3-18
Initializing the plotter, *see also*
 DF instruction and IN instruction
 general 3-5
 initialized conditions 3-14-3-18

I (Continued)

Input mask, IM 14-10-14-12
Input P1 and P2, IP 3-19-3-20
Input window, IW 9-8-9-10
I/O buffer, *see also* Logical I/O buffer and Physical I/O buffer
 general 8-1-8-2
Interactive programming 10-3
Invoke character slot, IV 13-34-13-35
In-use code table 13-12-13-14
ISO 7-bit character selection mode 13-17-13-18
ISO 8-bit character selection mode 13-18-13-20

K

KY instruction 10-13-10-15
Kanji
 alphabets in character set 101 D-1
 accessing characters D-2-D-4
 character set matrix D-6
 in symbol mode D-5
 terminating Kanji labels D-5
Keyboard mode *see* WD instruction

L

LB instruction 7-33-7-34
LO instruction 7-35-7-37
LT instruction 5-9-5-12
Label
 instruction (LB) 7-33-7-34
 terminator (default) 7-2
 using variables 7-4-7-6
Label origin, LO 7-35-7-37
Label terminator
 default 7-2
 sending the label terminator 7-3
Leased-line disconnect mode 16-21
Line type, LT 5-9-5-12
Logical I/O buffer 15-26-15-28

M

Masks, *see* IM instruction
Media sizes 3-8-3-10
Mirror images 9-5-9-6
Monitor mode, set 15-17-15-18

N

NR instruction 10-16
Not ready, NR 10-16

O

OA instruction 14-13-14-14
OC instruction 14-14
OD instruction 11-8
OE instruction 14-15-14-16
OF instruction 14-17
OG instruction 14-18
OH instruction 9-11
OI instruction 14-19
OK instruction 10-17-10-18
OL instruction 14-20-14-21
OO instruction 14-22-14-23
OP instruction 9-12
OS instruction 14-23-14-24
OT instruction 14-25-14-26
OW instruction 9-13
Obtaining plotter information 14-2
Output actual pen status, OA 14-13-14-14
Output buffer size when empty, ESC. L 15-13
Output buffer space, ESC. B 15-8
Output carousel type, OT 14-25-14-26
Output commanded pen status, OC 14-14
Output configurable memory size, ESC. S 15-20
Output digitized point and pen status, OD 11-8
Output error, OE 14-15-14-16
Output extended error, ESC. E 15-9-15-10
Output extended status, ESC. O 15-14-15-16
Output factors, OF 14-17
Output group count, OG 14-18
Output hard-clip limits, OH 9-11
Output identification HP-GL, OI 14-19
 device-control, ESC. A 15-7
Output instructions
 description 14-2-14-8
 in HP-IB configuration 14-2
 in RS-232-C configuration 14-3
Output key, OK 10-17-10-18
Output label length, OL 14-20-14-21
Output options, OO 14-22-14-23
Output P1 and P2, OP 9-12

O (Continued)

Output request flowchart 16-27
Output response, summary 14-5-14-6
Output status, OS 14-23-14-24
Output Window, OW 9-13

P

PA instruction 4-11
PB instruction 7-38-7-39
PD instruction 4-12
PG instruction 12-13
PM instruction 8-21-8-24
PR instruction 4-14-4-15
PS instruction 12-14
PT instruction 5-12-5-13
PU instruction 4-16
P-mask 14-12
P1 and P2
 default locations 3-8-3-10
 input instruction 9-8-9-10
Page feed, PG 12-13
Page size, PS 12-14
Parallel polling 14-7-14-8
Pascal 1-4-1-7
Pen
 automatic operations 10-4-10-5
 identifying location and position 4-2
 monitoring location and position 4-3
Pen down, PD 4-12-4-13
Pen sort buffer 8-3
Pen thickness, PT 5-12-5-13
Pen up, PU 4-16
Pen up merging 10-4
Physical I/O buffer 8-1-8-2
Plot absolute, PA 4-11
Plot buffer, PB 7-38-7-39
Plot relative, PR 4-14-4-15
Plotter off, ESC.Z 15-25
Plotter on, ESC.Y 15-25
Plotter units 2-3, 3-8-3-10
Polling, *see also* IM instruction
 general 14-7
 parallel 14-7-14-8
 serial 14-7
Polygon buffer
 general 8-2, 8-9
 determining approximate size 8-9
 determining exact size 8-12-8-13

Polygon mode

 appropriate instructions in 8-6
 circles in 8-7-8-8
 instruction (PM) 8-21-8-24

Polygons, counting points in 8-10-8-12

Program errors 1-9

Programming hints 1-8

Programming languages

 BASIC 1-4-1-7
 FORTRAN 1-4-1-7
 Pascal 1-4-1-7

R

RA instruction 5-14-5-15

RO instruction 9-14-9-15

RR instruction 5-16-5-17

Rectangles

 absolute 4-7-4-8
 general 4-6
 relative 4-9-4-10

Reducing a scaled picture 9-3

Relative

 character size 7-45-7-47
 movement 4-3

Reset, ESC.R 15-19

Rollfeed plotting *see* Long-axis plotting

Rotation

 general 9-7
 instruction (RO) 9-14-9-15
 program C-12-C-14

RS-232-C

 general 16-5

S

SA instruction 13-36

SC instruction 3-21-3-24

SG instruction 10-19

SI instruction 7-39-7-41

SL instruction 7-42-7-44

SM instruction 5-18-5-19

SP instruction 4-17

SR instruction 7-45-7-47

SS instruction 13-37

S-Mask 14-11

Scale, SC 3-21-3-24

Scaling, *see also* SC instruction

 description 2-4, 3-11-3-13

 isotropic and anisotropic scaling

 3-21-3-24

S (Continued)

Select alternate character set, SA 13-36
Select pen group, SG 10-19
Select pen, SP 4-17
Select standard character set, SS 13-37
Serial polling 14-7
Set extended output and handshake mode,
 ESC.N 16-28-16-29
Set handshake mode, ESC.P 16-29-16-30
Set handshake mode 1, ESC.H 16-22
Set handshake mode 2, ESC.I 16-23-16-25
Set monitor mode, ESC.Q 15-17-17-18
Set output mode, ESC.M 16-25-16-27
Set plotter configuration, ESC.@ 15-26
Size absolute character, SI 7-39-7-41
Size relative character, SR 7-45-7-47
Slant character, SL 7-42-7-44
Soft-clip limits (windows), *see also* IW
 instruction
 general 2-5-2-6
Software checking handshake
 flowchart 16-17
 general 16-6
 using device control instructions
 16-16-16-19
Status byte, *see* IM instruction
Switched/Datex-line disconnect mode 16-21
Symbol mode, SM 5-18-5-19
Symbol mode using Kanji D-5
Syntax
 device control instructions 15-3
 HP-GL instructions 3-2-3-4

T

TL instruction 5-20-5-23
Tick length, TL 5-20-5-23
Tick marks, *see* XT instruction, YT
 instruction, and TL instruction

U

UC instruction 13-38-13-43
UF instruction 5-23-5-27
User-defined character, UC 13-38-13-43
User-defined fill type, UF 5-23-5-27
User units 2-3

V

VS instruction 10-20-10-21
Velocity select, VS 10-20-10-21

W

WD instruction 10-22-10-23
WG instruction 6-21-6-26
Window
 general 9-2
 input instruction (IW) 9-8-9-10
 output instruction (OW) 9-13
 program C-9-C-11
Write to display, WD 10-22-10-23

X

Xon-Xoff handshake
 general 16-6
 using device-control instructions 16-11
XT instruction 5-28
X-tick, XT 5-28

Y

YT instruction 5-29
Y-tick, YT 5-29



HEWLETT
PACKARD

PART NO. 07595-90001

PRINTED IN U.S.A., JANUARY 1987