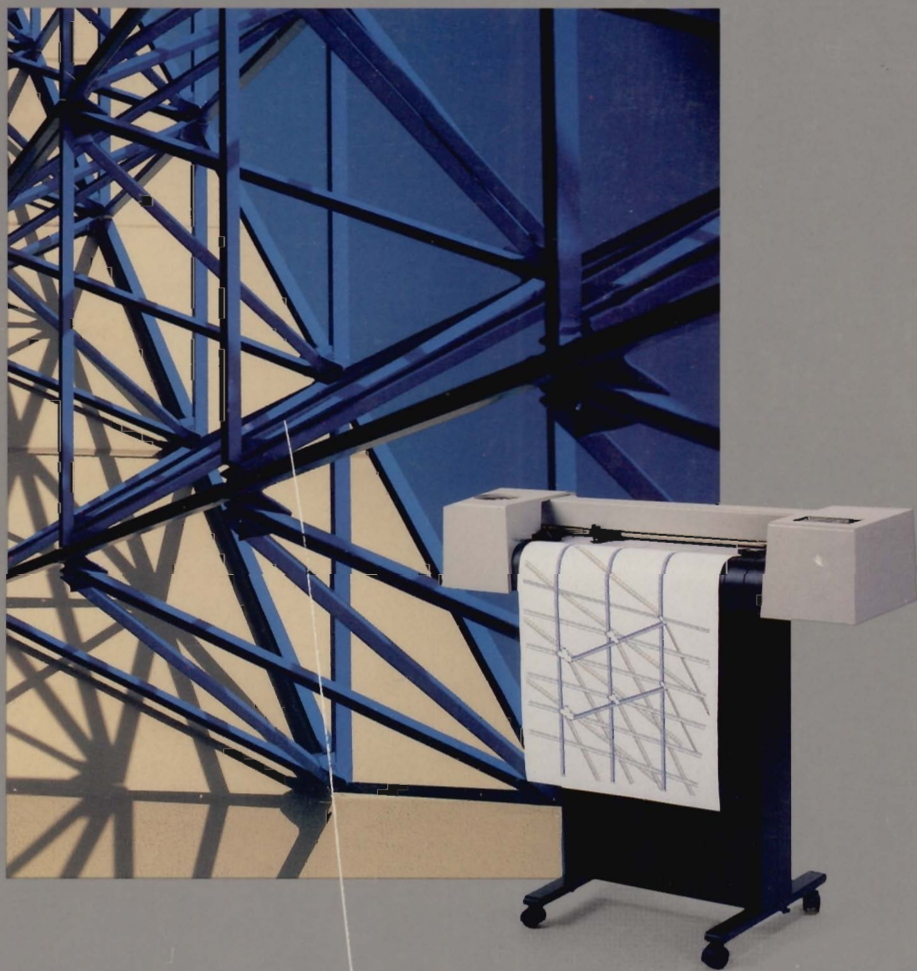


HEWLETT-PACKARD

F. VAN GRIEXEN

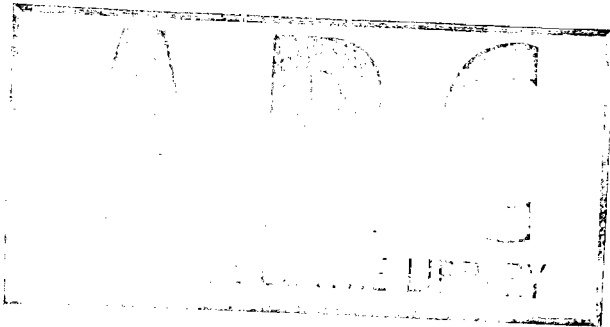
HP DraftPro Plotter Programmer's Reference



HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

HP DraftPro Plotter Programmer's Reference





NOTICE

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

Microsoft® is a U.S. registered trademark of Microsoft Corporation.

PRINTING HISTORY

First Edition — September 1986

How to Use This Documentation

This is a reference manual. It provides you with information concerning the Hewlett-Packard Graphics Language (HP-GL) and how it is used with your plotter.

This manual is not a tutorial. It does not teach you how to write a program. Your method of programming depends on your computer system, programming language, and level of expertise. This manual discusses many plotting concepts, including digitizing, interfacing and handshaking, and device control.

Once you are familiar with the fundamental plotting concepts and HP-GL, use this manual as a detailed reference to specific questions.

Manual Organization

Chapters 1 and 2 discuss general information regarding plotting concepts and HP-GL, including some details specific to the plotter.

Chapter 3 informs you of known (default) conditions when you turn on the plotter, and discusses preliminary HP-GL instructions.

Chapters 4 through 9 describe the most commonly used HP-GL instructions. These instructions enable you to plot lines, shapes (polygons), fill patterns, and labels (including character sets).

Chapters 10 through 12 explain how to obtain information from the plotter through the use of output instructions, and how to establish windows, rotate the coordinate system, and digitize points.

Chapters 13 and 14 discuss device-control instructions that allocate memory space and set plotter conditions for interfacing and handshaking.

Appendices include information on plotter error messages, ASCII character codes, character sets (including Kanji), and a glossary.

Manual Conventions

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

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

BOLD CONDENSED TYPE denotes a single ASCII character which should be sent to the plotter.

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.

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 13, respectively.

Additional Documentation

Following is a list of additional documentation pertinent to the your plotter.

- **HP DraftPro User's Guide** (Part No. 07570-90002). This manual is shipped with the plotter. It contains detailed information about loading paper and pens, using front-panel buttons, determining the best pen/paper combinations, and ordering available accessories. It also describes how to connect various computers with the plotter.
- **HP DraftPro Programmer's Pocket Guide** (Part No. 07570-90003). The Pocket Guide is a convenient reference list of all HP-GL and device-control instructions along with their parameters.

NOTE: If you are using a graphics software package with the plotter, be sure to follow any special instructions provided in the documentation accompanying the software. ■

Table of Contents

How to Use This Documentation	iii
Manual Organization	iii
Manual Conventions	iii
Additional Documentation	iv
Chapter 1: Programming Concepts	1-1
Understanding HP-GL	1-2
Sending HP-GL Using Your Programming Language	1-3
The Configuration Statement	1-3
Using the Program Examples	1-5
Complete Program Listings	1-5
HP-GL Strings	1-6
Hints For the Novice Programmer	1-7
A Note About Program Errors	1-8
Understanding the Plotter's Buffers	1-8
I/O Buffer	1-8
Polygon Buffer	1-9
Pen Sort Buffer	1-9
Notes for Buffer Allocation	1-10
Chapter 2: Plotting Concepts	2-1
Understanding the Coordinate System	2-1
Understanding the Units of Measure	2-2
Plotter Units	2-3
User Units	2-4
Understanding Graphic Limits	2-4
Hard-Clip Limits	2-5
Soft-Clip Limits	2-5
Scaling	2-7
Rotation	2-11
Axis Align	2-13
Chapter 3: Preliminary Setup Using HP-GL	3-1
Understanding HP-GL Instruction Syntax	3-1
Notations	3-2
Parameter Formats and Ranges	3-2
Syntax Compatibility with Other HP Plotters	3-3
Omitting Optional Parameters	3-4
Setting the Plotter to Known Conditions	3-4
Using the Scaling instructions	3-5
Working with the Scaling Points	3-5

Table of Contents (Continued)

Chapter 3: Preliminary Setup Using HP-GL (Continued)	
Effective Scaling	3-6
Isotropic and Anisotropic Scaling	3-6
DF, Default	3-8
IN, Initialize	3-11
IP, Input P1 and P2	3-12
NR, Not Ready	3-13
SC, Scale	3-14
Chapter 4: Drawing Lines	4-1
Selecting and Controlling the Pen	4-1
Monitoring Pen Position and Location	4-2
Absolute and Relative Movement	4-2
A Note on Relative Movement when Scaling is Off	4-5
Plotting With Variables	4-6
AP, Automatic Pen Operations	4-6
PA, Plot Absolute	4-8
PD, Pen Down	4-10
PR, Plot Relative	4-12
PU, Pen Up	4-14
SG, Select Pen Group	4-15
SP, Select Pen	4-16
VS, Velocity Select	4-17
Chapter 5: Drawing Circles, Arcs, Wedges, and Rectangles	5-1
Drawing Circles and Arcs	5-1
Maintaining Circle/Arc Smoothness	5-2
Drawing Wedges	5-5
Drawing Rectangles	5-6
Choosing a Fill Type	5-7
AA, Arc Absolute	5-8
AR, Arc Relative	5-10
CI, Circle	5-12
CT, Chord Tolerance	5-15
EA, Edge Absolute Rectangle	5-16
ER, Edge Relative Rectangle	5-18
EW, Edge Wedge	5-20
FT, Fill Type	5-24
PT, Pen Thickness	5-27
RA, Rectangle Absolute Fill	5-28

Table of Contents (Continued)

Chapter 5: Drawing Circles, Arcs, Wedges, and Rectangles (Continued)

RR, Rectangle Relative Fill	5-30
WG, Wedge Fill	5-32

Chapter 6: Drawing Polygons and Using the Polygon Buffer

Understanding and Defining Polygons	6-1
Drawing Polygons and Subpolygons	6-2
Drawing Circles in Polygon Mode	6-3
Using the Polygon Buffer	6-4
Determining the Approximate Buffer Size	6-5
Determining the Exact Buffer Size	6-7
Changing the Size of the Polygon Buffer	6-9
EP, Edge Polygon	6-9
FP, Fill Polygon	6-11
GM, Graphics Memory	6-13
PM, Polygon Mode	6-14

Chapter 7: Enhancing Your Plots

LT, Line Type	7-1
SM, Symbol Mode	7-5
TL, Tick Length	7-7
XT, X-Tick	7-9
YT, Y-Tick	7-10

Chapter 8: Labeling Your Plots

Using Labels	8-1
Default Label Conditions	8-1
Terminating Labels	8-2
Positioning Labels	8-3
The Character Plot Cell	8-5
Enhancing Labels	8-6
Adjusting Character Size	8-6
Changing Character Spacing	8-6
Adjusting Label Direction	8-6
Labeling With Variables	8-7
CP, Character Plot	8-8
DI, Direction Absolute	8-11
DR, Direction Relative	8-15
DT, Define Terminator	8-18

Table of Contents (Continued)

Chapter 8: Labeling Your Plots (Continued)	
DV, Direction Vertical	8-21
ES, Extra Space	8-23
LB, Label	8-25
LO, Label Origin	8-26
SI, Size Absolute Character	8-29
SL, Slant Character	8-31
SR, Size Relative Character	8-33
Chapter 9: Character Sets and Characters	9-1
Using Character Sets	9-1
Designating and Selecting Character Sets	9-3
Standard and Alternate Character Sets	9-3
Special Characters	9-4
Designing Characters	9-4
CA, Designate Alternate Character Set	9-5
CS, Designate Standard Character Set	9-6
SA, Select Alternate Character Set	9-7
SS, Select Standard Character Set	9-9
UC, User-defined Character	9-10
Chapter 10: Changing Picture Area and Orientation	10-1
Windowing: Setting Up Soft-Clip Limits	10-1
Enlarging or Reducing a Picture	10-2
Drawing Equal-Sized Pictures on One Page	10-3
Creating Mirror Images	10-5
Rotating a Picture	10-7
Using the Output Instructions in This Chapter	10-8
IW, Input Window	10-9
OH, Output Hard-Clip Limits	10-11
OP, Output P1 and P2	10-12
OW, Output Window	10-13
RO, Rotate Coordinate System	10-14
Chapter 11: Obtaining Information from the Plotter	11-1
Notes for Obtaining Plotter Output	11-1
Hints for the HP-IB Configuration	11-2
Hints for the RS-232-C Configuration	11-3
Using Output Instructions	11-3
Identifying the Pen Location and Position	11-3
Identifying the Plotter and Its Functions	11-4

Table of Contents (Continued)

Chapter 11: Obtaining Information from the Plotter (Continued)	
Obtaining Error Information	11-4
Obtaining Status Byte Information	11-5
Summary of Output Responses	11-5
Polling	11-6
Serial Polling	11-7
Parallel Polling	11-7
IM, Input Mask	11-8
OA, Output Actual Pen Status	11-12
OC, Output Commanded Pen Status	11-13
OE, Output Error	11-14
OF, Output Factors	11-16
OI, Output Identification	11-16
OO, Output Options	11-17
OS, Output Status	11-19
OT, Output Carousel Type	11-20
Chapter 12: Digitizing	12-1
Using Digitizing Instructions	12-1
Preparing Your Plotter for Use as a Digitizer	12-2
Digitizing with the Plotter	12-2
Manual Digitizing	12-2
Monitoring the Status Byte	12-4
HP-IB Interrupts and Polling	12-6
DC, Digitize Clear	12-7
DP, Digitize Point	12-7
OD, Output Digitized Point	12-8
Chapter 13: Device-Control Instructions	13-1
Understanding Device-Control Instructions	13-1
Sending Device-Control Instructions	13-2
Syntax for Device-Control Instructions	13-2
Using Device-Control Instructions	13-3
Setting the Plotter to Known Conditions	13-3
Identifying the Plotter	13-4
Monitoring the Changing the Buffer Sizes	13-4
Verifying Errors or Operating Conditions	13-4
ESC . A, Output Identification	13-5
ESC . B, Output Buffer Space	13-5
ESC . E, Output Extended Error	13-6

Table of Contents (Continued)

Chapter 13: Device-Control Instructions (Continued)

ESC . J, Abort Device-Control	13-8
ESC . K, Abort Graphics	13-8
ESC . L, Output Buffer Size When Empty	13-9
ESC . O, Output Extended Status	13-9
ESC . Q, Set Monitor Mode	13-11
ESC . R, Reset	13-14
ESC . S, Output Configurable Memory Size	13-14
ESC . T, Allocate Configurable Memory	13-15
ESC . Y or ESC . (, Plotter On	13-18
ESC . Z or ESC .), Plotter Off	13-18
ESC . @, Set Plotter Configuration	13-19

Chapter 14: Interfacing and Handshaking

The Hewlett-Packard Interface Bus (HP-IB)	14-1
Controlling Addressing Sequences	14-3
Reactions to Bus Commands DCL and SDC	14-5
About RS-232-C Interfacing and Handshaking	14-5
Choosing a Handshake	14-6
Handshaking Through Device-Control Instructions	14-8
Hardwire Handshake	14-8
Xon-Xoff Handshake	14-10
Enquire/Acknowledge Handshake	14-13
Software Checking Handshake	14-17
Data Transmission Modes	14-20
Normal Mode	14-20
Block Mode	14-20
Automatic Modem Disconnect Modes	14-22
Switched/Datex-Line Disconnect Mode	14-22
Leased-Line Disconnect Mode	14-23
ESC . H, Set Handshake Mode 1	14-23
ESC . I, Set Handshake Mode 2	14-24
ESC . M, Set Output Mode	14-28
ESC . N, Set Extended Output and Handshake Mode	14-31
ESC . P, Set Handshake Mode	14-32

Appendix A - Error Messages

No Operation (NOP) Instructions	A-3
---------------------------------------	-----

Table of Contents (Continued)

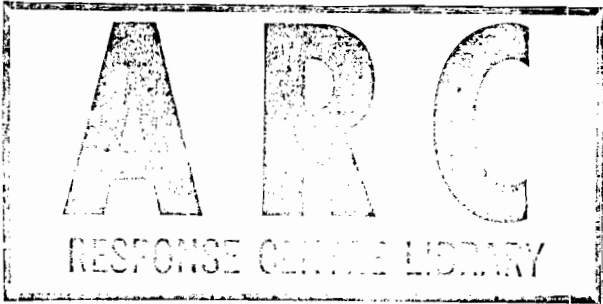
Appendix B - Reference Material B-1
 ASCII Character Codes B-1
 ASCII Control Codes B-2
 Character Sets B-5

Appendix C - Sample Plots and Program Listings C-1
 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: Digitizing (HP Series 200 BASIC) C-15

Appendix D - Using Kanji D-1
 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 G-1

Subject Index SI-1



CHAPTER





Programming Concepts

Before you can write graphics programs for your plotter, you need to be familiar with your computer and a programming language such as BASIC, FORTRAN, or Pascal. You can use any computer that inputs and outputs ASCII* information to and from a peripheral device. This manual assumes that you already have some experience using your programming language.

NOTE: If your computer uses 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 outputting literal strings to a peripheral. ■

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

- understanding HP-GL, including
 - graphics instructions
 - output instructions
- understanding device-control instructions
- using HP-GL with BASIC, FORTRAN, and Pascal
- using the examples in this manual

*American Standard Code for Information Interchange.

Understanding HP-GL

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 to control plotter operations such as input/output, handshaking, and allocating buffer space. HP-GL and device-control instructions are embedded in a program (using BASIC, FORTRAN, Pascal, etc.) as a literal string and output from the computer to the plotter.

Each HP-GL 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. You can follow many of the HP-GL mnemonics with numeric parameters that tell the plotter to execute (complete) the instruction in a particular way. For example, the following is a typical HP-GL instruction.

```
SP1;
```

The mnemonic tells the plotter to select a pen; the parameter instructs the plotter to select the pen in the number one stall in the pen carousel.

HP-GL instructions can be divided into two categories.

- graphics instructions
- output instructions

Of these, the graphics instructions category is the largest. The graphics instructions enable you to define your plotting area and set or select plotting conditions. This includes initializing the plotter, rotating the coordinate system, scaling the plotting area, selecting pens, and setting up standard and alternate character sets for labels.

The graphics instructions also let you draw geometrical shapes using a single instruction. For example, with one instruction you can draw a circle of any radius, an arc of any degree length, or any sized wedge of a circle. These shapes and features are inherent in your plotter, ready to be accessed through the use of HP-GL graphics instructions.

Output instructions enable you to obtain information from the plotter, such as error numbers, pen location and status, digitized points, hard-clip limits or the current window, current positions of P1 and P2, and plotter identification. All HP-GL instructions enter the input/output (I/O) buffer where, if pen sorting is disabled, they are executed on a first-in, first-out basis; if pen sorting is enabled, HP-GL instructions in the I/O buffer are sorted according to pen number, then executed. You can enable/disable pen sorting using the rear-panel switch (refer to the User's Guide) or using the AP (Automatic Pen Operations) instruction (refer to Chapter 3 of this manual).

Device-control instructions refer to operations such as data formatting, input/output (I/O), handshaking, and buffer allocation. The device-control instructions are used primarily with an RS-232-C interface. Although some device-control instructions can be used with an HP-IB interface, there is seldom need.

Device-control instructions are sent to the plotter as literal strings similar to sending HP-GL instructions. Unlike HP-GL instructions, however, device-control instructions *do not* enter the plotter's I/O buffer; the plotter executes them as they are received. Device-control instructions are discussed in more detail in Chapter 13, *Device-Control Instructions*, and Chapter 14, *Interfacing and Handshaking*.

Sending HP-GL Using Your Programming Language

Before you send any HP-GL instructions to your plotter, you must be certain that your computer and plotter are communicating properly. This is accomplished using a configuration statement that opens the lines of communication between the computer and plotter. After you have established communication, your programming language will determine how you send HP-GL instructions.

The Configuration Statement

To open the lines of communication between the computer and the plotter, you may need a configuration statement at the beginning of each program. Your configuration statement directs computer input and/or output to the plotter port. It will be specific

to your computer and the type of interface (RS-232-C or HP-IB) you are using.

The User's Guide contains sample programs to ensure that your computer and plotter are communicating. The following computers are listed in the User's Guide.

Apple IIc	HP Vectra
Apple IIe	HP 3000
Apple Macintosh	HP 9000 Series 200
Dec Vax	HP 9000 Series 300
HP Touchsreen	IBM PC/PC-XT/PC-AT

If you are using one of these computers, refer to Chapter 7, *Connecting Your Plotter to a Computer*, in the User's Guide for a sample communication program. Use the configuration statement found at the beginning of your computer's example program at the beginning of every program you write. If your computer is not listed, refer to your computer documentation and Chapter 14, *Interfacing and Handshaking*, in this manual, to help you determine your configuration requirements.

Here are some examples of opening lines for popular computers. In the program examples, you will see the line "Insert configuration statement here". That is where you should insert the opening statement for your computer. Refer to your computer documentation for a complete discussion of the parameters.

Apple IIc

RS-232-C interface

```
10 PRINT CHR$(4); "PR#n" (where n is the slot number)
20 PRINT CHR$(1); "10B"
30 PRINT CHR$(4); "IN#n" (where n is the slot number)
```

Apple IIe/II Plus

RS-232-C interface

```
10 PR#n:IN#n (where n is the slot number)
```

Apple Macintosh

RS-232-C interface

AT&T PC 6300

HP Vectra

IBM PC/PC-XT/PC-AT

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

HP Touchscreen (150) RS-232-C and HP-IB interfaces

```
10 OPEN "O",1,"PLT" (Series 100/BASIC)
```

```
10 OPEN "LPT3:" FOR OUTPUT AS #1 (GW BASIC)
```

Using the Program Examples

Examples are presented in one of two ways: either as complete programs, or as listings of pertinent HP-GL strings. The examples are designed to show the use of the instruction with which they appear.

Complete Program Listings

Examples presented as complete programs are written in a version of Microsoft BASIC for MS DOS operating systems. This BASIC is used by many popular personal computers.

If you are using Microsoft® BASIC, you can enter and run the full program examples as shown. Be sure to enter the proper configuration statement for your computer at the beginning of your program.

If you are not using Microsoft® BASIC, you need to insert the proper configuration statement at the beginning of the program and may need to change some of the program statements in the examples.

Below is a simple example of the way a complete program appears in this manual.

```
10 'Insert configuration statement here
20 PRINT #1, "IN;SP1;PU3000,5400;"
30 PRINT #1, "PD2600,4200,1400,3800,2600,3400;"
40 PRINT #1, "PD3000,2200,3400,3400,4600,3800;"
50 PRINT #1, "PD3400,4200,3000,5400;"
60 PRINT #1, "SP0;"
70 END
```

Using Non-BASIC Statements

If you are not using the Microsoft BASIC language version presented in this manual, or you are programming in a different language, you might need to change many of the BASIC commands to equivalent statements. Your computer's documentation should tell you how to do this. The following are some of the statements you may 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.
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.
FOR...NEXT X = 3.14	FOR...NEXT are loop statements. X = 3.14 is a variable assignment. Change these statements to whatever is comparable in your language.

HP-GL Strings

Since input/output instructions (e.g., PRINT # and INPUT #) vary among different computers, some examples present only the pertinent HP-GL strings (the two-letter mnemonics and applicable parameters). They are enclosed in quotation marks since most languages use quotation marks as string delimiters. *The plotter does not require the quotation marks; use whatever your computer requires.* Add the statements required by your system to send the string of instructions. For example, suppose you see the following HP-GL instruction printed in the manual.

```
“SP1;”
```

First use whatever opening statements your computer requires to define the output port and establish the plotter as the recipient of an output string. Then, send the string within a statement similar to the following, depending on your computer and language (only

a few computers are listed here to give you an idea of the variety of statements available).

HP 9000, Series 200, BASIC	PRINT "SP1;" or OUTPUT 205; "SP1;"
HP Touchscreen (150) IBM PC/PC-XT/AT	PRINT #1, "SP1;"
HP 9000, Pascal	WRITELN('SP1;')
HP 3000, FORTRAN	WRITE(6,10) 10 FORMAT(4HSP1;)

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 outputs information to peripheral devices and reads (inputs) information from another device.
- Know your programming language. When you encounter a new BASIC program statement in this manual, look up and use the equivalent statement in your language. Always use the syntax your language requires.
- Enter HP-GL 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. A common mistake is to substitute the letter "O" for the number zero, or a lowercase "l" for the number one. Substituting a semicolon for a comma, or omitting a quotation mark can make the difference between success and failure.
- Begin every program (as shown in the complete program examples) with your computer's configuration statement and either a DF or IN instruction.

When learning, study the basic HP-GL instructions in Chapters 4 through 9 and run the programming examples. Then try combining instructions in a simple program.

A Note about Program Errors

If your program contains HP-GL syntax or parameter errors, the plotter responds by blinking the light next to the **VIEW** button. The light continues to blink until you reset the plotter, initialize the plotter, or use the **OE** (Output Error) instruction to output the error number to your computer.

Each HP-GL instruction description includes a table of the HP-GL errors that are most likely to occur. Note that the plotter may or may not complete your plot correctly depending on the severity of the error.

Understanding the Plotter's Buffers

The plotter has three configurable buffers: input/output (I/O), polygon, 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 quantity of HP-GL 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 does not change. Change the size of the logical I/O buffer using the **ESC.@** device-control instruction. (Refer to Chapter 13 for complete descriptions of these device-control instructions.)

Note that only 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, HP-GL instructions are executed on a first-in, first-out basis; if pen sorting is enabled, HP-GL 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 Absolute Rectangle
- ER, Edge Relative Rectangle
- EW, Edge Wedge
- PM, Polygon Mode
- RA, Rectangle Absolute Fill
- RR, Rectangle Relative Fill
- WG, Wedge Fill

For more information concerning polygons and the polygon buffer, refer to Chapter 6. You can configure the Polygon buffer size using either the GM (Graphics Memory) instruction (described in Chapter 6) or the ESC.T device-control instruction (described in Chapter 13).

Pen Sort Buffer

When pen sorting is enabled, HP-GL 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 at the desired

location 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 pen sorting using either the rear-panel switch prior to turning on the plotter, or the AP (Automatic Pen Operations) instruction. You can turn on pen sorting with the AP instruction even if the rear-panel switch indicates that pen sorting is off. However, if the rear-panel pen sort switch is on, pen sort cannot be disabled using the AP instruction. The method you use to turn on pen sorting is the method you *must* use to turn it off.

Some output instructions are not useful when pen sorting is enabled. For example, the OA instruction, which outputs the actual location and position of the pen.

Notes for Buffer Allocation

The total configurable plotter memory is 7448 bytes. The following table shows the default, minimum, and maximum sizes for the plotter's buffers.

Buffer	Default	Minimum	Maximum
I/O	1024	2	7436
Pen sort	5400	12	7446
Polygon	1024	0	7434

If your program does not make use of 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. You can change the buffer allocations using the ESC.T device-control instruction or the GM instruction (applies to the pen sort and polygon buffers only). Note the following restrictions on buffer allocation.

- If the total ESC.T allocation exceeds 7448 bytes, the plotter establishes the buffers at their default sizes and sets I/O error number 13 (parameter out of range).
- If the total GM allocation exceeds 7448 bytes (including the I/O buffer), the buffer allocations remain unchanged and the plotter sets HP-GL error number 3 (bad parameter).

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.

Refer to Chapter 6 for more information regarding the GM instruction, and to Chapter 13 for details concerning the ESC.T instruction.

CHAPTER

2

Plotting Concepts

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

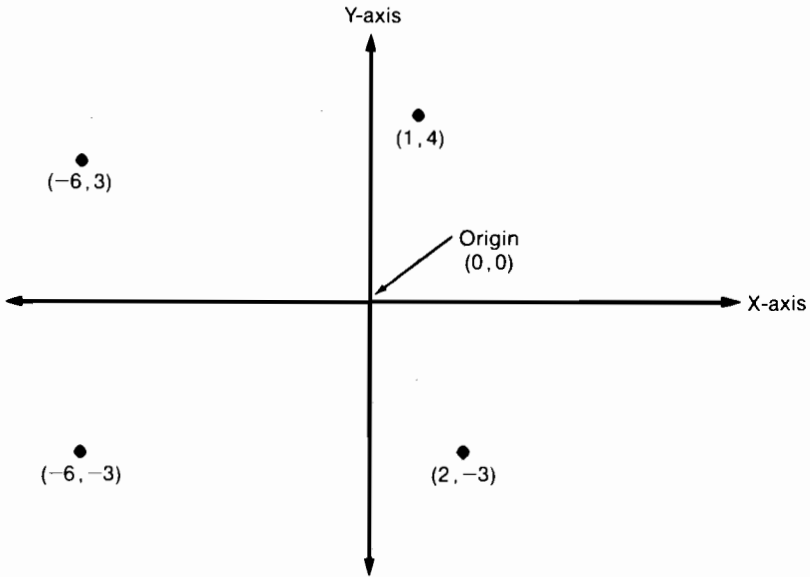
- graphic limits
- the coordinate system
- units of measure
- scaling
- absolute and relative pen movement

Understanding the Coordinate System

The plotting area is that portion of the paper in which the plotter's pen can draw. The system for locating points on the plotting area is based on the two-dimensional Cartesian coordinate system in which the plotting area is divided into a theoretical grid as shown in the following illustration. Each grid line represents a unit of measurement in one of two directions: horizontal or vertical. The horizontal grid line is referred to as the X-axis; the vertical grid line as the Y-axis. The intersection of these axes is called the origin. The plotter's default origin is at the center of the plotting area.

To locate any position (or point) in the plotting area, specify how many X-units and Y-units to move your pen from the origin. Each point, then, is referenced by the combination of its X-coordinate and Y-coordinate. Together, they are known as an

X,Y coordinate pair. The coordinate pair of the origin is 0, 0. Move to any location in the plotting area by specifying the coordinate pair of the target location; first the X-coordinate, then the Y-coordinate. For example, the coordinate pair of the point located 1 unit along the X-axis and 4 units along the Y-axis from the origin is 1, 4.



NOTE: X-coordinates to the left of the origin are expressed as negative values. The same is true for Y-coordinates below the origin. ■

Understanding the Units of Measure

You can express coordinates as one of two types of units: plotter units or user units. A plotter unit is a constant, fixed size. However, when defining and drawing your plot, another unit of measure may make more sense. A unit of measure that you define (through a process called scaling) is called a user unit.

Plotter Units

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

1 plotter unit = 0.025 mm or 0.00098 in.

40 plotter units = 1 mm

1016 plotter units = 1 in.

The numeric range of plotter units is **-32 768 to 32 767**. For practical purposes, though, the effective range of plotter units is limited to the size of the paper you are using.

The plotter unit ranges for C- and D-sized plotting areas are shown in the following table, along with the corresponding metric and architectural size plotting areas.

Paper Size	Mode	Maximum Plotter Unit Range	
		X-axis	Y-axis
C	Normal	-10 576 to 10 576	-7 556 to 7 556
	Expand	-10 976 to 10 976	-7 916 to 7 916
D	Normal	-16 192 to 16 192	-10 576 to 10 576
	Expand	-16 552 to 16 552	-10 976 to 10 976
A2	Normal	-11 280 to 11 280	-7 320 to 7 320
	Expand	-11 680 to 11 680	-7 680 to 7 680
A1	Normal	-15 740 to 15 740	-11 280 to 11 280
	Expand	-16 100 to 16 100	-11 680 to 11 680
Architectural C	Normal	-11 592 to 11 592	-8 064 to 8 064
	Expand	-11 992 to 11 992	-8 424 to 8 424
Architectural D	Normal	-17 208 to 17 208	-11 592 to 11 592
	Expand	-17 568 to 17 568	-11 992 to 11 992

NOTE: The values in the preceding table are based on exact paper size. Other contributing factors that affect the values of the plotting range include the position of the right pinch wheel (which affects the measured distance along the platen), the "squareness" of the media, and the accuracy of the alignment with the front and rear paper guides. ■

User Units

User units enable you to control the number of units along each coordinate axis and assign the origin (0, 0) to any location on the plotting area. You can let user units represent specific distances, elevations, units of time, temperatures, or whatever you require. User units are established through a process known as *scaling*. Scaling superimposes a user unit grid on the actual plotter unit grid.

When scaling, you specify the number of units along each axis so that the plotter automatically converts one user unit to a suitable number of plotter units. In effect, you are assigning a scale of user units to plotter units. This is similar to scaling on road-maps where one inch (a user unit) represents a specific number of miles.

User units need not be the same size on both axes. Nor does there need to be an equal number of user units on both axes. For example, you might want to chart changes in elevation over an extended distance and establish units of elevation along the Y-axis and the units of distance along the X-axis. Refer also to the discussion on scaling later in this chapter.

User units let you specify coordinates with fractions (to four decimal places), which makes it easier to directly represent your data. To plot the scaled data, the plotter converts user units to plotter units. The maximum range the plotter will accept when scaling is on is **-32 768.0000 to 32 767.9999**. If the converted number is not within the allowable range, the plotter generates a bad parameter error (error number 3). After converting user units to plotter units, the plotter rounds to the appropriate integer. (When the first decimal is 5 or more, the plotter increments positive numbers by 1; negative numbers are decremented by 1.)

For more information on user units and scaling, refer to *Scaling* later in this chapter.

Understanding Graphic Limits

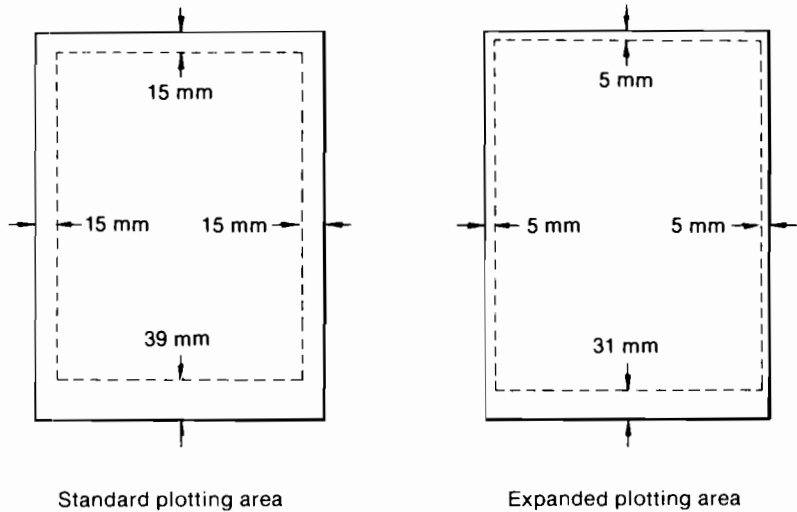
Graphic limits define a picture's border. The plotter recognizes two types of graphics limits: hard-clip and soft-clip limits. When you turn on or initialize the plotter, the hard-clip and soft-clip limits are identical. The word "clip" indicates that a portion of

the paper is being selected; that is, the plotter can only draw up to the clip limit and not beyond.

The hard-clip limits depend on the position of the rear-panel **EXPAND** switch at power on and the size of the paper loaded in the plotter. You can set soft-clip limits when necessary.

Hard-Clip Limits

The hard-clip limits represent the physical boundary for pen movement and correspond to the maximum plotting area available. At power on, the plotter reads the rear-panel **EXPAND** switch (for standard or expanded plotting areas) and sets the hard-clip limits as shown in the following illustration.

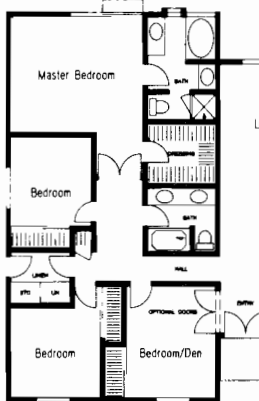


Note that the widest margin is always toward the front of the plotter. Once the margins are established by the **EXPAND** switch, they apply to both C- and D-size paper. Refer to the User's Guide for more information.

Soft-Clip Limits

Soft-clip limits temporarily restrict pen movement to a specified area of the page. Because the soft-clip limits enable you to draw a restricted portion of data they are often referred to as *windows*.

For example, consider the following floor plan. After plotting the full plan, suppose you decide to draw a new plot showing only the sleeping quarters and use the space on the right for text. To do this, use the program for the full floor plan and add soft-clip limits to restrict plotting to the left half.



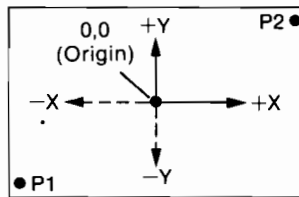
You could draw labels or another plot in this area

Using Soft-Clip Limits to Change the Plotting Area

Soft-clip limits are useful when you don't know the range of data you will be plotting; you can plot data within a specific window. Soft-clip limits are established using the IW (Input Window) instruction, which is described in Chapter 10.

Scaling

Scaling allows you to establish units of measure with which you are familiar, or which are more logical to your plot. Scaling relies on the relationship between two points: P1 and P2. These two points are called scaling points because they take on the user unit values that you specify in the SC (Scale) instruction. The scaling points are set 600 plotter units within the hard-clip limits whenever the plotter is turned on or initialized. P1 and P2 always represent absolute plotter unit locations on the paper and define the lower-left and upper-right corners of a rectangular plotting area. You can see this in the following illustration.



Default Orientation of P1 and P2

The default X,Y coordinates for P1 and P2 are listed in the following table for all supported paper sizes.

Paper Size	Mode	P1	P2
C	Normal	-9976, -6956	9976, 6956
	Expand	-10 376 -7316	10 376, 7316
D	Normal	-15 592, -9976	15 592, 9976
	Expand	-15 952, -10 376	15 952, 10 376

(Table continues)

Paper Size	Mode	P1	P2
A2	Normal Expand	-10 680, -6720 -11 080, -7080	10 680, 6720 11 080, 7080
A1	Normal Expand	-15 140, -10 680 -15 500, -11 080	15 140, 10 680 15 500, 11 080
Architectural C	Normal Expand	-10 992, -7464 -11 392, -7824	10 992, 7464 11 392, 7824
Architectural D	Normal Expand	-16 608, -10 992 -16 968, -11 392	16 608, 10 992 16 968, 11 392

NOTE: Because the values of P1 and P2 are dependent on the hard-clip limits, the same restrictions apply to the values in the preceding table as apply to the values for the maximum plotting range described earlier in the chapter. ■

Scaling lets you specify the range of units you want along each axis within the area defined by P1 and P2. That is, you specify the minimum and maximum X-axis values, then minimum and maximum Y-axis values. For example, to show variations in elevation over a thousand mile area, you might scale your X-axis from 0 to 1000 (miles), and your Y-axis from 2000 to 11 000 (feet).

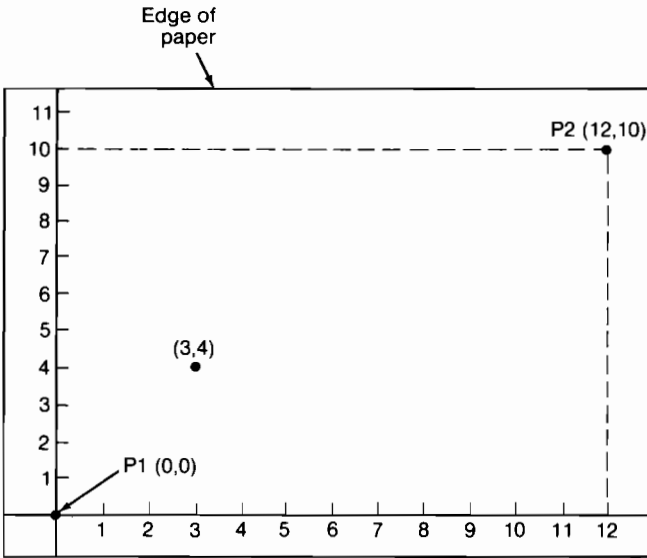
When you issue an SC instruction, P1 takes on the minimum X- and Y-coordinate values, P2 the maximum X- and Y-coordinate values. This is called mapping; the coordinate values are “mapped” onto P1 and P2. Continuing our example from above, P1 would have the coordinate value 0, 2000 and P2 1000, 11 000.

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 IP (Input P1 and P2) instruction or with buttons on the front panel (refer to the User’s Guide). ■

For a scaled plot, the entire paper is effectively divided into a theoretical grid based on the new user units. The actual size of the units depends on the locations of P1 and P2 and the range of parameters in the SC instruction. Using the user-unit P1 and P2 locations just given, the X-axis would contain 1000 user units, the Y-axis 9000. If you change the location of P1 and P2, the number of user units along each axis remains the same although their size has probably changed.

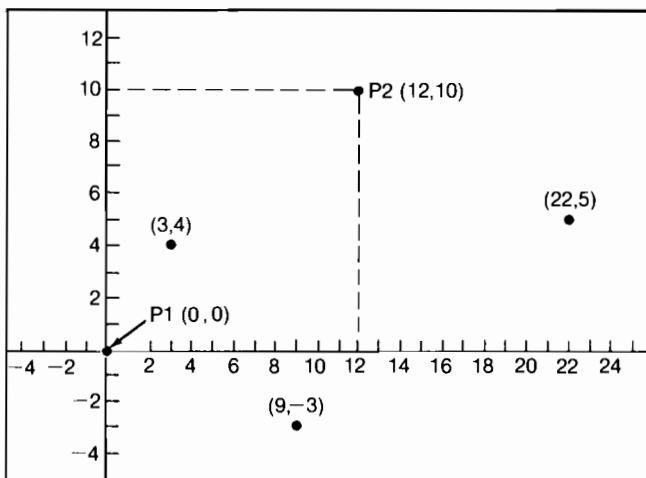
NOTE: When the size of the user units in the X-axis and the Y-axis is identical, the scaling is called isotropic. When the size of user units are not the same, the scaling is called anisotropic. If scaling is anisotropic, the plotter draws a circle as an ellipse. Refer to Chapter 3 for more information on isotropic and anisotropic scaling. ■

Once scaled, the plotting area is divided into the desired units. Subsequent plotting instructions will now be interpreted using these units. This means, 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). You can see this in the following illustration where the X-axis is scaled from 0 to 12 and the Y-axis from 0 to 10. Note too that scaling has moved the origin to the same location as P1.



User-Unit Scaling with Default P1 and P2

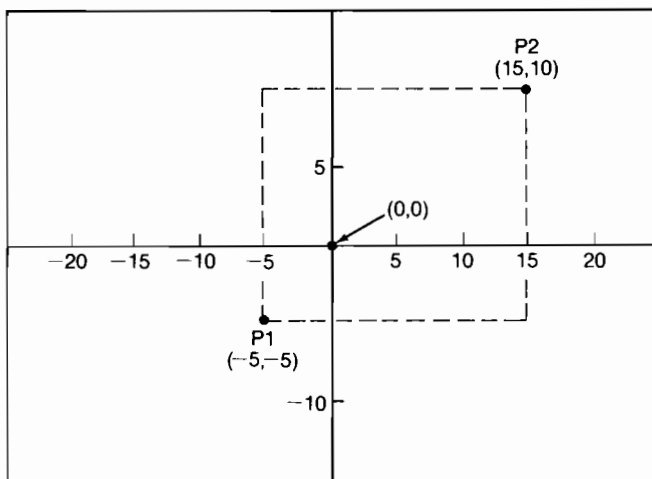
If you move the locations of P1 and P2, the *size* of user units in the plotting area may change. The previous illustration showed P1 and P2 in their default locations. In the following illustration, the locations of P1 and P2 are changed, but the scaling remains the same: the X-axis still contains 12 user units, the Y-axis 10 user units. Note that the actual sizes of the user units has decreased.



Same User-Unit Scaling with New P1 and P2

From this illustration it is easy to see that the size of user units is dependent on the locations of P1 and P2 and that the entire plotting area is scaled; not just the area defined by P1 and P2. Plotting is not limited by P1 and P2; only by hard-clip or soft-clip limits.

The following shows new locations for P1 and P2 and new scaling values. Note that the X-axis is scaled from -5 to 15 and the Y-axis from -5 to 10 . Compare this with the previous two illustrations and notice how the origin seems to move with respect to P1 and P2 with different scaling values.



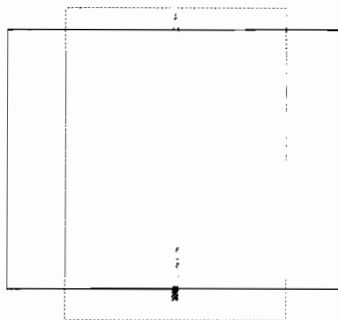
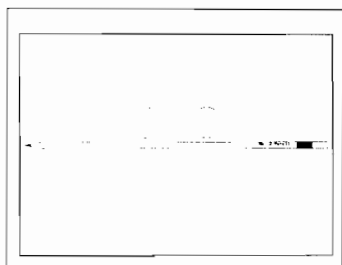
New P1 and P2 and User-Unit Scaling with Negative Values

An obvious benefit of user-unit scaling is that you can change the P1/P2 settings using the front-panel buttons so that your plot occupies more or less space on the paper. Contrast this flexibility with plotting in plotter units: since the size of a plotter unit never changes and the origin maintains a fixed physical location, plots always occupy the same space on any paper.

Rotation

You can change the plot's orientation on the paper by rotating the coordinate system 90 degrees about the origin. Normally, the X-axis runs the length of the paper. Use the RO (Rotate) instruction to programmatically rotate the coordinate system, including the scaling points, so that the Y-axis runs the length of the paper. Note that when P1 and P2 are in their default locations, the RO instruction rotates the scaling points outside the hard-clip limits. (This differs from the front-panel **ROTATE** button, which rotates the axes *and* places P1 and P2 inside the hard-clip limits.)

The following illustrations show a plot in its default orientation and after programmatically rotating its axes. Note that the RO instruction does not affect the isometric scaling of the circles. This is because P1 and P2 keep their logical values as they rotate with the coordinate system and so maintain scaling. Note, too, that extremities of the rotated plot may extend beyond the hard-clip limits.



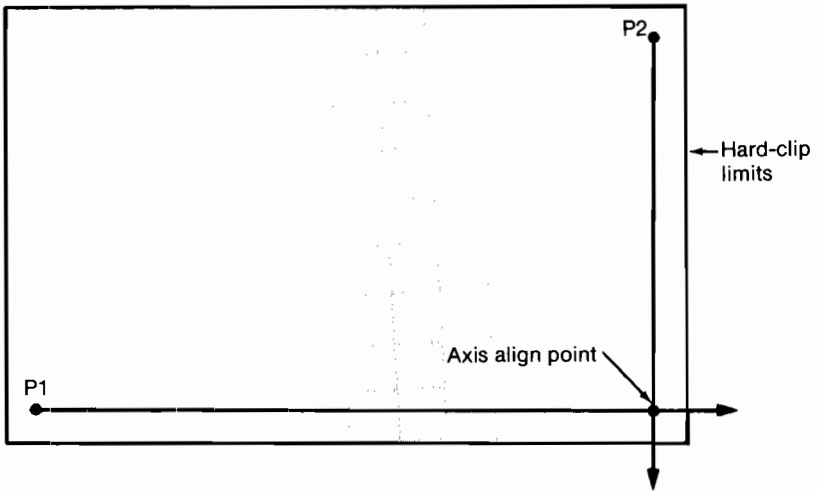
Note that the RO instruction is not identical with the front-panel **ROTATE** button. The front-panel button rotates the coordinate system and automatically moves P1 and P2 within the hard-clip limits. The RO instruction rotates the coordinate system only; it has no effect on the logical locations of P1 and P2.

Refer to Chapter 10 for more information regarding the RO instruction.

Axis Align

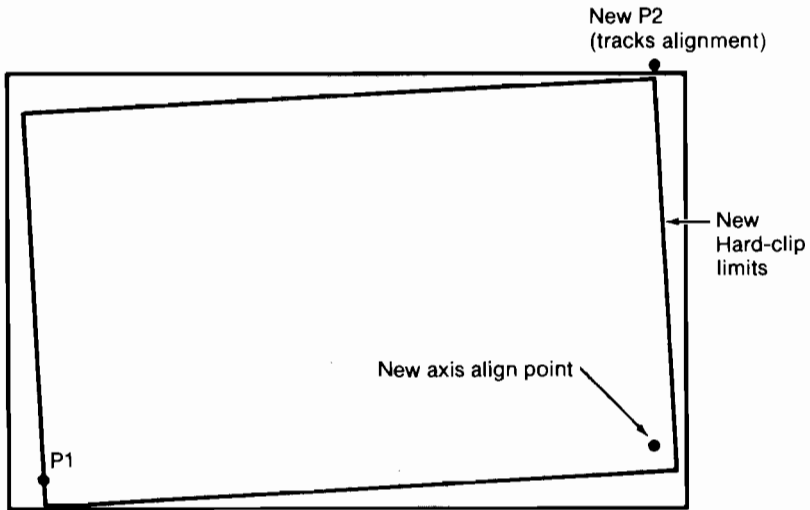
When you are using paper with preprinted lines or grids, you will want to use the front-panel **AXIS ALIGN** button to match the plotting axes to the preprinted grids. Use the axis alignment procedures described in the User's Guide. The following paragraphs detail how axis align works.

At power-on, the plotter sets P1 and P2 600 plotter units within the hard-clip limits. The axis align point is located at the intersection of a line parallel to the X-axis that passes through P1 and a line parallel to the Y-axis that passes through P2.



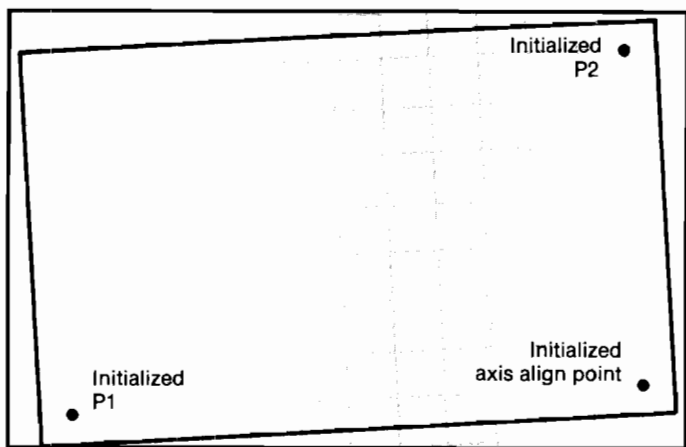
Default Location of Axis Align

When you establish a new axis align point (using the procedures described in the User's Guide), you are changing the angle between P1, the axis align point, and the physical hard-clip limits. The scaling point P2 moves relative to this new angle. The new angle also establishes new hard-clip limits that fit completely within the physical hard-clip area. Note that although you are changing physical locations of these points, they retain their logical values (i.e., their X,Y coordinate values do not change).



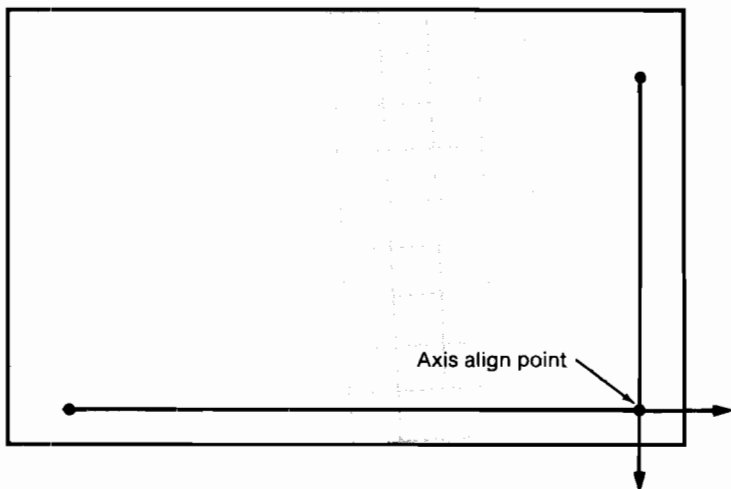
Effect of New Axis Align Point

When you subsequently initialize the plotter by sending the IN (Initialize) instruction or performing a front-panel reset, the locations of P1, P2, and the axis align point are all set 600 plotter units within the compressed hard-clip limits. (The physical locations and logical values change.)



Effect of Initialization on New Axis Align Point

When you load new paper, the plotter cancels the angle of alignment and sets the hard-clip limits to the value determined by the **EXPAND** switch. The axis align point is located at the intersection of a line parallel to the X-axis that passes through P1 and a line parallel to the Y-axis that passes through P2.



Axis Align Point After Initializing Then Loading Paper

To return all values to their default locations without turning off the plotter, you must load new paper, then send the IN instruction or perform a front-panel reset. This cancels any alignment angle before IN sets P1, P2, and the axis align point 600 plotter units inside the hard-clip limits.

Notes



CHAPTER

3

Preliminary Setup Using HP-GL

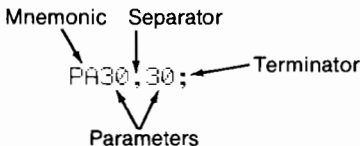
This chapter contains guidelines and instructions necessary to begin programming. You will learn HP-GL syntax conventions and begin using some of the instructions. This chapter also provides the following information.

- default conditions
- plotter initialization
- effective scaling procedures

Understanding HP-GL Instruction Syntax

Each HP-GL instruction is a two-letter mnemonic designed to remind you of its function. For example, SP is the Select Pen instruction. Many instructions use parameters to help define the function. For example, the SP instruction includes a parameter to specify the pen's carousel stall number.

HP-GL syntax is a set of rules that define how instructions must be written. The following is a typical HP-GL instruction in its basic form.



Each HP-GL mnemonic begins the instruction. You can write the mnemonic using either upper- or lowercase; this manual will always show the mnemonic uppercase. You need not separate the first parameter from the mnemonic, but you must separate multiple parameters from each other by a comma or a space. You end, or terminate, each instruction using a semicolon, the first letter of the next mnemonic, or, if you are using an HP-IB interface, a line feed. The parameter separator and instruction terminator are the only punctuation required. Each of the following instruction forms are valid. For clarity, the examples in this manual will use the first form.

PD;PU10,20; or PD PU10 20; or PDPU10,20;

Notations

Each instruction will list its syntax requirements using the following notations.

MNemonic the mnemonic is shown uppercase and separated from the parameter(s) or terminator

parameter parameters are listed in lowercase italics

() all parameters within parentheses are optional

(c...c) any number of characters

(...) any number of additional parameters of the specified type

;
HP-GL instruction terminator

[TERM] the output response terminator (see chapters 10 through 12)

Parameter Formats and Ranges

Each parameter must be specified in the format defined by the respective HP-GL instruction; the syntax format accompanies each instruction description throughout this manual. Parameter tables are also included to quickly define the parameters listed in the syntax. HP-GL instructions use three types of parameters: integers, real numbers, and character strings.

- **Integers** — A whole number from **-32 768** to **32 767**. Refer to the following section, *A Note about Rounding*, to determine how the plotter reacts to a decimal fraction included with an integer parameter.
- **Real numbers** — A number from **-32 768.0000** to **32 767.9999**. The decimal point and fraction (up to four significant digits) are optional.
- **Character strings** — Any sequence of one or more characters. Characters and character strings are used in drawing symbols and labeling plots. For more information, refer to Chapters 7 and 8.

The term “current units” in a parameters table indicates that the plotter interprets coordinate values as plotter units when scaling is off, or as user units when scaling is on. However, when using relative coordinates and scaling is off, it is a good idea to use only integers as the plotter monitors the fractional portion. Refer to *Using Absolute and Relative Movement* in Chapter 4.

A Note about Rounding

Some HP-GL instructions require that their parameter be an integer regardless of whether scaling is on or off. When the parameter must be an integer, the plotter rounds any included decimal fraction. If the first decimal digit is 5 or more, the plotter increments positive numbers by 1 and decrements negative numbers by 1. For example, the plotter rounds 4.6 to 5, and -3.7 to -4.

When the instruction interprets parameters in current units and scaling is off, the plotter truncates the decimal portion. For example, 4.6 is truncated to 4, and -3.7 is truncated to -3.

Syntax Compatibility with Other HP Plotters

The syntax rules implemented on the HP 7570A are extremely flexible and backwards compatible with the HP ColorPro, HP 7475, 7550, 7580, 7585, and 7586 graphics plotters. However, if you are interested in writing software to drive all of these plotters, you should note a minor difference in syntax: The 7570A and all of the above plotters allow a comma or space as a separator. *However, only the HP 7570A, HP ColorPro, and HP 7475 additionally allow a + or - sign as a separator.*

Omitting Optional Parameters

Some instructions have optional parameters. When you omit an optional parameter, it assumes a default value. For most HP-GL instructions, if you omit an optional parameter, you must omit **all subsequent parameters in the same instruction**. (The only exception is the pen control parameter in the UC (User-defined Character) instruction.)

For example, the FT (Fill Type) instruction has three optional parameters. The following shows one way to use all three parameters.

```
FT3,10,45;
```

If you omit the second parameter, you must omit the third. That is, you can send the following.

```
FT3;
```

The previous instruction sets default values for the second and third parameters. However, to specify a specific value for the third parameter, you must specify all of the previous parameters in the instruction. You cannot send either of the following instructions to get a default value for the second parameter and the specified value for the third parameter.

```
FT3,45; or FT3,,45;
```

Setting the Plotter to Known Conditions

Many plotter functions are set to known conditions by turning on the power switch or by resetting the plotter (refer to the User's Guide). Both of these procedures "initialize" the plotter. Most of these known conditions can be established by including the IN (Initialize) or DF (Default) instruction at the beginning of each program. The IN instruction is more powerful than the DF instruction, in that it sets more conditions.

The following summarizes the two programmatic methods of setting known conditions on the plotter.

- Use the DF instruction when you want to reestablish a limited number of plotter conditions while retaining certain conditions set in a previous program or by the front-panel buttons. For example, the DF instruction does not affect the current locations or values of P1 and P2. This allows you to retain P1 and P2 values from a previous program, or set them from the front-panel.
- Use the IN instruction to reset most of the plotter's functions to known conditions. For example, the IN instruction automatically sets P1 and P2 600 plotter units within the hard-clip limits.

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

Using the Scaling Instruction

If you are new to HP-GL programming, you will want to read *Scaling* in Chapter 2 before continuing.

Two instructions are used when scaling a picture. These are the SC (Scale) and IP (Input P1 and P2) instructions. The SC instruction defines the user units; the IP instruction establishes the locations of the scaling points P1 and P2. Since the size of user units is controlled by the distance between P1 and P2, it is always important to know the location of the scaling points.

Working with the Scaling Points

When the plotter is turned on or initialized, P1 and P2 are set 600 plotter units within the hard-clip limits. When you issue an SC instruction with parameters, the plotter converts P1 and P2 plotter unit coordinates to user unit coordinates. For example, if you scale the X-axis to range from 0 to 500 and the Y-axis to range from -100 to 100, P1 takes on the value of 0, -100 and P2 takes on the value of 500, 100. The SC instruction does not change the locations of P1 and P2, only their coordinate values. It is important to note that scaling is not limited to the rectangular area defined by P1 and P2, but extends over the entire plotting area.

Effective Scaling

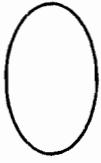
You can also use scaling to expand or contract your plots using one of two methods. The first method retains the locations of P1 and P2 while changing the number of units along both of the axes. Increase the number of units along both axes to draw a smaller plot with a larger border, or decrease the number of units along both axes to enlarge the plot.

The second method changes the locations of P1 and P2 while retaining the original user-unit scaling. By using the IP instruction to define a larger or smaller scaling rectangle, you can use the same coordinates to change the size of your original plot. By changing the locations of P1 and P2, you can change the size of the user units without changing the quantity of units along both axes.

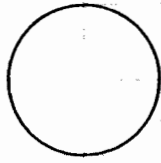
Note that you can lose portions of your plot by enlarging it beyond the hard-clip limits using either of the two methods above. For this reason, you should have your initial program draw your plots at their maximum size, so you can contract the size of your plots and not lose information.

Isotropic and Anisotropic Scaling

When establishing scaling, consider whether your picture will include such geometric shapes as circles and wedges where the height-to-width ratio must be preserved. When incorporating such shapes, you must first establish isotropic scaling. In isotropic scaling, the spacing of units along the X-axis is the same as unit spacing along the Y-axis. In this way, shapes such as circles and wedges are drawn without distortion. When the scaling is anisotropic, a circle becomes an ellipse. Compare the circles in the following illustration. Note that the radius for each of the circles is the same, only the scaling changes.



Anisotropic
1 Y-unit > 1 X-unit



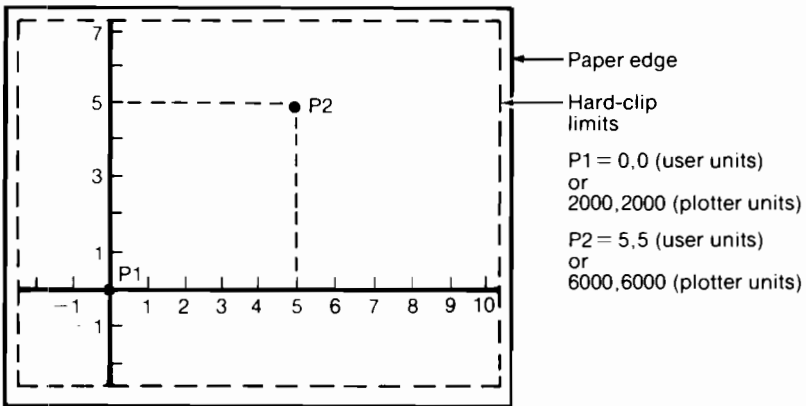
Isotropic
1 X-unit = 1 Y-unit



Anisotropic
1 X-unit > 1 Y-unit

Perhaps the easiest method to establish isotropic scaling is to use the IP instruction to set P1 and P2 so that they define a square area. Then use the SC instruction to establish the same number of user units along both axes as in the following illustration.

```
“IP2000,2000,6000,6000;”  
“SC0,5,0,5;”
```

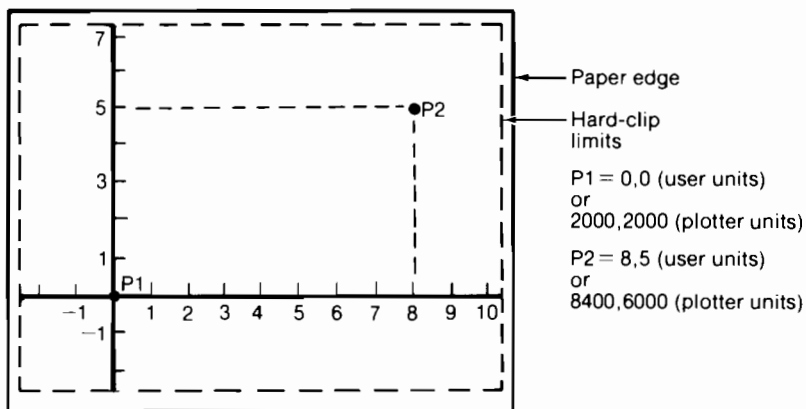


Isotropic Scaling: Method 1

When P1 and P2 do not define a square area, you can scale the area isotropically by setting your scale so that each user unit along the X- and Y-axes equals the same number of plotter units. For example, suppose P1 is at the location 2000, 2000 and P2 is at the location 8400, 6000 and you know you want 5 user units along the Y-axis; how many units do you specify along the X-axis to establish isotropic scaling? The Y-axis distance is 4000 plotter units (6000 - 2000), meaning that each of the 5 user units along

that axis is 800 plotter units ($4000/5$). The X-axis distance is 6400 plotter units ($8400 - 2000$), so the number of units along the X-axis is $6400/800$ or 8 user units.

```
“IP 2000,2000,8400,6000;”  
“SC 0,8,0,5;”
```



Isotropic Scaling: Method 2

Refer to the SC and IP instructions later in this chapter for additional information.

DF, Default

USE: Sets certain plotter functions to predefined default conditions. Use this instruction to return the plotter to a known state while maintaining the current rotation and settings of P1 and P2. When DF is used 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
Pen control	AP;	Lifts or stores a motionless pen after 15 seconds. Selects a pen only when it is required to draw. Pen sort is disabled. If previously enabled, the plotter executes the buffer before proceeding.*
Alternate set	CA;	ANSI ASCII English (Character set 0).
Standard set	CS;	ANSI ASCII English.
Chord tolerance	CT;	Chord angle of 5 degrees.
Digitize clear	DC;	Terminates digitize mode and turns off front-panel ENTER .
Direction absolute	DI1,0;	Horizontal characters.
Label terminator	DT;	ETX, ASCII decimal code 3.
Direction vertical	DV;	Off (set to horizontal).
Extra space	ES0,0;	No extra character space.
Fill type	FT;	Type 1, solid bidirectional fill. Spacing is 1% of P1/P2 X-axis distance. Angle is at zero degrees.
Input mask	IM233,0,0;	Recognizes all defined errors.
Input window	IW;	Set to hard-clip limits.
Label origin	LO1;	Standard labeling starting at current location.
Line type	LT;	Solid line (pattern length is 4% of P1/P2 diagonal distance).
Plotting mode	PA;	Absolute.

*Valid only when rear-panel pen sort switch is off

(Table continues)

Function	Equivalent Instruction	Default Condition
Polygon mode	PM0; PM2;	Polygon buffer cleared.
Pen thickness	PT;	0.3 mm.
Scaling	SC;	User-unit scaling off.
Character size	SI;	Absolute size: Width = 0.285 cm Height = 0.375 cm
Character slant	SL;	No slant.
Symbol mode	SM;	Off.
Character set selected	SS;	Standard character set selected.
Tick length	TL;	Positive and negative tick marks are 5% of $ P2_x - P1_x $ and $ P2_y - P1_y $.

In addition, the DF instruction updates the carriage-return point for labeling instructions to the current pen location. (Refer to *Positioning Labels* in Chapter 8 for more information regarding the carriage-return point.)

The DF instruction does not affect the following plotter conditions:

- locations of P1 and P2
- current pen, its location, and up/down position
- pen speed
- rotation
- buffer sizes (reestablish defaults using ESC . T or GM, Graphics Memory, instructions)
- generated errors (not cleared)
- RS-232-C handshaking protocol

The plotter automatically executes a DF whenever a sheet of paper is loaded that differs from the previous sheet in either dimension by more than 125 plotter units (approx. 1/8 inch).

RELATED

INSTRUCTIONS: IN, Initialize

ERROR:

Condition	Error	Plotter Response
1 or more parameters	2	executes instruction

IN, Initialize

USE: Resets most plotter conditions to their initial power-on conditions. Use this instruction to return the plotter to a known state and to cancel conditions that may have been set in a previous program.

SYNTAX: IN;

REMARKS: The IN instruction uses no parameters and sets the plotter to the same conditions as the DF instruction, with the following *additional* conditions.

- raises the pen.
- sets P1, P2, and the axis-align point 600 plotter units within the current hard-clip limits.

When you establish a new axis align point, the plotter establishes new hard-clip limits. To subsequently send IN sets P1, P2, and the axis align point 600 plotter units within the *new* hard-clip limits.

- sets bit position 3 of the status byte to 1 (to indicate the plotter has been initialized).
- clears any HP-GL error and I/O error conditions.

The IN instruction does not affect the pen speed or rotation.

NOTE: Buffer sizes established by ESC.T or GM (Graphics Memory) instructions are **not** affected by the IN instruction. If you have changed buffer sizes, reestablish the default buffer sizes by using an ESC.T or GM without parameters at the end of the program. ■

In this manual, program examples begin with “IN;” to clear unwanted conditions from the previous program. *This instruction has no effect on handshake protocol in any RS-232-C environment, but does clear any existing I/O error condition.*

ERROR:

Condition	Error	Plotter Response
1 or more parameters	2	executes instruction

IP, Input P1 and P2

USE: Establishes new or default locations for the scaling points P1 and P2. P1 and P2 are used by SC to establish user-unit scaling. The IP instruction is often used to ensure that a plot is always the same size, regardless of how P1 and P2 might have been set from the front panel or the size of paper loaded in the plotter. You can also use this instruction in advanced techniques such as plotting mirror images, enlarging/reducing plots, and enlarging/reducing relative character size or direction (refer to Chapter 10).

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

Parameter	Format	Range	Default
$P1_X, P1_Y$ ($P2_X, P2_Y$)	integer	-32 768 to 32 767 plotter units	depends on paper size

REMARKS: The plotter interprets the parameters as follows.

- $P1_X, P1_Y, (P2_X, P2_Y)$ — specify the location of P1 and P2 in plotter units. If you specify only the coordinates for P1, then P2 tracks P1 (retains the same relative distance from P1) and

could become located outside the hard-clip area. However, the P2 tracking function can be useful when preparing more than one equal-sized plot on one page. For an example, refer to *Changing Picture Area and Orientation* in Chapter 10.

- **no parameters** — sets P1 and P2 600 plotter units within the hard-clip limits, relative to the current axis rotation.

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

Refer to the discussions on scaling in Chapter 2 and earlier in this chapter for details of the effect P1 and P2 locations have on user-unit scaling.

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

NR, Not Ready

USE: Programmatically simulates pressing the front-panel **VIEW** button.

SYNTAX: NR;

REMARKS: After executing the NR instruction, the front-panel **VIEW** indicator is lighted and plotting is suspended. You must press the front-panel **VIEW** button to resume plotting on the same sheet. Loading a new sheet of media will also exit this state.

SC, Scale

USE: Establishes a user-unit coordinate system by mapping user-defined values onto the scaling points P1 and P2. For conceptual information on scaling, refer to Chapter 2 and earlier in this chapter.

SYNTAX: SC $X_{min}, X_{max}, Y_{min}, Y_{max}$; or SC;

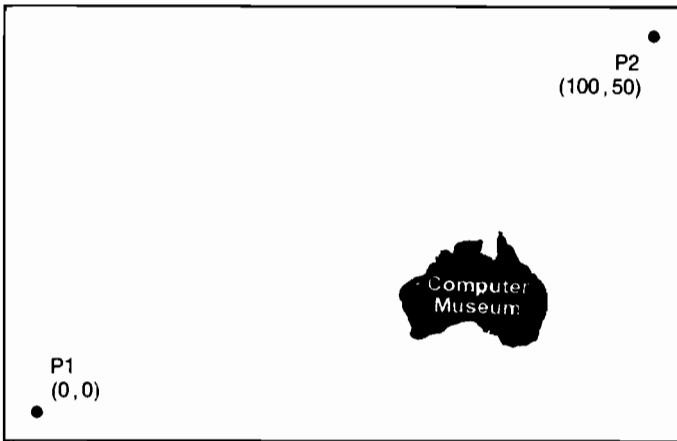
Parameter	Format	Range	Default
$X_{min}, X_{max},$ Y_{min}, Y_{max}	integer	-32 768 to 32 767 user units	none

REMARKS: The plotter interprets the parameters as follows.

- $X_{min}, X_{max}, Y_{min}, Y_{max}$ — set the minimum and maximum values for user units in the respective axis. You must specify all four parameters. The following instruction indicates that the X-axis will range from 0 to 100 user units, and the Y-axis will range from 0 to 50 user units.

SC 0 , 100 , 0 , 50 ;
 └──┬──┘ └──┬──┘
 X-axis range Y-axis range

These values are then mapped onto the scaling points P1 and P2. The minimum X- and Y-coordinates (the first and third parameters) are mapped onto P1; the maximum X- and Y-coordinates (the second and fourth parameters) are mapped onto P2. Using the example above, P1 would take on the coordinate value 0,0 and P2 would take on the coordinate value 100,50.



The SC instruction does not change the physical locations of P1 and P2, it only maps the parameter values onto those points. P1 and P2 retain these values until another SC instruction defines a new set of values or turns scaling off. After scaling, you can change the actual size of a user unit by altering the relative distance between P1 and P2, or by changing the parameters of the SC instruction.

- **no parameters** — turns off scaling so that the plotter interprets all subsequent coordinates as plotter units.

The plotter cannot interpret the parameters of all instructions as user units when scaling is on. Each instruction that can interpret parameters as user units includes the words “current units” in its parameter table. Specify the parameters for all other instructions in the units indicated by the instruction description. When scaling is on, the plotter interprets the parameters of the instructions on the next page in user-unit values.

AA, Absolute Arc
 AR, Relative Arc
 CI, Circle
 EA, Edge Absolute Rectangle
 ER, Edge Relative Rectangle
 EW, Edge Wedge
 FT, Fill Type
 IW, Input Window
 PA, Plot Absolute
 PD, Pen Down
 PR, Plot Relative
 PU, Pen Up
 RA, Rectangle Absolute
 RR, Rectangle Relative
 WG, Wedge Fill

Additionally, the response to the OC (Output Commanded Status), OD (Output Digitized Point), and OW (Output Window) instructions are interpreted in user units when scaling is on.

Note, when scaling is on, that you can specify fractions of plotting units. Decimal parameters are *not truncated*: The point 6.5, 9.5 is distinct from 6.6, 9.8.

Normally you will specify smaller coordinate values for P1 (X_{\min} and Y_{\min}) than you do for P2 (X_{\max} and Y_{\max}). However, you can specify X_{\min} and Y_{\min} larger than X_{\max} and Y_{\max} respectively and draw your plot as a mirror image — reversed and/or upside down — depending also on the relative locations of P1 and P2. Refer to Chapter 10 for mirror imaging plots.

ERRORS:

Condition	Error	Plotter Response
more than 4 parameters	2	ignores additional parameters
1 to 3 parameters	2	ignores instruction
$X_{\min} = X_{\max}$, $Y_{\min} = Y_{\max}$ or parameter out-of-range	3	ignores instruction

Notes

CHAPTER

4



Drawing Lines

To create accurate and effective plots, you must understand how to select a pen and control it when plotting. Controlling the pen means that you can raise and lower the pen, and move or draw from one location (X,Y coordinate) to another. This chapter will discuss the following.

- selecting and changing pens
- monitoring the pen up/down position
- moving to absolute or relative locations
- changing pen speed
- plotting with variables

Selecting and Controlling the Pen

You can select and change pens using either the front-panel buttons (as described in the User's Guide) or using the SP (Select Pen) instruction. This plotter also partially implements the SG (Select Pen Group) instruction, which can also be used to select a pen; however, SP is more commonly used for this function. Both instructions are described in detail later in this chapter.

Note that any pen in the holder that is inactive for 15 seconds is returned to the carousel; however, you can turn off this and other features using the AP (Automatic Pen Operations) instruction.

Monitoring the Pen Position and Location

When you turn on the plotter, it is initialized and set to certain default conditions. Among these conditions are the pen position and location. The pen position refers to whether the pen is up or down; the pen location refers to a coordinate location on your paper. As you draw, the plotter updates the pen location and position depending on the instruction just executed. Some instructions return the pen to its most recent location and position before the instruction was executed. Other instructions update the plotter with the location and position following the instruction execution.

At power on, the pen holder is up. To draw lines, you must select a pen, lower it to the paper, then move the pen to a new location. You can perform each of these functions using the front-panel buttons; but for most applications, this is impractical. You can programmatically lower the pen onto the paper using the PD (Pen Down) instruction and raise the pen using the PU (Pen Up) instruction.

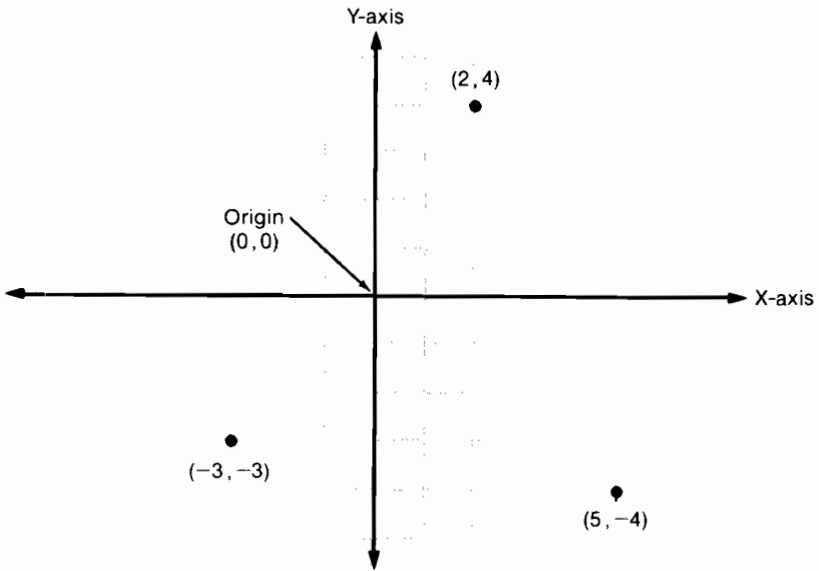
Note that some instructions include an automatic pen down feature; that is, the plotter automatically lowers the pen before executing the instruction, and then restores the pen to its previous up/down position. Refer to an instruction description to determine whether it includes an automatic pen down feature.

At power on, the plotter sets the pen holder just within the hard-clip limits at a corner of your paper. If you were to select a pen, issue the PD instruction, then move the pen to a new location, your plot would begin at this corner. To begin your plot at a specific location, use either the PA (Plot Absolute) or PR (Plot Relative) instructions. Use the PA instruction when you want to move the pen with respect to the coordinate origin. Use the PR instruction when you want to move the pen *relative* to the current, updated pen location.

Absolute and Relative Movement

So far all discussions have assumed that you are specifying coordinates in *absolute* terms. Absolute means that the coordinates have a fixed position on the coordinate system with respect to the origin (0, 0). In the following illustration, the coordinates

$-3, -3$; $2, 4$; and $5, -4$ are always in the same place, no matter where the pen is when the coordinates are issued.



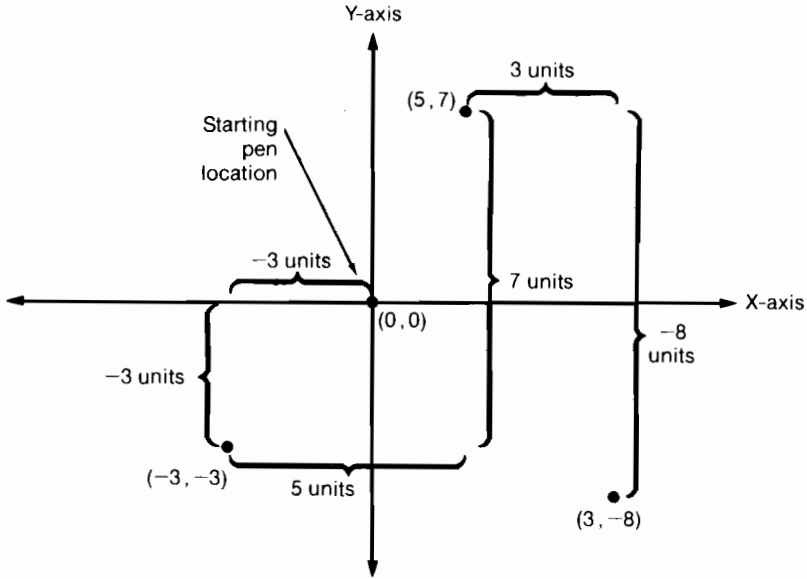
Absolute Coordinates

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.

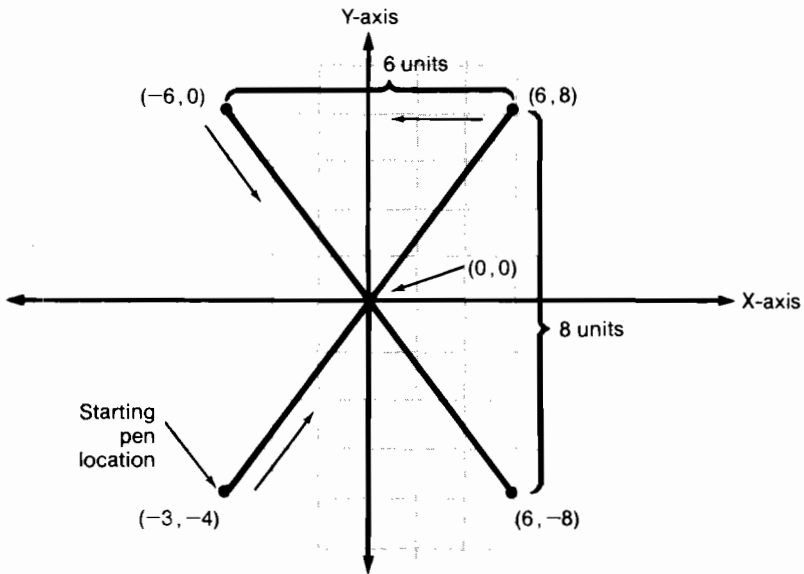
For example, assume that the pen is currently at the origin. To arrive at each location marked from left to right in the previous illustration, count 3 units to the left and 3 units down from the current pen location; these are both negative directions with respect to the origin. This is the relative 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 X-units and -8 Y-units (3, -8).



Relative Coordinates

Plotting instructions that include “relative” in their title will interpret X,Y coordinate parameters as relative increments. Relative plotting is particularly useful when you want to draw the same shape in different locations. It is also useful for shapes where you know the dimensions, but don’t want to calculate the absolute coordinates for each endpoint. For example, suppose you want to plot an “X” with a width of 6 units and a height of 8 units, and the current pen position is at the origin. Here is a sequence of relative coordinates that draw the X: -3, -4; 6, 8; -6, 0; 6, -8.



Relative Coordinates for Plotting X

A Note on Relative Movement when Scaling is Off

The plotter adds relative coordinates (increments) to the current pen location monitored by the plotter. The plotter automatically converts the new relative position to its absolute coordinates and updates the current monitored location in absolute coordinates.

The plotter will not move in decimal fractions of plotter units. The pen will move in integer increments only. However, the plotter monitors the complete decimal number even when scaling is off. You may affect future pen moves if you specify a series of pen moves which include fractional plotter units when scaling is off.

For example, if you tell the plotter to move relatively $0.6, 0.6$ plotter units, the pen will not move; the integer position is $0, 0$. If you then tell the plotter to move relatively $0.7, 0.7$, the pen will move. The monitored cumulative position is the point $1.3, 1.3$ ($0.6 + 0.7$), but the pen will move to the point indicated by the integer portion, $1, 1$.

This is true when using instructions that include the word "relative" in their title.

Plotting with Variables

In some programming circumstances, it might make more sense (and save time) to use variables for the parameters of an HP-GL instruction rather than specific values. The same parameter restrictions apply to variables as to specific values (i.e., integer or real within the valid range). You must also send the required instruction mnemonics, separators and terminators just as you would if you were sending specific parameter values.

Chapter 1 discusses the usual way to send an HP-GL instruction with specific parameter values. There are many acceptable methods that depend on each computer's format and output statements. The following is one method of sending variables to the plotter.

```
PRINT #1, "PA";X;"",";Y;";"
```

Note that the instruction mnemonic, separator, and the terminator are included within quotation marks. It is important that you separate the X and Y variables with a literal separator, such as a space (" ") or comma (","); otherwise, the plotter cannot differentiate the two coordinates, generating an error.

This method will be most effective in conjunction with FOR...NEXT statements or your language's specific looping statements.

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 15 seconds (to prevent the pen from drying out). Turns off automatic pen operations when you are using the digitizing sight and reestablishes them when you are ready to resume plotting.

SYNTAX: AP *n*; or AP;

Parameter	Format	Range	Default
n	integer	0 to 31*	7†

*This is the practical range. The allowable range is $-32\,768$ to $32\,767$. We recommend that you only use numbers within the valid range to ensure proper automatic pen operation.

†This is the default value when the rear-panel pen sort switch is off. If the rear-panel pen sort switch is on, the default range is 24 to 31, and the default value is 31. When the rear-panel pen sort switch is on, any number less than 24 is changed to 24.

REMARKS: The plotter enables or disables pen operations depending on the value of the parameter. The parameter *n* can be zero or any combination of the other values listed next.

- **0** — disables automatic pen operations. The plotter does not lift pens until commanded by PU or the front panel. The plotter does not store pens until commanded by SP or the front panel. The plotter retrieves pens immediately when selected by SP instruction whether or not they are required to draw.
- **1** — lifts a stationary pen after 15 seconds.
- **2** — temporarily stores the pen if it has not moved in 15 seconds. The plotter will reselect the pen when plotting resumes. If the plotter is unable to put the pen away, it lifts the pen (if down).

NOTE: This operation only affects pens the plotter selects from the carousel. If you manually put a pen into the pen holder, the plotter will not attempt to store the pen. ■

- **4** — does not retrieve a pen selected by the SP instruction until the new pen is required to draw.
- **8** — merges all pen up moves. If your program raises the pen and makes several moves before lowering the pen, the plotter tracks these moves without actually moving the pen. (You can turn this feature off programmatically only when the rear-panel pen sort switch is off.)
- **16** — sorts and plots vectors by pen number. Note that you can turn on pen sorting even when the rear-panel switch is in the off position. You can turn pen sorting off programmatically only when the rear-panel pen sort switch is off.

To turn on any subset of the pen operations, add the values for the operations you want to activate. For example, *AP3*; will turn on only operations 1 and 2. *AP7*; turns on operations 1, 2, and 4. *AP31*; turns on all of the automatic pen operations.

RELATED

INSTRUCTIONS: SP, Select Pen
SG, Select Pen Group

ERROR:

Condition	Error	Plotter Response
parameter less than -32 768 or greater than 32 767*	3	ignores instruction

*We recommend that you use only positive numbers within the valid range to ensure proper automatic pen operation.

PA, Plot Absolute

USE: Establishes absolute plotting mode and moves the pen to specified absolute coordinates.

SYNTAX: PA X,Y (...); or PA;

Parameter	Format	Range	Default
X,Y	real	-32 768.0000 to 32 767.9999 current units*	none

*When scaling is on, the range may be smaller. Refer to *User Units* in Chapter 2.

REMARKS: The plotter interprets the parameters as follows.

- **X,Y** — specifies the absolute coordinate location *relative to the origin (0, 0)* to which the pen will move. The plotter interprets X,Y coordinates 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. Generally, when you include more than one coordinate pair, the pen moves to each point in the order given, using the pen's

current up/down position. This is modified when the rear-panel pen sort switch is on, or you use the AP instruction to merge pen up moves.

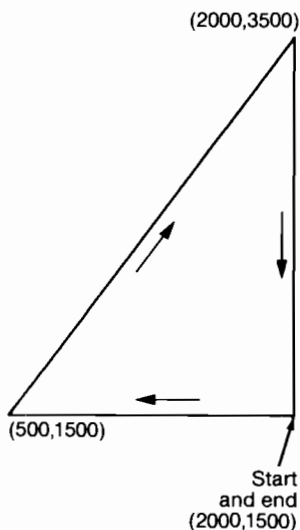
NOTE: When using an HP-IB interface, you may be limited to the number of coordinate pairs you can specify. The limitation is dependent on the ability of your controller to output long strings without a line feed. ■

- **no parameters** — establishes absolute plotting mode for subsequent instructions.

Execution of a PA instruction (with or without parameters) cancels relative plotting.

EXAMPLE: In the following, note that the PA instruction first moves the pen to a specific location (2000, 1500). After placing the pen down (PD), PA draws a triangle.

```
10 'Insert configuration statement here
20 PRINT #1, "IN;SP1;PA2000,1500;PD;"
30 PRINT #1, "PA500,1500,2000,3500,2000,1500;"
40 PRINT #1, "SP0;"
50 END
```



RELATED

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

ERRORS:

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

PD, Pen Down

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

SYNTAX: PD X,Y(...); or PD;

Parameter	Format	Range	Default
X,Y	real	-32 768.0000 to 32 767.9999 current units*	none

*When scaling is on, the range may be smaller. Refer to *User Units* in Chapter 2.

REMARKS: The plotter interprets the parameters as follows.

- **X,Y** — lowers the pen and draws to the points specified. Absolute or relative movement depends on the most recently executed PA or PR instruction. If no plot instruction has been previously specified, PA is assumed.

For example, if you execute a PD with parameters and either absolute plotting is in effect or you have not previously executed a plotting instruction, the plotter interprets the X,Y coordinates as absolute coordinates. Similarly, if relative plotting is in effect, the plotter draws to the relative point.

You can specify as many X,Y coordinate pairs as you want. Generally, when you include more than one coordinate pair, the pen moves to each point in the order given, using the pen's

current up/down position. This is modified when the rear-panel pen sort switch is on, or you use the AP instruction to merge pen up moves.

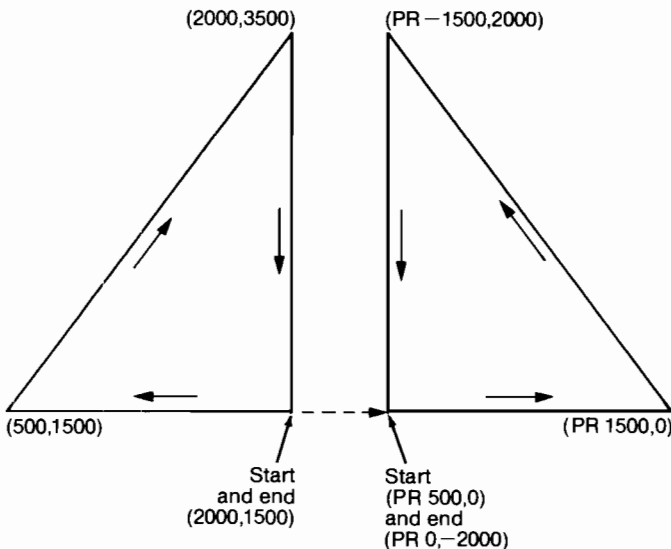
NOTE: When using an HP-IB interface, you may be limited to number of coordinate pairs you can specify depending on your controller's ability to output long strings without a line feed. ■

- **no parameters** — lowers the pen without moving it to a new location. This is the same as pressing the **PEN UP/DOWN** button on the front panel to lower the pen.

NOTE: The pen will lower only if it is within the current graphics limits (window). ■

EXAMPLE: In the following, the PD instruction is used with the PA (Plot Absolute) and PR (Plot Relative) instructions to draw two triangles. Compare this example with those for the PA and PR instructions.

```
10 *Insert configuration statement here
20 PRINT #1, "IN;SP1;PA2000,1500;"
30 PRINT #1, "PD500,1500,2000,3500,2000,1500;"
40 PRINT #1, "PU;PR500,0;"
50 PRINT #1, "PD1500,0,-1500,2000,0,-2000;"
60 PRINT #1, "SP0;"
70 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
parameter out of range	3	ignores wrong coordinate and all subsequent coordinates

PR, Plot Relative

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

SYNTAX: PR X,Y(...); or PR;

Parameter	Format	Range	Default
X,Y	real	-32 768.0000 to 32 767.9999 current units*	none

*When scaling is on, the range may be smaller. Refer to *User Units* in Chapter 2.

REMARKS: The PR instruction operates similarly to the PA instruction. The only difference is that PR moves are relative to the current pen position, whereas PA moves are absolute, with respect to the origin. The parameters are interpreted as follows.

- **X,Y** — specifies incremental moves relative to the current pen location. Each coordinate pair indicates a move relative to previous coordinate pair. The plotter interprets coordinates 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. Generally, when you include more than one coordinate pair, the pen moves to each point in the order given, using the pen's

current up/down position. This is modified when the rear-panel pen sort switch is on, or you use the AP instruction to merge pen up moves.

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

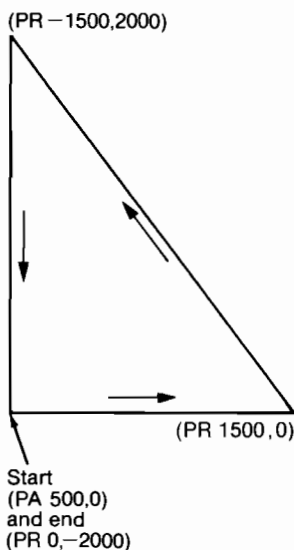
- **no parameters** — establishes relative plotting mode for subsequent instructions.

Execution of any PR instruction cancels absolute plotting.

When scaling is off, the pen will move in integer increments only. Nevertheless, the plotter still monitors the pen position using the complete decimal number. This can affect future pen positions. Refer to *A Note on Relative Movement when Scaling is Off* earlier in this chapter.

EXAMPLE:

```
10  *Insert configuration statement here
20  PRINT #1, "IN;SP1;PA1500,0;"
30  PRINT #1, "PD;PR1500,0,-1500,2000,0,-2000;"
40  PRINT #1, "SP0;"
50  END
```



RELATED

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

ERRORS:

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

*To stay within range when scaling is off, each succeeding X,Y increment when added to the current X,Y position cannot exceed 32 767 when referenced from point 0,0. For example, if the pen is at 6000,6000, the next relative coordinate must be within 26 767 (32 767 - 6000); otherwise, it is out of range.

PU, Pen Up

USE: Raises the pen from the plotting surface. Use this instruction to prevent stray lines from being drawn.

SYNTAX: PU X,Y(...); or PU;

Parameter	Format	Range	Default
X,Y	real	-32 768.0000 to 32 767.9999 current units*	none

*When scaling is on, the range may be smaller. Refer to *User Units* in Chapter 2.

REMARKS: The parameters are interpreted as follows.

- **X,Y** — raises the pen and moves to the specified points. Absolute or relative movement depends on the most recently executed PA or PR instruction. If no plot instruction has been previously used, PA is assumed.

For example, if you execute a PU with parameters and either absolute plotting is in effect or you have not previously executed a plotting instruction, the plotter interprets the X,Y coordinates as absolute coordinates. Similarly, if relative plotting is in effect, the plotter moves to the relative point.

You can specify as many X,Y coordinate pairs as you want. Generally, when you include more than one coordinate pair, the pen moves to each point in the order given. This is modified when the rear-panel pen sort switch is on, or you use the AP instruction to merge pen up moves.

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

- **no parameters** — The PU instruction raises the pen without moving the pen to a new location. This is the same as pressing the **PEN UP/DOWN** button on the front panel.

RELATED

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

ERRORS:

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

SG, Select Pen Group

USE: Selects a specified pen for plotting, or returns the current pen to the carousel. This instruction provides partial compatibility with other plotters and works the same as the SP (Select Pen) instruction.

SYNTAX: SG *pen number* ;

Parameter	Format	Range	Default
pen number	integer	0 to 8*	none

*This is the only range the plotter will accept and execute the instruction. The actual range is 0 to 32 767. Refer to the error table at the end of this instruction.

REMARKS: In other plotters, the SG instruction is used to alternate pens within predesignated groups. Since you cannot designate pen groups in this plotter, the SG instruction is functionally identical with the SP instruction.

RELATED

INSTRUCTIONS: SP, Select Pen

ERRORS:

Condition	Error	Plotter Response
parameter greater than 8 and less than 32767	none	ignores instruction
parameter less than 0	3	ignores instruction
parameter greater than 32767	3	ignores instruction

SP, Select Pen

USE: Moves pens between the carousel and the pen holder. Use this instruction to load a pen into the pen holder. You can also use it to select pens of different colors or widths during a plotting program. At the end of every program, use this instruction to return the pen to the carousel.

SYNTAX: SP *pen number*; or SP;

Parameter	Format	Range	Default
pen number	integer	0 to 8*	0

*This is the only range the plotter will accept and execute the instruction. The actual range is 0 to 32767. Refer to the error table at the end of this instruction.

REMARKS: The pen number parameter corresponds to the pen numbers marked on the pen carousel. The plotter interprets the parameter as follows.

- **pen number = 1-8** — selects the appropriate pen from the pen carousel. If there is currently a pen in the holder, the plotter stores it before selecting the new pen. After selecting a new pen, the pen holder returns to the current graphics location.

- **pen number = 0 or no parameter** — returns the pen currently in the pen holder to the 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.

An SP instruction remains in effect until a new pen is selected or the current pen is returned to a stall in the carousel. You can select and return pens by using the front-panel controls or by executing the SP instruction.

NOTE: If the automatic pen return feature of the AP instruction is on and the pen currently in the pen holder remains unused for 15 seconds, the plotter places the pen in the carousel and returns to its last location. When the plotter receives the next drawing instruction, the plotter selects the same pen (providing there was no intervening DF or IN instruction) before plotting. ■

RELATED

INSTRUCTIONS: AP, Automatic Pen Operation
SG, Select Pen Group

ERRORS:

Condition	Error	Plotter Response
parameter greater than 8 and less than 32 767	none	ignores instruction
parameter less than 0	3	ignores instruction
parameter greater than 32 767	3	ignores instruction

VS, Velocity Select

USE: Specifies pen speed. Use the instruction to optimize line quality and pen life for each pen and media combination. For optimum line quality, use a slow speed.

SYNTAX: VS *pen velocity* (*pen number*); or VS;

Parameter	Format	Range	Default
pen velocity	integer	1 to 40	40 cm/s
pen number	integer	1 to 8	all pens

REMARKS: The plotter interprets the parameters as follows.

- **pen velocity** — specifies the pen speed in centimetres per second (cm/s). You can increment the pen speed by 1 cm/s. The selected velocity applies only when the pen is down. Pen movement with the pen up is executed at 50 cm/s.
- **pen number** — applies the pen speed to a particular pen. When this parameter is omitted, the velocity applies to all pens.
- **no parameters** — sets the speed for all pens to the default value.

Note that you can use the front-panel **PEN SPEED** button with the pen number buttons to manually set the pen speed in 5 cm/s increments. However, the pen speed you specify using the front-panel buttons applies to all pens. Refer to the User's Guide for more information regarding the pen speed for your pen and media combination.

ERRORS:

Condition	Error	Plotter Response
pen velocity parameter greater than 40 cm/s	none	executed at 40 cm/s
pen number parameter greater than 8	none	instruction ignored
more than 2 parameters	2	extra parameter ignored
parameter less than or equal to 0	3	instruction ignored

Notes

CHAPTER

5





Drawing Circles, Arcs, Wedges, and Rectangles

You can use some individual instructions to draw complete geometric shapes. These shapes include circles, wedges, and rectangles. Additionally, you need only use a single instruction to draw a curved line segment, or arc. You can further enhance some of the shapes by filling them with shading patterns. This chapter discusses the instructions that draw these shapes and arcs, along with the following.

- controlling the smoothness of a circle or arc (chord tolerance)
- choosing a shading pattern (fill type)

Drawing Circles and Arcs

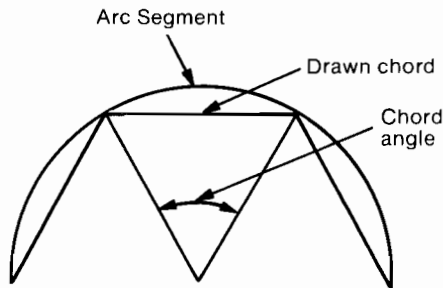
The circle instruction, `CI`, uses your current pen location as the center of the circle. All you need specify is the circle's radius because the `CI` instruction also includes an automatic pen down. Drawing an arc is only slightly more involved. Your current pen location becomes one end of the arc; you specify the center location (forming a radial distance) and the number of degrees through which you want the arc drawn. The `AA`, Arc Absolute, instruction draws arcs using absolute coordinates, the `AR`, Arc Relative, instruction draws arcs using relative coordinates.

Maintaining Circle/Arc Smoothness

The plotter draws curved lines, like those forming arcs and circles, as a series of straight lines called chords. Arcs and circles are found in the following instructions: AA and AR (Arc Absolute and Arc Relative), CI (Circle), EW and WG (Edge Wedge and Fill Wedge). The apparent smoothness of circles and arcs depends on the number of chords used to draw them. This is known as chord tolerance. Chord tolerance is determined in one of two ways: degrees mode or deviation distance mode. Degrees mode is the power-on default; use the CT, Chord Tolerance, instruction to change how chord tolerance is determined.

Degrees Mode

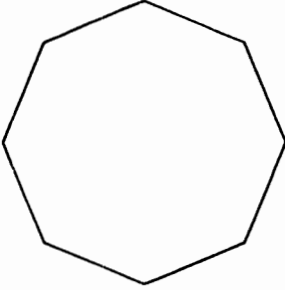
In degrees mode, you specify the maximum angle that defines each chord. This causes circles of any radius to be drawn with the same number of chords. As the radius increases, the chords become more apparent and the circle less smooth. The default chord angle is 5 degrees, which draws a circle with 72 chords ($360/5 = 72$).



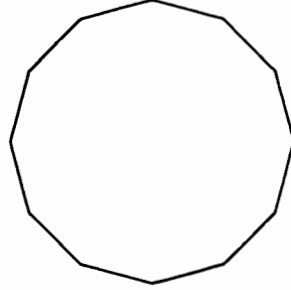
Degrees mode

Drawing Circles in Degrees Mode

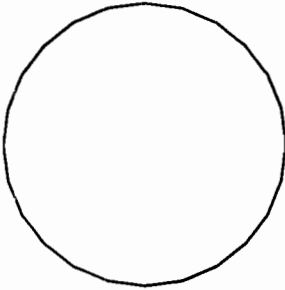
The following illustration demonstrates the effect the chord angle can have on circles of the same radius. Each of the circles was drawn using the CI instruction.



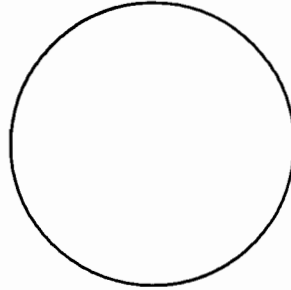
45-Degree chord angle



30-Degree chord angle



15-Degree chord angle

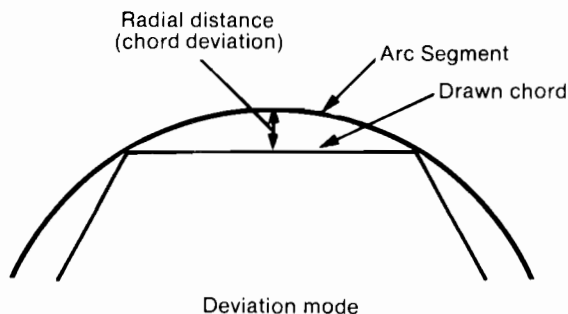


5-Degree chord angle

Note that the chords are more apparent in the circles that use a large chord angle. As the chord angle decreases, the circle becomes more smooth.

Deviation Distance Mode

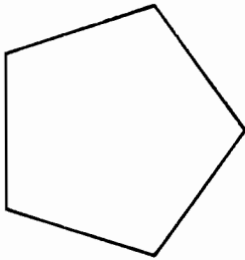
In deviation distance mode, you specify the maximum distance between the chord and the arc it represents. As the radius increases, then, more chords are required to draw the circle and maintain the distance between the chord and the arc. When you switch from degrees mode to deviation distance mode, the default deviation distance for any circle or arc is the same as if that circle were drawn with a chord angle of 5 degrees.



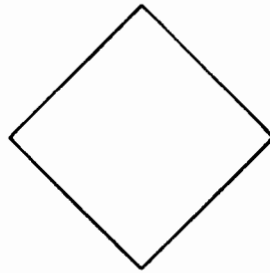
Drawing Circles in Deviation Distance Mode

How the Plotter Calculates Chords

Note that the value you specify for the chord tolerance parameter of any instruction is the *maximum* value in which the chord can be expressed. Rather than draw a circle with one partial chord, the plotter divides the circle so that it is drawn with complete chords. This is easier to understand when using degrees mode. For example, a chord angle of 89 and a chord angle of 90 give you dramatically different results because 89 is not a factor of 360. The closest factor of 360 that is less than 89 is 72.



chord angle of 89 (=72)



chord angle of 90

Similarly, chord angles of 90 and 91 produce the same shape since 91 is not a factor of 360.

NOTE: The plotter does not store changes in the chord angle parameter. If you want to maintain the same chord angle (other than the default angle) for each circle or arc in your plot, you must specify the chord angle with each instruction that includes the chord angle parameter. ■

Drawing Wedges

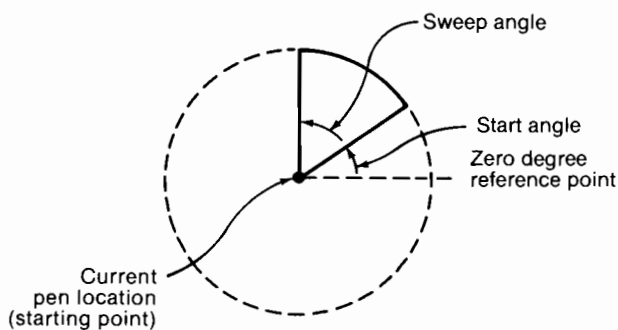
You can draw any sector of a circle using either of the two wedge instructions, EW (Edge Wedge) and WG (Wedge Fill). The EW instruction outlines a wedge, while the WG instruction draws a shaded wedge. Both instructions require the same information and use the plotter's polygon buffer to define the wedges before they are drawn. For these instructions to work properly, you must be sure the polygon buffer has enough memory to store the number of lines and chords that define the wedge.

The default polygon buffer size (1024 bytes) is large enough to store a wedge with 167 points. Refer to chapters 1, 6, and 13 for polygon buffer information, including how to count the points in a polygon and change the size of the polygon buffer.

Your current pen location is interpreted as the focal point of the wedge. From this point, you define a wedge specifying the following.

- the radius
- the start angle
- the sweep angle

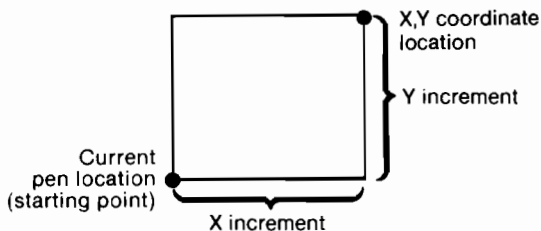
The radius determines the length of two sides of the wedge. The sign (positive or negative) of the radius determines the location of a “zero degree” reference point. The start angle is the number of degrees from the zero degree reference point at which you want to draw the first radius. The sweep angle is the number of degrees through which you want to draw the arc. You can optionally specify the arc’s chord tolerance. The following illustrates the different parameters of a wedge with a positive radius.



Drawing Rectangles

Four instructions give you great flexibility when drawing rectangles. Outline a rectangle using absolute or relative coordinates using the EA (Edge Absolute Rectangle) and ER (Edge Relative Rectangle) instructions. Draw a shaded rectangle using absolute coordinates or relative coordinates using the RA (Rectangle Absolute Fill) and RR (Rectangle Relative Fill).

All of the rectangle instructions use the same parameters. The plotter interprets your current pen location as one corner of the rectangle, all you need specify is the location of the opposite corner.

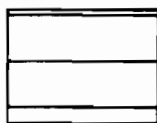


Choosing a Fill Type

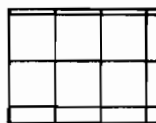
You can fill any closed geometric shape with a pattern. The FT, Fill Type, instruction lets you choose from three distinct patterns: solid, hatch, and cross-hatch.



Solid



Hatch



Cross-hatch

When using hatch and cross-hatch patterns, you can optionally change the spacing between the parallel lines, and the angle at which they are drawn. Both of these fill patterns are drawn using the current line type (refer to the LT, Line Type, instruction in Chapter 7). This allows you to create numerous fill patterns within a plot to increase its visual effectiveness.

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 tolerance)*;

Parameter	Format	Range	Default
X,Y	real	-32 768.0000 to 32 767.9999 current units*	none
arc angle	real	-32 768.0000 to 32 767.9999 (degrees)	none
chord tolerance chord angle chord deviation	real real	0.1 to 180 degrees† -32 768.0000 to 32 767.9999 current units	5 degrees equivalent to 5 degrees chord angle

*When scaling is on, the range may be smaller. Refer to *User Units* in Chapter 2.

†This is the practical range. The allowable range is -32 768.0000 to 32 767.9999. However, 0 is changed to 0.36; numbers from 180 to 360 are interpreted as $360 - n$; numbers greater than 360 are interpreted as n modulo 360 (the result following the above rules for numbers greater and less than 180).

REMARKS: The AA instruction does not include an automatic pen down feature; you must precede it with a PD instruction for the arc to be drawn. Arcs are drawn starting at the current pen location using the current line type. After drawing the arc, the pen location remains at the end of the arc, rather than returning to the beginning. The plotter interprets the parameters as follows.

- **X,Y** — specifies 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** — in degrees mode, specifies the maximum chord angle of the chords drawn to create the arc; in deviation distance mode, specifies the distance between the chord drawn and the arc it represents. For more information, refer to *Maintaining Circle/Arc Smoothness* at the beginning of this chapter.

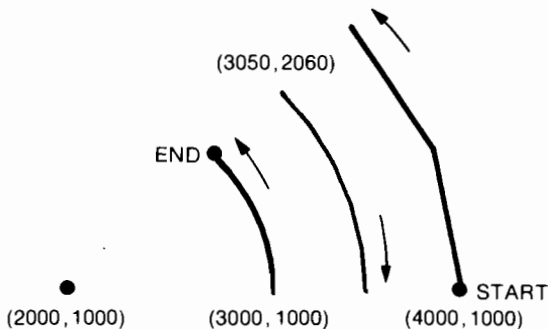
NOTE: The arc angle divided by the chord angle determines the number of chords in your arc (refer to *Maintaining Circle/Arc Smoothness* at the beginning of this chapter). If the number of chords in your arc is greater than 32767, the plotter will draw your arc with 1000 chords. This can occur when you draw an arc angle greater than 360 degrees using a small chord tolerance. ■

EXAMPLE: The following shows three arcs with the same center point, but different radii and chord tolerances. Also note the effect a negative arc angle parameter has on the direction the arc is drawn (line 40, center arc).

```

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, "PU;SP0;"
70 END

```



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

*This error is generated when the distance between the current pen location and the center point exceeds the -32768 to 32767 plotter unit range.

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 tolerance);*

Parameter	Format	Range	Default
X,Y	real	-32768.0000 to 32767.9999 current units*	none
arc angle	real	-32768.0000 to 32767.9999 (degrees)	none
chord tolerance chord angle	real	0.1 to 180 degrees†	5 degrees
chord deviation	real	-32768.0000 to 32767.9999 current units	equivalent to 5 degrees chord angle

*When scaling is on, the range may be smaller. Refer to *User Units* in Chapter 2.

†This is the practical range. The allowable range is -32768.0000 to 32767.9999 . However, 0 is changed to 0.36; numbers from 180 to 360 are interpreted as $360 - n$; numbers greater than 360 are interpreted as n modulo 360 (the result following the above rules for numbers greater and less than 180).

REMARKS: The AR instruction does not include an automatic pen down feature; you must precede it with a PD instruction for the arc to be drawn. Arcs are drawn starting at the current pen location using the current line type. After drawing the arc, the pen location remains at the end of the arc, rather than returning to the beginning. The plotter interprets the parameters as follows.

- **X,Y** — specify, relative to the current pen 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** — in degrees mode, specifies the maximum chord angle to create the chords that draw the arc; in deviation distance mode, specifies the distance between the chord drawn and the arc it represents. For more information, refer to *Maintaining Circle/Arc Smoothness* at the beginning of this chapter.

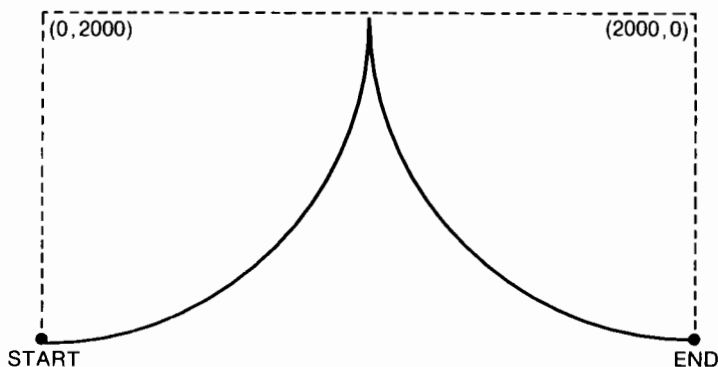
NOTE: The arc angle divided by the chord angle determines the number of chords in your arc (refer to *Maintaining Circle/Arc Smoothness* at the beginning of this chapter). If the number of chords in your arc is greater than 32 767, the plotter will draw your arc with 1000 chords. This can occur when you draw an arc angle greater than 360 degrees using a small chord tolerance. ■

EXAMPLE:

```

10  *Insert configuration statement here
20  PRINT #1, "IN;SP1;PA10,10;PD;"
30  PRINT #1, "AR0,2000,90;AR2000,0,90;"
40  PRINT #1, "PU;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

*This error is generated when the distance between the current pen location and the center point exceeds the -32 768 to 32 767 plotter unit range.

CI, Circle

USE: Draws a circle using the specified radius and chord angle.

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

Parameter	Format	Range	Default
radius	real	-32 768.0000 to 32 767.9999 current units*	none
chord tolerance chord angle	real	0.1 to 180 degrees†	5 degrees
chord deviation	real	-32 768.0000 to 32 767.9999 current units	equivalent to 5 degrees chord angle

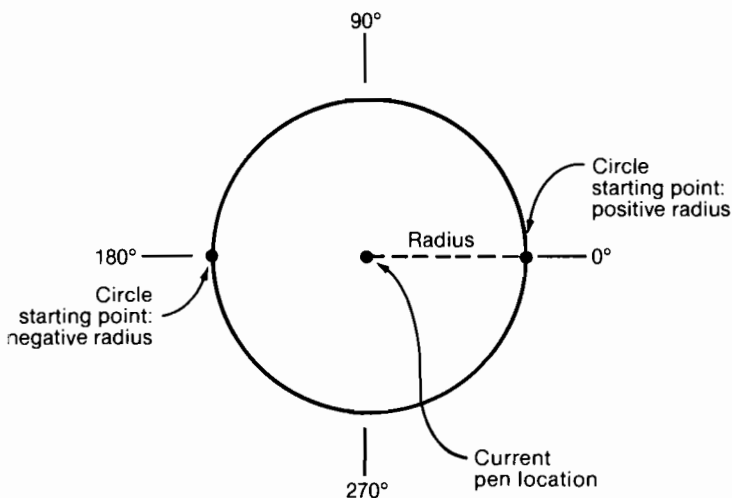
*When scaling is on, the range may be smaller. Refer to *User Units* in Chapter 2.

†This is the practical range. The allowable range is -32 768.0000 to 32 767.9999. However, 0 is changed to 0.36; numbers from 180 to 360 are interpreted as $360 - n$; numbers greater than 360 are interpreted as n modulo 360 (the result following the above rules for numbers greater and less than 180).

REMARKS: The CI instruction includes an automatic pen down. When the plotter receives a CI instruction, it moves the pen (in the up position) from the current location to the starting point on

the circumference, lowers the pen, and draws the circle. After drawing the circle, the plotter lifts the pen and returns it to the center of the circle and restores its up/down status. The plotter interprets the parameters as follows.

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



- **chord tolerance** — in degrees mode, specifies the maximum chord angle of the chords drawn to create the circle; in deviation distance mode, specifies the distance between the chord drawn and the arc it represents. For more information, refer to *Maintaining Circle/Arc Smoothness* at the beginning of this chapter.

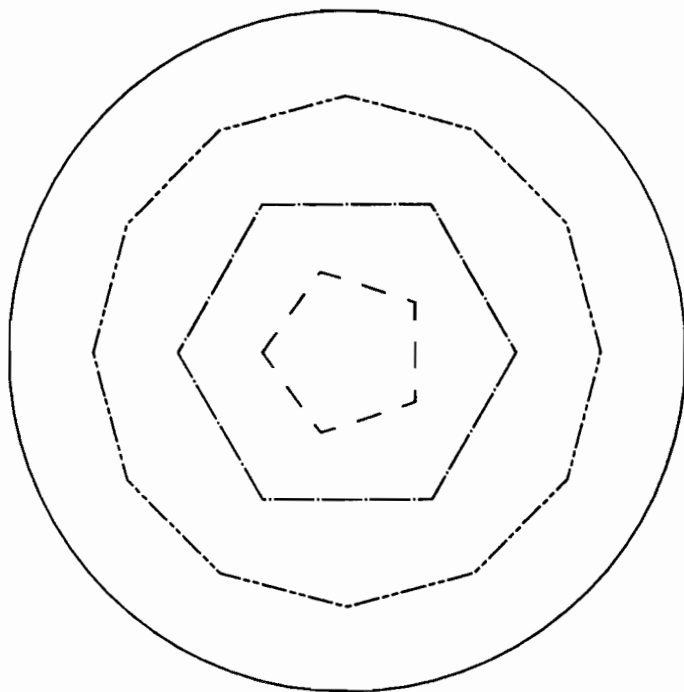
Because each chord of the circle is drawn using the current line type, you should not use adaptive line patterns to draw circles. Otherwise, the plotter draws the adaptive pattern for each chord in the circle (72 in the default chord angle mode). Not only may the pattern become indistinguishable, but the plotter requires more time than usual to draw the circle.

EXAMPLE: The following draws a series of concentric circles using combinations of chord angles and line types.

```

10  *Insert configuration statement here
20  PRINT #1, "IN;IP0,0,6000,6000;SC-100,100,-100,
    100;"
30  PRINT #1, "SP1;PA0,0;LT;CI80;LT6;CI-60,30;LT4;"
40  PRINT #1, "CI40,60;LT2;CI-20,72;SP0;"
50  END

```

**RELATED**

INSTRUCTIONS: AA, Arc Absolute
 AR, Arc Relative
 CT, Chord Tolerance

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 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
n	integer	0 to 1	0

REMARKS: A plotted circle or arc actually consists of a series of straight line segments, or chords. Increasing the number of chords increases the smoothness of the circle. Chord tolerance is interpreted as either a chord angle or a deviation distance. The plotter interprets the parameter as follows.

- **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.
- **no parameter** — is equivalent to *CT0*; and sets chord tolerance to degrees (chord angle) mode.

Refer to *Maintaining Circle/Arc Smoothness* at the beginning of this chapter for examples of chord tolerance modes.

RELATED

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

ERRORS:

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

EA, Edge Absolute Rectangle

USE: Defines and outlines a rectangle using absolute coordinates. Use the EA instruction to create rectangles for chart legends, floor plans, flow diagrams, schematics, or to frame any drawing.

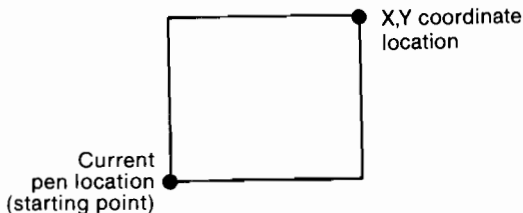
SYNTAX: EA X,Y;

Parameter	Format	Range	Default
X,Y	real	-32 768.0000 to 32 767.9999 current units*	none

*When scaling is on, the range may be smaller. Refer to *User Units* in Chapter 2.

REMARKS: The EA instruction defines a rectangle within the polygon buffer using absolute coordinates (you do not need to enter polygon mode), then edges it with a solid line using the current pen. This instruction includes an automatic pen down. The current pen location is the starting and ending point of the rectangle; the X,Y coordinates specify the opposite corner of the rectangle. When the rectangle is complete, the plotter restores the pen up/down status.

NOTE: The EA instruction clears any previous polygon from the polygon buffer before defining the rectangle. ■



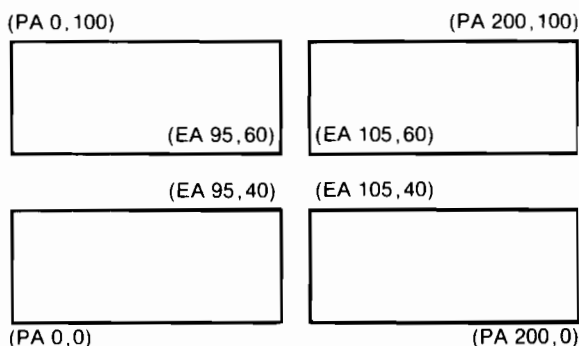
NOTE: The illustration shows the current pen location in the lower-left corner and the X,Y parameters in the upper-right corner. However, your current pen location can represent any corner of the rectangle, as long as the X,Y parameters represent the corner diagonally opposite the current pen location. ■

EXAMPLE: In each of the following rectangles, the PA instruction determines the rectangle starting point. Note the different placements of the rectangles' opposite corners with relation to each of the starting points. To draw the same rectangles using relative coordinates, refer to the ER, Edge Relative Rectangle, instruction. See also the RA (Rectangle Absolute Fill) and RR (Rectangle Relative Fill) instructions for examples of filling rectangular areas.

```

10  *Insert configuration statement here
20  FRINT #1, " IN;IP-3000,-1500,0,0;SC0,200,0,100;"
30  FRINT #1, "SP1;"
40  PRINT #1, "PA0,100;EA95,60;"
50  PRINT #1, "PA200,100;EA105,60;"
60  PRINT #1, "PA0,0;EA95,40;"
70  PRINT #1, "PA200,0;EA105,40;"
80  PRINT #1, "SP0;"
90  END

```



RELATED

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

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 Relative Rectangle

USE: Defines and outlines a rectangle using relative coordinates. Use ER to create rectangles for chart legends, floor plans, flow diagrams, schematics, or to frame any drawing.

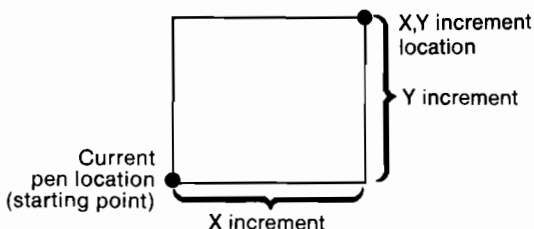
SYNTAX: ER X,Y;

Parameter	Format	Range	Default
X,Y	real	-32 768.0000 to 32 767.9999 current units*	none

*When scaling is on, the range may be smaller. Refer to *User Units* in Chapter 2.

REMARKS: The ER instruction defines a rectangle within the polygon buffer using relative increments from the current pen location (you do not need to enter polygon mode) and edges it with a solid line using the current pen. This instruction includes an automatic pen down. The current pen location is the starting and ending point of the rectangle; the X,Y increments specify the opposite corner of the rectangle. When the rectangle is complete, the plotter restores the pen up/down status.

NOTE: The ER instruction clears any previous polygon from the polygon buffer before defining the rectangle. ■



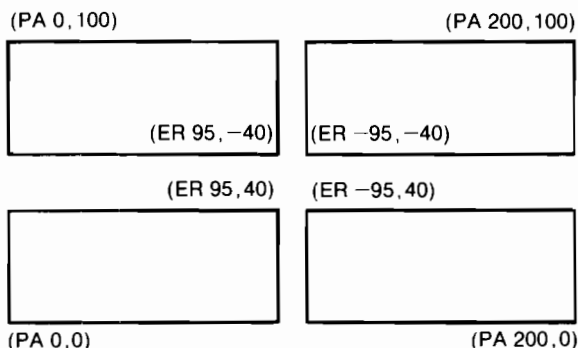
NOTE: The illustration shows the current pen location in the lower-left corner and the X,Y parameters in the upper-right corner. However, your current pen location can represent any corner of the rectangle, as long as the X,Y parameters represent the corner diagonally opposite the current pen location. ■

EXAMPLE: In each of the following rectangles, the PA instruction determines the rectangle starting point. Note the different placements of the rectangles' opposite corners with relation to each of the starting points. To draw the same rectangles using absolute coordinates, refer to the EA, Edge Absolute Rectangle, instruction. See also the RA (Rectangle Absolute Fill) and RR (Rectangle Relative Fill) instructions for examples of filling rectangular areas.

```

10 *Insert configuration statement here
20 PRINT #1, "IN;IP-3000,-1500,0,0;SC0,200,0,100;"
30 PRINT #1, "SP1;"
40 PRINT #1, "PA0,100;ER95,-40;"
50 PRINT #1, "PA200,100;ER-95,-40;"
60 PRINT #1, "PA0,0;ER95,40;"
70 PRINT #1, "PA200,0;ER-95,40;"
80 PRINT #1, "SP0;"
90 END

```



RELATED

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

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

EW, Edge Wedge

USE: Outlines any wedge. Use these instructions to produce sectors of a pie chart. (You can fill a wedge using the WG, Wedge Fill, instruction.)

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

Parameter	Format	Range	Default
radius	real	-32 768.0000 to 32 767.9999 current units*	none
start angle	real	0 to ± 360 degrees†	none
sweep angle	real	0 to ± 360 degrees†	none
chord tolerance chord angle	real	0.1 to 180 degrees††	5 degrees
chord deviation	real	-32 768.0000 to 32 767.9999 current units	equivalent to 5 degrees chord angle

*When scaling is on, the range may be smaller. Refer to *User Units* in Chapter 2.

†This is the practical range. The actual range is -32 768.0000 to 32 767.9999.

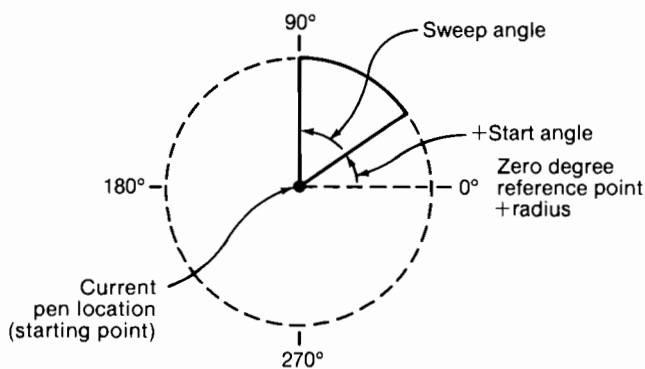
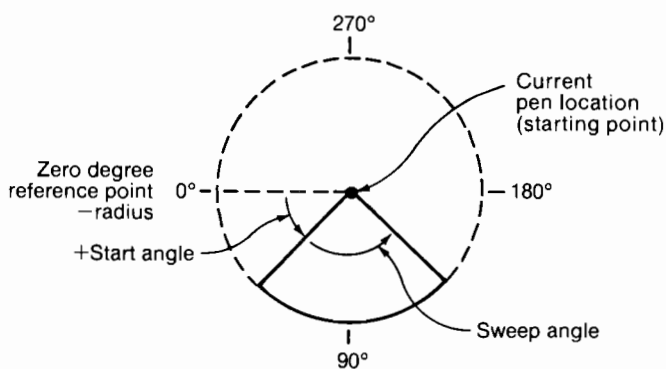
††This is the practical range. The allowable range is -32 768.0000 to 32 767.9999. However, 0 is changed to 0.36; numbers from 180 to 360 are interpreted as $360 - n$; numbers greater than 360 are interpreted as n modulo 360 (the result following the above rules for numbers greater and less than 180).

REMARKS: The EW instruction defines a wedge within the polygon buffer (you do not need to enter polygon mode) and edges it with a solid line (EW includes an automatic pen down). The current pen location is the starting and ending point of the wedge. When the wedge is complete, the pen up/down position is restored. The plotter interprets the parameters as follows.

- **radius** — measured in current units, specifies the distance from the current pen location to the start of the wedge's arc. The current pen location is the focal point of the wedge. The sign of the radius (+ or -) determines the orientation of the zero-degree reference point for the start angle and sweep angle. In the next illustration, note how a negative radius rotates the placement of the zero-degree reference point 180 degrees.
- **start angle** — specifies at what angle from the zero-degree reference point the plotter draws the first radius. The sign of the start angle determines whether the radius is drawn clockwise from the zero-degree reference point or counterclockwise. 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.

NOTE: Start angle values that are greater than 360 degrees are interpreted as modulo 360. ■

- **sweep angle** — specifies the number of degrees through which the arc of the wedge 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.

*Positive Radius**Negative Radius*

- **chord tolerance** — in degrees mode, specifies the maximum chord angle of the chords drawn to create the arc portion of the wedge; in deviation distance mode, specifies the distance between the chord drawn and the arc it represents. When using a large chord tolerance, note that the drawn chord is limited to the size of the sweep angle.

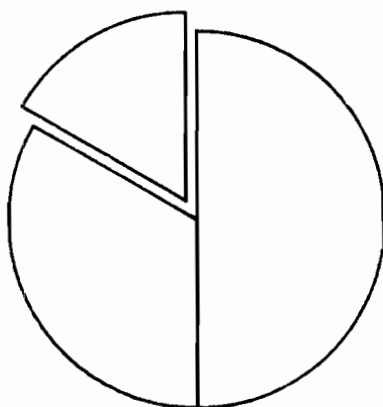
If the chord angle is greater than 180, then 360 minus the specified chord angle is used. If you specify a chord angle of zero, a chord angle of 0.36 is used.

EXAMPLE:

```

10 'Insert configuration statement here
20 PRINT #1, "IN;SP1;PA1000,500;"
30 PRINT #1, "EW-1000,90,180;"
40 PRINT #1, "EW-1000,330,120;"
50 PRINT #1, "PR-60,100;"
60 PRINT #1, "EW1000,90,60;"
70 PRINT #1, "SP0;"
80 END

```

**RELATED****INSTRUCTIONS:** WG, Wedge Fill**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

FT, Fill Type

USE: Selects the shading pattern for filling polygons, rectangles, or wedges. Use this instruction to enhance charts with solid fill, parallel lines (hatching), or cross-hatching patterns.

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

Parameter	Format	Range	Default
type	integer	1 to 4*	1
spacing	real	0 to 32767.9999 current units	1% of P1/P2 distance (fill types 3 & 4 only)
angle	real	0 to 180 degrees†	0 degrees

*This is the practical range. The actual range is 1 to 6 for compatibility with other HP plotters. This plotter uses the default fill type for values of 5 and 6.




†This is the practical range. The actual range is -32768.0000 to 32767.9999.

REMARKS: The plotter stores all three parameters of the FT instruction so that on subsequent FT instructions you can omit either the third or second and third parameters if you want them to remain the same. The plotter will use the corresponding values from the previous FT instruction in place of the missing parameters.

For example, after executing *FT3,100,45;*, you enter *FT4;*. The plotter will use the spacing and angle parameters from the first instruction and implement *FT4,100,45;*.

If you omit all parameters (*FT;*) the plotter uses the default values. The plotter interprets the parameters as follows.

- **type** — selects one of the following shading patterns.

Parameter Value	Fill Type	Pattern
1 or 2	solid	
3	hatch	
4	cross-hatch	

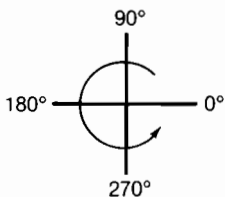
Although fill types 1 and 2 are solid, fill type 1 uses bidirectional filling, while fill type 2 uses unidirectional filling. For the highest quality solid fill on transparency film, use type 2. The spacing parameter is ignored; the line spacing for both of these fill types is determined using the PT (Pen Thickness) instruction.

Fill types 3 and 4 use the current line type as defined by the LT instruction. The plotter draws solid line types bidirectionally, nonsolid line types are drawn unidirectionally.

- **spacing** — is the distance in current units between parallel lines in the fill area for fill types 3 and 4. When scaling is on, the spacing parameter is interpreted as user units along the X-axis.

Until you specify a spacing parameter, the plotter uses the default spacing of 1% of the diagonal distance between P1 and P2. When the plotter is using the default spacing, subsequent changes in P1 and P2 can affect the spacing. After you specify a spacing parameter, the plotter uses the current unit spacing for subsequent fill types until you initialize the plotter, specify a new spacing parameter, or send FT without parameters.

- **angle** — is referenced counterclockwise from the positive direction of the X-axis as shown in the following illustration; 0 and 180 are horizontal, 90 and 270 are vertical. The angle applies to all fill types. For cross-hatching, the plotter draws the first set of lines at the specified angle; the second set are then drawn at that angle plus 90 degrees.



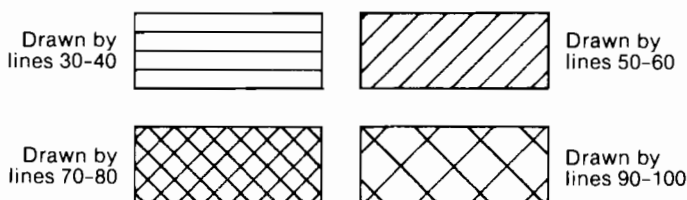
EXAMPLE: The following shows how you can create innumerable fill patterns by changing a single FT parameter at a time. You can also make dramatic changes by altering the line type as well. For more examples of area fill, refer to the following

instructions elsewhere in this chapter: WG, RA, and RR. Also refer to the PM (Polygon Mode) instruction and the FP (Fill Polygon) instruction in Chapter 6.

```

10  *Insert configuration statement here
20  PRINT #1, "IN;SP1;"
30  PRINT #1, "PA0,1000;FT3,100;"
40  PRINT #1, "RA1000,600;EA1000,600;"
50  PRINT #1, "PA2200,1000;FT3,100,45;"
60  PRINT #1, "RA1200,600;EA1200,600;"
70  PRINT #1, "PA0,0;FT4;"
80  PRINT #1, "RA1000,400;EA1000,400;"
90  PRINT #1, "PA2200,0;FT4,200;"
100 PRINT #1, "RA1200,400;EA1200,400;"
110 PRINT #1, "SP0;"
120 END

```



RELATED

INSTRUCTIONS: LT, Line Type
 PM, Polygon Mode
 PT, Pen Thickness
 RA, Rectangle Absolute Fill
 RR, Rectangle Relative Fill
 WG, Wedge Fill

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

PT, Pen Thickness

USE: Varies the spacing between parallel lines in solid-fill patterns according to the thickness of the pen-tip.

SYNTAX: *PT pen thickness*; or *PT*;

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

REMARKS: The pen thickness parameter represents the physical pen-tip width in millimetres. Pen-tip width is an average width. The *PT* instruction allows you to make the necessary adjustment to compensate for slight differences in pen tips.

When you specify a pen thickness, the plotter adjusts the spacing between parallel lines in solid-fill patterns to half of the pen thickness. If your solid fill has unwanted gaps showing between the lines, specify a smaller pen thickness parameter than your pen indicates. You can plot with a less dense solid fill by specifying a larger pen thickness than your pen indicates. The larger the pen thickness, the less strokes (and time) required to fill an area.

The *PT* instruction pertains only to the currently selected pen. It remains in effect until:

- a new pen is selected using an *SP* instruction or manually from the front panel
- a new *PT* instruction is executed
- the plotter is initialized or set to default conditions

EXAMPLE: Both of the following rectangles use the same pen. The rectangle on the left specifies a pen thickness of 0.3, the actual width of the pen tip; the rectangle on the right specifies a pen thickness of 0.7, more than twice the width of the pen tip.

```

10 *Insert configuration statement here
20 PRINT #1, "IN;SP1;PA5000,5000;"
30 PRINT #1, "PT.3;RR500,1000;"
40 PRINT #1, "PU;PR1000,0;"
50 PRINT #1, "PT.7;RR500,1000;"
60 PRINT #1, "SP0;"
70 END
    
```



Drawn by line 30



Drawn by line 50

RELATED

INSTRUCTIONS: FT, Fill Type

ERRORS:

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

RA, Rectangle Absolute Fill

USE: Fills a rectangular area defined by absolute coordinates. Use this instruction to fill rectangles required by chart legends, logos, and other plots.

SYNTAX: RA X,Y;

Parameter	Format	Range	Default
X,Y	real	-32768.0000 to 32767.9999 current units*	none

*When scaling is on, the range may be smaller. Refer to *User Units* in Chapter 2.

REMARKS: The RA instruction defines a rectangle within the polygon buffer using absolute coordinates (you do not need to enter polygon mode), then fills it using the current fill pattern (including line type). This instruction includes an automatic pen down. The current pen location represents one corner of the rectangle; you need specify only the absolute X,Y coordinates for the opposite corner of the rectangle. After filling the rectangle, the plotter returns the pen to the starting point and restores the pen up/down status.

EXAMPLE: The following illustrates the RA instruction used with different fill types (refer to the FT, Fill Type, instruction) to create rectangular areas as might be used in chart legends.

```

10  *Insert configuration statement here
20  PRINT #1, "IN;SP1;PA1000,1000;"
30  PRINT #1, "FT1;"
40  PRINT #1, "RA2000,1400;"
50  PRINT #1, "PR1100,0;FT4,100,45;"
60  PRINT #1, "RA3100,1400;"
70  PRINT #1, "PR1100,0;FT3,100,0;"
80  PRINT #1, "RA4200,1400;"
90  PRINT #1, "SP0;"
100 END

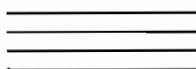
```



Drawn by
lines 30-40



Drawn by
lines 50-60



Drawn by
lines 70-80

Refer to the EA (Edge Absolute Rectangle) or ER (Edge Relative Rectangle) instructions to outline the rectangular areas, and to the RR (Relative Rectangle Fill) instruction to create these same rectangles using relative coordinates.

RELATED

INSTRUCTIONS: EA, Edge Absolute Rectangle
ER, Edge Relative Rectangle
FT, Fill Type
LT, Line Type
RR, Rectangle Relative Fill

ERRORS:

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

RR, Relative Rectangle Fill

USE: Fills a rectangular area defined using relative coordinates. Use this instruction to fill rectangles required by chart legends, logos, and other plots.

SYNTAX: RR X,Y;

Parameter	Format	Range	Default
X,Y	real	-32768.0000 to 32767.9999 current units*	none

*When scaling is on, the range may be smaller. Refer to *User Units* in Chapter 2.

REMARKS: The RR instruction defines a rectangle within the polygon buffer using relative increments (you do not need to enter polygon mode), then fills it using the current fill pattern (including line type, if applicable). This instruction includes an automatic pen down. The current pen location represents one corner of the rectangle; you need specify only the relative X,Y increment for the opposite corner of the rectangle. After filling the rectangle, the plotter returns the pen to the starting point and restores the pen up/down status.

EXAMPLE: The following illustrates the RR instruction used with different fill types (refer to the FT, Fill Type, instruction) to create rectangles such as those you might use for chart legends. Compare this program with the example for the RA instruction to understand the differences between the parameter values.


```

10  *Insert configuration statement here
20  PRINT #1, "IN;SP1;PA1000,1000;"
30  PRINT #1, "FT1;"
40  PRINT #1, "RR1000,400;"
50  PRINT #1, "PR1100,0;FT4,100,45;"
60  PRINT #1, "RR1000,400;"
70  PRINT #1, "PR1100,0;FT3,100,0;"
80  PRINT #1, "RR1000,400;"
90  PRINT #1, "SP0;"
100 END

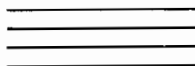
```



Drawn by
lines 30-40



Drawn by
lines 50-60



Drawn by
lines 70-80

RELATED

INSTRUCTIONS: EA, Edge Absolute Rectangle
ER, Edge Relative Rectangle
FT, Fill Type
LT, Line Type
RA, Rectangle Absolute Fill

ERRORS:

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

WG, Wedge Fill

USE: Defines and fills any sector of a circle. Use this instruction with different fill types, line types, and pen thicknesses to produce individualized wedges.

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

Parameter	Format	Range	Default
radius	real	-32 768.0000 to 32 767.9999 current units*	none
start angle	real	0 to ± 360 degrees [†]	none
sweep angle	real	0 to ± 360 degrees [†]	none
chord tolerance chord angle	real	0.1 to 45 degrees ^{††}	5 degrees
chord deviation	real	-32 768.0000 to 32 767.9999 current units	equivalent to 5 degrees chord angle

*When scaling is on, the range may be smaller. Refer to *User Units* in Chapter 2.

[†]This is the practical range. The actual range is -32 768.0000 to 32 767.9999.

^{††}This is the practical range. The allowable range is 32 768.0000 to 32 767.9999. However, 0 is changed to 0.36; numbers from 180 to 360 are interpreted as $360 - n$; numbers greater than 360 are interpreted as n modulo 360 (the result following the above rules for numbers greater and less than 180).

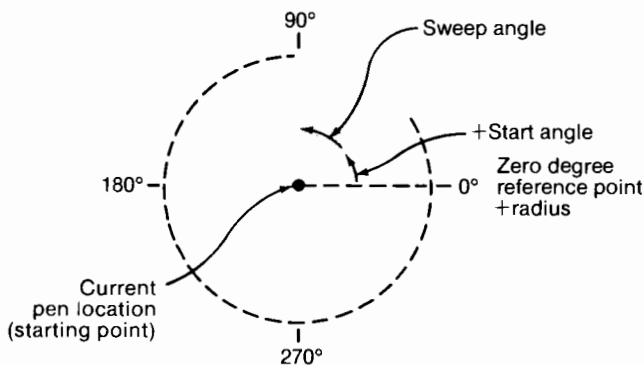
REMARKS: The WG instruction defines a wedge within the polygon buffer (you do not need to enter polygon mode), then fills it using the current fill pattern. The current pen location is the focal point of the wedge. When the wedge is complete, the plotter returns the pen to this location and restores its up/down status. The plotter interprets the parameters as follows.

- **radius** — specifies the distance from the current pen location to the wedge's arc. The sign of the radius determines the orientation of the zero-degree reference point for the start angle and sweep angle. In the next illustration, note how a negative radius rotates the placement of the zero-degree reference point 180 degrees.
- **start angle** — specifies at what angle from the zero-degree reference point the plotter draws the first radius. The sign of the start angle determines whether the radius is drawn clockwise from the zero-degree reference point or counterclockwise.

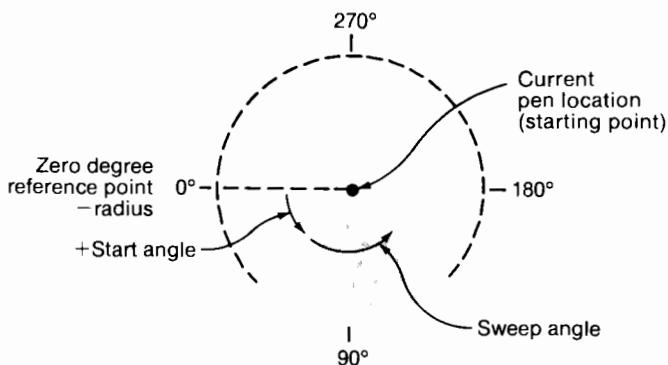
A positive start angle positions the radius counterclockwise from the zero-degree reference point; a negative start angle positions the radius clockwise from the zero-degree reference point.

NOTE: The plotter interprets all values greater than 360 as modulo 360. ■

- **sweep angle** — specifies the number of degrees through which the arc of the wedge 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.



Positive Radius



Negative Radius

- **chord tolerance** — in degrees mode, specifies the maximum chord angle of the chords drawn to create the arc portion of the wedge; in deviation distance mode, specifies the distance between the chord drawn and the arc it represents. When using a large chord tolerance, note that the drawn chord is limited to the size of the sweep angle.

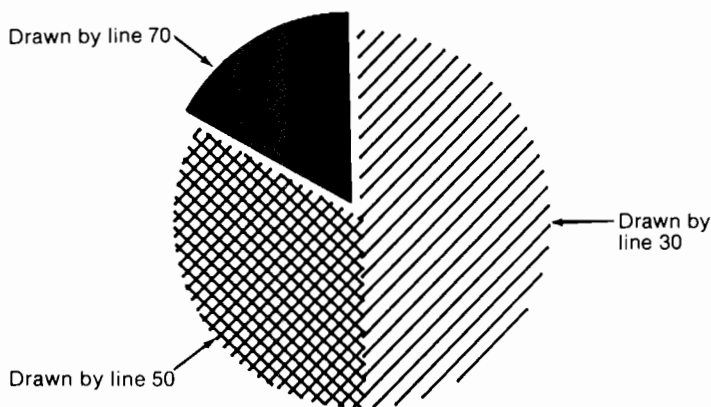
If the chord angle is greater than 180, then 360 minus the specified chord angle is used. If you specify a chord angle of zero, a chord angle of 0.36 is used.

EXAMPLE: The following illustrates how you can combine the WG instruction with different fill types to create a pie chart with an offset wedge.

```

10  *Insert configuration statement here
20  PRINT #1, "IN;SP1;PA1000,500;FT3,75,45;"
30  PRINT #1, "WG-1000,90,180;"
40  PRINT #1, "FT4,60;"
50  PRINT #1, "WG-1000,330,120;"
60  PRINT #1, "PR-60,100;FT1,0,0;"
70  PRINT #1, "WG1000,90,60;"
80  PRINT #1, "SP0;"
90  END

```



To give the wedges more definition, you can outline them using either the EW (Edge Wedge) instruction or the EP (Edge Polygon) instruction. You can use the EP instruction because each wedge

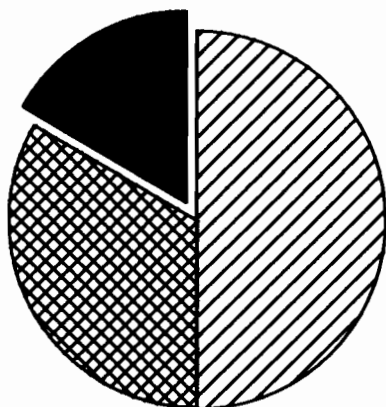
is defined within the polygon buffer before it is filled. Insert either set of the following program lines into the previous program to outline the wedges that program draws.

Using the EW instruction

```
35 PRINT #1, "EW-1000,90,180;"
55 PRINT #1, "EW-1000,330,120;"
75 PRINT #1, "EW1000,90,60;"
```

Using the EP instruction

```
35 PRINT #1, "EP;"
55 PRINT #1, "EP;"
75 PRINT #1, "EP;"
```



Note that an EP instruction edges the polygon currently in the polygon buffer and therefore must always follow the instructions that define the polygon's shape. Refer to the EP, Edge Polygon, instruction in Chapter 6. Because the EW instruction includes the same parameters as the WG instruction, you can place the EW instruction anywhere in the program listing.

RELATED

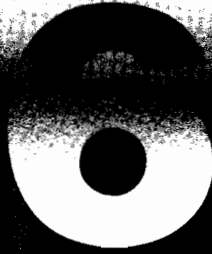
INSTRUCTIONS: EW, Edge Wedge
FT, Fill Type
LT, Line Type

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	fills buffer contents

Notes

CHAPTER





Drawing Polygons and Using the Polygon Buffer

This chapter discusses 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.

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 O is a polygon composed of two subpolygons; the outer circle, and the inner circle. You can use polygon mode to create numerous shapes or logos.

When defining polygons, you can use the following HP-GL 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 issue the IN (Initialize) instruction while in polygon mode, the plotter exits polygon mode and immediately begins executing subsequent instructions. You must exit polygon mode to execute other HP-GL instructions.

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 PM, Polygon Mode, instruction to enter polygon definition mode.
2. Define the shape of your polygon using any combination of the instructions listed previously. (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 FP, Fill Polygon, instruction, and/or outline it using the EP, Edge Polygon, instruction.

Refer to the PM (Polygon Mode) instruction later in this chapter for a full description of the parameters that open and close polygons and subpolygons.

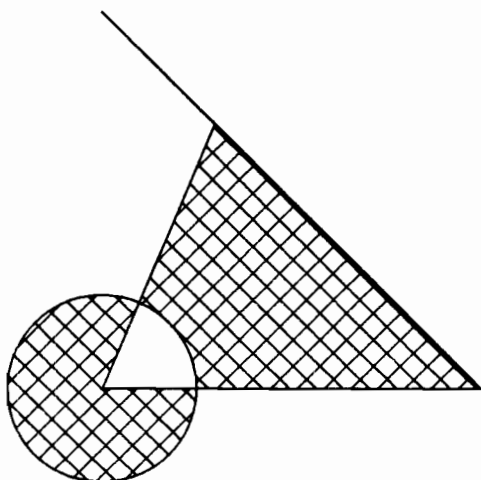
You can define points with the pen up or down. However, the EP instruction only draws between points that are defined when the pen is down. The FP instruction fills all vertices, regardless of whether the pen is up or down when defined.

*The output instructions are OA, OC, OD, OE, OF, OH, OI, OO, OP, OS, and OW. (Refer to Chapters 10, 11, and 12.)

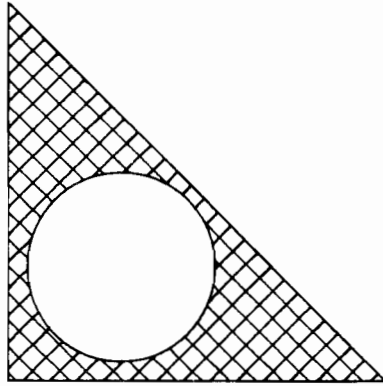
Drawing Circles in Polygon Mode

Polygon mode interprets the CI (Circle) instruction differently from the other HP-GL 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 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 EP (Edge Polygon) instruction example shows the correct method of drawing 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 drawn incompletely. ■

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 wedge and rectangle instructions (discussed later in this chapter). 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 drawn incompletely. The default size of the polygon buffer is 1024 bytes. This size lets the polygon buffer store any polygon having up to 93 points; this is sufficient for most polygons.

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 14). Enlarge the polygon buffer using either the GM (Graphics Memory) instruction described later in this chapter or the ESC . T instruction described in Chapter 14.

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.

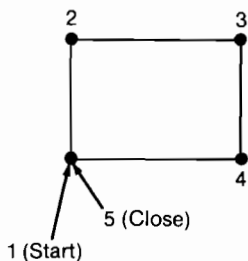
Determining the Approximate Buffer Size

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.

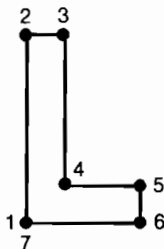
$$\# \text{ of bytes} = \# \text{ of points} \times 11$$

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

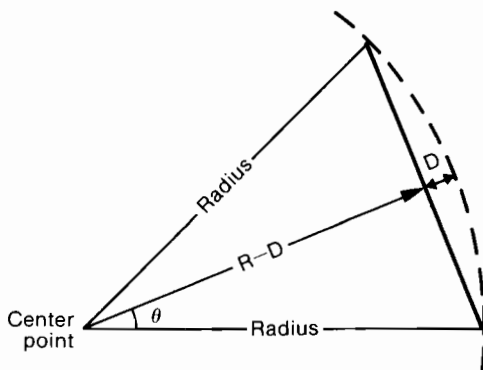
$$\# \text{ of points} = \text{arc size (degrees)} / \text{chord angle (degrees)} + 1$$

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 mode 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 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 where 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.

$$\text{number of bytes} = [(p1 + f1) + (p2 + f2) + \dots]$$

Each segment of the equation is described as follows:

p represents 2 bytes needed by any of the following.

- a PU or PD that changes the pen up/down position *and* has X,Y coordinates following, or
- a PM0 or PM1 instruction. (Note that a PM1 instruction includes an automatic pen up. A *PM1;PU...* instruction, then, uses only 2 bytes since the PU instruction does not change the pen status.)

f is defined by the following formula.

$$f = (9 \times \text{total number of points}) + (\text{total number of points}/256) \times 2$$

(Use the integer portion of the division by truncating any decimal fraction, not by rounding. That is, for 200 points, $f = (9 \times 200) + ((200/256) \times 2) = 1800 + 0 = 1800$; because $200/256 = 0$, not 0.7813.)

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.

PM0 uses 2 bytes	2
Current pen location is first point; solve f using 1 point	9
PD changes pen status, using 2 bytes	2
2 points follow PD; solve f using 2 points	18
PD does not change pen status and 4 points follow; solve f using 4 points	36
PD does not change pen status and 3 points follow; solve f using 3 points	27
PM1 uses 2 bytes*	2
PU does not change pen status and 1 point follows; solve f using 1 point	9
PD changes pen status, using 2 bytes	2
4 points follow PD; solve f using 4 points	36
<hr/>	
Total number of bytes required for this polygon is	143

Contrast this number with the previous formula, that for an approximate buffer size. The polygon has 15 points, so the approximate size of the polygon buffer is 165 bytes (15×11). You save 22 bytes for the I/O or pen sort buffers using the more precise equation.

*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 9 (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 GM, Graphics Memory, instruction and the ESC.T instruction, which changes configurable memory allocations. The GM instruction described later in this chapter, allocates memory to two of the three buffers; it does not change the size of the I/O buffer. The ESC.T instruction is a device-control instruction that allocates memory to all three buffers (ESC.T is described in Chapter 14).

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 HP-GL instructions. It does not purge the contents of the I/O or pen sort buffers 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* in Chapter 1 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).

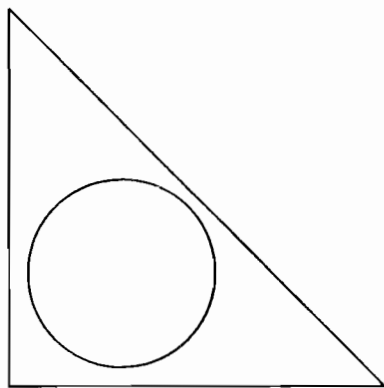
SYNTAX: EP;

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;PA0,0;"
30 PRINT #1, "PM0;PD2000,0,0,2000,0,0;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

ERRORS:

Condition	Error	Plotter Response
use of a parameter	2	executes instruction
no polygon defined	none	ignores 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.

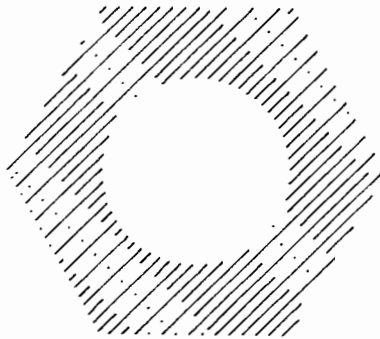
The polygon is filled using the current pen, fill type, and line type (if the fill type is not solid). On completion of the FP instruction, the plotter restores the original location and pen up/down status.

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 EP, Edge Polygon, 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;FT3,50,45;"
50  PRINT #1, "FP;"
60  PRINT #1, "SP0;"
70  END

```



RELATED

INSTRUCTIONS: EP, Edge Polygon
 FT, Fill Type
 LT, Line Type
 PM, Polygon Mode
 RA, Rectangle Absolute Fill
 RR, Relative Rectangle Fill
 WG, Wedge Fill

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 two of the three buffers in the configurable graphics memory.

SYNTAX: GM (polygon buffer)(downloadable character buffer*)
(,replot buffer*)(,vector buffer*)(,pen sort buffer);
or GM;

Parameter	Format	Range	Default
polygon buffer	integer	0 to 7434 bytes	1024 bytes
downloadable character buffer*	integer	0	0
replot buffer*	integer	0	0
vector buffer*	integer	0	0
pen sort buffer	integer	12 to 7446 bytes	5400 bytes

REMARKS: The plotter has a total of 7448 bytes of available memory in three buffers: the polygon buffer, the pen sort buffer and the physical I/O buffer. The GM instruction does not change the size of the physical I/O buffer. Use the ESC.T instruction (Chapter 13) if you need to change the physical I/O buffer, along with the polygon and pen sort buffers.

When the I/O buffer size is set to its minimum size (2 bytes), you can allocate 7446 bytes between the remaining two buffers, with a minimum of 12 going to the pen sort buffer. If you increase any buffer above its default size, you must decrease the size of another buffer by an equivalent amount.

The GM instruction is executed like other HP-GL 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

*This plotter does not have these buffers and ignores any values in these positions. These parameter positions provide compatibility with other HP plotters.

the polygon, you can immediately reset the polygon and pen sort buffers to more reasonable sizes with no loss of data.

The GM instruction clears the polygon buffer before allocating memory; it does not affect the I/O buffer. If the total memory allocation (including the current I/O buffer) exceeds the available memory (that is, greater than 7448 bytes), the plotter ignores the instruction. For example, the instruction *GM4000,0,0,0,2500*; is valid only if the I/O buffer has been previously reduced from its default size of 1024 to 948; otherwise, the allocation exceeds 7448 bytes and the instruction is ignored. On the other hand, the instruction *GM4000,0,0,0,2400*; does not require that you change the I/O buffer from its default size.

NOTE: If you specify less 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 pen sort buffer to 5400.

The plotter allows you to specify more than five parameters to allow for compatibility with future HP plotters. ■

RELATED

INSTRUCTIONS: **ESC . T** (device-control instruction for allocating memory)

ERRORS:

Condition	Error	Plotter Response
parameter less than minimum range	none	sets buffer to minimum
sum of parameters (and I/O buffer) exceeds 7448 bytes	3	ignores instruction

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 FP (Fill Polygon) instruction and/or outline them using the EP (Edge Polygon) instruction.

SYNTAX: *PM n*; or *PM*;

Parameter	Format	Range	Default
n	integer	0, 1, and 2	0

REMARKS: In polygon mode, you define the area of the polygon(s) using HP-GL 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

<i>PM</i>	Polygon mode
<i>PA/PR</i>	Plotting modes
<i>PU/PD</i>	Pen positions
<i>AA/AR</i>	Arc instructions
<i>CI</i>	Circle instruction
<i>CT</i>	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 HP-GL 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 HP-GL 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 following 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 93 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 HP-GL instructions *except* IN and all HP-GL 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, "PD2000,3000,2000,2000;"
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
parameter out of range	3	ignores instruction
buffer overflow	7	ignores overflowing points

Notes

CHAPTER

7



Enhancing Your Plots

You can increase the visual effectiveness of your plots in several ways. Previous chapters have demonstrated the use of different fill types as a means of giving your plot greater definition. This chapter illustrates the use of the following.

- dotted and/or dashed line patterns
- symbols
- tick marks on axes
- grids

By using some of these enhancement features, you can emphasize or minimize the relative importance of specific areas of a plot.

LT, Line Type

USE: Specifies the line pattern to use when drawing lines and filling polygons with a nonsolid pattern. 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 diagonal distance between P1 and P2

*This is the practical range, the allowable range is -32768.0000 to 32767.9999.

REMARKS: The plotter interprets the parameters as follows.

- **pattern number** — produces the corresponding line pattern. Line patterns can be either a fixed or adaptive type. That is, a fixed pattern remains the specified length and any unused part of the pattern is carried over to the next line. Adaptive line patterns adjust the length of the pattern so that one or more complete patterns fit within the specified line. With adaptive line patterns, there is no unused portion of a line pattern to carry over.

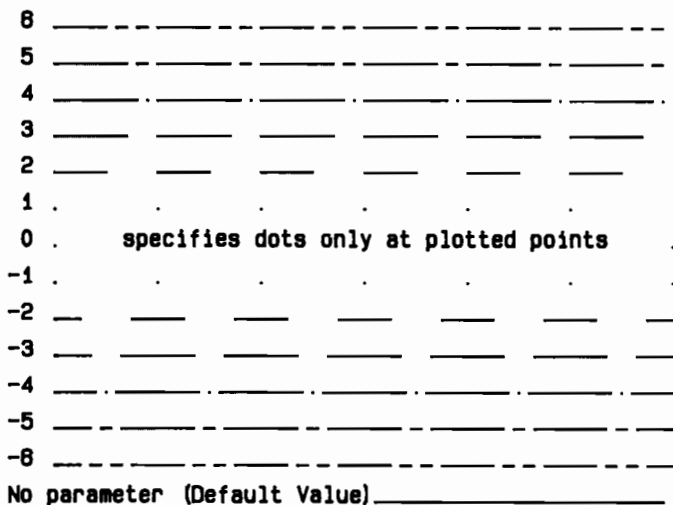
A positive pattern number (1 to 6) represents a fixed line type and uses the actual specified pattern length to draw lines. Any unused part of the pattern is carried over to the next line, unless you have issued a PU instruction. The PU instruction clears the carry-over portion of the pattern segment; the next line will be drawn from the beginning of the pattern.

A line pattern of zero (0) draws dots at the PA and/or PR coordinates only.

A negative pattern number (-1 to -6) represents an adaptive line type and automatically adjusts the pattern length so that each line contains one or more complete patterns.

NOTE: Do not use an adaptive line type for drawing circles, arcs, or polygons that use an abundance of these shapes. The plotter will attempt to draw the complete pattern in every chord; you could lose the effectiveness of the line type, wasting time drawing the circle. ■

The line types, fixed and adaptive, are as follows.



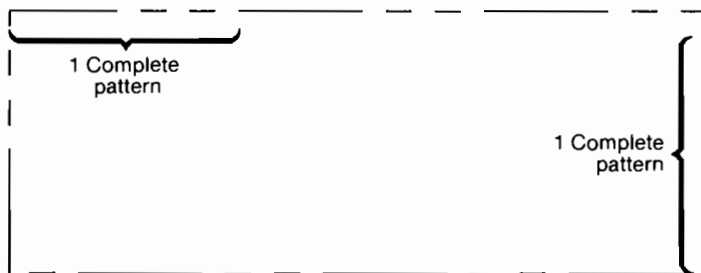
- **pattern length** — specifies the length of one complete pattern as a percentage of the diagonal distance between the scaling points P1 and P2. If you don't specify a length, the plotter uses the last value specified. If you haven't previously specified a pattern length, the plotter uses a pattern length of 4% of the diagonal distance between P1 and P2.
- **no parameters** — draws a solid line and resets the pattern length to its default. This is *not* the same as sending *LTO*;

EXAMPLE: The following illustrates the difference between the fixed and adaptive line types.

```

10 'Insert configuration statement here
20 PRINT #1, "IN;IP-2500,-2000,2500,2000;"
30 PRINT #1, "SC0,10,0,10;SP1;"
40 ' fixed line type
50 PRINT #1, "LT6,20;PA1,6;PD1,9,9,9,9,6,1,6;"
60 ' adaptive line type
70 PRINT #1, "LT-6,20;PA1,1;PD1,4,9,4,9,1,1,1;"
80 PRINT #1, "SP0;"
90 END

```



Fixed Line Type



Adaptive Line Type

RELATED

INSTRUCTIONS: EP, Edge Polygon
 FP, Fill Polygon
 FT, Fill Type

ERRORS:

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

SM, Symbol Mode

USE: Draws a specified character at each X,Y coordinate. By drawing a specified character at each data point, you can differentiate data contained in geometric drawings, scattergrams, or in multiple-line graphs.

Note that a second character allows access to Kanji characters in symbol mode. Refer to Appendix D for details concerning accessing Kanji characters.

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

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

*ASCII decimal code 59 is a symbol mode terminator. In many of the character sets, decimal code 59 is the semicolon, which usually terminates most HP-GL instructions. You can also terminate symbol mode by sending any of the ASCII control characters (decimal codes 0-31).

REMARKS: Symbol mode lets you specify a printing character to draw at subsequent X,Y locations (using PA, PR, PD, or PU). Symbol mode includes an automatic pen down feature that draws each symbol centered over each coordinate location. After drawing the symbol, the plotter restores the current up/down pen position. This allows you to draw the symbol at each point without a connecting line.

The character drawn is 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. Also, the size (SI and SR), slant (SL), and direction (DI and DR) instructions affect how the character is drawn.

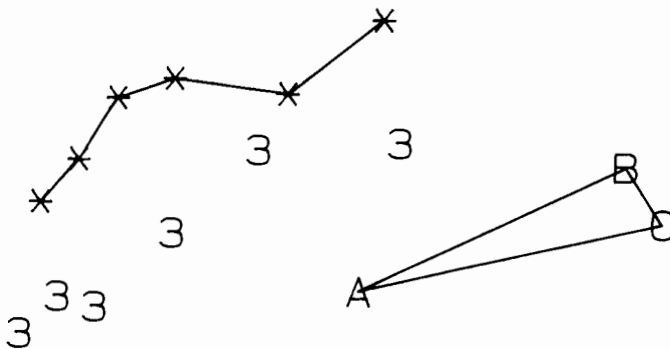
A specified symbol remains in effect until the symbol is replaced with another, symbol mode is terminated, or the plotter is initialized or set to default conditions. Note that you cannot use negative numbers as a symbol; the plotter will print only the minus sign and set an error.

EXAMPLE: The following plots symbols in three situations: with the pen down for a line graph, with the pen up for a scattergram, and with the pen down for a geometric drawing.

```

10  *Insert configuration statement here
20  PRINT #1, "IN;SP1;";
30  PRINT #1, "SM*;PA200,1000;";
40  PRINT #1, "PD400,1230,600,1560,900,1670,
      1500,1600,2000,2000;PU;";
50  PRINT #1, "SM3;PA100,300;";
60  PRINT #1, "PA300,500,500,450,900,850,1350,1300,
      2100,1350;";
70  PRINT #1, "SMA;PA1900,560;";
80  PRINT #1, "SMB;PD3300,1250;";
90  PRINT #1, "SMC;PA3500,950;";
100 PRINT #1, "SM;PA1900,560;PU;SP0;";
110 END

```



ERROR:

Condition	Error	Plotter Response
more than 1 character as a symbol	2	prints only the first character



TL, Tick Length

USE: Specifies the length of the tick marks on the X-and Y-axis. Use this instruction to adjust both the positive and negative portions of tick marks, and to draw grids.

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

Parameter	Format	Range	Default
positive tick (,negative tick)	real	-100 to 100 percentage*	0.5%

*This is the practical range. The total range is -32 768.0000 to 32 767.9999.

REMARKS: The plotter interprets the parameters as follows.

- **positive tick** — determines the length of the tick mark drawn to the positive side of the X- and/or Y-axis.
- **negative tick** — determines the length of the tick mark drawn to the negative side of the X- and/or Y-axis.
- **no parameters** — reestablishes default values.

Both parameters are specified as a percentage of the horizontal and vertical distances between P1 and P2. When used with the XT (X-Tick) instruction, the parameters are a percentage of the absolute value of the vertical scale ($|P2_y - P1_y|$). When used with the YT (Y-Tick) instruction, the parameters are a percentage of the absolute value of the horizontal scale ($|P2_x - P1_x|$). The default value is 0.5% of the horizontal and vertical distances for positive and negative ticks.

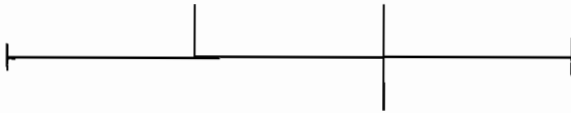
Use the instruction with both parameters to draw a tick that extends on both sides of the axis. To suppress the negative portion of the tick, use the instruction with only the positive tick parameter. To suppress the positive portion of the tick, specify zero (0) for the positive tick and the desired percentage for the negative tick.

Because tick length is a percentage of P1/P2 distance, only when the area defined by P1 and P2 defines a square will X-ticks and Y-ticks be the same length. A TL instruction remains in effect

until another TL instruction specifies new values, or the plotter is initialized or set to default conditions.

EXAMPLE: The following draws an X-tick with the default positive and negative length. Next, a positive tick with a length 2% of the vertical distance between P1 and P2, followed by a positive and negative tick, each with a length of 2% of the P1/P2 distance. Finally, note how the new locations of P1 and P2 change the length of the tick mark.

```
10 *Insert configuration statement here
20 PRINT #1, "IN;SP1;PA0,0;"
30 PRINT #1, "XT;PD;PR1000,0;"
40 PRINT #1, "TL2;XT;PR1000,0;"
50 PRINT #1, "TL2,2;XT;PR1000,0;"
60 PRINT #1, "IP0,0,5000,5000;"
70 PRINT #1, "XT;PU;SP0;"
80 END
```



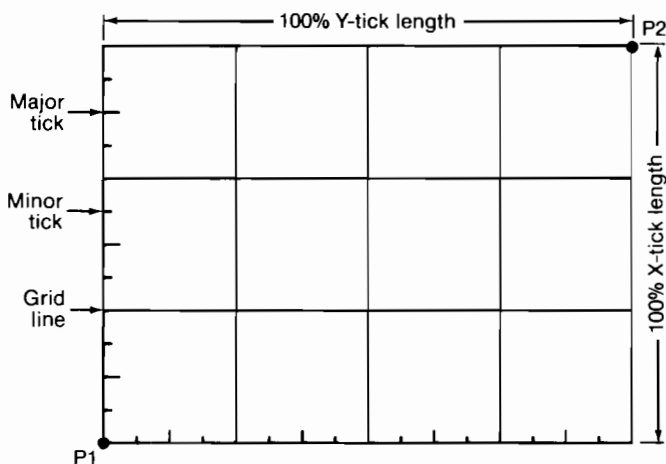
The following establishes locations for P1 and P2 and creates a grid using major and minor ticks.

```
10 *Insert configuration statement here
20 PRINT #1, "IN;SP1;IP200,1000,3000,3100;"
30 PRINT #1, "SC0,40,0,30;"
40 PRINT #1, "PA0,30;PD0,0,40,0,40,30,0,30;"
50 FOR I = 1 TO 3
60 PRINT #1, "PD;PR0,-2.5;TL1.5;YT;PR0,-2.5;"
70 PRINT #1, "TL3;YT;PR0,-2.5;TL1.5;YT;"
80 PRINT #1, "PR0,-2.5;TL100;YT;"
90 NEXT I
100 FOR J = 1 TO 4
110 PRINT #1, "TL100;XT;PR2.5,0;TL1.5;XT;"
120 PRINT #1, "PR2.5,0;TL3;XT;PR2.5,0;"
130 PRINT #1, "TL1.5,0;XT;PR2.5,0;"
140 NEXT J
```

```

150 PRINT #1, "SP0;"
160 END

```



RELATED

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

ERRORS:

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

XT, X-Tick

USES: Draws a vertical tick at the current pen location. Use this instruction to draw vertical tick marks on axes, and with the TL (Tick Length) instruction to draw grid lines, or lines centered on or starting with the current pen location.

SYNTAX: XT;

REMARKS: The XT instruction includes an automatic pen down. After drawing the tick, the plotter restores the pen up/down status. The length of the tick mark is determined by the TL (Tick

Length) instruction. The default length of the tick mark is 0.5% of $|P_{2Y} - P_{1Y}|$ for each positive and negative tick. Change the length of the positive and negative tick marks using the TL instruction.

EXAMPLE:

```
10  *Insert configuration statement here
20  PRINT #1, "IN;SP1;PA200,500;"
30  PRINT #1, "XT;PD;PR1000,0;XT;PR1000,0;XT;
    PR1000,0;XT;"
40  PRINT #1, "PU;SP0;"
50  END
```



RELATED

INSTRUCTIONS: TL, Tick Length
YT, Y-Tick

ERROR:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

YT, Y-Tick

USE: Draws a horizontal tick at the current pen location. Use this instruction to draw horizontal tick marks on axes, grid lines, or lines centered on or starting with the current pen location.

SYNTAX: YT;

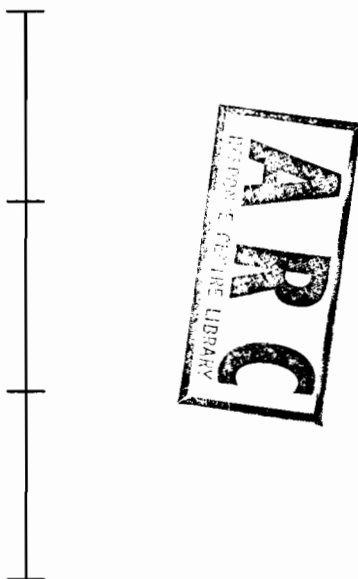
REMARKS: The YT instruction includes an automatic pen down. After drawing the tick, the pen up/down status is restored. The length of the tick mark is determined by the TL (Tick Length) instruction. The default length of the tick mark is 0.5% of $|P_{2X} - P_{1X}|$ for each positive and negative tick. Change the length of the positive and negative tick marks using the TL instruction.

EXAMPLE:

```

10 *Insert configuration statement here
20 PRINT #1, "IN;SP1;PA200,500;"
30 PRINT #1, "YT;PD;PR0,1000;YT;PR0,1000;YT;
    PR0,1000;YT;"
40 PRINT #1, "PU;SP0;"
50 END

```

**RELATED**

INSTRUCTIONS: TL, Tick Length
 XT, X-Tick

ERROR:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

CHAPTER

8



Labeling Your Plots

Labels are an important part of graphic communication. This chapter presents basic labeling techniques so you can label your plots for greater clarity. The plotter's default character set is used for demonstrating the following (Chapter 9 discusses the other character sets available).

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

Using Labels

When used with graphics, labels emphasize areas of a chart or plot that require special attention and/or explanation. Drawing labels is easy using HP-GL instructions. You can control virtually all facets of the label's appearance, for example, its position, size, slant, and direction. The instructions that affect all of these features are described in this chapter.

Default Label Conditions

When default conditions are established, the plotter automatically selects the ANSI ASCII English character set. For simplicity, this chapter uses the default character set to illustrate the basic labeling concepts. Chapter 9 discusses how to access different character sets. The following lists other default label conditions.

- label terminator: **ETX** (ASCII decimal code 3)
- starting position: current pen location
- character size: width = 0.285 cm; height = 0.375 cm
- direction: horizontal
- slant: none (vertical)

You can change any of these using the appropriate instruction described later in this chapter.

Terminating Labels

The LB (Label) instruction lets you use any printing characters to define your label. To indicate the end of your label, you must terminate your label string using a special label terminator instead of the usual HP-GL terminator. The label terminator tells the plotter to exit from the label mode. If you don't use the label terminator, everything following the LB instruction is printed in the label, including other HP-GL instructions.

The default label terminator is the nonprinting end-of-text character **ETX** (decimal code 3). This is accessed by `CHR$(3)` in BASIC. If your system cannot send this terminator, refer to the DT (Define Label Terminator) instruction later in this chapter.

Sending the Label Terminator and Other Nonprinting Characters

You can send the default terminator, along with other nonprinting characters (also known as ASCII control characters) such as a carriage return or line feed using a character string function like `CHR$` (provided by BASIC), or by producing the character directly from the keyboard. For simplicity, the examples in this chapter use the `CHR$` function. If necessary, substitute your language's string function (the Pascal equivalent is `CHR`, FORTRAN uses the `FORMAT` statement plus a string function).

When you use a function such as `CHR$`, you must separate it from the label string. To be sure that the character produced by the string function is sent immediately after label string without any extra space, you may need to concatenate, or link, the label string and the character string with one of the following symbols: `+`, `&`, or `;`. Use the symbol your system requires.

The following shows one method of concatenating your label string and terminator.

```
“LBThis is a label”+CHR$(3)
```

Check the ASCII character and keyboard table in your computer documentation for the proper value to use. The following lists the current BASIC CHR\$ equivalents.

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

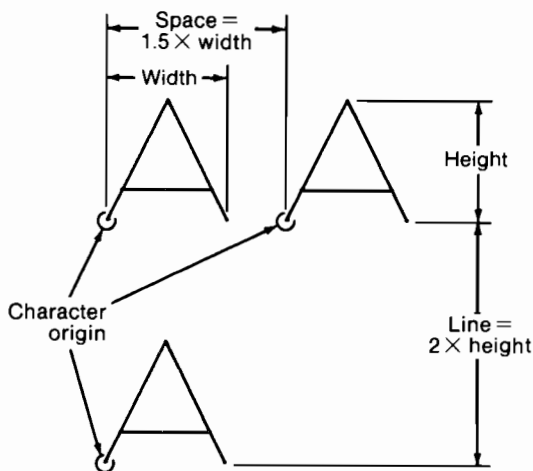
You can also produce these ASCII control characters using many computer keyboards. The following shows the combination of keys, when pressed simultaneously, that produce the same characters.

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

If you can produce the character directly from the keyboard in this way, you can enter the character within the label string; you do not need to use a concatenating character as you do with the CHR\$ function.

Positioning Labels

The following illustration defines the relationships of a space, a line, and a character's origin, width, and height, based on the default conditions for labels.



When you issue an LB instruction, the current pen location becomes the label origin. After drawing the first character, the pen moves to the next character origin. After drawing each subsequent character, the pen location is updated to the new character origin. This continues until the end of the character string, unless an embedded control character such as a carriage return or a line feed is included.

When a carriage return is included in a label string (i.e., before the label terminator), the pen moves to the carriage-return point (the pen location when the LB instruction was received). When either a line feed or an inverse line feed is included in a label string, the carriage-return point and the pen move vertically with respect to their current locations.

NOTE: The carriage-return point is updated to the current pen location after these instructions are executed: AA, AR, PA, PR, DI, DR, DF, IN, RO, and with PU or PD instructions when parameters are used. In addition, moving to a new point using the front-panel P1, P2, and CURSOR CONTROL buttons updates the carriage-return point to the new location. Although the current pen location is updated after an LB instruction, the carriage-return point is not. ■

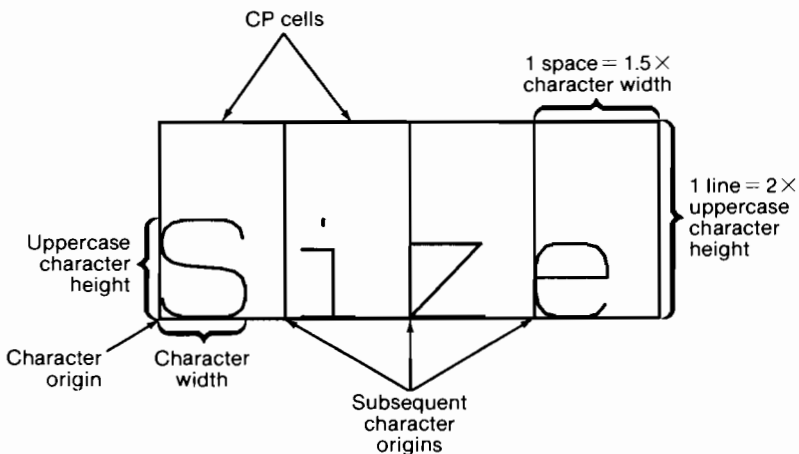
The plotter updates the carriage-return point after any of the following.

- Moving the pen using any of the following instructions: AA, AR, CP (vertically), PA, PD, PR, or PU.
- Moving the pen using the front-panel **CURSOR CONTROLS** or **P1**, **P2**, or **AXIS ALIGN** buttons.
- Setting plotter conditions using any of the following instructions: DF, DI, DR, DV, IN, or RO.
- Clearing or resetting the plotter, or rotating the coordinate system using the front-panel buttons.

NOTE: Although the current pen location is updated after an LB instruction, the carriage-return point is not. ■

The Character Plot Cell

Think of the character plot (CP) cell as a rectangular area around a character that includes blank areas above and to the right of the character. The character origin is always in the lower-left corner of the CP cell. The actual size of the CP cell depends on the specified character size and any extra space as specified by the ES (Extra Space) instruction. The default height of a CP cell is twice the height of any uppercase character. The default width is 1.5 times the character width of a capital A. The height and width of the CP cell are equivalent to one line and one space, respectively.



You can use the CP (Character Plot) instruction to position labels from the current pen location horizontally and/or vertically by specifying the number of CP cells to move.

Another instruction, LO (Label Origin), lets you position the complete label string relative to the current pen location. That is, you can center the complete label string relative to your current pen location, or left- or right-justify labels along axes or data points.

Enhancing Labels

You can adjust certain characteristics of any label in several ways. Some of these are by changing the size, shape, the label direction, or the spacing between characters in the label. Most of these depend on an understanding of the character plot cell, described above.

Adjusting Character Size

You can change the size of the characters in your label either by specifying an absolute size (using the SI (Size Absolute) instruction) or a relative size (using the SR (Size Relative) instruction) dependent on the distance between P1 and P2. Note that whenever you change the size of the characters, you also change the size of the CP cell.

Changing Character Spacing

To change spacing, use the ES (Extra Space) instruction. This instruction does not affect the character size; it only allows you to add to or subtract from the space around the character in the CP cell. You can use ES to adjust the vertical distance between lines and/or the horizontal space between characters.

Adjusting Label Direction

Under default conditions, the plotter draws labels parallel to the X-axis in a positive direction. When you rotate the coordinate system using the RO instruction, the label direction rotates with the axes. The following illustrates the default label direction. The label direction is also determined by the following.

- absolute direction (DI) or relative direction (DR)
- horizontal or vertical direction mode (DV)

The DI (Direction Absolute) instruction sets the angle of the label independently of P1 and P2. The DR (Direction Relative) instruction relies on the locations of P1 and P2 so that labels maintain the same relationship to the scaled plot. The DV (Direction Vertical) instruction stacks the characters vertically and is especially useful when labeling using the Kanji character set.

Labeling with Variables

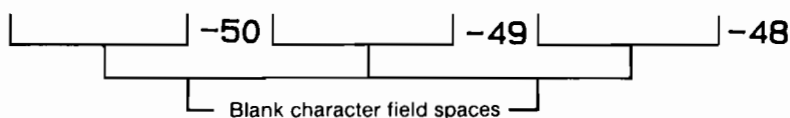
You may want to label the plot using variables (rather than literals) to define a label string. The principles for sending variables in label instructions are similar to those for sending variables in plotting instructions (refer to Chapter 4). However, the format for the character field is important because labels are positioned differently and plotted with extra spaces, depending on the format you use. Various computers and computer languages use different conventions to define the character-field format in which variables are printed. The following are three conventions.

Quotation mark	Used by many computers to define the literal characters to be sent; variables are not included within quotation marks.
Comma	Used by most computers between variables to cause the variable to be right-justified in a specific character-field width.
Semicolon	Used by most computers between variables for close spacing of variables in labels.

Use whatever convention is applicable to the programming language you are using and the spacing you want for your labels.

The following illustrates the use of the comma to establish fixed spacing when using variables for labeling. The number of blank character-field spaces and the printing of a sign vary with different computers.

```
X = -50
"LB",X,X+1,X+2,CHR$(3)
```



To avoid unexpected placement of the labels defined by variables, refer to your computer manual for a definition of the conventions used to define the variable spacing.

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	-32 768.0000 to 32 767.9999 CP cell widths	none
lines	real	-32 768.0000 to 32 767.9999 CP cell heights	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 *Positioning Labels* earlier in this chapter.) How the plotter interprets the parameters depends on the mode of direction: horizontal or vertical (refer to DV (Direction Vertical) instruction).

When the direction mode is horizontal (default), the plotter interprets the parameters as follows.

- **spaces** — moves the specified number of CP cells to the left or right depending on the sign of the parameter. Negative values move the pen to the left; positive values move the pen to the right. Left and right are relative to the current label direction.

A space is defined as the width of one character plot cell, or 1.5 times the character width (specified by SI or SR). The width includes any extra space adjustments made using the ES instruction.

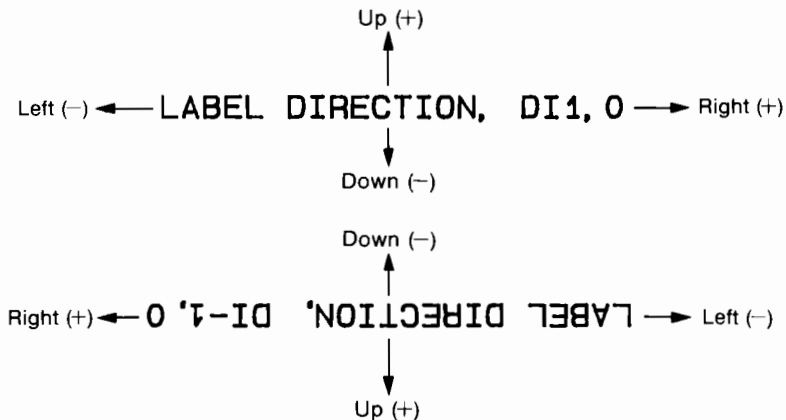
Note that this parameter does not affect the carriage-return point.

- **lines** — moves the specified number of CP cells up or down from the current pen location. Negative values move the pen down the specified number of lines; positive values move the pen up. Up and down are relative to the label direction.

A line is defined as the height of one character plot cell, or 2 times the uppercase character height (specified by SI or SR). The height includes any extra space adjustments made using the ES instruction.

This parameter moves the location of the carriage-return point up or down depending on the parameter value.

- **no parameters** — returns to the carriage-return point and moves one line down (carriage return, line feed). This location becomes the new carriage return point.



When the direction mode is vertical (refer to the DV instruction), the plotter interprets the parameters as follows.

- **spaces** — moves the specified number of CP cells up or down from the current pen location depending on the sign of the parameter. Negative values move the pen up; positive values move the pen down. This parameter does not affect the carriage-return point.
- **lines** — moves the specified number of CP cells left or right from the current pen location. Negative values move the pen to the left the specified number of lines; positive values move the pen to the right. This parameter moves the location of the carriage-return point to the left or right depending on the parameter value.
- **no parameters** — returns to the carriage-return point and moves one line to the left (carriage return, line feed). This location becomes the new carriage return point.

Refer to the DV (Direction Vertical) instruction later in this chapter for an example of using the CP instruction in vertical labeling.

CP instructions use the current pen position (up or down), and all moves are made with respect to the current character origin. The CP instruction affects only the placement of the next label; you must issue new CP instructions to affect subsequent labels.

EXAMPLE: In the following example, line 50 is printed on two lines to fit on this page; you should send line 50 to the plotter as one string. Note that the carriage return and line feed characters in line 50 of the program listing have the same effect as the CP instruction without parameters in line 60.

```
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, Direction Absolute
DR, Direction Relative
ES, Extra Space
LO, Label Origin
SI, Size Absolute
SR, Size Relative

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 a line chart, schematic drawings, blueprints, and survey boundaries.

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

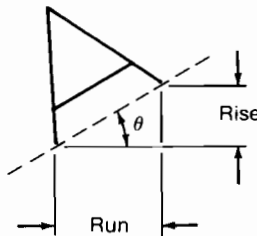
Parameter	Format	Range	Default
run (or $\cos \theta$)	real	-32 768.0000 to 32 767.9999	1
rise (or $\sin \theta$)	real	-32 768.0000 to 32 767.9999	0

REMARKS: The P1 and P2 settings have no effect on the label direction. The DI instruction updates the carriage-return point to the current pen location.

You can express the parameters in measured units as run and rise, or using the trigonometric functions cos and sin according to the following relationship:

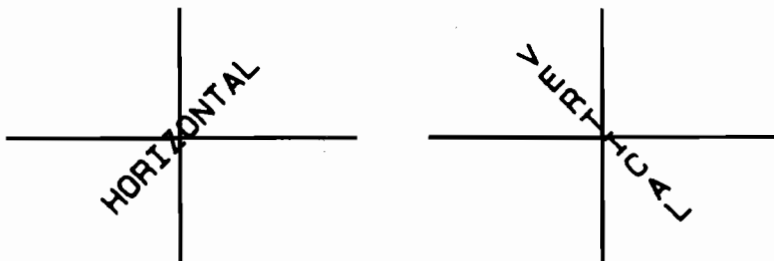
where run and rise = number of measured units, and

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



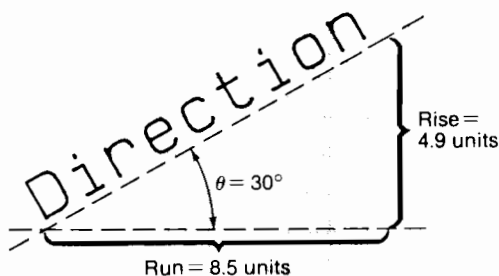
The positive run is defined as the direction in which the label is plotted (i.e., along the positive X-axis for horizontal mode and along the negative Y-axis for the vertical mode). The rise is perpendicular to the run. When labeling in horizontal mode, a positive rise parameter indicates that the label will slope in a positive direction along the Y-axis. When labeling in vertical mode (when letters are plotted upright, but stacked vertically), a positive rise parameter indicates that the label will slope in a positive direction along the X-axis.

The following show the positive run and rise directions for both modes.



Suppose you want to plot your label in the direction shown in the following graph. You can easily do this one of the two ways: measure the run and rise, or measure the angle. To use the first method, extend lines along the label and parallel to the X-axis. You can use these measurements of the run and rise as the parameters of the DI instruction (*DI8.5, 4.9*).

Or, if you know the angle, you can use the trigonometric values (since $\sin \theta / \cos \theta = \text{rise/run}$). In this example, $\theta = 30^\circ$; $\cos 30 = 0.866$ and $\sin 30 = 0.5$. Thus, you can use these as the parameters of the DI instruction (*DI.866,.5*). Whichever set of parameters you use, the label will be drawn in the same direction as shown in the following figure.



DI 8.5,4.9;
or
DI .866,.5;

If you know the angle, you can specify the actual cosine and sine values, or you can use the SIN and COS function available on most computers. A table of cosine values and sine values for selected angles follows.

θ	Cosine	Sine
0	1	0
-30	0.87	-0.50
-45	0.71	-0.71
-60	0.50	-0.87
-90	0	-1

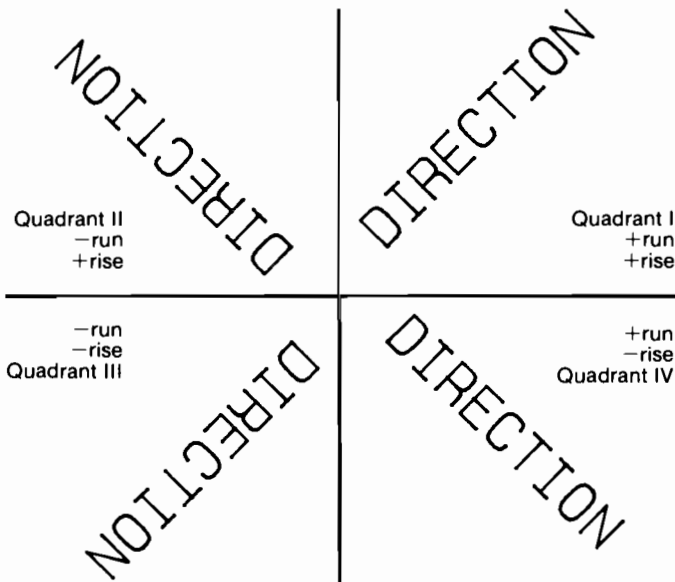
θ	Cosine	Sine
0	1	0
30	0.87	0.50
45	0.71	0.71
60	0.50	0.87
90	0	1

EXAMPLE: The size and sign of the two parameters in the instruction determine the amount of rotation. In the following illustration, the signs of the parameters determine the quadrant in which the label is drawn.

```

10  *Insert configuration statement here
20  PRINT #1, "IN;SP1;SI.3,.5;PA3000,3000;"
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

```



RELATED

INSTRUCTIONS: DR, Direction Relative
 SI, Size Absolute
 SR, Size Relative

ERRORS:

Condition	Error	Plotter Response
more than 2 parameters	2	uses first 2 parameters
1 parameter	2	uses previous direction
parameter out of range	3	uses previous direction
both parameters = 0	3	uses previous direction

DR, Direction Relative

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.*

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

Parameter	Format	Range	Default
run (or $\cos \theta$)	real	-32 768.0000 to 32 767.9999	1% of $P2_X - P1_X$
rise (or $\sin \theta$)	real	-32 768.0000 to 32 767.9999	0% of $P2_Y - P1_Y$

REMARKS: The plotter interprets the parameters as follows.

- **run** — specifies a percentage of $P2_X - P1_X$.
- **rise** — specifies a percentage of $P2_Y - P1_Y$.
- **no parameters** — establishes relative direction and sets the label direction according to the direction mode (as set by the DV (Direction Vertical) instruction).

The function of run and rise is similar to the DI instruction with the important difference that the run and rise parameters are percentages of the X- and Y-distances between P1 and P2. The size and sign of the two parameters determine the amount of rotation.

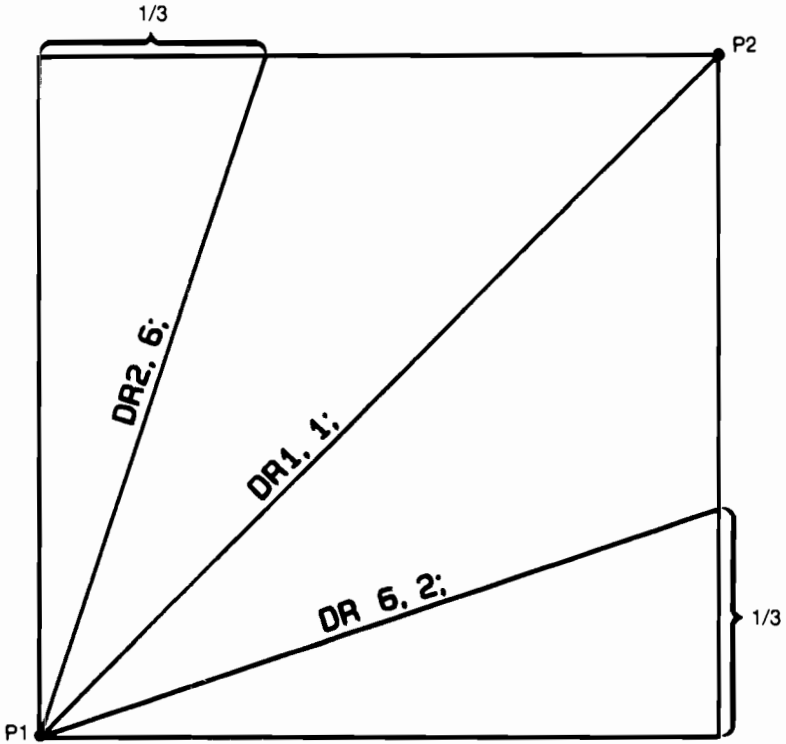
The following discusses the effects that changing P1 and/or P2 have on relative label direction. Think of the label directions as parallel to a line starting at the lower-left scaling point (usually P1) and intersecting at either the top or the right edge of the P1/P2 area. To calculate where the intersection is, first determine which is larger: the run or the rise. There are three possibilities.

- If run = rise, the line goes directly from P1 to P2.
- If run < rise, the line intersects the *top* of the plotting area, a fraction of the way *across* toward P2. The fraction is run/rise. Thus if run = 2 and rise = 6, the line intersects the top one-third (2/6) of the way toward P2.
- If rise < run, the line intersects the *side* of the plotting area, a fraction of the way *up* toward P2. The fraction is rise/run. Thus if rise = 2 and run = 6, the line intersects the side one-third (2/6) of the way toward P2.

Since labeling starts at the current pen location, labels are parallel to these lines, not necessarily on them. Also, negative parameters have the same effect on direction as described for the DI instruction.

EXAMPLE: The following shows what effect parameter equality and inequalities have on the label direction.

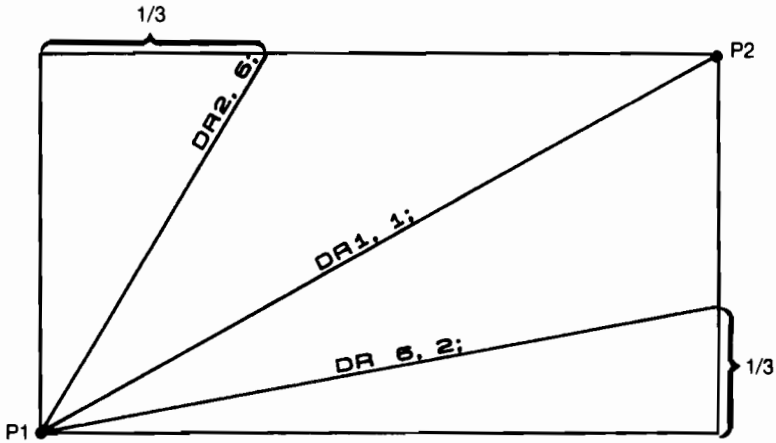
```
10  *Insert configuration statement here
20  PRINT #1, "IN;SP1;IP0,0,3600,3600;"
30  PRINT #1, "SC0,1000,0,1000;SR2,3;"
40  PRINT #1, "PA0,0;PR;PD0,1000,1000,0,0,-1000;"
50  PRINT #1, "PR-1000,0;PA1000,1000;PU0,0;DR1,1;"
60  PRINT #1, "CP15,.25;LB DR1,1;" +CHR$(3)
70  PRINT #1, "PU333.33,1000;PD0,0;PU;DR2,6;"
80  PRINT #1, "CP15,.25;LB DR2,6;" +CHR$(3)
90  PRINT #1, "PU1000,333.33;PD0,0;PU;DR6,2;"
100 PRINT #1, "CP15,.25;LBDR 6,2;" +CHR$(3)
110 PRINT #1, "SP0;"
120 END
```

When you replace line 20 in the previous example with the following line, you change the relative locations of P1 and P2.

```
20 PRINT #1, "**IN;SP1;IP5000,0,8600,2000;"
```

Note that each directional line is drawn the same fraction of the way toward P2; however, the angles of the lines have changed in order to maintain the correct relative direction. Notice also that the character size has changed because a relative size (SR) was specified in line 30.



RELATED

INSTRUCTIONS: DI, Direction Absolute
 IP, Input P1 and P2
 SI, Size Absolute
 SR, Size Relative

ERRORS:

Condition	Error	Plotter Response
more than 2 parameters	2	uses first 2 parameters
1 parameter	2	uses previous direction
parameter out of range	3	uses previous direction
both parameters = 0	3	uses previous direction

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 send the default label terminator (**ETX**, decimal code 3).

SYNTAX: *DT label terminator;*

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

REMARKS: The label initiated by the LB instruction can be terminated only by the label terminator. If your computer cannot send the default label terminator (**ETX**) using either the keyboard or as a character string function (**CHR\$**), you can use this instruction to specify a character your computer can send.

The character following the DT mnemonic is interpreted as the new label terminator. Label instructions react differently depending on which character you specify to be the label terminator, as follows.

printing character	terminates the label instruction and prints the character. The ASCII printing character decimal equivalents are codes 32 through 58 and 60 through 126.
nonprinting character (control character)	terminates the label instruction and performs the function specified by the character. The ASCII control characters are codes 1 through 31.

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

Refer to Appendix D for instructions when using this instruction with the Kanji character set.

EXAMPLE: The following program shows how to change the label terminator to printing and nonprinting characters. 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 a line feed
- CHR\$(13) sends a carriage return

NOTE: Although some instructions are printed on two lines to fit on this page, you should send these to the plotter as one string. ■

```

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)+"SP0;"
130  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.

ERROR:

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

DV, Direction Vertical

USE: Specifies vertical as the direction for subsequent labels. Use this instruction to “stack” horizontal characters in a column against a vertical axis. This is especially useful when using the Kanji character set.

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.

- **0** — sets the direction for labels to horizontal.
- **1** — sets the direction for labels to vertical.
- **no parameter** — sets the direction for labels to horizontal.

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.

```

L  L  L
0  0  0
1  2  3

L  L  L
0  0  0
4  5  6

L  L  L
0  0  0
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 a line feed
CHR$(13) sends a carriage return

```

```

10  *Insert configuration statement here
20  PRINT #1, "IN;SP1;PA3000,3000;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, "PA4000,3000;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
      E B
      F C
      G H I

      ABC
      DEF

      GHI

```

RELATED

INSTRUCTIONS: DI, Direction Absolute
DR, Direction Relative
LO, Label Origin

ERRORS:

Condition	Error	Plotter Response
more than 1 parameter	2	uses first parameter
parameter 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	-32 768.0000 to 32 767.9999 CP cells	0
lines	real	-32 768.0000 to 32 767.9999 CP cells	0

REMARKS: The plotter interprets parameters as follows.

- **spaces** — adds (positive number) or subtracts (negative number) spaces between characters. The “space” is determined by the current character plot cell. You can specify fractional spaces; for legibility, do not specify more than one extra space.
- **lines** — adds (positive number) or subtracts (negative number) lines between character lines. The “line” is determined by the current character plot cell. You can specify fractional lines.
- **no parameters** — defaults the spaces and lines between characters to the dimensions of the character plot cell, as set by the most recent SI or SR instruction.

EXAMPLE: The following illustrates the difference of negative spacing, positive spacing, and default spacing.

```
10 *Insert configuration statement here
20 PRINT #1, "IN;SP1;PA0,-3000;"
30 PRINT #1, "ES-.1,-.25;"
40 PRINT #1, "LBES-.1,-.25; causes"+CHR$(13)
    +CHR$(10)+CHR$(3)
50 PRINT #1, "LBthis spacing."+CHR$(3)
60 PRINT #1, "CP;CP;"
70 PRINT #1, "ES.2,.25;"
80 PRINT #1, "LBES.2,.25; causes"+CHR$(13)
    +CHR$(10)+CHR$(3)
90 PRINT #1, "LBthis spacing."+CHR$(3)
100 PRINT #1, "CP;CP;"
110 PRINT #1, "ES;"
120 PRINT #1, "LBES; causes"+CHR$(13)
    +CHR$(10)+CHR$(3)
130 PRINT #1, "LBthis spacing."+CHR$(3)
140 PRINT #1, "SP0;"
150 END
```

ES-.1, -.25; causes
this spacing.

ES.2, .25; causes
this spacing

ES; 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. Use this instruction to annotate drawings or create text-only charts.

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

(where CHR\$(3) is the ASCII character **ETX** (CTRL-C) or the label terminator you defined using the DT instruction)

Parameter	Format	Range	Default
<i>c...c</i>	label	any character	none

REMARKS: The characters following the LB instruction up to the label terminator form the label. The plotter draws all printing characters using the currently selected character set (refer to Chapter 9). You can include nonprinting characters such as a carriage return or line feed. These characters are not drawn, but cause the plotter to perform the specified function.

The direction, size, and slant of the characters and the spacing between characters assume default values if not previously specified by the DI, DR, SI, SR, SL, and/or ES instructions. The label begins at the current pen location unless its placement has been set using an LO, Label Origin, instruction. After drawing each character, the pen moves to the next origin of the following character. When finished drawing the label, the pen location is updated to be the next character origin. (Refer to *Positioning Labels* earlier in the chapter.)

Labels can be terminated only by sending a label terminator at the end of the character string or by clearing or resetting the plotter (refer to the User's Guide).

EXAMPLE: The following is an example of the LB instruction and the label it creates. Note that CHR\$(3) is the ASCII equivalent of the end-of-text character **ETX**.

```
10 'Insert configuration statement here
20 PRINT #1, "IN;SP1;PA100,100;"
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
DI, Direction Absolute
DR, Direction Relative
DT, Define Label Terminator
ES, Extra Space
LO, Label Origin
SA, Select Alternate Character Set
SI, Character Size Absolute
SL, Character Slant
SR, Character Size Relative
SS, Select Standard Character Set

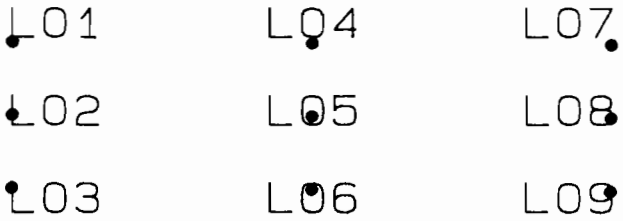
LO, Label Origin

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

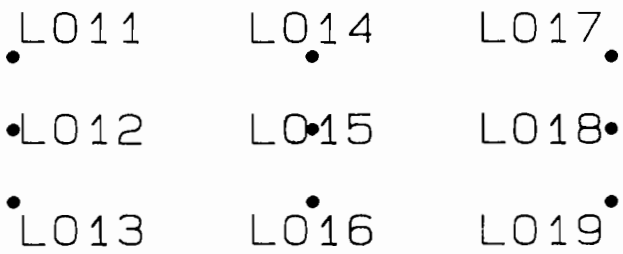
SYNTAX: LO *position number* ;

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

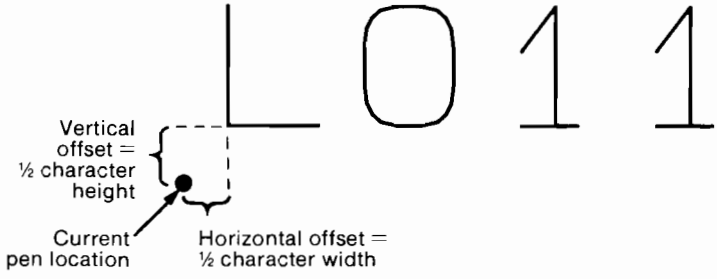
REMARKS: The following illustrates label origins 1 through 9. Each dot represents the current pen location.



The label origins resulting from LO 11 through LO 19 are the same as LO 1 through LO 9, except 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. (Any extra space in effect does not affect the label origin location.)

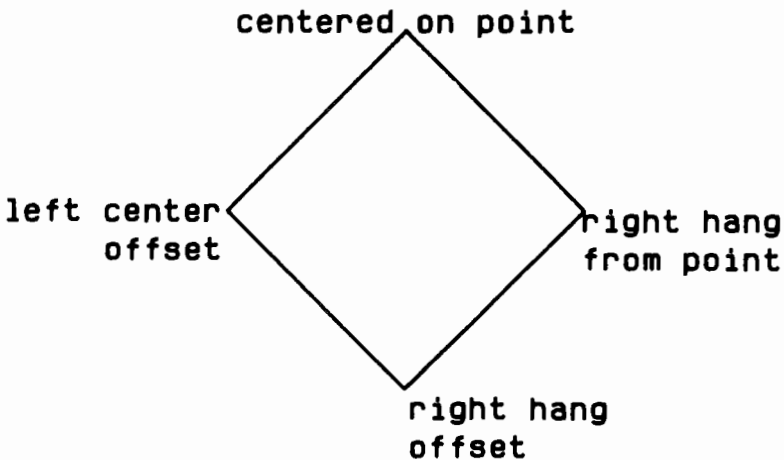


An LO instruction remains in effect until replaced by another LO instruction, or the plotter is initialized or set to default conditions.

Note that label origins 4 through 9 and 14 through 19 cause the plotter to store characters in a 150 character buffer. The label is not plotted until a carriage return or label terminator is received. The plotter truncates at 150 characters any label that exceeds that amount between carriage returns and/or label terminators.

EXAMPLE: The following shows some of the effects of different label origins.

```
10  *Insert configuration statement here
20  PRINT #1, "IN;SI.187,..269;SP1;PA0,1000;"
30  PRINT #1, "PD-1000,0,0,-1000,1000,0,0,1000;"
40  PRINT #1, "LO4;"
50  PRINT #1, "LBcentered on point"+CHR$(3)
60  PRINT #1, "PU-1000,0;LO18;"
70  PRINT #1, "LBleft center"+CHR$(13)+CHR$(10)
      + "offset"+CHR$(3)
80  PRINT #1, "PU0,-1000;LO13;"
90  PRINT #1, "LBright hang"+CHR$(13)+CHR$(10)
      + "offset"+CHR$(3)
100 PRINT #1, "PA1000,0;LO3;"
110 PRINT #1, "LBright hang"+CHR$(13)+CHR$(10)
      + "from point"+CHR$(3)
120 PRINT #1, "SP0;"
130 END
```



SI, Size Absolute Character

USE: Specifies the size of 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	-32 768.0000 to 32 767.9999 centimetres	0.285 cm
height	real	-32 768.0000 to 32 767.9999 centimetres	0.375 cm

REMARKS: An SI instruction without parameters, *SI;*, establishes the character size at default values. When used, the parameters specify the actual width and height of the character. 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. Subsequent examples show how negative parameters produce mirror image labels.

```
10 *Insert configuration statement here
20 PRINT #1, "IN;SP1;PA300,5000;"
30 PRINT #1, "LBPlot"+CHR$(3)
40 PRINT #1, "PA300,4200;"
50 PRINT #1, "SI1,1.5;LBPlot"+CHR$(3)
60 PRINT #1, "SP0;"
70 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;LBP1ot”+CHR$(3)
```

ɹoI9

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

```
“SI.3,-.45;LBP1ot”+CHR$(3)
```

bIOf

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

```
“SI-.3,-.45;LBP1ot”+CHR$(3)
```

ɹoIɹ

RELATED

INSTRUCTIONS: SR, Size Relative Character Set
DI, Direction Absolute
DR, Direction Relative
LB, Label

ERRORS:

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

SL, Slant Character

USE: Specifies the slant at which label characters 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.

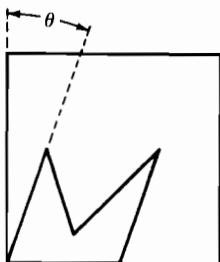
SYNTAX: SL *tangent*; or SL;

Parameter	Format	Range	Default
tangent	real	-32 768.0000 to 32 767.9999*	0 (no slant)

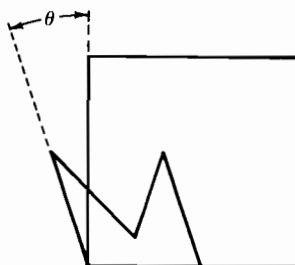
*This is the actual range. The practical range is ± 0.05 to ± 2 for default characters and ± 0.05 to ± 3.5 for large characters.

REMARKS: An SL instruction without parameters, *SL*; reestablishes the character slant to the default value of zero; that is, vertical (no slant). This is the same as *SL0*;

The parameter is the tangent of the angle θ from vertical, as shown in the following illustration. The tangent is interpreted as an angle ± 90 degrees from vertical.



Positive Slant



Negative Slant

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

Angle	Tangent	Angle	Tangent
0	0	0	0
-10	-0.18	10	0.18
-20	-0.36	20	0.36
-30	-0.58	30	0.58
-40	-0.84	40	0.84
-45	-1.00	45	1.00
-90	infinity	90	infinity (error)

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 you specify a new slant, or the plotter is either initialized or set to default conditions.

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

```

10  *Insert configuration statement here
20  PRINT #1, "IN;SP1;SI.7,1;PA1000,1000"
30  PI = 3.141593
40  A = TAN(20*(PI/180))
50  PRINT #1, "SL";A;" ;LBSlant"+CHR$(3)
60  PRINT #1, "PA1000,300;"
70  PRINT #1, "SL-.36;LBSlant"+CHR$(3)
80  PRINT #1, "SP0;"
90  END

```

Slant
Slant

NOTE: Before using the TAN function, check your computer documentation to see how your system interprets angles. This version of Microsoft BASIC interprets angles as radians, so line 40 in this program converts radians to degrees. On the HP Series

200 computers, execute the BASIC statement DEG before using the TAN function if you want to express the angle in degrees. ■

ERRORS:

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

SR, Size Relative Character

USE: Specifies the relative size of characters as a percentage of the distance between scaling points 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	-32 768.0000 to 32 767.9999 percentage	0.285 centimetres
height	real	-32 768.0000 to 32 767.9999 percentage	0.375 centimetres

REMARKS: An SR instruction without parameters, SR;, establishes relative character sizing and produces the same size characters (in centimetres) as listed for default sizes under the SI instruction. Once relative sizing is established, however, the character size adjusts (expands or contracts) with subsequent changes in the locations of P1 and P2.

The character size you specify with SR is a *percentage* of the plotter-unit distance between P1 and P2. The plotter calculates the actual character width and height from the specified parameters as follows.

$$\text{actual character width} = \text{width}/100 \times (P2_x - P1_x)$$

$$\text{actual character height} = \text{height}/100 \times (P2_y - P1_y)$$

For example, suppose P1 and P2 are located at -10 075, -7090 and 10 075, 7090, respectively. If you establish relative sizing and specify a width of 2 and a height of 3.5, the plotter would determine the actual character size as follows.

$$\text{width} = 2/100 \times 20\,150 = 403 \text{ plotter units or } 1.01 \text{ cm}$$

$$\text{height} = 3.5/100 \times 14\,180 = 496 \text{ plotter units or } 1.24 \text{ cm}$$

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 replaced with new relative parameters, an SI instruction is executed, or the plotter is either initialized or set to default conditions.

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

```
10  *Insert configuration statement here
20  PRINT #1, "IN;SP1;PA300,7000;SR;LBDEFAULT SIZE"
    +CHR$(3)
25  *
30  PRINT #1, "IP0,0,5500,5500;PA300,6500;"
40  PRINT #1, "LBNEW P1 AND P2 CHANGE LABEL SIZE"
    +CHR$(3)
45  *
50  PRINT #1, "PA300,6000;SR2.5,2.5;"
60  PRINT #1, "LBNEW SR INSTRUCTION"+CHR$(13)
    +CHR$(10)
70  PRINT #1, "CHANGES LABEL SIZE"+CHR$(3)+"SP0;"
80  END
```

DEFAULT SIZE

NEW P1 AND P2 CHANGE LABEL SIZE

NEW SR INSTRUCTION CHANGES LABEL SIZE

NOTE: Either negative SR parameters or switching the relative location of P1 and P2 will produce mirror images of labels. When P1 is in the lower left and P2 is in the upper right, a negative width parameter mirrors characters right-to-left. You can get a similar result by moving P1 to the right of P2 and a positive width parameter. When you combine both of these mirroring techniques, P1 in the lower right, P2 in the upper left, and a negative width parameter, the label appears normal. You can achieve a similar effect using the height parameter to mirror a label from top-to-bottom. ■

RELATED

INSTRUCTIONS: DI, Direction Absolute
DR, Direction Relative
IP, Input P1 and P2
LB, Label
SI, Size Absolute

ERRORS:

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

CHAPTER

9



Character Sets and Characters

The plotter has 20 fixed-space vector font character sets (sets 0–9 and 30–39) and the drafting set (set 99) immediately available for labeling. The Kanji character set (sets 100 and 101) is available only as an option. This chapter shows you how to switch from one character set to another for labeling with special characters, as found in mathematical expressions or foreign languages, and how to create characters not found in the sets.

Using Character Sets

The plotter's character sets all occupy equal horizontal space, and each character is drawn using a fixed number of vectors (lines). The drafting set is designed to provide reliable character recognition in situations where photo reduction may cause image degradation and loss of resolution. For example, the plotter draws the characters "B", "8", and "S" in such a way as to avoid confusion. The set also contains symbols that are used in drafting.

The next table lists the character sets with their ISO (International Standards Organization) registration number, if any. Refer to Appendix B for a complete illustration of the characters in each set.

Fixed-Space Vector Font	Character Set	ISO Registration Number
0	ANSI ASCII	006
1	9825 Character Set	—
2	French/German	—
3	Scandinavian	—
4	Spanish/Latin American	—
5	Special Symbols	—
6	JIS ASCII	014
7	Roman Extensions	—
8	Katakana	013
9	ISO IRV (International Reference Version)	002
30	ISO Swedish	010
31	ISO Swedish for Names	011
32	ISO Norway, Version 1	060
33	ISO German	021
34	ISO French	025
35	ISO United Kingdom	004
36	ISO Italian	015
37	ISO Spanish	017
38	ISO Portuguese	016
39	ISO Norway, Version 2	061
99	Drafting	—
100	Kanji	—
101	Kanji	—

The plotter lets you designate a standard character set and an alternate character set for labeling. 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. Character set 0 is as follows.

```
! "#$%&' () *+, - ./0123456789: ; <=>?@
ABCDEFGHIJKLMNPOQRSTUVWXYZ [\] ^ _ `
abcdefghijklmnopqrstuvwxyz { | } ~
```

Since the plotter uses this set automatically, you need select another set only when necessary. Most sets draw identical upper- and lowercase letters and numbers (the exceptions being sets 5, 7, 8, and 99). The differences between the character sets are the

additional characters needed for certain languages, for example, the ç in French/German set 2 and the Æ in Scandinavian set 3.

When you want to use another character set, you must first designate the character set either as a standard set or alternate set, before you can select it for labeling. Designate character sets using either the CS (Designate Standard Character Set) or CA (Designate Alternate Character Set) instructions. Select either the standard or alternate set for use with the SS (Select Standard Character Set) or SA (Select Alternate Character Set) instructions.

Designating and Selecting Character Sets

If you always intend to label with the default 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 before you can select them (using either SA or SS) for labeling.

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 within a label string using the shift-out ASCII control character (decimal code 14). Switch from the alternate set to the standard set using the shift-in ASCII control character (decimal code 15).

Special Characters

There are two ways to access special characters in any set.

- Use the equivalent ANSI ASCII 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 the 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
```

Some characters, such as the underscore in set 4, automatically backspace before drawing the character. To use these characters, include them in the label string *after* the character to be underscored or accented.

```
“CS0;CA4;LBS_E_T_0_”+CHR$(14)  
+“ S_E_T_4_”+CHR$(3)
```

S _ E _ T _ 0 _ SET4

Designing Characters

If you need a special label character or symbol that is not included in any of the character sets, you can use the UC, User-defined Character, instruction to create your own symbol. When you use the UC instruction, you will define your character using X,Y increments on a grid superimposed on a character plot cell. This lets the plotter draw your user-defined symbol using the same character sizing as the other characters you are using. For more information on the character plot cell, refer to *The Character Plot Cell* and the CP instruction in Chapter 8.

CA, Designate Alternate Character Set

USE: Designates a character set as the alternate character set to use 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	0-9, 30-39, 99, 100 & 101*	0

*The Kanji character set (sets 100 and 101) are available as an option on the HP-IB cartridge only. Refer to Appendix D for more information on using Kanji.

REMARKS: Specifies the character set to be used when the alternate set is selected by the SA instruction or the control character shift-out (decimal 14) in a label string. Refer to *Designating and Selecting Character Sets* earlier in this chapter.

- **set** — designates one of the following sets for all labeling operations when the alternate set is selected by the SA instruction prior to labeling or by the control character shift-out (decimal 14) within a label string.

0-9 and 30-39 designate fixed-space vector fonts.

99 designates the fixed-space vector drafting font.

100 and 101 designate the Kanji character sets.

- **no parameter** — designates the default character set of 0 (ANSI ASCII English). The plotter also selects this character set when initialized or set to default conditions.

The CA instruction does not select the character set for current use. After using the CA instruction, use the SA instruction to select it for use. Once you have used the SA instruction to select the alternate character set, a new CA instruction causes 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 you select it with an SA instruction or shift-out character.

EXAMPLE: Refer to the SA, Select Alternate Character Set, instruction for an example using this instruction.

RELATED

INSTRUCTIONS: CS, Designate Standard Character Set
SA, Select Alternate Character Set
SS, Select Standard Character Set

ERRORS:

Condition	Error	Plotter Response
more than 1 parameter	2	uses first parameter
parameter out of range	3	ignores instruction
invalid character set	5	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 containing the characters you will most use for labeling.

SYNTAX: CS *set*; or CS;

Parameter	Format	Range	Default
set	integer	0-9, 30-39, 99, 100 & 101*	0

*The Kanji character set (sets 100 and 101) are available as an option cartridge only. Refer to Appendix D for more information on using Kanji.

REMARKS: Specifies the character set to be used as the standard set when selected by the SS instruction or the control character shift-in (decimal 15) in a label string. Refer to *Designating and Selecting Character Sets* earlier in this chapter.

- **Set** — designates one of the following sets for all labeling when the standard set is selected by the SS instruction prior to labeling or by the control character shift-in (decimal 15) within a label string.

0-9 and 30-39 designate fixed-space vector fonts.

99 designates the fixed-space vector drafting font.

100 and 101 designate the Kanji character sets.

- **no parameter** — designates the default character set of 0 (ANSI ASCII English). The plotter also selects this character set when initialized or set to default conditions.

If you execute a CS instruction while the standard set is being used, 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 SA, Select Alternate Character Set, instruction for an example using this instruction.

RELATED

INSTRUCTIONS: CA, Designate Alternate Character Set
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
unsupported set	5	ignores instruction

SA, Select Alternate Character Set

USE: 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 ASCII control character "shift-out" (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 (decimal 15) is encountered, or the plotter is initialized or set to default conditions.

EXAMPLE: Although some instructions are printed on two lines to fit on this page, you should send these to the plotter as one string.

```

10  *Insert configuration statement here
20  PRINT #1, "IN;SP1;PA-1400,-1400;"
30  PRINT #1, "CS0;CA4;"
40  PRINT #1, "LBEEnglish"+CHR$(3)+"CP;"
50  PRINT #1, "LBor Spanish--"+CHR$(3)+"SA;
    LB#su compan"+CHR$(124)+"ia?"
60  PRINT #1, "CP;CA2;SS;LBor French--"+CHR$(3)
    +"SA;LBde fa\on que"+CHR$(15)+CHR$(3)
70  PRINT #1, "CP;CA33;LBor German--"+CHR$(3)
    +"SA;LB1{~b sich anhand"+CHR$(3)
80  PRINT #1, "SP0;"
90  END

```

```

English
or Spanish--¿su compañía?
or French--de façon que
or German--läßb sich anhand

```

NOTE: If your keyboard does not include the backslash (\) in line 60, send ASCII decimal code 92. If your keyboard does not include the open brace ({} or tilde (~) in line 70, send ASCII decimal codes 123 and 126, respectively. ■

RELATED

INSTRUCTIONS: CA, Designate Character Set Alternate
 CS, Designate Standard Character Set
 SS, Select Standard Character Set

ERROR:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameters

SS, Select Standard Character Set

USE: 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” (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.

EXAMPLE: Refer to SA, Select Alternate Character Set, instruction for an example using this instruction.

RELATED

INSTRUCTIONS: CA, Designate Alternate Character Set
CS, Designate Standard Character Set
SA, Select Alternate Character Set

ERROR:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameters

UC, User-defined Character

USE: Draws characters of your own design. Use this instruction to create characters or symbols not included in the plotter's character sets or to draw logos.

SYNTAX: UC (*pen control*,) *X-increment*, *Y-increment* (...), (*pen control*) (...); or UC;

Parameter	Format	Range	Default
pen control	integer	9999 to 32 767 = pen down -9999 to -32 768 = pen up	pen up
X-increment, Y-increment	integer	-9998 to 9998	none

REMARKS: The characters are drawn with relative increments using a primitive grid that is superimposed on the character plot (CP) cell. Interpret the parameters as follows.

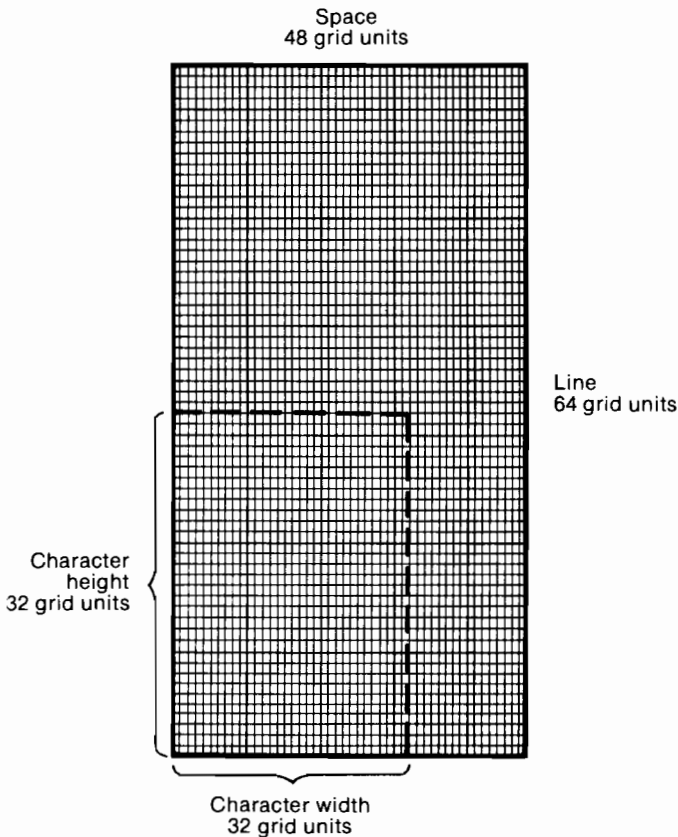
- **pen control** — is for lifting and setting down the pen, and is interpreted as follows.
 - pen down** — any integer within 9999 to 32 767, inclusive.
 - pen up** — any integer within -9999 to -32 768, inclusive.
- **X-increment, Y-increment** — any integer within -9998 to 9998, inclusive.
- **no parameters** — moves the pen one character plot cell in a positive direction (refer to the DI (Direction Absolute) and DR (Direction Relative) instructions in Chapter 8).

The primitive grid resolution is as follows.

	Horizontal mode	Vertical mode
1 CP space	48 grid units	44 grid units
1 CP line	64 grid units	64 grid units

Characters defined within a 32×32 unit grid use the same amount of cell space as each character in any of the plotter's

character sets. That is, each character in a 32×32 unit grid is proportionate to all other characters. The grid imposed on the character plot cell and the 32×32 unit character space is shown in the following illustration.



The UC instruction initially raises the pen (regardless of the current pen status). To draw a character, then, you must include at least one pen down. The pen remains 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.

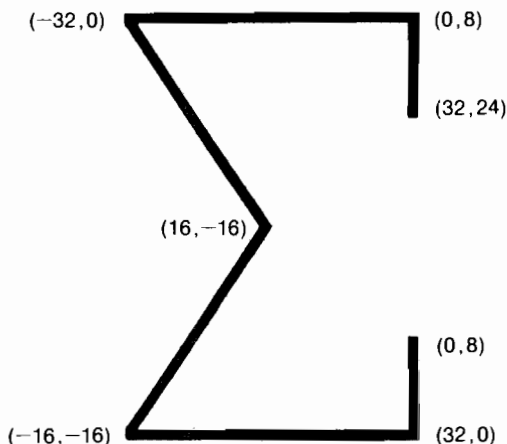
The UC instruction establishes your current pen location as the lower-left corner of the grid; this is the character plot origin (0, 0).

Using the pen control parameters and X,Y increments, move the pen relatively to the points that define your character. (For more information regarding relative movement, refer to *Using Absolute and Relative Movement* in Chapter 4.)

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.

Confining your user-defined character to the width and height specified above (32×32 grid units) 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 adjacent character plot cells. 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 (the lower-left corner). 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.

EXAMPLE: The following illustration shows a sigma on a character plot grid, and the program line that draws it.



```
“UC32,24,9999,0,8,-32,0,16,-16,-16,-16,32,0,0,8;”
```


The following uses the program line on the previous page to illustrate its proportions compared with uppercase letters.

```

10  *Insert configuration statement here
20  PRINT #1, "IN;SP1;PA1000,1000;"
30  PRINT #1, "UC32,24,9999,0,8,-32,0,16,-16,-16,-16,
      32,0,0,8;"
40  PRINT #1, "LBIGMA"+CHR$(3)
50  PRINT #1, "SP0;"
60  END

```

Σ I G M A

Note that adding an SI or SR instruction to line 20 does not change the relative sizes between the sigma and the other letters.

The following defines a resistor symbol that extends lengthwise beyond one character plot cell. Remember, on completion of a user-defined character, the pen moves one CP cell to the right of the character origin. The CP instruction (line 40) is included, then, to prevent the 1000 from overlapping the resistor symbol. Line 30 is shown 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;PA1000,5000;"
30  PRINT #1, "UC0,16,9999,7,0,6,16,12,-32,12,32,
      12,-32,12,32,12,-32,6,16,7,0;"
40  PRINT #1, "CP2,0;LB1000 ohms"+CHR$(3)
50  PRINT #1, "SP0;"
60  END

```

⚡ 1000 ohms

ERRORS:

Condition	Error	Plotter Response
odd number of increments	2	ignores odd increment
parameter out of range	3	terminates instruction with out of range parameter

CHAPTER

10



Changing Picture Area and Orientation

For most plotting situations, the pen 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.

Windowing: Setting Up Soft-Clip Limits

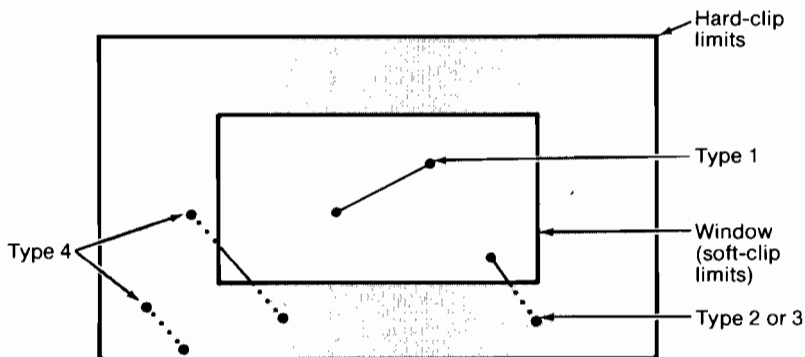
Soft-clip limits temporarily restrict pen movement to a rectangular area or window. When you turn on or initialize the plotter, or set it to its default conditions, the soft-clip limits are the same as the hard-clip limits. You establish the soft-clip window using the IW (Input Window) instruction. This has the same effect as the hard-clip limits; the plotter does not draw vectors or labels outside the window. When the pen reaches the window boundary, it lifts and stops moving. It remains in this location until another instruction returns the pen to a point within the window limits.

The following illustration shows that there are four types of line segments that you can specify from one point to another.

Last Point

New Point

- | | | |
|------------------------|----|---------------------|
| 1. Inside window area | to | inside window area |
| 2. Inside window area | to | outside window area |
| 3. Outside window area | to | inside window area |
| 4. Outside 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 3000×2000 plotter units in its X- and Y-axes, respectively, its

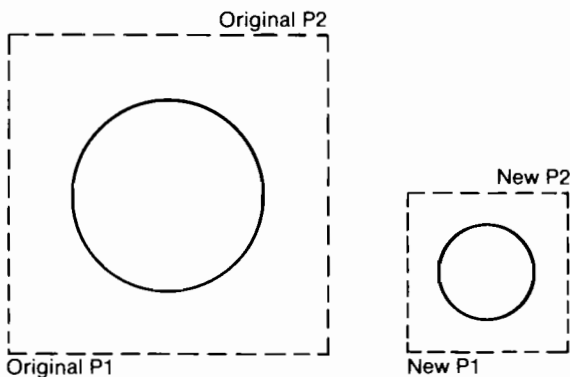
aspect ratio is 3:2. To enlarge the plot, set P1 and P2 to define a larger area that maintains a 3:2 ratio.

The following illustrates this using a square 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, "PAS,5;CI3;"
50 PRINT #1, "IP7000,2000,9000,4000;"
60 PRINT #1, "PAS,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 paper for a side-by-side comparison. This could be useful in 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. That is, you can use the IP instruction to move the location of P1 only; P2 will retain the

same relative distance from P1. (Depending on the locations of P1 and P2 when you move P1, P2 could move outside the hard-clip limits. The scaling remains in effect for the entire P1/P2 area, although some of the area is outside the hard-clip limits.)

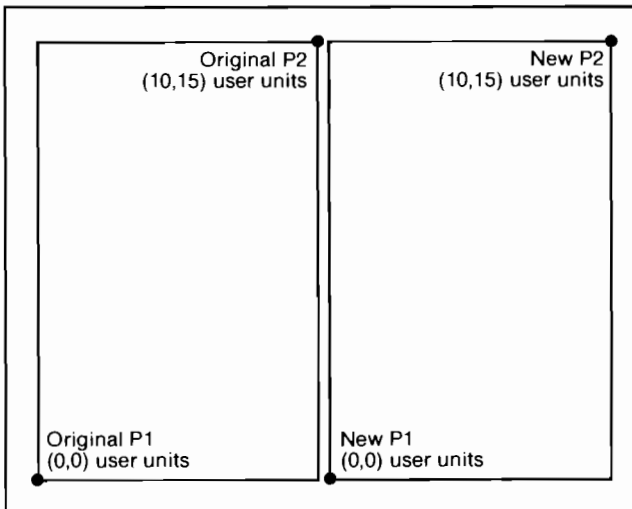
When drawing equal-sized pictures on the same paper, let P1 and P2 define an area that is no more than half of the current plotting area. Restricting your plot to this area ensures that two complete pictures of equal size would fit within the current hard-clip limits.

For example, the following program locates P1 and P2 on the left side of the paper and scales the area. After drawing a boundary using the scaling parameters, the IP instruction moves only the P1 location to the right side of the paper. Since P2 automatically tracks P1, the scaled 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: Neither of the P1/P2 frames are 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, an interesting phenomenon known as 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 DI (Direction Absolute), DR (Direction Relative), and/or the SR (Size Relative) instructions. The DI, DR, and SR instructions are discussed in Chapter 8, *Labeling Your Plots*.

The following program uses a subroutine to draw the exact same picture (an arrow) four times. 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.)

```

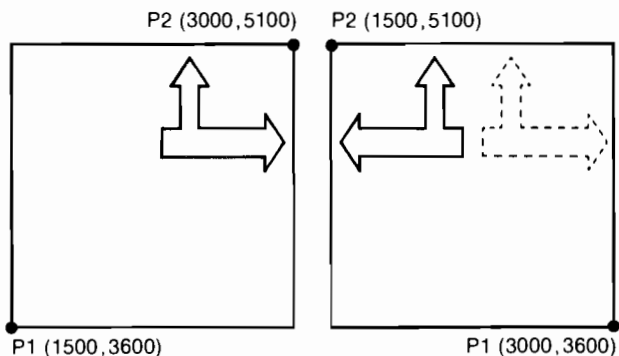
10  *Insert configuration statement here
20  PRINT #1, "IN;SP1;IP1500,3600,3000,5100;"
30  PRINT #1, "SC-15,15,-10,10;"
40  GOSUB 130
50  PRINT #1, "IP3000,3600,1500,5100;"
60  GOSUB 130
70  PRINT #1, "IP1500,5100,3000,3600;"
80  GOSUB 130
90  PRINT #1, "IP3000,5100,1500,3600;"
100 GOSUB 130
110 PRINT #1, "SP0;"

```

```

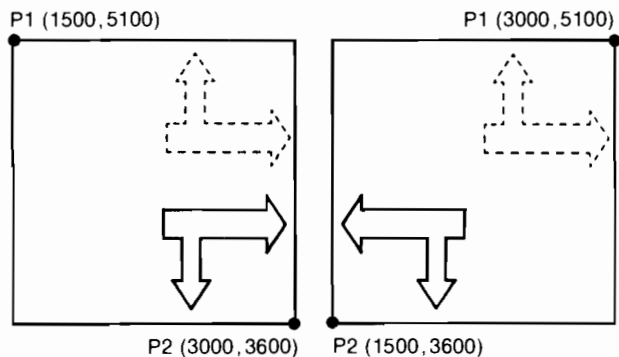
120 END
130 'draws the arrow
140 PRINT #1, "PA1,2;PD1,4,3,4,3,7,2,7,4,9,6,7,
      5,7;"
150 PRINT #1, "PD5,4,12,4,12,5,14,3,12,1;"
160 PRINT #1, "PD12,2,1,2;PU;"
170 RETURN

```



*Normal
(Lines 20-40)*

*Reversed
(Lines 50-60)*



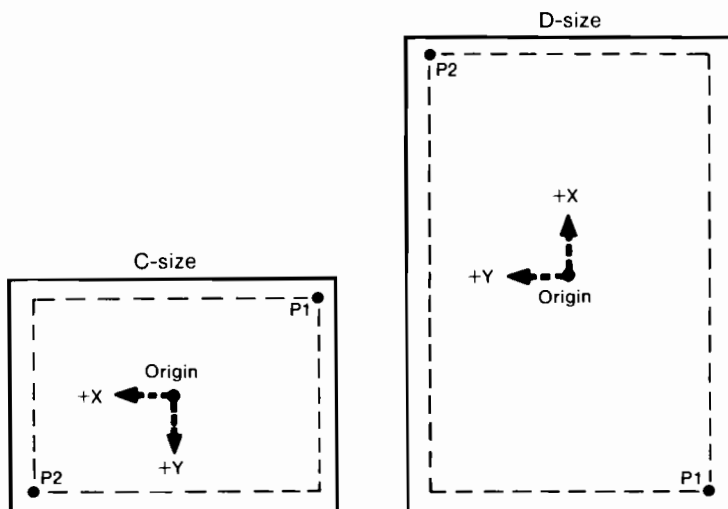
*Upside Down
(Lines 70-80)*

*Upside Down and Reversed
(Lines 90-100)*

Rotating a Picture

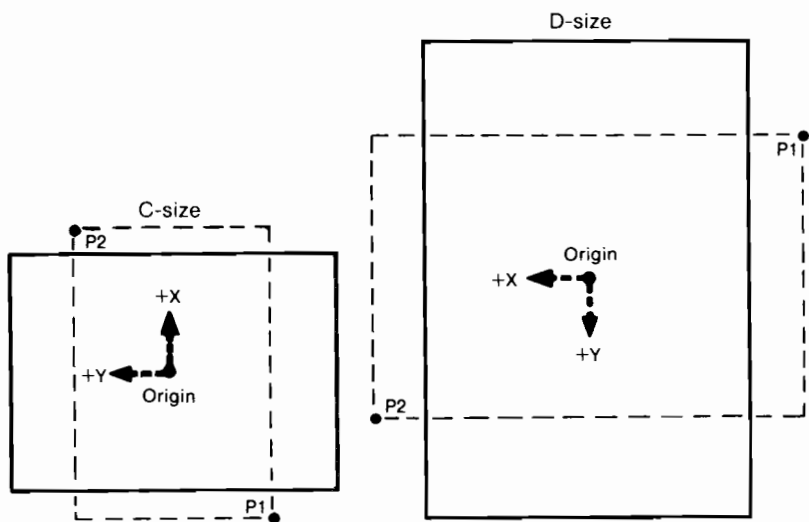
The plotter always sets the X-axis parallel to the longest edge of your media. However, you can change this orientation using the RO (Rotate) instruction to rotate the coordinate system 90 degrees. Note that you can rotate the orientation only once, and then rotate it back again. Rotations are not cumulative.

The following shows the default orientation of the axes and locations of P1 and P2.



Front of Plotter

The following illustration shows the orientation of the axes and locations of P1 and P2 after using the RO instruction with a 90 degree rotate parameter. The X,Y coordinates for P1 and P2 do not change. Note that the direction of rotation is dependent on paper size.



Front of Plotter

Note that the locations of P1 and P2 are now off the paper. (The front-panel Rotate button rotates the coordinate system and places P1 and P2 within the hard-clip limits; refer to 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. This is equivalent to using the front-panel **ROTATE** button. However, when you reset your coordinate system to its default orientation, you must 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 learn this information from the plotter by using the OW (Output Window), OH (Output Hard-clip Limits), and OP (Output P1 and P2) instructions described later in this chapter.

Output instructions are a special breed of HP-GL instructions. They perform no plotting. They provide information to be read by the computer. Read *Notes for Obtaining Plotter Output* in Chapter 11 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 is restricted to this area. Use this instruction when you want to be sure that the plotter draws only the portion of your plot that falls within a specific plotting area.

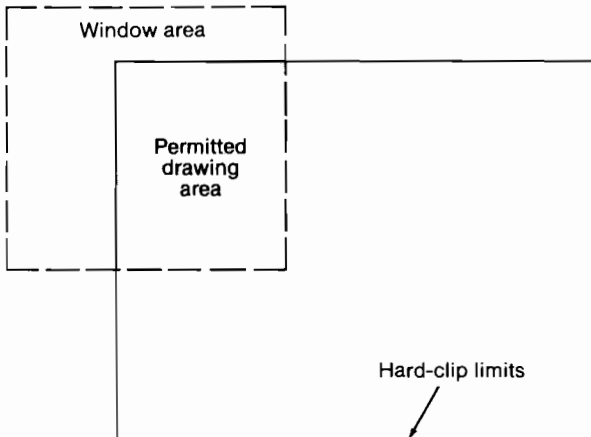
SYNTAX: IW X_1, Y_1, X_2, Y_2 ; or IW;

Parameter	Format	Range	Default
X_1, Y_1, X_2, Y_2	real	-32 768.0000 to 32 767.9999 current units*	hard-clip limits for paper size

*Regardless of scaling, the plotter rounds all decimal fractions to the appropriate integer (refer to *A Note About Rounding* in Chapter 3). The rounded integer reflects 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 RO (Rotate) instruction moves the window outside of the hard-clip limits.

When you turn the plotter on, the window is automatically set to the (mechanical) hard-clip limits. You can define a window anywhere within (or congruent with) 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. ■

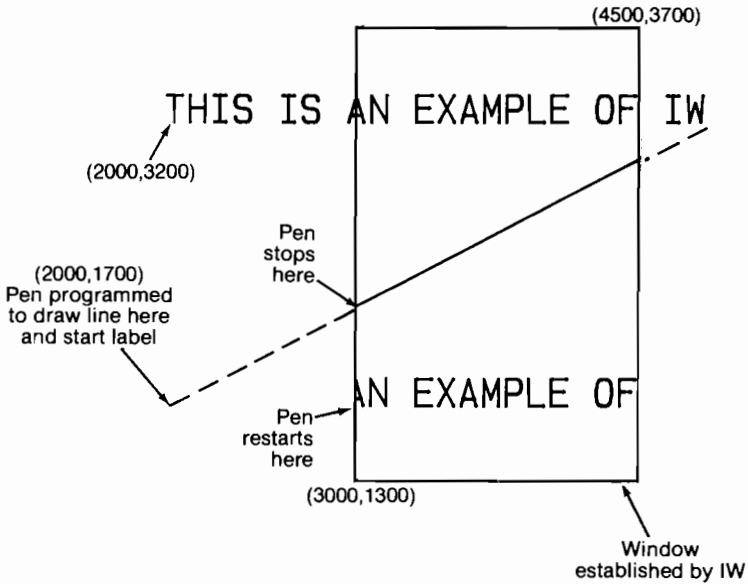
The IW instruction remains in effect until another IW instruction is executed, or the plotter is initialized or set to default conditions.

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.2..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****INSTRUCTION:** 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;

RESPONSE: $X_{LL}, Y_{LL}, X_{UR}, Y_{UR}$ [TERM]

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 response.

ERROR:

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 IW (Input Window) instruction to programmatically set the window to P1 and P2.

SYNTAX: OP;

RESPONSE: P1_X,P1_Y,P2_X,P2_Y [TERM]

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 OS, Output Status instruction).

EXAMPLE: Note that your computer may use different statements and/or a different 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 here
20  PRINT #1, "IN;OP;"
30  INPUT #1,A,B,Y,Z
40  PRINT A,B,Y,Z
```

RELATED

INSTRUCTION: IP, Input P1 and P2

ERROR:

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;

RESPONSE: $X_{LL}, Y_{LL}, X_{UR}, Y_{UR}$ [TERM]

REMARKS: When scaling is on, the coordinates are expressed in user units; otherwise, they are in plotter units. The coordinate pairs represent the lower-left and upper-right corners of the current window. 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
```

RELATED

INSTRUCTION: IW, Input Window

ERROR:

Condition	Error	Plotter Response
1 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.

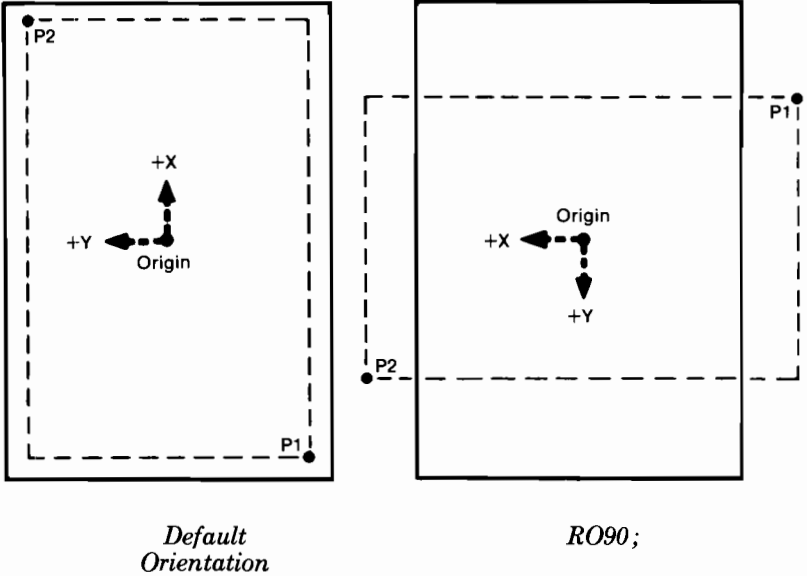
Rotations are not cumulative; you can toggle the rotate function on and off only. Scaling points P1 and P2 rotate with the coordinate system. However, they maintain the same X,Y coordinate values as before the rotation. This means that P1 and P2 can be located outside of the hard-clip limits. Follow *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 executing an

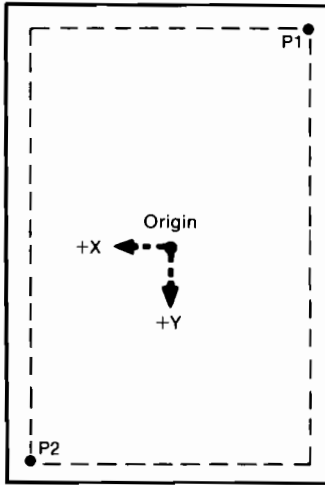
OA (Output Actual Pen Location) or OC (Output Commanded Pen Location) instruction after the rotation (refer to Chapter 11).

The RO instruction remains in effect until the rotation is changed by another RO instruction, a front-panel reset, or you turn the plotter off and on.

EXAMPLE: The following illustration shows the default orientation on D-size paper and the result of rotating the orientation (*RO90;*) without relocating P1 and P2.



The next illustration shows the locations of P1 and P2 when you follow the rotation with the IP instruction.



RO90; IP;

ERRORS:

Condition	Error	Plotter Response
more than 1 parameter	2	uses first parameter only
parameter out of range	3	ignores instruction
<i>RO90;</i> after plot is already rotated 90 degrees	none	ignores instruction

Notes

CHAPTER

11



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 10 and 12 for descriptions of additional output instructions.

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 the 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 the 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 13 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 an 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 OA (Output Actual Pen Status) instruction and the OC (Output Commanded Pen Status) 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 OI (Output Identification) instruction outputs the plotter ID to the computer. The OO (Output Options) instruction outputs the capabilities of the plotter.

Obtaining Error Information

Two HP-GL instructions are useful for error retrieval; these are the OE (Output Error) instruction and the IM (Input Mask) 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 cause the front-panel **VIEW** light to flash and set the error bit of the status byte and response to the OS (Output Status) 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 a service request message. ■

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 P	User units: real number, ≤ 11 digits (including sign and decimal point) 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	
0	7	128	8
1	6	64	7
2	5	32	6
3	4	16	5
4	3	8	4
5	2	4	3
6	1	2	2
7	0	1	1

Preset

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.

Initializing the plotter or sending an IM instruction without parameters sets all parameters to their default values.

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 table on the following page. 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 light will display on the front panel. Use the output error instruction, OE (described later in this chapter) to output the exact error number to your computer.

E-Mask Bits

Error Number	Meaning	Bit Number	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 that you want to generate a request in the table on the following page. 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 Number	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 Number	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 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 buttons.

NOTE: When in digitize mode, use the OD (Output Digitized Point) to determine the above information. Refer to Chapter 12. ■

SYNTAX: OA ;

RESPONSE: X,Y,pen status [TERM]

REMARKS: The pen location and up/down 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).

Only use the OA instruction when pen sorting is off. Because pen sorting can change the order in which vectors are drawn, the plotter may not execute the OA instruction at the desired location. You could get unexpected results if you are basing subsequent plotting on these coordinates.

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

ERROR:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

OC, Output Commanded Pen Status

USE: Outputs the location (X,Y coordinates) 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;

RESPONSE: X,Y,pen status [TERM]

REMARKS: When scaling is off, the X,Y coordinates are in plotter units; integers. When scaling is on, coordinates are interpreted as user units; real numbers with up to four decimal digits (e.g., 1.1234). 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 OA, Output Actual Status, instruction. Refer to the OA, Output Actual Status, instruction example, changing the OA to OC.

RELATED

INSTRUCTIONS: OA, Output Actual Status

ERROR:

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;

RESPONSE: error number [TERM]

REMARKS: The OE instruction outputs an integer (within the range 0 to 8) 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 you 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
8	Not used

Executing an OE instruction clears bit position 5 of the status byte (if previously set) and the front-panel error light (if lit) is turned off (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;OE;"
30  INPUT #1,A
40  PRINT A
50  END

```

```

20  PRINT #1, "IN;SP1;PA1000,1000,20;ED;OE;"

```

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).

RELATED

INSTRUCTION: IM, Input Mask

ERROR:

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;

RESPONSE: 40,40 [TERM]

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).

ERROR:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

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 you are writing a program that can run on more than one plotter model.

SYNTAX: OI;

RESPONSE: 7570A [TERM]

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).

ERROR:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

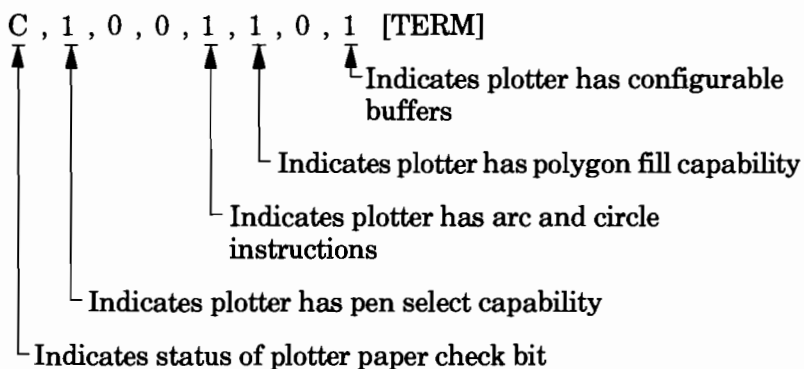
OO, Output Options

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;

RESPONSE: n,n,n,n,n,n,n,n [TERM]

REMARKS: The plotter outputs eight ASCII integers as follows, all separated by commas. A zero indicates that the particular plotter function is not supported.



C takes on a value of either 0 or 2. When paper is loaded and unmarked, the value is 0. After plotting on the paper, the value will be 2. The value is set to zero when you load new paper. It cannot 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).

ERROR:

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;

RESPONSE: status number [TERM]

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 Number
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; 0 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.

If the output response is 1, the largest decimal value that can be subtracted is 1. This means the corresponding condition (pen down) has been met.

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

ERROR:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

OT, Output Carousel Type

USE: Increases compatibility with other plotters. This plotter cannot distinguish what type of pen carousel is present.

SYNTAX: OT;

RESPONSE: -1, 255 [TERM]

ERROR:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

Notes

CHAPTER

12

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. This chapter discusses the HP-GL 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 plotter. If not, refer to Chapter 11, *Obtaining Information From the Plotter*.

NOTE: The HP 150 Touchscreen does not support the ability to read data (such as digitized points) from the plotter when in an HP-IB configuration. Some computers (such as HP Series 80 computers) require an I/O ROM to obtain digitized points. Other computers using the HP-IB interface cannot digitize while the plotter is in listen-only mode. In listen-only mode, the plotter cannot send the coordinates of a digitized point to the computer. ■

Using Digitizing Instructions

The HP-GL digitizing instructions, in the order used, are:

- DP, Digitize Point
- OD, Output Digitize Point
- DC, Digitize Clear

Use DP (Digitize Point) to enter digitize mode. After entering the point, use OD (Output Digitized Point) to output the X,Y coordinates of the point and the pen status (up/down). The DC (Digitize Clear) 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, which is available as an accessory (Part No. 07585-60191). Load the sight into the pen holder just as you would load a pen (refer to the User's Guide for more information). Digitize with the sight in the pen down position for the highest accuracy.

NOTE: To avoid smearing ink on the tip of the digitizing sight or possibly damaging the sight, do not load the digitizing sight directly into the pen carousel. ■

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.
2. Have the program display or print 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** buttons. 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 OD (Output Digitized Point) 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 11 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 lines 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, no examples are given. For information regarding interfacing and handshaking, refer to Chapter 14.

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

ERROR:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

DP, Digitize Point

USE: Places the plotter in digitize mode turning on the front-panel **ENTER** light. Use the OD instruction to input the coordinates of a point 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: DC, Digitize Clear
 OD, Output Digitized Point
 OS, Output Status

ERROR:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

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;

RESPONSE: X,Y,pen status [TERM]

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 (RS-232-C or HP-IB).

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: DC, Digitize Clear
 DP, Digitize Point
 OS, Output Status

ERROR:

Condition	Error	Plotter Response
1 or more parameters	2	ignores parameter(s)

CHAPTER

10

Device-Control Instructions

Device-control instructions differ from HP-GL 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 this 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 14 discusses the device-control instructions used in establishing interfacing and handshaking conditions.

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 HP-GL 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 HP-GL syntax.
- Device-control instructions don’t enter the plotter’s parser but are processed immediately. (HP-GL instructions enter the parser 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 HP-GL; 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 14 use the CHR\$ function. The following illustrates how the ESC . L instruction is sent using this function.

```
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).
- () All items 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 output response terminator. The default output response terminator when using an RS-232-C interface is a carriage return (**CR**); you can change this using the ESC . M instruction (described in Chapter 14). All output responses in an HP-IB configuration terminate with a carriage return and line feed (**CR LF**).

Using Device-Control Instructions

All device-control instructions with the word “output” in their title cause the plotter to generate an output response. Immediately after sending a device-control instruction, have your computer read the output response into a variable. Refer to *Notes for Obtaining Plotter Output* in Chapter 11.

Setting the Plotter to Known Conditions

You should already be acquainted with the following methods for establishing known plotter conditions.

- turn the plotter power switch on
- reset the plotter using the front-panel buttons
- send the DF or IN instruction to the 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 HP-GL instructions in the order in which 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 parsed HP-GL instruction and clears the buffers of all remaining HP-GL 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 14.

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.

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 the physical I/O buffer, polygon buffer, and pen sort buffer, or all buffers (depending on parameters).

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, 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, (in Eavesdrop only) 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 parse information from the computer in an Eavesdrop environment.
- **ESC.Z** or **ESC.)**, Programmed-Off, causes the plotter to pass data between the computer and terminal in an Eavesdrop environment. These instructions have no effect in a standalone environment or over HP-IB.

ESC.A, Output Identification

USE: Outputs the plotter's model number.

SYNTAX: **ESC.A**

RESPONSE: model number, firmware revision level [TERM]

REMARKS: The plotter outputs a 9-character string consisting of its model number (7570A) followed by a comma and the firmware revision number. Be sure to read the output responses appropriately. An HP-IB interface terminates the response with a carriage return and line feed; an RS-232-C interface with a carriage return, or as set by the **ESC.M** instruction (refer to Chapter 14).

RELATED

INSTRUCTION: **OI**, Output Identification

ESC.B, Output Buffer Space

USE: Outputs the plotter's currently available logical I/O buffer space.

SYNTAX: **ESC.B**

RESPONSE: available logical I/O buffer space [TERM]

REMARKS: The plotter outputs the number of unused bytes in the logical I/O buffer. The response can be as low as 0, especially if you reduce the size of the buffer (with the **ESC.T** or **ESC.@**

instructions). 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.

An HP-IB interface terminates the response with a carriage return and line feed; an RS-232-C interface with a carriage return, or as set by the ESC . M instruction (refer to Chapter 14).

RELATED

INSTRUCTION: ESC . L, Output Buffer Size When Empty

ESC . E, Output Extended Error

USE: *For RS-232-C users*, this instruction outputs any RS-232-C related I/O error. Use this instruction when debugging a program to determine which errors have occurred. Additionally, when used in conjunction with the ESC . @ instruction (refer to Chapter 14), you can perform block I/O error checking.

For HP-IB users, this instruction outputs a number defining a device-control error and turns off the front-panel error light (unless an HP-GL error has also occurred). Use the instruction to determine what type of device-control instruction error has occurred (if any).

SYNTAX: ESC . E

RESPONSE: error number [TERM]

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.

(Table continues)

Error No.	Meaning
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 data 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.

An HP-IB interface terminates the response with a carriage return and line feed; an RS-232-C interface with a carriage return, or with the character specified by the ESC.M instruction.

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. This instruction also aborts any plotting currently in progress. 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

INSTRUCTION: ESC . K, Abort Graphics

ESC . K, Abort Graphics

USE: Aborts any partially decoded HP-GL instruction and discards remaining instructions in the I/O and pen sort 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 and pen sort 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.

ESC . L, Output Buffer Size When Empty

USE: Outputs the size (in bytes) of the logical I/O buffer. Use this instruction to determine when the I/O buffer is empty.

SYNTAX: ESC . L

RESPONSE: logical I/O buffer size [TERM]

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. ■

An HP-IB interface terminates the response with a carriage return and line feed; an RS-232-C interface with a carriage return, or with the character specified by the ESC . M instruction.

RELATED

INSTRUCTION: 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

RESPONSE: operating status [TERM]

REMARKS: When using 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 table on the following page.

Bit Position	Logic State	Decimal Value	Meaning
0	0	0	Not used.
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	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	Not used.
7	0	0	Not used.
8	0	0	EXPAND mode is off.
	1	256	EXPAND mode is on.
9-15	*	*	Not used.

*Undefined.

Possible outputs and their meanings are described in the table on the following page.

	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

*Add 256 when EXPAND switch is on.

RELATED

INSTRUCTION: 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 the rear-panel Eavesdrop switch is on and you are using an RS-232-C interface (refer to the User's Guide for more information).

SYNTAX: ESC . Q *n*:

Parameter	Format	Range	Default
<i>n</i>	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.
- **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.

NOTE: You can also enable or disable monitor mode with bits 2 and 3 of the ESC.@ instruction. This method is more difficult than using the ESC.Q instruction and is not recommended. ■

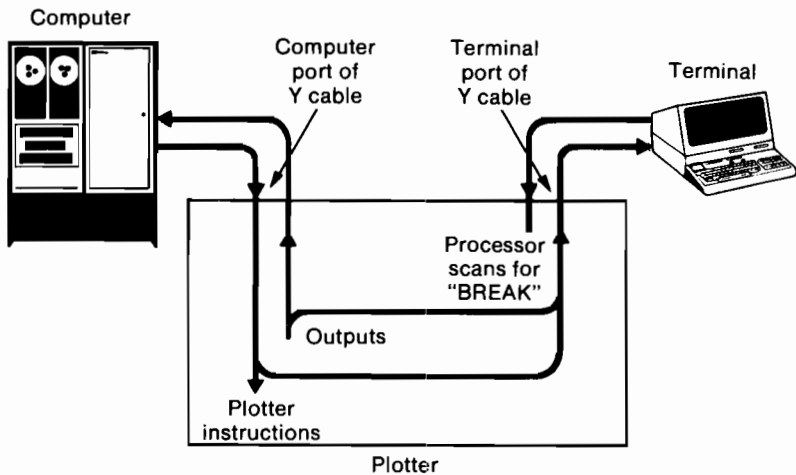
At power-on in Eavesdrop mode, the plotter is in the programmed-off state. Monitor mode is available only when the plotter is programmed-on; refer to ESC.Y or ESC.(later in this chapter. When the plotter is programmed-off, data is transferred between the computer and terminal in a transparent manner.

While in monitor mode 1 or 2, 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.

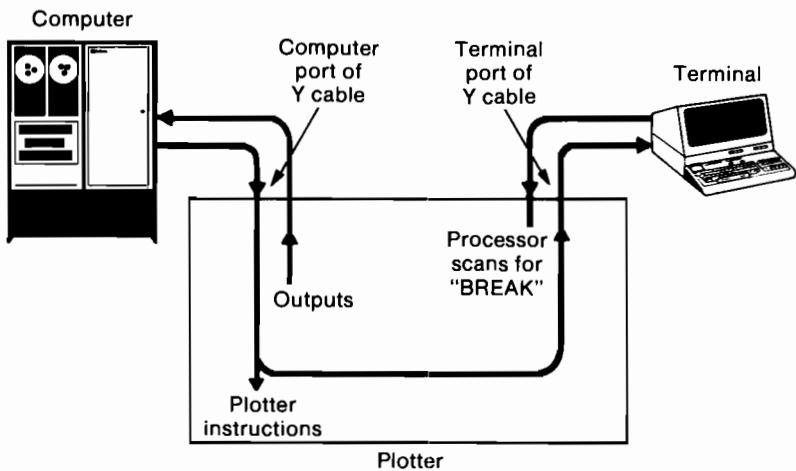
In monitor mode 2, with an echo terminator defined, the plotter output responses are sent to the computer, but not to the terminal. The computer is working in an echoplex environment, such that plotter responses to the computer are echoed to the terminal; if the plotter explicitly sent output responses to the terminal, the terminal would receive the responses twice (the explicit response and the echoed response). In monitor mode 2, with no echo terminator defined, the plotter output responses are sent to the terminal and the computer.

In monitor mode 1, only the parsed data from the plotter's buffer is transmitted to the terminal. If the computer echoes a plotter output response, this response never gets echoed to the terminal because it is not parsed by the plotter. The plotter must explicitly send any output response to the terminal.

The following illustrate data flow for the two monitor modes.



Parse Mode



Receive Mode

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 for ESC . J, ESC . K, and ESC . T elsewhere in this chapter. Refer to Chapter 14 for a detailed description of the ESC . P instruction.)

After executing 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 three buffers: the physical I/O buffer, pen sort buffer, or polygon 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
n	integer	0 to 6	0

RESPONSE: memory size [TERM]

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	Not used.
4	Not used.
5	Not used.
6	Pen sort buffer.

The plotter outputs the total amount of bytes in user-definable memory or the memory available in one of the three buffers, according to the parameter used.

RELATED

INSTRUCTION: ESC . T, Allocate Configurable Memory

ESC . T, Allocate Configurable Memory

USE: Allocates memory in user-definable RAM, which consists of three buffers: the physical I/O buffer, polygon buffer, and pen sort buffer. Use this instruction to change the sizes of all three buffers as needed.

SYNTAX: ESC . T (*physical I/O buffer size*); (*Polygon buffer size*); 0; 0; 0; (*Pen sort buffer size*):

Parameter	Format	Range	Default
physical I/O buffer size	integer	2 to 7436 bytes	1024 bytes
Polygon buffer size	integer	0 to 7434 bytes	1024 bytes
not used			0 bytes
not used			0 bytes
not used			0 bytes
Pen sort buffer size	integer	12 to 7446 bytes	5400 bytes

REMARKS: You can divide the plotter's configurable memory between the I/O, polygon, and pen sort 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 polygon and pen sort buffers.

When the ESC.T is executed, it performs the following operations (in sequence).

- Clears the I/O buffer, polygon, 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 size** — 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, HP-IB 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 size** — refer to *Using the Polygon Buffer* in Chapter 6 to determine an appropriate buffer size for your polygon(s).
- The third, fourth, and fifth parameters are reserved for buffers not implemented by this plotter and are always set to zero. You must always include these parameters either by specifying zeroes or by using semicolons only to indicate default values.
- **pen sort buffer size** — allows the plotter to sort the vectors according to the pen required for drawing. In turn, 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

ERROR:

Condition	Error	Plotter Response
sum of parameters exceeds 7448 bytes	13	sets all buffers to their default sizes

ESC.Y or ESC.(, Plotter On

USE: Enables the plotter to accept data and interpret it as HP-GL 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: In Eavesdrop (the rear-panel Eavesdrop switch is on), 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 computer 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.) or ESC.Z

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 hardware 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 7436 bytes*	1024 bytes
I/O conditions	integer	0 to 31*	3

*This is the practical range. The actual range the plotter accepts is 0 to 32767. We recommend that you restrict your values to the practical range specified.

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. Since the logical I/O buffer is the limiting factor, you should always increase it when you enlarge the physical I/O buffer.

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 hardware 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.

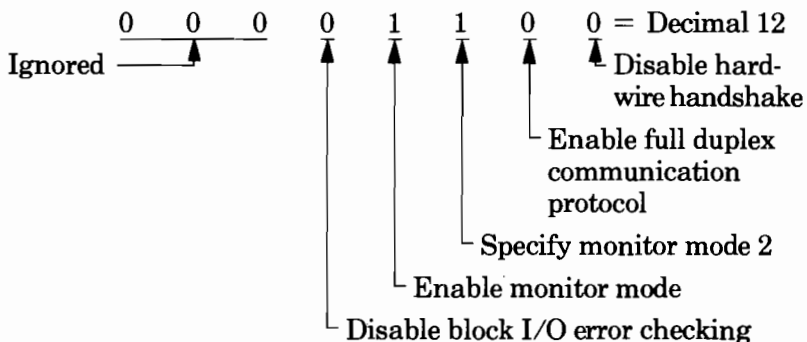
At power-on, bits 0 and 1 are set to 1; bits 2 through 4 are set to zero. When hardware handshake is enabled, the plotter uses the Data Terminal Ready (DTR) line in the same manner that buffer threshold indicators are used in the Xon-Xoff handshake. DTR is set high at power-on and is not set low until 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 is half empty or until there are 40 bytes more than 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 error 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$(27)+".@5000;12:"
```



CHAPTER



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 and 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.

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	CO
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 11 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. Setting the plotter address is detailed in the User's Guide.

When using the HP-IB, the plotter responds to 11 device-control instructions. The following are the valid device-control instructions 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 13. The plotter recognizes but ignores eight additional device-control

instructions. Of these eight device-control instructions, five 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	>	Δ
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 send 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 clears the I/O and pen sort buffers and resets the parser to begin accepting a new instruction, and disables any current output. (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.

About 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 the rear-panel Parity and Odd switches 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 (using the rear-panel switches) to match the baud rate of the computer; otherwise, the plotter will not be able to understand the data.
- **Number of stop bits** — On the plotter, baud rates 75 and 110 use two stop bits while baud rates of 200 and faster use one stop bit. (You can select one or two stop bits with a baud rate of 150.)

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 set a handshake method. The plotter can implement any of the following four handshakes.

- hardwire (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 Hewlett-Packard 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 default, or power-on, handshake. 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). You cannot use a hardwire handshake in any other configuration, but you can use other handshakes in this configuration. You cannot use the hardwire handshake in an eavesdrop configuration.

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. An Xon-Xoff handshake transmits a control character from the plotter 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 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 enquire/acknowledge 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 cannot 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.

Following is a list of device-control instructions that directly affect your handshake and handshake capabilities. Use these instructions in combination with the device-control instructions in Chapter 13 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

NOTE: These device-control instructions are for RS-232-C users. The HP-IB takes care of the interfacing and handshaking conditions automatically. Refer to Chapter 13 for more information concerning the device-control instructions that apply directly to HP-IB. ■

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 default handshake when you turn on your plotter; you don't have to use device-control instructions in your program to initiate it. Most personal computers can implement a hardwire handshake; refer to your computer's documentation.

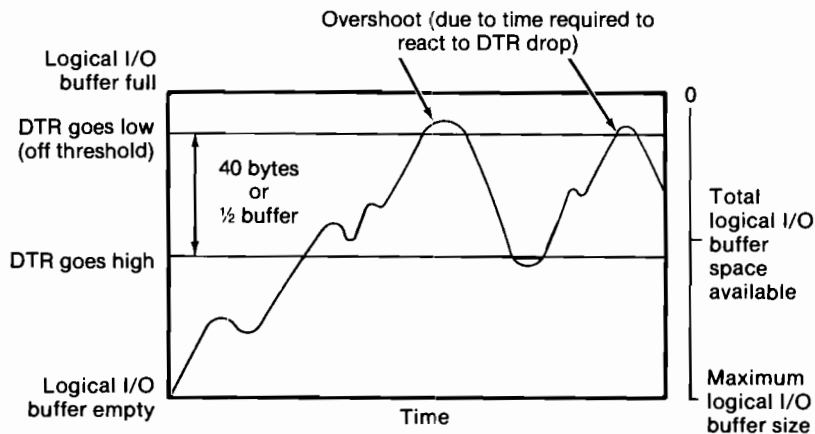
In a hardwire handshake, the computer monitors one of the interface lines from the plotter. The hardwire handshake works as follows.

1. 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 plotter raises the DTR line when either the I/O buffer has 40 bytes more than the threshold level available, or half the buffer is empty, whichever is smaller.

2. 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 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 overflowing the plotter's I/O buffer.



Use either of the following device-control instructions to initiate a hardwire 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 . @. 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.

```
CHR$(27)+" .@;3:"
CHR$(27)+" .I10:"
```

The following specifies the parameter values indicated by the above example.

ESC . @

I/O Buffer size = 1024 bytes (default)
 Hardwire handshake = 3-wire data communication
 (enabled)

ESC . I

“Off” Threshold level = 10 bytes

Xon-Xoff Handshake

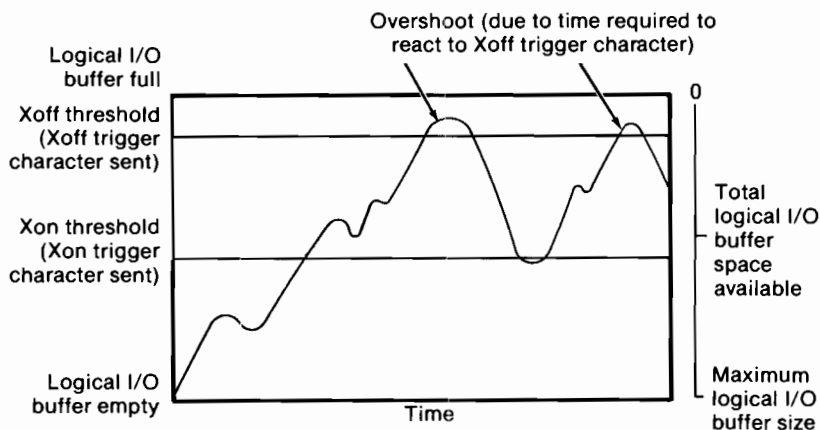
Refer to your computer system’s documentation to determine whether or not you can use the Xon-Xoff handshake.

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, 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 (approximately half the I/O buffer), 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 an Xon-Xoff handshake in your program.

- **ESC . P**
- **ESC . I** and **ESC . N**

Using ESC . P to Initiate an 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 terminator, 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 to Initiate an 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 specified by the above example are as follows.

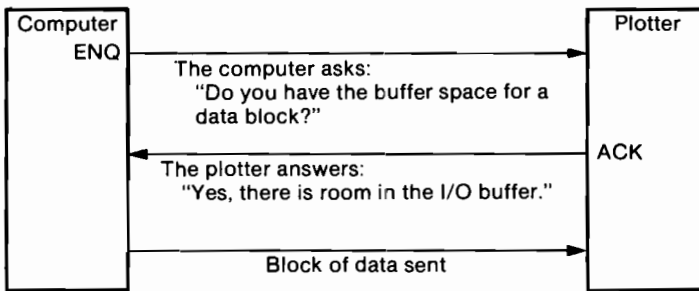
ESC . I
 Xoff threshold level = 10 bytes
 Enquiry character = omitted (indicates Xon-Xoff)
 Xon trigger character = 17 (ASCII **DC1**)

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 an Enquire/Acknowledge Handshake

Use ESC . P with a parameter of 2 (**ESC . P2**;) to initiate the handshake that sets several 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 terminator, and output terminator.

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 an Enquire/Acknowledge Handshake

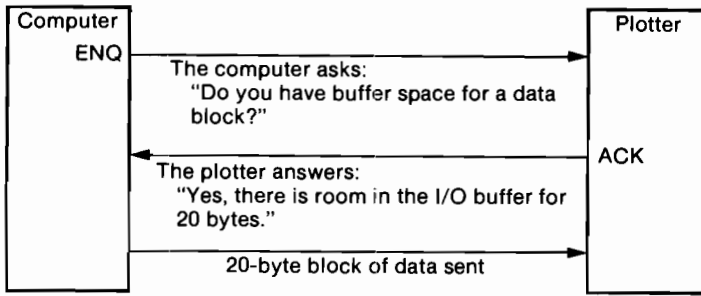
The ESC . I instruction also lets you establish an enquire/acknowledge handshake. If your computer does not *require* that you use the output trigger character, echo terminator, or output initiator with the handshake characters, use ESC . I to initiate the enquire/acknowledge handshake.

```
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 inter-character delay and immediate response string.

Using `ESC.H` to Initiate an 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 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. 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 terminator = none
- Output terminator = 13 (ASCII **CR**)

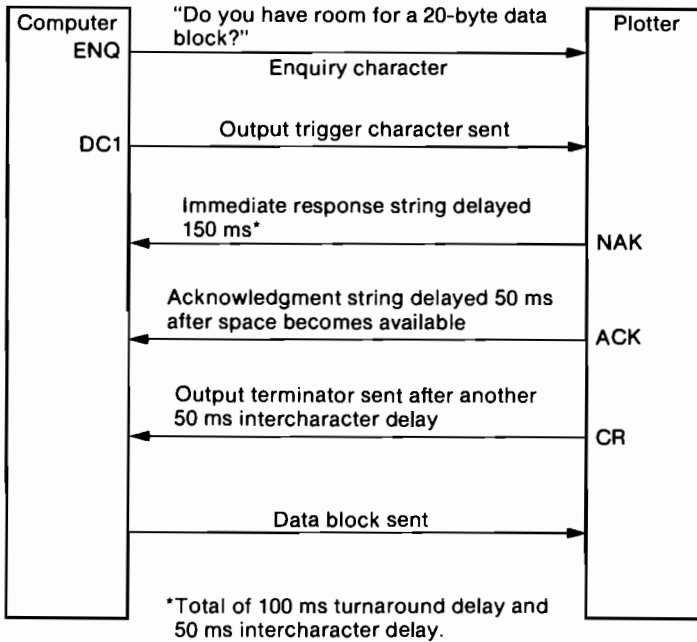
ESC.N

- 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 Enquire/Acknowledge 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.

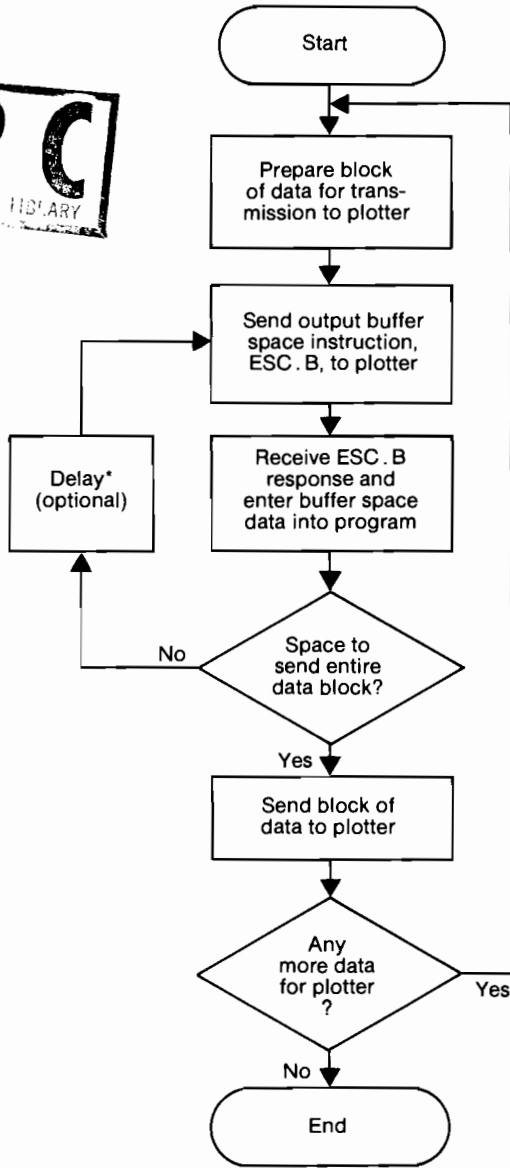
*Enquire/Acknowledge Instruction and
Parameters Used in Handshake Response*

ESC . H block size enquiry character acknowledgment string	ESC . I block size enquiry character acknowledgment string
ESC . M turnaround delay output trigger character echo 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. Add a delay after the ESC . B responses that are insufficient for the data block size 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 terminator 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 13 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. (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 13 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 when 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 13 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: →		Enable block I/O checking.
Data block A →		Send a block of data. [Assume a character gets garbled (e.g., bad parity).]
ESC.E →		Any I/O errors?
	← 15 [TERM]	Parity error. At this point, the plotter discards the block because an error occurred.

(Table continues)

Block I/O Error Checking (Continued)

Computer	Plotter	Comments
Data block A →		Retransmit the block.
ESC . E →		Any I/O errors?
	← 0 [TERM]	No errors. Plotter executes block.
Data block B →		Send a block of data. [Assume a handshake character gets lost, and buffer overflows.]
ESC . E →		Any I/O errors?
	← 16 [TERM]	Buffer overflow. Plotter discards block because an error occurred.
Data block B →		Retransmit the block.
ESC . E →		Any I/O errors?
	← 0 [TERM]	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. Hardware handshake cannot be used when either disconnect mode is active. To leave either mode, turn the plotter off and on again.

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. Activate this mode by pressing the front panel **PEN SELECT** button 5 during power-up until the plotter has completed its initialization cycle (all front-panel lights are no longer lit).

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. Activate this mode by pressing the front-panel **PEN SELECT** button 6 during power-up until the plotter has completed its initialization cycle (all front-panel lights are no longer lit).

NOTE: If you are using an eavesdrop cable and you set up a switched-line monitoring mode or a leased-lines monitoring mode, the plotter will not be able to monitor the other signal lines and you will not be able to output data. Also, if either switched-lines monitoring or leased-lines monitoring are operational, you cannot use a hardwire handshake. ■

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.

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.

SYNTAX: **ESC . H** (*data block size*); (*enquiry character*); (*acknowledgment string*): or **ESC . H**:

Parameter	Format	Range	Default
data block size	integer	0 to 7436 bytes*	80 bytes
enquiry character	ASCII value	0 to 26, 28 to 126†	0 (no character)
acknowledgment string	ASCII value	0 to 126	0 (no character)

*This is the practical range. The actual range the plotter will accept is 0 to 32 767.

†Printable characters (ASCII codes 32 to 126) should be avoided as they are required to send the HP-GL instructions. The recommended characters are those with codes 1-26 and 28-31.

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. ASCII 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. ASCII 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.

The parameters for this instruction are dependent on the handshake being configured.

SYNTAX

for Xon-Xoff: **ESC . I** (*Xoff threshold level*); (*omitted parameter*); (*Xon trigger character(s)*):

Parameter	Format	Range	Default
Xoff threshold level	integer	*	80 bytes
omitted parameter			
Xon trigger character(s)	ASCII value	decimal codes 0 to 126†	0 (no character)

*The practical range is one byte less than the logical I/O buffer size. The actual range the plotter accepts is 0 to 32 767; however, any value greater than the logical buffer size is changed to one byte less than the logical I/O buffer size.

†You can enter up to 10 character codes for the Xon trigger characters.

for Enquire/Acknowledge: **ESC . I** (*data block size*); (*enquiry character*); (*acknowledgment string*):

Parameter	Format	Range	Default
data block size	integer	0 to 7436 bytes*	80 bytes
enquiry character	ASCII value	0 to 26, 28 to 126†	0 (no character)
acknowledgment string	ASCII value	decimal codes 0 to 126††	0 (no character)

*This is the practical range. The actual range the plotter accepts is 0 to 32 767, however, any value greater than the practical range is changed to one byte less than the logical I/O buffer size.

†Printable characters (ASCII codes 32 to 126) should be avoided as they are required to send the HP-GL instructions. The recommended characters are those with codes 1-26 and 28-31.

††You can enter up to 10 ASCII character codes for the acknowledgment string.

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 than half the logical I/O buffer 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.

NOTE: When the Xoff threshold level is less than half the logical I/O buffer size, the Xon threshold level is set to half the logical I/O buffer size. ■

- **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 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. ASCII 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 separated from the previous one by a semicolon. Zero is not transmitted and terminates the string. ASCII 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). This instruction is valid when using an RS-232-C interface only.

SYNTAX: `ESC . M (turnaround delay);(output trigger character);(echo terminator);(output terminator);(output initiator character);`

Parameter	Format	Range	Default
turnaround delay	integer	0 to 32767 milliseconds	0
output trigger character	ASCII value	character codes 0 to 4, 6 to 26, and 28 to 126	0 (no character)
echo terminator	ASCII value	character codes 0 to 4, 6 to 26, and 28 to 126	0 (no character)
output terminator	ASCII value	one or two character codes 0 to 4, 6 to 26, and 28 to 126	13 (carriage return)
output initiator	ASCII value	character codes 0 to 126	0 (no character)

REMARKS: The plotter interprets the parameters as follows.

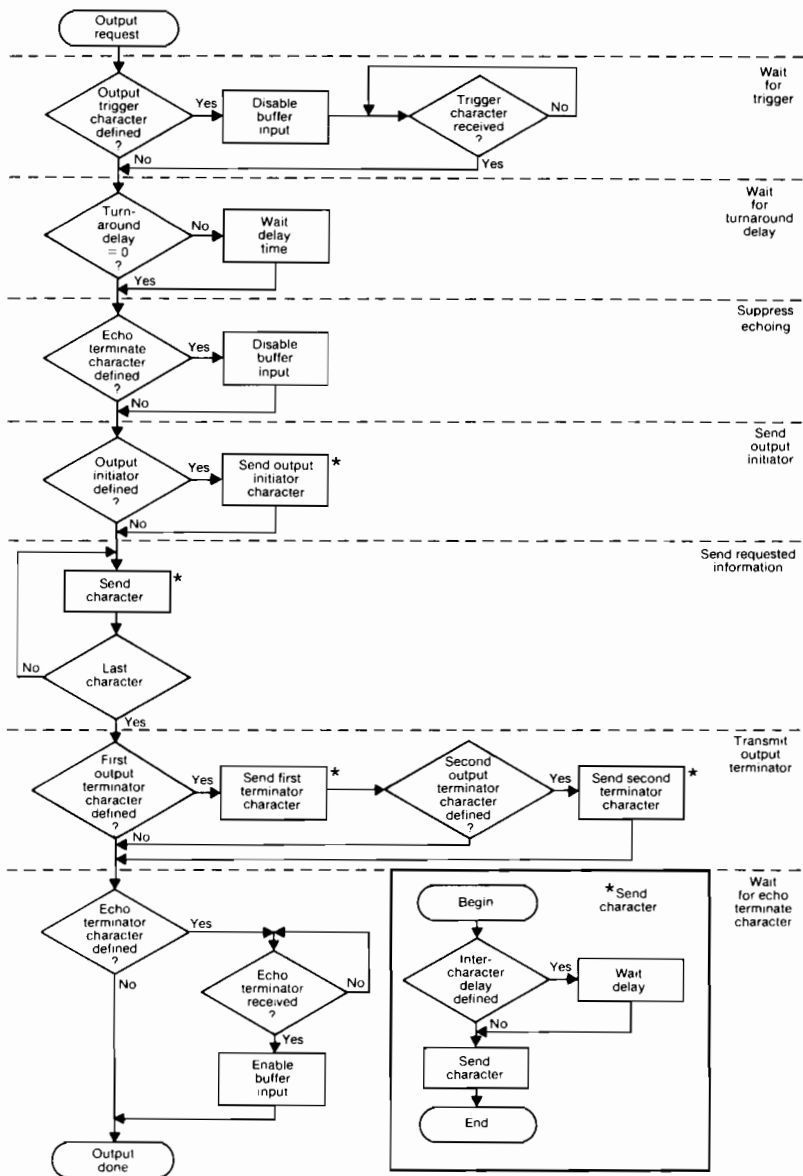
- **turnaround delay** — prevents the plotter from sending data until the computer is ready to receive and process it. (An inter-character delay set by ESC . N increments the turnaround delay.)
- **output trigger character** — is the last character output by the computer when making a request of the plotter. This signals the plotter to respond to the request. Most computers use the **DC1** character (decimal code 17) for 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 terminator. 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).
- **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). Many computers use **STX** (ASCII decimal code 2) 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 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 32 767 milliseconds	0
handshake dependent parameter For Xon-Xoff: Xoff trigger character(s)	ASCII value	characters 0 to 126*	0 (no character)
For Enquire/Acknowledge: immediate response string	ASCII value	characters 0 to 126*	0 (no response)

*You can enter up to 10 ASCII codes for either the Xoff trigger characters or the Enquire/Acknowledge immediate response string.

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: Simplifies specification of the handshaking method the plotter uses when talking to the computer. You can use this instruction to select among four standard handshakes.

SYNTAX: **ESC . P n:** or **ESC . P:**

Parameter	Format	Range	Default
n	integer	0, 1, 2, or 3	0 (none)

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				Predefined Handshake Parameters Established
0 None	1 Xon- Xoff*	2 Enq/ Ack*	3 Hardwire (default)	
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	Enquiry 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

ERROR:

Condition	Error	Plotter Response
parameter greater than 3	13	ignores the instruction

Notes



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 13). 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 0-9, 30-39, 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.

- AF, Advance Full Page
- AH, Advance Half Page
- AS, Select Pen Acceleration
- EC, Enable Cutter
- FR, Advance Frame
- FS, Select Pen Force
- GP, Designate Pen Group
- PG, Advance Page
- VA, Adaptive Velocity
- VN, Normal Velocity

Notes



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 Standard 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 continues)

Reaction to Nonprinting Control Characters (Continued)

Code	Decimal Character	ASCII 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 22 character sets. All printing characters are fixed-spaced vector fonts. The following table lists these character sets.

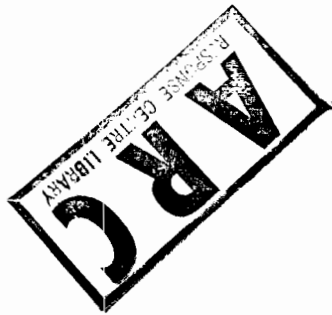
Fixed-Space Vector Font	Character Set	ISO Registration Number
0	ANSI ASCII	006
1	9825 Character Set	—
2	French/German	—
3	Scandinavian	—
4	Spanish/Latin American	—
5	Special Symbols	—
6	JIS ASCII	014
7	Roman Extensions	—
8	Katakana	013
9	ISO IRV (International Reference Version)	002
30	ISO Swedish	010
31	ISO Swedish for Names	011
32	ISO Norway, Version 1	060
33	ISO German	021
34	ISO French	025
35	ISO United Kingdom	004
36	ISO Italian	015
37	ISO Spanish	017
38	ISO Portuguese	016
39	ISO Norway, Version 2	061
99	Drafting	—
100*	Kanji	—
101*	Kanji	—

*Refer to Appendix D, *Using Kanji*, for detailed information concerning the Kanji character set.

Character Sets

The following pages show the plotter's character sets 0-9, 30-39, and 99. (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 (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. ■



Decimal Value	SET																				
	0	1	2	3	4	5	6	7	8	9	30	31	32	33	34	35	36	37	38	39	99
33	!	!	!	!	!	!	!	À	.	!	!	!	!	!	!	!	!	!	!	!	!
34	"	"	"	"	"	"	"	Â	["	"	"	"	"	"	"	"	"	"	"	"
35	#	#	£	£	£	#	#	È]	#	#	#	#	#	£	£	£	£	#	\$	¢
36	\$	\$	\$	\$	\$	\$	\$	Ê	.	¤	¤	¤	¤	¤	\$	\$	\$	\$	\$	\$	\$
37	%	%	%	%	%	%	%	Ë	.	%	%	%	%	%	%	%	%	%	%	%	%
38	&	&	&	&	&	&	&	Ï	ƒ	&	&	&	&	&	&	&	&	&	&	&	&
39	Ï	ƒ
40	(((((((Ï	ƒ	((((((((((((
41)))))))	Ï	ƒ))))))))))))
42	*	*	*	*	*	*	*	Ï	ƒ	*	*	*	*	*	*	*	*	*	*	*	*
43	+	+	+	+	+	+	+	Ï	ƒ	+	+	+	+	+	+	+	+	+	+	+	+
44	Ï	ƒ
45	-	-	-	-	-	-	-	Ï	ƒ	-	-	-	-	-	-	-	-	-	-	-	-
46	Ï	ƒ
47	/	/	/	/	/	/	/	Ï	ƒ	/	/	/	/	/	/	/	/	/	/	/	/
48	0	0	0	0	0	0	0	Ï	ƒ	0	0	0	0	0	0	0	0	0	0	0	0
49	1	1	1	1	1	1	1	Ï	ƒ	1	1	1	1	1	1	1	1	1	1	1	1
50	2	2	2	2	2	2	2	Ï	ƒ	2	2	2	2	2	2	2	2	2	2	2	2
51	3	3	3	3	3	3	3	Ï	ƒ	3	3	3	3	3	3	3	3	3	3	3	3
52	4	4	4	4	4	4	4	Ï	ƒ	4	4	4	4	4	4	4	4	4	4	4	4
53	5	5	5	5	5	5	5	Ï	ƒ	5	5	5	5	5	5	5	5	5	5	5	5
54	6	6	6	6	6	6	6	Ï	ƒ	6	6	6	6	6	6	6	6	6	6	6	6
55	7	7	7	7	7	7	7	Ï	ƒ	7	7	7	7	7	7	7	7	7	7	7	7
56	8	8	8	8	8	8	8	Ï	ƒ	8	8	8	8	8	8	8	8	8	8	8	8
57	9	9	9	9	9	9	9	Ï	ƒ	9	9	9	9	9	9	9	9	9	9	9	9
58	:	:	:	:	:	:	:	Ï	ƒ	:	:	:	:	:	:	:	:	:	:	:	:
59	:	:	:	:	:	:	:	Ï	ƒ	:	:	:	:	:	:	:	:	:	:	:	:
60	<	<	<	<	<	<	<	Ï	ƒ	<	<	<	<	<	<	<	<	<	<	<	<
61	=	=	=	=	=	=	=	Ï	ƒ	=	=	=	=	=	=	=	=	=	=	=	=
62	>	>	>	>	>	>	>	Ï	ƒ	>	>	>	>	>	>	>	>	>	>	>	>
63	?	?	?	?	?	?	?	Ï	ƒ	?	?	?	?	?	?	?	?	?	?	?	?
64	@	@	@	@	@	@	@	Ï	ƒ	@	@	@	@	@	@	@	@	@	@	@	@

Decimal Value	SET																																			
	0	1	2	3	4	5	6	7	8	9	30	31	32	33	34	35	36	37	38	39	99															
65	A	A	A	A	A	□	A	ê	チ	A	A	A	A	A	A	A	A	A	A	A	A	A														
66	B	B	B	B	B	○	B	ô	ツ	B	B	B	B	B	B	B	B	B	B	B	B	B														
67	C	C	C	C	C	△	C	û	テ	C	C	C	C	C	C	C	C	C	C	C	C	C														
68	D	D	D	D	D	+	D	á	ト	D	D	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	E	E														
70	F	F	F	F	F	◊	F	ó	ニ	F	F	F	F	F	F	F	F	F	F	F	F	F														
71	G	G	G	G	G	⊕	G	ú	又	G	G	G	G	G	G	G	G	G	G	G	G	G														
72	H	H	H	H	H	X	H	à	ヲ	H	H	H	H	H	H	H	H	H	H	H	H	H														
73	I	I	I	I	I	Z	I	è	ノ	I	I	I	I	I	I	I	I	I	I	I	I	I														
74	J	J	J	J	J	γ	J	ò	ハ	J	J	J	J	J	J	J	J	J	J	J	J	J														
75	K	K	K	K	K	X	K	ù	ヒ	K	K	K	K	K	K	K	K	K	K	K	K	K														
76	L	L	L	L	L	*	L	ã	フ	L	L	L	L	L	L	L	L	L	L	L	L	L														
77	M	M	M	M	M	X	M	ē	ハ	M	M	M	M	M	M	M	M	M	M	M	M	M														
78	N	N	N	N	N	i	N	ö	キ	N	N	N	N	N	N	N	N	N	N	N	N	N														
79	O	O	O	O	O	•	O	ü	ワ	O	O	O	O	O	O	O	O	O	O	O	O	O														
80	P	P	P	P	P	-	P	Â	ミ	P	P	P	P	P	P	P	P	P	P	P	P	P														
81	Q	Q	Q	Q	Q	i	Q	î	シ	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q														
82	R	R	R	R	R	R	R	ø	リ	R	R	R	R	R	R	R	R	R	R	R	R	R														
83	S	S	S	S	S	S	S	€	セ	S	S	S	S	S	S	S	S	S	S	S	S	S														
84	T	T	T	T	T	T	T	â	テ	T	T	T	T	T	T	T	T	T	T	T	T	T														
85	U	U	U	U	U	U	U	í	リ	U	U	U	U	U	U	U	U	U	U	U	U	U														
86	V	V	V	V	V	V	V	ø	シ	V	V	V	V	V	V	V	V	V	V	V	V	V														
87	W	W	W	W	W	W	W	æ	シ	W	W	W	W	W	W	W	W	W	W	W	W	W														
88	X	X	X	X	X	X	X	Ä	リ	X	X	X	X	X	X	X	X	X	X	X	X	X														
89	Y	Y	Y	Y	Y	Y	Y	ì	ル	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y														
90	Z	Z	Z	Z	Z	Z	Z	Ö	ル	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z														
91	[[[[[[[Ü	ロ	[Ä	Ä	€	Ä	•	[•	i	Ä	€	[[
92	\	\	\	\	\	\	\	É	ワ	\	Ö	Ö	ø	Ö	ç	\	ç	Ñ	Ç	ø	ø	ø														
93]]]]]]]	i	ワ]	Ä	Ä	Ä	Ü	§]	é	ç	Ö	Ä]]														
94	^	↑	^	^	^	^	^	β	^	^	^	^	^	^	^	^	^	^	^	^	^	^														
95	-	-	-	-	-	-	-	Ô	-	-	-	-	-	-	-	-	-	-	-	-	-	-														
96	-	-	-	-	-	-	-	Á	-	-	-	-	-	-	-	-	-	-	-	-	-	-														

Decimal Value	SET																				
	0	1	2	3	4	5	6	7	8	9	30	31	32	33	34	35	36	37	38	39	99
97	a	a	a	a	a	n	a	Ã		a	a	a	a	a	a	a	a	a	a	a	a
98	b	b	b	b	b	o	b	ã		b	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	c
100	d	d	d	d	d	u	d	d		d	d	d	d	d	d	d	d	d	d	d	d
101	e	e	e	e	e	e	í	í		e	e	e	e	e	e	e	e	e	e	e	e
102	f	f	f	f	f	f	ì	ì		f	f	f	f	f	f	f	f	f	f	f	f
103	g	g	g	g	g	g	ó	ó		g	g	g	g	g	g	g	g	g	g	g	g
104	h	h	h	h	h	h	ò	ò		h	h	h	h	h	h	h	h	h	h	h	h
105	i	i	i	i	i	i	ï	ï		i	i	i	i	i	i	i	i	i	i	i	i
106	j	j	j	j	j	j	õ	õ		j	j	j	j	j	j	j	j	j	j	j	j
107	k	k	k	k	k	k	š	š		k	k	k	k	k	k	k	k	k	k	k	k
108	l	l	l	l	l	l	š	š		l	l	l	l	l	l	l	l	l	l	l	l
109	m	m	m	m	m	m	ú	ú		m	m	m	m	m	m	m	m	m	m	m	m
110	n	n	n	n	n	n	ÿ	ÿ		n	n	n	n	n	n	n	n	n	n	n	n
111	o	o	o	o	o	o	ÿ	ÿ		o	o	o	o	o	o	o	o	o	o	o	o
112	p	p	p	p	p	p	þ	þ		p	p	p	p	p	p	p	p	p	p	p	p
113	q	q	q	q	q	q	þ	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	r	r
115	s	s	s	s	s	s	š	š		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	t	t
117	u	u	u	u	u	u	ú	ú		u	u	u	u	u	u	u	u	u	u	u	u
118	v	v	v	v	v	v	v	v		v	v	v	v	v	v	v	v	v	v	v	v
119	w	w	w	w	w	w	w	w		w	w	w	w	w	w	w	w	w	w	w	w
120	x	x	x	x	x	x	x	x		x	x	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	y	y	y	y
122	z	z	z	z	z	z	z	z		z	z	z	z	z	z	z	z	z	z	z	z
123	{	π	{	{	«	{	ä	ä	æ	ä	é	{	à	.	ã	æ	μ
124		†			o		ö	ö	ø	ö	ù		ò	ñ	ç	ø	°
125	}	-	}	}	»	}	á	á	â	ü	è	}	è	ç	õ	ã	°
126	~	~	~	-	±	-	-	ü	-	B	-	i	~	.		~	~

APPENDIX



Sample Plots and Program Listings

The sample programs in this appendix are designed to give you a better understanding of some of the fundamental HP-GL concepts, such as scaling, rotation, using windows, and digitizing. Four of the five programs are written in Microsoft BASIC; the fifth program is written in HP Series 200 BASIC (Version 3.0).

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.

With the exception of the HP Series 200 sample program, the plot for each sample program is shown after the listing it represents. Each plot fits on C-size paper. The HP Series 200 sample program concerns digitizing, the size of the drawing you digitize determines the paper size you should use when you draw it.

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 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,-7000,10000,7000;"
35  *
45  *draws box around the quadrant
50  PRINT #1, "SP1;PU-10000,7000;"
60  PRINT #1, "PD-100,7000,-100,100,-10000,100,
    -10000,7000;"
65  *
75  *draws figures
80  PRINT #1, "FU-5000,3500;"
90  GOSUB 600
105 *
115 *numbers quadrant
120 PRINT #1, "SR1.2,2.4;PU-3000,4000;LB# 1" +
    CHR$(3)

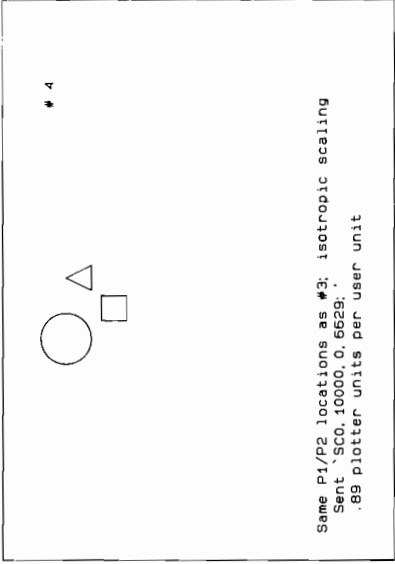
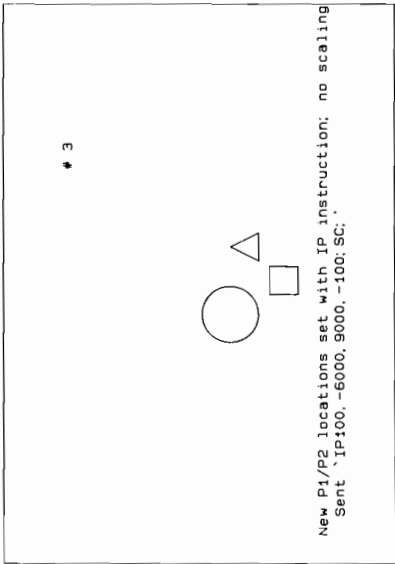
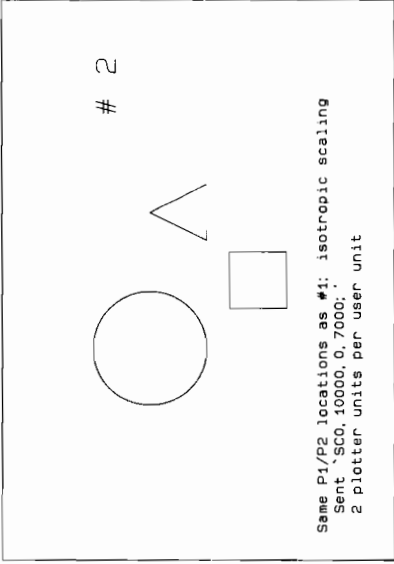
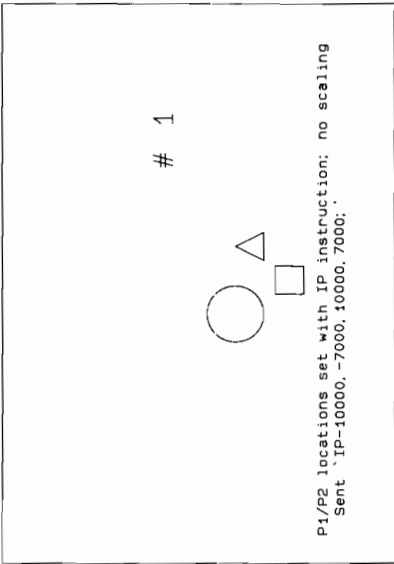
```

```

130 PRINT #1, "SI;"
135 '
145 'labels quadrant
150 PRINT #1, "PU-10000,100;"
160 PRINT #1, "CP3,4;PR0,0;PA;"
170 PRINT #1, "LBP1/P2 locations set with IP
      instruction; no scaling"+CHR$(13)
      +CHR$(10)+CHR$(3)
180 PRINT #1, "LB Sent 'IP-10000,-7000,
      10000,7000;'" +CHR$(3)
185 '
195 ' Section 2 - Same P1/P2 locations with
      isotropic scaling
200 PRINT #1, "SC0,10000,0,7000;"
205 '
210 PRINT #1, "SP2;PU5050,3550;"
220 PRINT #1, "PD5050,7000,10000,7000,10000,3550,
      5050,3550;"
225 '
230 PRINT #1, "PU9000,6000;SR1.2,2.4;LB# 2"+
      CHR$(3)
235 '
240 PRINT #1, "PA7525,6300;"
250 GOSUB 600
255 '
260 PRINT #1, "PU5050,3550;SI;CP3,4;PR0,0;PA;"
270 PRINT #1, "LBSame P1/P2 locations as #1;
      isotropic scaling"+CHR$(13)+
      CHR$(10)+CHR$(3)
280 PRINT #1, "LB Sent 'SC0,10000,0,7000;'" +
      CHR$(13)+CHR$(10)+CHR$(3)
290 PRINT #1, "LB 2 plotter units per user
      unit"+CHR$(3)
295 '
305 'Section 3 - Turn off scaling, move P1/P2
310 PRINT #1, "SR1.2,2.4;IP100,-6000,9000,-100;
      SC;"
320 PRINT #1, "SP3;PU-10000,-7000;"
330 PRINT #1, "PD-100,-7000,-100,-100,
      -10000,-100,-10000,-7000;"
335 '
340 PRINT #1, "PU-5000,-3500;"

```

```
350 GOSUB 600
355 '
360 PRINT #1, "PU-3000,-1300;LB# 3"+CHR$(3)
370 PRINT #1, "PU-10000,-7000;SI;CP3,4;PR0,0;PA;"
380 PRINT #1, "LBNew P1/P2 locations set with IP
      instruction; no scaling"+CHR$(13)+
      CHR$(10)+CHR$(3)
390 PRINT #1, "LB Sent 'IP100,-6000,9000,-100;
      SC;'"+CHR$(3)
395 '
400 'Section 4 - New P1/P2, isotropic scaling
410 PRINT #1, "IP;SR1.2,2.4;IP100,-6000,9000,-100;
      SC0,10000,0,6629;"
420 PRINT #1, "PU0,-1123;SP4;"
430 PRINT #1, "PD11123,-1123,11123,6629,0,6629,
      0,-1123;"
435 '
440 PRINT #1, "PU5000,5979;"
450 GOSUB 600
455 '
460 PRINT #1, "PU9000,5629;"
470 PRINT #1, "LB# 4"+CHR$(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;LBOoriginal
      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$(3)
300 GOSUB 400
310 '
320 'return P1 to original values; label scaling
340 PRINT #1, "IP-4200,-6900;"
350 PRINT #1, "SC;PU0,9000;LO5;SI.45,.65;CP0,-5;"
360 PRINT #1, "LBDrawn with scale
      SC0,5000,0,8214"+CHR$(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;
      PD1400,250;PU;"
490 PRINT #1, "SI.15,.25;LO4;PU800,420;LB7570A"+
      CHR$(3)
500 'restore program defaults
510 PRINT #1, "SI;LO2;"
520 RETURN
```

Drawn with scale SC0, 5000, 0, 8214



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;LO11;LBP1"+
      CHR$(3)
60   PRINT #1, "PA-300,6650;CI20;LO19;LBP2"+CHR$(3)
70   PRINT #1, "PU-3950,6650;CP0,-3;LO1;"
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;LO11;LBP1"+CHR$(3)
240  PRINT #1, "PU3950,6650;CI20;LO19;LBP2"+CHR$(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

```

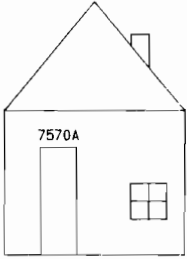
```

285  *
295  `outlines window area
300  PRINT #1, "LT-2;PU0,6978;PD3830,6978,
      3830,13956,0,13956,0,6978;PU;"
310  PRINT #1, "LT;IW;"
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
405  `draws two houses
415  `drawing the left house
420  PRINT #1, "SP2;PR;PU-1700,1700;ER2000,-1600;"
430  PRINT #1, "PD1000,1200,1000,-1200;"
440  PRINT #1, "PU-400,480;PD0,360,-200,0,0,-120;"
450  PRINT #1, "PU-1000,-2320;PD0,1200,400,0,
      0,-1200;"
460  PRINT #1, "PU600,800;ER400,-400;"
470  PRINT #1, "PU200,-400;PD0,400;PU-200,-200;
      PD400,0;PU;"
480  `label address and return to carriage return point
490  PRINT #1, "SI.15,.25;L04;PU-1200,680;LB7570A"+
      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

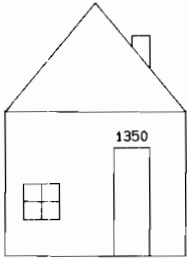
```

P2

No window; sent
`SC0, 3830, 0, 13956;`



7570A

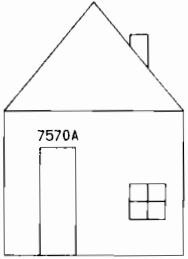


1350

P1

Detailed description: This diagram shows two house outlines. The top house is labeled '7570A' and has a chimney on the right side of its roof, a door on the left, and a window on the right. The bottom house is labeled '1350' and has a chimney on the right side of its roof, a window on the left, and a door on the right. The label 'P2' is in the top right corner and 'P1' is in the bottom left corner.

P2



7570A

↑
Window; sent
`IW0, 6978, 3830, 13956;`

P1

Detailed description: This diagram shows a single house outline labeled '7570A' with a chimney on the right side of its roof, a door on the left, and a window on the right. The house is enclosed within a dashed rectangular border. Below the dashed box, an upward-pointing arrow is followed by the text 'Window; sent' and a code string. The label 'P2' is in the top right corner and 'P1' is in the bottom left corner.

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*; to the plotter before drawing the pencil.)

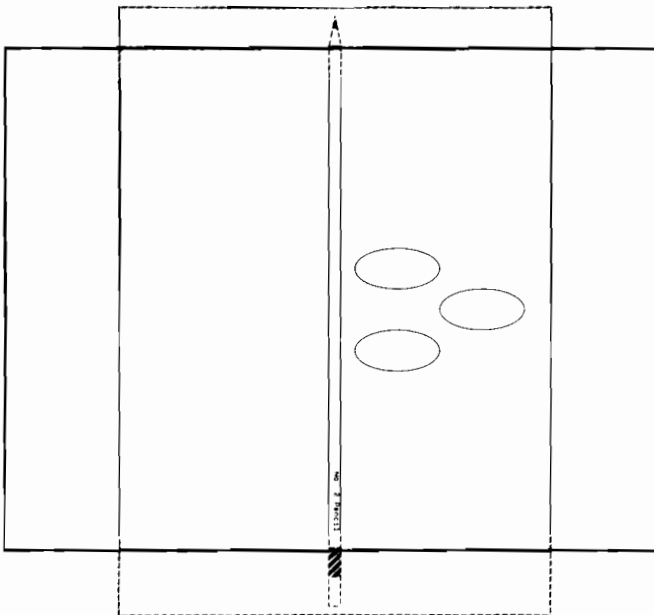
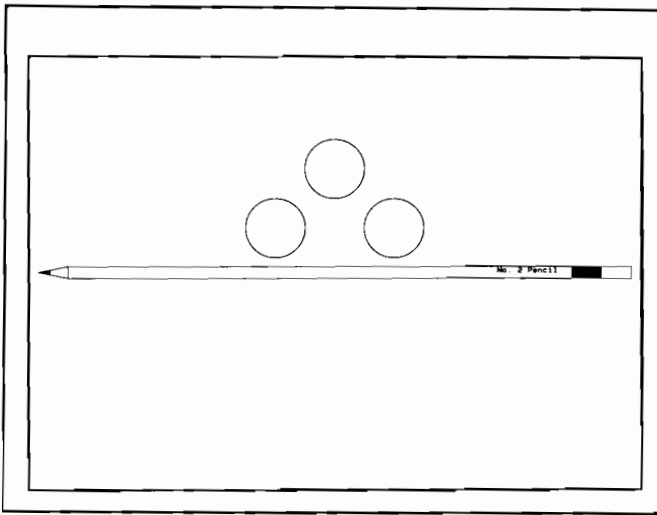
After drawing the pencil, the plotting area is scaled isotropically for unrotated P1 and P2 values. Since pressing the front-panel **ROTATE** button sets P1 and P2 within the hard-clip limits, a rotated plot with the same scaling is anisotropic. Anisotropic scaling will distort geometric shapes such 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;FP;"
120 PRINT #1, "PU-8800,200;PD-8800,-200;PU;"
130 PRINT #1, "PU7800,200;"
140 PRINT #1, "PM0;PD8800,200,8800,-200,7800,-200,
      7800,200;PU;PM2;"
150 PRINT #1, "FT3,40,0;FP;FT;"
160 PRINT #1, "SP2;PU5300,0;LBNo. 2 Pencil"+
      CHR$(3)

```

```
170  '
180  'draw a frame around plot
190  PRINT #1, "SP3;PU-10100,-7200;"
200  PRINT #1, "EA10100,7200;"
210  '
220  'isometric scale for unrotated P1/P2 values
230  PRINT #1, "SC0,9800,0,6900;SP4;"
240  '
250  'draw the circles
260  PRINT #1, "PU4900,5175;CI500;"
270  PRINT #1, "PU3900,4175;CI500;"
280  PRINT #1, "PU5900,4175;CI500;"
290  '
300  PRINT #1, "SP0;"
310  END
```



Example 5: Digitizing (HP Series 200 BASIC)

This program was written on an HP9000, series 200 computer using BASIC 3.0 and the standard HP-IB interface. It illustrates one way you can use DraftPro as a digitizer. This program lets you enter up to 250 digitized points.

The program uses the IM instruction so that the plotter generates a service request (SRQ) whenever a digitized point has been entered. All other status conditions are masked such that they do not generate a service request. The key loop for generating the SRQ and entering points is in lines 570-750.

When all points have been digitized, the program prompts you to load a clean sheet of paper and enter a pen number to draw a test plot of the digitized data. In a real application, the data would likely be stored in a file for plotting later.

Note that although this program is written expressly for digitizing, you can expand it to include other functions needing computer processing attention.

```

10  COM /Intr_block/ INTEGER I,X(1:250),Y(1:250),
      P(1:250),Slc,Ifcode
20  !
30  INTEGER Ifprior,Intr_mask,Ime,Ime,Imp
40  DIM A$(1),Clear_crt$(2),Esc_dot$(2)
50  !
60  ! Initialize variables
70  !
80  Ifprior=8 ! interface priority
90  Intrmask=2 ! I/F mask for service request only
100 Ifcode=7 ! interface code
110 Slc=705 ! select code for plotter
120 !
130 Ime=223 ! IM error mask (errors 1-5 and 7)
140 Ime=4 ! IM serial mask (digitize point avail.)
150 Imp=0 ! IM parallel mask (no parallel poll)
160 !
170 I=1 ! Data array index

```

```
180 !
190 Esc_dot$=CHR$(27)&“.” ! Set prefix for device-
200 ! control instructions
210 Clear_crt$=CHR$(255)&CHR$(75) ! Sent to KBD will
    clear screen
220 !
230 !
240 ABORT Ifcode ! Clear the I/F
250 !
260 OUTPUT KBD;Clear_crt$ END
270 !
280 ! Prompt user to load drawing to be digitized and
    to remove digitizing
290 ! sight if in the pen arm of the plotter
300 !
310 PRINT TABXY(4,5);“REMOVE the digitizing sight
    from the pen arm of the plot”
320 PRINT TABXY(4,7);“Load the drawing to be
    digitized in the plotter and press
    CONTINUE.”
330 PAUSE
340 !
350 OUTPUT KBD;Clear_crt$ END
360 !
370 ! Initialize the plotter using AP instruction to
380 ! disable automatic pen put away and pen pick up.
390 !
400 OUTPUT S1c;“R00;IN;SP0;AP0;” END
410 !
420 ! Instruct user on how to enter data points
430 !
440 PRINT TABXY(4,2);“PLACE DIGITIZING SIGHT IN
    PEN ARM”
450 PRINT TABXY(4,5);“Do the following for each point
    you want to digitize:”
460 PRINT TABXY(6,7);“1) Place the sight in the down
    position.”
470 PRINT TABXY(6,9);“2) Use the cursor buttons on
    the plotter to position the sight”
480 PRINT TABXY(9,10);“over the next point.”
```



```

490 PRINT TABXY(6,12);"3) Raise the sight if you want
      the pen to move here without drawing."
500 PRINT TABXY(6,14);"4) Press the plotter's Enter
      button to digitize the point."
510 PRINT TABXY(6,16);"5) Repeat until all points are
      digitized."
520 !
530 ! Send IM for SRQ on digitize point available
540 !
550 OUTPUT S1c USING "K,K,2( """, """,K), """: """,#";
      "IM";Ime;Ime;Imp
560 !
570 ! Enable interrupts on the computer
580 !
590 ON INTR Ifcode,Ifprior CALL Enter_pt
600 ENABLE INTR Ifcode;Intr_mask
610 !
620 ! Enter the digitize state and place computer
630 ! into wait state until all points entered.
640 !
650 ON KEY 4 LABEL "DONE",9 GOTO End_entry
660 OUTPUT S1c;"DP;" END
670 PRINT TABXY(4,18);"Press computer softkey DONE
      when all points have been entered."
680 !
690 Idle: ! WAIT for user to enter all points
700 GOTO Idle
710 !
720 End_entry: ! Points digitized
730 OFF KEY 4
740 !
750 DISABLE INTR Ifcode
760 OUTPUT S1c;"DC;" END ! Exit digitize mode
770 I=I-1 ! point to last point entered
780 !
790 !
800 OUTPUT KBD;Clear_crt$ END
810 !
820 ! Prompt user to remove the digitizing sight and
830 ! digitized drawing, then load paper and enter pen
840 ! number for drawing plot.
850 !

```

```

860 PRINT TABXY(4,3);"REMOVE the digitizing sight."
870 PRINT TABXY(4,5);"REMOVE the digitized drawing."
880 PRINT TABXY(4,7);"Load paper for a test plot."
890 INPUT "When you are ready to draw, ENTER the pen
      number to use.",Pen0
900 !
910 ! Get the pen and draw the plot
920 !
930 OUTPUT KBD;Clear_crt$ END
940 DISP "Making test plot"
950 OUTPUT Slc USING "K,K,"";";";"#";"SP";Pen0
960 CALL Draw_plot
970 !
980 DISP "Plot instructions sent. Remove the test
      plot when error reported."
990 OUTPUT Slc;"SP0;OE;" END ! Check for errors
1000 ENTER Slc;Err_hpgl
1010 !
1020 OUTPUT Slc;"IN;" END ! restore auto-pen features
1030 !
1040 DISP "HP-GL Error reported ";Err_hpgl
1050 END
1060 !
1070 !
2000 Enter_pt: !
2010 !
2020 SUB Enter_pt ! Interrupt routine to retrieve
      digitized points
2030 !
2040 COM /Intr_block/ INTEGER I,X(1:250),Y(1:250),
      P(1:250),Slc,Ifcode
2050 !
2060 INTEGER Maxi,Reply
2070 !
2080 Maxi=250 ! maximum array size
2090 !
2100 Reply=SPOLL(Slc)
2110 !
2120 DISP "Max. Points is ";Maxi;" Entering point ";I
2130 !
2140 IF NOT BIT(Reply,6) THEN
2150 ! was not the plotter that interrupted

```

```
2160     DISP "Plotter did not cause interrupt. Press
          CONTINUE to exit routine."
2170     PAUSE
2180     GOTO No_point
2190 ELSE
2200     IF NOT BIT(Reply,2) THEN
2210         ! improper serial mask
2220         DISP "Plotter sent SRQ for status byte ";
          Reply;" Press CONTINUE to exit routine."
2230         PAUSE
2240         GOTO No_point
2250     ELSE
2260         ! SRQ for digitize point
2270         OUTPUT Slc;"OD;"
2280         ENTER Slc;X(I),Y(I),P(I)
2290         I=I+1
2300     END IF
2310     !
2320 END IF
2330 !
2340 IF I>MaxI THEN
2350     ! array used up; tell user, exit digitize mode
2360     DISP "Data space is filled. CANNOT digitize any
          more points."
2370     GOTO No_point
2380 END IF
2390 !
2400 OUTPUT Slc;"DP;" END ! Set plotter for next point
2410 !
2420 No_point: ! unable to enter a point
2430 !
2440 ENABLE INTR Ifcode ! enable interrupts
2450 SUBEND ! Enter_pt
2460 !
2470 !
3000 Draw_plot: !
3010 !
3020 SUB Draw_plot ! plots the data with current pen
3030 !
3040 COM /Intr_block/ INTEGER I,X(1:250),Y(1:250),
          P(1:250),Slc,Ifcode
3050 !
```

```
3060  INTEGER J,Pstate
3070  !
3080  Pstate=P(1)
3090  IF Pstate=0 THEN
3100    OUTPUT S1c;"PU" END
3110  ELSE
3120    OUTPUT S1c;"PD" END
3130  END IF
3140  OUTPUT S1c USING "K,""", """,K,#";X(1),Y(1)
3150  !
3160  FOR J=2 TO I
3170    IF P(J)<>Pstate THEN
3180      ! Change pen status
3190      Pstate=P(J)
3200      IF Pstate=0 THEN
3210        OUTPUT S1c;" ;PU" END
3220      ELSE
3230        OUTPUT S1c;" ;PD" END
3240      END IF
3250      OUTPUT S1c USING "K,""", """,K,#";X(J),Y(J)
3260      !
3270    ELSE
3280      ! just send coordinates
3290      OUTPUT S1c USING "2(""", """,K),#";X(J);
          Y(J)
3300    END IF
3310  NEXT J
3320  !
3330  OUTPUT S1c;" ;PU;" END ! Pick up pen
3340  !
3350  SUBEND ! Draw_plot
```



Using Kanji

In the DraftPro plotter, the 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 DraftPro 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	Character 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

(Table continues)

First Byte Value	Character Group	Second Byte Value Range
39	Cyrillic alphabet	33-65, 81-113
48-78 79	Kanji	33-126 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.

```
“LB” ;CHR$(65);CHR$(42);CHR$(3)
```

or

```
“LBA*” ;CHR$(3)
```

On many keyboards, ASCII decimal value 65 corresponds to the capital letter A and decimal value 42 corresponds to the asterisk character (*). You can use either method to access Kanji characters.

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.

```

10  'insert configuration statement here
20  PRINT #1, "IN;SP1;PA0,0;"
30  PRINT #1, "CS0;CA101;SA;DV1;"
40  PRINT #1, "LBJ8;z"+CHR$(13)+CHR$(10)+
      CHR$(3)
50  PRINT #1, "LB";CHR$(37)+CHR$(59);CHR$(37)+
      CHR$(67);CHR$(37)+CHR$(72);CHR$(10)+
      CHR$(3)
60  PRINT #1, "SS;LB101"+CHR$(3)
70  PRINT #1, "SP0;"
80  END

```

```

      セ 文
      ツ 字
      ト

```

```

      1
      0
      1

```

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.

	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
33		、	。	，	．	・	：	；	？	！		
34	◆	□	▣	△	▲	▽	▼	※	〒	→	←	↑
35												
36	あ	お	い	う	え	お	か	が				
37	ア	ア	イ	ウ	エ	オ	カ	ガ				
38	A	B	Γ	Δ	E	Z	H	Θ	I	K	Λ	M
39	A	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К
48	亜	啞	娃	阿	哀	愛	挨	始	逢	葵	茜	穉
49	院	陰	隱	韻	吋	右	宇	烏	羽	迂	雨	卯
50	押	旺	横	欧	毆	王	翁	襖	鶯	鷗	黄	岡
51	魁	晦	械	海	灰	界	皆	絵	芥	蟹	開	階
52	粥	刈	刈	瓦	乾	侃	冠	寒	刊	勘	勸	卷
53	機	帰	毅	気	汽	畿	祈	季	稀	紀	徽	規

FIRST BYTE

D. Using Kanji

Decimal Value	SECOND BYTE											
	45	46	47	48	49	50	51	52	53	54	55	56
33	ノ	、	”	^	—	—	、	ゞ	、	ゞ	”	全
34	↓	=										
35				0	1	2	3	4	5	6	7	8
36	き	ぎ	く	ぐ	け	げ	こ	ご	さ	ざ	し	じ
37	キ	ギ	ク	グ	ケ	ゲ	コ	ゴ	サ	ザ	シ	ジ
38	N	三	O	Π	P	Σ	T	T	Φ	X	Ψ	Ω
39	Л	M	H	O	Π	P	C	T	V	Φ	X	Ц
48	悪	握	渥	旭	葦	芦	鱒	梓	圧	幹	扱	宛
49	鵜	窺	丑	碓	臼	渦	嘘	唄	鬱	蔚	鰻	姥
50	沖	荻	億	屋	憶	臆	桶	牡	乙	俺	卸	恩
51	貝	凱	効	外	咳	害	崖	慨	概	涯	碍	蓋
52	喚	堪	姦	完	官	寬	干	幹	患	感	慣	憾
53	記	貴	起	軌	輝	飢	騎	鬼	龜	偽	儀	妓

Decimal Value	SECOND BYTE											
	57	58	59	60	61	62	63	64	65	66	67	68
33	々	ノ	〇	一	一	一	/	\	~			...
34												
35	9								A	B	C	D
36	す	ず	せ	ぜ	そ	ぞ	た	だ	ち	ぢ	っ	っ
37	ス	ズ	セ	ゼ	ソ	ゾ	タ	ダ	チ	ヂ	ツ	ツ
38									α	β	γ	δ
39	ㇿ	ㇿ	ㇿ	ㇿ	ㇿ	ㇿ	ㇿ	ㇿ	ㇿ	ㇿ	ㇿ	ㇿ
48	姐	虻	飴	絢	綾	鮎	或	粟	裕	安	庵	按
49	既	浦	瓜	閏	樽	云	運	雲	荏	餌	叡	營
50	温	穩	音	下	化	仮	何	如	価	佳	加	可
51	街	該	鎧	骸	淫	馨	蛙	垣	柿	蠣	鈎	劃
52	換	敢	柑	桓	棺	款	歡	汗	漢	澗	灌	環
53	宜	戯	技	擬	欺	犧	疑	祇	義	蟻	誼	議

FIRST BYTE

D. Using Kanji

D. Using Kanji

Decimal Value	SECOND BYTE											
	69	70	71	72	73	74	75	76	77	78	79	80
33	..	'	'	"	"	()	()	[]	{
34												
35	E	F	G	H	I	J	K	L	M	N	O	P
36	づ	て	で	と	ど	な	に	ぬ	ね	の	は	ば
37	ツ	テ	デ	ト	ド	ナ	ニ	ヌ	ネ	ノ	ハ	バ
38	ε	ς	η	θ	ι	κ	λ	μ	ν	ξ	ο	π
39												
48	陪	案	闇	鞍	杏	以	伊	位	依	偉	罍	夷
49	嬰	影	映	曳	榮	永	泳	洩	瑛	盈	穎	穎
50	嘉	夏	嫁	家	寡	科	暇	果	架	歌	河	火
51	嚇	各	廓	擴	攪	格	核	殼	獲	確	獲	覺
52	甘	監	看	竿	管	簡	緩	缶	翰	肝	鑑	莞
53	翔	菊	鞠	吉	屹	喫	桔	橘	詰	砧	杵	黍

Decimal Value	SECOND BYTE											
	81	82	83	84	85	86	87	88	89	90	91	92
33	}	<	>	<	>	「	」	『	』	【	】	+
34												
35	Q	R	S	T	U	V	W	X	Y	Z		
36	ぱ	ひ	び	ぴ	ふ	ぶ	ぷ	へ	へ	へ	ほ	ぼ
37	パ	ヒ	ビ	ピ	フ	ブ	プ	ヘ	ヘ	ヘ	ホ	ボ
38	ρ	σ	τ	υ	φ	χ	ψ	ω				
39	a	б	в	г	д	е	ё	ж	з	и	й	к
48	委	威	尉	惟	意	慰	易	椅	為	畏	異	移
49	英	衛	詠	銳	液	疫	益	駅	悦	謁	越	閱
50	珂	禍	禾	稼	箇	花	苛	茄	荷	華	菓	蝦
51	角	赫	較	郭	閣	隔	革	学	岳	樂	額	顎
52	觀	諫	貫	還	鑑	間	閑	関	陷	韓	館	館
53	却	客	脚	虐	逆	丘	久	仇	休	及	吸	宮

FIRST BYTE

D. Using Kanji

Decimal Value	SECOND BYTE											
	93	94	95	96	97	98	99	100	101	102	103	104
33	-	±	X	÷	=	≠	<	>	≤	≥	∞	∴
34												
35					a	b	c	d	e	f	g	h
36	ぽ	ま	み	む	め	も	ゃ	や	ゅ	ゆ	よ	よ
37	ポ	マ	ミ	ム	メ	モ	ャ	ヤ	ユ	ユ	ヨ	ヨ
38												
39	л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц
48	維	緯	胃	菱	衣	謂	違	遺	医	井	亥	域
49	榎	厭	円	園	堰	奄	宴	延	怨	掩	援	沿
50	課	擘	貨	迎	過	霞	蚊	俄	峨	我	牙	画
51	掛	笠	桴	櫃	梶	鯽	湯	割	喝	恰	括	活
52	丸	含	岸	巖	玩	癌	眼	岩	翫	贖	雁	頑
53	弓	急	救	朽	求	汲	泣	灸	球	究	窮	笈

Decimal Value	SECOND BYTE												
	105	106	107	108	109	110	111	112	113	114	115	116	
FIRST BYTE	33	♂	♀	'	'	"	°	¥	\$	¢	£	%	#
	34												
	35	i	j	k	l	m	n	o	p	q	r	s	t
	36	ら	り	る	れ	ろ	わ	わ	ぬ	え	を	ん	
	37	ラ	リ	ル	レ	ロ	ワ	ワ	ヅ	エ	ヲ	ン	ヴ
	38												
	39	ч	ш	щ	б	ы	ь	э	ю	я			
	48	育	郁	磯	一	吉	溢	逸	稻	茨	芋	鱈	允
	49	演	炎	炤	煙	燕	猿	縁	艶	苑	菌	遠	鉛
	50	臥	芽	蛾	賀	雅	餓	駕	介	会	解	回	塊
	51	渴	滑	葛	褐	轄	且	鯉	叶	椈	樺	鞆	株
	52	顔	願	企	伎	危	喜	器	基	奇	嬉	寄	岐
53	級	糾	給	旧	牛	去	居	巨	拒	抛	拳	渠	

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	カ	ケ
38
39
48	印	咽	員	因	姻	引	飲	淫	胤	蔭
49	鴛	塩	於	汚	甥	凹	央	奧	往	応
50	懐	廻	快	怪	悔	恢	懷	戒	拐	改
51	兜	竈	蒲	釜	鎌	齒	鳴	栢	茅	萱
52	希	幾	忌	揮	机	旗	既	期	棋	棄
53	虚	許	距	鋸	漁	禦	魚	亨	亨	京

Decimal Value	SECOND BYTE											
	33	34	35	36	37	38	39	40	41	42	43	44
54	供	狹	僑	兇	競	共	凶	協	匡	卿	叫	喬
55	掘	窟	杏	靴	轡	窪	熊	隈	采	栗	繰	桑
56	検	権	牽	犬	献	研	硯	絹	具	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	懇	昏	昆	根
59	察	抄	撮	擦	札	殺	薩	雑	臯	鯖	捌	鏑
60	次	滋	治	爾	壘	痔	磁	示	而	耳	白	蒔
61	宗	就	州	修	愁	拾	洲	秀	秋	終	繡	習
62	勝	匠	升	召	肖	商	唱	嘗	獎	妾	娼	宵
63	拭	植	殖	燭	織	職	色	触	食	蝕	辱	尻
64	澄	摺	寸	世	瀨	畝	是	凄	制	勢	姓	征
65	織	羨	腺	舛	舩	薦	詮	賤	踐	選	遷	錢
66	臆	截	贈	造	促	側	則	即	息	捉	束	測

FIRST BYTE

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	鍬	勳	君	薰	訓	群	軍	郡	卦	袈	祓	係
56	賢	軒	遣	鍵	險	踮	驗	鹼	元	原	蔽	幻
57	広	庚	康	弘	恒	慌	抗	拘	控	攻	昂	晃
58	栖	混	痕	紺	良	魂	些	佐	又	唆	嗟	左
59	鮫	皿	晒	三	傘	參	山	慘	撒	散	棧	燦
60	辞	汐	鹿	式	識	鳴	竺	軸	穴	栗	七	叱
61	臭	舟	菟	衆	襲	讐	蹴	輯	週	囿	酬	集
62	將	小	少	尚	庄	床	廠	彰	承	抄	招	掌
63	伸	信	侵	唇	娠	寢	審	心	慎	振	新	晉
64	性	成	政	整	星	晴	棲	栖	正	清	牲	生
65	銑	閔	鮮	前	善	漸	然	全	禪	繕	膳	糲
66	足	速	俗	属	賊	族	統	卒	袖	其	揃	存

FIRST BYTE

Decimal Value	SECOND BYTE												
	57	58	59	60	61	62	63	64	65	66	67	68	
FIRST BYTE	54	狹	矯	陶	脅	興	蒼	鄉	鏡	響	饗	驚	仰
	55	傾	刑	兄	啓	圭	珪	型	契	形	徑	惠	慶
	56	弦	減	源	玄	現	絃	眩	言	諺	限	乎	個
	57	更	杭	校	梗	構	江	洪	浩	港	溝	甲	皇
	58	差	查	沙	磋	砂	詐	鎖	袞	坐	座	挫	債
	59	珊	産	算	纂	蚕	讚	贊	酸	餐	斬	暫	殘
	60	執	失	嫉	室	悉	湿	漆	疾	質	寔	蔀	襍
	61	醜	什	住	充	十	從	戎	柔	汁	浹	獸	縱
	62	捷	昇	昌	昭	晶	松	梢	樟	樵	沼	消	涉
	63	森	榛	浸	深	申	疹	真	神	秦	紳	臣	芯
	64	盛	精	聖	声	製	西	誠	誓	請	逝	醒	青
	65	憎	塑	咀	措	曾	曾	楚	狙	疏	疎	礎	祖
	66	孫	尊	損	村	遜	他	多	太	汰	詔	唾	墮

Decimal Value	SECOND BYTE											
	69	70	71	72	73	74	75	76	77	78	79	80
54	凝	堯	曉	業	局	曲	極	玉	桐	籽	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	絃	絞	網	耕	考	肯	肱	腔
58	催	再	最	哉	塞	妻	宰	彩	才	採	栽	歲
59	仕	仔	伺	使	刺	司	史	嗣	四	士	始	姊
60	悃	柴	芝	屢	藥	縞	舍	写	射	捨	赦	斜
61	重	銃	叔	夙	宿	淑	祝	縮	肅	塾	熟	出
62	湘	燒	焦	照	症	省	硝	礁	祥	稱	章	笑
63	薪	親	診	身	辛	進	針	震	人	仁	刃	塵
64	靜	齊	稅	脆	隻	席	惜	戚	斥	昔	銜	石
65	租	粗	素	組	蘇	訴	阻	遯	鼠	僧	魚	双
66	妥	惰	打	柁	舵	橈	陀	馱	驛	忤	连	对

Decimal Value	SECOND BYTE												
	81	82	83	84	85	86	87	88	89	90	91	92	
FIRST BYTE	54	均	巾	錦	斤	欣	欽	琴	禁	禽	筋	緊	芹
	55	繼	繫	罰	莖	荊	螢	計	詣	警	輕	頸	鵝
	56	狐	粧	袴	股	胡	蕪	虎	誇	跨	鈷	雇	顧
	57	膏	航	荒	行	衡	講	貢	購	郊	酵	鉍	礦
	58	濟	災	采	犀	碎	砦	祭	齋	細	菜	裁	載
	59	姿	子	屍	市	師	志	思	指	支	攷	斯	施
	60	煮	社	紗	者	謝	車	遮	蛇	邪	借	勺	尺
	61	術	述	俊	峻	春	瞬	竣	舜	駿	准	循	旬
	62	粧	紹	肖	萑	蔣	蕉	衝	裳	訟	証	詔	詳
	63	壬	尋	甚	尽	腎	訊	迅	陣	鞞	筭	諷	須
	64	積	籍	績	脊	責	赤	跡	蹟	碩	切	拙	接
	65	叢	倉	喪	壯	奏	爽	宋	層	匝	惣	想	搜
	66	耐	岱	帶	待	怠	態	戴	替	泰	滯	胎	腿

Decimal Value	SECOND BYTE											
	93	94	95	96	97	98	99	100	101	102	103	104
54	菌	衿	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	閤	降	頂	香	高	鴻	剛	劫	号	合	壕
58	際	劑	在	材	罪	財	呀	坂	阪	堺	紳	肴
59	旨	枝	止	死	氏	獅	祉	私	糸	紙	紫	肢
60	杓	灼	爵	酌	積	錫	若	寂	弱	惹	主	取
61	楯	殉	淳	準	潤	盾	純	巡	遵	醇	順	処
62	象	賞	醬	鉦	鍾	鐘	障	鞘	上	丈	丞	乘
63	酢	凶	厨	逗	吹	垂	帥	推	水	炊	睡	粹
64	損	折	設	窃	節	說	雪	絕	舌	蟬	衲	先
65	掃	挿	搔	操	早	曹	巢	槍	槽	漕	燥	争
66	苔	袋	貸	退	逮	隊	黛	鯛	代	台	大	第

Decimal Value	SECOND BYTE												
	105	106	107	108	109	110	111	112	113	114	115	116	
FIRST BYTE	54	狗	玗	矩	苦	軀	馱	駝	駒	具	愚	虞	喰
	55	潔	穴	結	血	訣	月	件	俟	倦	健	兼	券
	56	檣	瑚	碁	語	誤	護	翻	乞	鯉	交	佼	侯
	57	拷	濠	豪	轟	麴	克	刻	告	国	穀	酷	鶴
	58	咲	崎	崎	碕	鷺	作	削	咋	搾	昨	朔	柵
	59	脂	至	視	詞	詩	試	誌	諮	資	賜	雌	飼
	60	守	手	朱	殊	狩	珠	種	腫	趣	酒	首	儒
	61	初	所	暑	曙	渚	庶	緒	署	書	薯	諸	諸
	62	冗	剩	城	場	壤	嬢	常	情	擾	糸	杖	淨
	63	翠	哀	遂	醉	錐	錘	隨	瑞	髓	崇	嵩	數
	64	千	占	宣	專	尖	川	戰	扇	撰	柱	栴	泉
	65	瘦	相	窓	糟	綵	綜	聰	草	莊	葬	蒼	藻
66	醜	題	鷹	滝	瀧	卓	啄	宅	托	挾	拓	沢	

Decimal Value	SECOND BYTE									
	117	118	119	120	121	122	123	124	125	126
54	空	偶	寓	遇	隅	串	櫛	釧	脣	屈
55	劍	喧	罝	堅	嫌	建	畫	懸	拳	捲
56	候	倅	光	公	功	効	勾	厚	口	向
57	黑	獄	鹿	腰	飶	忽	惚	骨	狛	込
58	窄	策	索	錯	桜	鮭	笹	匙	冊	刷
59	齒	事	似	侍	児	字	寺	慈	持	時
60	受	呪	寿	授	樹	緩	需	囚	収	周
61	助	叙	女	序	徐	恕	鋤	除	傷	償
62	狀	冒	穢	蒸	讓	釀	錠	囑	埴	飾
63	枢	趨	雖	据	杉	椳	菅	頗	雀	裾
64	浅	洗	染	潜	煎	煽	旋	穿	筭	線
65	装	走	送	遭	鎗	霜	騷	像	堉	憎
66	濯	豚	託	鐸	濁	諾	苜	厠	蛸	只

Decimal Value	SECOND BYTE											
	33	34	35	36	37	38	39	40	41	42	43	44
67	叩	但	達	辰	奪	睨	巽	豎	迎	棚	谷	狸
68	帖	帳	疔	弔	張	彫	徵	懲	挑	暢	朝	潮
69	邸	鄭	釘	鼎	泥	摘	擢	敵	滴	的	笛	適
70	董	蕩	藤	討	膳	豆	踏	逃	透	鐙	陶	頭
71	如	尿	菲	任	妊	忍	認	濡	襪	衤	寧	葱
72	函	箱	裕	箸	肇	筭	櫨	幡	肌	畑	畠	八
73	鼻	柎	稗	匹	疋	髟	彥	蔭	菱	肘	強	必
74	福	腹	複	覆	淵	躬	扌	沸	仏	物	耐	分
75	法	泡	烹	砲	縱	胞	芳	萌	蓬	蜂	褒	訪
76	曼	蔓	味	未	魅	巳	箕	岬	密	蜜	湊	蓑
77	論	輸	唯	佑	優	勇	友	宥	幽	悠	憂	捐
78	痢	裏	裡	里	離	陸	律	率	立	蔭	掠	略
79	蓮	連	鍊	呂	魯	櫓	炉	賂	路	露	勞	婁

FIRST BYTE

Decimal Value	SECOND BYTE											
	45	46	47	48	49	50	51	52	53	54	55	56
67	鱒	樽	誰	丹	單	嘆	坦	担	探	旦	艱	淡
68	煤	町	眺	聽	脹	腸	蝶	調	謀	趨	眺	眺
69	鏞	溺	哲	徹	撤	輒	迭	鉄	典	眞	天	展
70	騰	閔	働	動	同	堂	導	撞	撞	洞	撞	童
71	猫	熱	年	念	捻	燃	燃	粘	乃	迺	之	埜
72	鉢	潑	兇	醜	髮	伐	罰	拔	筏	謁	鳩	斷
73	畢	筆	逼	檜	姬	媛	紐	百	謬	悽	彪	標
74	吻	噴	噴	憤	扮	焚	奮	粉	糞	紛	雲	又
75	豐	邦	鋒	飽	鳳	鵬	乏	亡	傍	並	行	苑
76	稔	脈	妙	耗	民	眠	務	夢	無	牟	子	露
77	有	柚	湧	涌	猶	猷	由	祐	裕	誘	遊	豆
78	劉	流	溜	琉	留	硫	粒	隆	竜	龍	任	憲
79	郎	弄	朗	樓	榔	浪	漏	牢	狼	龍	立	贗

Decimal Value	SECOND BYTE											
	57	58	59	60	61	62	63	64	65	66	67	68
67	湛	炭	短	端	箏	綻	耽	胆	蛋	誕	鍛	団
68	長	頂	鳥	勅	抄	直	朕	沈	珍	賃	鎮	陳
69	店	添	纏	甜	貼	轉	顛	点	伝	殿	澱	田
70	瞬	苟	道	銅	峠	錦	匿	得	德	瀆	特	督
71	囊	惱	濃	納	能	腦	膿	農	硯	蚤	巴	把
72	埴	蛤	隼	伴	判	半	反	叛	帆	搬	斑	板
73	氷	漂	瓢	票	表	評	豹	廟	描	病	秒	苗
74	聞	丙	併	兵	摒	幣	平	弊	柄	並	蔽	閉
75	帽	忘	忙	房	暴	望	某	棒	冒	紡	肪	彭
76	鵠	掠	婿	娘	冥	名	命	明	盟	迷	銘	鳴
77	郵	雄	融	夕	予	余	与	誉	輿	預	傭	幼
78	旅	虜	了	亮	僚	兩	凌	寮	料	梁	涼	狝
79	蠟	郎	六	麓	祿	助	錄	論	倭	和	話	歪

FIRST BYTE

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	津	墜	椎	梟	迫	鎚	痛	通	塚	柎	掘	槻
69	電	兔	吐	堵	塗	妬	屠	徒	斗	杜	渡	登
70	禿	篤	毒	獨	誅	朽	橡	凸	突	殺	屆	鳶
71	播	霸	杷	波	派	琶	破	婆	罵	芭	馬	俳
72	汜	汎	版	犯	班	畔	繁	般	藩	販	範	采
73	錨	鉞	蒜	蛭	鰭	品	彬	斌	浜	瀕	貧	賈
74	陞	米	頁	僻	壁	癖	碧	別	瞥	蔑	篋	偏
75	謀	貌	貿	鉞	防	吠	頰	北	僕	卜	墨	撲
76	姪	牝	滅	免	棉	綿	緬	面	麵	摸	模	茂
77	妖	容	庸	揚	搖	擁	躍	楊	樣	洋	溶	熔
78	療	瞭	稜	糧	良	諒	遼	量	陵	領	力	綠
79	賄	臨	惑	粹	鷺	互	巨	鷓	詔	藁	蕨	椀

FIRST BYTE

Decimal Value	SECOND BYTE											
	81	82	83	84	85	86	87	88	89	90	91	92
67	恥	智	池	痴	稚	置	致	蚰	遲	馳	築	畜
68	佃	漬	朽	辻	蕙	綴	鍔	椿	潰	坪	壺	孀
69	菟	賭	途	都	鍍	砥	礪	努	度	土	奴	怒
70	苦	寅	酉	澗	順	屯	惇	敦	沌	豚	遁	頓
71	堯	拝	排	敗	杯	盃	牌	背	肺	輩	配	倍
72	頰	頰	飯	挽	晚	番	盤	磐	蕃	蛮	匪	卑
73	頰	敏	瓶	不	付	埠	夫	婦	富	畠	布	府
74	変	片	篇	編	辺	返	遍	便	勉	媿	弁	鞭
75	朴	牧	陸	穆	釗	勃	没	殆	堀	幌	奔	本
76	妄	孟	毛	猛	盲	網	耗	蒙	儲	木	默	目
77	用	窯	羊	耀	葉	蓉	要	謠	踊	遙	陽	養
78	倫	厘	林	淋	燐	琳	臨	輪	隣	鱗	麟	溜
79	湾	碗	腕									

FIRST BYTE

Decimal Value	SECOND BYTE											
	93	94	95	96	97	98	99	100	101	102	103	104
67	竹	筑	蓄	逐	秩	窒	茶	嫡	着	中	仲	宙
68	紬	爪	吊	鈞	鶴	亭	低	停	偵	剗	貞	呈
69	倒	党	冬	凍	刀	唐	塔	塘	套	宕	島	嶋
70	吞	曇	鈍	奈	那	内	乍	屈	薙	謎	灘	捺
71	培	媒	梅	煤	煤	狽	買	壳	賄	陪	這	蠅
72	否	妃	庇	彼	悲	扉	批	披	斐	比	泌	疲
73	怖	扶	敷	斧	普	浮	父	符	腐	膚	芙	譜
74	保	舖	鋪	圃	捕	步	甫	補	輔	穗	募	墓
75	翻	凡	盆	摩	磨	魔	麻	埋	妹	味	枚	每
76	空	勿	餅	尤	戾	粿	貰	問	闕	紋	門	奴
77	慾	抑	欲	沃	浴	翌	翼	淀	羅	螺	裸	來
78	望	淚	累	類	令	伶	例	冷	勵	嶺	伶	令
79												

FIRST BYTE

Decimal Value	SECOND BYTE											
	105	106	107	108	109	110	111	112	113	114	115	116
67	忠	抽	昼	柱	注	虫	衷	註	耐	鑄	駐	樗
68	晁	定	帝	底	庭	廷	弟	悌	抵	挺	提	梯
69	悼	投	搭	東	桃	禱	棟	盜	淘	湯	濤	灯
70	鍋	櫛	馴	繩	暇	南	楠	軟	難	汝	二	尼
71	秤	矧	菽	伯	剝	博	拍	柏	泊	白	箔	粕
72	皮	碑	秘	緋	罷	肥	被	誹	費	避	非	飛
73	負	賦	赴	阜	附	侮	撫	武	舞	葡	蕪	部
74	慕	戊	暮	母	簿	菩	倣	捧	包	呆	報	奉
75	哩	楨	幕	膜	枕	鮪	証	鱒	枳	亦	俣	又
76	也	冶	夜	爺	耶	野	弥	矢	厄	役	約	藥
77	萊	賴	雷	洛	絡	落	酪	乱	卵	嵐	欄	藍
78	礼	苓	鈴	隸	零	靈	麗	齡	曆	歷	列	劣
79												

FIRST BYTE

Decimal Value	SECOND BYTE									
	117	118	119	120	121	122	123	124	125	126
67	猪	猪	苧	著	貯	丁	兆	涸	喋	寵
68	汀	碇	禎	程	締	幌	訂	諦	蹄	遞
69	燈	当	痘	禱	等	答	筒	糖	統	到
70	式	邇	勺	賑	肉	虹	甘	日	乳	入
71	舶	薄	迫	曝	漠	爆	縛	莫	駁	麦
72	樋	簸	備	尾	微	枇	毘	琵	眉	美
73	封	楓	風	葺	落	伏	副	復	幅	服
74	宝	峰	峯	崩	庖	抱	捧	放	方	朋
75	抹	未	沫	迄	儘	繭	磨	万	慢	滿
76	訳	躍	靖	柳	藪	鏡	愉	愈	油	癒
77	藍	蘭	覧	利	吏	履	李	梨	理	璃
78	烈	裂	廉	恋	憐	漣	煉	簾	練	聯
79

FIRST BYTE

Glossary

Absolute Plotting — Plotting to a point whose location is specified with respect to the fixed origin (0,0).

Address — The address specifies the plotter's location on the HP-IB (IEEE-488) interface cable (bus).

Arc — A portion of the circumference of a circle.

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.

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.

Byte — Eight bits; the size of a computer word. Used by ASCII binary code to represent alphanumeric characters.

Carriage-Return Point — The point the pen moves to when the plotter receives a carriage return (while in label mode).

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. The character occupies the lower-left portion of the cell, so that there is a blank area to the right and above the character.

Glossary (Continued)

Character Set — A group of characters, each of which is defined by a unique ASCII binary code. Typically, a character set contains related characters, such as a character set composed of math symbols, or characters in a foreign language.

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 point 10,20 are constants.

Control Character — A character that starts, modifies, or stops computer or peripheral operation.

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 eliminate mistakes in a computer program.

Decimal Code — The decimal equivalent value of an ASCII character. Refer to the ASCII code table in Appendix B.

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, so that it can be understood by the computer.

Glossary (Continued)

Edge — The outline of a polygon.

End of File (EOF) — A “marker” that signals the end of a file.

Execute — To carry out an instruction or perform a routine. For example, when a plotter executes the select pen instruction, it gets a pen.

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.

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: hardware, 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.

*Electronic Industries Association.

Glossary (Continued)

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 that they are able to function in a compatible and coordinated fashion. Standards which 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 and/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 blank area above it. The default size of a line is 2 times the height of a capital A.

Line Feed (LF) — A non-printing ASCII character that moves the plotter pen down one line (when in label mode).

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 abbreviation that is easy to remember. HP-GL instructions are two-letter mnemonics that represent the function of the instruction. For example, SP for select pen, and LT for line type.

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.

Glossary (Continued)

Modulo — A method of converting an out-of-range parameter into an acceptable parameter. For example, when using arc and circle instructions, the upper limit of an angle parameter is 360 degrees. Should you send a parameter of 1000 instead of 0 to 360, the plotter divides $1000/360$ and uses the remainder as the parameter.

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's storage space.

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, *SP2;*, directs the plotter to select pen number 2; using a parameter of 1 directs the plotter to 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 — When an instruction is parsed, it is broken into components so that it can be executed. For example, an HP-GL instruction would be broken into a mnemonic, parameters, separators, and a terminator.

Plotter Units — The fixed units the plotter understands. Each plotter unit is 0.025 mm.

Point — A location in the plotting area defined by an X,Y coordinate pair.

Poll — In an HP-IB (IEEE-488) configuration, polling is a process used by the computer to determine which device is requesting service.

Polygon — A closed shape. Polygons can be simple shapes such as circles, rectangles, and wedges, or more complex shapes such as block letters.

Glossary (Continued)

Polygon Buffer — A portion of the plotter's buffer that stores the polygon that is being defined.

Polygon Mode — A mode established by the PM instruction. In this mode, the points used to define a polygon are temporarily stored in the plotter's polygon buffer (rather than being executed).

Relative Plotting — Plotting to a point whose location is specified *relative* to the current pen position.

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: *PA 10 , 20 ;*.

Soft-Clip Limits — That part of the plotting area defined by the IW 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.

Status Byte — A byte which reflects the plotter's status.

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, or symbols.

*Electronic Industries Association.

Glossary (Continued)

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 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.

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.

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.

Variable — A value that can be changed; usually represented by a letter or group of letters.

Window — The part of the plotting area in which plotting can occur. Also referred to as soft-clip limits.

Notes

Subject Index

A

- AA instruction 5-8-5-10
- AP instruction 4-6-4-8
- AR instruction 5-10-5-12
- Abort device-control instruction 13-8
- Abort graphics instruction 13-8
- Absolute
 - arc 5-8-5-10
 - character size 8-29, 8-30
 - coordinates 2-1, 2-2,
 - direction 8-11-8-15
 - movement 4-2, 4-3
 - plotting 4-8-4-10
 - rectangle 5-28-5-30
- Alternate character set
 - general 9-3, 9-4
 - designating 9-5, 9-6
 - selecting 9-7-9-9
- Arc absolute
 - instruction 5-8-5-10
- Arc relative
 - instruction 5-10-5-12
- Arcs
 - general 5-1-5-6
 - maintaining arc smoothness 5-2
- ASCII characters
 - for each character set B-5-B-8
 - in terminating labels 8-19, 8-20
 - table of control characters B-2, B-3
 - using CHR\$ function 8-3
 - using keyboard 8-3
- Automatic modem disconnect modes 14-22, 14-23
- Automatic pen operations instruction 4-6-4-8
- Axis align
 - effect of new axis align point on hard-clip limits 2-14
 - after initializing 2-14, 2-15
 - after loading paper 2-15
 - power-on location 2-13

B

- BASIC
 - in program examples 1-5
 - using non-BASIC statements 1-6
- Baud rate 14-6
- Block transmission mode 14-20-14-22
- Buffers, *see also* I/O buffer, Logical I/O buffer, Physical I/O buffer, Polygon buffer, and Pen sort buffer
 - allocating size using ESC.T instruction 1-10, 1-11, 13-15-13-18
 - allocating size using GM instruction 1-10, 1-11, 6-13, 6-14
 - general 1-8-1-11

C

- CA instruction 9-5, 9-6
- CI instruction 5-12-5-14
- CP instruction 8-8-8-11
- CT instruction 5-15, 5-16
- CS instruction 9-6, 9-7
- Carriage-return point 8-4, 8-5
- Character plot cell
 - default size 8-5
 - general 8-5, 8-6
 - in horizontal mode 8-8, 8-9
 - in vertical mode 8-10
- Character plot instruction 8-8-8-11
- Characters, ASCII B-1-B-3, B-5-B-8
- Character sets
 - designating and selecting 9-3, 9-4
 - general 9-1-9-3
 - Kanji D-7-D-30
 - list of 9-2, B-4
 - standard 9-6, 9-7, 9-9
 - tables B-5-B-8
 - alternate 9-5, 9-6, 9-7-9-9

Subject Index (Continued)

C (Continued)

- Character size
 - absolute 8-29, 8-30
 - relative 8-33-8-35
- Character slant 8-31-8-33
- Character strings 3-3
- Characters, user-defined, *see also*
 - Character plot cell
 - general 9-4
 - instruction 9-10-9-13
- Chord tolerance 5-2-5-6
 - chord angle 5-2, 5-3
 - chord calculation 5-4, 5-5
 - degrees mode 5-2, 5-3
 - deviation distance mode 5-4
- Chord tolerance instruction 5-15, 5-16
- Circle instruction 5-12-5-14
- Circles
 - general 5-1-5-6
 - smoothness 5-2-5-6
- Creating mirror images of plots 10-5, 10-6
- Configurable memory, *see* Buffers
- Configurable memory size, output 13-14, 13-15
- Configuration
 - statement 1-3-1-5
- Coordinate system 2-1, 2-2
- Current units 3-3

D

- DC instruction 12-7
- DF instruction 3-8-3-11
- DI instruction 8-11-8-15
- DP instruction 12-7, 12-8
- DR instruction 8-15-8-18
- DT instruction 8-18-8-21
- DV instruction 8-21-8-23
- Data bits 14-5
- Data transmission
 - modes 14-20-14-22
- Define label terminator
 - instruction 8-18-8-21

- Degrees Mode 5-2, 5-3
- Designing characters 9-4
- Deviation distance mode 5-4
- Device-control instructions
 - abort 13-8
 - description 1-3
 - for changing buffer sizes 13-4
 - for identifying the plotter 13-4
 - for setting plotter to known conditions 13-3
 - for verifying errors or operating conditions 13-4, 13-5
 - general 13-1
 - sending 13-2
 - syntax 13-2
 - using 13-3-13-5
- Digitize clear instruction 12-7
- Digitize point instruction 12-7, 12-8
- Digitizing
 - general 12-1
 - HP-IB interrupts and polling 12-6
 - instructions 12-1, 12-7-12-9
 - manual 12-2, 12-3
 - monitoring the status
 - byte 12-4-12-6
 - over HP-IB with HP 150 Touchscreen 12-1
- Direction absolute
 - instruction 8-11-8-15
- Direction mode, *see* DV instruction
- Direction relative
 - instruction 8-15-8-18
- Direction vertical
 - instruction 8-21-8-23
- Disconnect modes 14-22, 14-23

E

- EA instruction 5-16-5-18
- EP instruction 6-9-6-11
- ER instruction 5-18-5-20

Subject Index (Continued)

E (Continued)

ES instruction 8-23-8-25
EW instruction 5-20-5-23
ESC.(13-18
ESC.) 13-18, 13-19
ESC.@ 13-19-13-21
ESC.A 13-5
ESC.B 13-5, 13-6
ESC.E 13-6, 13-7
ESC.H 14-23, 14-24
ESC.I 14-24-14-27
ESC.J 13-8
ESC.K 13-8
ESC.L 13-9
ESC.M 14-28-14-30
ESC.N 14-31, 14-32
ESC.O 13-9-13-11
ESC.P 14-32, 14-33
ESC.Q 13-11-13-14
ESC.R 13-14
ESC.S 13-14, 13-15
ESC.T 13-15-13-18
ESC.Y 13-18
ESC.Z 13-18, 13-19
E-mask 11-9, 11-10
Edge absolute rectangle
instruction 5-16-5-18
Edge polygon
instruction 6-9-6-11
Edge relative rectangle
instruction 5-18-5-20
Edge wedge
instruction 5-20-5-23
Enlarging a scaled picture 10-2,
10-3
Equal-sized pictures on one page,
drawing 10-3, 10-4
Enquire/Acknowledge
handshake
general 14-7
using device-control
instructions 14-13, 14-14
using ESC.H
instruction 14-15-14-17
using ESC.I instruction
14-14, 14-15, 14-17

using ESC.P
instruction 14-14
Errors
identifying 11-4, 11-5
HP-GL 11-15, A-1
device-control A-2
Examples
complete program listings 1-5
configuration statements 1-4,
1-5
HP-GL strings 1-6, 1-7
Extra space
instruction 8-23-8-25

F

FP instruction 6-11, 6-12
FT instruction 5-7, 5-24-5-26
Fill types 5-7, 5-24-5-26
Fill rectangle
absolute 5-28-5-30
Fill rectangle relative 5-30, 5-31
Fill polygon 6-11, 6-12
Fill wedge 5-32-5-36
Flowchart
output request 14-30
software checking
handshake 14-18
FORTRAN 1-1

G

GM instruction 6-13, 6-14
Graphics, abort 13-8
Graphics Memory
instruction 6-13, 6-14

H

Handshaking, *see also*
Enquire/Acknowledge
handshake,
Hardware handshake, Software
checking handshake, Xon-
Xoff handshake

Subject Index (Continued)

H (Continued)

- general 14-5, 14-6
- choosing a
 - handshake 14-6-14-8
- using device-control instructions 14-8-14-20
- Hard-clip limits 2-4, 2-5
- Hardwire handshake
 - general 14-7
 - using device-control instructions 14-8-14-10
 - using ESC . @
 - instruction 14-10
 - using ESC.P
 - instruction 14-10
- Horizontal direction mode
 - positive run and rise in 8-12
- HP-GL
 - description 1-2
 - errors 11-15, A-1
 - NOP (No Operation) A-3
 - obtaining HP-GL
 - errors 11-14-11-16
 - sending HP-GL
 - instructions 1-3-1-7
- HP-IB
 - addressing 14-3, 14-4
 - general 14-1
 - interface functions 14-2
 - reactions to DCL and SDC 14-5
 - valid device-control instructions 14-2

I

- IM instruction 11-8-11-12
- IN instruction 3-11, 3-12
- IP instruction 3-12, 3-13
- IW instruction 10-9-10-11
- Initialize instruction 3-11, 3-12
- Initializing the plotter, *see also*
 - DF instruction and IN instruction
 - general 3-4, 3-5

- Input Mask
 - instruction 11-8-11-12
- Input P1 and P2 instruction 3-12, 3-13
- Input Window
 - instruction 10-9-10-11
- I/O buffer, *see also* Logical I/O
 - buffer and Physical I/O buffer
 - configurable size 1-10
 - general 1-8, 1-9

K

- 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

L

- LB instruction 8-25, 8-26
- LT instruction 7-1-7-5
- Label
 - default label conditions 8-1, 8-2
 - direction 8-11-8-18, 8-21-8-23
 - enhancing 8-6, 8-7
 - extra space 8-23-8-25
 - instruction 8-25, 8-26
 - label direction (default) 8-2
 - origin 8-21, 8-22, 8-26-8-28
 - positioning 8-3-8-5
 - terminator (default) 8-2
 - sending the label
 - terminator 8-2
 - size (default) 8-2
 - slant (default) 8-2
 - using variables 8-7
- Label terminator
 - default 8-2
 - sending the label
 - terminator 8-2

Subject Index (Continued)

L (Continued)

- Leased-line disconnect mode 14-23
- Line types
 - fixed and adaptive 7-2, 7-3
 - instruction 7-1-7-5
- Logical I/O buffer
 - general 1-8, 1-9
 - configuring size 13-19-13-21

M

- Masks, *see also* IM instruction
 - E-mask 11-9, 11-10
 - P-mask 11-11
 - S-mask 11-10, 11-11
- Mirror images 10-5, 10-6
- Monitor mode, set 13-11-13-14, 13-19-13-21

N

- NR instruction 3-13
- Not ready instruction 3-13
- Normal transmission mode 14-20
- Numbers
 - integer 3-3
 - real 3-3
 - rounding 3-3

O

- OA instruction 11-12, 11-13
- OC instruction 11-13, 11-14
- OD instruction 12-8, 12-9
- OE instruction 11-14-11-16
- OF instruction 11-16
- OH instruction 10-11, 10-12
- OI instruction 11-16, 11-17
- OO instruction 11-17, 11-18
- OP instruction 10-12, 10-13
- OS instruction 11-19, 11-20
- OT instruction 11-20

- OW instruction 10-13, 10-14
- Obtaining plotter information 11-1-11-8
- Omitting parameters 3-4
- Output actual pen status instruction 11-12, 11-13
- Output buffer size when empty instruction 13-9
- Output buffer space instruction 13-5, 13-6
- Output carousel type instruction 11-20
- Output commanded pen status instruction 11-13, 11-14
- Output configurable memory size instruction 13-14, 13-15
- Output digitized point and pen status instruction 12-8, 12-9
- Output error instruction 11-14-11-16
- Output extended error instruction 13-6, 13-7
- Output extended status instruction 13-9-13-11
- Output factors instruction 11-16
- Output hard-clip instruction 10-11, 10-12
- Output identification instruction
 - HP-GL 11-16, 11-17
 - device-control 13-5
- Output instructions
 - description 1-3, 11-1, 11-2
 - in HP-IB configuration 11-2, 11-3
 - in RS-232-C configuration 11-3
 - using 10-8, 11-3-11-5
- Output options instruction 11-17, 11-18
- Output P1 and P2 instruction 10-12, 10-13
- Output request flowchart 14-30
- Output response, summary 11-5, 11-6
- Output status instruction 11-19, 11-20

Subject Index (Continued)

O (Continued)

Output Window instruction
10-13, 10-14

P

PA instruction 4-8-4-10
PD instruction 4-10-4-12
PM instruction 6-14-6-18
PR instruction 4-12-4-14
PT instruction 5-27, 5-28
PU instruction 4-14, 4-15
P-mask 11-11
P1 and P2
 default locations 2-7, 2-8
 description 2-7
 input instruction 3-12, 3-13
Parity 14-5
Parallel polling 11-7, 11-8
Parse mode 13-11-13-14
Pascal 1-1
Pen
 automatic operations 4-6-4-8
 identifying location and
 position 11-3, 11-4, 11-12,
 11-13
 monitoring location and
 position 4-2, 11-14, 11-15
 moving 4-2
 selecting 4-1
Pen down instruction 4-10-4-12
Pen up instruction 4-14, 4-15
Pen sort buffer
 general 1-9, 1-10
 configuring size 13-15-13-18
 output size 13-14, 13-15
Pen sorting 4-7
Pen up merging 4-7
Physical I/O buffer
 general 1-8, 1-9
 configuring size 13-15-13-18
 output size 13-14, 13-15
Plot absolute
 instruction 4-8-4-10
Plot relative
 instruction 4-12-4-14

Plotter, identifying 11-16, 11-17,
 13-4, 13-5
Plotter functions,
 identifying 11-4
Plotter options, identifying
 11-17, 11-18
Plotter units, description and
 range 2-2, 2-3
Polling, *see also* IM instruction
 general 11-6, 11-7
 parallel 11-7, 11-8
 serial 11-7
Polygon buffer
 general 1-9, 6-4
 configuring size 6-9,
 13-15-13-18
 general 1-10
 determining approximate
 size 6-5
 determining exact size 6-7,
 6-8
 output size 13-14, 13-15
Polygon mode
 appropriate instructions in
 1-9, 6-1
 circles in 6-3, 6-4
 instruction 6-14-6-18
Polygons
 closing 6-16
 counting points in 6-5-6-7
 edging 6-9-6-11
 filling 6-11, 6-12
 general 6-1
 drawing 6-2
Primitive grid resolution 9-10
Program errors 1-8
Programming hints 1-7
Programming languages
 BASIC 1-1
 FORTRAN 1-1
 Pascal 1-1

R

RA instruction 5-6, 5-28-5-30
RO instruction 10-14-10-16

Subject Index (Continued)

R (Continued)

- RR instruction 5-6, 5-30, 5-31
- Receive mode 13-11-13-14
- Rectangles
 - absolute 5-16-5-18, 5-28-5-30
 - general 5-6-5-7
 - relative 5-18-5-20, 5-30, 5-31
- Reducing a scaled picture 10-2, 10-3
- Relative
 - arc 5-10-5-12
 - character size 8-33-8-35
 - coordinates 4-3-4-5
 - direction 8-15
 - movement 4-2-4-4
 - movement when scaling is off 4-5
 - plotting 4-12-4-14
 - rectangles 5-18-5-20, 5-30, 5-31
- Reset instruction 13-14
- Rotation
 - general 2-11, 2-12, 10-7, 10-8
 - instruction 10-14-10-16
 - program C-12-C-14
- RS-232-C
 - general 14-5-14-8

S

- SA instruction 9-7-9-9
- SC instruction 3-14-3-16
- SG instruction 4-15, 4-16
- SI instruction 8-29, 8-30
- SL instruction 8-31-8-33
- SM instruction 7-5, 7-6
- SP instruction 4-16, 4-17
- SR instruction 8-33-8-35
- SS instruction 9-9
- S-mask 11-10, 11-11
- Scale instruction 3-14-3-16
- Scaling, *see also* SC instruction
 - description 2-7-2-11
 - expanding or contracting scaled plots 3-6,

- isotropic and anisotropic scaling 3-6-3-8
 - using with IP instruction 3-5
- Select alternate character set instruction 9-7-9-9
- Select pen group instruction 4-15, 4-16
- Select pen instruction 4-16, 4-17
- Select standard character set instruction 9-9
- Serial polling 11-7
- Set handshake mode 1
 - instruction 14-23, 14-24
- Set handshake mode 2
 - instruction 14-24-14-27
- Set monitor mode
 - instruction 13-11-13-14
- Size absolute character 8-29, 8-30
- Size relative
 - character 8-33-8-35
- Slant character 8-31-8-33
- Soft-clip limits (windows), *see also* IW instruction
 - general 2-5-2-7, 10-1, 10-2
- Software checking handshake flowchart 14-18
 - general 14-7, 14-8, 14-17-14-19
 - using ESC.B instruction 14-19, 14-20
- Standard character set
 - general 9-3, 9-4
 - designating 9-6, 9-7
 - selecting 9-9
- Status byte, *see also* IM instruction
 - general 11-5
 - output instruction 11-19, 11-20
- Stop bits 14-6
- Summary of output responses 11-5, 11-6
- Switched/Datex-line disconnect mode 14-22
- Symbol mode instruction 7-5, 7-6

Subject Index (Continued)

S (Continued)

Symbol mode using Kanji D-5

Syntax

compatibility with other
plotters 3-3

HP-GL instructions 3-1

notations 3-2

parameter formats and
ranges 3-2

T

TL instruction 7-7-7-9

Tick length instruction 7-7-7-9

Tick marks, *see* XT instruction,

YT instruction, and

TL instruction

Transmission

modes 14-20-14-22

U (Continued)

UC instruction 9-10-9-13

User-defined character

instruction 9-10-9-13

User units, description and

range 2-2, 2-4

V

VS instruction 4-17, 4-18

Variables 4-6, 11-1, 11-2

Velocity select instruction 4-17,

4-18

Vertical direction mode

positive run and rise in 8-12

W

WG instruction 5-2, 5-5,
5-32-5-36

Wedges

general 5-5, 5-6

edging, 5-20-5-23, 6-9-6-11

filling, 5-24-5-26, 5-32-5-36

Window

general 10-1, 10-2

input instruction 10-9-10-11

output instruction 10-13,
10-14

program C-9-C-11

X

Xon-Xoff handshake

general 14-7

using device-control

instructions 14-10, 14-11

using ESC . I instruction

14-12, 14-13

using ESC . P

instruction 14-12

XT instruction 7-9, 7-10

X-tick instruction 7-9, 7-10

Y

YT instruction 7-10, 7-11

Y-tick instruction 7-10, 7-11

4. Drawing Lines

8. Labeling

12. Digitizing

B. Reference Material

3. HP-GL Prelim.

7. Plot Enhancement

11. Obtaining Output

A. Error Messages

2. Plotting Concepts

6. Polygons

10. Area/Orientation

14. Interfacing

1. Program Concepts

5. Geometric Shapes

9. Character Sets

13. Device-Control



HEWLETT
PACKARD



D. Using Kanji

C. Sample Listings