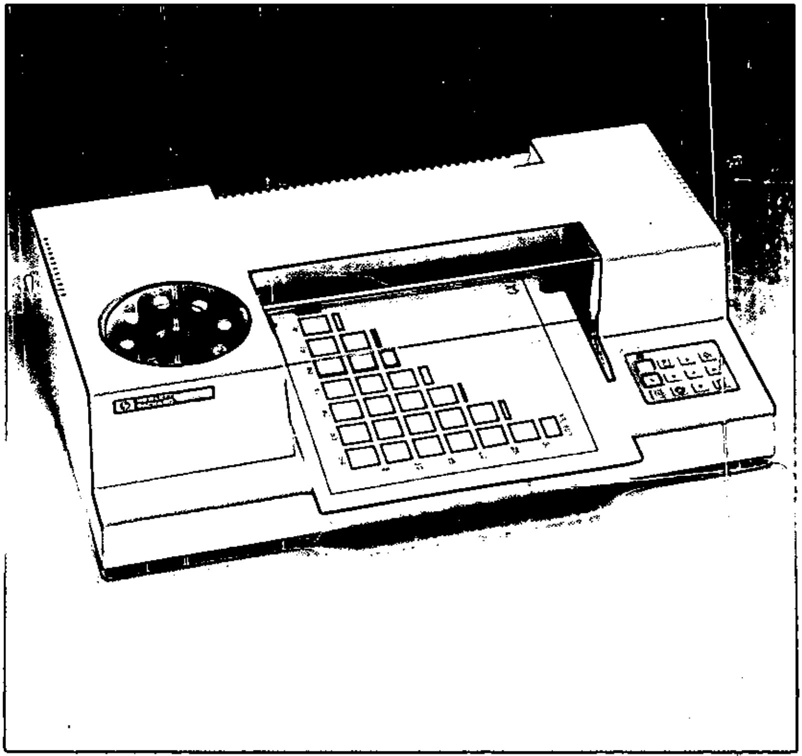
HEWLETT-PACKARD

HP ColorPro Graphics Plotter Programming Manual

Includes Instructions for the HP 17440
Graphics Enhancement Cartridge



NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 5963 SAN DIEGO, CA U.S.A.

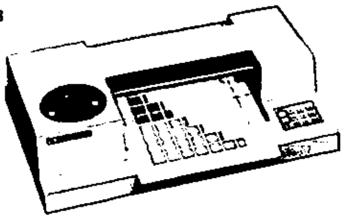
POSTAGE WILL BE PAID BY ADDRESSEE

HEWLETT
PACKARD

San Diego Division
Attn: Marketing Communications
16399 West Bernardo Drive
San Diego, CA U.S.A. 92127-9989

Your Comments Please...

You can help us improve our manuals by sharing your comments and suggestions. Please complete this questionnaire, and return it to us. All comments and suggestions become the property of HP.



Thank you for your help.

please tear off and mail in

HP ColorPro Graphics Plotter

1. Please answer these questions and use the lines below for additional comments.

Could you find the information you needed?	☐ Yes	☐ No
Did you understand the concepts and language?	☐ Yes	☐ No
Were the examples clear and useful?	☐ Yes	☐ No
Were there specific sections, pages, illustrations, that you found particularly helpful ☐ or confusing ☐?	☐ Yes	☐ No

Comments (please include page numbers): ____________________

____

____

2. Please describe your role(s) when using the plotter?

☐ I use a software package ☐ I install/maintain the plotter

☐ I write graphics programs ☐ Other ____________

3. What computers are used with this plotter? ____________

____

4. Indicate how you use graphics and what industry you're in.

☐ Pie/bar/line charts	☐ Business
☐ Flow charts/scheduling charts	☐ Manufacturing
☐ Logos or other art	☐ Science/Engineering
☐ Other ____________	☐ Other ____________

Please elaborate: ____________________

____

5. Would you like more information on Hewlett-Packard:

☐ Plotters	☐ Graphics Software
☐ Low-Frequency Measurement Plotting System	☐ Supplies Catalog
	☐ Other ____________

Name ____________________

Address ____________________

____

Business/home phone: ____________________

07440-90001 August 1985

please tear off and mail in

HP ColorPro Graphics Plotter

Programming Manual

Includes Instructions for the HP 17440 Graphics Enhancement Cartridge

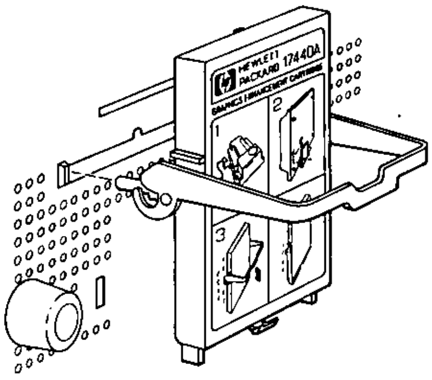
HEWLETT PACKARD

How to Use This Manual

This manual is written for the HP ColorPro plotter with two different functional levels:

- the standard HP ColorPro plotter, or
- the HP ColorPro plotter with an installed HP 17440 Graphics Enhancement Cartridge.

How you use this manual depends on whether or not your plotter is equipped with a cartridge, which slides into position underneath the plotter. It looks like this:



Graphics Enhancement Cartridge

The following figure is a replica of the cartridge; placed next to a section's title, it indicates that the information in that section pertains only to the cartridge.

For Standard Plotters

If you are programming a plotter that does not have a cartridge, you should *ignore* all sections marked with the cartridge symbol. These sections do not apply to a plotter without a cartridge.

For Plotters with a Graphics Enhancement Cartridge

If you are programming a plotter with a cartridge, all information introduced with the cartridge symbol is applicable. The cartridge provides additional HP-GL instructions, character sets, and interfacing capabilities. You should view the plotter and cartridge as one unit, and use all of the information in this manual.

For First Encounters with HP-GL

If you have never written programs using HP-GL (Hewlett-Packard Graphics Language), begin with Chapter 1. It introduces you to various programming languages and contains a simple example using HP-GL. The second chapter explains some basic plotting concepts, and the remaining chapters present the HP-GL instructions.

For Experienced HP-GL Programmers

If you are an experienced HP-GL programmer, you might find the Pocket Guide or the instruction summary in Appendix D of this manual most helpful. Also, look through Chapter 7 to learn about the polygon and area-fill instructions (available with the cartridge) not found in some earlier plotters. If you are interested in the differences in syntax between this and other HP plotters, read Chapter 3.

Other HP ColorPro Documentation

The Operating Manual (Part No. 07440-90002). 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.

The Pocket Guide (Part No. 07440-90003). The Pocket Guide is a convenient reference list of all HP-GL and device-control instructions, along with their parameters. You can order this from CSO or your HP Sales and Support Office.

A Note to Software Users

Many graphics software packages are available for the plotter. Check with your plotter or computer sales representative for information on software packages that can run the plotter with your computer. If you are using one of these, be sure to follow any special instructions provided in the documentation accompanying the software.

Table of Contents

Table of Contents (Continued)

Table of Contents (Continued)

Table of Contents (Continued)

Table of Contents (Continued)

Table of Contents (Continued)

Table of Contents (Continued)

Table of Contents (Continued)

Table of Contents (Continued)

Table of Contents (Continued)

CHAPTER 1

CHAPTER 1

Introduction to Programming the Plotter

What You'll Learn in This Chapter

This chapter will help acquaint you with your plotter and tell you how to make it work for you. You will be introduced to HP-GL (Hewlett-Packard Graphics Language) and device-control instructions, and will see a simple programming example.

About the Plotter

Your plotter has the capacity to produce almost any plot that you require. It accepts two standard media sizes:

ISO* A4 (210 × 297 mm)   ANSI* A(8.5 × 11 in.)

You can use three types of plotting media: standard chart paper, glossy presentation paper, and overhead transparency film (8.5 × 11 in. only). You can also use eight pens at any given time, choosing among different pen colors and pen widths. Use standard fiber-tip pens for paper and transparency fiber-tip pens for overhead transparency film and glossy paper.

Labels and text are easy to draw using any of five character sets, including three European sets.

*International Standards Organization (ISO)
American National Standards Institute (ANSI)

About the Graphics Enhancement Cartridge

The graphics enhancement cartridge gives the plotter expanded handshaking capability and plotting functions. With its 14 additional HP-GL instructions, you can easily draw arcs and circles, outline and fill polygons, and access 14 more character sets.

How This Manual Is Organized

The remainder of this chapter, along with Chapter 2, introduces you to programming languages and some basic plotting concepts to help get you started. The rest of the chapters present HP-GL instructions, grouped according to function. Each chapter begins with a section called *What You'll Learn in This Chapter*. Read this section to determine which instructions are presented and whether you need to read the chapter at all.

Most chapters also contain a list of important words under *Terms You Should Understand*; all of these words are defined in the glossary at the end of the manual.

Remember, if your plotter does not include a graphics enhancement cartridge, you can ignore all material designated with this symbol:

Each instruction begins with a paragraph titled **USES.** Read this to determine whether the instruction will be useful to you. In this way, you can learn HP-GL in a manner that is tailored to your application.

The instructions you are likely to use often are presented in Chapters 3 through 7. Then, in Chapter 8, you will find some sample programs that show you how to use the instructions together to create common types of charts. Appendix D presents an alphabetical summary of all instructions. The summaries are keyed to the page number in the manual where the instruction is discussed in detail.

Understanding 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 commas, counting both to the left and right of the decimal point (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 12, respectively.

Programming Languages — Which Should You Use?

In order to write graphics programs for the plotter, you will need to learn the programming languages that the computer and the plotter understand. This manual assumes that you already have some general experience programming with your computer's language. You can use any computer that can output in ASCII* and any language that outputs literal strings.

To help you understand how your computer communicates with the plotter, some programming languages are described next. Begin with the plotter's HP-GL instructions. Then read about your computer's language to see how it communicates with the plotter. Three common computer languages are described: BASIC,

*American Standard Code for Information Exchange

FORTRAN, and Pascal. If you are programming in another language, your computer documentation should tell you how to output literal strings to a peripheral (in this case, literal strings of HP-GL to the plotter).

HP-GL (Hewlett-Packard Graphics Language)

HP-GL instructions are codes that access the plotter's graphics functions. This manual provides full details on the capabilities of the HP-GL instruction set. Each instruction consists of a two-letter mnemonic designed to remind you of its function. For example, SP is the select pen instruction, and PD is the pen down instruction. Often these two-letter mnemonics are followed by numerical parameters that tell the plotter how to execute (complete) the instruction. A typical HP-GL instruction looks like this:

```
PD1500,1500;
```

To program the plotter, you must *insert the HP-GL instructions with their parameters in an appropriate output statement from your computer's language.* This is referred to as "sending" an HP-GL instruction to the plotter. You'll learn more about this in the section *How to Use the Examples in this Manual,* later in this chapter. When you send HP-GL instructions to the plotter, they enter the plotter's internal buffer; the plotter executes them on a first-in/first-out basis.

Device-Control Instructions (RS-232-C Only)

Device-control instructions control operations such as data formatting, input/output, handshaking, and buffer allocation. A device-control instruction is a three-character sequence which may or may not have parameters. A typical device-control instruction looks like this:

```
ESC.L
```

Device-control instructions are sent to the plotter as literal strings in a manner similar to sending HP-GL instructions. *Unlike HP-GL, however, device-control instructions do not enter the plotter's buffer and are executed immediately upon receipt.* Chapters 12 and 13 contain full details on these instructions.

BASIC

BASIC (Beginner's All-purpose Symbolic Instruction Code) is a common programming language on many computers. It uses statements that resemble English to perform many complex operations. Some graphics statements may be included in your implementation of BASIC for the CRT, but not necessarily for the plotter.

If you have an HP Series 80 or HP Series 200 computer, graphics statements (sometimes called AGL — A Graphics Language) for the plotter are available as an extension to BASIC. Like other BASIC statements, these statements resemble English, but they translate their functions into HP-GL so that the plotter can understand them. Refer to this chapter's section titled *HP Series 80 and Series 200 Computers — A Note to Users.*

If you do not have an HP Series 80 or HP Series 200 computer, your computer's BASIC graphics statements aren't translated into HP-GL for the plotter. Therefore, you will need to program the plotter directly by using BASIC output commands to send HP-GL strings.

FORTRAN and Pascal

FORTRAN (FORmula TRANslator) and Pascal are high-level programming languages. They generally do not include graphics statements that translate functions into HP-GL. Therefore you must program the plotter by sending strings of HP-GL instructions in a WRITE or WRITELN statement. (Refer to *Examples Presented as HP-GL Strings* in this chapter.)

How to Use the Examples in This Manual

The examples in this manual are designed primarily to show the use of the instruction with which they appear. If you are new to programming, try entering and running some examples on your computer. You might then wish to change some parameters in an instruction and rerun the program. The examples are presented in two ways, either as complete programs or as listings of only the pertinent HP-GL strings. These two types of examples are described in the following paragraphs.

Examples Presented as Complete Programs

Some examples are presented as complete programs, written in a version of Microsoft® BASIC for MS™-DOS operating systems. This BASIC is used by the HP Touchscreen and HP Touchscreen Max, the IBM PC, and several other popular personal computers. (If you are using an RS-232-C interface, be sure you have established the proper handshaking protocol before running these programs. Refer to the Operating Manual and Chapter 13 of this manual.)

Following is a simple example of the way a complete program appears in this manual. The configuration statement in line 10 is for the HP Touchscreen; however, *the statement in the screened area will vary according to the language of your computer.*

```
10  OPEN "O",#1,"PLT:"
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
```

You will always need a configuration statement at the beginning of your program. It will be specific to your computer and the type of interface (RS-232-C or HP-IB) you are using.

If you are using Microsoft® BASIC, you can enter and run these programs as shown; just be sure to have the proper configuration statement for your computer in line 10.

If you are not using Microsoft® BASIC, you will need to insert the proper configuration statement in line 10 *and* may need to change the PRINT #1, statements as well.

The next two sections in this chapter, *Examples of Opening Configuration Statements* and *Examples of Other BASIC Statements*, will show you how to make these changes on some computers. Or, if you have one of the computers listed on the next page, refer to Appendix A for the program listing. Incorporate the screened lines in the beginning of your programs to establish communication between the computer and plotter. The example can serve as a guide when you write your own programs. The computers listed in Appendix A are:

Apple IIe/II Plus	HP The PORTABLE
Apple IIc	HP Touchscreen
Apple III	HP Series 200
AT&T PC 6300	IBM AT
Compaq	IBM PC/XT
DEC Rainbow 100	Olivetti M24
DEC Pro	Sirius 1
HP Series 80	

If your computer is not listed in Appendix A, refer to your computer documentation to learn how to make your computer and plotter communicate.

Examples of Opening Configuration Statements

Following are common forms of a configuration statement that can be used in line 10 of the BASIC programming examples. Refer to your computer documentation for a complete discussion of the parameters.

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)

IBM PC/PC-XT/AT RS-232-C interface

```
10 OPEN  "COM1:9600,S,7,1,RS,CS65535,DS,CD" AS #1
```

AT&T PC 6300 RS-232-C interface

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

Apple IIc RS-232-C and IEEE-488 interfaces

`10 PRINT  CHR$(4);  "PR#n"` (for slot number n)

Apple IIe/II Plus RS-232-C and IEEE-488 interfaces

`10 PR#n` (for slot number n)

Examples of Other BASIC Statements

You might need to change other BASIC statements if the BASIC on your computer is different from the MS™-DOS version used here, or if you are programming in another language. Your computer's documentation should tell you how to do this. Here are the statements that you will likely need to change:

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

Examples Presented as HP-GL Strings

Since input/output instructions (e.g., PRINT # and INPUT #) do vary so much 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 this string of HP-GL instructions is printed in the manual:

```
"SP1;"
```

First use whatever opening statements are required to define the computer's output port and establish the plotter as the recipient of an output string. Then, send the string within a statement similar to one shown here, 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 Series 200, BASIC	`OUTPUT 705; "SP1;"` or `PRINT "SP1;"`
HP Touchscreen (150) Computer or IBM PC/PC-XT/AT, BASIC	`PRINT #1, "SP1;"`
HP 9000, Pascal	`WRITELN(file,'SP1;');`
HP 3000, FORTRAN	`WRITE(6,10)` `10 FORMAT(X,4HSP1;)`

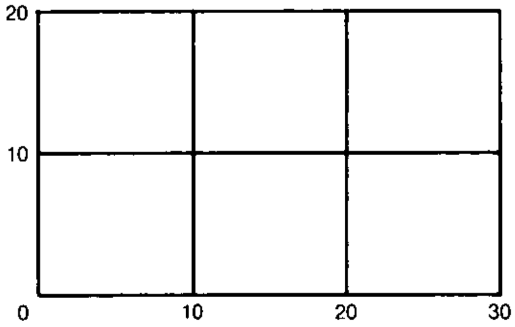
HP Series 80 and HP Series 200 Computers — A Note to Users

If you use BASIC on an HP Series 80* or HP Series 200 computer (formerly models HP 9816, HP 9826 and HP 9836), you can use high-level BASIC graphics statements to program your plotter. These computers actually encode, or translate, the BASIC graphics statements into HP-GL instructions before sending them on to the plotter. For this reason, you might find it easier to use these graphics statements for much of your plotting.

*Series 80 computers must contain a plotter ROM: Part No. 00085-15002 for the HP-85, and Part No. 00087-15002 for the HP-86 and HP-87.

For example, compare the BASIC graphics statements CLIP and GRID with their equivalent HP-GL instructions: PA, PD, TL, XT, and YT. In either case, the plotter draws grid lines every 10 units within a rectangle, as shown. Yet, the CLIP and GRID statements are easier to enter in the computer.

BASIC Graphics	*HP-GL*
`CLIP 0,30,0,20`	`TL100;PA30,0;PD;XT;`
`GRID 10,10,0,0`	`PA20,0;XT;PA10,0;XT;`
	`PA0,0;PA0,10;YT;PA0,20;YT;`

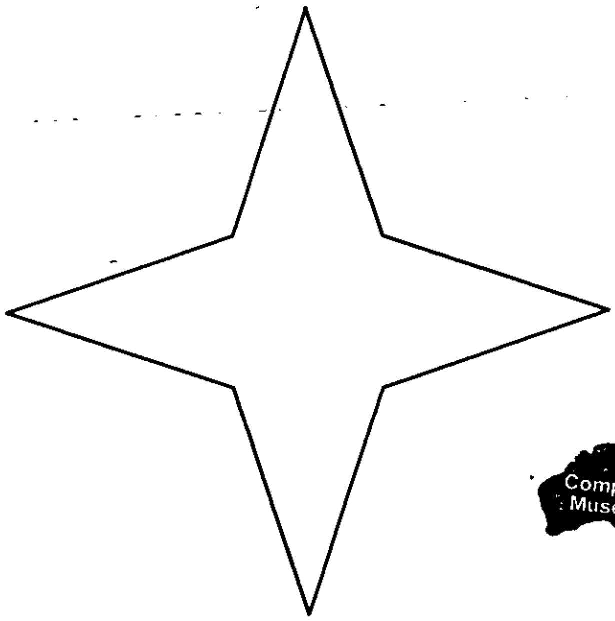


NOTE: This example is not a complete program. It requires additional scaling statements to establish the plotting area and units. These statements were omitted here to help clarify the direct comparison between the BASIC graphics statements and their equivalent instructions. ■

Developing a Plot

With the Standard Plotter

Following is a simple program that draws a star. There are only two BASIC statements that you might have to change for your system — the configuration statement in line 10 and the form of the PRINT #1 statement. You may want to run this program to see the plotted results.



```
10  REM  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
```

Line 10 is a computer-dependent statement that identifies the plotter as the recipient of the strings of HP-GL instructions.

Line 20 issues the following instructions: IN, the initialize instruction; SP, the select pen instruction; and PA, the plot absolute instruction.

IN establishes most conditions in the plotter to be the same as they are when the plotter is first turned on. It is a good idea to begin a program with IN (or a slightly different version called DF, the default instruction) so that any conditions that might have been previously set in the plotter are cleared.

SP causes the pen holder to take the specified pen from the carousel. Finally, the PA instruction establishes the beginning pen position on the paper.

In line 30, the first pen down instruction, PD, lowers the pen and proceeds to connect the X,Y coordinate locations that form the star. The PD instructions continue in lines 30 through 50. You can use as many or as few programs lines as you wish, depending on the length of the line your computer will accept.

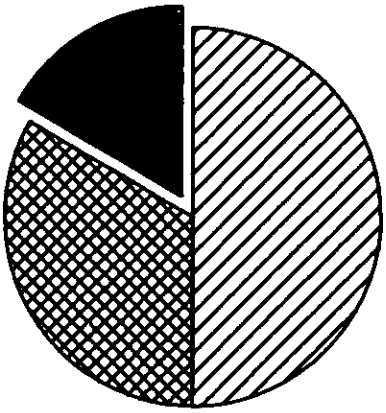
(The IN instruction is presented in Chapter 3; the SP, PA, and PD instructions are in Chapter 4.)

Line 60 uses the SP instruction to return the pen to the carousel.

With the Graphics Enhancement Cartridge

Following is a short program that illustrates a few of the HP-GL instructions available on the graphics enhancement cartridge. It plots a pie chart with three sectors.

As with the previous program, there are only two BASIC statements that you might have to change for your system — the configuration statement in line 10 and the form of the PRINT#1 statement.



```
10   REM  Insert configuration statement here
20   PRINT #1,  "IN;SP2;PA7000,2000;"
30   PRINT #1,  "FT3,75,45;"
40   PRINT #1,  "WG -1000,90,180;"
50   PRINT #1,  "EP;"
60   PRINT #1,  "SP3;PR-60,110;FT1,0,0;"
70   PRINT #1,  "WG-1000,270,60;"
80   PRINT #1,  "EP;"
90   PRINT #1,  "SP4;PA7000,2000;FT4,60,45;"
100  PRINT #1,  "WG-1000,330,120;"
110  PRINT #1,  "EP;"
120  PRINT #1,  "SP0;"
130  END
```

This program is explained fully under the *Fill Wedge Instruction, WG*, in Chapter 7. Here is a brief explanation.

Line 10 is a computer-dependent statement that identifies the plotter as the recipient of the strings of HP-GL instructions.

Line 20 issues the following instructions: IN, the initialize instruction; SP, the select pen instruction; and PA, the plot absolute instruction.

Lines 30 through 110 are known as the "body" of the program; their content will depend on the purpose of the program.

Line 120 uses the SP instruction to return the pen to the carousel.

CHAPTER 2

CHAPTER 2

Fundamental Plotting Concepts

What You'll Learn in This Chapter

In this chapter, you will gain the foundation necessary to understand plotting concepts. For example, in order to use the plotting instructions, you need to be able to specify a location on the paper. This chapter explains:

- the coordinate system
- the plotter's units of measure
- user units and scaling
- graphics limits
- pen status and movement
- absolute and relative coordinates

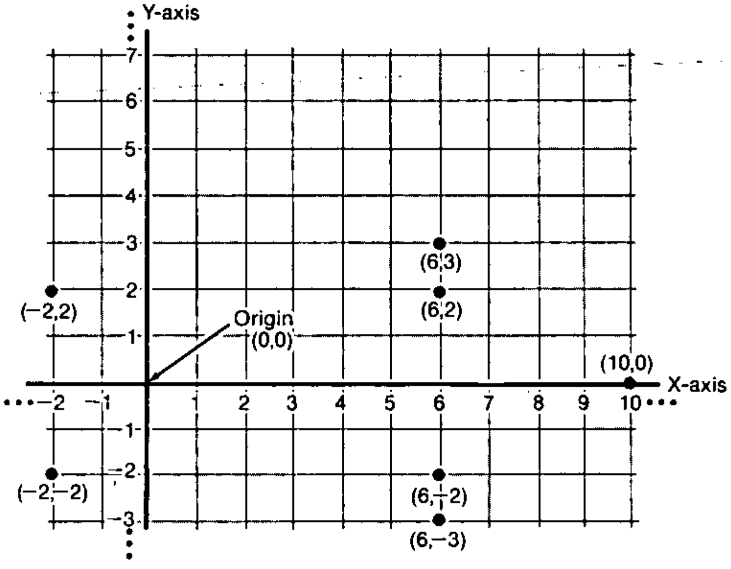
You will find this chapter useful if you have never done any graphics programming or if you have written programs for a graphics terminal but not for a plotter.

Introduction to the Plotter Coordinate System

The plotting area is that portion of the paper in which the plotter's pen can draw. Think of this plotting area as divided into a grid, as shown in the illustration on the next page. Each grid line represents a unit of measurement in one of two directions — horizontal or vertical. The horizontal direction is known as the X-axis; the vertical direction is known as the Y-axis. The intersection of these two axes provides a convenient reference point known as the *origin.*

To locate a position (or point) on the plotting area, you simply tell the plotter how many X-axis units and how many Y-axis units to move away from the origin. These units are known as the *X-coordinate* and the *Y-coordinate*; together, they are an *X, Y coordinate pair.* To specify a position, you must always specify a complete X,Y coordinate pair, with the X-coordinate first and the Y-coordinate second.

The origin is defined as the X,Y coordinate pair (or point) 0,0. If you move to the right of or above the origin, you specify a positive X-coordinate or Y-coordinate, respectively. Likewise, if you move to the left of or below the origin, you specify a negative X-coordinate or Y-coordinate, respectively. Look at the illustration again to locate these points: 0,0;−2,2;6,2;6,3;10,0;6,−3;6,−2;−2,−2;0,0. Now draw a straight line between each point in the order listed. (If you connected the points correctly, you should have drawn an arrow.) This is the way a picture is defined on a two-dimensional coordinate system.



Plotter Coordinate System

Plotting with the Coordinate System

You can express coordinates as two types of units — *plotter units* or *user units*. The plotter always moves in plotter units, which are a constant, fixed size. But you might find that some other unit is more convenient for your application. Any unit that you, the user, define is called a user unit. When you define user units, the plotter *automatically converts* the user units to plotter units so that it can find the correct location on the paper. There is no need for you to perform this conversion. Both plotter units and user units are discussed in detail in the following paragraphs.

Plotter Units

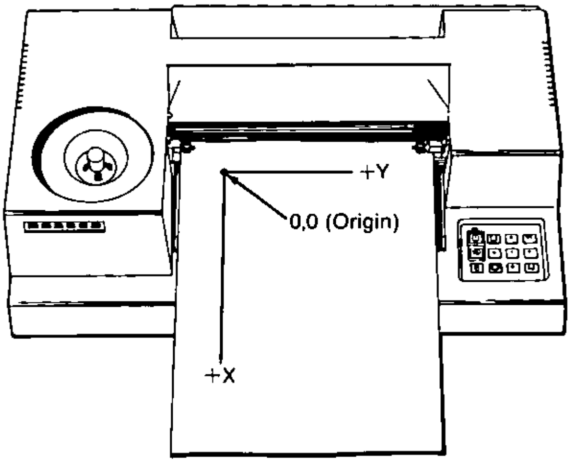
A plotter unit is the smallest move you can tell the plotter to make (this is also known as the addressable resolution of the plotter). On the next page are some measurements pertaining to plotter units.

1 plotter unit = 0.025 mm or 0.00098 in.

40 plotter units = 1 mm

1016 plotter units = 1 in.

When you view paper as it lies in the plotter, the plotter-unit origin (0,0) lies inside the upper-left corner of the paper. The Y-axis extends to the right along the short side of the paper. The X-axis extends downward along the long side of the paper.*



Default Orientation of Plotter Coordinate System

When you are using plotter units, the pen can plot only to the positive X,Y coordinates. However, the plotter understands negative coordinates. In fact, the numerical range of plotter units that the plotter understands is −32 768.0000 to 32 767.9999. For practical purposes, though, the effective range of plotter units is limited to the size of the paper you are using.

User Units

User units are units that you, the user, choose to fit your application. For example, you might want to make a bar chart with total

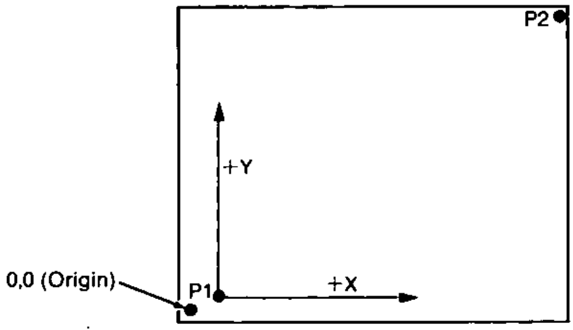
*You can change the orientation of this system with the front-panel buttons or with the rotate instruction, RO. The RO instruction is presented in Chapter 9.

sales in dollars along the Y-axis and the number of months along the X-axis. Using a process known as *scaling,* you can specify your units 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.

User units can represent months, years, dollars, francs, distances, temperatures, population, or whatever meets your requirements. And user units are flexible. You can assign the origin (0,0) to be anywhere on the paper. This means you can specify negative X,Y coordinates (to show, for example, losses on a line chart). You can also specify coordinates with fractions, which makes it easier to directly represent your data. The following section on *scaling* explains users units in more detail.

Scaling

Before we can discuss the process of scaling, you should understand what is meant by the scaling points P1 and P2. The plotter sets these points automatically when you turn it on. From now on, view the paper according to a typical X- and Y-axis orientation (rather than how the paper lies in the plotter); this is how you would usually view a plot. When the paper lies horizontally, P1 defines the lower-left corner of a rectangular area, and P2 defines the upper-right corner. You can see this in the illustration below.



Default Orientation of P1 and P2

P1 and P2 always represent an absolute plotter-unit location on the paper. For both A- and A4-size paper, the default X,Y coordinates for P1 and P2 are listed in the table below.

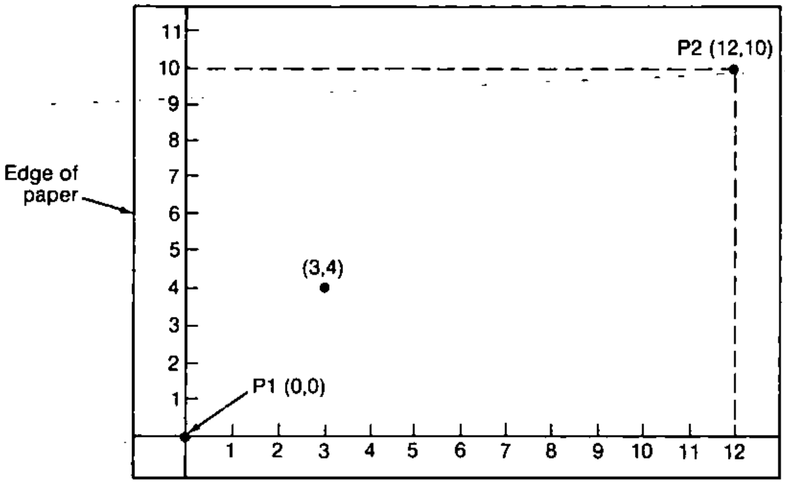
Default P1/P2 Locations
$P1_X = 250$
$P1_Y = 279$
$P2_X = 10\,250$
$P2_Y = 7479$

You can change the plotter-unit locations of P1 and P2 with the input P1 and P2 instruction, IP, or with buttons on the front panel (refer to the Operating Manual).

P1 and P2 are called "scaling points" because they take on the user-unit values that you specify in the scale instruction, SC. For details on the SC instruction, see Chapter 3. The scaling concepts discussed here only require that you know that the SC instruction specifies the minimum and maximum user-unit coordinates.

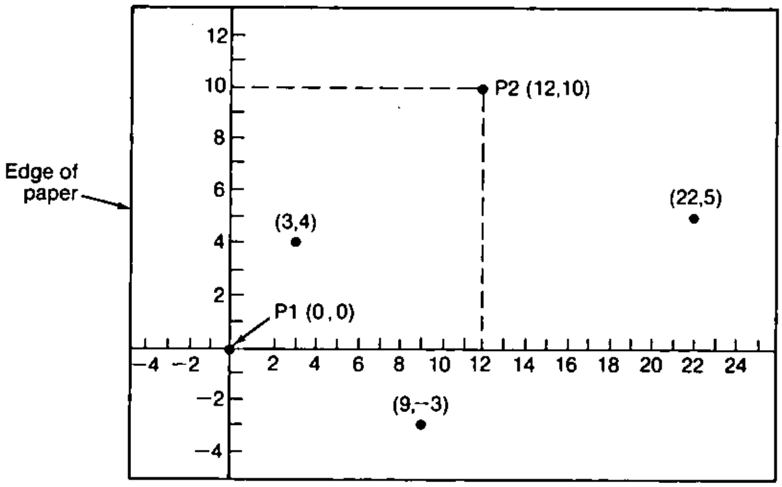
When you issue an SC instruction, P1 takes on the first X,Y coordinate values, and P2 takes on the second X,Y coordinate values. This is sometimes also referred to as mapping; the coordinate values are "mapped onto" P1 and P2. The entire paper is then effectively divided into an imaginary grid based 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.

For example, to divide the Y-axis into 10 units representing thousands of dollars, and the X-axis into 12 units representing months, specify minimum X,Y coordinates of 0,0 and maximum coordinates of 12,10. P1 and P2 then take on these new user-unit values and the entire plotting area is divided into these units. Subsequent plotting instructions will now be interpreted using these units. This means that if you tell the plotter to move to the point 3,4, the plotter will move to the location equivalent to 3,4 user units (*not* 3,4 plotter units).



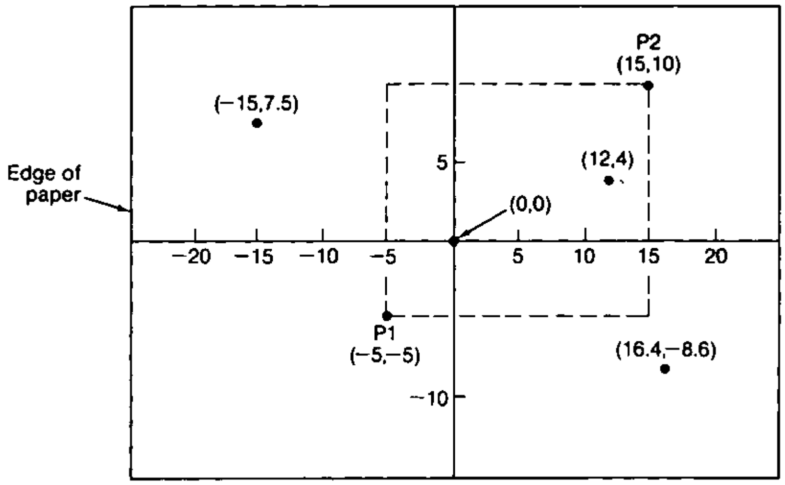
User-Unit Scaling with Default P1 and P2

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



Same User-Unit Scaling with New P1 and P2

To further illustrate the flexibility of user-unit scaling, the following diagram shows P1 and P2 with the user-unit values of −5,−5 and 15,10 respectively. Remember that when user-unit scaling has been established, you can plot anywhere on the plotting area using negative or positive values.



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

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

Graphics Limits

The plotter recognizes two basic types of graphics limits: the *hard-clip limits* and the *soft-clip limits*. ("Clip" indicates that some portion of the paper is being selected. Think of clipping an article out of a newspaper; by clipping away the edges, you select a smaller portion of the newspaper page.)

Hard-clip refers to a physical boundary, the limits beyond which the pen cannot more. *Soft-clip*, on the other hand, refers to a software-controlled boundary that temporarily limits pen movement. When the plotter is first turned on, the soft-clip limits are set equal to the hard-clip limits. Each of these concepts is discussed in more detail in the following paragraphs.

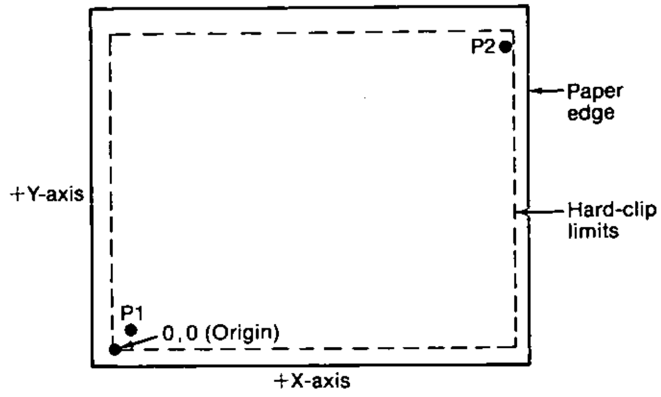
Hard-Clip Limits

The "hard-clip" limits represent the physical boundary for pen movement. When the plotter reads the paper-size switch, it automatically sets the hard-clip limits within a few millimetres of the paper-drive wheels. This allows room for the wheels to move without rolling over plotted lines and possibly smearing ink. The hard-clip limits correspond to the maximum plotting range. The maximum plotting ranges for standard paper sizes are listed in the following table.

Paper Size	**Maximum Plotting Range (In Plotter Units)**	
	X-axis	**Y-axis**
A4 (210 × 297 mm)	0 - 10 900 (272.5 mm/10.68 in.)	0 - 7650 (191.25 mm/7.5 in.)
A (8.5 × 11 in.)	0 - 10 300 (257.5 mm/10.09 in.)	0 - 7650 (191.25 mm/7.5 in.)

NOTE: If you use paper which exceeds 11 in. or 297 mm in length, the plotter still will plot within the indicated limits. ■

The illustration on the next page shows the relationship between the axis orientation, the P1/P2 scaling points, and the hard-clip limits.



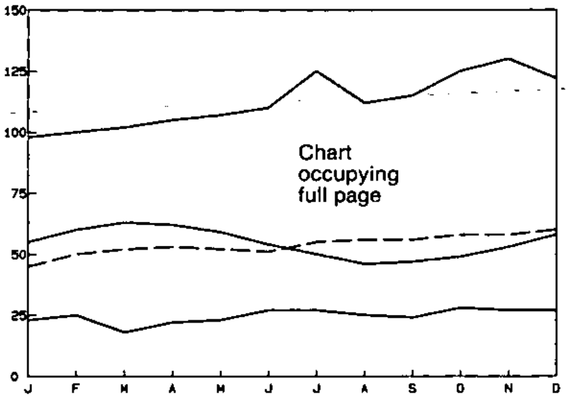
Hard-Clip Limits

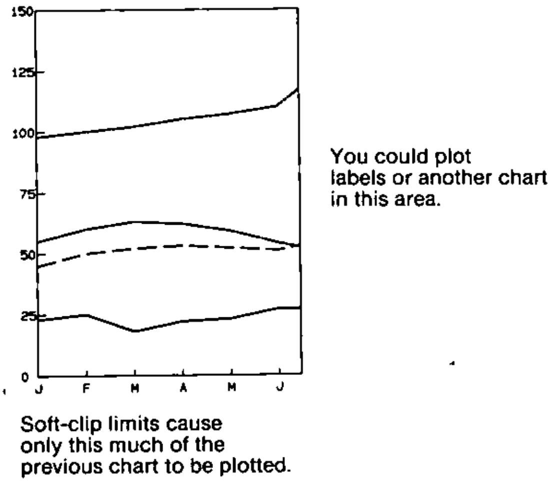
NOTE: You cannot change the hard-clip limits using HP-GL. However, some BASIC graphics statements implemented on HP computers do change the *effective* hard-clip limits. The PLOTTER IS statement, for example, reads the P1/P2 scaling points and sets the effective hard-clip limits to these points, because it does not allow plotting outside this area. In this case hard-clip limits relate to how the computer defines them, not how the plotter defines them. For more information, refer to your computer's documentation on its BASIC graphics statements. ■

Soft-Clip Limits

As previously mentioned, soft-clip limits temporarily restrict pen movement to a specified area of the page. Usually, you will use soft-clip limits to define a portion of the page when you want assurance that nothing will be drawn beyond that portion.

For example, look at the line chart on the next page. After plotting the full chart, suppose that you decide to make a new plot showing only the first 6 months of data and using the space on the right for text. To do this, use the program for the full line chart, and add soft-clip limits to restrict plotting to the left half. Then, add lines to the end of the program with new soft-clip limits and instructions for the labels.





Using Soft-Clip Limits to Change the Plotting Area

Soft-clip limits are often referred to as *windows*. (When you look at the clipped area, it is almost like seeing part of the chart through a window.) In HP-GL, use the input window instruction, IW, to define the lower-left and upper-right boundaries of a window.

Soft-clip limits can be used in conjunction with user-unit scaling and the scaling points P1 and P2 to provide a great deal of flexibility with respect to where you plot data, how much of your data is plotted, and reducing/enlarging plots. The specifics are beyond the scope of this chapter. For more information, refer to Chapter 9.

Pen Movement

Before learning the individual HP-GL instructions, you should understand how the plotter finds the correct positions on the paper. Then you'll be ready to put the instructions to the best use for your application.

Current Pen Status and Pen Position

When you first turn on the plotter, it initializes itself and establishes the default P1/P2 position. It also sets other pre-determined conditions known as *defaults*. Among these defaults are the *pen status* and the *pen position*. You can find a full list of default conditions under the *Default Instruction, DF*, and the *Initialization Instruction, IN*, in Chapter 3.

The pen status refers to whether or not a pen is loaded in the holder, and whether the pen is up or down. At power on, the plotter assumes there is no pen in the holder, which is up. Before the plotter will actually draw lines, you must

1. Select a pen with an SP instruction (or PEN SELECT button).
2. Put the pen down with a PD instruction (or PEN DOWN button).

The pen remains down for subsequent plotting instructions until you issue a pen up instruction, PU, or lift the pen with the PEN UP button. Also, the pen remains in the holder until you select a new pen or return the pen to the carousel.*

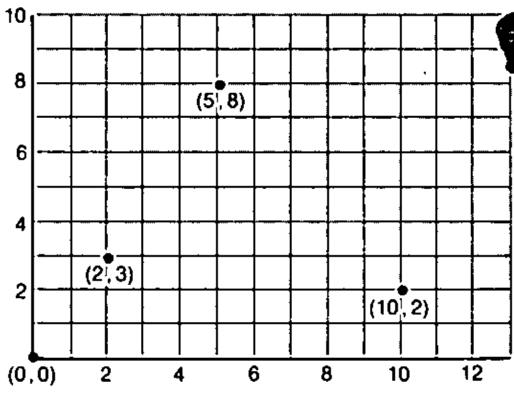
Every time you change pens or the up/down position, the pen status information is updated in the plotter. If you plot a program and portions of the plot aren't drawn, check your program to be sure you have positioned the pen down before the affected instruction(s). The descriptions of each plotting instruction tell you what happens to the current pen status (i.e., whether it is updated at the end of the instruction, or whether it is restored to the status that existed before the instruction was issued.)

*The plotter automatically lifts the pen if it remains inactive for 5 seconds.

The plotter keeps track of the current pen position in a similar manner. At power-on, the plotter sets the current pen position (in plotter units) to 215,7650. Unless you specify a different position, any plotting instruction will begin at this location. Whenever you issue a plotting instruction, the pen begins at its current position and moves to the new location. Usually, the plotter then updates the "current pen position" with the new location.

Absolute and Relative Movement

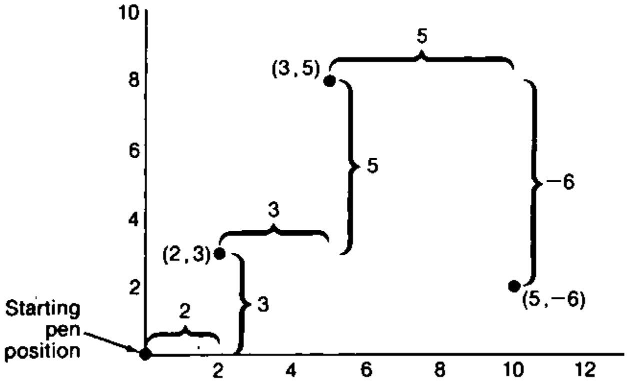
You know that the plotter interprets the pen position according to a coordinate system, and that you can specify coordinates using plotter units or user units. However, so far all discussions have assumed that you are specifying coordinates in *absolute* terms. Absolute means that the coordinates have an absolute fixed position on the coordinate system with respect to the origin (0,0). In the following illustration, the coordinates 2,3;5,8; and 10,2 are always in the same place, regardless of where the pen is when the coordinates are issued.



Absolute Coordinates

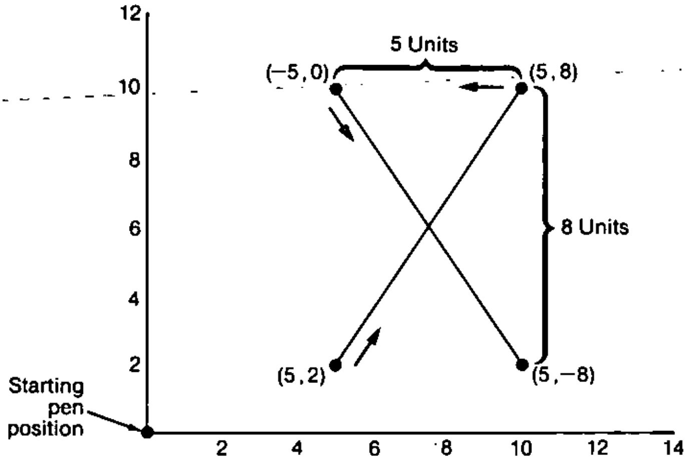
You can also specify coordinates whose location will be determined relative to the current pen position; these are known as *relative* coordinates. Relative coordinates are more accurately called increments, because their values represent the number of units the pen moves from its current pen position. (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 the lower-left point marked in the previous illustration, count 2 units to the right and 3 units above the current pen position. This is the relative position 2,3. Now count the X-units and the Y-units from this position to the upper point. You are now at the relative position 3,5. Continue to the lower-right point, arriving at the relative position 5,−6. The new coordinates are shown below.



Relative Coordinates

The 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 the letter X with a width of 5 units and a height of 8 units, and the current pen position is at the origin. Here is the sequence of relative coordinates to draw the X: 5,2;5,8; −5,0; 5,−8.



Relative Coordinates for Plotting X

NOTE: Relative coordinates (increments) add to the current pen position monitored by the plotter. The plotter automatically converts the new relative position to its absolute coordinates and updates the current monitored position in absolute coordinates. ■

CHAPTER 3

CHAPTER 3

Preliminary Setup Using HP-GL

What You'll Learn in This Chapter

This chapter presents the fundamentals you will need to know in order to program the plotter successfully. It also introduces you to four HP-GL instructions. You will learn about:

- HP-GL syntax
- plotter initialization
- default conditions
- scaling the plotting area into user units

HP-GL Instructions Covered

DF	Default Instruction
IN	Initialize Instruction
IP	Input P1 and P2 Instruction
SC	Scale Instruction

Terms You Should Understand

If you are unfamiliar with any of these terms, refer to the glossary at the end of the manual.

Default	Scaling
Initialize	Scaling Points
Literal String	Separator
Mnemonic	Syntax
Parameter	Terminator
Plotter Units	User Units

The Plotter's HP-GL Instruction Set

A glance at the following tables will give you an idea of the range of functions provided by HP-GL on both the standard plotter and the graphics enhancement cartridge. Chapter references are provided so that you can find an instruction quickly. The following sections then explain HP-GL syntax, and the use of parameters. Appendix D also contains a summary of each instruction's function and syntax.

HP-GL Instruction Set

Instruction	Description	Chapter
CA	Designate alternate character set	6
CP	Character plot	6
CS	Designate standard character set	6
DC	Digitize clear	11
DF	Default	3
DI	Absolute direction	6
DP	Digitize point	11
DR	Relative direction	6
DT	Define label terminator	6
IM	Input mask	10
IN	Initialize	3
IP	Input P1 and P2	3
IW	Input window	9

(Table continues)

HP-GL Instruction Set		
Instruction	**Description**	**Chapter**
LB	Label	6
LT	Line type	5
OA	Output actual position and pen status	10
OC	Ouput commanded position and pen status	10
OD	Output digitized point and pens status	11
OE	Output error	10
OF	Output factors	10
OH	Output hard-clip limits	9
OI	Output identification	10
OO	Output options	10
OP	Output P1 and P2	9
OS	Output status	10
OW	Output window	9
PA	Plot absolute	4
PD	Pen down	4
PR	Plot relative	4
PU	Pen up	4
RO	Rotate coordinate system	9
SA	Select alternate character set	6
SC	Scale	3
SI	Absolute character size	6
SL	Character slant	6
SM	Symbol mode	5
SP	Select pen	4
SR	Relative character size	6
SS	Select standard character set	6
TL	Tick length	5
UC	User-defined character	6
VS	Velocity select	4
XT	X-axis tick	5
YT	Y-axis tick	5

Expanded HP-GL Instruction Set with the HP Graphics Enhancement Cartridge		
Instruction	**Description**	**Chapter**
AA	Arc absolute	7
AR	Arc relative	7
CI	Circle	7
EA	Edge rectangle absolute	7
EP	Edge polygon	7
ER	Edge rectangle relative	7
EW	Edge wedge	7
FP	Fill polygon	7
FT	Fill type	7
PM	Polygon mode	7
PT	Select pen thickness	7
RA	Fill rectangle absolute	7
RR	Fill rectangle relative	7
WG	Fill wedge	7

HP-GL Syntax

An HP-GL instruction begins with a two-letter mnemonic, which may be upper- or lowercase. If parameters are required following the mnemonic, they must be separated from each other by at least one *comma* or *space*, or by a + or − sign which may be preceded by commas or spaces. In its basic form, an HP-GL instruction looks like this.

```
PA 500,500;
```

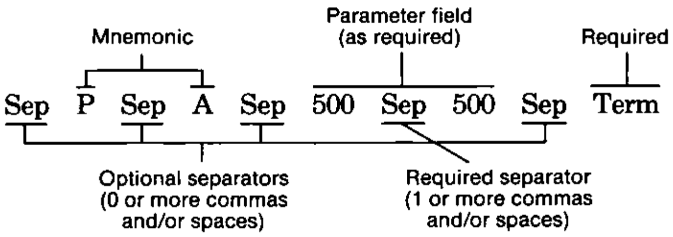
In both the HP-IB and RS-232-C interface configurations, HP-GL instructions are terminated by a semicolon or the first letter of the next mnemonic. In the HP-IB configuration, HP-GL instructions are also terminated by a line feed (**LF**).

The label instruction, LB, is a special case: it must be terminated with the label terminator character. This character defaults to the ASCII end-of-text character, **ETX** (decimal code 3), but may be changed from its default value using the define terminator instruction, DT.

A summary of this information appears in the following table.

Required Parameter Separator	Optional Separator	Required Terminator	
		HP-IB	RS-232-C
One or more commas and/or spaces, or a + or − sign	Zero or more commas and/or spaces	; or LF or the next mnemonic	; or the next mnemonic

Only the terminator and the separators between parameters are required. Other separators may be inserted, as shown in this diagram.


Mnemonic
Parameter field (as required)
Required
Sep P Sep A Sep 500 Sep 500 Sep Term
Optional separators (0 or more commas and/or spaces)
Required separator (1 or more commas and/or spaces)

HP-GL Syntax Requirements

The syntax examples in this manual do not add separators between the letters of the mnemonic. Also, parameters are separated by commas instead of spaces. (Commas are recommended because some computers eliminate spaces, especially when sending variables. Commas are also necessary for complete backwards compatibility with other HP plotters. See *Compatibility with Syntax of Other HP-GL Plotters* in this chapter.)

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

```
PD;PU10,20;          PD PU 10 20;          PDPU10,20;
    |
Recommended
```

Omitting Optional Parameters

Some instructions have optional parameters which, when omitted, assume a default value. In order to omit a parameter, **all subsequent parameters in the same instruction must be omitted.** (The only exception is the pen control parameter in the user-defined character instruction, UC.)

For example, the fill type instruction, FT, has three optional parameters. If you include all three, one way to send FT to the plotter is:

```
FT 3,100,45;
```

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

`FT3;` **(not** `FT 3,,45;` where 45 is interpreted as the second parameter)

Parameter Formats and Ranges

The parameter fields must be specifed in the format defined by each respective HP-GL instruction (as shown in the syntax table accompanying each instruction's description throughout this manual.) The format can be of three types:

1. **Integer Format** — An integer from **−32 768.0000 to 32 767.9999.** Decimal portions of parameters which must be integers are truncated.
2. **Real Format** — A number that falls within two ranges: either **−32 768.0000 to 32 767.9999** or the subset **−128.0000 to 127.9999.** The decimal point may be omitted when no decimal fraction is specified. If no sign is specified, the parameter is assumed to be positive.

NOTE: *When scaling is on*, coordinates are interpreted as user units. Any fractional part of a parameter up through the fourth digit is used.

When scaling is *off*, coordinates are interpreted as plotter units. The fractional portion of the parameter is truncated. However, the plotter still *monitors* the fractional portion. Refer to the *Plot Absolute and Plot Relative Instructions, PA and PR*, in Chapter 4. ■

3. **Label Fields** — Any sequence of characters. Refer to the *Label Instruction, LB*, (Chapter 6) for a complete description.

The term "current units" in a parameters table indicates that you can express parameters of that HP-GL instruction in either plotter units or user units, depending on whether scaling is off or on.

Notations Used in This Manual for Expressing Syntax

The syntax shown under the description for each HP-GL instruction uses the following notations:

MNemonic	— For readability, the mnemonic is shown uppercase and separated from the parameters and/or terminator.
required parameter	— All typeset items are required parameters.
( )	— All items in parentheses are optional.
c . . . c	— Any number of labeling characters.
(,. . .)	— Any number of the specified parameter (there must be an even number of X,Y coordinates).
term	— Required terminator. A semicolon or the next mnemonic are valid terminators. For HP-IB configurations, a line feed (LF) is also valid.
[TERM]	— The output terminator. For HP-IB configurations, all output responses include a carriage return and line feed [CR LF] as a terminator. For RS-232-C configurations, all

output responses include a terminator as defined by the set output mode instruction, ESC . M (refer to Chapter 13). The default output terminator is a carriage return [**CR**].

NOTE: The output terminator, denoted by [TERM] is sent from the plotter to the computer at the end of a response to an output instruction. This differs from the HP-GL terminator, denoted by *term*, which indicates the end of an HP-GL instruction sent to the plotter from the computer. You will learn more about output instructions and output terminators in Chapter 10. ■

Compatibility with Syntax of Other HP-GL Plotters

The syntax rules implemented on the HP ColorPro plotter are extremely flexible and backwards compatible with the HP 7470, HP 7475, HP 7550, HP 7580, HP 7585, and HP 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 HP ColorPro and all of the above plotters allow a comma or space as a separator. *However, only the HP ColorPro, HP 7470 and HP 7475 additionally allow a* + or − sign as a separator.

Also note that the flexible syntax of all of the above plotters will not be able to run earlier plotters such as the HP 9872. If you wish to write software that will run earlier HP plotters, you should use the more rigorous syntax of the HP 9872, which is shown next.

$$\underbrace{\text{XX}}_{\text{Instruction}}\ \text{Parameters}\ \underbrace{\text{(,Parameter)}}_{\substack{\text{Optional}\\ \text{parameter}}}\ \underbrace{\text{[; or LF]}}_{\text{Terminator}}$$

The HP 9872 plotter ignores spaces. It does not allow commas between the characters of the mnemonic; only one comma must separate parameters, and only a semicolon or **LF** may be used as the terminator. In addition, parameters requiring integer format may not contain a decimal point or a decimal fraction.

Default Instruction, DF

USES: The DF instruction sets certain plotter functions to predefined default conditions. Use this instruction to return the plotter to a known state while maintaining the locations of P1 and P2 and any settings from the front panel. When DF is used, unwanted graphics parameters such as character size, slant, or scaling are not inherited from another program.

SYNTAX: *DF term*

EXPLANATION: No parameters are used. The plotter conditions affected by the DF instruction are listed in the following table.

Default Conditions

Function	Equivalent Instructions	Conditions
Plotting mode	PA;	Absolute
Line type	LT;	Solid line
Line pattern length	LT;	4% of the diagonal distance between P1 and P2
Input window	IW;	Set to current hard-clip limits
Relative character direction	DR1,0;	Horizontal
Relative character size	SR;	Width=0.75% of $(P2_X - P1_X)$ Height=1.5% of $(P2_Y - P1_Y)$
Standard character set	CS0;	Set 0
Alternate character set	CA0;	Set 0
Character set selected	SS;	Standard
Character slant	SL0;	0 degrees
Label Terminator	DT**ETX**;	**ETX** (ASCII decimal code 3)
Symbol mode	SM;	Off

(Table continues)

Default Conditions (Continued)

Function	Equivalent Instructions	Conditions
Tick length	TL;	tp = tn = 0.5% or $\lvert P2_X - P1_X \rvert$ for Y-tick and 0.5% of $\lvert P2_Y - P1_Y \rvert$ for X-tick
Mask value	IM 223,0,0	Recognizes all defined errors except 6
Digitize clear	DC;	Off
Scale	SC;	User-unit scaling off
Pen velocity	VS;	40 cm/s (15.7 in./s)
Chord angle	—	5 degrees
Fill type	FT;	Type 1, bidirectional solid fill
Fill spacing	FT;	1% of diagonal distance between P1 and P2
Fill angle	FT;	0 degrees
Pen thickness	PT;	0.3 mm
Polygon mode	PM0,PM2;	Polygon buffer cleared

When DF is executed, the carriage-return point for labeling instructions is updated to the current pen position.

Absolute character size defined with the SI instruction defaults to the same as *SR;*.

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

- locations of P1 and P2
- current pen, its position, and up/down status
- 90-degree rotation
- generated errors (not cleared)

The following table summarizes the error conditions or unexpected results that might occur with the DF instruction.

Condition	Error	Plotter Response
parameters used	2	executes instruction

Initialize Instruction, IN

USES: The IN instruction returns the plotter to its 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 or from the front panel. In this manual, program examples begin with *IN;* to clear unwanted conditions from the previous program. *This instruction has no effect on the polygon buffer size and handshake protocol in any RS-232-C environment.*

SYNTAX: *IN term*

EXPLANATION: No parameters are used. The IN instruction sets the plotter to the same conditions as the DF instruction, and sets these *additional* conditions:

- raises the pen
- sets P1 and P2 to default conditions
- cancels 90-degree rotation
- sets bit position 3 of the status word to 1 (to indicate the plotter has been initialized)
- clears any HP-GL or RS-232-C error condition

The following table summarizes the error conditions or unexpected results that might occur with the IN instruction.

Condition	Error	Plotter Response
parameters used	2	executes instruction

Input P1 and P2 Instruction, IP

USES: The IP instruction allows you to establish new or default locations for the scaling points P1 and P2. P1 and P2 are used by the scale instruction, SC, to establish user-unit scaling. The IP instruction is often used to ensure that a plot is always the same size, regardless of how P1 and P2 might have been set from the front panel. This instruction can also be used in advanced techniques such as plotting mirror images, enlarging/reducing plots, and enlarging/reducing relative character size or direction (refer to Chapters 6 and 9).

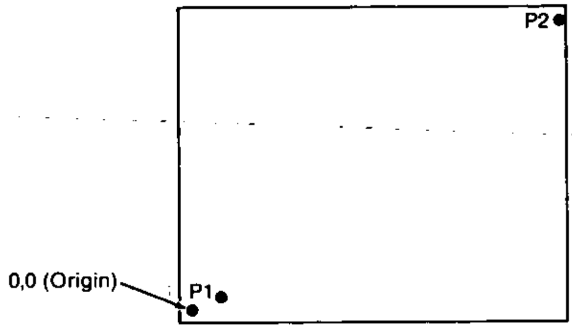
SYNTAX: *IP* $P1_X$, $P1_Y$ (, $P2_X$, $P2_Y$) *term*
or
IP *term*

Parameter	Format	Range	Default
X,Y coordinates	integer	−32 768 to 32 767 plotter units	$P1_X = 250$ $P1_Y = 279$ $P2_X = 10250$ $P2_Y = 7479$

EXPLANATION: Issuing IP without parameters (*IP;*) sets P1 and P2 to the default positions, adjusted by any current axis rotation.

Specifying the P2 X,Y coordinates is optional. However, if they are not specified, then P2 tracks P1 and its coordinates change so that the X- and Y-distances between P1 and P2 do not change. P2 could become located outside the plotting area when this happens. This tracking function can be useful for preparing more than one equal-sized plot on one page. For an example, refer to *Techniques for Manipulating the Plotting Area* in Chapter 9.

The usual orientation of P1 and P2 is for P1 to define the lower-left corner of a rectangular area and for P2 to define the upper-right corner.



It is possible to specify coordinates such that P1 and P2 define different diagonal corners (e.g., P1 is lower-right and P2 is upper-left). If so, and scaling is on, the plot will be reversed and/or turned upside down (a mirror image). These possibilities are discussed in Chapter 6.

NOTE: If an IP instruction is executed without parameters after the axes have been rotated 90 degrees by the *RO 90;* instruction, the default locations of P1 and P2 are changed to reflect the rotation. The X,Y coordinates for each scaling point are reversed. For example, *IP;* after an axis rotation sets P1 to be 279,250 and P2 to be 7479,10 250. ■

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.

The Scale Instruction, SC, (presented next in this chapter) shows examples of changing the locations of P1 and P2. Many other examples throughout this manual also use IP to change P1 and P2.

The following table summarizes the error conditions or unexpected results that might occur with the IP instruction.

Condition	Error	Plotter Response
no parameters	none	establishes default P1 and P2, adjusted by any rotation
1 or 3 parameters	2	ignores instruction
more than 4 parameters	2	uses first 4 parameters
parameter out-of-range	3	ignores instruction
P2 tracking P1 puts $P2_X$ or $P2_Y$ out-of-range	3	sets parameter to ±32 767

Scale Instruction, SC

USES: The SC instruction establishes a user-unit coordinate system by mapping values onto the scaling points P1 and P2. Thus, you can plot in user units convenient to your application. For a discussion of scaling, refer to *User Units* and *Scaling* in Chapter 2.

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

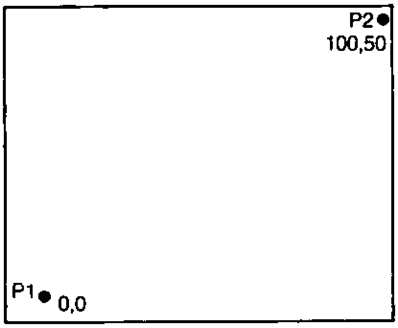
Parameter	Format	Range	Default
X- and Y-ranges	integer	−32 768 to 32 767 user units	none

EXPLANATION: When you use parameters, you must specify all four of them. The first two parameters are the X-axis range, and the last two parameters are the Y-axis range. For example,

$$\text{SC } \underbrace{0,100}_{\text{X-axis range}},\underbrace{0,50}_{\text{Y-axis range}};$$

The first and third parameters (X_{min} and Y_{min}) are the coordinate pair that is mapped onto P1; the second and fourth parameters

(X_{max} and Y_{max}) are the coordinate pair that is mapped onto P2. (This is different from the IP instruction, where the parameters are expressed as X,Y coordinate pairs rather than as ranges.)



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

After an SC instruction has been executed, only those plotting instructions that can be issued in "current units" (see the parameters table for each instruction) are interpreted as user units. All instructions that can only be issued in plotter units are still interpreted as plotter units.

Following are the instructions that can be interpreted as user units.

HP standard plotter	PA, PU, PR, PD
with cartridge	AA, AR, CI, EA, ER, EW, RA RR, WG, and FT (spacing)

Also note that, when scaling is on, decimal parameters of plotting locations are *no longer truncated;* thus, the point 6.5 , 9.5 is distinct from 6.6 , 9.8.

An SC instruction without parameters (*SC;*) turns off user-unit scaling; all subsequent plotting instructions will then be interpreted in plotter units.

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

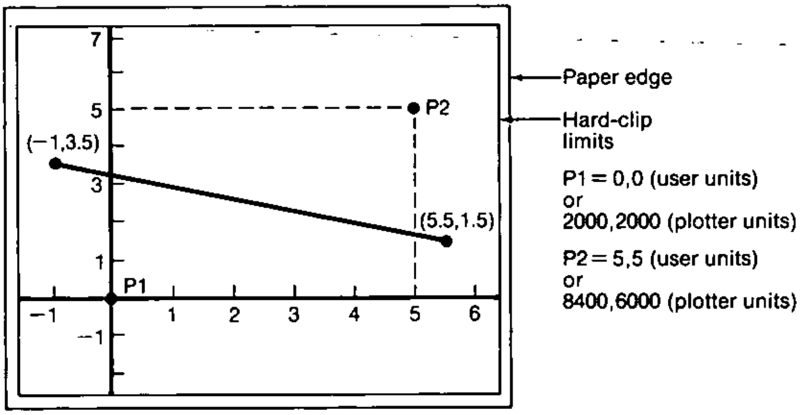
The following table summarizes the error conditions or unexpected results that might occur with the SC instruction.

Condition	Error	Plotter Response
no parameters	none	turns scaling off
more than 4 parameters	2	uses first 4 parameters
less than 4 parameters	2	ignores instruction
$X_{min} = X_{max}$ or $Y_{min} = Y_{max}$	3	turns scaling off
parameter out-of-range	3	ignores instruction

Examples — Establishing Scaling

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

```
"IP 2000,2000,8400,6000;"
"SC 0,5,0,5;"
```



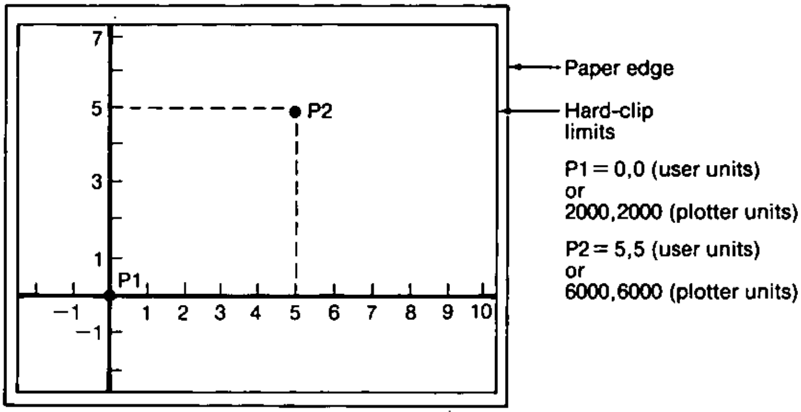
7
5
(−1,3.5)
3
1
−1
P1
P2
(5.5,1.5)
−1
1
2
3
4
5
6
Paper edge
Hard-clip limits
P1 = 0,0 (user units) or 2000,2000 (plotter units)
P2 = 5,5 (user units) or 8400,6000 (plotter units)

User Units Extending beyond the P1, P2 Range

Another thing to consider when establishing user-unit scaling is whether your plots will include geometric or actual shapes. If so, you will need to establish *equal*, or *isotropic*, scaling. Isotropic means the spacing of units along the X-axis is equal to the spacing along the Y-axis. When the scaling is isotropic, shapes such as circles will be plotted accurately without distortion. In the previous illustration, the scaling was *anisotropic*, or *unequal* along the X- and Y-axes. If you wanted to plot a circle using that scaling, it would turn out as an ellipse (egg-shaped) instead of a circle.

The easiest way to establish isotropic scaling is to set P1 and P2 so they define a square area. Then assign a scale that places the same number of units along the X-axis as along the Y-axis. This is shown in the figure on the next page.

```
"IP 2000,2000,6000,6000;"
"SC 0,5,0,5;"
```

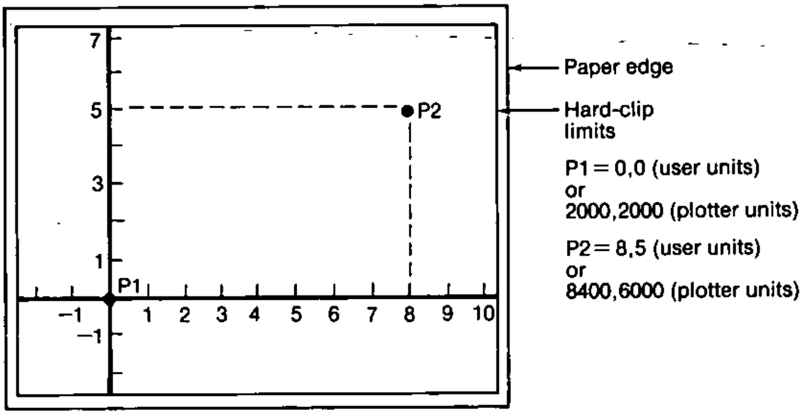


Paper edge
Hard-clip limits
P1 = 0,0 (user units) or 2000,2000 (plotter units)
P2 = 5,5 (user units) or 6000,6000 (plotter units)
P1
P2

Isotropic Scaling: Method 1

Another method for establishing isotropic scaling is to specify your scale so that each user unit equals the same number of plotter units in each direction. Using the P1/P2 locations that were set up in the first example, notice that the Y-axis distance from P1 to P2 is 4000 plotter units, or 5 user units. This means that each user unit is 800 plotter units (4000/5 = 800). To achieve this proportion along the X-axis, which is 6400 plotter units from P1 to P2, you must specify 8 user units (6400/800 = 8), as shown below. Compare the figure on the next page with the last one. Notice that both have the same number of units within the hard-clip limits even though P1 and P2 are in different locations.

```
"IP 2000,2000,8400,6000;"
"SC 0,8,0,5;"
```



7
5
3
1
-1
P1
P2
-1
1
2
3
4
5
6
7
8
9
10
Paper edge
Hard-clip limits
P1 = 0,0 (user units) or 2000,2000 (plotter units)
P2 = 8,5 (user units) or 8400,6000 (plotter units)

Isotropic Scaling: Method 2

CHAPTER 4

CHAPTER 4

Pen Control and Plotting

What You'll Learn in This Chapter

Now that you understand the unit systems in which data can be represented, you are ready to create plots. In this chapter, you will learn how to:

- select or change pens
- raise and lower the pen
- plot lines in absolute and relative mode
- change pen speed
- send variables as part of a plotting instruction

HP-GL Instructions Covered

SP	Select Pen Instruction
PU	Pen Up Instruction
PD	Pen Down Instruction
PA	Plot Absolute Instruction
PR	Plot Relative Instruction
VS	Velocity Select Instruction

Terms You Should Understand

If you are unfamiliar with any of these terms, refer to the glossary at the end of the manual.

Absolute Plotting
Constant
Current Units
Point
Relative Plotting
Variable

Select Pen Instruction, SP

USES: The SP instruction 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. Then, use SP at the end of a program to return the pen to the carousel.

SYNTAX: *SP* pen number *term*
or
SP *term*

Parameter	Format	Range	Default
pen number	integer	0 to 8	none

EXPLANATION: The pen number parameter corresponds to the pen numbers marked on the carousel. The pen number is interpreted as follows.

1. **Pen Number = 1 to 8.** Pen numbers 1 through 8 cause the appropriate pen to be placed in the pen holder. If there is currently a pen in the holder, this pen is stored before the new pen is selected. After selecting a new pen, the pen holder returns to the current graphics position.

2. **Pen Number = 0 or No Parameter.** 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.

The following table summarizes the error conditions or unexpected results that might occur with the SP instruction.

Condition	Error	Plotter Response
no parameter	none	returns pen to stall, if possible
pen number not in 0 to 8 range	none	ignores instruction

Pen Instructions, PU and PD

USES: The pen up instruction, PU, raises the pen; the pen down instruction, PD, lowers the pen. Use PU or PD with parameters to draw or move to the points specified by the parameters.

SYNTAX: *PU* X,Y (,...) *term*
or
PU *term*

and

PD X,Y (,...) *term*
or
PD *term*

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

*If scaling is on, the range may be smaller.

EXPLANATION: The PU and PD parameters are interpreted as follows.

1. **X- and Y-coordinates.** When parameters are included, PU and PD are interpreted as forms of one of the plot instructions, PA and PR, whichever has been most recently executed. (If no plot instruction has been previously executed, PA is assumed.)

   For example, if you execute *PD X, Y;* and absolute plotting is in effect, the X,Y coordinates are interpreted as absolute coordinates. Similarly, if relative plotting is in effect, *PD X, Y;* causes the plotter to draw to the relative point X,Y. For complete information on how the parameters are interpreted, refer to the *Plot Absolute and Plot Relative Instructions, PA and PR,* located next in this chapter.

2. **No Parameters.** Without parameters, the PU and PD instructions respectively raise and lower 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.

NOTE: The pen will be lowered only if it is within the current graphics limits (window) and is not in the "up" portion of a dashed line pattern. ■

Plot Absolute and Plot Relative Instructions, PA and PR

USES: The PA instruction establishes absolute plotting mode and moves the pen to specified absolute points. The PR instruction establishes relative plotting mode and moves the pen to specified points. You can use either instruction after PU to move to a point with the pen up, or after PD to draw a line to a point with the pen down.

SYNTAX:

```
PA  X,Y (,...)  term
    or
PA  term

and

PR  X,Y (,...)  term
    or
PR  term
```

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

*If scaling is on, the range may be smaller.

EXPLANATION: The PA and PR parameters are interpreted as follows.

1. **X- and Y-coordinates.** For PA, the X,Y coordinates specify the absolute position *relative to the origin* to which the pen will move. For PR, the X,Y coordinates (increments) specify the points *relative to the current pen position* to which the pen will move.

   Both coordinates of each pair must be specified, and you may specify as many coordinate pairs as you wish. (On an HP-IB interface, this is limited by the ability of your controller to output long strings without a line feed.) Refer to *Absolute and Relative Movement* in Chapter 2 for details.

   When you include more than one coordinate pair, the pen moves to each point in the order given, using the current up/down status. The up/down status is based on the most recent PU or PD instruction. This concept is discussed in the next section, titled *Plotting with PA, PU, and PD.*

   When scaling is *on*, coordinates are interpreted as user units. Any fractional portion of a parameter is used.

   When scaling is *off*, coordinates are interpreted as plotter units. For positive numbers, any fractional portion of a parameter is discarded. For example, both 123.4 and 123.8 become 123. For negative numbers (relative plotting only), the fractional portion is discarded and the integer is changed to the next more negative number. For example, both −123.4 and −123.9 become −124.

   NOTE: Sending decimal parameters when scaling is off may cause unexpected results. Although the pen will move only in integer increments, the plotter still monitors the pen position using the complete decimal number. This can affect future pen positions if you specify a series of pen moves.

For example, if you tell the plotter to move to a point 0.6,0.6, the pen will not move (the integer position is 0,0). However, if you then tell the plotter to make an additional relative move 0.7,0.7, the pen will move. The monitored cumulative position is the point 1.3,1.3 (0.6 + 0.7 = 1.3), but the pen will move to the point indicated by the integer portion — 1,1. ■

2. **No Parameters.** The PA instruction without parameters establishes absolute plotting mode for subsequent HP-GL instructions. The PR instruction without parameters establishes relative plotting mode for subequent instructions.

Execution of a PA, IN, or DF instruction will cancel relative plotting.

The following table summarizes the error conditions or unexpected results that might occur with the PA or PR instructions.

Condition	Error	Plotter Response
no parameters	none	establishes absolute plotting (PA) establishes relative plotting (PR)
odd number of coordinates	2	uses each coordinate pair; ignores extra coordinate
parameter out-of-range*	3	ignores wrong coordinate pair and the rest of the instruction

*For relative plotting when scaling is *off:* to be in range, 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) or it is out-of-range.

Plotting with PA, PU, and PD

The pen is up at power-on and after an IN instruction. Therefore, the following string of instructions will cause the pen to move to the point 2000,2000 without drawing a line.

```
"IN; PA 2000,2000;"
```

In order to draw from that point to another point, you must put the pen down. The following instructions draw a line from 2000,2000 to 5000,5000.

```
"IN; PA 2000,2000; PD 5000,5000;"
```

Note that the PD instruction was used to do absolute plotting. This is equivalent to issuing these separate instructions:

```
"IN; PA 2000,2000; PD; PA 5000,5000;"
```

You can also insert PD or PU within a list of X,Y coordinate pairs. When you do this, commas or spaces are required between numeric parameters, but are optional before and after the two-letter mnemonics. In the following example, the designated commas are the optional separators and the semicolon is the required terminator.

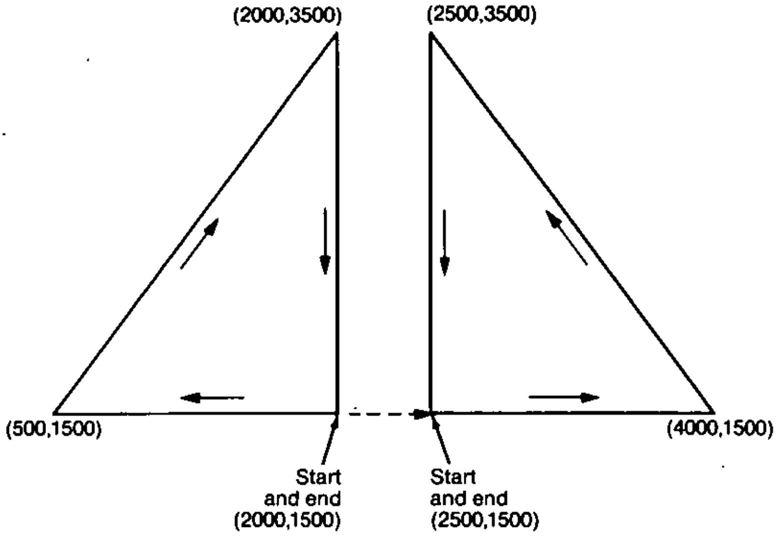
$$\mathrm{PA},X_1,Y_1,\mathrm{PD},X_2,Y_2,X_3,Y_3,\mathrm{PU},X_4,Y_4;$$

Optional

The following program illustrates the concepts of using PA, PD, and PU to draw two triangles. The resulting plot is also shown, along with coördinates and arrows showing pen direction.

Line 10 establishes the interface conditions between the computer and plotter. The PRINT #1 statement sends the strings of HP-GL instructions to the plotter. Change the first line and the PRINT #1 statements as necessary for your computer.

```
10  REM  Insert configuration statement here
20  PRINT #1, "IN;SP1;PA2000,1500;"
30  PRINT #1, "PD 500,1500,2000,3500,2000,1500;"
40  PRINT #1, "PU2500,1500;"
50  PRINT #1, "PD 4000,1500,2500,3500,2500,1500;"
60  PRINT #1, "SP0;"
70  END
```

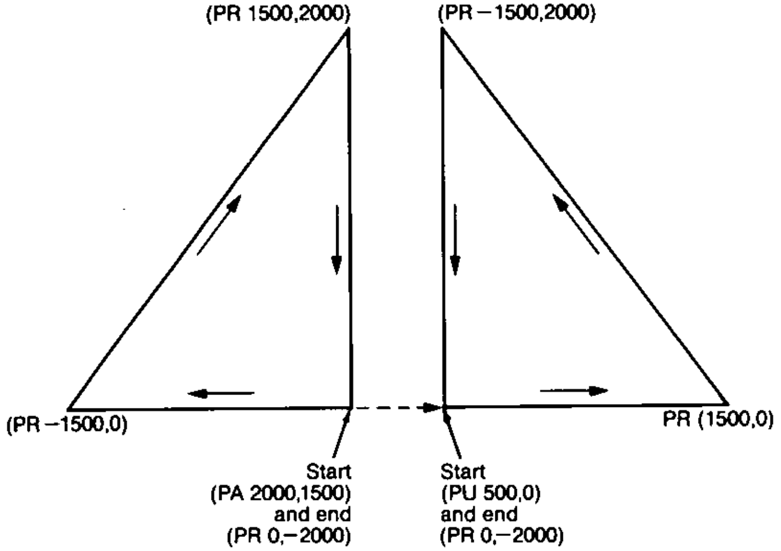


Plotting with PR, PU, and PD

The PR instruction operates with PU and PD in the same manner as described in the previous section. 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 following program illustrates this difference by using PR to draw the same triangles previously drawn with PA. Again, change line 10 and the PRINT #1 statements as necessary for your computer.

```
10  REM  Insert configuration statement here
20  PRINT #1, "IN;SP1;PA2000,1500;"
30  PRINT #1, "PR;PD -1500,0,1500,2000,0,-2000;"
40  PRINT #1, "PU 500,0;"
50  PRINT #1, "PD 1500,0,-1500,2000,0,-2000;"
60  PRINT #1, "SP0;"
70  END
```


(PR 1500,2000)
(PR −1500,2000)
(PR −1500,0)
PR (1500,0)
Start (PA 2000,1500) and end (PR 0,−2000)
Start (PU 500,0) and end (PR 0,−2000)

Velocity Select Instruction, VS

USES: The VS instruction establishes the speed of the pen when it is down. Use the instruction to slow speed to 10 cm/s when plotting on transparency film or glossy paper. Also, reducing the pen speed can produce slightly denser lines on any plotting surface. Pen movement with the pen up is always executed at 52 cm/s.

SYNTAX: *VS* pen speed *term*
or
VS *term*

Parameter	Format	Range	Default
pen speed	real	1 to 40*	40 cm/s (1.2 g acceleration)

*This is the practical range. The allowable range is 0 to 127.9999.

EXPLANATION: The pen speed parameter sets the actual pen-down speed in cm/s. You can set the speed in increments as small as 1.0 cm/s. Use of the instruction with a parameter also changes the acceleration to 0.6 g.

A VS instruction without parameters (*VS*;) establishes the default speed and acceleration.

The following table summarizes the error conditions or unexpected results that might occur with the VS instruction.

Condition	Error	Plotter Response
no parameters	none	uses default values
parameter > 40 and < 127.9999	none	uses default values
more than 1 parameter	2	uses first parameter
parameter > 127.9999	3	ignores instruction

Plotting with Variables

In many plotting applications, you might prefer to use variables for the parameters of an HP-GL instruction. (For example, you can set up a program for drawing a line chart, and then just change the data points each time you plot the chart.) This is simple to do. Just remember these principles:

- The values of all parameters have the same restrictions (integer or real in a valid range) when sent as variables as they do when sent as constants.
- HP-GL mnemonics, their separators, and their terminators all must be sent to the plotter along with the variable parameters.

NOTE: There are several ways that you can send HP-GL instructions to the plotter. Also, specific methods for defining output format vary from computer system to computer system. Those discussed here are specific to Microsoft® BASIC. However, the principles are applicable to any computer. ■

Methods for Sending Variable Parameters

The usual way to send an HP-GL instruction with parameters which are constants is to send the mnemonic, parameters, separators, and terminator as a literal string. (On most computers, literal strings are delimited by quotation marks.) As shown below, this results in a compact field of characters being sent to the plotter, which reduces activity on the interface and saves time.

Instructions: `PRINT #1, "PA10,20,80,90;"`

Characters received: `PA10,20,80,90;`

If you were to send variable parameters in the same manner, as shown next, you would generate an error because the plotter would not recognize the variables. The plotter looks at the characters X and Y and tries to interpret them as an HP-GL mnemonic, but does not recognize them.

Instructions: `PRINT #1, "PA10,20,X,Y;"`

Characters received: `PA10,20,X,Y;` and error 1 set (instruction not recognized)

Following are two acceptable methods of sending variables to the plotter, along with the characters that the plotter receives. There are many acceptable methods that depend on each computer's format and output statements; these two are shown to help you understand how to choose a method. (Assume that the variables have been defined as X = 80 and Y = 90.) Note that the mnemonic and the terminator are always sent as literal strings.

(1) Instructions: `PRINT #1, "PA10,20,",X,",",Y,";"`

Characters received:

```
PA10,20,       80           ,              90
```

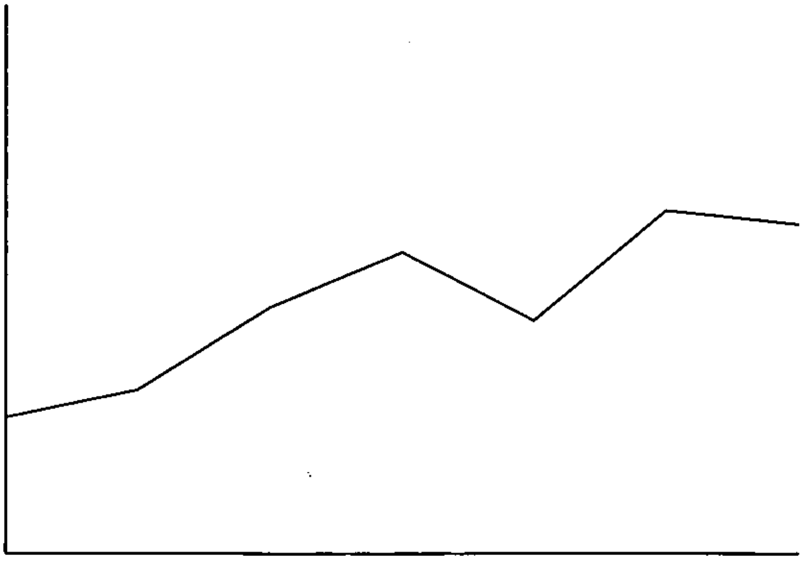
(2) Instructions: `PRINT #1, "PA10,20,";X;",";Y;";"`

Characters received: `PA10,20, 80 , 90 ;`

Methods 1 and 2 are similar, except for the use of commas versus semicolons between the parameters. Here the commas result in fields of 14 characters being sent to the plotter. If the parameters are less than 14 characters, this means unnecessary blanks are sent, resulting in extra activity over the interface bus. The semicolons compress these fields, except for leading and trailing blanks between the numeric variables. Be sure your system sends at least one leading and trailing blank between variables.

Example — Plotting a Line Chart with Variable Parameters

The BASIC program on the next page illustrates method 2 for sending variables as part of an HP-GL instruction. If you want to plot this line chart (reduced below) with different data points, simply substitute new X,Y coordinate pairs in the DATA statements (lines 80 and 90).



```
10    REM  Insert configuration statement here
20    PRINT #1, "IN;SP1;SC-20,80,-10,60;"
30    FOR I=1 TO 7
40     READ X,Y
50     PRINT #1, "PA";X;",";Y;";PD;"
60    NEXT I
70    PRINT #1, "PU 0,40;PD0,0,60,0;SP0;"
80    DATA 0,10,10,12,20,18,30,22,40,17
90    DATA 50,25,60,24
100   END
```

Program Explanation

10 establishes HP-IB or RS-232-C interface conditions

20 initializes the plotter; selects pen 1; assigns user-unit scale to the plotting area

30 starts a loop

40 reads one set of X,Y coordinates from line 80 or 90

50 moves to the point specified by the current X,Y coordinates — the first point is moved to with the pen up, and all subsequent points are moved to with the pen down

60 closes the loop

70 moves with the pen up to the top of the Y-axis; draws the Y-and X-axes with the pen down; returns the pen to the carousel

80 defines the first five X,Y coordinate pairs to be read by line 40

90 defines the next two X,Y coordinate pairs to be read by line 40

CHAPTER 5

CHAPTER 5

Enhancing Your Plots

What You'll Learn in This Chapter

This chapter presents enhancements that make your data easier to interpret. You will learn how to:

- draw tick marks on axes
- create grids
- draw a symbol at each data point
- draw with dashed or dotted line patterns

HP-GL Instructions Covered

XT	X-Tick Instruction
YT	Y-Tick Instruction
TL	Tick Length Instruction
SM	Symbol Mode Instruction
LT	Line Type Instruction

Terms You Should Understand

If you are unfamiliar with any of these terms, refer to the glossary at the end of the manual.

Grid
Tick

Tick Instructions, XT and YT

USES: The XT instruction draws a vertical X-tick at the current pen location. The YT instruction draws a horizontal Y-tick at the current pen location. Use these instructions to draw tick marks on axes, and to draw grids.

SYNTAX: *XT term*
or
YT term

EXPLANATION: Both instructions cause a tick mark to be drawn at the current pen position; XT draws a vertical tick, and YT draws a horizontal tick. They include an automatic pen down feature. After the tick is drawn, the current pen status (up or down) is restored. You can vary the length of the tick mark with the tick length instruction, TL. However, if you do not specify a tick length, defaults are used as follows: 0.5% of $|P2_X - P1_X|$ for YT, and 0.5% of $|P2_Y - P1_Y|$ for XT.

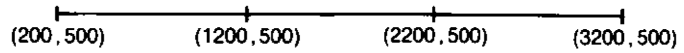
The following table summarizes the error conditions or unexpected results that might occur with the XT and YT instructions.

Condition	Error	Plotter Response
1 or more parameters	2	draws tick

Example — Drawing X-Ticks

The following example draws a horizontal line 3000 plotter units long, placing X-ticks 1000 plotter units apart. Refer to the TL instruction, described next, for another example that draws ticks.

```
"IN;SP2;PA200,500;XT;PD;PR1000,0;XT;"
"PR1000,0;XT;PR1000,0;XT;SP0;"
```



Tick Length Instruction, TL

USES: The TL instruction specifies the length of the tick marks on the X- and Y-axes. Use this instruction to set the lengths of both the positive and negative portions of tick marks, or to draw a grid.

SYNTAX: *TL* tp (,tn) *term*
or
TL *term*

Parameter	Format	Range	Default
tp and tn	real	−128.0000 to 127.9999 percentage	0.5% of $\lvert P2_X - P1_X \rvert$ and of $\lvert P2_Y - P1_Y \rvert$

EXPLANATION: A TL instruction without parameters (*TL ;*) sets default values. Each parameter is specified as a *percentage* of the vertical or horizontal distance between P1 and P2. Parameter values are a percentage of the vertical scale length $\lvert P2_Y - P1_Y \rvert$ when used with the XT instruction. With the YT instruction, parameter values are a percentage of the horizontal scale length $\lvert P2_X - P1_X \rvert$. The parameters are interpreted as follows.

1. **Tp.** The positive tick length, tp, determines the length of the *upward* portion of the tick marks drawn along the X-axis, and the *right-side* portion of the tick marks drawn along the Y-axis.
2. **Tn.** The negative tick length, tn, determines the length of the *downward* portion of the tick marks drawn along the X-axis, and the *left-side* portion of the tick marks drawn along the Y-axis.

Unless the area defined by P1 and P2 is square, the tick lengths on the X- and Y-axes will differ even if the same tick length percentage value is specified for both XT and YT.

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 tp parameter. To suppress the positive portion of the tick, specify 0 (zero) for tp and specify the desired percentage for tn.

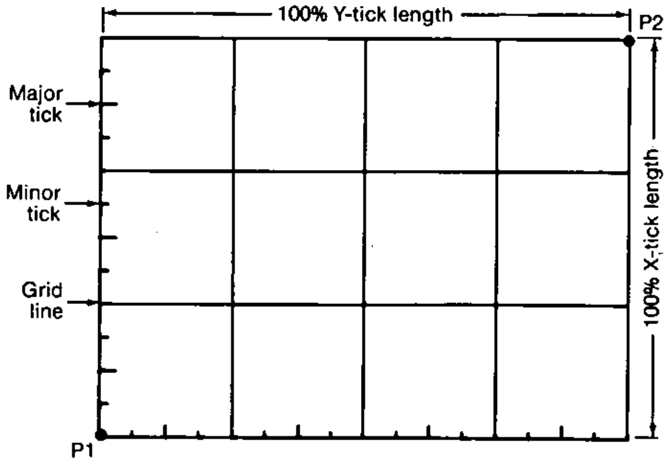
A TL instruction remains in effect until another TL instruction is executed or the plotter is initialized or set to default conditions.

The following table summarizes the error conditions or unexpected results that might occur with the TL instruction.

Condition	Error	Plotter Response
no parameters	none	sets tp and tn to 0.5%
1 parameter	none	draws only positive portion of tick when XT or YT is executed
more than 2 parameters	2	uses first 2 parameters
parameter out-of-range	3	ignores instruction

Example — Drawing Grid Lines, Major Ticks, and Minor Ticks

The following program illustrates how to draw both grid lines and major and minor ticks along the Y- and X-axes.



```
10   REM  Insert configuration statement here
20   PRINT #1,  "IN;IP200,1000,3000,3100;"
30   PRINT #1,  "SC 0,40,0,30;SP2;"
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;PR 0,-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;PR 2.5,0;TL1.5;XT;"
120   PRINT #1, "PR2.5,0;TL3;XT;PR 2.5,0;"
130   PRINT #1, "TL1.5,0;XT;PR2.5,0;"
140  NEXT J
150  PRINT #1,  "SP0;"
160  END
```

Program Explanation

10 establishes HP-IB or RS-232-C interface conditions

20 initializes the plotter; establishes scaling points P1 and P2

30 assigns user-unit scale; selects pen 2

40 sets the starting pen position; frames the P1/P2 area

50 starts a loop to repeat lines 60–80 three times in order to draw horizontal ticks and grid lines in the entire P1/P2 area

60 places pen down; draws to the first tick position; sets a tick length of 1.5%; draws a minor Y-tick; draws to the next tick position

70 sets a tick length of 3%; draws the major Y-tick; draws to the next tick position; sets a tick length of 1.5%; draws a minor Y-tick

80 draws to the next tick position; sets a tick length of 100%; draws the grid line

90 closes the loop

100	starts a loop that repeats lines 110-130 four times to draw vertical ticks and grid lines in the entire P1/P2 area
110-130	similar to lines 50-70, except that X-ticks are drawn
150	returns the pen to the carousel

Symbol Mode Instruction, SM

USES: Use the SM instruction with the PA and PR instructions to draw a symbol at each X,Y coordinate. By drawing a specified character at each data point, you can differentiate data contained in scattergrams, geometric drawings, or multiple-line graphs.

SYNTAX: SM character *term*
or
SM term

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

*The semicolon (decimal code 59) is an HP-GL terminator and cannot be used as a symbol. Use it only to cancel symbol mode (*SM ;*).

EXPLANATION: After an SM instruction has been executed, subsequent PA and PR instructions function as described in Chapter 4, except that the symbol mode character is drawn at each X,Y coordinate, centered over the point. The SM instruction includes an automatic pen down feature. After the symbol is drawn, the current pen status (up or down) is restored.

The character is drawn in the character set selected at the time the SM instruction is executed. The character does not change even if a new character set is subsequently selected. Also, the size (SI and SR), slant (SL), and direction (DI and DR) instructions affect how the character is drawn. (Refer to Chapter 6.)

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

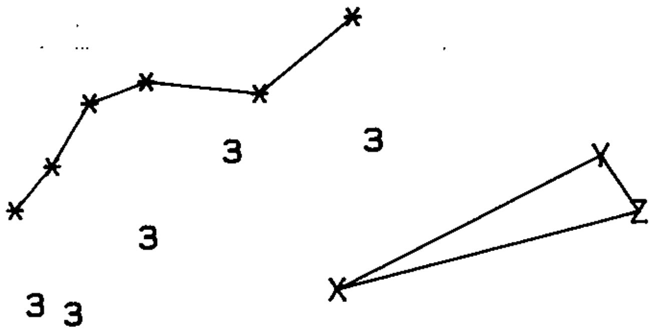
The following table summarizes the error conditions or unexpected results that might occur with the SM instruction.

Condition	Error	Plotter Response
no parameter, or parameter is a control character	none	cancels symbol mode

Example — Plotting in Symbol Mode

The following instructions plot 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.

```
"IN;SP1;SM*;PA200,1000;"
"PD400,1230,600,1560,900,1670,1500,1600,2000,2000;"
"PU;SM;PA100,300;SM3;"
"PA300,500,500,450,900,850,1350,1300,2100,1350;PU;"
"SM;PA1900,560;PD;SMY;PA3300,1250;"
"SMZ;PA3500,950;SMX;PA1900,560;PU;SP0;"
```



Line Type Instruction, LT

USES: The LT instruction specifies the line pattern that will be used with PA, PD, and PR instructions.* Use the LT instruction to differentiate between curves on multiple-line graphs. Also use LT to emphasize or de-emphasize other plotted lines and shapes.

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

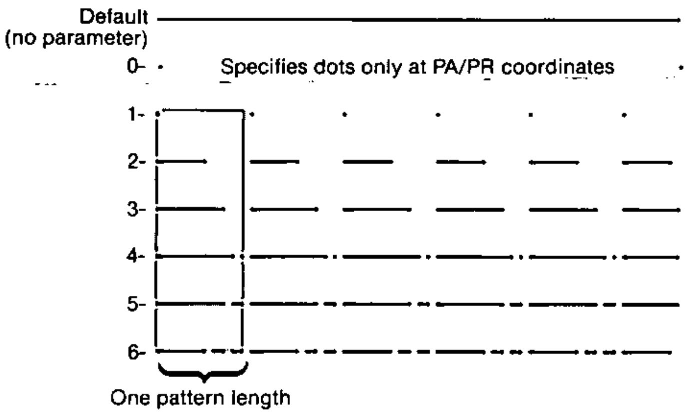
Parameter	Format	Range	Default
pattern number	real	0 to 6**	solid line
pattern length	real	0 to 100**	4% of the diagonal distance between P1 and P2

**This is a practical range. The allowable range is −128.0000 to 127.9999.

EXPLANATION: An LT instruction without parameters (*LT;*) draws a solid line. The pattern number and pattern length parameters are interpreted as follows.

1. **Pattern Number.** The pattern numbers and the line patterns they each generate are shown on the next page. The pattern number uses the actual specified pattern length (see the shaded portion) to draw vectors.

*With a cartridge, the LT instruction specifies the line pattern used with the AA, AR, CI, and EP instructions, and, if not solid fill, the FP, WG, RA, and RR instructions. The FT instruction also uses the current line type for designing area-fill patterns.



Line Types

2. **Pattern Length.** The optional pattern length parameter specifies the length of one complete pattern and is expressed as a percentage of the diagonal distance between the scaling points P1 and P2. Any portion of a pattern that is not used to draw the specified vector is carried over to the next vector (unless a PU is issued). If you do not specify a pattern length, the plotter uses the last value issued.

Once selected, the line type remains in effect until another valid LT instruction is executed or the plotter is initialized or set to default conditions.

NOTE: If a vector ends in the pen up portion of a positive line pattern, a pen down instruction, PD, will not lower the pen until the next vector instruction is executed. Also, the pen up instruction, PU, clears any carry-over portion of a pattern segment; in other words, the pattern for the next vector will begin at the beginning of the new line. ■

The following table summarizes the error conditions or unexpected results that might occur with the LT instruction.

Condition	Error	Plotter Response
no parameters	none	defaults to solid line type
1 parameter	none	establishes specified line type with a 4% pattern length
more than 2 parameters	2	uses first 2 parameters
parameter out-of-range	0, 2, or 3	usually ignores instruction; response varies with different out-of-range values

Notes

CHAPTER 6

CHAPTER 6

Labeling Fundamentals

What You'll Learn in This Chapter

This chapter presents all the labeling instructions that enable you to annotate plots or make text charts. You will learn how to:

- position labels
- change the label size, slant, and direction
- use an alternate character set
- define your own characters

HP-GL Instructions Covered

LB	Label Instruction
DT	Define Terminator Instruction
SI	Absolute Character Size Instruction
SR	Relative Character Size Instruction
SL	Character Slant Instruction
DI	Absolute Direction Instruction
DR	Relative Direction Instruction
CP	Character Plot Instruction
CS	Designate Standard Character Set Instruction
CA	Designate Alternate Character Set Instruction
SS	Select Standard Character Set Instruction
SA	Select Alternate Character Set Instruction
UC	User-Defined Character Instruction

Terms You Should Understand

If you are unfamiliar with any of these terms, refer to the glossary at the end of the manual.

Carriage-Return Point
Character Origin
Character Plot Cell
Character Set
Current Units
Label Terminator
Line
Space

Label Instruction, LB

USES: The LB instruction plots text using the currently defined character set. Use this instruction to annotate drawings or create text-only charts.

SYNTAX: *LB* c . . . c *term*

(where *term* is ETX (decimal code 3) or the label terminator defined by the DT instruction)

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

EXPLANATION: Following is an example of the LB instruction and the label it creates.

```
PRINT #1, "LBThis is a label."+CHR$(3)
           ┬                   ┬
    HP-GL mnemonic      Label terminator
```

```
This is a label.
```

As you can see, plotting text is simple. The plotter automatically uses the default character set and default character size, slant, and direction, unless you specify otherwise with HP-GL instructions.*

*Refer to the DT, DI, DR, SI, SR, and SL instructions in this chapter for defaults and methods of changing these conditions.

The label begins at the current pen position. After the label is drawn, the pen position is updated to be the next character origin (refer to *Character Positioning* later in this discussion).

Characters

All printing characters following the LB instruction are drawn using the currently defined character set. (The method for selecting character sets other than the default character set 0, ANSI ASCII English, is described in this chapter in *Plotter Character Sets*.

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

Label Terminators

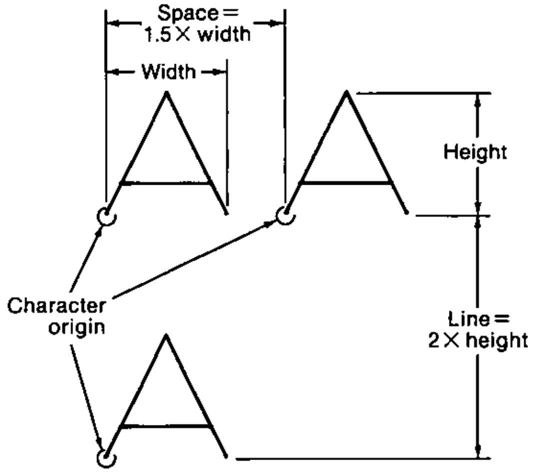
To terminate the LB instruction, you *must* use a special label terminator instead of the usual HP-GL terminator. The terminator tells the plotter to exit from the label mode. (If you don't use the label terminator, everything following the LB mnemonic will be printed in the label, including other HP-GL instructions.) The default label terminator is the nonprinting end-of-text character ETX (decimal code 3). This is accessed by the CHR$(3) in BASIC. (Refer to *How to Send the Label Terminator and Other Nonprinting Characters* later in this section.) If you wish, you can define a different terminator with the DT instruction.

The following table summarizes the error conditions or unexpected results that might occur with the LB instruction.

Condition	Error	Plotter Response
label terminator invalid or omitted	none	labels all characters (including HP-GL mnemonics and parameters) until valid label terminator is encountered

Character Positioning

The following illustration defines the relationships of a space, a line, and a character's origin, width, and height. The plotter assumes a default physical size for all of these dimensions, unless you change them with the SI and SR instructions.

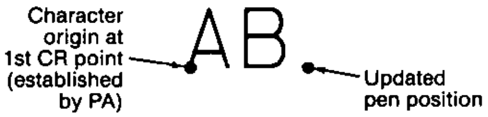


Relationship of a Character's Dimensions

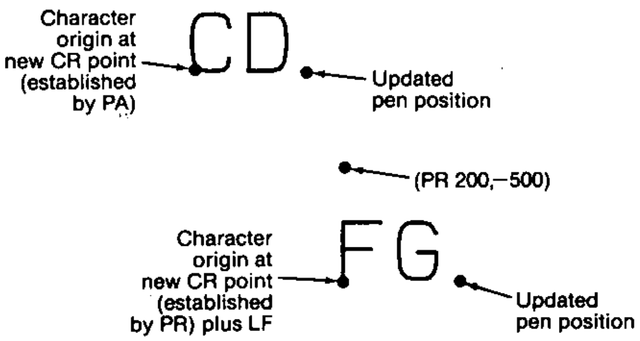
When you issue an LB instruction, the current pen position becomes the first character origin. After the first character is drawn, the pen moves to the next character origin and draws the next character (if any). This continues until the end of the label string, unless an embedded control character such as a carriage return (**CR**) or line feed (**LF**) is encountered. If a line feed is encountered, the pen moves down one line from its current position. If a carriage return is encountered, the pen moves to the carriage-return point (usually the current pen position when the LB instruction was executed, adjusted up or down by any line feeds or inverse line feeds). At the end of the string, the pen position is updated to what would be the character origin for the next character. These concepts are shown in the following illustration.

NOTE: The carriage-return point is updated to be the current pen position after these instructions are executed: PA, PR, DI, DR, DF, IN, and RO.* 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 pen position. Note that although the current pen position is updated after an LB instruction, the carriage-return point is *not* updated, except as described above. ■

```
"PA 3010,4000;"
"LBAB"+CHR$(3)
```



```
"PA 3010,2000;"
"LBCD"+CHR$(3)
"PR 200,-500;"
"LB"+CHR$(13)+CHR$(10)+"FG"+CHR$(3)
```


Character origin at new CR point (established by PA)
CD
Updated pen position
(PR 200,-500)
Character origin at new CR point (established by PR) plus LF
FG
Updated pen position

Carriage-Return Points and Updated Pen Positions

* also AA and AR with the graphics enhancement cartridge

How to Send the Label Terminator and Other Nonprinting Characters

You can usually send nonprinting characters such as **LF**, **CR** and **ETX** in one of two ways: using a character string function such as CHR$ (provided by BASIC), or producing a character directly from the keyboard. Each method is described below.

1. On many computers, including the HP Touchscreen (150), you will have to use a string function such as CHR$(*value*), which produces the character whose decimal code value is specified. For example,

CHR$(10)	produces	**LF**	(line feed)
CHR$(13)	produces	**CR**	(carriage return)
CHR$(3)	produces	**ETX**	(end-of-text, default label terminator)
CHR$(14)	produces	**SO***	(shift-out)
CHR$(15)	produces	**SI***	(shift-in)

   Check the ASCII character and keyboard table in your computer documentation for the proper value to use. (You can also refer to the programs for individual computers in Chapter 3 of the Operating Manual; they use string functions for labeling.)

   When you use a function such as CHR$, you will have to separate it from the label string, as shown on the next page. If necessary, substitute your language's string function for CHR$. In addition, notice that, on the HP Touchscreen (150), the concatenation (linking) symbol + is used between the label string and the string function. This ensures that the character produced by the string function is sent immediately after the label string without any extra spaces. On some computers, & or ; is used as the linking symbol; if this is true for your computer, substitute & or ; for +.

*Shift-out (decimal code 14) and shift-in (decimal code 15) can be used to shift between a standard and alternate character set. See *Labeling with Standard and Alternate Character Sets* in this chapter.

2. On many HP computers, such as the HP Series 80 and HP Series 200, you can produce these characters by simultaneously striking the keys indicated below:

CTRL and **J**	produces	**LF**	(line feed)
CTRL and **M**	produces	**CR**	(carriage return)
CTRL and **C**	produces	**ETX**	(terminate label)
CTRL and **N**	produces	**SO**	(shift-out)
CTRL and **O**	produces	**SI**	(shift-in)

   If you produce the character directly from the keyboard in this way, you can simply enter the character within the label string. Again, check your computer documentation for the proper keys to use.

Instruction Using String Function

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

Equivalent Instruction Using Keyboard

```
"LBLabel␃"
      or
"LBLabel␍␊return␃"
```

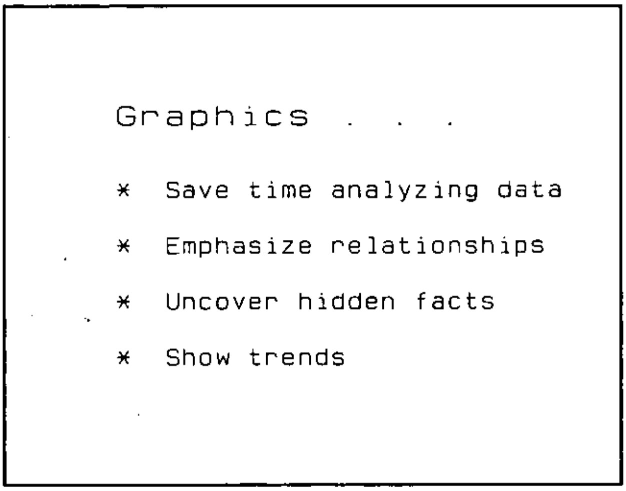
Whichever method you use, here is the plotted output:

```
Label      or      Label
                   return
```

Example — Creating a Text Chart

One of the simplest and most common graphics applications is the text chart. Following is a reduced example of a text chart and the program which created it.

```
10   REM  Insert configuration statement here
20   PRINT #1,  "IN;SP1;PU1500,6100;"
30   PRINT #1,  "SR3,4.5;LBGraphics . . . "+CHR$(3)
40   PRINT #1,  "SR2,3.8;PU1500,4800;"
50   PRINT #1,  "LB*  Save time analyzing data"+CHR$(3)
60   PRINT #1,  "PU1500,3800;"
70   PRINT #1,  "LB*  Emphasize relationships"+CHR$(3)
80   PRINT #1,  "PU1500,2800;"
90   PRINT #1,  "LB*  Uncover hidden facts"+CHR$(3)
100  PRINT #1,  "PU1500,1800;"
110  PRINT #1,  "LB*  Show trends"+CHR$(3)+"SP0;"
120  END
```



Program Explanation

10 establishes HP-IB or RS-232-C interface conditions

20 initializes the plotter; selects pen 1; positions the pen at the starting point of the chart

30 establishes a character size relative to the distance between P1 and P2 (see *Relative Character Size Instruction, SR*, in this chapter); draws the first label

40 establishes a smaller character size; positions the pen at the start of the next label

50 draws the label

The remainder of the program continues the sequence of positioning the pen and drawing the next label.

Define Label Terminator Instruction, DT

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

SYNTAX: *DT* label terminator *term*

Parameter	Format	Range	Default
label terminator	label	any character except **NULL** (decimal code 0)	**ETX** (decimal code 3)

EXPLANATION: The character following the DT mnemonic is interpreted to be 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.
nonprinting character (control character)	terminates the label instruction and performs the function specified by the character.

NOTE: A DT instruction with no parameter (*DT;*) does *not* establish **ETX** as the default terminator, since the character immediately following *DT* is taken as the parameter. Only a DF or IN instruction or use of **ETX** as the instruction's parameter will reestablish **ETX** as the label terminator. ■

The following table summarizes the error conditions or unexpected results that might occur with the DT instruction.

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

Example — Changing Label Terminators

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

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   REM  Insert configuration statement here
20   PRINT #1, "IN;SP1;SC0,5000,0,5000;PA0,4500;"
30   PRINT #1, "LBDefault control "+CHR$(13)+
                CHR$(10)+CHR$(3)
40   PRINT #1, "LBcharacter ETX "+CHR$(13)+
                CHR$(10)+CHR$(3)
50   PRINT #1, "LBterminates with end-of-text
                function"+CHR$(3)
60   PRINT #1, "PA0,3900;DT#;"
70   PRINT #1, "LBPrinting characters"+CHR$(13)+
                CHR$(10)+"#"
80   PRINT #1, "LBterminate, but are also printed.#"
90   PRINT #1, "PA0,3400;DT"+CHR$(13)+";"
100  PRINT #1, "LBControl characters terminate"+
                CHR$(10)+CHR$(13)
110  PRINT #1, "LBand perform their function."+
                CHR$(13)+"SP0;"
120  END
```

```
Default control
character ETX
terminates with end-of-text function

Printing characters
#terminate, but are also printed.#

Control characters terminate
and perform their function.
```

Labeling with Variables

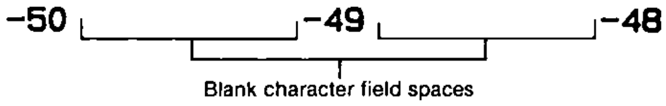
You may want to label the plot using variables (rather than literals) to define a label string. The principles for sending variables in label instructions are similar to those for sending variables in plotting instructions (refer to Chapter 4). However, the *format* for the character field is important because labels will be 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 will be printed. Three conventions are:

quotation marks	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 numeric variable to be right-justified in a specific character-field width.
semicolon	used by most computers between variables for close spacing of numeric variables in labels.

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.

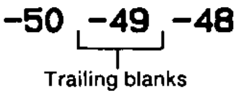
The following example illustrates the use of the comma to establish fixed spacing when using variables for labeling. The second statement causes the plotter to label the values of X, X+1, and X+2. The number of blank character-field spaces and the printing of a sign vary with different computers.

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



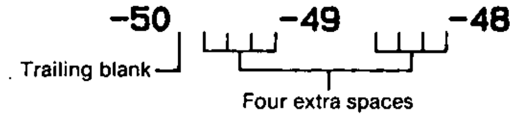
The following example illustrates the closer spacing achieved in Microsoft® BASIC when semicolons separate variables. A semicolon omits the extra blank spaces, but usually leaves space for a sign and a trailing blank.

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



If you require a specific number of spaces in a label field, enclose them within quotation marks. The following example puts four extra spaces between each of the integers. Note that four spaces enclosed in quotation marks are sent between each variable, but that the semicolon suppresses extra blank spaces.

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



Adjusting Character Size, Spacing, and Position

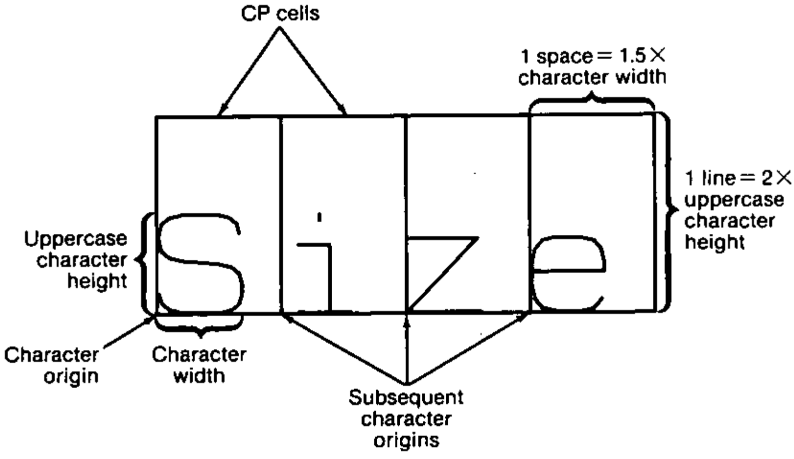
You may want to learn how to adjust conditions such as character size, slant, direction, space, and position. Most of these conditions are based on the concept of the "character plot cell," which is a rectangular grid used to define characters generated by the LB, SM, and UC instructions. The character plot cell is described in this section, followed by a summary of how various character conditions relate to this concept. The instructions for changing character conditions are presented in the remainder of this chapter.

The Character Plot Cell

You can think of the character plot (CP) cell as a rectangular area around a character that includes space above and to the right of the character, as shown in the illustration below.

The character origin is always in the lower-left corner of the CP cell. The height of the CP cell is always 2 times the height of an uppercase character. The width of the CP cell is 1.5 times the width of a character.

The height and width of the CP cell are equivalent to one line and one space on the plotter.


CP cells
1 space = 1.5× character width
1 line = 2× uppercase character height
Uppercase character height
Size
Character origin
Character width
Subsequent character origins

Character Plot Cell

Character Size

The absolute character size (SI) and relative character size (SR) instructions define the width and height of the characters. The character size determines the size of the CP cell, as described previously. Thus, whenever you change character size, you automatically change the CP cell size.

Direction and Slant

The absolute direction (DI) and relative direction (DR) instructions define the orientation of the label on the page. For example, you can label horizontally (parallel to the X-axis), or you can label at any other angle. You can also change the slant of the characters with the character slant (SL) instruction to create an italicized effect. The direction and slant do not affect the size of the CP cell.

Spacing and Position

Default spacing is as described previously under *The Character Plot Cell.*

You can control the character position in a number of ways. Since labels are drawn at the current pen position, you can arrive at that position with a plotting instruction such as PA or PR. Or, you can use the character plot (CP) instruction to move the pen position in terms of the CP cell. This is useful for aligning labels. For example, when indenting to the third character of a previous label, you can use the CP instruction to move two CP cells, rather than measuring that distance in current units.

Absolute Character Size Instruction, SI

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

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

Parameter	Format	Range	Default
width	real	−128.0000 to 127.9999	0.187 cm
height	real	−128.0000 to 127.9999	0.269 cm

EXPLANATION: An SI instruction without parameters (*SI;*) sets the character size to default values. When used, the parameters specify the actual width and height of the character. The size is absolute, and is not affected by changes in P1 or P2.

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

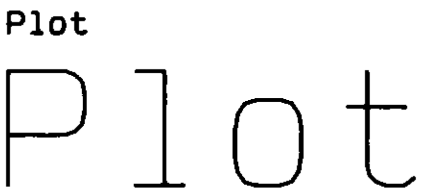
The following table summarizes the error conditions or unexpected results that might occur with the SI instruction.

Condition	Error	Plotter Response
no parameters	none	establishes default values
1 parameter	2	ignores instruction
more than 2 parameters	2	uses first 2 parameters
parameter out-of-range	3	ignores instruction

Example — Changing Absolute Character Size

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

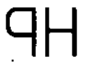
```
"IN;SP1;PA300,5000;"
"SI;LBPlot"+CHR$(3)
"PA300,4200;"
"SI 1,1.5;LBPlot"+CHR$(3)+"SP0;"
```



Example — Using Negative Parameters to Mirror Labels

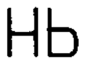
Negative parameters produce mirror images of labels. A negative width parameter mirrors labels in the right-to-left direction.

```
"SI-.35,.6;LBHP"+CHR$(3)
```



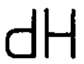
A negative height parameter mirrors labels in the top-to-bottom direction.

```
"SI.35,-.6;LBHP"+CHR$(3)
```



Two negative parameters mirror labels in both directions, causing the label to appear to be rotated 180 degrees.

```
"SI-.35,-.6;LBHP"+CHR$(3)
```



NOTE: The interactions of the SI, SR, DI, and DR instructions are complex when using negative parameters. Refer to *Parameter Interaction in Labeling Instructions* in this chapter. ■

Relative Character Size Instruction, SR

USES: The SR instruction specifies the relative size of characters as a percentage of the distance between scaling points P1 and P2. This means that character sizes are adjusted proportionally when P1 or P2 change positions. At power-on, the plotter assumes character sizes are relative. Use this instruction to change the percentage values for relative character sizes, or to reestablish relative character sizes after an absolute character size instruction, SI, has been executed.

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

Parameter	Format	Range	Default
width	real	−128.0000 to 127.9999	0.75% of $\|P2_X - P1_X\|$
height	real	−128.0000 to 127.9999	1.5% of $\|P2_Y - P1_Y\|$

EXPLANATION: An SR instruction without parameters (*SR;*) sets the character size to default values. If P1 and P2 are at their default locations, this produces the same size characters (in centimetres) as listed previously under the SI instruction. However, with the SR instruction, the character size adjusts (expands or contracts) with subsequent changes in the locations of P1 and P2.

The character size you specify with SR is a *percentage* of the distance in plotter units between the X,Y coordinates of P1 and P2. The plotter calculates the actual character width and height from the specified parameters as follows:

actual character width = $(\text{width}/100) \times |P2_X - P1_X|$

actual character height = $(\text{height}/100) \times |P2_Y - P1_Y|$

For example, if you have default P1 and P2 settings and specify a width of 2 and a height of 3.5,

width = $(2/100) \times (10000)$ = 200 plotter units or 0.5 cm

height = $(3.5/100) \times (7200)$ = 252 plotter units or 0.63 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.

width = $(2/100) = (5000 - 100)$ = 98 plotter units or 0.245 cm

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

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

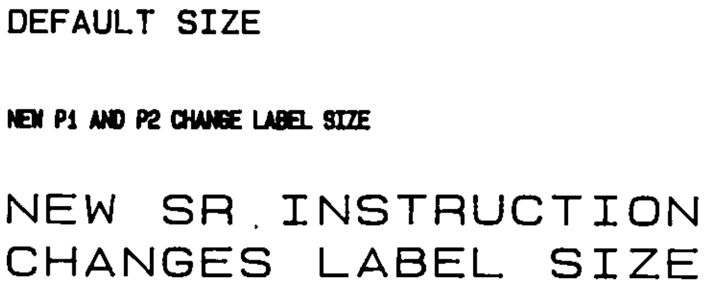
The following table summarizes the error conditions or unexpected results that might occur with the SR instruction.

Condition	Error	Plotter Response
no parameters	none	establishes default values
1 parameter	2	ignores instruction
more than 2 parameters	2	uses first 2 parameters
parameter out-of-range	3	ignores instruction

Example — Changing Relative Character Size

The following instructions show how changes in P1 and P2 affect labels drawn while an SR instruction is in effect. The first line initializes the plotter, which establishes default P1 and P2 settings along with the equivalent of *SR;*. Then the first label is drawn. Next, P1 and P2 are changed to define a square area, and a new label is drawn with the default character size. Finally, the character size is changed without changing P1 and P2, and another label is drawn. Notice that the new character size has equal parameters of 2.5; because the P1/P2 area is square, the characters are square.

```
"IN;SP1;PA300,7000;LBDEFAULT SIZE"+CHR$(3)
"IP0,0,5500,5500;PA300,6500;"
"LBNEW P1 AND P2 CHANGE LABEL SIZE"+CHR$(3)
"PA300,6000;SR2.5,2.5;"
"LBNEW SR INSTRUCTION"+CHR$(13)+CHR$(10);
"CHANGES LABEL SIZE"+CHR$(3)+"SP0;"
```



```
DEFAULT SIZE

NEW P1 AND P2 CHANGE LABEL SIZE

NEW SR INSTRUCTION
CHANGES LABEL SIZE
```

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

Character Slant Instruction, SL

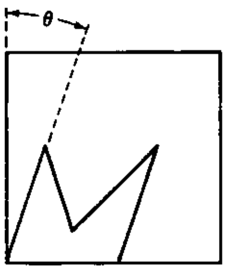
USES: The SL instruction specifies the slant at which characters are drawn. Use this instruction to create slanted text for emphasis.

SYNTAX: *SL* *tan θ* *term*
or
SL *term*

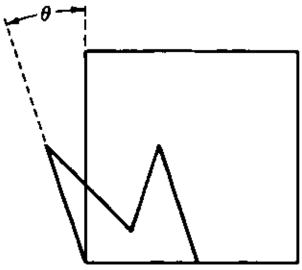
Parameter	Format	Range	Default
tangent θ	real	−128.0000 to 127.9999	0 (no slant)

EXPLANATION: An SL instruction without parameters (*SL ;*) sets the character slant to the default value of no slant (the same as *SL 0 ;*).

The parameter is the tangent of the angle θ from vertical, as shown on the next page.



Positive Slant



Negative Slant

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

The tangent is interpreted as an angle between ±90 degrees.

You can specify the actual tangent value, or you can use the TAN function available on most computers. Both methods are shown in the example at the end of this section. A table of tangent values for selected angles follows.

θ	Tangent
0	0
−10	−0.18
−20	−0.36
−30	−0.58
−40	−0.84
−45	−1.00
−50	−1.19
−60	−1.73
−90	infinity (error)

θ	Tangent
0	0
10	0.18
20	0.36
30	0.58
40	0.84
45	1.00
50	1.19
60	1.73
90	infinity (error)

A practical parameter range for this instruction is ±0.05 to ±1.5.

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

The following table summarizes the error conditions or unexpected results that might occur with the SL instruction.

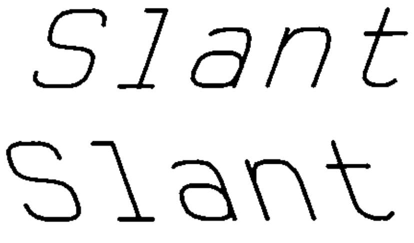
Condition	Error	Plotter Response
no parameters	none	establishes no slant (vertical characters)
more than 1 parameter	2	uses first parameter
parameter out-of-range	3	ignores instruction

Example — Specifying Character Slant

The following instructions show you two methods for specifying the slant parameter. The first label is drawn at a slant of 20 degrees by specifying the parameter as a variable generated by the TAN function. The second label is drawn at a slant of −20 degrees by using the tangent value given in the previous table.

NOTE: If you want to use the TAN function on your computer, check your computer documentation to see how your computer interprets angles. This version of Microsoft® BASIC interprets angles as radians, so line 40 in this program converts degrees to radians. On the HP Series 200 computers, simply execute the BASIC statement DEG before using the TAN function. ■

```
10 REM  Insert configuration statement here
20 PRINT #1,  "IN;SP1;SI.7,1.0;PA500,3000;"
30 PI = 3.141593
40 A=TAN(20*(PI/180))
50 PRINT #1,  "SL";A;";"
60 PRINT #1,  "LBSlant"+CHR$(3)
70 PRINT #1,  "PA500,2300;SL-.36;LBSlant"+CHR$(3)
80 PRINT #1,  "SP0;"
90 END
```



Absolute Direction Instruction, DI

USES: The DI instruction specifies the direction in which labels are drawn, independent of P1 and P2 settings. Use this instruction to change labeling direction when you want to label a curve in a line chart or label schematic drawings and blueprints.

SYNTAX: *DI* run, rise *term*

or

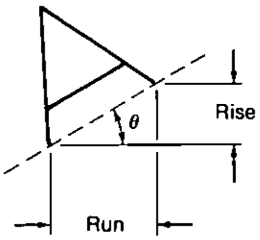
DI *term*

Parameter	Format	Range	Default
run (or cos θ)	real	−128.0000 to 127.9999	1
rise (or sin θ)	real	−128.0000 to 127.9999	0

EXPLANATION: A DI instruction without parameters (*DI;*) sets the label direction to horizontal (parallel to the X-axis). The settings of P1 and P2 do not have any effect on the label direction.

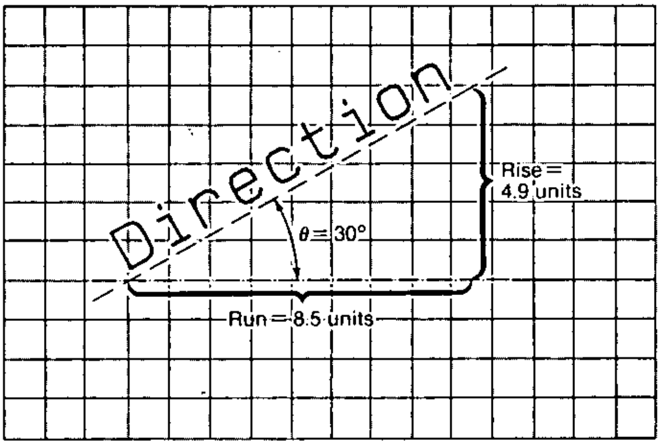
You can express the parameters as run and rise, or using trigonometric functions cos and sin according to the following relationship:

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



Suppose you want your label to be plotted in the direction shown in the following graph. You can do this in either of two ways: measure the run and the rise, or the angle. To use the first method, extend lines along the label and parallel to the X-axis. If you measure the run and rise, you can use these 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/cos = rise/run). In this example, $\theta = 30°$; 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 functions available on most computers. Both methods are shown in the example at the end of this section. 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

At least one parameter must be nonzero. For example:

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

A DI instruction remains in effect until another DI or DR instruction is executed, or the plotter is initialized or set to default conditions. A DI instruction updates the carriage-return point to the current pen position.

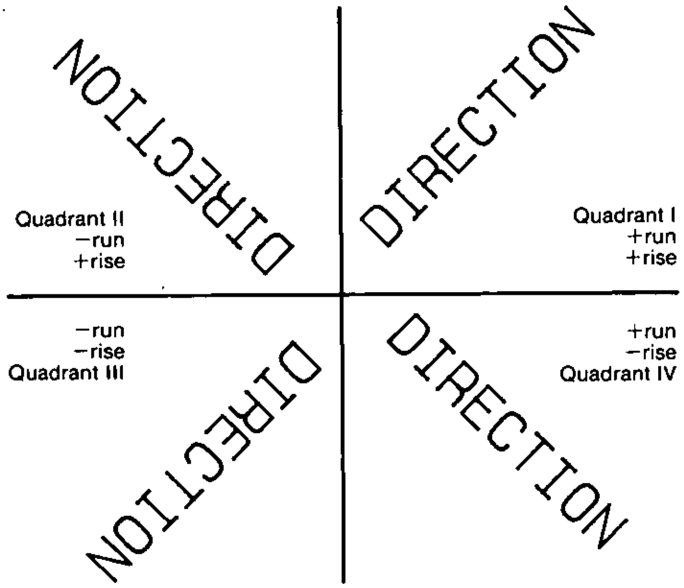
The following table summarizes the error conditions or unexpected results that might occur with the DI instruction.

Condition	Error	Plotter Response
no parameters	none	establishes horizontal label direction
more than 2 parameters	2	uses first 2 parameters
1 parameter	2	ignores instruction
parameter out-of-range	3	ignores instruction
both parameters are zero (0)	3	ignores instruction

Examples — Rotating Label Direction

The size and sign of the two parameters in the DI instruction determine the amount of rotation. In the following example, a label is drawn four times using each possible combination of positive and negative parameters. If you imagine the current pen position to be the origin of a coordinate system, you can see that the signs of the parameters determine which quadrant the label will be in.

```
"IN;SP1;SI.3,.5;PA3000,3000;"
"DI1,1;LB  DIRECTION"+CHR$(13)+CHR$(3)
"DI1,-1;LB  DIRECTION"+CHR$(13)+CHR$(3)
"DI-1,-1;LB  DIRECTION"+CHR$(13)+CHR$(3)
"DI-1,1;LB  DIRECTION"+CHR$(13)+CHR$(3)
"SP0;"
```

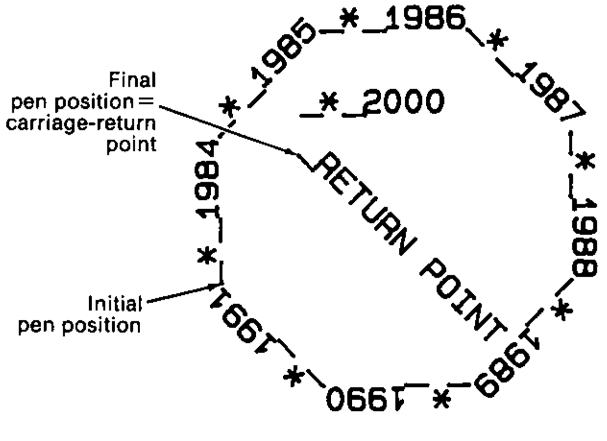


The instructions on the next page label the years 1984 through 1991, in a circular pattern starting with a vertical label. The direction in which each year is labeled is changed in increments of 45 degrees. Then the labels in the center are drawn to illustrate the use of the COS and SIN functions as parameters.

NOTE: If you want to use the COS and SIN functions on your computer, check your computer documentation to see how your computer interprets angles. This version of Microsoft® BASIC interprets angles as radians, so lines 130–140 and 170–180 convert degrees to radians when using the COS and SIN functions. On the HP Series 200 computers, simply execute the BASIC statement DEG when using the SIN and COS functions. ■

The label -*-2000 contains both a carriage return, CHR$(13), and a line feed, CHR$(10), *before* the label terminator, CHR$(3), so the pen position at the end of that label is one line below the beginning of that label. You can see that DI instructions update the carriage-return point by observing the pen's position at the end of the program. The final character in the last label is a carriage return, and the pen returns to the carriage-return point, the position of the pen at the last DI instruction.

```
10    REM  Insert configuration statement here
20    PRINT #1, "IN;SP1;PA6000,3000;"
30    PRINT #1, "DI0,1;LB_*_1984"+CHR$(3)
40    PRINT #1, "DI1,1;LB_*_1985"+CHR$(3)
50    PRINT #1, "DI1,0;LB_*_1986"+CHR$(3)
60    PRINT #1, "DI1,-1;LB_*_1987"+CHR$(3)
70    PRINT #1, "DI0,-1;LB_*_1988"+CHR$(3)
80    PRINT #1, "DI-1,-1;LB_*_1989"+CHR$(3)
90    PRINT #1, "DI-1,0;LB_*_1990"+CHR$(3)
100   PRINT #1, "DI-1,1;LB_*_1991"+CHR$(3)
110   PRINT #1, "PA6450,3900;"
120   PI = 3.141593
130   A=COS(0*(PI/180))
140   B=SIN(0*(PI/180))
150   PRINT #1, "DI";A;",";B;";"
160   PRINT #1, "LB_*_2000"+CHR$(13)+CHR$(10)+CHR$(3)
170   C=COS(-45*(PI/180))
180   D=SIN(-45*(PI/180))
190   PRINT #1, "DI";C;",";D;";"
200   PRINT #1, "LB_RETURN POINT"+CHR$(13)+CHR$(3)
210   PRINT #1, "SP0;"
220   END
```



Relative Direction Instruction, DR

USES: The DR instruction specifies the direction in which labels are drawn, relative to the scaling points P1 and P2. This means that *label direction is adjusted when P1 and P2 change so that labels maintain the same relationship to the plotted data.* At power-on, the plotter assumes label direction is relative. Use this instruction to change the percentage values for relative label direction, or to reestablish relative direction after a DI instruction has been executed.

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

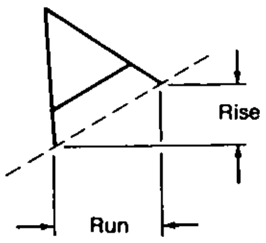
Parameter	Format	Range	Default
run	real	−128.0000 to 127.9999	1% of $P2_X - P1_X$
rise	real	−128.0000 to 127.9999	0% of $P2_Y - P1_Y$

EXPLANATION: A DR instruction without parameters (*DR* ;) sets the label direction to horizontal (parallel to the X-axis). When you include parameters, the actual angle at which the label is plotted will vary with the settings of P1 and P2. At least one parameter must be nonzero.

You can express the parameters as run and rise according to the following relationship:

$$\frac{\text{rise}}{\text{run}}$$

where:



The concept of specifying the run and the rise is similar to the DI instruction (refer to the discussion under the *Absolute Direction Instruction, DI*, described previously.) However, there is a very important distinction between the DR and DI instructions: in the DR instruction, the run and rise parameters are a *percentage* of the X- and Y-distances between P1 and P2, as shown below. As a result, when the settings of P1 and P2 change, and percentage remains constant, the angle changes. The examples at the end of this section show how relative label direction changes when P1 and P2 are changed.

run — interpreted as a percentage of $P2_X - P1_X$

rise — interpreted as a percentage of $P2_Y - P1_Y$

A DR instruction remains in effect until another DR or DI instruction is executed, or the plotter is initialized or set to default conditions. A DI instruction updates the carriage-return point to the current pen position.

The following table summarizes the error conditions or unexpected results that might occur with the DR instruction.

Condition	Error	Plotter Response
no parameters	none	establishes horizontal label direction
more than 2 parameters	2	uses first 2 parameters
1 parameter	2	ignores instruction
parameter out-of-range	3	ignores instruction
both parameters are zero (0)	3	ignores instruction

Example — Effects of Changing P1 or P2 on Relative Label Direction

The following description and program example will help you visualize the direction of labeling when using the DR instruction with various parameters.

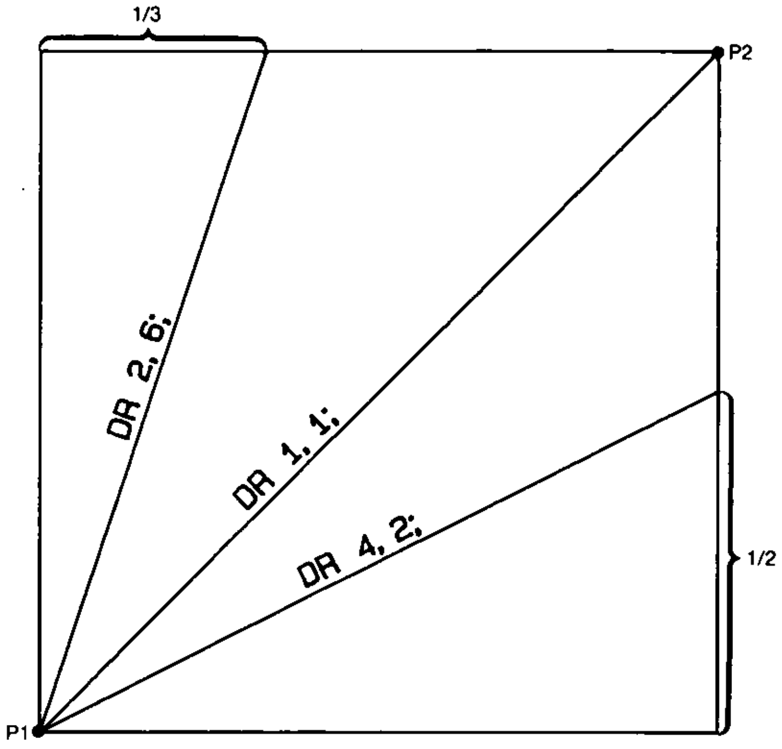
Think of label directions as being parallel to a line starting at the lower-left scaling point (usually P1) and intersecting the top edge or the opposite side of the P1/P2 area. To calculate where the intersection is, first determine which is larger: the run or the rise. There are three possibilities, as follows:

- If run = rise, the directional line will go directly from P1 to P2.
- If run < rise, the directional line will intersect the top of the plotting area, a fraction of the way *across* toward P2. The fraction is determined by run/rise. Thus, if run = 2 and rise = 6, the line intersects the top of the plotting area one-third (2/6) of the way toward P2.
- If rise > run, the directional line will intersect the side of the plotting area, a fraction of the way *up* toward P2. The fraction is determined by rise/run. Thus, if run = 4 and rise = 2, the line intersects the side of the plotting area one-half (2/4) of the way toward P2.

Remember, since labeling starts at the current pen position, labels will be parallel to these lines, not necessarily on them. Also, negative parameters have the same effect on direction as described under the DI instruction.

NOTE: The interactions of the SI, SR, DI, and DR instructions are complex when using negative parameters. Refer to *Parameter Interaction in Labeling Instructions* in this chapter. ■

The following example illustrates the concepts of relative label directions.



```
10 REM 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;LBDR 1,1;"+CHR$(3)
70  PRINT #1, "PU333.33,1000;PD0,0;PU;DR2,6;"
80  PRINT #1, "CP15,.25;LBDR 2,6;"+CHR$(3)
90  PRINT #1, "PU1000,500;PD0,0;PU;DR4,2;"
100 PRINT #1, "CP15,.25;LBDR 4,2;"+CHR$(3)
110 PRINT #1, "SP0;"
120 END
```

Program Explanation

10 establishes HP-IB or RS-232-C interface conditions

20 initializes the plotter; selects pen 1; sets the scaling points P1 and P2 to define a square area

30 scales the P1/P2 area into user units; sets a relative character size

40 moves to P1; outlines the P1/P2 area

50 finishes outline of P1/P2 area; draws a line from P1 to P2 and lifts the pen; establishes a relative label direction

60 moves the pen 15 spaces and 0.25 lines (refer to the CP instruction next in this chapter); labels "DR 1 , 1 ;"

70 draws a line between P1 and a point that is one-third of the way to P2 along the top of the P1/P2 area and lifts the pen; establishes a new relative label direction

80 moves the pen 15 spaces and 0.25 lines; labels "DR 2 , 6 ;"

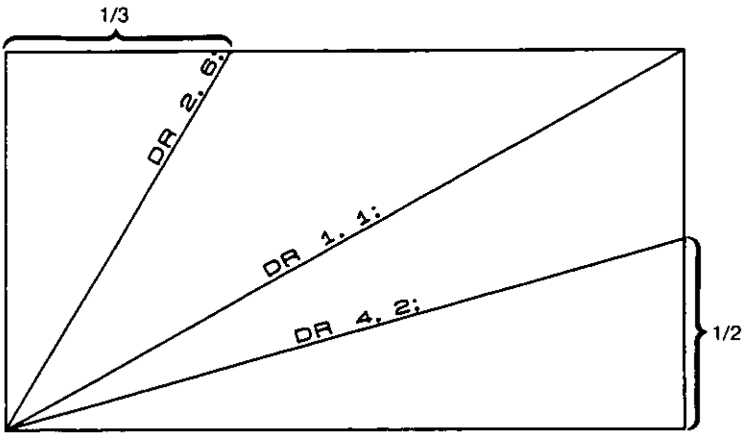
90 draws a line between P1 and a point that is one-half of the way to P2 along the side of the P1/P2 area and lifts the pen; establishes a new relative label direction

100 moves the pen 15 spaces and 0.25 lines; labels "DR 4 , 2 ;"

110 returns the pen to the carousel

Now change the relative positions of P1 and P2 by replacing line 20 in the previous program with the following line. Notice 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 was specified in line 30.

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



Character Plot Instruction, CP

USES: The CP instruction moves the pen the specified number of character plot cells. Use this instruction to move the pen any number of character spaces or lines from the current pen position. For example, you can move the pen any number of character spaces in order to indent or center a label.

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

Parameter	Format	Range	Default
spaces	real	−128.0000 to 127.9999	none
lines	real	−128.0000 to 127.9999	none

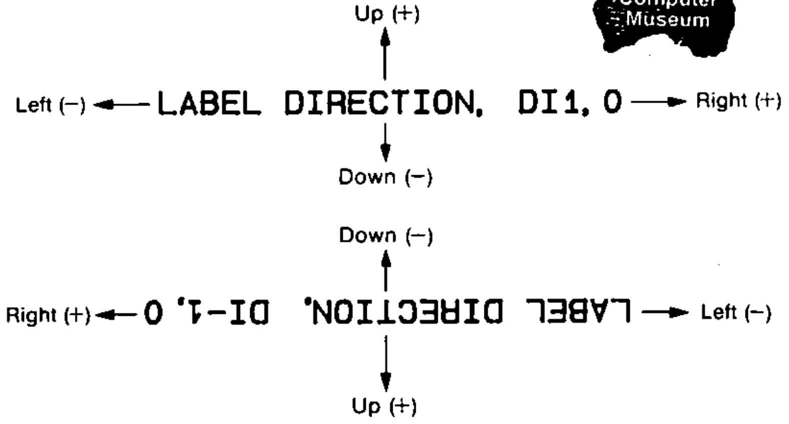
EXPLANATION: A CP instruction without parameters (*CP;*) performs a carriage return and line feed, moving one line down and returning to the carriage-return point. When specified, the parameters are interpreted as follows. (For more information on spaces, lines, and the character plot cell, refer to *Adjusting Character Size, Spacing, and Position* earlier in this chapter.)

1. **Spaces.** Positive space values specify the number of spaces the pen will move to the right of the current pen position; negative values specify the number of spaces the pen will move to the left. Right and left are relative to current label direction. Refer to the figure below.

   A space is defined as the width of one character plot cell, or 1.5 times the uppercase character width. The character width is specified in the SI or SR instructions.

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

   A line is defined as the height of one character plot cell. The height of the character plot cell is 2 times the uppercase character height, which is specified in the SI or SR instructions.



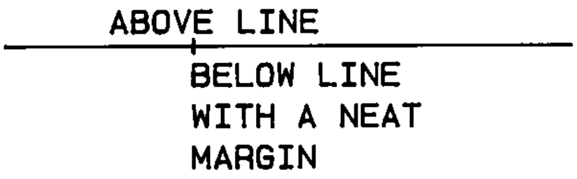
CP instructions are executed with the current pen status (up or down), and all moves are made with respect to the current character origin. The CP instruction affects only the placement of the next label; you must issue new CP instructions to affect subsequent labels.

The following table summarizes the error conditions or unexpected results that might occur with the CP instruction.

Condition	Error	Plotter Response
no parameters	none	performs carriage return and line feed
1 parameter	2	ignores instruction
more than 2 parameters	2	uses first 2 parameters
parameter out-of-range	3	ignores instruction

Example — Using CP to Align Labels

The following example shows how to use the CP instruction to produce lettering along a line (but not directly on top of it), and to align labels along a left margin.



```
10  REM  Insert configuration statement here
20  PRINT #1, "IN;SP1;PA4000,7000;PD1000,7000;PU;"
30  PRINT #1, "CP5,.35;LBABOVE LINE"+CHR$(3)
40  PRINT #1, "PA2000,7000;XT;CP0,-.95;"
50  PRINT #1, "LBBELOW LINE"+CHR$(13)+CHR$(10)+CHR$(3)
60  PRINT #1, "LBWITH A NEAT"+CHR$(3)
70  PRINT #1, "CP;LBMARGIN"+CHR$(3)+"SP0;"
80  END
```

Program Explanation

10 establishes HP-IB or RS-232-C interface conditions

20 initializes the plotter; selects pen 1; draws a line 3000 plotter units long, ending at the point 1000,7000 (this is the starting position and the carriage-return point for the label in line 30); lifts the pen

30 moves 5 spaces to the right and 0.35 lines up; labels "ABOVE LINE"

40 moves to the point 2000,7000 (this is the new starting position and carriage-return point for the label in line 50); draws an X-tick to mark the new carriage-return point; moves 0.95 lines down

50 labels "BELOW LINE"; uses carriage return and line feed to move to the carriage-return point and down 1 line

60 labels "WITH A NEAT"

70 moves to the carriage-return point and down 1 line; labels "MARGIN"; returns pen to the carousel

Notice that a fractional line parameter was specified in line 30 so that the label would not be drawn on the line, as would happen with a line parameter of zero. Also notice that the instruction *CP;* in line 70 causes the same moves as the carriage return and line feed in line 50.

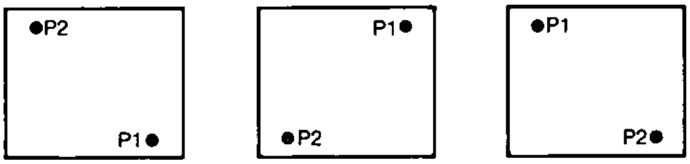
Parameter Interaction in Labeling Instructions

There are three interacting factors that affect the direction and mirroring of labels:

- label direction as specified by the DI or DR instructions
- the sign of the parameters for the SI or SR instructions
- the relative positions of P1 and P2 (if DR or SR is in effect)

It is not a good idea to do these all at once. For easily visualized results, you should locate P1 and P2 in their standard positions (P1 in the lower-left and P2 in the upper-right corner of the plot).

Following is a summary of how changes in the standard position of P1 and P2, as illustrated on the next page, can affect labeling.



Non-Standard P1 and P2 Locations

When you use DI with either SI or SR, changes in the normal position of P1 and P2 do *not* affect label direction. Refer to the SI and SR instructions for a discussion of how negative parameters cause labels to be mirrored.

When you use DR and SI together, changes in the setting of P1 and P2 do effect label direction. The plotter multiplies the algebraic differences ($P2_X - P1_X$) and ($P2_Y - P1_Y$) by the run and rise parameters of the DR instruction. The resulting parameters, when applied to the standard coordinate system, determine the label baseline. Mirroring about this baseline is determined by the SI parameters.

When you use DR and SR together, interactions are most complex. You will find it simplest to use standard settings of P1 and P2 in this case. DR parameters interact with the algebraic differences in P1 and P2 to establish label direction; SR parameters interact with these differences to create mirroring. Signs of both parameters and P2/P1 differences are important. A negative sign in either the parameter or the difference will affect both the DR and SR instructions. When both parameter and P2/P1 differences are negative, standard direction (no mirroring) results.

Plotter Character Sets

When default conditions are established, the plotter automatically sets both the standard and alternate character sets to character set 0, ANSI ASCII English, which follows:

```
            CHARACTER SET 0
! "#$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`
abcdefghijklmnopqrstuvwxyz{|}~
```

Since the plotter uses this set automatically, there is no need for you to select another character set unless you have a need for it.

The plotter has 5 character sets available. All characters are fixed-space; that is, they occupy an equal horizontal space. Each character is drawn using a fixed number of vectors.

All sets draw identical uppercase and lowercase letters and numbers. The differences between the character sets are in the *additional* characters that are needed for a certain language, for example the ç in French/German set 2 and the Æ in Scandinavian set 3. The sets are defined in the following table; refer to Appendix B for a complete illustration of the characters in each set.

The graphics enhancement cartridge enables the plotter to access 14 additional character sets, all fixed-space.

Standard Plotter		
Set Number	Name	ISO Registration Number
0	ANSI ASCII	006
1	HP 9825 HPL Character Set	—
2	French/German	—
3	Scandinavian	—
4	Spanish/Latin American	—

Graphics Enhancement Cartridge		
Set Number	Name	ISO Registration Number
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 Norwegian, Version 1	060
33	ISO German	021
34	ISO French	025
35	ISO British	004
36	ISO Italian	015
37	ISO Spanish	017
38	ISO Portuguese	016
39	ISO Norwegian, Version 2	061

If you wish, you can designate a standard set and an alternate set using the CS and CA instructions, respectively. You can then label with both character sets either by selecting the desired set with the SS and SA instructions, or by using a technique called shift in/shift out. These two methods are described in the next section, *Labeling with Standard and Alternate Character Sets.* Refer also to the descriptions of the CS, CA, SS, and SA instructions, which follow that section.

Labeling with Standard and Alternate Character Sets

If you always intend to label with the default set, ANSI ASCII English, you do not need to execute any of the CS, CA, SS, or SA instructions before labeling. However, if you intend to use a different set, you will find that these instructions are very useful. This section describes methods for using these instructions and provides examples. The syntax of each instruction is described in detail following this section.

Selecting Sets with the SS and SA Instructions

The principles for labeling with different character sets are very simple. Just remember that the *designate* character set instructions, CS and CA, tell the plotter which of the character sets you want available for labeling, whereas the *select* character set instructions, SS and SA, tell the plotter which of these two designated sets to use. Follow these steps for designating and selecting character sets.

1. Designate the standard and alternate character sets by executing the CS and CA instructions.

   When initialized or set to default conditions, the plotter assigns both the standard and alternate sets to be set 0. Therefore, you only need to execute a CS or CA instruction if you want to designate a character set *other than set 0.*

2. Select either the designated standard set or the designated alternate set as the set to be used for subsequent labeling by executing an SS or SA instruction.

When initialized or set to default conditions, the plotter assumes you are labeling with the standard set. Therefore, if you want to begin by labeling with the standard set, you do *not* need to execute an SS instruction. However, if you want to begin with the alternate set, execute CA, then execute SA. Throughout your program, you can "toggle" between the sets by executing SS or SA, as appropriate.

The easiest way to label characters in any character set is to enter the ANSI ASCII equivalent from your keyboard. Or, you can use a computer language-dependent function such as CHR$ to enter the equivalent decimal code. For example, to cause the plotter to label the " ¿ " in set 4, you can either use "#" (the ASCII equivalent character) from the keyboard, or CHR$(35) (the equivalent decimal code). A full table of ASCII decimal codes and characters appears in Appendix B.

The following HP-GL instructions illustrate how to designate, select, and label with standard and alternate character sets. Notice that the label string in the LB instruction shows the character that appears on the computer keyboard, unless the character is not available on the keyboard; in this case, the CHR$ function is used to access the character. (Refer to *How to Send the Label Terminator and Other Nonprinting Characters* under the LB instruction in this chapter for more information on using the CHR$ function.) Also, notice that the "SET 4" label uses the underscore character in set 4 that automatically backspaces before it is drawn.

```
"IN;SP1;PA3000,4000;"
"CS0;CA4;"
"SS;LBS_E_T_0_"+CHR$(3)
"SA;LB  S_E_T_4_"+CHR$(3)
"CP-14,-2;LB#su compan"+CHR$(124)+"ia?"+CHR$(3)
```

```
S_E_T_0_  SET4

¿su compañia?
```

Selecting Sets with the Shift-in/Shift-out Control Characters

You might find that you need to access a different character set in the middle of a label string. Using SS and SA to toggle between sets can be inefficient. Instead, you can use two control characters, the shift-out (SO, decimal code 14) and the shift-in (SI, decimal code 15), to change sets without leaving label mode. The shift-out character shifts to the alternate set designated by the CA instruction, whereas the shift-in character shifts back to the standard set designated by the CS instruction.

Following are two examples that show you how to "shift out" to the alternate character set. If you *cannot* access these characters from your keyboard, use the CHR$(14) and CHR$(15) (or equivalent) functions to specify the decimal code. You can access the French cedilla by striking \ on the keyboard or using CHR$(92).

```
"IN;SP1;PA3000,3000;"
"CA2;"
"LBEnglish"+CHR$(14)+"  Fran"+CHR$(92)+"ais"+CHR$(3)
```

```
English  Français
```

If you can access the characters from your keyboard, use SO and SI. For example, on HP Series 80 and HP Series 200 computers, shift-out is accessed by holding down CTRL while pressing N; shift-in is accessed by holding down CTRL while pressing O.

```
"IN;SP1;PA3000,3000;"
"CA2;"
"LBEnglish␎  Fran\ais␃"
```

Designate Standard Character Set Instruction, CS

USES: The CS instruction designates one of the character sets as the standard character set to be used in labeling instructions. Use this instruction to change the standard set to one with characters appropriate to your application. This instruction is particularly useful if you plot most of your labels in a language other than English.

SYNTAX: *CS* set *term*
or
CS *term*

Parameter	Format	Range	Default
set	integer	0 to 4 without cartridge	0
		0 to 4, 6 to 9, 30 to 39 with cartridge	0

EXPLANATION: The character set designated by the CS instruction is used for all labeling operations when the standard set is selected by the SS instruction or by the control character shift-in (decimal 15) in a label string. A CS instruction without parameters (*CS;*) establishes the default character set 0 (English). The CS instruction remains in effect until another CS instruction is executed, or the plotter is initialized or set to default conditions.

If you execute a CS instruction while the standard set is selected (e.g., an SS instruction has been executed), subsequent labeling will be done with the set designated by the new CS instruction.

However, if you execute a CS instruction while the alternate set is selected (e.g., an SA instruction has been executed), subsequent labeling will not be done with the new standard set until it is selected with an SS instruction.

For an example using the CS instruction, refer to *Labeling with Standard and Alternate Character Sets* earlier in this chapter.

The following table summarizes the error conditions or unexpected results that might occur with the CS instruction.

Condition	Error	Plotter Response
no parameters	none	designates character set 0
more than 1 parameter	2	uses first parameter
parameter out-of-range (<−32768, >32767)	3	ignores instruction
parameter in-range, but invalid character set	5	ignores instruction

Designate Alternate Character Set Instruction, CA

USES: The CA instruction designates one of the character sets as the alternate character set to be used in labeling instructions. Use this instruction to provide an additional character set that you can easily access in a program.

SYNTAX: *CA* set *term*
or
CA *term*

Parameter	Format	Range	Default
set	integer	0 to 4 without cartridge	0
		0 to 4, 6 to 9, 30 to 39 with cartridge	0

EXPLANATION: The character set designated by the CA instruction is used for all labeling operations when the alternate set is selected by the SA instruction or by the control character shift-out (decimal 14) in a label string. A CA instruction without parameters (*CA;*) establishes the default character set 0 (ANSI ASCII English). The CA instruction remains in effect until another CA instruction is executed, or the plotter is initialized or set to default conditions.

If you execute a CA instruction while the alternate set is selected (an SA instruction has been executed), subsequent labeling will be done with the set designated by the new CA instruction.

However, if you execute a CA instruction while the standard set is selected (an SS instruction has been executed), subsequent labeling will not be done with the new alternate set until it is selected with an SA instruction.

For an example using the CA instruction, refer to *Labeling with Standard and Alternate Character Sets* earlier in this chapter.

The following table summarizes the error conditions or unexpected results that might occur with the CA instruction.

Condition	Error	Plotter Response
no parameters	none	designates character set 0
more than 1 parameter	2	uses first parameter
parameter out-of-range (<−32 768, >32 767)	3	ignores instruction
parameter in-range, but invalid character set	5	ignores instruction

Select Standard Character Set Instruction, SS

USES: The SS instruction selects the *standard* set designated by the CS instruction as the character set to be used for subsequent labeling. Use this instruction to shift from the currently selected alternate set to the designated standard set.

SYNTAX: *SS term*

EXPLANATION: The SS instruction tells the plotter to draw subsequent labeling instructions using characters from the 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 SS instruction 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.

For an example using the SS instruction, refer to *Labeling with Standard and Alternate Character Sets* earlier in this chapter.

The following table summarizes the error conditions or unexpected results that might occur with the SS instruction.

Condition	Error	Plotter Response
no parameters	none	selects standard set
1 or more parameters	2	selects standard set

Select Alternate Character Set Instruction, SA

USES: The SA instruction selects the *alternate* set designated by the CA instruction as the character set to be used for subsequent labeling. Use this instruction to shift from the currently selected standard set to the designated alternate set.

SYNTAX: *SA term*

EXPLANATION: The SA instruction tells the plotter to draw subsequent labeling instructions using characters from the character set designated by the CA instruction. The SA instruction is equivalent to using the control character "shift-out" (decimal 14) within a label string.

The SA instruction remains in effect until an SS instruction is executed, or the plotter is initialized or set to default conditions.

For an example using the SA instruction, refer to *Labeling with Standard and Alternate Character Sets* earlier in this chapter.

The following table summarizes the error conditions or unexpected results that might occur with the SA instruction.

Condition	Error	Plotter Response
no parameters	none	selects alternate set
1 or more parameters	2	selects alternate set

User-Defined Character Instruction, UC

USES: The UC instruction draws a character that you design. Use this instruction to create symbols or characters not included in the plotter's character sets, or to draw logos.

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

Parameter	Format	Range	Default
pen control	integer	99 to 127.9999 = pen down −99 to −128.0000 = pen up	pen up
X,Y increments	real	−98.9999 to +98.9999	none

EXPLANATION: A UC instruction without parameters (*UC;*) causes the pen to move one character space. A UC instruction with parameters draws the character specified by the parameters. The parameters are interpreted as follows.

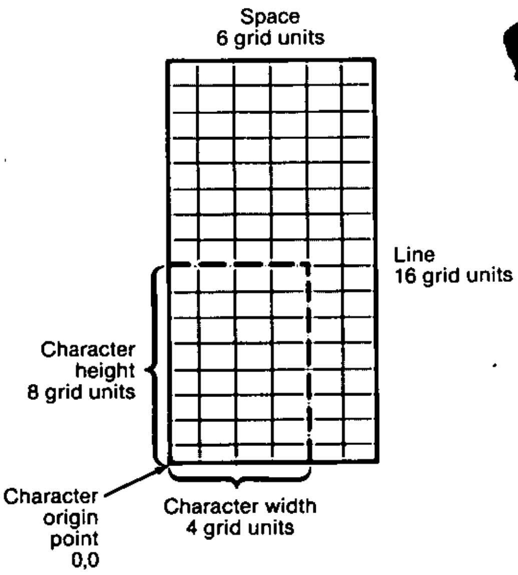
1. **Pen Control.** The UC instruction initially raises the pen (regardless of the current pen status). Therefore, *in order to draw the character you must include at least one pen down parameter.* Once a you specify a pen down parameter, the pen remains down for subsequent X,Y increment moves until you specify a pen up parameter or terminate the UC instruction. Include as many pen control parameters as you need to draw your character.

2. **X,Y Increments.** The X,Y increment pairs represent *relative* positions on a grid that is superimposed on the character plot cell. The cell is divided into 6 horizontal units and 16 vertical units. Refer to the figure on the next page.

The UC instruction initially sets the pen position at 0,0 on the grid. The pen then moves consecutively to the points defined by the X,Y increment pairs. Include as many X,Y increment pairs as you need to draw your character. The X,Y increments are interpreted as follows. All references to the right, left, up, and down are relative to the current label direction.

- The X-increment specifies the number of grid units that the pen will move horizontally from the current pen position on the grid. A positive increment moves the pen to the right; a negative increment moves the pen to the left.
- The Y-increment specifies the number of grid units that the pen will move vertically from the current pen position on the grid. A positive increment moves the pen up; a negative increment moves the pen down.

Upon completion of the character, the pen is raised and moves one character plot cell to the right of the current character's origin. The current pen status is then restored.



Grid on a Character Plot Cell

You can extend your character into the area normally reserved for the space around a character, and even into the next character plot cell. This is illustrated in the example at the end of this section. However, if you want your character to be the *same size as characters in any character set, confine your character to a 4 × 8 grid* as shown in the figure on the previous page.

The number of grid units in a character plot cell does not change. However, the size of the cell can change. That is, if you issue a character size instruction (SI or SR), the cell will always be 1½ times the specified width and 2 times the specified height. Thus, the cell can be larger or smaller, but the number of grid units in the cell will always be the same.

User-defined characters are drawn using the current direction (DI or DR), size (SI or SR), and slant (SL).

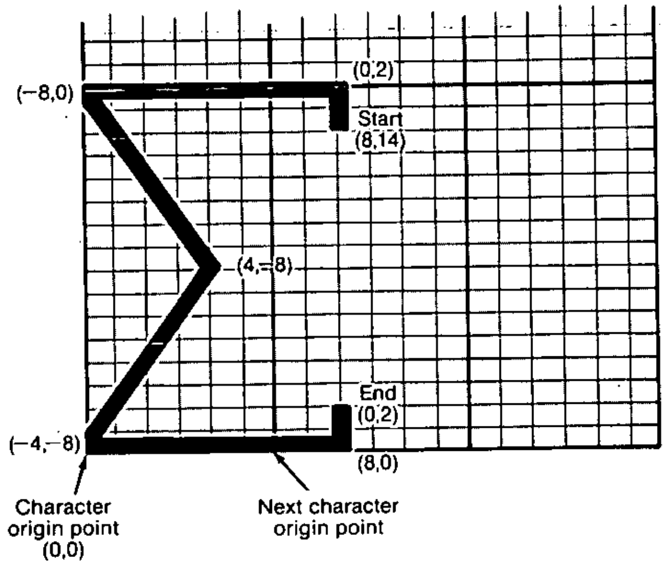
The following table summarizes the error conditions or unexpected results that might occur with the UC instruction.

Condition	Error	Plotter Response
no parameters	none	moves 1 character space
no pen down parameters	none	pen remains up while plotting the character
odd number of increments	2	executes increments as pairs up to the extra increment
parameter out-of-range	3	executes up to the pair with out-of-range parameter

Example — Defining Your Own Characters with the UC Instruction

As shown in the next illustration, user-defined characters are not limited to a single character plot cell. In this case, the Σ is drawn to the full height of the cell, and its width extends into the next cell.

```
"UC8,14,99,0,2,-8,0,4,-8,-4,-8,8,0,0,2;"
```

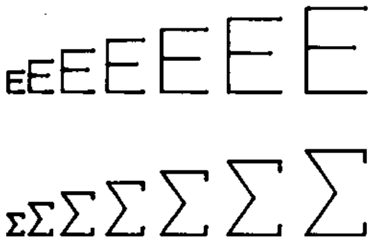


Drawing a Sigma

The program on the next page generates a series of E's and Σ symbols. The capital E is labeled by the LB instruction, and is included to show that the SI instruction has the same effect on both user-defined characters and normal characters.

NOTE: Although line 50 is printed on two lines to fit on this page, you should send it to the plotter as one string. ■

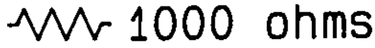
```
10   REM  Insert configuration statement here
20   PRINT #1,  "IN;SP1;PA1000,1000;"
30   FOR A =.19 TO .89 STEP .1
40     PRINT #1,  "SI";A;",";A*1.4;";"
50     PRINT #1,  "UC4,7,99,0,1,-4,0,2,-4,-2,-4,
                       4,0,0,1;"
60   NEXT A
70   PRINT #1,  "PA1000,1750;"
80   FOR B=.19 TO .89 STEP .1
90     PRINT #1,  "SI";B;",";B*1.4;";"
100    PRINT #1,  "LBE"+CHR$(3)
110  NEXT B
120  PRINT #1, "SP0;"
130  END
```



Example — Embedding a User-Defined Character in a Label

You may want to use a character you have defined as part of a label. The following program defines a resistor symbol which extends lengthwise beyond one CP cell. The CP instruction is added to move the current pen position beyond the limits of the user-defined character. (Remember, upon completion of a user-defined character, the pen normally moves *one* CP cell to the right of the character origin. Without the CP instruction, the number 1000 would overlap the resistor symbol.)

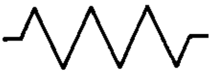
```
"IN;SP1;PA1000,5000;SI.25,.4;"
 "UC0,4,99,1.75,0,1.5,4,3,-8,3,8,3,-8,3,8,
  3,-8,1.5,4,1.75,0;"
 "CP3.25,0;LB1000 ohms"+CHR$(3)
```



NOTE: Although these examples print the UC instruction on two lines to fit on this page, you should send it to the plotter as one string. ■

As mentioned in the explanation of the UC instruction, you can change the size of a user-defined character with the SI or SR instructions. You can also change the size by specifying different increments in the UC instruction itself. In the following example, each increment parameter in the previous UC instruction is doubled, resulting in a resistor symbol twice the previous size.

```
"IN;SP1;PA1000,4000;SI.25,.4;"
"UC0,8,99,3.5,0,3,8,6,-16,6,16;6,-16,6,16,
 6,-16,3,8,3.5,0;"
```



CHAPTER 7

Drawing Circles, Arcs, and Polygons

What You'll Learn in This Chapter

The instructions discussed in this chapter can be executed *only* if you have a cartridge installed in your plotter. In this chapter you will learn how to:

- plot circles, arcs, and rectangles
- define your own polygons
- outline and fill polygons
- allocate memory to the polygon or I/O buffer (RS-232-C only)

HP-GL Instructions Covered

CI	Circle Instruction
AA	Arc Absolute Instruction
AR	Arc Relative Instruction
FT	Fill Type Instruction
PT	Pen Thickness Instruction
WG	Fill Wedge Instruction
EW	Edge Wedge Instruction
RA	Fill Rectangle Absolute Instruction
EA	Edge Rectangle Absolute Instruction

RR	Fill Rectangle Relative Instruction
ER	Edge Rectangle Relative Instruction
PM	Polygon Mode Instruction
EP	Edge Polygon Instruction
FP	Fill Polygon Instruction

Terms You Should Understand

If you are unfamiliar with any of these terms, refer to the glossary at the end of the manual.

Arc	Modulo
Chord	Polygon
Chord Angle	Polygon Buffer
Cross-Hatch	Polygon Mode
Edge	Subpolygon
Fill Type	Truncate
Hatch	Wedge

Circle Instruction, CI

USES: The CI instruction draws a circle using the specified radius and chord angle. Use this instruction to draw circles and polygons.

SYNTAX: *CI* radius (,chord angle) *term*

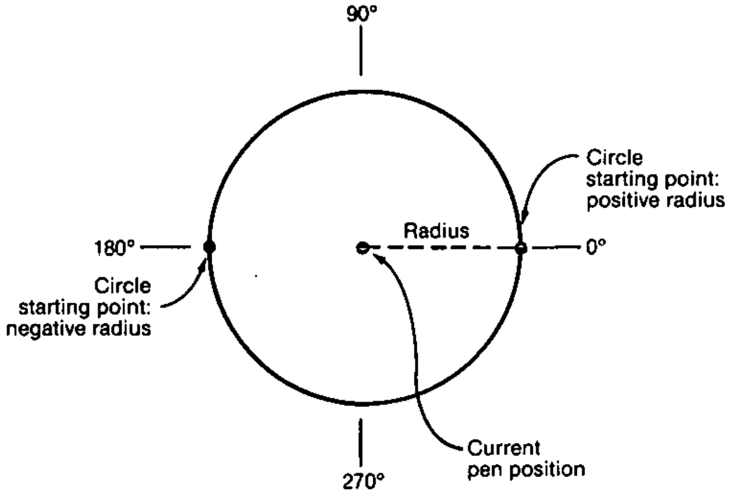
Parameter	Format	Range	Default
radius	real	−32 768.0000 to 32 767.9999 current units	none
chord angle	real	0.1 to 180 degrees*	5 degrees

*This is the practical range. The allowable range is −32 768.0000 to 32 767.9999 degrees.

EXPLANATION: The radius and chord angle parameters are interpreted as follows.

1. **Radius.** The current pen position is the center of the circle. The radius determines the size of the circle. The sign of the radius parameter defines the starting point of the circle: a

circle with a positive radius is drawn beginning at the 0-degree point; a circle with a negative radius is drawn beginning at the 180-degree point.


90°
Circle starting point: positive radius
Radius
180°
0°
Circle starting point: negative radius
Current pen position
270°

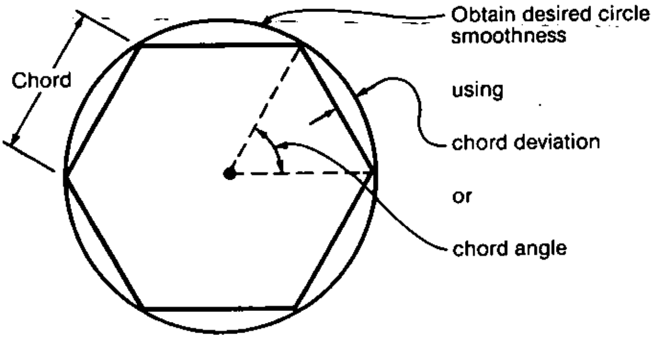
When scaling is *on*, the radius parameter is interpreted as user units.

When scaling is *off*, use plotter units for the radius parameter. Positive numbers are truncated. When you use negative numbers as the parameter, the number is changed to the next more negative number and truncated. For example, both −123.4 and −123.9 become −124.

2. **Chord Angle.** A plotted circle is actually made up of a series of straight line segments, or chords, as shown on the following page. Increasing the number of chords (by decreasing the chord angle parameter) increases the smoothness of the circle. However, this also increases the length of time required to draw the circle.

   If you specify a chord angle parameter greater than 180 degrees, then a chord angle of 360 minus the chord angle is used. If you specify a chord angle of 0 degrees, then a chord angle of 0.36 is used.

The sign of this parameter is ignored. The default chord angle is 5 degrees, which causes the circle to be made up of 72 chords. The example on the next page illustrates the effects of different chord angles.



The CI instruction includes an automatic pen down. When a CI instruction is received, the pen lifts, moves from the center of the circle (the current pen position) to the starting point on the circumference, lowers the pen, draws the circle, then returns with the pen up to the center of the circle. After the circle is drawn, the previous pen status (up or down) is restored. To avoid drawing lines to the center of the circle, move to and from the circle's center with the pen up. Each chord of the circle is drawn using the currently defined line type.

The following table summarizes the error conditions or unexpected results that might occur with the CI instruction.

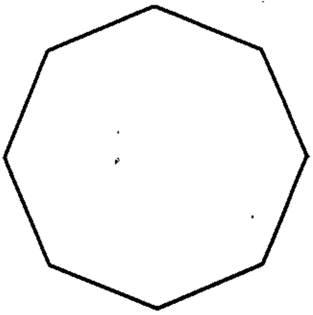
Condition	Error	Plotter Response
no parameters	none	ignores instruction
1 parameter	none	draws a circle with the specified radius using a chord angle of 5 degrees
more than 2 parameters	2	uses first 2 parameters
chord angle > 180	none	chord angle = 360 − chord angle
chord angle = 0	none	chord angle = 0.36
parameter out-of-range	3	ignores instruction

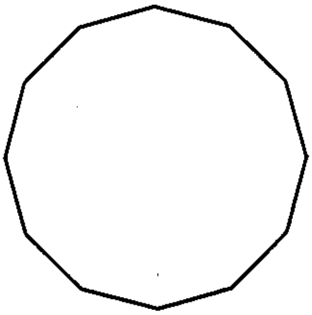
Example — Effects of Chord Angle on Circle Smoothness

The following instructions draw circles with the same radius, but with different chord angles. Note that the circle becomes smoother as the chord angle parameter decreases.

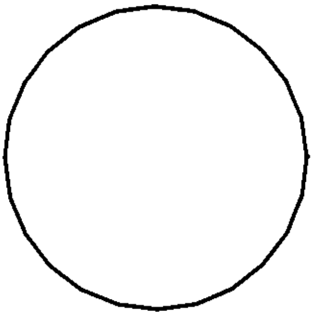
```
"IN;SP1;"
"PA3700,6050;CI800,45;"
"PA6300,6050;CI800,30;"
"PA3700,3950;CI800,15;"
"PA6300,3950;CI800,5;"
"SP0;"
```

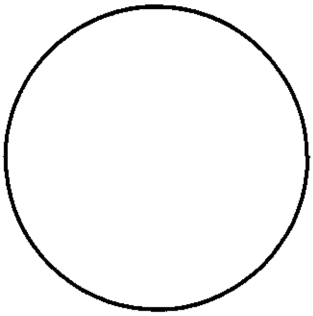
NOTE: This is a program segment. The above instructions must be sent in combination with computer language statements such as OUTPUT#, PRINT, etc., as discussed in Chapter 1. ■


45-Degree chord angle


30-Degree chord angle

Polygons

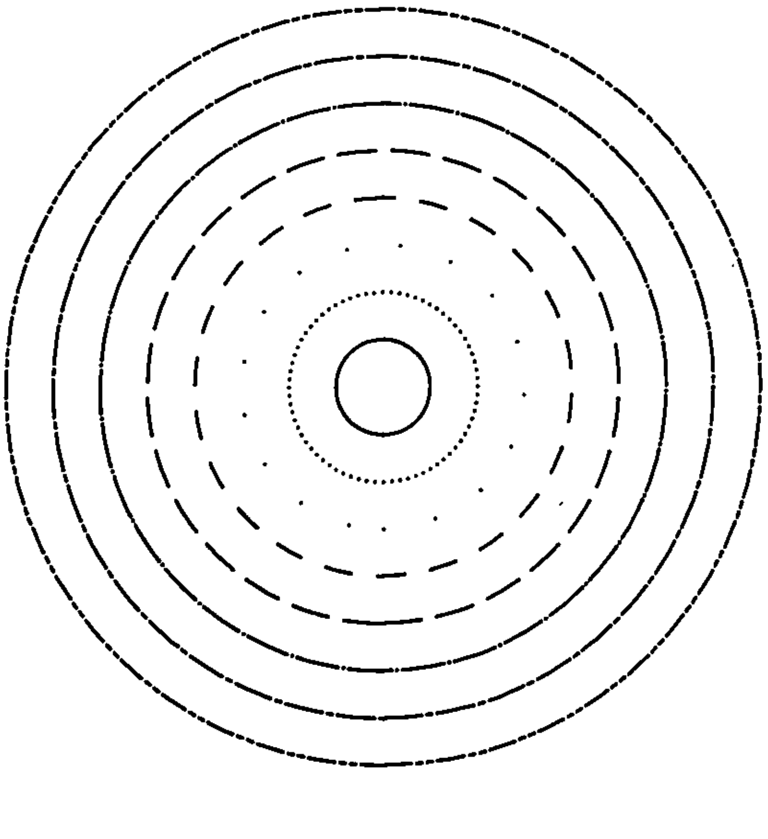

15-Degree chord angle


5-Degree chord angle

Example — Drawing Circles with Different Radii and Line Types

The following instructions draw eight circles with increasing radii and different line types.

```
"IN;SP1;"
"IP1000,1000,6000,6000;"
"SC-100,100,-100,100;"
"PA0,0;LT;CI10;LT0;CI-20;LT1;CI30;"
"LT2;CI-40;LT3;CI50;LT4;CI-60;"
"LT5;CI70;LT6;CI80;"
"SP0;"
```



Arc Instructions, AA and AR

USES: The AA and AR instructions each draw an arc which starts at the current pen position and uses the specified center point. Use the arc absolute instruction, AA, to specify the center point in absolute coordinates. Use the arc relative instruction, AR, to specify the center point in relative coordinates.

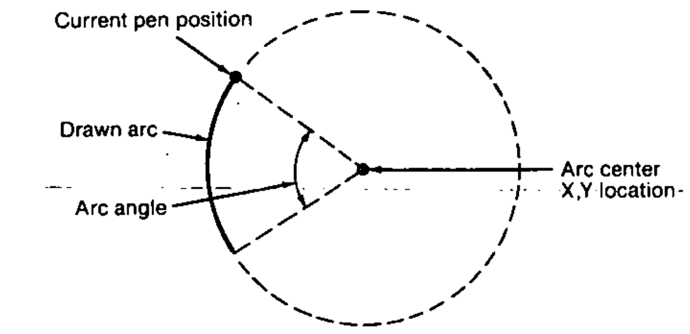
SYNTAX: *AA* X,Y, arc angle(,chord angle) *term*
AR X,Y, arc angle(,chord angle) *term*

Parameter	Format	Range	Default
X,Y coordinates	real	−32 768.0000 to 32 767.9999 current units	none
arc angle	real	−32 768.0000 to 32 767.9999 degrees	none
chord angle	real	0.1 to 180 degrees*	5 degrees

*This is the practical range. The allowable range is −32 768.0000 to 32 767.9999.

EXPLANATION: The parameters are interpreted as follows. For in-depth discussions of absolute and relative plotting, refer to Chapter 2 and to the *Plot Absolute Instruction, PA*, and the *Plot Relative Instruction, PR*, in Chapter 4.

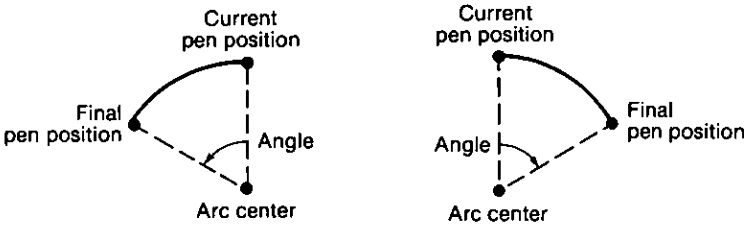
1. **X- and Y-Coordinates.** For AA, the X,Y coordinates specify the absolute position *relative to the origin* to which the pen will move. For AR, the X,Y coordinates (increments) specify the points *relative to the current pen position* to which the pen will move. These coordinates specify the center. The center is the center of the circle that would be drawn if the arc were 360 degrees. Refer to the following illustration.



If scaling is *on*, coordinates are interpreted as user units. Any fractional part of the parameter is used.

If scaling is *off*, specify coordinates in plotter units. Positive numbers are truncated. When you use negative numbers as the parameter, the number is changed to the next more negative number and truncated. For example, both −123.4 and −123.9 become −124.

2. **Arc Angle.** The arc angle is the angle through which the arc is drawn. A positive angle draws counterclockwise from the current pen position, and a negative angle draws clockwise, as shown below.



Positive Angle *Negative Angle*

3. **Chord Angle.** The chord angle parameter governs the smoothness of the arc. As discussed under the *Circle Instruction, CI*, increasing the number of chords (by decreasing the chord angle parameter) generates a smoother arc. The chord angle parameter is interpreted as degrees; the sign of this parameter is ignored.

If you specify a chord angle greater than 180 degrees, then a chord angle of 360 minus the chord angle is used. If you specify a chord angle of 0, then a chord angle of 0.36 is used.

Arcs are drawn starting at the current pen position using the current pen status (up or down) and line type. After the arc has been drawn, the pen position will remain at the end of the arc, rather than the returning to the beginning.

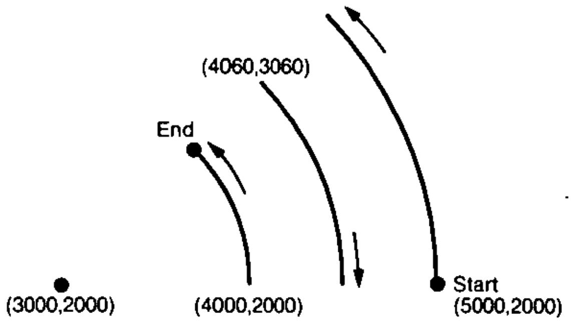
The following table summarizes the error conditions or unexpected results that might occur with the AA and AR instructions.

Condition	Error	Plotter Response
no parameter	none	ignores instruction
1 or 2 parameters	2	ignores instruction
more than 4 parameters	2	uses first 4 parameters
chord angle > 180	none	chord angle $= 360 -$ chord angle
chord angle $= 0$	none	chord angle $= 0.36$
parameter out-of-range	3	ignores instruction

Example — Using the AA Instruction to Draw Arcs

The following instructions illustrate the effects of changing the distance to the center point while keeping the same arc angle. They also show the effect of changing the sign of the arc angle. Note that if the pen is down at the beginning of the instruction, the pen will remain down at the end of each arc. You must lift the pen with the PU instruction in order to move to the beginning of the next arc without drawing a line.

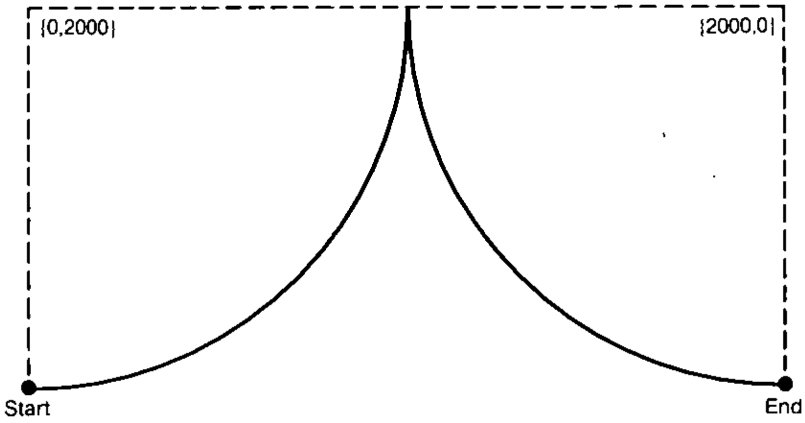
```
"IN;SP1;"
"PA5000,2000;"
"PD;AA3000,2000,45;"
"PU4060,3060;PD;AA3000,2000,-45;"
"PU4000,2000;PD;AA3000,2000,45;"
"SP0;"
```



Example — Using the AR Instruction to Draw Arcs

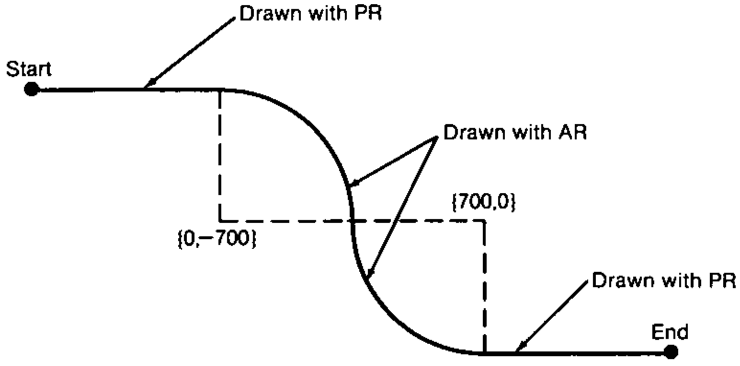
The following instructions draw an arc centered around 0,2000 plotter units relative to the starting pen position, followed by an arc centered around 2000,0 plotter units relative to the new starting pen position. Note that the PD instruction is required to draw the arcs.

```
"IN;SP1;"
"PA10,10;PD;"
"AR0,2000,90;AR2000,0,90;"
"SP0;"
```



The next instructions show the effects of changing both the relative centers of the arcs, and the signs of the angles. The first arc is drawn clockwise around the center (negative angle), whereas the second arc is drawn counterclockwise (positive angle).

```
"IN;SP1;"
"PA10,5000;PD;PR1000,0;"
"AR0,-700,-90;"
"AR700,0,90;PR1000,0;"
"SP0;"
```



Fill Type Instruction, FT

USES: The FT instruction selects the type of area shading used with an FP, RA, RR, or WG instruction. Use this instruction to enhance pie charts, bar charts, and other charts with solid fill, parallel lines (hatching), or cross-hatching.

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

Parameter	Format	Range	Default
fill type	integer	1 to 4	1
spacing	real	1 to 32 767.9999 current units	depends on fill type
angle	real	−32 768.0000 to 32 767.9999 degrees (module 360)	0 degrees

EXPLANATION: The FT instruction can have three parameters: fill type, spacing, and angle. If you omit parameters and this is the first FT instruction since the plotter was turned on or set to default conditions, the plotter uses the default parameter values. If this is not the first FT instruction, and you omit parameters, the plotter uses the parameter values from the previous FT instruction. If you omit all parameters (*FT;*), the plotter uses the default values.

The following paragraphs describe each parameter.

1. **Fill Type.** The four parameter values and corresponding fill types are listed below. For the highest quality solid fill on transparency film, use type 2.

0 or 1	■	Solid. (Bidirectional filling; lines with spacing as defined in the pen thickness instruction, PT, discussed later in this chapter. The fill type spacing parameter will be ignored.)
2	■	Solid. (Unidirectional filling; lines with spacing as defined in the PT instruction. The fill type spacing parameter will be ignored.)

3	▤	Parallel lines. (Always bidirectional for solid line type; unidirectional for dotted or dashed line types.)
4	▦	Cross-hatch. (Always bidirectional for solid line type; unidirectional for dotted or dashed line types.)

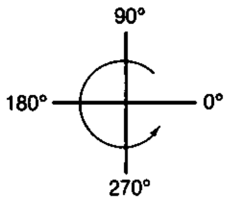
All fill types are drawn using the current pen. Fill types 3 and 4 use the current line type, whereas solid-fill types 1 and 2 always use a solid line. In this case, any line type specified by the LT instruction is ignored until the solid fill is completed.

2. **Spacing.** The spacing parameter is ignored for solid-fill types 1 and 2. Spacing is determined by the PT instruction. For fill types 3 and 4, the spacing parameter is the distance in current units between parallel lines in the fill area.

   For fill types 3 and 4, the spacing parameter is interpreted as plotter units if scaling is off or user units if scaling is on. When scaling is on, the spacing parameter is interpreted using units along the X-axis.

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

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

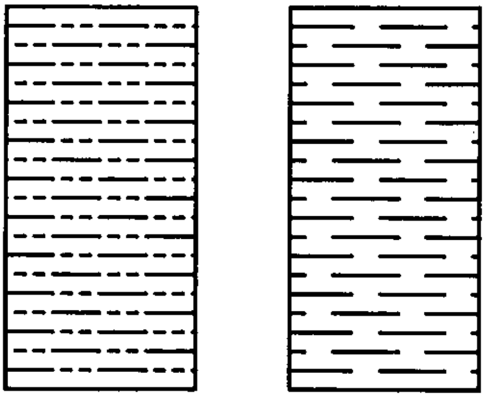


The following table summarizes the error conditions or unexpected results that might occur with the FT instruction.

Condition	Error	Plotter Response
no parameters	none	establishes solid fill at an angle of 0 degrees
1 parameter	none	establishes specified fill type with spacing and angle of previous FT instruction
2 parameters	none	establishes specified fill type and spacing with angle of previous FT instruction
more than 3 parameters	2	uses first 3 parameters
parameter out-of-range	3	ignores instruction

Example — Effects of Line Types

The following BASIC program shows the effects of using various line types to fill a rectangular area. For more examples of area fill, refer to the following sections, later in this chapter: the *Wedge Instructions, WG and EW*; the *Absolute Rectangle Instructions, RA and EA*; the *Relative Rectangle Instructions, RR and ER*; the *Polygon Mode Instruction, PM;* and the *Fill Polygon Instruction, FP.*



```
10 REM  Insert configuration statement here
20 PRINT #1, "IN;SP1;"
30 PRINT #1, "PA0,0;LT6;FT3,100,0;"
40 PRINT #1, "RA1000,2000;EA1000,2000;"
50 PRINT #1, "PA1500,0;"
60 PRINT #1, "LT3;RA2500,2000;"
70 PRINT #1, "EA2500,2000;"
80 PRINT #1, "SP0;"
90 END
```

Program Explanation

10 establishes HP-IB or RS-232-C interface conditions

20 initializes the plotter; selects pen 1

30 sets the starting pen position; selects line type 6; selects fill type 3 with lines spaced every 100 plotter units at an angle of 0 degrees

40 defines, fills, and edges a rectangle

50 sets the starting pen position for the next rectangle

60 selects line type 3; defines and fills a rectangle using the same fill pattern specified in line 30

70 outlines the rectangle

80 returns the pen to the carousel

Pen Thickness Instruction, PT

USES: The PT instruction determines the spacing between the lines used to fill rectangles, wedges, and polygons when you select solid fill. Use this instruction with the FT, RA, RR, WG, and FP instructions to produce a high-quality solid fill.

SYNTAX: *PT* pen thickness *term*
or
PT *term*

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

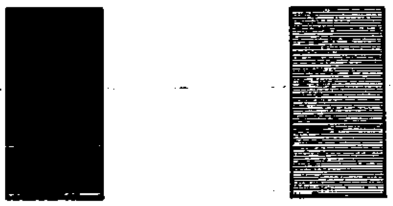
EXPLANATION: The pen thickness parameter represents the physical pen-tip width in millimetres. Following is a list of typical pen thicknesses. Specify the number that corresponds to the pen you are using. The spacing between the lines will be 0.5 × pen thickness.

Pen Type	Thickness
Narrow felt-tip (paper or transparency)	0.3 mm
Wide felt-tip (transparency)	0.6 mm
Wide felt-tip (paper)	0.7 mm

The pen thickness determines the spacing of lines needed to produce a solid fill. The thicker the pen, the less strokes needed to fill an area. If your solid fill has gaps showing between the lines, use a smaller pen thickness parameter. If you want to plot faster with a less dense fill, use a larger pen thickness parameter.

NOTE: Pen tip width is an average width. Adjustments might be necessary to compensate for slight differences in pen tips. ■

Both of the following illustrations are produced by the instructions *PT . 7 ; FT 1 ;*. The difference in density is a result of the two different pen widths used. The illustration on the left was drawn using a 0.7-mm wide pen; *PT . 7 ;* produces the optimum solid fill for a 0.7-mm wide pen. The illustration on the right was drawn using a 0.3-mm wide pen; the plotting is faster than if *PT . 3 ;* had been specified, but the solid fill is less dense.



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

The following table summarizes the error conditions or unexpected results that might occur with the PT instruction.

Condition	Error	Plotter Response
more than 1 parameter	2	uses first parameter
parameter out-of-range	3	ignores instruction
new pen selected	none	establishes 0.3-mm pen thickness

Wedge Instructions, WG and EW

USES: The fill wedge instruction, WG fills any wedge (sector) of a circle; the edge wedge instruction, EW, outlines any wedge. Use these instructions to produce sectors of a pie chart.

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

Parameters	Format	Range	Default
radius	real	−32 768.0000 to 32 767.9999 degrees	none
start angle	real	−32 768.0000 to 32 767.9999 degrees	none
sweep angle	real	−32.768.0000 to 32 767.9999 degrees	none
chord angle	real	0.1 to 180 degrees*	5 degrees

*This is the practical range. The allowable range is −32 768.0000 to 32 767.9999.

EXPLANATION: The WG and EW instructions are interpreted by the plotter in the same way. The only difference is that the WG instruction produces a filled wedge, whereas the EW instruction produces an outlined wedge. It is good programming practice to outline areas *after* they have been filled; this creates a clean edge.

The following paragraphs describe each parameter.

1. **Radius.** The radius determines the size of the wedge. It specifies the distance from the current pen position (which becomes the center of the wedge), to any point on the circumference of the wedge.

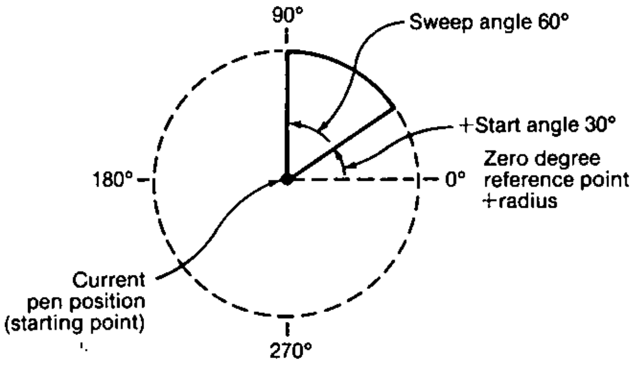
   The radius is in plotter units if scaling is off and user units if scaling is on. If user units are not the same size in the X-and Y-directions, distorted wedges will be drawn. The sign of the radius determines the zero-degree reference point for the start angle and sweep angle, as shown on the next page.

   When scaling is *on*, the radius parameter is interpreted as user units. Any fractional part of the parameter is used.

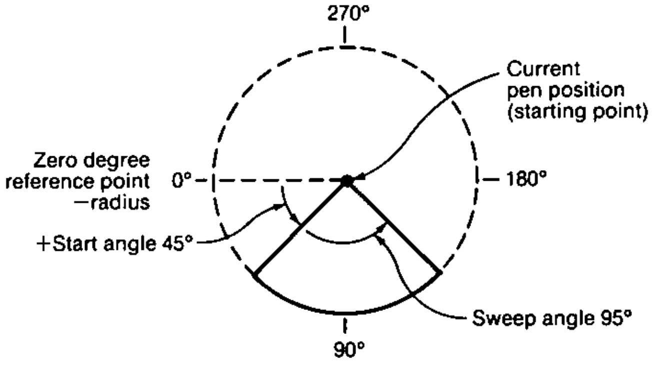
   When scaling is *off*, use plotter units for the radius parameter. Positive numbers are truncated. When you use negative numbers as the parameter, the number is changed to the next more negative number and truncated. For example, both −123.4 and −123.9 become −124.

2. **Start Angle.** The start angle specifies the start of the arc. 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.

3. **Sweep Angle.** The 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.


90°
Sweep angle 60°
+Start angle 30°
Zero degree reference point +radius
0°
180°
Current pen position (starting point)
270°

Positive Radius


270°
Current pen position (starting point)
Zero degree reference point −radius
0°
180°
+Start angle 45°
Sweep angle 95°
90°

Negative Radius

4. **Chord Angle.** As discussed under the CI instruction, increasing the number of chords (by decreasing the chord angle parameter) generates a smoother arc. The chord angle parameter is interpreted as degrees. The sign of this parameter is ignored. If the chord angle is greater than 180, then a chord angle of 360 minus the chord angle is used. If you specify a chord angle of zero, a chord angle of 0.36 is used.

   NOTE: If you have an RS-232-C plotter, the number of chords per arc is limited to 103 when you use the default size of the polygon buffer. To use a larger number of chords, use the ESC.T instruction, explained in Chapter 12, to increase the size of the polygon buffer. ■

The WG instruction defines and fills a wedge using the current pen and fill type (and line type if the fill is not solid). The EW instruction defines and edges a wedge using the current pen and a solid line.

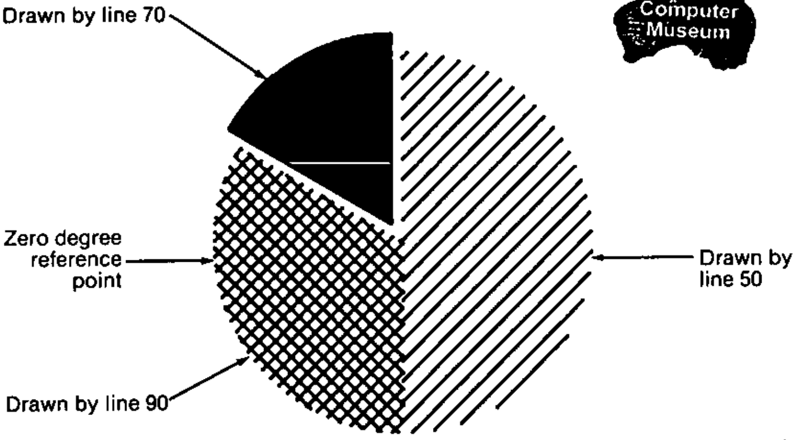
With both WG and EW, the starting point for the wedge is the current pen position, which is the center of the circle that would be drawn if the wedge were 360 degrees. The WG and EW instructions include an automatic pen down. Upon completion of the wedge, the pen returns to the previous pen position and the pen status is restored.

The following table summarizes the error conditions or unexpected results that might occur with the WG and EW instructions.

Condition	Error	Plotter Response
no parameters	none	ignores instruction
fewer than 3 parameters	2	ignores instruction
more than 3 parameters	2	uses first 4 parameters
chord angle > 180	none	chord angle = 360 − chord angle
chord angle = 0	none	chord angle = 0.36
parameter out-of-range	3	ignores instruction
polygon buffer overflow	7	with WG, does not fill the wedge; with EW, edges buffer contents. All data that overflows is lost.

Example — Defining and Filling Wedges Using the WG Instruction

The following BASIC program illustrates how to use the WG instruction to create a pie chart with one wedge offset for emphasis.



```
10 REM  Insert configuration statement here
20 PRINT #1; "IN;SP1;"
30 PRINT #1, "PA7000,2000;"
40 PRINT #1, "FT3,75,45;"
50 PRINT #1, "WG-1000,90,180;"
60 PRINT #1, "SP3;PR-60,100;FT1,0,0;"
70 PRINT #1, "WG-1000,270,60;"
80 PRINT #1, "SP4;PA7000,2000;FT4,60,45;"
90 PRINT #1, "WG-1000,330,120;"
100 PRINT #1, "SP0;"
110 END
```

Program Explanation

10 establishes HP-IB or RS-232-C interface conditions

20 initializes the plotter; selects pen 1

30 sets the starting pen position (center of the pie)

40 selects fill type 3 with lines spaced every 75 plotter units at an angle of 45 degrees

50 defines and fills a wedge, starting at 90 degrees, sweeping counterclockwise through 180 degrees

60 selects pen 3; sets a new pen position to offset the wedge; selects fill type 1 at a 0-degree angle with spacing defined by default PT .3 (established by IN in line 20)

70 defines and fills a wedge with the same zero-degree reference point and radius, starting at 270 degrees, sweeping counterclockwise through 60 degrees

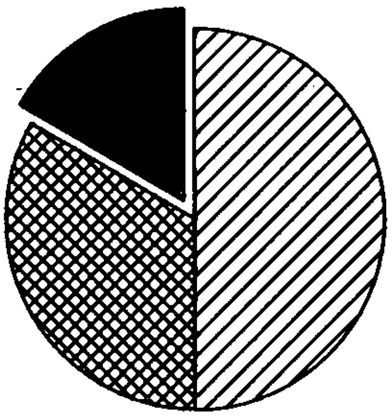
80 selects pen 4; sets the pen position back to the original center of the pie; selects fill type 4 with lines spaced every 60 plotter units at an angle of 45 degrees

90 defines and fills a wedge with the same zero-degree reference point and radius, starting at 330 degrees, sweeping counterclockwise through 120 degrees

100 returns the pen to the carousel

Example — Outlining the Wedges Using the EW or EP Instructions

There are two methods for outlining a wedge. One uses the edge wedge instruction, EW, and the other uses the edge polygon instruction, EP. This section presents both methods, along with reasons that you might use each method. Both methods produce the following plot.



A. To edge the wedges in the previous program using the EP instruction, add these lines to that program.

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

75 PRINT #1, "EP;"

95 PRINT #1, "EP;"
```

Notice that an EP instruction follows each WG instruction. This is because EP outlines whatever currently resides in the polygon buffer. Only one wedge resides in the buffer at a given time; therefore EP must be executed following each wedge. The EP instruction is very simple and efficient. Refer to the *Edge Polygon Instruction, EP,* later in this chapter.

B. To edge the wedges in the previous program using the EW instruction, add these lines to that program. (If you have already added lines 55, 75, and 95 in method A, replace them with the following lines.)

```
55 PRINT #1, "EW-1000,90,180;"

75 PRINT #1, "EW-1000,270,60;"

95 PRINT #1, "EW-1000,330,120;"
```

Notice that the EW instruction requires parameters to define the wedge before it can outline it. The EW instruction is more flexible than the EP instruction, because you can place EW instructions anywhere in the program. Place EW after a WG instruction for a smooth edge. You can also execute an EW instruction alone (without an accompanying WG instruction).

Absolute Rectangle Instructions, RA and EA

USES: The fill rectangle absolute instruction, RA, defines and fills a rectangle; the edge rectangle absolute instruction, EA, defines and outlines a rectangle. Both instructions use absolute coordinates. Use the RA and EA instructions to create charts that require rectangles; for example, bar charts, flow charts, and organization charts. (To fill and outline rectangles using relative coordinates, refer to the *Relative Rectangle Instructions, RR and ER*, next in this chapter.)

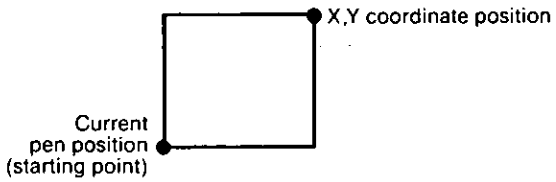
SYNTAX: *RA* X,Y *term*
EA X,Y *term*

Parameter	Format	Range	Default
X,Y coordinates	real	−32 768.0000 to 32 767.9999 current units	none

EXPLANATION: The current pen position is the starting point of the rectangle. The X,Y coordinates specify the opposite corner of the rectangle, as shown in the following illustration.

When scaling is *on*, coordinates are interpreted as user units. Any fractional portion of a parameter is used.

When scaling is *off*, coordinates are interpreted as plotter units. For *positive* numbers, any fractional portion of a parameter is discarded. For example, both 123.4 and 123.8 become 123. For *negative* numbers, the fractional portion is discarded and the integer is changed to the next more negative number. For example, both −123.4 and −123.9 become −124.



NOTE: The illustration shows the current pen position in the lower-left corner and the X,Y parameters in the upper-right corner. However, both positions can be in any corner as long as the X,Y parameters are in the corner opposite the current pen position. ■

The RA instruction defines and fills a rectangle using the current pen and fill type (and the current line type if the fill is not solid). The EA instruction defines and edges a rectangle using the current pen and a solid line. Both the RA and EA instructions include an automatic pen down. When the rectangle is complete, the pen returns to the starting point and restores the pen status (up/down).

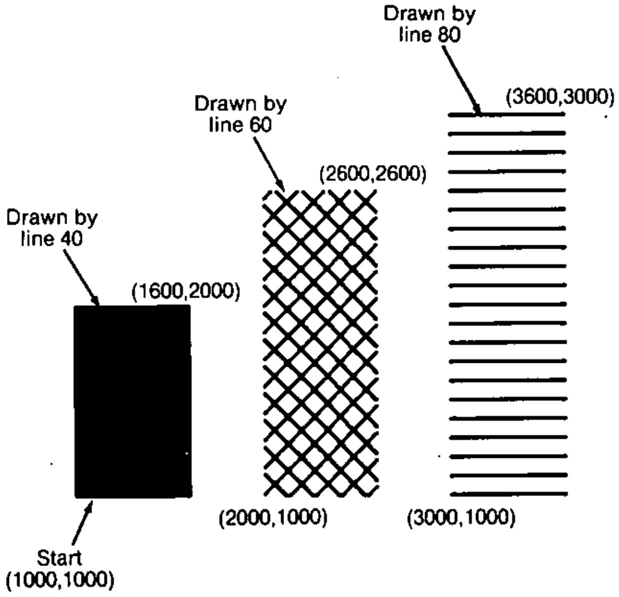
NOTE: The RA and EA instructions clear the polygon buffer and then use the buffer for the rectangle definition before filling or edging the rectangle. A rectangle requires enough buffer space to hold five points. The default buffer size is sufficient; however, if you wish to change the polygon buffer size, refer to *The Polygon Buffer* at the end of this chapter. ■

The following table summarizes the error conditions or unexpected results that might occur with the RA and EA instructions.

Condition	Error	Plotter Response
no parameters	none	ignores instruction
1 parameter	2	ignores instruction
more than 2 parameters	2	uses first 2 parameters
parameter out-of-range	3	ignores instruction
polygon buffer overflow	7	if RA, does not fill the rectangle; if EA, edges buffer contents. All data that overflows is lost.

Example — Drawing Rectangles Using Absolute Coordinates

The following BASIC program demonstrates the RA instruction used with the FT instruction to create rectangles such as those you might use in a bar chart. (To create the same rectangles with relative coordinates, refer to the *Relative Rectangle Instructions, RR and ER*, next in this chapter.) Following this program is another example that shows two methods for outlining rectangles.



```
10 REM  Insert configuration statement here
20 PRINT #1, "IN;SP1;"
30 PRINT #1, "PA1000,1000;"
40 PRINT #1, "FT1;RA1600,2000;"
50 PRINT #1, "PA2000,1000;"
60 PRINT #1, "FT4,100,45;RA2600,2600;"
70 PRINT #1, "PA3000,1000;"
80 PRINT #1, "FT3,100,0;RA3600,3000;"
90 PRINT #1, "SP0;"
100 END
```

Program Explanation

10 establishes HP-IB or RS-232-C interface conditions

20 initializes the plotter; selects pen 1

30 sets the starting pen position for the first rectangle

40 selects fill type 1; defines and fills the rectangle (with default PT .3 established by IN in line 20)

50 sets the starting pen position for the next rectangle

60 selects fill type 4 with lines spaced every 100 plotter units at an angle of 45 degrees; defines and fills the rectangle

70 sets the starting pen position for the next rectangle

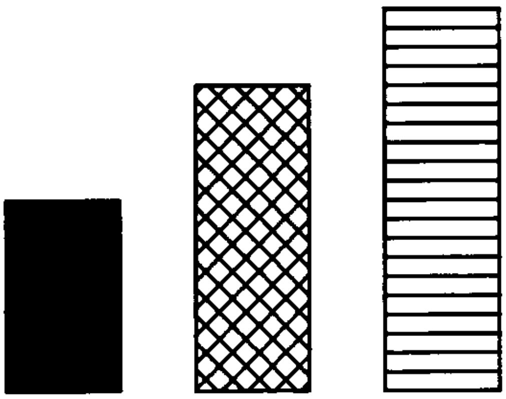
80 selects fill type 3 with lines spaced every 100 plotter units at an angle of 0 degrees; defines and fills the rectangle

90 returns the pen to the carousel

Example — Outlining Rectangles Using Absolute Coordinates

A good graphics programming habit is to outline areas after they have been filled. This creates a clean edge.

There are two methods for outlining a rectangle using absolute coordinates. One method uses EA; the other uses the EP instruction. Both methods produce the following plot; an explanation follows.



A. To edge the rectangles using EP, add these lines to the preceding program.

```
45 PRINT #1, "EP;"

65 PRINT #1, "EP;"

85 PRINT #1, "EP;"
```

Notice that an EP instruction follows each RA instruction. This is because EP outlines whatever currently resides in the polygon buffer. Refer to the *Edge Polygon Instruction, EP,* later in this chapter, for details.

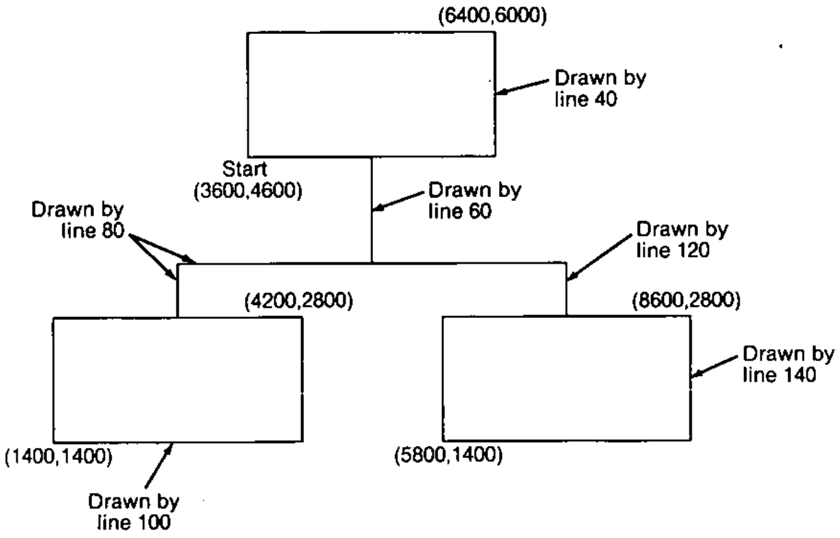
B. To edge the rectangles using EA, add these lines to the preceding program. (If you used method A, above, replaces lines 45, 65, and 85 with the following lines.)

```
45 PRINT #1, "EA1600,2000;"

65 PRINT #1, "EA2600,2600;"

85 PRINT #1, "EA3600,3000;"
```

Notice that the EA instruction requires parameters to define the rectangle before outlining it (unlike the EP instruction). The EA instruction is more flexible than the EP instruction, because you can place EA instructions anywhere in the program. You can also execute an EA instruction alone (without an accompanying RA instruction), as shown in the following example.

Example — Drawing an Organization Chart Using Absolute Coordinates

The following BASIC program illustrates how you can use the EA instruction to set up the outline of an organization chart.



```
10 REM  Insert configuration statement here
20 PRINT #1, "IN;SP1;"
30 PRINT #1, "PA3600,4600;"
40 PRINT #1, "EA6400,6000;"
50 PRINT #1, "PA5000,4600;"
60 PRINT #1, "PD5000,3400;"
70 PRINT #1, "PU7200,3400;"
80 PRINT #1, "PD2800,3400,2800,2800;"
90 PRINT #1, "PU4200,2800;"
100 PRINT #1, "EA1400,1400;"
110 PRINT #1, "PA7200,3400;"
120 PRINT #1, "PD7200,2800;"
130 PRINT #1, "PU8600,2800;"
140 PRINT #1, "EA5800,1400;"
150 PRINT #1, "SP0;"
160 END
```

Program Explanation

10 establishes HP-IB or RS-232-C interface conditions

20 initializes the plotter; selects pen 1

30 sets the starting pen position using absolute coordinates

40 defines and outlines the top rectangle

50 moves to the starting pen position for the first vertical connecting line and establishes absolute coordinates for all subsequent PU and PD instructions

60 draws the first vertical connecting line

70 moves to the right endpoint of the horizontal connecting line

80 draws the horizontal connecting line, followed by the left vertical connecting line

90 moves to the starting pen position for the left rectangle

100 defines and outlines the left rectangle

110 moves to the starting pen position for the right vertical connecting line

120 draws the right vertical connecting line

130 moves to the starting pen position for the right rectangle

140 defines and outlines the right rectangle

150 returns the pen to the carousel

Relative Rectangle Instructions, RR and ER

USES: The fill rectangle relative instruction, RR, defines and fills a rectangle; the edge rectangle relative instruction, ER, defines and outlines a rectangle. Both instructions use relative coordinates. Use RR and ER to create charts that require rectangles; for example, bar charts, flow charts, and organization charts. (To fill and outline rectangles using absolute coordinates, refer to the *Absolute Rectangle Instructions, RA and EA*, earlier in this chapter. Review that section for details of how rectangle instructions are interpreted by the plotter, and for error conditions.)

SYNTAX: *RR* X,Y *term*
ER X,Y *term*

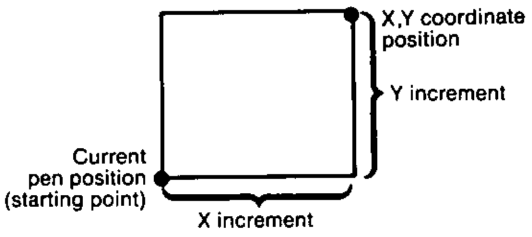
Parameter	Format	Range	Default
X,Y coordinates	real	−32 768.0000 to 32 767.9999 current units	none

EXPLANATION: The only difference between the RA/EA instructions and the RR/ER instructions is that RA and EA use absolute coordinates to define a rectangle, whereas RR and ER use relative coordinates. The use of relative coordinates by the RR/ER instructions is shown next.

When scaling is *on*, coordinates are interpreted as user units. Any fractional portion of a parameter is used.

When scaling is *off*, coordinates are interpreted as plotter units. For *positive* numbers, any fractional portion of a parameter is discarded. For example, both 123.4 and 123.8 become 123. For *negative* numbers, the fractional portion is discarded and the integer is changed to the next more negative number. For example, both −123.4 and −123.9 become −124.

NOTE: Sending decimal parameters when scaling is off may cause unexpected results. Although the pen will move only in integer increments, the plotter still monitors the pen position using the complete decimal number. This can affect future pen positions if you specify a series of pen moves. ■

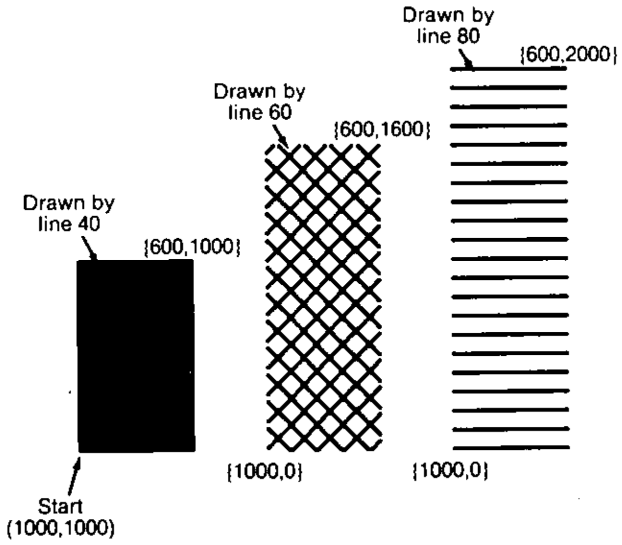


NOTE: The preceding illustration shows the current pen position in the lower-left corner and the X,Y parameters in the upper-right corner. However, both positions can be in any corner as long as the X,Y parameters are in the corner opposite the current pen position. ■

Example — Drawing Rectangles Using Relative Coordinates

The following example uses relative rectangle instructions to draw the same plots presented previously for the absolute rectangle instructions. Compare this program with the previous program to understand the differences between the coordinates used.

Program explanations are not included here, because the explanations are the same as for the previous program, except for the type of coordinates used.



```
10 REM  Insert configuration statement here
20 PRINT #1, "IN;SP1;"
30 PRINT #1, "PA1000,1000;"
40 PRINT #1, "FT1;RR600,1000;"
50 PRINT #1, "PR 1000,0;"
60 PRINT #1, "FT4,100,45;RR600,1600;"
70 PRINT #1, "PR 1000,0;"
80 PRINT #1, "FT3,100,0;RR600,2000;"
90 PRINT #1, "SP0;"
100 END
```

Example — Outlining Rectangles Using Relative Coordinates

This example provides two methods of outlining rectangles, using relative coordinates: one uses EP; the other uses ER. To outline the rectangles shown in the previous example, add the the following program lines to the program listed in that example, *Drawing Rectangles Using Absolute Coordinates.*

To outline using EP, add these lines to the previous program.

```
45 PRINT #1, "EP;"

65 PRINT #1, "EP;"

85 PRINT #1, "EP;"
```

To outline using ER, add these lines to the previous program.

```
45 PRINT #1, "ER600,1000;"

65 PRINT #1, "ER600,1600;"

85 PRINT #1, "ER600,2000;"
```

Example — Drawing an Organization Chart Using Relative Coordinates

This example uses relative coordinates to draw the organization chart shown with the EA instruction. Compare this program with the example program shown previously with the EA instruction, to understand the differences between the coordinates used. A program explanation is not included here, because the explanation is the same as for the previous program, except for the types of coordinates used.

To draw the organization chart shown with the EA instruction, using ER instead of EA, enter and run the following program.

```
10 REM Insert configuration statement here
20 PRINT #1, "IN;SP1;"
30 PRINT #1, "PA3600,4600;"
40 PRINT #1, "ER2800,1400;"
50 PRINT #1, "PR1400,0;"
60 PRINT #1, "PD0,-1200;"
70 PRINT #1, "PU2200,0;"
80 PRINT #1, "PD-4400,0,0,-600;"
90 PRINT #1, "PU1400,0;"
100 PRINT #1, "ER-2800,-1400;"
110 PRINT #1, "PR3000,600;"
120 PRINT #1, "PD0,-600;"
130 PRINT #1, "PU1400,0;"
140 PRINT #1, "ER-2800,-1400;"
150 PRINT #1, "SP0;"
160 END
```

Polygon Mode Instruction, PM

USES: The PM instruction places the plotter in polygon definition mode. In this mode, you can define polygons using other HP-GL instructions. These instructions (and associated X,Y coordinates) are stored in the polygon buffer (explained at the end of this chapter). In order to draw a polygon, you must fill it with the fill polygon instruction, FP, and/or outline it with the edge polygon instruction, EP. Both FP and EP are explained later in this chapter.

Use polygon mode when designing shapes such as block letters, logos, surface charts, or any filled or outlined area.

SYNTAX: *PM* n *term*

Parameter	Format	Range	Default
n	integer	0 to 2	0

EXPLANATION: The PM instruction accepts only three parameters, as follows. The subsequent paragraphs explain how to use each parameter, and provides examples. The order in which you use these parameters is important.

0	Clears the polygon buffer and enters polygon mode
1	Closes current polygon (or subpolygon)
2	Closes current polygon (or subpolygon) and exits polygon mode

Use *PM0;* to clear the polygon buffer and enter polygon mode. While in polygon mode, the plotter stores certain HP-GL instructions in its polygon buffer, where they can be accessed by the FP and EP instructions to draw the polygon. The following table lists those instructions.

Stored in Polygon Buffer	
PM	Polygon mode instruction
PA/PR	Plotting instructions
PU/PD	Pen instructions
AA/AR	Arc instructions
CI	Circle instruction

The only other HP-GL instructions you can use while in polygon mode are the initialize instruction, IN, and all HP-GL output instructions.* The plotter stores the instructions shown in the preceding table in the polygon buffer; IN and the HP-GL output instructions are executed. Exit polygon mode to execute other HP-GL instructions.

*OA, OC, OD, OE, OF, OH, OI, OO, OP, OS, OW

NOTE: While in polygon mode, the CI instruction is interpreted differently then other HP-GL instructions. Refer to the example *Using the CI Instruction in Polygon Mode*, later in this chapter, for specifics. ■

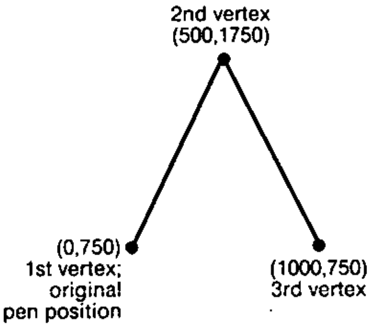
When you define polygons, the pen position before *PM 0;* is the first point (vertex) of the polygon, and the first point stored in the polygon buffer. For example, if you execute the following instructions, *PA 0, 1750; PM 0;*, the X,Y coordinates *0,1750* will be the first point of your polygon. Each subsequent pair of coordinates defines a point of the polygon.

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

NOTE: The *PM 0;* or *PM;* instruction clears the polygon buffer and then uses the buffer to hold the points which define the polygon. The default buffer size is sufficient for a polygon with approximately 86 points (vertices). To prevent polygon buffer overflow (error 7), you must allocate enough memory to the polygon buffer to hold the polygon. For full details on determining the proper memory allocation, refer to *The Polygon Buffer* at the end of this chapter. ■

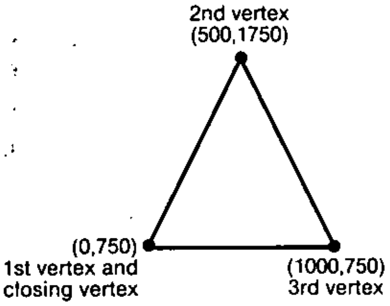
It is good programming practice to "close" the polygon before exiting polygon mode. Closing a polygon means adding the final vertex that defines a continuous shape; the last coordinates will be the same as the first. Refer to the following illustration and sequence of instructions.

```
"PU 0,750;PM0;"
"PD500,1750,1000,750;EP;"
```



Open Polygon

```
"PA 0,750; PM0;"
"PD500,1750,1000,750,0,750;PM2;EP;"
```



Closed Polygon

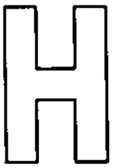
If you have not closed the polygon, executing *PM1;* or *PM2;* forces closure by adding a point to close the polygon.

Use *PM1;* to close the current polygon (or subpolygon). When you use *PM1;*, the point after *PM1;* becomes the first point of the next subpolygon. This point is always moved to with the pen up, regardless of the current pen status. Each subsequent coordinate pair after *PM1;* defines a point of the subpolygon.

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, forcing closure. Unless you exit polygon mode, the plotter will ignore all HP-GL instructions *except* IN and all HP-GL output instructions.

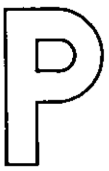
The following paragraphs summarize the sequence you should use when defining one polygon, such as the letter **H,** or a series of subpolygons, such as the letter **P** (the outine and the "hole"). **It is important to use this sequence when defining polygons.** The program example following this section uses this sequence.

To define a polygon (e.g. the letter H):



1. Move the pen to the starting location.
2. Execute *PM0;* to enter polygon mode.
3. Define the shape of the polygon (using instructions such as PA or PR).
4. Execute *PM2;* to exit polygon mode.
5. Fill or edge the polygon using the FP and/or EP instruction(s).

To define a series of subpolygons (e.g. the letter P):



1. Move the pen to the starting location.
2. Execute *PM0;* to enter polygon mode.
3. Define the first polygon (using instructions such as PA or PR).
4. Execute *PM1;* to end the first subpolygon.
5. Define the second subpolygon, using HP-GL instructions.
6. Execute *PM1;* to end the second subpolygon.

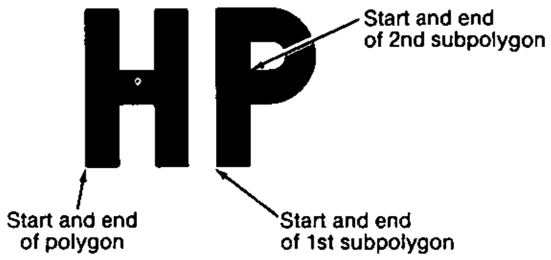
   Repeat steps 5 and 6 for any additional subpolygons.
7. Exit polygon mode with *PM2;*.
8. Fill or edge the polygon using the FP and/or EP instruction(s).

The following table summarizes the error conditions or unexpected results that might occur with the PM instruction.

Condition	Error	Plotter Response
no parameter	none	clears the polygon buffer and enters polygon mode
parameter out-of-range	3	ignores instruction
use of an HP-GL instruction other than IN or an output instruction while in polygon mode	1	ignores illegal instruction
too many points; polygon buffer overflow	7	ignores all points that overflowed

Example — Creating Block Letters in Polygon Mode

The following BASIC program demonstrates how to define the letter H as a polygon, and the letter P as two subpolygons. When designing polygons, drawing them first on grid paper can help you determine the proper coordinates. The letters in this program are defined using relative coordinates, starting at the lower-left corner of each letter and moving up and clockwise around the letter. Remember, the plotter will not draw the polygons until instructed to fill or edge them. For details, refer to the *Fill Polygon Instruction, FP*, and the *Edge Polygon Instruction, EP*, next in this chapter.


Start and end of 2nd subpolygon
Start and end of polygon
Start and end of 1st subpolygon

```
10 REM  Insert configuration statement here
20 PRINT #1, "IN;SP1;"
30 PRINT #1, "PA0,0;PM0;PR;"
40 PRINT #1, "PD0,1000,200,0,0,-400;"
50 PRINT #1, "PD250,0,0,400,200,0,0,-1000,-200,0;"
60 PRINT #1, "PD0,400,-250,0,0,-400,-200,0;"
70 PRINT #1, "PU;PM2;"
80 PRINT #1, "FP;EP;PU850,0;"
90 PRINT #1, "PM0;PD0,1000,325,0;"
100 PRINT #1, "AR0,-300,-180;"
110 PRINT #1, "PD-125,0,0,-400,-200,0;PM1;"
120 PRINT #1, "PU200,600;PD0,200,150,0;"
130 PRINT #1, "AR0,-100,-180;"
140 PRINT #1, "PD-150,0;PU;PM2;"
150 PRINT #1, "FP;EP;"
160 PRINT #1, "SP0;"
170 END
```

Program Explanation

10 establishes HP-IB or RS-232-C interface conditions

20 initializes the plotter; selects pen 1

30 sets the initial pen position; clears the polygon buffer and enters polygon mode; specifies relative coordinates

40 lowers the pen and defines three points of the letter H

50 defines the next five points of the H

60 defines the last four points of the H (including the *closing* vertex)

70 lifts the pen; exits polygon mode

80 fills the H using default values (solid, bidirectional fill); outlines the H; moves with the pen up to the starting position for the outside of the letter P

90 clears the polygon buffer and enters polygon mode; lowers the pen and defines two points of the outline of the P (beginning from the lower-left corner)

100 defines the arc of the P in relative coordinates, and a counterclockwise angle of 180 degrees

110 defines the last three points of the outline of the P (including the closing point at the lower-left corner); closes this subpolygon

120 moves with the pen up to the starting position for the inside of the P; defines three points of the inside of the P

130 defines the arc of the P

140 defines the final (closing) point of the inside of the P; lifts the pen; exits polygon mode

150 fills the P; outlines the P

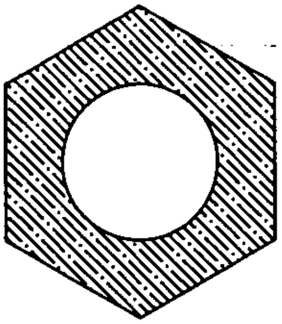
160 returns the pen to the carousel

Example — Using the CI Instruction in Polygon Mode

A CI instruction is interpreted slightly differently from the other HP-GL instructions that you can use in polygon mode. When the CI instruction is used in polygon mode, a circle is always considered to be a complete subpolygon; the plotter treats CI as if it were preceded and followed by PM1. Thus, the first coordinate points entered after a CI instruction become the first point of the *next* subpolygon. When a circle is to be the first element of a polygon, move the pen to the center point *before* executing PM0.

Since PM1 closes the current subpolygon (if it is open), this usually adds another point to the subpolygon and changes the pen position. Close the current subpolygon before executing a CI instruction if you do not want the pen position (and location of the circle) to change.

The following BASIC program demonstrates the CI instruction used to define the two subpolygons needed to draw a hexagonal nut.



```
10 REM  Insert configuration statement here
20 PRINT #1, "IN;SP1;"
30 PRINT #1, "PA3000,3000;"
40 PRINT #1, "PM0;CI1000,60;CI500;PM2;"
50 PRINT #1, "LT4;FT3,50,45;FP;LT;EP;"
60 PRINT #1, "SP0;"
70 END
```

Program Explanation

10 establishes HP-IB or RS-232-C interface conditions

20 initializes the plotter; selects pen 1

30 sets the initial pen position (the center of the circle)

40 clears the polygon buffer and enters polygon mode; specifies a circle with a radius of 1000 plotter units and chord angle of 60 degrees as the first subpolygon (the outer hexagon); specifies a circle with a radius of 500 plotter units and default chord angle of 5 degrees as the second subpolygon (the inner circle); exits polygon mode

50 selects line type 4; defines fill type 4 with lines spaced every 50 plotter units at an angle of 45 degrees; fills the polygon; selects a solid line type; edges the polygon

60 returns the pen to the carousel

Edge Polygon Instruction, EP

USES: The EP instruction outlines the polygon that is currently stored in the polygon buffer. Use this instruction to edge polygons that have been defined while in polygon mode and also with polygons defined with the rectangle and wedge instructions (WG, RA, and RR).

SYNTAX: *EP term*

EXPLANATION: The EP instruction outlines any polygon that has been previously placed in the polygon buffer. The EP instruction 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. Upon completion of the EP instruction, the original pen position and status (up/down) are restored.

For examples using the EP instruction, refer to the RA, RR, WG, and PM instructions in this chapter.

The following table summarizes the error conditions or unexpected results that might occur with the EP instruction.

Condition	Error	Plotter Response
previous PM, RA, RR, or WG instruction overflowed the polygon buffer	none*	edges between those points that remain in the buffer
no polygon previously defined	none	ignores instruction
1 or more parameters	2	edges the currently stored polygon

*Error 7 would have occured at the time the buffer overflowed.

Fill Polygon Instruction, FP

USES: The FP instruction fills the polygon that is currently stored in the polygon buffer, using the fill type specified in the FT instruction. Use the FP instruction to fill polygons that have been defined while in polygon mode and also with polygons defined with the rectangle and wedge instructions.

SYNTAX: *FP term*

EXPLANATION: The FP instruction fills the polygon that has been previously placed in the polygon buffer. The FP instruction accesses the data in the polygon buffer without clearing the buffer or changing the data in any way.

The polygon is filled using the current pen, fill type, and line type. If subpolygons are defined, the FP instruction fills *alternating* areas, beginning with the outside area. (To see how this works, refer to the examples at the end of this section.) Upon completion of the FP instruction, the original pen position and status (up/down) are restored.

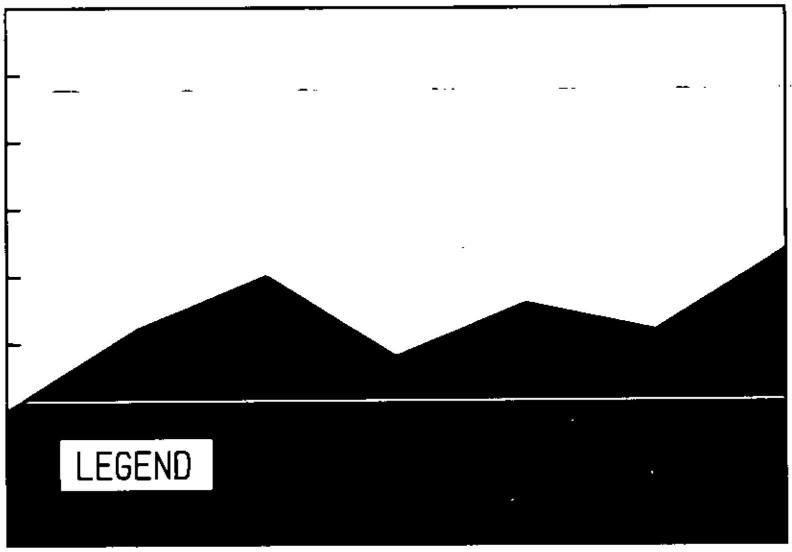
The following table summarizes the error conditions or unexpected results that might occur with the FP instruction.

Condition	Error	Plotter Response
previous PM, RA, RR, or WG instruction overflowed the polygon buffer	none*	ignores instructions; does not fill polygon
no polygon previously defined	none	ignores instruction
1 or more parameters	2	fills the currently stored polygon

*Error 7 would have occured at the time the buffer overflowed.

Example — Creating a Surface Chart

The following BASIC program shows how to use the PM and FP instructions to draw and fill an area in a surface chart, leaving a blank area for the legend. (This program also uses labeling and tick instructions that are described in Chapters 5 and 6, respectively.)



```
10 REM  Insert configuration statement here
20 PRINT #1, "IN;SC-20,80,-10,60;"
30 PRINT #1, "SP2;PA0,0;"
40 PRINT #1, "PM0;PD0,10,10,16;"
50 PRINT #1, "PD20,20,30,14,40,18,50,16;"
60 PRINT #1, "PD60,22,60,0,0,0;PM1;"
70 PRINT #1, "PU4,4;PD4,8,16,8,16,4,4,4;PU;"
80 PRINT #1, "PM2;FP;EP;"
90 PRINT #1, "SP1;SI.25,.5;"
100 PRINT #1, "PU5.5,5;LBLEGEND"+CHR$(3)
110 PRINT #1, "PA0,0;ER60,40;"
120 PRINT #1, "TL1.5,0;"
130 FOR X=0 TO 60 STEP 10
140   PRINT #1, "PA";X;",0;XT;"
150 NEXT X
160 PRINT #1, "TL1,0;"
170 FOR Y=0 TO 40 STEP 5
180 PRINT #1,"PA0,";Y;";YT;"
190 NEXT Y
200 PRINT #1, "SP0;"
210 END
```

Program Explanation

10 establishes HP-IB or RS-232C interface conditions

20 initializes the plotter; scales the plotting area into user units

30 selects pen 2; sets the initial pen position

40 clears the polygon buffer and enters polygon mode; lowers the pen and defines two points of the outside of the surface area (beginning from the user-unit origin)

50 defines the next four points

60 defines the last three points (including returning to the first vertex at 0,0); closes this subpolygon

70 moves with the pen up to the starting position of the "legend" area (this is also the first point of the subpolygon); lowers the pen and defines four points; lifts the pen

80 exits polygon mode; fills the surface area using default fill conditions; edges the surface and legend areas

90 selects pen 1; defines the character size

100 moves with the pen up to the starting position for the label; draws the label ("Legend") — refer to the *Label Instruction, LB,* in Chapter 6 for an explanation of the label terminator CHR$(3)

110 moves to the user-unit origin; defines the chart area

120 defines the size of the X-axis tick marks

130 BASIC statement that starts the loop for drawing the X-axis ticks at intervals of 10 user units

140 moves to the tick location; draws the tick

150 BASIC statement that ends the loop

160 defines the size of the Y-axis tick marks

170 BASIC statement that starts the loop for drawing the Y-axis ticks at intervals of 5 user units

180 moves to the tick location; draws the tick

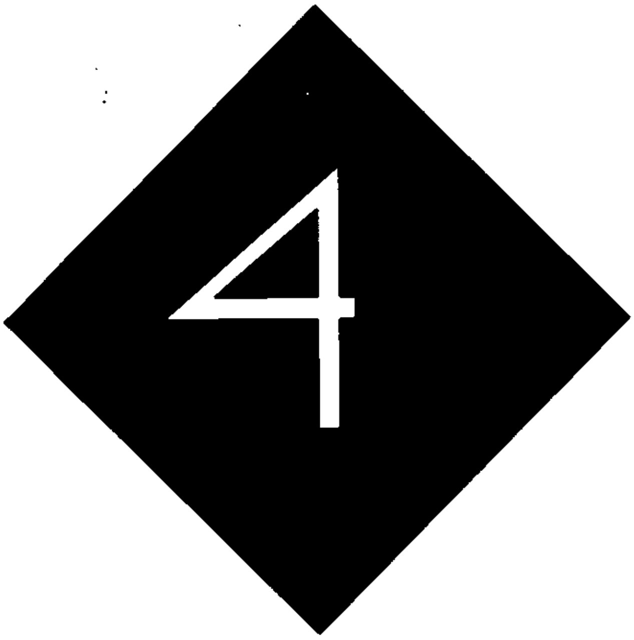
190 BASIC statement that ends the loop

200 returns the pen to the carousel

Example — Filling Alternate Subpolygons

The following BASIC program shows the effects of defining and filling several subpolygons. In addition, it shows how to use the **ESC**.T instruction to increase the size of the I/O buffer and decrease the size of the polygon buffer. This is done because the default size of the polygon buffer is larger than is necessary to create the plot. For further details about the polygon buffer, refer to *The Polygon Buffer* at the end of this chapter, and the *Allocate Configurable Memory Size Instruction, ESC. T,* in Chapter 12.)

NOTE: If you are using the RS-232-C interface, you can use the program example as is. If you are using the HP-IB interface, you cannot use the device-control instructions in lines 20 through 40. Delete those lines. HP Touchscreen (150) computers will require use of a separate file for output and input, only one of which may be open at a given time. Refer to Appendix A for an example. ■



```
10 REM Insert configuration statement here
20 PRINT #1, CHR$(27)+".T1274;700:"
30 PRINT #1, CHR$(27)+".L"
40 INPUT #1,L
50 PRINT #1, "IN;SP1;"
60 PRINT #1, "PA2000,1500;PM0;"
70 PRINT #1, "PA0,3500,2000,5500,4000,3500;"
80 PRINT #1, "PA2000,1500;PM1;PU;"
90 PRINT #1, "PA2150,4500;PD;PA2150,3650;"
100 PRINT #1, "PA2250,3650,2250,3500;"
110 PRINT #1, "PA2150,3500,2150,2800;"
120 PRINT #1, "PA2000,2800,2000,3500;"
130 PRINT #1, "PA1000,3500,2150,4500;"
140 PRINT #1, "PM1;PU;"
150 PRINT #1, "PA1350,3650;PD;PA2000,4200;"
160 PRINT #1, "PA2000,3650,1350,3650;"
170 PRINT #1, "PM2;FP;EP;"
180 PRINT #1, "SP0;"
190 END
```

Program Explanation

10 establishes HP-IB or RS-232-C interface conditions

20 allocates 1274 bytes to the I/O buffer and 700 bytes to the polygon buffer

30 causes the program to pause until all of the bytes are allocated in the buffer by requesting that the plotter output the I/O buffer size when it is empty

40 reads the output response generated by line 30 (the actual response is not important, but it should be read by the computer to avoid an error). Refer to Chapter 12 for hints on using the ESC.L instruction with the ESC.T instruction and for methods of reading the output response.

50 initializes the plotter; selects pen 1

60 sets the initial pen position; enters polygon mode

70 defines the second, third, and fourth points of the first subpolygon (the diamond shape)

80	completes the definition of the first subpolygon; exits the subpolygon; lifts the pen
90	sets the initial pen position for the second subpolygon (the outline of the "4"); lowers the pen; defines the second point of the subpolygon
100–130	defines the second subpolygon
140	exits the second subpolygon; lifts the pen
150	sets the starting point of the third subpolygon (the triangle in the "4"); lowers the pen; defines the second point of the subpolygon
160	defines the third subpolygon
170	exits polygon mode; fills alternate subpolygons; edges the subpolygons
180	returns the pen to the carousel

The Polygon Buffer

The polygon buffer is used whenever polygons are created with the PM, RA, RR, EA, ER, WG, or EW instructions. The points defining the polygon are stored in the polygon buffer. When you want to fill or edge a polygon, access the points from this buffer using the FP and EP instructions.

The default size of the polygon buffer is sufficient for most plots. If you are using an HP-IB (IEEE-488) interface, the plotter's entire buffer space of 2000 bytes is allocated to the polygon buffer when you use PM. This cannot be changed. If you are using an RS-232-C interface, 950 bytes are allocated to the polygon buffer and 1024 bytes are allocated to the I/O buffer. Read the rest of this chapter if you have an RS-232-C interface, and are interested in reallocating buffer space.

The size of the polygon buffer is specified by the second parameter of the ESC.T instruction. If your program does not use the PM, RA, RR, EA, ER, WG, or EW instructions, you can allocate the 950 bytes normally allocated to the polygon buffer to the I/O buffer. The larger the physical I/O buffer, the more data the computer can send to the plotter at a given time.

At the default setting of 950 bytes, the polygon buffer can store any polygon containing up to 86 points. To determine how many bytes the buffer needs for your plot, convert the number of points in the largest polygon into bytes, as described in the following section.

For specifics on using the ESC.T instruction to change the buffer sizes, refer to the *Allocate Configurable Memory Instruction, ESC.T*, in Chapter 12.

Determining the Approximate Size of a Polygon

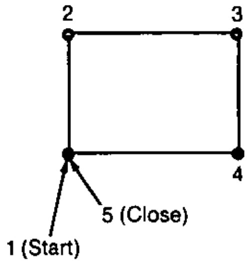
To determine how much space to allocate to the polygon buffer, you need to convert the number of points in your polygon into bytes. The following formula is approximate, but it will suit most situations. For a more accurate determination, read the paragraphs titled *Determining the Exact Size of a Polygon*, later in this section.

# of bytes = # of points × 11

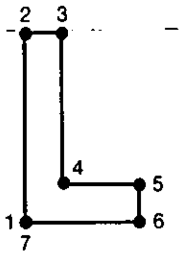
Counting the Points in Your Polygon

The starting pen position and each subsequent point define a polygon. As shown in the following example, although a rectangle has 4 corners, it is defined by 5 points.

```
"PU0,0;PM0;PD0,100,200,100,200,0,0,0;PM2;EP;"

         or

"PA0,0;EA200,100;"
```



The following shape is defined by 7 points.



```
"PU0,0;PM0;PD0,1000,200,1000,200,200;"
"PD600,200,600,0,0,0;PM2;EP;"
```

When an arc or circle defines a polygon, the number of points depends on the number of chords in the arc. Use the formula below to determine the number of points used to draw an arc or circle.

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

Thus, a full circle with the default chord angle of 5 degrees consists of 73 points ((360/5) + 1 = 73). On the other hand, a 45-degree arc with an angle of 30 degrees consists of 16 points ((45/3) + 1 = 16).

Determining the Exact Size of a Polygon

In most situations, the methods described above will provide the best approximation. If you need a more precise method so that you can allocate unused bytes to the I/O buffer, read this section.

To determine the total number of bytes required for any polygon, use the equation shown next. Each segment of the equation is described in the following paragraphs; an example is provided at the end of this section.

$$\text{total \# of bytes} = [(p_1 + f_1) + (p_2 + f_2) + \ldots]$$

where

p is 2 bytes needed by a PU, PD, PM0, or PM1 instruction,

and

f is defined by the following formula:

$$f = (9 \times \text{\# of points}) + \left(\frac{\text{\# of points}}{256}\right) \times 2$$

NOTE: A PU or PD uses 2 bytes *only* if the pen status changes *and* X,Y coordinates follow. The result of the division should be truncated. Thus, for 1 point, $f = (9 \times 1) + ([1/256] \times 2) = 9 + 0 = 9$ (remember, $[1/256] = 0$, **not** 0.078). Similarly, for 4 points, $f = (9 \times 4) + ([4/256] \times 2) = 36 + 0 = 36$. ■

The easiest way to understand this equation is to try an example. Consider the polygon created for the surface chart in the example given previously under the *Fill Polygon Instruction, FP.*

```
"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;"
```

The total number of bytes used by a polygon depends on:

PM0 uses 2 bytes	2
Current pen position (0,0) counts as 1 point, so solve *f* using 1 point:	9
PD uses 2 bytes:	2
2 points follow PD, so solve *f* using 2 points	18
4 points follow PD, so solve *f* using 4 points	36
3 points follow PD, so solve *f* using 3 points	27
PM1* uses 2 bytes:	2
1 point follows PU, so solve *f* using 1 point	9
PD uses 2 bytes:	2
4 points follow PD, so solve *f* using 4 points	36
Total number of bytes required for this polygon:	143

*In this example, the last point before PM1 closed the subpolygon, and was already 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 add this point, so you would need to add 9 (*f* for one point) after the PM1. This also holds true for PM2 if you do not include a point that closes the polygon.

Contrast this number with the previous general formula of 11 × # of points. This polygon has 15 points, so according to that formula, the polygon would require 165 bytes. By using the more precise equation, you would save 22 bytes, which can be allocated to the I/O buffer.

CHAPTER 8

CHAPTER 8

Putting the Instructions to Work

What You'll Learn in This Chapter

In this chapter you'll learn how to use instructions together to develop a plot. The following examples are designed to show you how to integrate many instructions into a complete program, how data might be handled, and how subroutines are used to program a task that is common to many plots and could be used in several programs.

Remember that these programs are written in Microsoft® BASIC. They use techniques such as FOR . . . NEXT loops and subroutines to read data and draw plots. If necessary, check your computer documentation for the correct methods of implementing these techniques.

The first program draws a line chart, one of the most common types of plots. You can use line charts to plot almost any kind of data — sales data, factory output, sales volume, data from laboratory experiments, population trends, etc. The concepts of plotting and labeling demonstrated here can be used in almost any application.

The second program draws a stacked bar chart; the third program draws a pie chart. Each requires a graphics enhancement cartridge. The sales data are differentiated in bars or wedges by solid fill, cross-hatching and parallel hatching. The programs demonstrate how to define fill types and how to fill and outline rectangles and wedges.

The first program is explained in detail, and is organized to show you how to develop a program. The second two programs are explained more briefly, because the concepts of developing these programs are similar to developing the line chart.

Line Chart

For this line chart, you will scale, draw, and label an X- and Y-axis and plot 1985 sales by region. The following paragraphs develop segments of the program in a logical sequence. The complete plot and program are shown later in the section titled *Program Listing.*

Setup and Scaling

For emphasis and readability, you should draw the data curves and title with wide pens. Narrow pens are usually sufficient for axes and labels. For this line chart, the suggested pen order for the carousel is:

1 = black, P. 3	5 = green, P. 7
2 = black, P. 7	6 = turquoise, P. 7
3 = red, P. 7	7 = unused
4 = blue, P. 7	8 = unused

If you do not have any wide pens (P.7), use narrow pens (P.3). You may purchase wide pens from Hewlett-Packard; part numbers are listed in the Operating Manual under *Accessories Available.*

Begin your program with the appropriate configuration statement for your computer. Then, using the IN or DF instruction, set the plotter to known conditions and cancel any parameters that may have been set in a previous program. IN is used here to be sure *all* conditions (such as P1/P2 settings) are set to a default state.

Select a pen (*SP1;*) and establish scaling points for this plot. The parameters of the IP instruction determine the location of the scaling points, P1 and P2. The location of these points provides a convenient area for the scale, which is assigned in the scaling statement *SC1,12,0,150;*. Since this chart shows one year's sales by month, the X-axis (commonly representing time) is scaled from 1 to 12. The Y-axis is scaled in thousands from 0 to 150 so that all sales data will fall inside this range. Labels and titles will be placed outside this area.

You will either need to know the range of your data or be willing to try some plots with different scales to determine what your scale statement should be. Thousands or millions of dollars are common scales.

Once the scale is established, draw a frame for the data area. Here *PU1,0;* moves the pen to the first point with the pen up. Then pen is then lowered and connects the four corners.

The first three program lines to accomplish the above are:

```
20    PRINT #1,   "IN;SP1;IP1250,750,9250,6250;"
30    PRINT #1,   "SC1,12,0,150;"
40    PRINT #1,   "PU1,0;PD12,0,12,150,1,150,1,0;PU;"
```

NOTE: If compatibility with some older HP plotters (such as the HP 9872 or HP 7220) is desired, use PA to begin plotting. To raise and lower the pen, use separate PU and PD instructions. In addition, the stricter syntax of these plotters would be required. ■

The Axes and Their Labels

You are now ready to draw and label the axes. The absolute label size instruction, *SI.2,.3;*, creates characters slightly larger than the default character size. The tick length is established by the instruction *TL1.5,0;*. The resulting ticks will be 1.5% of the horizontal or vertical distances between scaling points.

This program uses a FOR... NEXT loop to draw the axes. For the X-axis, let X range from 1 to 12 to represent 12 months of data. The loop does four things: moves to the integer location on the X-axis, draws a tick mark, establishes the label origin, and draws the label. Note that the X-parameter of the plotting instruction is a variable. If you do not know how to send a variable to the plotter, consult your computer's documentation and *Plotting with Variables* in Chapter 4. Since the XT instruction draws a tick whether the current pen status is up or down, be sure the pen is up to avoid unwanted lines between the ticks, labels, and axis.

Place the labels in a DATA statement in order to use the looping technique for labeling axes. (At some point, you might want to access data for the latest 12 months. If your data were stored with a data code, you could use a similar technique to read the labels and data from a file and properly label your chart for the data your were then plotting.) Then access the labels with a string variable in the LB instruction. Refer to *Labeling with Variables* in Chapter 6 for hints on sending variables in labels.

To position the labels, the program uses the CP instruction to center the label under the tick. By moving one-third character space back and one line down, the single character label is centered under the tick with enough space to be easily read. Finally, the axis title, Calendar Month, is centered and drawn under the axis.

The following lines contain the statements that perform the functions just described.

```
50    PRINT #1,  "SI.2,.3;TL1.5,0;"
60    FOR X = 1 TO 12
70      PRINT #1, "PA";X;",0;XT;"
80      READ A$
90      PRINT #1, "CP-.33,-1;LB"+A$+CHR$(3)
100   NEXT X
110   PRINT #1, "PA6.5,0;CP -7,-2.5;"
120   PRINT #1, "LBCalendar Month"+CHR$(3)

500   DATA "J","F","M","A","M","J"
510   DATA "J","A","S","O","N","D"
```

The Y-axis is created in a similar manner, except that the program uses the loop's index for the label value and two different CP instructions for labels of three digits and labels of less than three digits.

The lines which draw the Y-axis and label it follow.

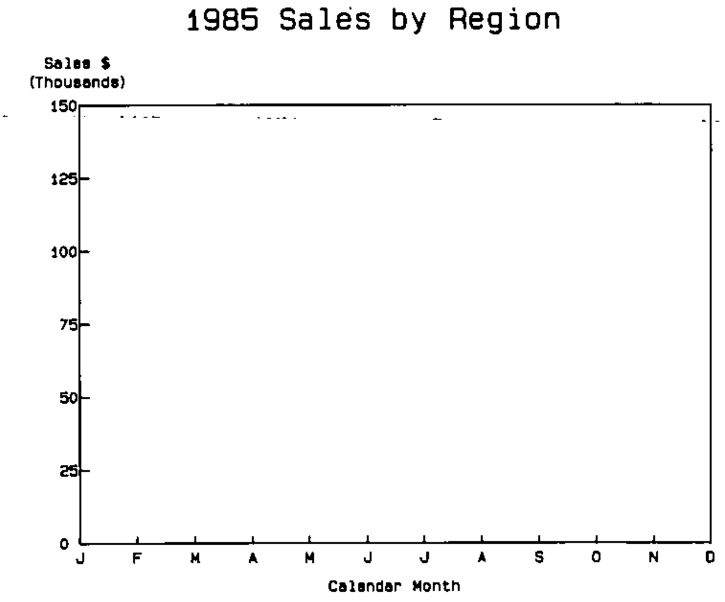
```
130  FOR Y = 0 TO 150 STEP 25
140    PRINT #1,  "PA1,";Y;";YT;"
150    IF Y<100 THEN PRINT #1, "CP-3,-.25;"
160    IF Y>99 THEN PRINT #1, "CP-4,-.25;"
170    PRINT #1, "LB";Y;CHR$(3)
180  NEXT Y
190  PRINT #1, "PA1,150;CP-3.5,2;"
200  PRINT #1, "LBSales $"+CHR$(3)+"CP-9,-1;"
210  PRINT #1, "LB(Thousands)"+CHR$(3)
```

Because the most important parts of the chart are the data and title, change to a wide pen. Next, move to the top center of the chart, increase the character size, and label the chart title.

The program lines that title the chart are:

```
220  PRINT #1, "SP2;PA6,150;SI.4,.6;CP-9.5,2;"
230  PRINT #1, "LB1985 Sales by Region"+CHR$(3)
```

Here's what the chart looks like so far.



Plotting the Data

You are now ready to draw lines. The first data line is drawn with parameters included when the program was written. Therefore, if the data changes, it will be necessary to change the plot instructions in the program.

The first line is drawn with pen 6 using the default solid line type. After drawing the line, the pen moves (up) to an area appropriate for labeling. After the character size is changed to match that used to label the axes, the plotter labels "South America." Actually, each of the data line's labels were inserted near the end of the creation process and involved trial and error to achieve satisfactory placement. Each label is drawn after each line of data is plotted.

The program lines which plot the lowest line and the corresponding label are:

```
240  PRINT #1, "SP6;LT;PA1,23;PD2,25,3,18,4,22;"
250  PRINT #1, "PD5,23,6,27,7,27,8,25,9,24,10,28;"
260  PRINT #1, "PD11,27,12,27;PU3.6,16;"
270  PRINT #1, "SI.2,.3;LBSouth America"+CHR$(3)
```

The program plots the three remaining lines from data read at execution time using nested FOR... NEXT loops and a READ statement. You can use this technique to plot a chart that will be replotted often with new data. If the necessary file statements were added, the data could be on a tape or disk file instead of in a DATA statement as shown here.

The first FOR... NEXT loop beginning in line 280 runs 3 times, once for each of the remaining data lines. With each loop sequence, a new pen color and line type (3, 4, 5) are selected in line 290.

In line 300 the second FOR... NEXT loop begins. It runs 12 times to read each of the 12 values in the DATA statement and draw to each point. As with the first data line, the corresponding label is drawn (lines 340-360) after each line is plotted.

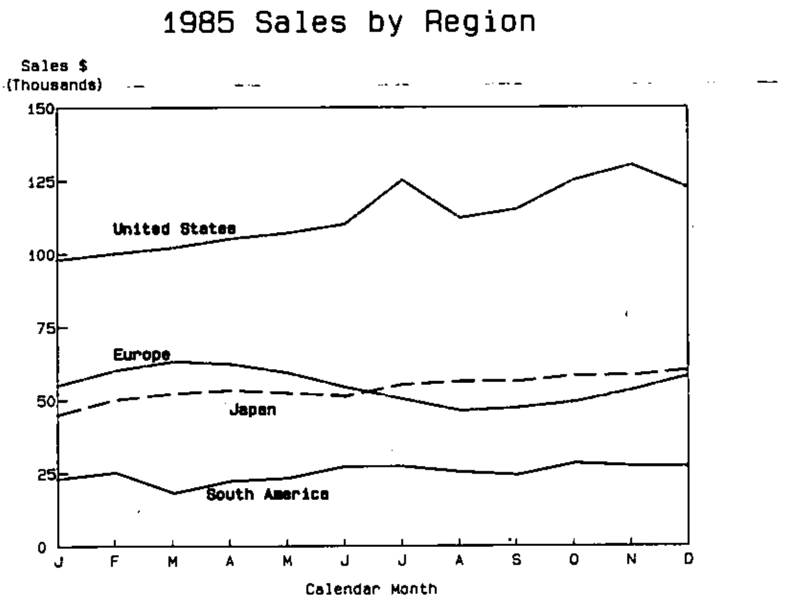
NOTE: Since this program uses variables as plot parameters, be sure they are sent to the plotter with a valid separator between them. Here, a comma has been inserted between variables to *ensure* that they are separated, even though many systems do not require this. Computers often send a leading and/or trailing blank space, or allow for a sign space before numeric variables. The plotter will treat a blank, comma, or a plus or minus sign as a separator between numeric parameters. Know your computer before sending variables with plot instructions. ■

The loops that draw the remaining three chart lines and the corresponding data statements follow. Although 340-360 are each printed on two lines to fit on this page, send them to the plotter as one continuous string.

```
280  FOR I = 1 TO 3
290     PRINT #1, "SP";I+2;";LT";I+2;";"
300     FOR X = 1 TO 12
310       READ  Y
320       PRINT#1, "PA";X;",";Y;"PD;"
330     NEXT X
340     IF I=1 THEN PRINT #1, "PU4,45;LBJapan"
                              +CHR$(3)
350     IF I=2 THEN PRINT #1, "PU2,64;LBEurope"
                              +CHR$(3)
360     IF I=3 THEN PRINT #1, "PU2,107;LBUnited
                              States"+CHR$(3)
370  NEXT I
380  PRINT #1, "SP0;"

520  DATA 45,50,52,53,52,51,55,56,56,58,58,60
530  DATA 55,60,63,62,59,54,50,46,47,49,53,58
540  DATA 98,100,102,105,107
550  DATA 110,125,112,115
560  DATA 125,130,122,0,0
570  END
```

Program Listing

A reduced version of the plot is shown next, followed by a complete listing of the program. This listing contains all of the BASIC statements necessary to have this program run on an HP Touchscreen (150) computer. Line 10 must include the proper configuration instructions necessary to establish interface conditions. You might need to make changes for your computer's BASIC. Or, you can use another programming language and send the HP-GL instructions using that language's output and looping techniques.



```
10   REM  Insert configuration statement here
20   PRINT #1,  "IN;SP1;IP1250,750,9250,6250;"
30   PRINT #1,  "SC1,12,0,150;"
40   PRINT #1,  "PU1,0;PD12,0,12,150,1,150,1,0;PU;"
50   PRINT #1,  "SI.2,.3;TL1.5,0;"
60   FOR X =.1 TO 12
70     PRINT #1, "PA";X;",0;XT;"
80     READ A$
90     PRINT #1, "CP-.33,-1;LB"+A$+CHR$(3)
100  NEXT X
110  PRINT #1, "PA6.5,0;CP -7,-2.5;"
120  PRINT #1, "LBCalendar Month"+CHR$(3)
130  FOR Y = 0 TO 150 STEP 25
140    PRINT #1,  "PA1,";Y;";YT;"
150    IF Y<100 THEN PRINT #1, "CP-3,-.25;"
160    IF Y>99 THEN PRINT #1, "CP-4,-.25;"
170    PRINT #1, "LB";Y;CHR$(3)
180  NEXT Y
190  PRINT #1, "PA1,150;CP-3.5,2;"
200  PRINT #1, "LBSales $"+CHR$(3)+"CP-9,-1;"
210  PRINT #1, "LB(Thousands)"+CHR$(3)
```

```
220  PRINT #1, "SP2;PA6,150;SI.4,.6;CP-9.5,2;"
230  PRINT #1, "LB1985 Sales by Region"+CHR$(3)
240  PRINT #1, "SP6;LT;PA1,23;PD2,25,3,18,4,22;"
250  PRINT #1, "PD5,23,6,27,7,27,8,25,9,24,10,28;"
260  PRINT #1, "PD11,27,12,27;PU3.6,16;"
270  PRINT #1, "SI.2,.3;LBSouth America"+CHR$(3)
280  FOR I = 1 TO 3
290    PRINT #1, "SP";I+2;";LT";I+2;";"
300    FOR X = 1 TO 12
310      READ  Y
320      PRINT#1, "PA";X;",";Y;"PD;"
330    NEXT X
340    IF I=1 THEN PRINT #1, "PU4,45;LBJapan"
                               +CHR$(3)
350    IF I=2 THEN PRINT #1, "PU2,64;LBEurope"
                               +CHR$(3)
360    IF I=3 THEN PRINT #1, "PU2,107;LBUnited
                               States"+CHR$(3)
370  NEXT I
380  PRINT #1, "SP0;"
500  DATA "J","F","M","A","M","J"
510  DATA "J","A","S","O","N","D"
520  DATA 45,50,52,53,52,51,55,56,56,58,58,60
530  DATA 55,60,63,62,59,54,50,46,47,49,53,58
540  DATA 98,100,102,105,107
550  DATA 110,125,112,115
560  DATA 125,130,122,0,0
570  END
```

Bar Charts and Pie Charts

Two kinds of area fill are commonly used in bar and pie charts; solid fill and hatching. Solid fill totally covers the area with color, whereas hatching fills the area with evenly spaced parallel lines. If there are two sets of parallel lines at 90-degree angles to each other, the fill is called "cross-hatching." To avoid optical illusions that affect one's perception of the size of a bar or pie segment, place hatching or cross-hatching at a 45-degree angle from vertical.

NOTE: For both the bar and pie charts, use the same pen order in the carousel that was recommended for the line chart. ■

Producing a Bar Chart

Overview

This program plots a stacked bar chart, titled "Sales Volume by Region." A stacked bar chart is appropriate when you want to compare parts of an element over time. In this case, the sales volumes of three different regions are compared over a period of three years. For ease of understanding, stacked bar charts should not contain more than six sets of stacks; limit each stack to three or four segments.

The following paragraphs describe the program according to groups of line numbers. The line numbers refer to the complete program listing, which is presented after all of the descriptions.

Setting Up the Data (Lines 10–170)

After dimensioning arrays and declaring the label and bar segment data, a loop is used to read the data into the arrays. The arrays will be used later to plot the data. This method is particularly convenient when you anticipate having new data at a future date, because you will only need to change the first few lines of the program to plot a new chart.

Scaling the Axes (Lines 180–210)

The X-axis is scaled by years, from 1982.3 to 1986.3. The fractions were chosen to allow for space before and after each bar. The Y-axis is scaled from 0 to 500 to represent sales in thousands of dollars.

Plotting the Title (Lines 220–260)

The title and axes are drawn with a wide pen for emphasis. The title is drawn first, with characters that are a little more than twice the default size.

Labeling the Axes (Lines 270–400)

The bars on the X-axis are labeled without tick marks, using a narrow pen and characters that are slightly larger than default size. The labels on each axis are offset from the origin using the PA instruction (lines 310 and 350) and then positioned appropriately with the CP instruction.

Labeling the Bar Segments (Lines 410–510)

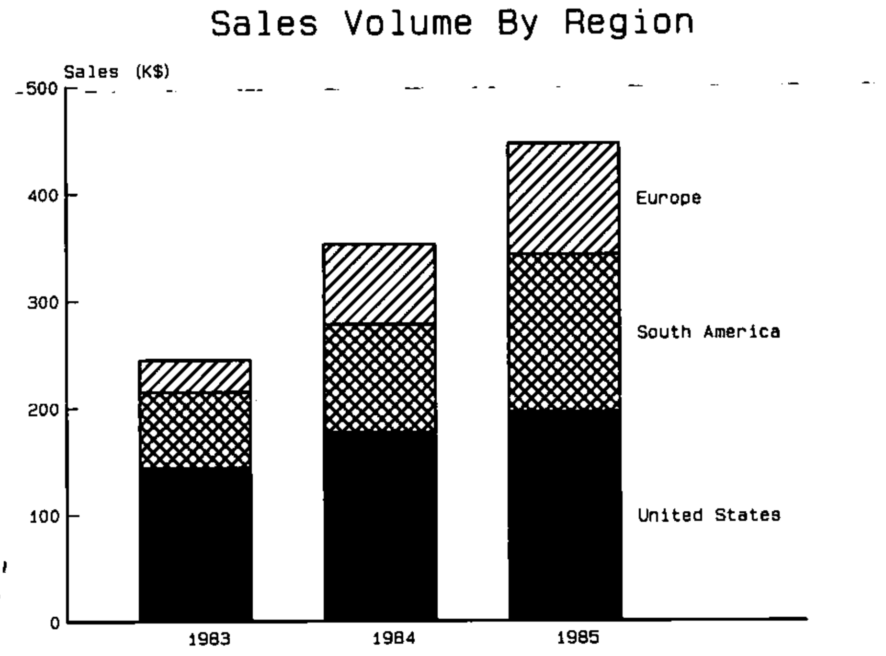
The data for labeling each bar segment was input previously using read and data statements (in lines 90–110). Using this data, each segment label is centered next to the right-most bar by computing a Y-axis position that is equal to the height of the prior segments plus one-half the height of the current segment.

Filling and Edging Each Segment (Lines 520–770)

The data for each bar segment was stored previously in a three-by-three array (in lines 120–160). Each array element contains the height of a segment with respect to the Y-axis scaling. This information is used to draw the bar segments from bottom to top using the FT, RA, and EA instructions to define, fill, and edge each stacked rectangular segment. Wide pens from stalls 3, 4, and 5 of the carousel are used for filling the bars.

Program Listing

The complete bar chart and listing follow. Note that the program includes comments denoted by REM. You do not have to enter these comments in your computer; they are included here to further explain the logic of the program. If you do enter them, check your computer documentation for the correct comment symbol for your computer.



```
10  REM Insert configuration statement here
20   REM  Generalized bar chart
30   REM
40   REM  Place chart and label data in arrays
50   REM  with lower bounds of 1.
60    OPTION BASE 1
70    DIM L$(3),B(3,3)
80   REM
90    DATA United States,South America, Europe
100   DATA 144,177,196,71,101,147,30,75,104
110   READ L$(1),L$(2),L$(3)
120   FOR I=1 TO 3
130     FOR J=1 TO 3
140       READ B(I,J)
150     NEXT J
160   NEXT I
170  REM
180  REM  Initialize plotter; set scaling points
190  REM  P1 and P2;  scale axes.
200   PRINT #1, "IN;IP1000,1000,9000,6750;"
```

(Program listing continues)

```
210   PRINT #1, "SC1982.3,1986.3,0,500;"
220  REM Label title using wide pen and large
230  REM letters; then draw axes.
240   PRINT #1, "SP2;PA1984.3,550;SI.4,.6;CP-10,0;"
250   PRINT #1, "LBSales Volume By Region"+CHR$(3)
260   PRINT #1, "PA1982.3,500;PD1982.3,0,1986.3,0;"
270  REM  Select narrow pen; reset character size.
280  REM  Set tick length; then label axis.
290   PRINT #1,  "PU;SP1;SI.2,.3;TL1.5,0;"
300   FOR X=1983 TO 1985
310     PRINT #1, "PA";X;",0;CP-1.83,-1;LB";X;CHR$(3)
320   NEXT X
330  REM  Add ticks and then labels to the Y-axis.
340   FOR Y=0  TO 500 STEP 100
350     PRINT #1,  "PA 1982.3,";Y;";YT;"
360     IF Y=0 THEN PRINT #1, "CP-2.5,-.25;"
370     IF Y<>0 THEN PRINT #1, "CP-4.5,-.25;"
380     PRINT #1, "LB";Y;CHR$(3)
390   NEXT Y
400   PRINT #1, "PA 1982.3,510;LBSales (K$)"+CHR$(3)
410  REM  Center segment labels using prior height
420  REM  plus one-half current height.
430   FOR I=1 TO 3
440     Y=0
450       FOR J=1 TO I-1
460         Y=Y+B(J,3)
470       NEXT J
480     Y=Y+B(I,3)/2
490     PRINT #1,  "PA1985.4,";Y;";"
500     PRINT #1,  "CP0,-.25;LB"+L$(I)+CHR$(3)
510   NEXT I
520  REM  Draw and fill each bar using wide pens.
530   FOR I = 1 TO 3
540     PRINT #1, "SP";I+2;";PT.7;"
550     K=1
560     FOR X=1983 TO 1985
570       Y1=0
580  REM  Compute Y-axis start point
590  REM  for each bar segment.
600       FOR J=1 TO I-1
610         Y1=Y1+B(J,K)
620       NEXT J
```

(Program listing continues)

```
630  REM  Compute Y-axis end point
640  REM  for each bar segment.
650       Y2=Y1+B(J,K)
660       K=K+1
670  REM  Select fill type
680       IF I=1 THEN PRINT #1, "FT1;"
690       IF I=2 THEN PRINT #1, "FT4,.05,45;"
700       IF I=3 THEN PRINT #1, "FT3,.05,45;"
710  REM  Move to start point for segment;
720  REM  then fill and outline defined bar area.
730       PRINT #1, "PA";X-.3;",";Y1;";"
740       PRINT #1, "RA";X+.3;",";Y2;";"
750       PRINT #1, "EA";X+.3;",";Y2;";"
760     NEXT X
770   NEXT I
780  REM  Put pen away and end program.
790   PRINT #1, "SP0;"
800   END
```

Producing a Pie Chart

Overview

This program plots a pie chart, titled "Sales Dollar Distribution." Pie charts easily convey information concerning parts of a whole. In this case, sales dollar distribution is broken into four groups: R&D, Administration, Marketing, and Manufacturing. For ease of understanding, pie charts should not be broken into less than three nor more than six parts.

The following paragraphs describe the program according to groups of line numbers. The line numbers refer to the complete program listing, which is presented after all of the descriptions.

Setting Up the Data (Lines 10–180)

The start angle, mid-point, and stop angle for each segment are placed into arrays using read and data statements, the same method as was used in the bar chart. In addition, the pie segment labels are read into arrays.

Scaling the Chart (Lines 190–220)

To center the pie chart, P1 and P2 are repositioned with the IP instruction. The plotting area is scaled so that 0,0 is near the center of the page.

Plotting the Title (Lines 230-250)

The title is centered with the CP instruction, the character size is set to 0.4-cm wide and 0.6-cm high with the SI instruction, and a wide pen is selected with the SP instruction.

Plotting the Labels (Lines 260-450)

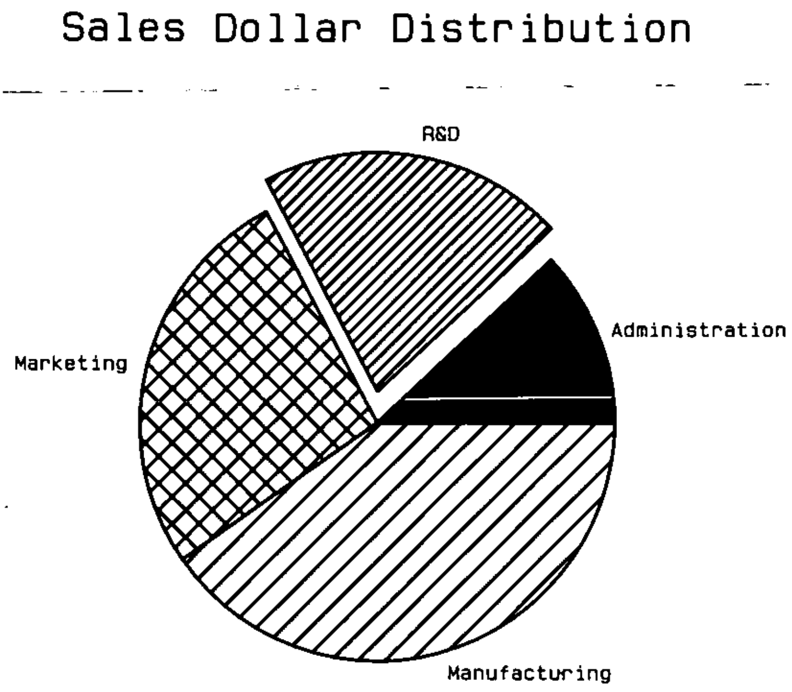
Each segment is labeled using the data previously read in line 180. For readability, the labels are drawn with a narrow pen and with a character size that is smaller than the one used for the title. The labels are centered and offset from the appropriate segment with the CP instruction. The distance used to position the label for segment 2 is increased, since the second segment is offset.

Filling and Edging Each Segment (Lines 460-640)

In this loop, a wide pen and a fill type are selected for each segment with the SP and FT instructions. Lines 490 and 500 adjust the center point for the offset segment. The sweep angle for each segment is then computed (sweep angle = stop angle − start angle), and the segments are filled and outlined with the WG and EP instructions.

Program Listing

The complete pie chart and listing follow. As with the bar chart, comment lines are indicated by REM. You can delete the lines if you wish.



```
10    REM  Insert configuration statement here
20    REM  Pie chart
30    REM  Place chart and label data in arrays
40    REM  with lower bounds of 1.
50     OPTION BASE 1
60     DIM W$(4),P(4,3)
70     DATA 0,21.5,43,43,80.5,118,118,166.5
80     DATA 215,215,287,360
90     DATA Administration,R&D,Marketing
100    DATA Manufacturing
110   REM  Read start, mid-point, and stop angle
120   REM  for each segment.  Then read labels.
130    FOR I=1 TO 4
140      FOR J=1 TO 3
150        READ P(I,J)
160      NEXT J
170    NEXT I
180    READ W$(1), W$(2), W$(3),W$(4)
```

(Program listing continues)

```
190  REM  Initialize plotter; set scaling points
200  REM  P1 and P2; scale the chart.
210   PRINT #1, "IN;IP2250,500,8250,7100;"
220   PRINT #1, "SC-10,10,-10,12;"
230  REM  Label title using wide pen, large letters.
240   PRINT #1, "SP2;PA0,12;SI.4,.6;CP-12.34,0;"
250   PRINT #1, "LBSales Dollar Distribution"+CHR$(3)
260  REM  Label each wedge using narrow pen and
270  REM  small letters.
280   PRINT #1, "SP1;SI.2,.3;"
290  REM  Set PI variable to convert from
300  REM  degrees to radians.
310   PI=3.141593
320   FOR I = 1 TO 4
330     REM  Move to center arc.
340     R=8
350     IF I=2 THEN R=9
360     X=R*COS(P(I,2)*(PI/180))
370     Y=R*SIN(P(I,2)*(PI/180))
380     PRINT #1, "PA";X;",";Y;";"
390     REM  Determine label origin; draw label.
400     L=LEN  (W$(I))
410     IF I=1 THEN PRINT #1, "CP0,-.25;"
420     IF I=3 THEN PRINT #1, "CP";-L;",-.25;"
430     IF I=4 THEN PRINT #1, "CP0,-.5;"
440     PRINT #1, "LB"+W$(I)+CHR$(3)
450   NEXT I
460  REM Draw and fill the wedges using pens 3-6.
470   FOR I=1 TO 4
480     PRINT #1, "SP";I+2;";PT.7;"
490     X=0
500     Y=0
510     IF I=2 THEN X=COS(P(I,2)*(PI/180))
520     IF I=2 THEN Y=SIN(P(I,2)*(PI/180))
530     IF I=1 THEN PRINT #1, "FT1;"
540     IF I=2 THEN PRINT #1, "FT3,.3,45;"
550     IF I=3 THEN PRINT #1, "FT4,.6,45;"
560     IF I=4 THEN PRINT #1, "FT3,.6,45;"
570     REM  Compute the sweep angle.
580     S=P(I,3)-P(I,1)
```

(Program listing continues)

```
590      REM  Fill the wedge.
600      PRINT #1, "PA";X;",";Y;";"
610      PRINT #1, "WG7.5,";P(I,1);",";S;";"
620      REM  Outline the wedge.
630      PRINT #1, "PA";X;",";Y;";EP;"
640    NEXT I
650  REM Put pen away and end program.
660    PRINT #1, "SP0;"
670    END
```

CHAPTER 9

CHAPTER 9

Changing the Plotting Area

What You'll Learn in This Chapter

In this chapter you will learn more about the graphics limits and how to alter the plotting area to suit special applications you might have. Specifically, you will learn how to:

- define a window that restricts plotting to a certain area
- output certain plotter coordinates to the computer
- rotate the axes
- proportionally reduce/enlarge the plotting area
- create mirror images

HP-GL Instructions Covered

IW	Input Window Instruction
OW	Output Window Instruction
OH	Output Hard-Clip Limits Instruction
OP	Output P1 and P2 Instruction
RO	Rotate Coordinate System Instruction

Terms You Should Understand

If you are unfamiliar with any of these terms, refer to the glossary at the end of the manual.

Clipping
Hard-Clip Limits
Soft-Clip Limits
Window

Relationship of Plotting Instructions and Graphics Limits

For most plotting situations, the pen will probably draw as you expect it to. However, proper movement depends on what your graphics limits are and what X,Y coordinates you specify. This section describes the situations that affect the movement of the pen. It applies directly to these plotting instructions: PA, PR, CP, LB, PB, UC, XT and YT.

This section also applies to the following instructions on the cartridge: EA, ER, CI, AA, AR, RA, RR, WG, EW, EP, and FP.

Types of X,Y Coordinates

The pen motion caused by plotting instructions depends on the type of X,Y coordinates specified. There are three types of X,Y coordinates:

1. Inside window area

2. Outside window area and in-range

3. Out-of-range

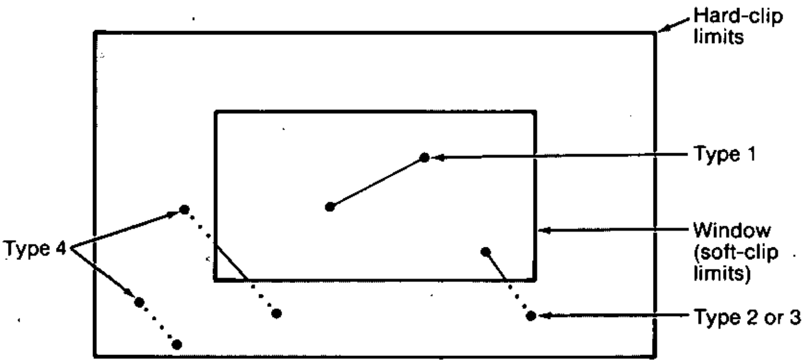
Out-of-range coordinates are defined as either X or Y, or both, being less than −32 768.0000 or greater than +32 767.9999. Specifying out-of-range coordinates sets error 3 (bad parameter) and results in the plotting instruction being ignored. (In a PA, PR, PU, or PD instruction, where you can specify multiple coordinate pairs, only the pair with the out-of-range coordinate and all subsequent pairs are ignored; the previous pairs are still executed.)

The other two types of coordinates (inside window and outside window) are discussed below. Remember that plotting occurs only within the currently defined window. At power-on, the currently defined window is the graphics (hard-clip) limits.

Plotting Inside and Outside of Windows

There are, in general, four types of line segments that can be specified from a given point to some new point, as follows:

	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



In type 1, the pen moves from the last point to the new point with the pen up or down as programmed.

In type 2, the pen moves from the last point toward the new point and stops where the line segment between these points intersects the window limit. The pen up/down state is as programmed until the intersection point is reached. Then, the pen lifts and remains in this position until a plotting instruction is received that brings the pen back inside the window (refer to type 3).

In type 3, the pen movement depends on whether the pen is programmed to be up or down. If the pen is programmed to be up, it moves directly to the new point. If the pen is programmed to be down, it moves, up, to the point where the line segment between the last point and the new point intersects the window limit. When the pen reaches this point, the pen assumes its programmed down position. The pen then continues plotting to the new point.

In type 4, no pen movement occurs unless some part of the line segment between the last and new points is within the window area. If part of the line is in the window area, the pen moves, up, to the first intersection point with the window. The pen then moves as programmed (up or down) to the second intersection point, where it lifts and stops moving. Thus, this situation is a combination of type 3, followed by type 1, followed by type 2. If none of the line segments lies within the window area, the X,Y coordinates of the current pen position are updated even though the pen does not move.

Input Window Instruction, IW

USES: The IW instruction defines a rectangular area, or window, that establishes soft-clip limits. Subsequent programmed pen motion will be restricted to this area. Use this instruction when you want to be sure that your plot falls within a specific area. For example, you might define a window for a graph on part of the page, and then define a new window on the rest of the page for labels. By establishing windows, you can be certain that stray lines or labels won't appear in the wrong areas.

SYNTAX: *IW* $X_{LL}, Y_{LL}, X_{UR}, Y_{UR}$ *term*
or
IW *term*

Parameter	Format	Range	Default
X,Y coordinates	integer	−32 768 to 32 767	A4-size 0,0,10 900,7650 A-size 0,0,10 300,7650

EXPLANATION: The four parameters of the IW instruction specify, *in plotter units*, the X,Y coordinates of the lower-left (LL) and upper-right (UR) corners of the window area.

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 *Relationship of Plotting Instructions and Graphics Limits* at the beginning of this chapter.

NOTE: Pen movement directed by the front-panel CURSOR CONTROL buttons is not restricted by a window. ■

An IW instruction without parameters (*IW;*) defaults the window to be the current hard-clip limits.

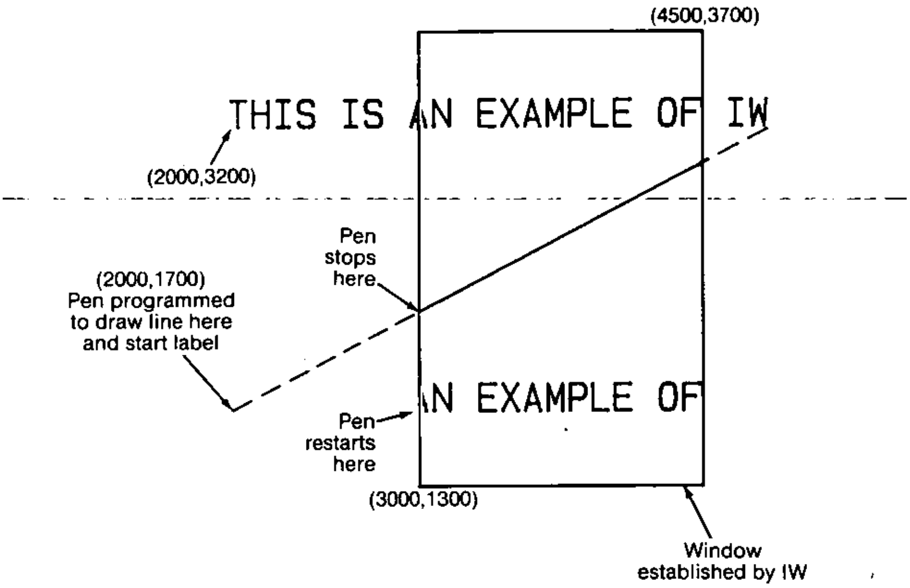
The IW instruction remains in effect until another IW instruction is executed, or the plotter is initialized or set to default conditions.

The following table summarizes the error conditions or unexpected results that might occur with the IW instruction.

Condition	Error	Plotter Response
no parameters	none	establishes hard-clip limits as default window
more than 4 parameters	2	uses first 4 parameters
1, 2, or 3 parameters	2	ignores instruction
parameter out-of-range	3	ignores instruction
hard-clip limits < parameter < 32 767	0	resets to corresponding hard-clip parameter
−32 768 < parameter < 0	0	resets parameter to 0

Example — Effects of Specifying a Window on Labels and Lines

The following example 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. (For an example of using windows to enlarge a portion of a plot, refer to *Techniques for Manipulating the Plotting Area* later in this chapter.)



```
10   REM  Insert configuration statement here
20   PRINT #1,   "IN;SP1;"
30   PRINT #1,   "SI.2,.35;PA2000,3200;"
40   PRINT #1,   "LBTHIS IS AN EXAMPLE OF IW"+CHR$(3)
50   PRINT #1,   "IW 3000,1300,4500,3700;"
60   PRINT #1,   "PD2000,1700;"
70   PRINT #1,   "LBTHIS IS AN EXAMPLE OF IW"+CHR$(3)
80   PRINT #1,   "PU3000,1300;PD4500,1300,4500,3700;"
90   PRINT #1,   "PD3000,3700,3000,1300;"
100  PRINT #1,   "SP0;"
110  END
```

Program Explanation

10 establishes HP-IB or RS-232-C interface conditions

20 initializes the plotter; selects pen 1

30 sets the character size; moves to the starting position for the first label

40 draws the label

50 specifies a window 1500 plotter units wide and 2400 plotter units high

60 draws a line from the end of the first label to the beginning of the next label (pen only physically draws within the window)

70 draws the second label (pen only draws within the window)

80-90 moves to the lower-left corner of the window; outlines the window

100 returns the pen to the carousel

Output Window Instruction, OW

USES: The OW instruction outputs the X,Y coordinates of the current window in which plotting can occur. Use this instruction when you need to know the size of the current window.

SYNTAX: *OW term*

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

EXPLANATION: After an OW instruction is received, the plotter outputs, in plotter units, four ASCII integers. These represent the X,Y coordinates of the lower-left (LL) and upper-right (UR) corners of the current window.

After you send the OW instruction, your program should read the plotter's output response. Refer to *Hints for Obtaining Plotter Output* in Chapter 10.

The following table summarizes the error conditions or unexpected results that might occur with the OW instruction.

Condition	Error	Plotter Response
no window previously specified	none	outputs current hard-clip limits
1 or more parameters	2	outputs current window coordinates

Output Hard-Clip Limits Instruction, OH

USES: The OH instruction outputs the X,Y coordinates of the current hard-clip limits. Use this instruction to determine the plotter-unit dimensions of the physical area in which plotting can occur. This instruction is particularly helpful to determine the new hard-clip limits when the axes have been rotated.

SYNTAX: *OH term*

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

EXPLANATION: After an OH instruction is received, the plotter outputs four integers in ASCII, which represent the X,Y coordinates of the lower-left (LL) and upper-right (UR) corners of the current hard-clip limits. The coordinates are always output in plotter units.

The hard-clip limits determine the maximum physical plotting area. They are initially established when the plotter reads the the setting of the paper-size switch on the rear panel. Although any subsequent IW instruction can further reduce the plotting area by establishing soft-clip limits, these limits *do not affect* the hard-clip limits output by the OH instruction. However, rotation of the axes with the RO instruction will reverse the X,Y coordinates output by OH.

The default hard-clip limits for each paper size are shown in the table on the next page.

Paper Size	Hard-Clip Limits			
	X_{LL}	Y_{LL}	X_{UR}	Y_{UR}
A4	0	0	10 900	7650
A	0	0	10 300	7650
A4 (rotated)	0	0	7650	10 900
A (rotated)	0	0	7650	10 300

After you send the OH instruction, your program should read the plotter's output response. Refer to *Hints for Obtaining Plotter Output* in Chapter 10.

The following table summarizes the error conditions or unexpected results that might occur with the OH instruction.

Condition	Error	Plotter Response
1 or more parameters	2	outputs current hard-clip limits

Output P1 and P2 Instruction, OP

USES: The OP instruction outputs the X,Y coordinates of the current scaling points P1 and P2. Use this instruction to compute the number of plotter units per user unit when scaling is on, to determine the numeric coordinates of P1 and P2 when they have been set from the front panel, or in conjunction with the IW instruction to set P1 and P2 within the window.

SYNTAX: *OP term*

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

EXPLANATION: After an OP instruction is received, the plotter outputs four integers in ASCII, which represent the X,Y coordinates of P1 and P2 in plotter units. The possible range is −32 768 to 32 767. Upon completion of output, bit position 1 of the status byte is cleared if set (refer to the *Output Status Instruction, OS*, in Chapter 10).

After you send the OP instruction, your program should read the plotter's output response. Refer to *Hints for Obtaining Plotter Output* in Chapter 10.

The following table summarizes the error conditions or unexpected results that might occur with the OP instruction.

Condition	Error	Plotter Response
1 or more parameters	2	outputs current P1 and P2 coordinates

Rotate Coordinate System Instruction, RO

USES: The RO instruction rotates the plotter's coordinate system (either plotter units or user units) 90 degrees about the plotter-unit coordinate origin. Use this instruction when you want to change the orientation of your plots. Normally, plots are oriented horizontally on the page; with the RO instruction, you can easily make a vertical plot.

SYNTAX: *RO* n *term*
or
RO *term*

Parameter	Format	Range	Default
n	integer	0 or 90 degrees	0 degrees

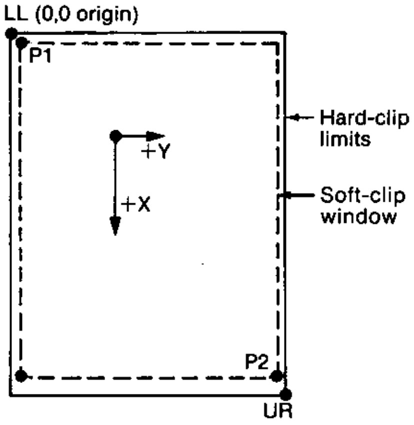
EXPLANATION: Only two parameters are allowed with the RO instruction, as follows.

n = 0 Cancel rotation; reestablish default orientation of coordinate system

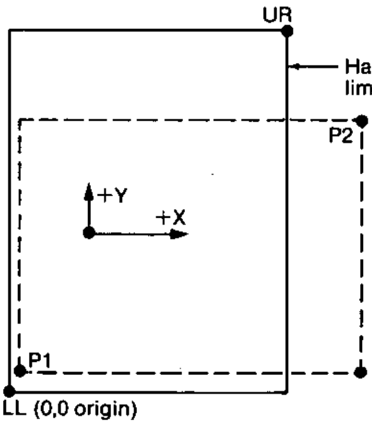
n = 90 Rotate coordinate system 90 degrees about the plotter-unit origin

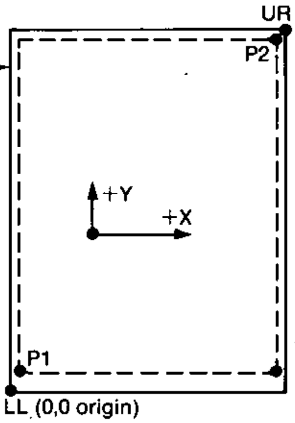
An RO instruction without parameters (*RO;*) is the same as *RO0;*. When you issue *RO90;*, the plotter's coordinate system will rotate counter-clockwise. P1 and P2 rotate with the coordinate system. However, they maintain the same X,Y coordinate values as before rotation. This means that P2 can be located outside of the hard-clip limits. Refer to the lower-left figure.

Follow the RO instruction with the IP and IW instructions, *RO 90; IP; IW;*, to reset P1, P2 and the soft-clip window all within the rotated hard-clip limits. Refer to the lower-right figure. IP



Default Orientation (RO; or RO 0;)





Rotated 90 Degrees (RO 90;)

Rotated 90 Degrees (RO 90; IP; IW; or **ENTER + VIEW***)* effectively switches the X,Y coordinates of each point (for example, if P1 was 250,279 before rotation, it will become 279,250 after rotation). Be sure to include the IW instruction in this sequence; otherwise; the soft-clip window will be rotated in the manner indicated in the lower-left figure on the previous page.

You can also turn rotation on and off from the front panel. Hold down **ENTER** and then press **VIEW** to turn on rotation; press again in the same sequence to turn it off. The rotation will be the same as if *RO 90; IP; IW;* were executed.

The physical position of the pen does not change when the coordinate system is rotated. Instead, the plotter updates the pen's internal graphics (X,Y coordinate location) position to reflect the new coordinate system. You can obtain the coordinates of the new pen position by executing an OA or OC instruction after the rotation.

The RO instruction remains in effect until the rotation is changed by another RO instruction or the plotter is initialized. Rotations are not cumulative; in other words, you cannot continue rotating beyond the first 90-degree rotation.

The following table summarizes the error conditions or unexpected results that might occur with the RO instruction.

Condition	Error	Plotter Response
no parameters	none	establishes default orientation
more than 1 parameter	2	uses first parameter
parameter out-of-range	3	ignores instruction
RO 90; when plot is already rotated 90 degrees	none	ignores instruction

Techniques for Manipulating the Plotting Area

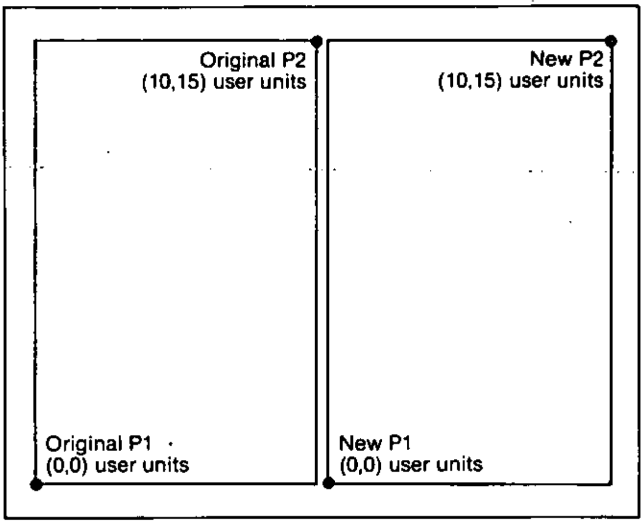
The IP and IW instructions provide a great deal of flexibility for controlling the size of the plotting area and the proportions of a plot. You've already seen this in Chapter 3, where you learned how to set P1,P2 and assign a scale so that isotropic scaling would be established and geometric shapes would be drawn correctly. In this section, you will learn more techniques, such as placing more than one equal-sized plot on a page, enlarging/reducing plots, and creating mirror images.

Preparing Equal-Sized Plots on One Page

The easiest way to prepare equal-sized plots on one page is to take advantage of the fact that P2 follows P1 whenever P1 is changed. The following program illustrates the procedure (you could also change P1 and P2 settings from the front panel in the same way).

First, this program specifies P1 and P2 on the left-hand side of the page. Then it assigns a scale of 0 to 10 along the X-axis and 0 to 15 along the Y-axis, and draws a rectangle to illustrate the boundaries of the P1/P2 frame. Of course, you could assign any scale and plot anything here. Then the IP instruction specifies a new P1 location; *P2 automatically tracks P1* to establish a new P1/P2 frame with the same dimensions as the first. You don't have to perform the arithmetic to specify a new P2. Finally, a rectangle is again drawn around the P1/P2 boundaries, using the same scaling as for the first P1/P2 frame.

```
10  REM  Insert configuration statement here
20  PRINT #1, "IN;IP 0,200,4950,7650;"
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, "IP5120,200;"
60  PRINT #1, "PA0,0;PD10,0,10,15,0,15,0,0;"
70  PRINT #1, "SP0;"
80  END
```



IP 0,200,4950,7650; *IP 5120,200;*

Using IP to Create Equal-Sized Plots

NOTE: These P1/P2 frames are not windows or graphics limits. When drawing the plots for each half of the page, the pen can actually move anywhere within the hard-clip limits. However, *since P1 and P2 moved, the scale assigned to them moved with them.* This allows you to use the same coordinates on both halves of the page. If you did not assign a scale to P1 and P2 with the instruction, you would have to calculate the new plotter-unit coordinates for the plot on the second half of the page. ■

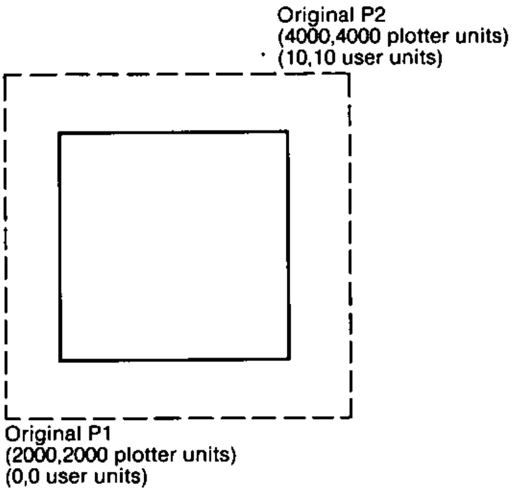
Reducing/Enlarging Plots

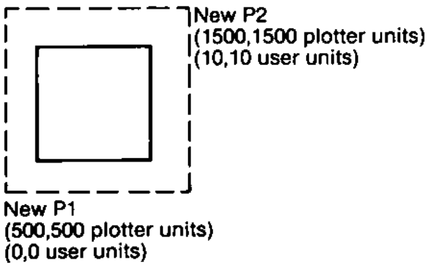
The basic technique for changing the size of a plot is to assign a scale to P1 and P2, and then to move P1 and P2 to a smaller or larger portion of the page. As long as scaling is active, the changes in P1 and P2 produce the effect of reducing or enlarging the plot. This is particularly useful when you want to be able to draw your plot on any portion of the page (as described above).

The following program is very simple in order to show the concept. First, the program specifies a square P1/P2 area, and assigns a scale of 0 to 10 along both axes. Then it draws a square centered in the P1/P2 frame. Next, it specifies a new, smaller

P1/P2 frame. Without changing the scaling, it redraws the same square, again centered. This time the square is much smaller. To enlarge the square, specify a new P1/P2 frame that is larger.

```
10    REM  Insert configuration statement here
20    PRINT #1, "IN;IP 2000,2000,4000,4000;"
30    PRINT #1, "SC0,10,0,10;"
40    PRINT #1, "SP1;PA2,2;PD8,2,8,8,2,8,2,2;PU;"
50    PRINT #1, "IP 500,500,1500,1500;"
60    PRINT #1, "PA2,2;PD8,2,8,8,2,8,2,2;"
70    PRINT #1, "SP0;"
80    END
```





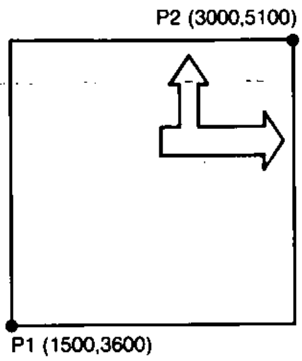
Creating Mirror Images

For most applications, you will set P1 and P2 so that P1 is in the lower-left corner and P2 is in the upper-right corner of the P1/P2 frame. However, you can change the relationships of P1 to P2. When you do, a phenomenon known as mirror images occurs. Mirror images can happen with all plotting and labeling instructions. Mirror images from labeling instructions can be very complex; refer to *Parameter Interaction in Labeling Instructions* in Chapter 6.

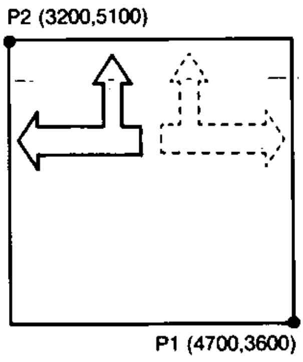
Mirror images from plotting instructions are easier to illustrate. The following program uses a subroutine to draw the same plot (arrows) four times. The main program sets P1 and P2, then goes to the subroutine to outline the P1/P2 frame and draw the arrows. It then sets P1 and P2 in different corners, and repeats the drawing. This process continues until all four possible combinations of P1 and P2 have been plotted. (The original "normal" plot is shown in each plot so that you can better see the orientation of the mirror image.)

NOTE: Scaling is the same in each case; P1 always has the value −15,−10 user units; P2 always has the value 15,10 user units. ■

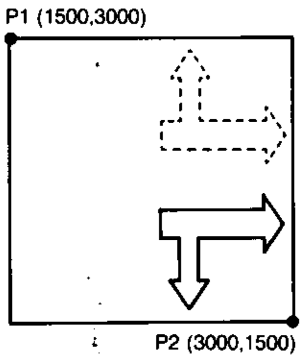
```
10   REM  Insert configuration statement here
20   PRINT #1, "IN;SP1;IP1500,3600,3000,5100;"
30   PRINT #1, "SC-15,15,-10,10;"
40   GOSUB 130
50   PRINT #1, "IP4700,3600,3200,5100;"
60   GOSUB 130
70   PRINT #1, "IP1500,3000,3000,1500;"
80   GOSUB 130
90   PRINT #1, "IP4700,3000,3200,1500;"
100  GOSUB 130
110  PRINT #1,  "SP0;"
120  END
130  PRINT #1, "PU-15,-10;"
140  PRINT #1, "PD15,-10,15,10,-15,10,-15,-10;PU;"
150  PRINT #1, "PA1,2;PD1,4,3,4,3,7,2,7,4,9,6,7,5,7;"
160  PRINT #1, "PD5,4,12,4,12,5,14,3,12,1;PU;"
170  PRINT #1, "PD12,2,1,2;PU;"
180  RETURN
```



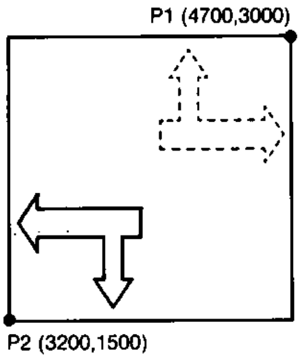
Normal
(Lines 20–40)



Reversed
(Lines 50–60)



Upside Down
(Lines 70–80)



Upside Down and Reversed
(Lines 90–100)

CHAPTER 10

CHAPTER 10

Obtaining Information from the Plotter

What You'll Learn in This Chapter

In this chapter you'll learn how to obtain information from the plotter about:

- HP-GL errors
- plotter model number
- plotter capabilities
- plotter status for debugging and digitizing
- HP-GL error status and request for service
- pen position and status
- the number of plotter units per millimetre

This chapter concludes with a summary of output response types.

To use output instructions, your computer must be able to read data in from the plotter. Most computer systems can do this; however, it is easier on some computers than on others. Appendix A provides program examples that use the output identification instruction, OI, on several personal computers. If your computer is listed, use the program as a guide when including output instructions in your programs. If your computer is not listed, and you are not sure how your computer system reads in data, check your system documentation. The program examples included in this chapter illustrate more simple uses of output instructions.

NOTE: HP Touchscreen (150) computers, when used in an HP-IB configuration, cannot read output from the plotter if programmed in a high-level language. ■

HP-GL Instructions Covered

IM	Input Mask Instruction
OE	Output Error Instruction
OI	Output Identification Instruction
OO	Output Options Instruction
OS	Output Status Instruction
OA	Output Actual Position and Status Instruction
OC	Output Commanded Position and Pen Status Instruction
OF	Output Factors Instruction

Other HP-GL Output Instructions

The following output instructions are described in other chapters of this manual, according to their function.

OD	Output Digitized Point — Chapter 11
OH	Output Hard-Clip Limits Instruction — Chapter 9
OP	Output P1 and P2 Instruction — Chapter 9
OW	Output Window Instruction — Chapter 9

Hints for Obtaining Plotter Output

When the plotter executes an output instruction, it makes the requested information available. This is called the output response. To use the output response in a program, your computer must read the response with an input statement (such as INPUT #, ENTER, READ, or READLN). **It is best to read the output response immediately.**

Use the following sequence to request information from the plotter:

1. Send the output instruction just as you would send any HP-GL instruction (for example, *PRINT #1, "OE;"*). Note that none of the output instructions use parameters.

2. Read the plotter's output response using an appropriate computer input statement for your language (*INPUT #2, A,* for example).

When you read the output response, specify the correct number and type of variables to hold the plotter's output response, as shown in the following table using a BASIC example.

If the output response is ...	**the variable *type* should be**
numeric (integer or real)	A
character string	A$

For instructions that cause a mutiple output response, be sure to read **all** of the output response, even if you need to know only part of the information. For example, the OO instruction outputs eight integers. In this case, use a statement such as:

INPUT #2, A, B, C, D, E, F, G, H

Check your computer documentation for any special configuration requirements.* Read the note for HP-IB or RS-232-C configurations, next in this section, for considerations that depend on the interface you are using.

Notes for HP-IB Configurations

The plotter must be in addressable mode in order to send the output response to the computer. If the plotter is in the listen-only mode, the computer will not get a response to its read statement, and typically your program will halt. (Addressable and listen-only modes are described in Chapter 14.)

Reading the plotter's output response removes it from the plotter's output buffer. If you do not read *all* of the output response, the plotter retains any remaining portion of the response until it is obtained by a subsequent read operation.

*Some computers (such as the HP Series 80 computers) require an input/output (I/O) ROM in order to obtain information from the plotter.

The plotter signals the end of its output response with an output response terminator, denoted in this manual by *[TERM]*. In an HP-IB configuration, the output response terminator is a carriage-return followed by a line feed (CR LF).

The plotter sends a signal through a wire in the interface cable concurrently with the output of the line feed (LF) portion of the output terminator. This signal is interpreted by some computers to mean that the read statement has been satisfied and program execution can resume.

Notes for RS-232-C Configurations

Even though output occurs automatically, you need to read the output response if you wish to use it in your program. It is best to read the output response immediately.

The plotter signals the end of its output response with an output response terminator, denoted in this manual by *[TERM]*. In an RS-232-C configuration, the default output response terminator is a carriage-return (CR). You can change the output response terminator using the ESC.M device-control instruction, as explained in Chapter 12.

Input Mask Instruction, IM

USES: Use the IM instruction to control 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. (The HP-IB service request and parallel polling are described in Chapter 14.)

SYNTAX: *IM* E-mask value (, S-mask value (, P-mask value)) *term*
or
IM *term*

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

*Error numbers 1, 2, 3, 4, 5, and 7 will be reported.

EXPLANATION: The parameters of the IM instruction are explained below. (These parameters set bits of the status byte, which is explained in detail in the *Output Status Instruction, OS*, later in this chapter.) If you are using an RS-232-C interface, you can only use the E-mask parameter. If you are using an HP-IB interface, you can use all three parameters.

1. **E-mask.** To report HP-GL errors, look up the error(s) you want to report in the following table. Then use the decimal value of the associated bit as the E-mask parameter value. To report more than one error, add the decimal values of those errors and use the sum as the parameter. For example, to report errors 3 and 5, you would use the sum of their bit decimal values (4 + 16 = 20) as the E-mask parameter (e.g. *IM20;*).

   When an HP-GL error occurs, each error number is compared with the E-mask bits you have specified. If the two match, the error bit (bit 5) of the status byte is set. Use the output error instruction, OE, (described later in this chapter) to read the error number.

E-Mask Bits

Error No.	Meaning	Bit No.	Decimal Value
1	Instruction not recognized	0	1
2	Wrong number of parameters	1	2
3	Bad parameter received	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

*Error 7 will be reported only if a cartridge is installed.

2. **S-mask.** To send a service request, look up the conditions you want to trigger a request in the following table. 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 trigger a service request, you would use the sum of their bit decimal values (1 + 4 = 5) as the S-mask parameter (e.g. *IM20,5;*).

   When one of the conditions occurs, a bit of the status byte changes value. Then, the status byte is 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; it will not be set again until one of the conditions specified by the S-mask value occurs.

S-Mask Bits

Status Bit No.	Meaning	Decimal Value
0	Pen down	1
1	P1 or P2 changed	2
2	Digitized point available	4
3	Initialized	8
4	Ready for data (paper loaded)	16
5	Error	32
6	Not used	64
7	Not used	128

3. **P-mask.** To specify which conditions will trigger 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 trigger a positive response to a parallel poll, use the sum (1 + 16 = 17) as the parameter (e.g. *IM20,5,17;*).

P-Mask Bits

Status Bit No.	Meaning	Decimal Value
0	Pen down	1
1	P1 or P2 changed	2
2	Digitized point available	4
3	Initialized	8
4	Ready for data (paper loaded)	16
5	Error	32
6	Not used	64
7	Not used	128

The following table summarizes the error conditions or unexpected results that might occur with the IM instruction.

Condition	Error	Plotter Response
no parameters	none	establishes default values
more than 3 parameters	2	uses first 3 parameters
parameter out-of-range	3	ignores instruction

Output Error Instruction, OE

USES: The OE instruction outputs an error number which corresponds to the first HP-GL error received by the plotter. Use this instruction to identify HP-GL errors in your programs.

SYNTAX: *OE term*

RESPONSE: HP-GL error number [TERM]

EXPLANATION: When the plotter receives the OE, it reports back an integer in the range of 0 through 7, corresponding to the first HP-GL error that occurred. The error numbers are defined on the next page.

HP-GL Error No.	Meaning
0	No error
1	Instruction not recognized
2	Wrong number of parameters
3	Parameter out-of-range
4	Not used
5	Unknown character set
6	Position overflow
7*	Polygon buffer overflow
8	Not used

*Error 7 is reported only if a cartridge is installed.

After an OE instruction is executed, bit position 5 of the status byte is cleared (if set).

Including a parameter with the OE instruction generates error 2 (wrong number of parameters), and the plotter executes the OE instruction anyway.

Example — Using OE to Identify HP-GL Errors

The example on the next page shows how to use the output error instruction, OE, to identify an HP-GL error in a program. In this example, program line 20 has two errors. The OE instruction reports back the first error only.

NOTE: HP Touchscreen (150) computers will require use of a separate file for output and input, only one of which may be open at a given time. Refer to Appendix A for an example. ■

```
10 REM 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 (wrong number of parameters)

Program Explanation

10 establishes HP-IB or RS-232-C interface conditions

20 initializes the plotter; selects pen 1; establishes initial pen position (note extra parameter); sends a bad instruction; sends the output error instruction

30 inputs the error number (from the OE instruction) to the computer

40 prints the error number on the computer's screen

Output Identification Instruction, OI

USES: The OI instruction outputs the plotter's model number. This information is useful in a remote operating configuration (where several plotters are connected to the computer), to determine which model of plotter is on-line.

SYNTAX: *OI term*

RESPONSE: 7440A [TERM]

EXPLANATION: The plotter always outputs its model number and letter as a 5-character string.

Using a parameter with the OI instruction generates error 2 (wrong number of parameters), and the plotter executes the OI instruction anyway.

Output Options Instruction, OO

USES: In a remote operating configuration, this instruction can be used to determine which options are available in the plotter that is on-line. Software packages use this feature to determine which plotter capabilities exist.

SYNTAX: *OO term*

RESPONSE: (The output response will be affected by the use of the graphics enhancement cartridge, as described below.)

EXPLANATION: The plotter outputs eight ASCII integers, separated by commas. Refer to the appropriate section below, depending on whether or not you are using a graphics enhancement cartridge.

Without a cartridge — The plotter will output the following information:

```
0,1,0,0,0,0,0,0  [TERM]
  ↑
  └──────────── Indicates plotter has pen select capability
```

With a cartridge — The plotter will output the following information:

```
0,1,0,0,1,1,0,1  [TERM]
  ↑     ↑ ↑   ↑
  │     │ │   └ Indicates plotter has configurable memory
  │     │ │       capability
  │     │ └──── Indicates plotter has polygon fill capability
  │     └────── Indicates plotter has arc and circle instructions
  └──────────── Indicates plotter has pen select capability
```

Output Status Instruction, OS

USES: The OS instruction outputs the decimal value of the status byte. Use this instruction in debugging operations and in digitizing applications.

SYNTAX: *OS term*

RESPONSE: status [TERM]

EXPLANATION: When the plotter receives an OS instruction, the internal 8-bit status byte is converted to an ASCII integer between 0 and 255. To read the status byte, have your computer read the output response. Then, refer to the following table to find the largest decimal value that can be subtracted from the output response. The condition corresponding to the decimal value has been met. For example, 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.

If you subtract the largest decimal value from the output response, and there is a remainder, look for the next largest decimal value. The corresponding condition has been met. Repeat this process until the remainder is zero. For example, when the plotter is first turned on, the status is 24. The plotter is ready for data (24 − 16 = 8) and initialized (8 − 8 = 0).

Bit No.	Meaning	Decimal Value
0	Pen down	1
1	P1 and P2 newly established; cleared by OP	2
2	Digitized point available; cleared by OD	4
3	Initialized; cleared by OS	8
4	Ready for data (paper loaded)	16
5	Error, cleared by OE	32
6	Request service (always 0 for OS: 0 or 1 for HP-IB serial poll)	64
7	Not used	128

Executing an OS instruction clears bit 3.

Using a parameter with the OS instruction generates error 2 (wrong number of parameters); the plotter executes the OS instruction anyway.

Output Actual Position and Pen Status Instruction, OA

USES: The OA instruction outputs the X,Y coordinates and pen status (up/down) associated with the pen position. You might use this information to position a label or figure, to determine the parameters of some desired window, or to determine the pen's current position if it has been moved using front-panel cursor keys.

SYNTAX: *OA term*

RESPONSE: X,Y, pen status [TERM]

EXPLANATION: The pen position and status are output to the computer as integers in ASCII, separated by commas. The X,Y coordinates are output in plotter units; their ranges are the current hard-clip limits. The pen status is either 0 (pen up) or 1 (pen down).

Using a parameter with the OA instruction generates error 2 (wrong number of parameters), and the plotter executes the OA instruction anyway.

Example — Using OA to Determine Pen Location and Status

The following example shows how to use the output actual position and pen status instruction, OA, to determine pen location and status. Using the front-panel cursor keys, move your pen to the desired location. Then run the following program.

```
10 REM Insert configuration statement here
20 PRINT #1, "IN;OA;"
30 INPUT #2,X,Y,P
40 PRINT X,Y,P
50 END
```

Program Explanation

10 establishes HP-IB or RS-232-C interface conditions

20 initializes the plotter; sends the output actual position and pen status instruction

30 computer reads the pen position and status from the plotter

40 displays the pen position and status on the computer's screen

Output Commanded Position and Pen Status Instruction, OC

USES: The OC instruction outputs the X,Y coordinates and pen status (up/down) associated with the last valid pen position instruction. You might use this information to position a label or figure, or determine the parameters of an instruction which moved the pen to the limits of some window. This instruction is especially useful when the pen is physically at the plotting limits and the pen position does not coincide with the instructed position, or when you want output in user units.

SYNTAX: *OC term*

RESPONSE: X,Y,pen status [TERM]

EXPLANATION: The pen position is output as decimal numbers or integers in ASCII, up to four significant digits (i.e. 4.9999). The pen status is output as an integer.

When scaling is off, the X,Y coordinates are output as integers in plotter units. When scaling is on, coordinates are output in user units. Any fractional portion of a parameter is used. In either case, the range is −32 768.0000 to 32 767.9999. The pen status is either 0 (pen up) or 1 (pen down). The plotter outputs a negative sign for negative numbers; positive signs are suppressed.

Using a parameter with the OC instruction generates error 2 (wrong number of parameters), and the plotter executes the OC instruction anyway.

Output Factors Instruction, OF

USES: The OF instruction outputs the number of plotter units per millimetre in each axis. This instruction lets you use the plotter with software which needs to know the size of a plotter unit.

SYNTAX: *OF term*

RESPONSE: 40,40 [TERM]

EXPLANATION: In response to an OF, the plotter outputs the ASCII integers 40 and 40, separated by a comma. These factors indicate that there are 40 plotter units per millimetre in the X-axis and in the Y-axis (0.025 mm per plotter unit).

Using a parameter with the OF instruction generates error 2 (wrong number of parameters), and the plotter executes the OF instruction anyway.

Summary of Output Response Types

The following table shows the number and type of items in the response to each HP-GL output instruction. The table includes all output instructions explained in Chapters 9, 10, and 11. Use this table when programming in languages such as FORTRAN which require you to specify the type of and number of digits in a variable.

Output Response Types

Instruction	Number of Parameters Returned*	Type and Range
OA	3	integers, all $\leqslant$ 5 digits
OC	3	decimals, all $\leqslant$ 11 digits
OD	3	integers, all $\leqslant$ 5 digits
OE	1	integer, 1 digit
OF	2	integers, 2 digits each
OH	4	integers, all $\leqslant$ 5 digits
OI	1	5-character string
OO	8	integers, 1 digit each
OP	4	integers, all $\leqslant$ 6 digits
OS	1	integer, $\leqslant$ 3 digits
OW	4	integers, all $\leqslant$ 5 digits

*In addition to these parameters, the output terminator [TERM] is always sent at the end of output, and commas are sent to separate parameters.

CHAPTER 11

CHAPTER 11

Digitizing

What You'll Learn in This Chapter

You can use your plotter as a digitizer as well as a plotter. Digitizing means moving the pen to a point on the plotting surface, entering the point, and sending the X,Y coordinates of that point to the computer.

In this chapter you'll learn how to:

- use HP-GL instructions to digitize
- verify that an X,Y coordinate point has been entered
- write a program for digitizing

The three digitize instructions are presented first, followed by programming advice.

NOTE: In order to use the plotter to digitize, your computer must be able to read data from the plotter. If you are programming in a high-level language, such as BASIC, you cannot digitize using HP Touchscreen (150) computers in an HP-IB configuration. Some computers (such as HP Series 80 computers), require an I/O ROM to obtain digitized points. ■

HP-GL Instructions Covered

DP	Digitize Point Instruction
OD	Output Digitized Point and Pen Status Instruction
DC	Digitize Clear Instruction

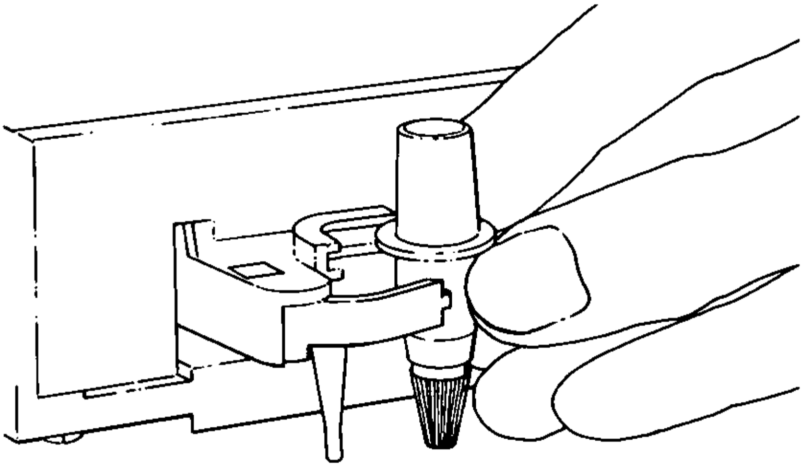
Terms You Should Understand

If you are unfamiliar with any of these terms, refer to the glossary at the end of the manual.

Digitize
Output Terminator

Preparing Your Plotter for Use as a Digitizer

In order to digitize using your plotter, insert a pen in the pen holder, as shown in the following illustration. We recommend using a *capped* pen to avoid leaving marks on your paper. For highest accuracy, digitize with the pen in the *down* position.



Loading a Pen In the Pen Holder

NOTE: If you are using the HP-IB interface, you cannot digitize while the plotter is in listen-only mode. When the plotter is in listen-only mode, it cannot send the coordinates of a digitized point to the computer. ■

Digitize Point Instruction, DP

USES: The DP instruction allows you to digitize points using the plotter. Use this instruction when inputting data for a graphics program or to obtain the coordinates of a point or points on a plot.

SYNTAX: *DP term*

EXPLANATION: When the plotter receives the DP instruction, automatic pen lift is suppressed. Use the cursor keys to move the pen to the desired position, lower the pen, and then press **ENTER**. 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 status (up/down).

Using a parameter with the DP instruction generates error 2 (wrong number of parameters), and the plotter executes the DP instruction anyway.

Output Digitized Point and Pen Status Instruction, OD

USES: The OD instruction is used to output the X,Y coordinates and pen status (up/down) associated with the last digitized point. Use this instruction after DP and **ENTER** to return the coordinates of the digitized point to the computer.

SYNTAX: *OD term*

RESPONSE: X,Y , pen status [TERM]

EXPLANATION: The pen position (X,Y coordinates in plotter units) and status (up/down) are output to the computer as three integers in ASCII, separated by commas and followed by the output terminator. The ranges of the X,Y coordinates are the hard-clip limits of the plotter. The pen status is either 0 (pen up) or 1 (pen down).

When the plotter receives the OD instruction bit position 2 of the status byte is cleared.

After you send the OD instruction, have your program read the plotter's output response. The timing of output depends on which interface you are using (RS-232-C or HP-IB). Refer to *Hints for Obtaining Plotter Output* in Chapter 10, for more information.

Using parameters with the OD instruction generates error 2 (wrong number of parameters), and the plotter executes the OD instruction anyway.

Digitize Clear Instruction, DC

USES: Use the DC instruction to terminate digitize mode. If you are using an interrupt routine in a digitizing program to branch to another plotting function, you could use DC to clear the digitize mode immediately after branching.

SYNTAX: *DC term*

EXPLANATION: When the plotter receives the DC instruction, digitize mode is terminated, and automatic pen lift is reactivated.

Using a parameter with the DC instruction generates error 2 (wrong number of parameters), and the plotter executes the DC instruction anyway.

Digitizing with the Plotter

When you are digitizing, make sure that a point has been entered before attempting to retrieve that point with the OD instruction. The three methods for doing this are explained next, and program examples are given.

Manual Method

The manual method is the easiest method of digitizing to understand. It is not efficient in applications where many points will be entered, or in cases where human intervention in program execution is not possible. To implement the manual method:

1. In a program, send a DP instruction to the plotter.
2. Next, use a statement that will cause the program to display or print a message prompting you to enter a point.

3. Follow the prompt with a statement that will cause the program to pause until instructed to continue. Using the BASIC INPUT statement and entering an empty string will work on some systems. Some versions of BASIC use statements such as STOP, WAIT, or PAUSE.

4. Move the pen to the point to be entered, using the front-panel cursor control keys. Complete final positioning with the pen down. Press the **ENTER** button on the plotter.

   NOTE: It is normal for the plotter to generate sound while the pen is down. ■

5. Now cause the program to resume. If you used an INPUT statement in step 3, press the RETURN key on the computer. The way you resume program execution depends on the statement you used to halt the program.

6. The next steps of the program, in order, should be an OD instruction to the plotter, a read statement by the computer to read the X,Y coordinates and the pen status, and then 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 instruction until you enter a point and cause the program to resume. The following example uses this method.

You can also perform a simpler procedure using OA or OC instead of OD. If you do, omit the DP in step 1 and pressing **ENTER** in step 4. Refer to Chapter 10 for descriptions of the OA and OC instructions.

Example — Digitizing Using the Manual Method

The following program digitizes a single point and displays the coordinates and pen status.

NOTE: Check your computer documentation to see how your computer reads input from the plotter, it may vary from the example shown here. HP Touchscreen (150) computers will require use of a separate file for output and input, only one of which may be open at a given time. Refer to Appendix A for an example. ■

```
10 REM Insert configuration statement here
20 PRINT #1, "DP;"
30 PRINT "Enter a point, then press RETURN"
40 INPUT N$
50 PRINT #1, "OD;"
60 INPUT #1, X,Y,P
70 PRINT X,Y,P
80 END
```

Monitoring the Status Byte

The second method monitors bit position 2 (the third least significant bit) of the plotter's status byte, which is set when a digitized point is available. Refer to the *Output Status Instruction, OS*, in Chapter 10 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. The status byte can be operated on arithmetically to check for the availability of a digitized point. Executing successive divisions of a number 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 example uses this method.

Example — Digitizing by Monitoring the Status Byte

The following sequence of BASIC instructions will check the proper bit of the status byte. In line 50, use your computer's BASIC read statement to read the status byte into a variable called Status. (INT is a function that returns the integer portion of a number.)

```
10 REM Insert configuration statement here
20 PRINT #1, "DP;"
30 PRINT "Enter a point by pressing ENTER"
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
```

Program Explanation

10	establishes HP-IB or RS-232-C interface conditions
20	prepares plotter to accept a digitized point
30	prompts you to enter a point on the plotter and press ENTER on the plotter
40	sends the output status instruction
50	reads the status
60	shifts bits right by two positions
70	if a point hasn't been obtained, reads status again
80	outputs the digitized point
90	reads X,Y coordinates and pen status (up/down)
100	displays X,Y coordinates and pen status

Example — Digitizing Many Points

In many applications, a large number of points need to be digitized. When the computer is used to monitor bit position 2, the data points may or may not be processed immmediately. 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 follows. When prompted to enter a point, use the cursor keys to move the pen to the desired position. Now press the **ENTER** button on the plotter. Continue for all 25 points. Their coordinates will be displayed on the computer's screen after they have all been entered.

```
10 REM  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 REM 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

Advanced programmers, who are thoroughly familiar with the HP-IB interface, polling techniques, and interrupts can use the third method, described in this section. 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 instruction (e.g. *IM 223 , 4 , 0 ;*) 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 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 simply send an OD 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 instruction (refer to Chapter 10). Because interrupts and polling are highly machine-dependent and beyond the scope of this manual, no examples are given. Refer also to the *Input Mask Instruction, IM*, in Chapter 10, and *Serial and Parallel Polling* in Chapter 14.

CHAPTER 12

CHAPTER 12

Device-Control Instructions

What You'll Learn in This Chapter

In order to use device-control instructions, you must have a plotter with an RS-232-C interface. Device-control instructions control internal plotter conditions, such as: buffer size, input/output conditions, and data flow. Chapter 13 explains the device-control instructions used to establish RS-232-C interface conditions and handshakes.

In this chapter you will learn how to use device-control instructions to:

- change the size allocations of the plotter's buffers
- set I/O (input/output) conditions
- obtain information about the plotter's status
- abort HP-GL instructions and clear the plotter's buffer

Device-Control Instructions Covered

ESC.T	Allocate Configurable Memory Instruction
ESC.@	Set Plotter Configuration Instruction
ESC.B	Output Buffer Space Instruction
ESC.L	Output Buffer Size When Empty Instruction
ESC.S	Output Configurable Memory Size Instruction
ESC.A	Output Identification
ESC.E	Output Extended Error Instruction

ESC.O	Output Extended Status Instruction
ESC.K	Abort Graphics Instruction
ESC.J	Abort Device-Control Instruction

Terms You Should Understand

If you are unfamiliar with any of these terms, refer to the glossary at the end of the manual.

Buffer
Decimal Code
Execute
I/O Buffer
Parse

What Is a Device-Control Instruction?

Device-control instructions serve two basic purposes: to control memory (buffer) allocation, and to control data transfer between the computer and the plotter (handshaking operations). In addition, some device-control instructions provide information about certain internal plotter conditions (similar to the HP-GL output instructions). The device-control instructions which refer specifically to initiating an RS-232-C interface and/or handshake are explained in Chapter 13.

The plotter executes device-control instructions as it receives them (as opposed to entering the I/O buffer and being executed on a first-in/first-out basis, as is the case with HP-GL instructions). The plotter recognizes these instructions because they begin with the single ASCII escape character, **ESC** (decimal code 27).

How and When Do You Send a Device-Control Instruction to the Plotter?

The principles for sending device-control instructions to the plotter are the same as for HP-GL: you must send them as a literal string in a computer output statement such as PRINT #, PRINT, OUTPUT, WRITE, or WRITELN (discussed in Chapter 1). Accessing the character **ESC** is similar to accessing any other single

ASCII character. You have two options: use the decimal code in your program or use your keyboard to send the character.

If you refer to the ASCII code table in Appendix B, you will see that 27 is the decimal code for the ASCII character **ESC**. Many implementations of BASIC use the function CHR$(27) to send the decimal code of the **ESC** character. If you are using a language other than BASIC, check your documentation to determine how to send the **ESC** character. Following is an example of sending the ESC.L device-control instruction in a BASIC program.

```
   PRINT #1, CHR$(27)+".L"
```

Another option is to use your keyboard to send the character. On some computers, you press a single **ESCAPE** key. On others, you can access **ESC** by pressing a combination of keys; for example, on HP Series 200 computers, you press **CTRL [**. For more information on these options, refer to *How to Send the Label Terminator and Other Nonprinting Characters* under the *Label Instruction, LB,* in Chapter 6.

If your computer immediately executes a function when the **ESCAPE** key is pressed, you must use the decimal code in your program, as described above.

How Do You Receive Information from the Plotter?

All device-control instructions with the word "output" in their title will cause information to be output from the plotter. The principles for reading this information from the plotter are the same as for HP-GL: to use the plotter's output response in a program, you must enter it into the computer with a computer input statement such as INPUT #, INPUT, ENTER, READ, or READLN.

Remember that an output response caused by a device-control instruction is available immediately. Read the information immediately to avoid potential errors. Refer to *Hints for Obtaining Plotter Output* in Chapter 10 for details.

Syntax for Device-Control Instructions

A complete device-control instruction includes at least the three-character sequence consisting of "**ESC**" and "." followed by the character or capital letter indicating its function (e.g., S, T, or @). Some instructions also include parameters and a terminator, according to the following syntax conventions.

These syntax conventions are used with the device-control instructions in this chapter and in Chapter 13.

ESC	Denotes the single ASCII character, **ESCAPE** (decimal code 27). On many computers you can access this character by striking a key (or keys) on the keyboard, or you can use a function such as CHR$ (27) (in BASIC) to send its decimal code.
[ ]	Brackets indicate that all enclosed parameters are optional.
( )	Parentheses indicate that each individual parameter is optional.
;	The semicolon separates parameters. **NOTE:** There is no semicolon between the three-character command sequence (e.g., **ESC**.M) and the first parameter. ■
:	The colon terminates any instruction which has parameters and can occur after any valid number of parameter entries. If the instruction doesn't have parameters, don't use the colon terminator.
[TERM]	Unless changed by an ESC.M instruction, all RS-232-C output responses include a carriage return [**CR**] as a terminator.
Default Values; Omitting Parameters	You can omit any parameter by entering only the semicolon delimiter; this sets the parameter to its default value. (Thus, unlike HP-GL, it *is* possible to send an instruction such as **ESC**.*M;;;10;13*:. The

parameters that have been omitted are set to their default values.) You can omit all parameters by placing only the colon terminator after the three-character device-control instruction.

Allocate Configurable Memory Instruction, ESC.T

USES: You must have a cartridge installed to use this instruction. The ESC.T instruction allocates memory into the two separate buffers of the plotter's memory. Use this instruction to customize memory allocation to fit the requirements of your application.

SYNTAX: ESC.T [(I/O buffer size;polygon buffer size)]:

Parameter	Format	Range	Default
physical I/O buffer	real	2 to 1974 bytes	1024
polygon buffer	real	0, 4 to 1972 bytes*	950

*If you specify 1 to 3 bytes, the plotter automatically sets the polygon buffer to 4 bytes.

EXPLANATION: The plotter's configurable memory can be divided between the physical I/O buffer and the polygon buffer. You can vary the number of bytes in each buffer, depending on the requirements of your application. The total number of bytes for both buffers cannot exceed 1974; if this happens, the plotter reduces both buffers to their default values.

The individual buffers are described next. Following these descriptions are some considerations to be aware of when executing the ESC.T instruction.

1. **Physical I/O Buffer.** The first parameter specifies the size, in bytes, to be allocated in the physical I/O buffer. Setting the physical I/O buffer too small slows down data transmission. If you are *not* using the polygon buffer, consider increasing the size of the physical I/O buffer. The larger the buffer, the more instructions the computer can send to the plotter at a

given time, freeing the computer to perform other tasks while the plotter executes instructions.

NOTE: In addition to storing HP-GL instructions, part of the physical I/O buffer is used for I/O functions. This portion of the buffer is called the *logical I/O buffer*. If you decide to change the size of the physical I/O buffer, read the *Set Plotter Configuration Instruction, ESC. @*, next in this chapter, for details about how the physical I/O and the logical I/O buffer interact. ■

2. **Polygon Buffer.** This parameter specifies the size of the the polygon buffer. All polygons defined by the wedge, rectangle, and polygon instructions (WG, EW, RA, RR, EA, ER, and PM) are stored in the polygon buffer. If your program *does not* use any of these instructions, reduce the size of this buffer to enable the I/O buffer to utilize the unused bytes. If your program uses any of these instructions, allocate enough bytes for the largest polygon. A good approximation is to allocate 11 bytes for every point in the largest polygon. The default size of the polygon buffer is sufficient for a polygon with 86 points.

   If you need to allocate buffer space more precisely, refer to the *Polygon Buffer* in Chapter 7 for a more exact method of determining the number of points in a polygon.

   NOTE: In an HP-IB configuration, 2000 bytes are available for polygons. ■

The following table summarizes the error conditions or unexpected results that might occur with the ESC . T instruction.

Condition	Error	Plotter Response
more than 2 parameters	none	ignores extra parameters
only 1 parameter specified	none	remaining bytes allocated to other buffer
sum of both parameters > 1974	none	both parameters set to their default values
parameter value is less than the minimum (2 or 4) number of bytes	none	parameter set to minimum number of bytes
odd number of bytes specified	none	next lower even number allocated

Hints for Using the ESC.T Instruction to Allocate Memory

When the ESC.T instruction is executed, it performs the following operations in this sequence:

- Clears the I/O buffer.
- Clears the polygon buffer and allocates memory as specified in its parameters.
- Resets the parser.

Because of this sequence of operations, there are two programming hints that you should be aware of, described next.

1. **Allocate buffer sizes at the beginning of a program, when the buffers are empty — before you send any HP-GL instructions to the plotter.** There are two reasons for this. First, if the buffers are not empty, all data stored in them will be lost when they are cleared by ESC.T. Second, if you execute an instruction that makes use of polygon buffer space and you have not previously allocated enough space, HP-GL error 7 (polygon buffer overflow) will occur. If error 7 occurs, check the buffer allocations with the output configurable memory size instruction, ESC.S, described later in this chapter.

2. **After executing the ESC.T instruction, execute the ESC.L instruction.** The ESC.T instruction can take a few milliseconds to clear the buffers and complete all memory allocations. Therefore, your program should wait for this to happen before sending data to the plotter. The easiest way to do this is to use the ESC.L instruction. This instruction waits until the I/O buffer is empty and then outputs the size of the logical I/O buffer. Therefore, the ESC.L instruction does not provide an output response until the ESC.T instruction has finished.

   In this application of the ESC.L instruction, the numerical value of the logical I/O buffer size is not important. However, your program should read the output response before sending more data, in order to avoid potential errors. For examples of allocating memory with ESC.T and ESC.L, refer to the *Fill*

Polygon Instruction, FP, in Chapter 7. The ESC.L instruction is described later in this chapter.

You can also execute an ESC.L instruction *before* an ESC.T instruction. Do this to be sure the I/O buffers are empty. This method can prevent the data loss that would occur when ESC.T was executed.

Another way to determine when the I/O buffer is empty is to check bit 3 of the extended status word, using the ESC.O instruction, as described later in this chapter.

Set Plotter Configuration Instruction, ESC.@

USES: Use the ESC.@ instruction to set the size of the logical I/O buffer, and configure I/O conditions such as hardwire handshaking and normal or block mode.

SYNTAX: **ESC**.@ [(logical I/O buffer size);(I/O conditions)]:

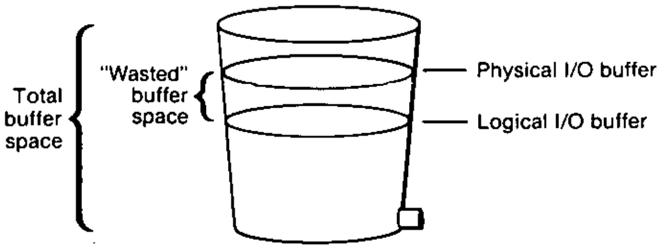
Parameter	Format	Range	Default
logical I/O buffer size*	real	1 to 1974	1024 or size of physical I/O buffer
I/O conditions	real	0 to 31	hardwire handshake, normal mode

*This parameter can only be used if the cartridge is installed.

EXPLANATION: The two parameters of the ESC.@ instruction are interpreted as follows.

1. **Logical I/O Buffer.** You must have a cartridge installed to use this parameter. This parameter specifies the size of the logical I/O buffer, which is part of the physical I/O buffer (described previously under the *Allocate Configurable Memory Instruction, ESC.T.*) The logical I/O buffer limits plotter I/O functions, such as the number of HP-GL instructions waiting to be parsed, and threshold levels in certain handshaking methods.

For maximum efficiency, the logical I/O buffer should be set to the same size as the physical I/O buffer; this is the default setting of the logical I/O buffer. Since the logical I/O buffer cannot be larger than the physical I/O buffer, increase the size of the physical I/O buffer *before* increasing the size of the logical I/O buffer. If the physical I/O buffer has been decreased, the logical I/O buffer is automatically decreased to the same size.



2. **I/O Conditions.** To set I/O conditions, look up the decimal values of the conditions you need in the following table. Then use the decimal value of the associated bit as the 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 enable hardwire handshake and block mode, you would use the sum of their bit decimal values (1 + 16 = 17) as the second parameter (e.g., **ESC**.@ *; 17* :).

   If you omit this parameter, the plotter sets hardwire handshake and normal mode.

Bit No.	Description	Logic State	Decimal Value
0	Disable hardwire handshake	0	0
	Enable hardwire handshake (power-on default)	1	1
	You must have a cartridge installed to use this bit		
4	Enable normal mode (default)	0	0
	Enable block mode	1	16

Example — Setting the Size of the Logical I/O Buffer and I/O Conditions

You must have a cartridge installed to implement the instructions shown in this example. In the following sequence of instructions, ESC . T sets the size of the physical I/O buffer to 1024 bytes (and defaults the polygon buffer). ESC . L ensures the allocation is complete, and ESC . @ sets the size of the logical I/O buffer to 1024 bytes (the same size as the physical I/O buffer), enables block mode, and disables hardwire handshake.

```
ESC.T1024:ESC.LESC.@1024;16:
```

Remember to add a computer-dependent read statement after ESC . L to read the output response, and avoid potential errors.

The following table summarizes the error conditions or unexpected results that might occur with the ESC . @ instruction.

Condition	Error	Plotter Response
logical I/O buffer size < 0	12	sets logical I/O buffer equal to physical I/O buffer
logical I/O buffer parameter > physical I/O buffer size	none	sets logical I/O buffer equal to physical I/O buffer

Output Buffer Space Instruction, ESC . B

USES: The ESC . B instruction outputs the number of bytes available in the logical I/O buffer without waiting for the buffer to become empty. Use this instruction to check the size of the logical I/O buffer when you have changed the size with the ESC . @ instruction, or when the logical I/O buffer has been automatically reduced by a reduction in the physical I/O buffer (with the ESC . T instruction). You can also use this instruction in a software checking handshake, as explained in Chapter 13.

SYNTAX: **ESC** . B

RESPONSE: available buffer space [TERM]

EXPLANATION: The ESC . B instruction outputs the number of unused bytes in the I/O buffer. When you turn the plotter on, the output response is 60 without a cartridge; 1024 with a cartridge. If you use the ESC . T or the ESC . @ instructions to change the buffer size, the response can be as low as 0 bytes.

If your program continually interrogates the plotter until the response to ESC . B indicates a specific amount of available space, have the program pause a few seconds between successive ESC . B instructions. This pause allows the plotter time to execute other instructions, thus increasing the amount of available space.

Output Buffer Size When Empty Instruction, ESC . L

USES: The ESC . L instruction outputs the size, in bytes, of the logical I/O buffer. Use this instruction to determine when the I/O buffer is empty. This is particularly helpful to determine when a memory allocation instruction (ESC . T) has been completed, before allowing the plotter to parse the next instruction.

SYNTAX: **ESC** . L

RESPONSE: buffer size [TERM]

EXPLANATION: The ESC . L instruction outputs an integer which is the size of the logical I/O buffer. The response is not transmitted by the plotter until the buffer is empty.

When the plotter is turned on, the output response without a cartridge is 60; with a cartridge installed, the output response is 1024. The response range can be as low as 1 or as large as 1974, if the size is altered using the ESC . T instruction.

NOTE: If your plotter is in block mode, using ESC . L may disrupt communication. If you use an ESC . L while in block mode, send it immediately after an ESC . E instruction. Send the ESC . E instruction, read the response, then send the ESC . L instruction and read its response, before sending additional HP-GL instructions. ■

Output Configurable Memory Size Instruction, ESC . S

USES: You must have a cartridge installed to use this instruction. The ESC . S instruction outputs the total number of bytes allocated in the configurable memory, or the number of bytes in its two buffers. Use this instruction to determine current memory allocation or to confirm the allocation performed by the ESC . T instruction.

SYNTAX: **ESC**.S (buffer size) :

Parameter	Format	Range	Default
buffer	real	0 to 2	0

RESPONSE: buffer [TERM]

EXPLANATION: The ESC . S instruction outputs the number of bytes allocated in the buffer, as specified by its parameter. This response is an integer with a value between 0 and 1974.

Executing the ESC . S instruction without any parameters (**ESC**. *S* :) causes the plotter to output the *total* number of bytes in both buffers of the configurable memory. The valid range of parameters and their functions follows.

Parameter	Function
0	requests total number of bytes in both buffers of the configurable memory (always 1974)
1	requests number of bytes in the physical I/O buffer
2	requests number of bytes in the polygon buffer

The following table summarizes the error conditions or unexpected results that might occur with the ESC . S instruction.

Condition	Error	Plotter Response
parameter < 0	12	outputs 1974
parameter > 2	none	outputs 0

Output Identification Instruction, ESC . A

USES: The ESC . A instruction outputs the plotter's model number. Use this instruction to obtain the plotter's model number when you want to know which plotter is currently on-line.

SYNTAX: **ESC**.A

RESPONSE: plotter model number, firmware revision level [TERM]

EXPLANATION: The ESC . A instruction is similar to the HP-GL OI instruction in that they both output the plotter's model number. However, the ESC . A instruction is executed immediately (instead of entering the buffer, as the OI instruction does), and ESC . A also outputs the firmware revision level.

The model number is always 7440A, output in the character string. The firmware revision level is output as integers. The model number and firmware revision level are separated by a comma.

Output Extended Error Instruction, ESC . E

USES: The ESC . E instruction outputs a number that defines any RS-232-C-related I/O error. Use this instruction in program debugging to determine which errors have occurred. In addition, if you have a cartridge installed, you can use this instruction in conjunction with the ESC . @ instruction to perform block I/O error checking, as explained in Chapter 13.

SYNTAX: **ESC**.E

RESPONSE: error condition [TERM]

EXPLANATION: The ESC . E instruction outputs the number of the I/O error as an integer in ASCII. The range is 0, or 10–16, as described in the following table.

NOTE: In block mode, the block of data is accepted if there are no errors. The block of data is discarded if errors are found. ■

Error No.	Meaning
0	No I/O error has occurred.
10	New output has been generated 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	Invalid character received after first two characters (**ESC** .) in a device-control instruction.
12	Invalid character received while parsing a device-control instruction. The parameter containing the invalid character 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 termination) or the next **ESC** character (abnormal termination) is received.

(Table Continues)

Error No.	Meaning
	NOTE: The receipt of a character other than another parameter, a semicolon, or a colon will result in error 12 overwriting error 14. ■
15	A framing error or parity error has been detected.
16	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.

Output Extended Status Instruction, ESC . O

USES: The ESC . O instruction outputs the plotter's extended status. Use this instruction to obtain information about the current operating status of the plotter.

SYNTAX: ESC.O

RESPONSE: plotter operating status [TERM]

EXPLANATION: The ESC . O instruction is similar to the HP-GL OS instruction in that they both output information about the plotter's status. However, the ESC . O instruction outputs more information, and it is executed immediately.

To determine the plotter's status, use the following procedure. Have your program read the output response. Then refer to the following table to find the largest decimal value that can be subtracted from the output response. The bit corresponding to this value is set and the corresponding condition has been met. For example, if the output response is 128, the largest decimal value that can be subtracted is 128 (128 − 128 = 0). This means the corresponding condition (standard mode*) has been met.

If you subtract the largest decimal value from the output response, and there is a remainder, look for the next largest decimal value. The corresponding condition has been met. Repeat

*Standard mode is always set. This mode is for compatibility with other HP plotters.

this process until the remainder is zero. For example, if the output response is 144, standard mode (144 − 128 = 16) and the view state are active (16 − 16 = 0).

Bit No.	Meaning	Logic State (0 = OFF, 1 = ON)	Decimal Value
0–2	Not used	0	0
3**	I/O buffer is not empty	0	0
	I/O buffer is empty and ready for data	1	8
4 5	Remote state; processing HP-GL instructions	0 0	0 0
4 5	Not-ready state; paper is not loaded, graphics are suspended	1 0	16 0
4 5	View state; graphics temporarily suspended	0 1	0 32
6	Not used	0	0
7	Standard mode; always set	1	128
8–15	Not used	0	0

**It is inefficient to continually poll this bit. An alternative is to use ESC . L, as described later in this chapter.

If you only care about certain bits being set, you can check those bits using one of the methods described under *Monitoring the Status Byte,* in Chapter 11.

If your application requires that your program continually interrogate the plotter with ESC . O until a certain bit has been set, have the program pause several seconds between successive ESC . O instructions. The pause allows the plotter time to execute other instructions, giving the plotter the opportunity to set the desired bit.

Abort Graphics Instruction, ESC . K

USES: The ESC . K instruction aborts any partially parsed HP-GL instructions and clears the remaining instructions from the buffer. Use this instruction as part of an initialization sequence when starting a new program or to terminate plotting of any HP-GL instructions remaining in the buffer.

SYNTAX: ESC.K

EXPLANATION: This instruction aborts any partially parsed HP-GL instructions, but permits the instruction being executed to finish.

All pending HP-GL instructions in the buffer are discarded, and the parser is reset. Any data entered since block I/O error checking was enabled (with ESC . @) is aborted.

Abort Device-Control Instruction, ESC . J

USES: The ESC . J instruction aborts the execution of device-control instructions. You can use this instruction in an initialization sequence when you first turn on the plotter.

SYNTAX: ESC.J

EXPLANATION: This instruction aborts any device-control instruction that may be partially parsed or executed. The remaining parameters of partially parsed instructions are defaulted. All pending or partially transmitted output requests, from HP-GL or device-control instructions (including handshake outputs), are immediately terminated.

Intermediate output operations, such as turnaround delay and echo suppression, are aborted and buffer input is enabled. The handshake and output mode parameters remain as specified. (Refer to Chapter 13 for additional information about handshaking and the associated parameters.)

CHAPTER 13

RS-232-C/CCITT V.24 Interfacing

What You'll Learn in This Chapter

This chapter is for owners of plotters with an RS-232-C interface. This chapter explains the steps necessary to establish data compatibility between a computer and your plotter. Once compatibility has been established, a handshake is needed to control the manner in which data is transferred between the devices. Depending on your knowledge of interfacing and handshaking, you will probably only need to read the sections on interfacing, and the specific handshake you will use.

NOTE: All information in this chapter applies equally to RS-232-C and CCITT V.24 interfaces, except where noted. For purposes of simplicity, only the term RS-232-C is used. ■

Appendix A of this manual repeats the program listings from the Operating Manual, and highlights the program lines that establish critical interface and handshake conditions. Use these program lines in your application programs, in the sequence shown in the appendix, to set interfacing and/or handshaking conditions. Read the rest of this chapter if your computer is *not* listed, or if you need more detail.

This chapter contains information about:

- interfacing; establishing communication compatibility between your plotter and computer
- choosing a handshake
- how to use device-control instructions to initiate a handshake
- specific device-control instructions used in interfacing and handshaking
- data transmission modes
- automatic modem-disconnection modes

Refer to Appendix A, *Interfacing Supplementary,* if you need the RS-232-C/CCITT V.24 pin allocations. For additional information about interfacing and handshaking, the *RS-232-C Interfacing and Handshaking Guide*, Application Note 6 (Part No. (11)5953-9770) is available through HP Sales and Support offices.

Device-Control Instructions Covered

ESC.P	Set Handshake Mode Instruction
ESC.M	Set Output Mode
ESC.N	Set Extended Output and Handshake Mode
ESC.H	Set Handshake Mode 1 Instruction
ESC.I	Set Handshake Mode 2 Instruction
ESC.B	Output Buffer Space Instruction
ESC.E	Output Extended Error Instruction
ESC.R	Reset Instruction

What Is Interfacing, Why Is It Necessary?

Two RS-232-C devices may not be able to communicate, once interconnected. Interfacing establishes communication by matching a set of conditions between the computer and the plotter. The following section provides a checklist of the interface conditions you must set to achieve data and timing compatibility between your plotter and computer. The requirements of your computer system will dictate the interface conditions you must set on your plotter.

Follow the instructions in this section to establish compatibility. Then, read about initiating a handshake. *Handshaking is needed in addition to interfacing.*

Check Plotter-to-Computer Interconnection

Confirm that you have the required RS-232-C interface cable, and connect the plotter as shown in the Operating Manual. Read the rest of this chapter if your computer is *not* listed, or if you need more detail.

Setting Up the Plotter: a Checklist

The following table lists the plotter's factory-set interface conditions. *To achieve compatibility, you must set your plotter's switches to be compatible with your computer.* First, check your system's documentation to determine which values your computer uses. Then, write your computer's requirements in the table below, in the Computer Requirement column. This should help you determine if adjustments need to be made to achieve compatibility. The subsequent paragraphs tell you how to use the plotter's rear-panel switches to change plotter conditions.

Read the paragraphs pertaining to the conditions you need to change.

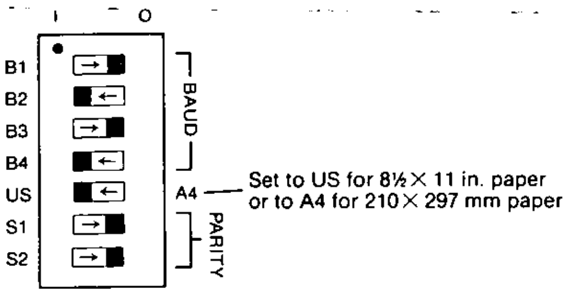
RS-232-C Interface Condition Checklist

Condition	Plotter Factory-Set Default	Computer Requirement
Baud rate	9600	________
Stop bits	1	________
Data bits	8*	________
Parity	off	________
Handshake	hardwire**	________

*7 data bits plus 1 bit which may contain parity information. This is a fixed condition; it cannot be changed.

**Handshakes are not switch-selectable. To select a handshake other than hardwire, refer to the section on *Handshaking Guidelines,* later in this chapter.

If your plotter is set to the factory-set default conditions, the rear-panel switches will be set as shown in the following illustration. To use the defaults, make sure the switches are set as shown here.


I
O
B1
B2
B3
B4
US
S1
S2
BAUD
A4
PARITY
Set to US for 8½ × 11 in. paper or to A4 for 210 × 297 mm paper

Factory-Set Switch Settings

1. Set Baud Rate and Stop Bits

If your computer is listed in Chapter 3 of the Operating Manual, refer to that chapter for recommended baud rate and stop bit switch settings for your computer. If your computer is not listed, check your system documentation to determine the baud rate at which your computer sends data, and the number of stop bits required.

Refer to the following table and set the plotter's rear-panel switches to match your computer's baud rate and stop bit settings.

Baud Rate and Stop Bit Switch Settings

	Switch Settings							
	1 Stop Bit				2 Stop Bits			
Baud Rate	**B1**	**B2**	**B3**	**B4**	**B1**	**B2**	**B3**	**B4**
75	—	—	—	—	1	0	0	0
110	—	—	—	—	0	1	0	0
150	1	1	0	0	—	—	—	—
200	0	0	1	0	—	—	—	—
300	1	0	1	0	1	1	0	1
600	0	1	1	0	0	0	1	1
1200	1	1	1	0	1	0	1	1
2400	0	0	0	1	0	1	1	1
4800	1	0	0	1	1	1	1	1
9600*	0	1	0	1	0	0	0	0

*Factory-set defaults are 9600 baud and 1 stop bit.

2. Data Bits

The plotter is configured to use standard 8-bit ASCII code. There are seven data bits and one bit that may be used for parity error checking. Your data must be in 8-bit ASCII data format. (If it is not in this format, you will need to use some sort of protocol converter.)

3. Set Parity

If your computer is listed in Chapter 3 of the Operating Manual, refer to that chapter for recommended parity switch settings. If your system is not listed, check your system documentation to determine whether or not your computer uses parity checking.

If your computer requires parity, use the plotter's rear-panel switches (S1 and S2) to set parity to "odd" or "even", according to your computer system's requirements. If your computer does *not* require parity, leave the plotter's parity switches in the "off" position.

Refer to the following table and set the plotter's rear-panel switches to match your computer's parity requirement.

Parity Switch Settings

	Switch Settings	
Parity	**S1**	**S2**
Off	0	0 or 1*
Even	1	0
Odd	1	1

← Factory-set default

*Setting S2 to 0 sets parity bits sent by the plotter to 0 (space parity); setting S2 to 1 sets the parity bits to 1 (mark parity).

Handshaking Guidelines

Since computers send data faster then a plotter can process it, a handshake method is required to prevent data from being lost. The plotter can use any of four handshakes: hardwire (default), Xon-Xoff, enquire/acknowledge, and software checking. Most personal computers use hardwire handshake.

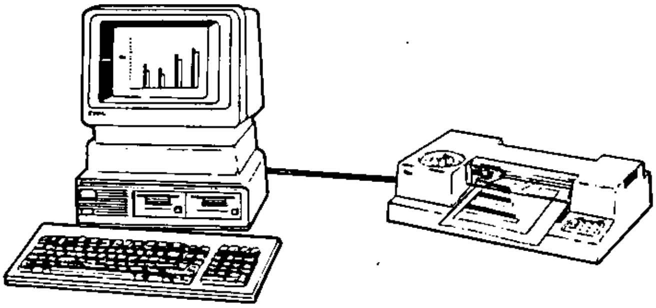
Consult your system's documentation or the installation manual for your computer and/or graphics software package for information on which handshake to use. Most graphics software packages designed for use with Hewlett-Packard RS-232-C plotters contain the instructions necessary to set up a handshake. You may need to fill in parameters suitable to your system, to be used by the software. The information you need to do this is found in the installation guides for the software.

If your system documentation recommends a handshake, read the section appropriate to that handshake, later in this chapter. *You must know your computer's requirements to have your computer and plotter communicate efficiently*. If there isn't a recommendation in your documentation, the following section should help you choose an efficient handshake.

The sections on Xon-Xoff, enquire/acknowledge, and software checking handshakes contain program segments that illustrate how to initiate a handshake using device-control instructions. Each instruction used is explained in detail at the end of this chapter. Review Chapter 12 for device-control instruction syntax.

Plotters Hardwired to the Computer

If your plotter is hardwired to your computer, it means there is no intermediate hardware between the plotter and computer; a cable goes directly from plotter to computer. This is the only case in which the hardwire handshake can be used, but other handshakes may be used. Sometimes the term automatic handshake is used to refer to this type of handshake. When you turn on the plotter, it is already set to use a hardwire handshake.



Plotter Hardwired to a Computer

If You Can't Use Hardwire Handshake . . .

If your plotter is not hardwired, check your system documentation to determine which handshake(s) your computer can use. The Xon-Xoff and enquire/acknowledge handshakes are efficient handshakes. Use a software checking handshake if you can't use any of the other handshakes.

Xon-Xoff and some enquire/acknowledge handshakes are implemented in the computer's operating system or device driver. Therefore, the computer must support these handshakes for you to be able to use them. Software checking and one type of enquire/acknowledge handshake are implemented in the program, and can be used on any computer.

The Xon-Xoff handshake is efficient when sending variable or fixed-length records. Although the enquire/acknowledge handshake is quite efficient, it is inefficient when there is a wide range of record lengths. There are two types of enquire/acknowledge handshakes. One can be implemented in the computer operating system; the computer *must* be set up to use this handshake. The other type of enquire/acknowledge handshake is implemented in your program.

The software checking handshake is the least efficient handshake. Its advantage is that it works on any computer that supports two-way communication and requires no knowledge of your operating system or computer's handshaking characteristics. You may want to use this handshake at least temporarily to get a program running. To use this handshake, you must include device-control instructions in your program to repeatedly check the availability of I/O buffer space.

Now that you have an idea of which handshake to use, read the corresponding section presented next.

Hardwire Handshake

As the name implies, the hardwire handshake takes place in the hardware rather than the software. To use hardwire handshake there must not be intermediate hardware (such as modems) between the plotter and computer. Since hardwire handshake is the plotter's default handshake, you don't have to use device-control instructions in your program to initiate it. Check your computer's documentation to determine if your computer can use hardwire handshake. Most personal computers can use hardwire handshake.

Since computers are generally capable of sending data faster than a plotter can process it, a handshake method is needed to prevent data from being lost. In a hardwire handshake, the computer monitors one of the interface lines from the plotter. When the plotter has room in its buffer for data, it signals the computer to send data by setting the line high (turning it on). The plotter's buffer has a threshold level that determines when the buffer is so full that it is in danger of overflowing and losing data. When the threshold level is reached, the plotter sets the 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 to send data. This system prevents the computer from over-filling the plotter's I/O buffer (buffer overflow) and losing data.

Initiating Hardwire Handshake Using Device-Control Instructions

When you turn the plotter on, it is already set to use hardwire handshake. To initiate hardwire handshake in your program, you can use the ESC . P or the ESC . @ device-control instructions. This would be useful to override a different handshake setting. This section will tell you how to initiate a hardwire handshake using device-control instructions, and provides examples of how they are used. Each device-control instruction is explained in detail, at the end of this chapter.

Using **ESC** . *P3:* is the simplest way to initiate a hardwire handshake with device-control instructions, because the instruction sets several parameter values for you. If your computer requires parameters values other than those set by **ESC** . *P3* :, use ESC . @, ESC . M, and ESC . N at the beginning of your program. The following examples show device-control instructions that you should include at the beginning of every program to set up a hardwire handshake.

Example — Using ESC . P to Initiate Hardwire Handshake

Using the set handshake mode instruction, ESC . P, with a parameter of 3 initiates a hardwire handshake with the values listed below.

ESC . P3:

Output terminate character	= 13 (ASCII, **CR**)
Threshold level	= 20 bytes without cartridge 80 bytes with cartridge

Be aware that these parameter values may not be the best for your computer. If your computer requires other values, note the following:

ESC . M will change the output terminate character
ESC . I will change the threshold level

Override parameter values set by ESC.P by placing the appropriate instruction *after* the ESC.P instruction.

Example — Using ESC.@ and ESC.I to Initiate Hardwire Handshake

This example uses the ESC.@ and ESC.I instructions to initiate a hardwire handshake. You might want to use ESC.@ if hardwire handshake had been disabled by a previous program.

In this example, the size parameter has been omitted; therefore, the plotter will use the default buffer size. The second parameter enables the hardwire handshake. The ESC.I instruction sets a threshold level of 10 bytes. Since the other parameters of ESC.I are unnecessary for a hardwire handshake, they are not included.

```
ESC.@;1:
ESC.I10:
```

These instructions specify the parameter values as:

Buffer size	= 60 bytes (default without cartridge) 1024 bytes (default with cartridge)
Hardwire handshake	= 1 (enabled)
Threshold level	= 10 bytes

Xon-Xoff Handshake

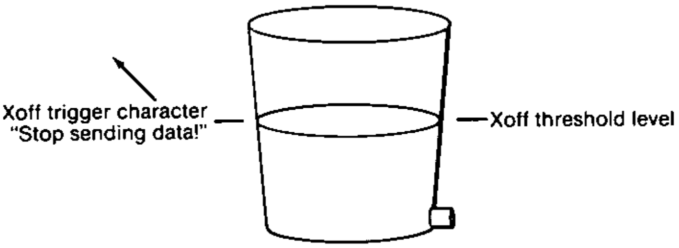
Check your system documentation to determine if you can use the Xon-Xoff handshake. The following paragraphs describe how the Xon-Xoff handshake prevents data loss. The subsequent section describes how to initiate Xon-Xoff handshake, using device-control instructions in your program.

To understand the way Xon-Xoff handshake works, think of the plotter's buffer as a bucket, which serves to contain water (data). The water source in this discussion is the computer. As water is used (data is processed), it drains out of a spout in the bottom of the bucket.

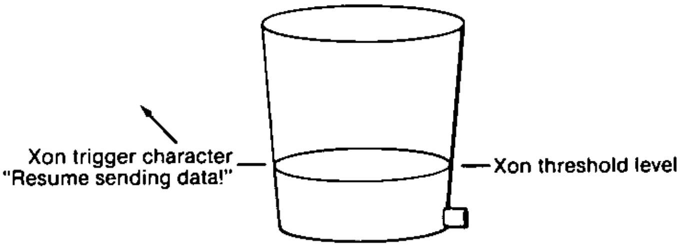
The bucket has two water lines. The uppermost line (Xoff threshold level) indicates when the bucket is about to overflow. The lower line (Xon threshold level) indicates when more water is needed. To communicate about water level, special characters are used (the Xon and Xoff trigger characters).

The following diagrams illustrate the way the Xon-Xoff handshake works.

1. Water enters the bucket faster than it can flow out, and the bucket starts to fill.
2. The water level reaches the overflow level and a danger signal is sent to the water source, telling it to turn off the water.



3. There is a delay between the time the warning signal is sent and the time the water is shut off. Allow extra room for water sent during this delay, to avoid losing water.
4. Water continually drains out of the bucket. When the water level reaches the lower line, a signal is sent to the source to turn the water on again.



The overflow level of the plotter's buffer (the Xoff threshold level) is automatically set by the plotter. When the level of data reaches the Xoff threshold level, the Xoff trigger character is sent to the computer, saying "Stop sending data." The plotter's buffer empties as data is processed. When the buffer is only half full, the Xon threshold level is reached. At this point, the Xon trigger character is sent to the computer, saying "There's enough room in the buffer for more data." This process is repeated until all of the data has been sent.

Initiating Xon-Xoff Handshake Using Device-Control Instructions

This section will tell you how to initiate an Xon-Xoff handshake using device-control instructions, and provides examples of how they are used. Include the device-control instructions at the beginning of every program. Each instruction is explained in detail at the end of this chapter.

You can initiate Xon-Xoff handshake from your program, using ESC . P or the ESC . N and the ESC . I device-control instructions. Using ESC . P is the simplest way to initiate an Xon-Xoff handshake, because the instruction sets several parameter values for you. If you want to set parameter values other than those set by **ESC** . *P1* :, use ESC . N in conjunction with ESC . I.

Example — Using ESC . P to Initiate an Xon-Xoff Handshake

Using the set handshake mode instruction, ESC . P, with a parameter of 1 initiates an Xon-Xoff handshake with the parameter values listed below.

ESC . P1 :

This instruction specifies the parameters as:

Intercharacter delay	= 10 milliseconds
Xoff trigger character	= 19 (ASCII **DC3**)
Xoff threshold level	= 20 bytes without cartridge 80 bytes with cartridge
Xon trigger character	= 17 (ASCII **DC1**)
Turnaround delay	= 50 milliseconds
Echo terminate character	= 10 (ASCII **LF**)
Output terminate character	= 13 (ASCII **CR**)

Be aware that these parameter values may not be the best for your computer. If your computer requires other values, note the following:

ESC . I will change the Xoff trigger character and Xoff threshold level.
ESC . N will change the intercharacter delay and Xoff trigger character.
ESC . M will change the turnaround delay, echo terminate, and output terminate character.

Override parameter values set by **ESC** . *P1* : by placing the appropriate instruction *after* ESC . P.

Example — Using ESC . N and ESC . I to Initiate Xon-Xoff Handshake

The following instructions will initiate an Xon-Xoff handshake. This example illustrates a handshake where the computer does not require an intercharacter delay. Since the second parameter of the ESC . I instruction (enquiry character) is not needed in an Xon-Xoff handshake, omit it with a semicolon, or specify it as zero.

```
ESC.N0;19:
ESC.I10;;17:
```

These instructions specify the parameter values as:

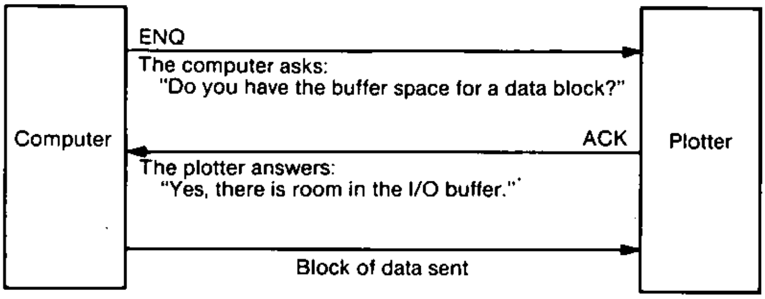
Intercharacter delay	= 0
Xoff trigger character	= 19 (ASCII **DC3**)
Xoff threshold level	= 10 bytes
Enquiry character	= omitted with Xon-Xoff
Xon trigger character	= 17 (ASCII **DC1**)

These are commonly used Xon-Xoff trigger character values. Check your system documentation to determine which values are required.

Enquire/Acknowledge Handshake

The purpose of the enquire/acknowledge handshake is to prevent the computer from sending the plotter more data than its buffer can accommodate. To communicate about the availability of buffer space, special characters are used: the enquiry character and acknowledgment string. The computer sends the enquiry character to the plotter, asking, "Do you have room in your buffer for data?" Before sending data to the plotter, the computer waits until it receives an acknowledgement string from the plotter, signaling, "Yes, I have room in my buffer for data." In this way, the plotter is never sent more data than its buffer can accommodate.

This handshake method derives its name from the two ASCII characters, **ENQ** and **ACK**, used on some Hewlett-Packard systems as the enquiry character and acknowledgment string. The diagram below illustrates how an enquire/acknowledge handshake works.


ENQ
The computer asks:
"Do you have the buffer space for a data block?"
Computer
ACK
Plotter
The plotter answers:
"Yes, there is room in the I/O buffer."
Block of data sent

How an Enquire/Acknowledgement Handshake Works

Choosing the Proper Enquire/Acknowledge Handshake

There are two kinds of enquire/acknowledge handshakes: one is usually handled in the program (mode 1), and the other is usually controlled by the computer operating system (mode 2). If your computer documentation lists the enquire/acknowledge handshake, you can probably use mode 2. Mode 1 is a form of a software checking handshake and can be used on all computers. A mode 1 handshake is time consuming because it requires an extra read and write statement each time a block of data is to be sent.

Initiating Enquire/Acknowledge Handshake Using Device-Control Instructions

You can initiate mode 1 from your program using the ESC.H device-control instruction. You can initiate mode 2 using ESC.P or ESC.I in conjunction with ESC.M and ESC.N, if your system requires either of the delays set by these two instructions. The capabilities and requirements of your computer system will dictate which mode you use.

The following table summarizes the minimum instructions and parameters needed to use modes 1 and 2. Each instruction is explained in detail, at the end of this chapter.

Enquire/Acknowledge Instructions and Parameters Used in Handshake Response

MODE 1 **Usually Placed in** **Application Program**	**MODE 2** **Usually Placed in** **Operating System**
ESC.H	ESC.I
block size	block size
enquiry character	enquiry character
acknowledgment string	acknowledgment string
ESC.M	ESC.M
turnaround delay	turnaround delay
output trigger character	ESC.N
echo terminate character	intercharacter delay
output terminator	immediate response string
output initiator	
ESC.N	
intercharacter delay	
immediate response string	

Example — Using ESC.P to Initiate Enquire/Acknowledge Handshake (Mode 2)

This example uses the ESC.P instruction to initiate an enquire/acknowledge handshake, mode 2. Using **ESC**.*P2:* to initiate the handshake is very simple, because the instruction sets several parameter values for you, as shown below.

```
ESC.P2:
```

This instruction automatically specifies the parameter values as:

Block size	= 20 bytes without cartridge 80 bytes with cartridge
Enquiry character	= 5 (ASCII **ENQ**)
Acknowledgment string	= 6 (ASCII **ACK**)
Output terminator	= 13 (ASCII **CR**)
Echo terminate character	= 10 (ASCII **LF**)
Output trigger character	= 17 (ASCII **DC1**)

Be aware that these parameter values may not be the best for your system. If your computer requires other values, note the following:

ESC.I will change the block size, enquiry character, and acknowledgment string.
ESC.M will change the output terminate, echo terminate, and output trigger character. You can also add a turnaround delay or output initiator.

Override parameter values set by **ESC**.*P2:* by placing the appropriate instruction *after* ESC.P.

Example — Using ESC.H to Initiate Enquire/Acknowledge Handshake (Mode 1)

Mode 1 is a form of a software checking handshake, usually used when the handshake is part of your program. If your system requires ESC.M and ESC.N parameters in addition to a turnaround delay, intercharacter delay, and immediate response string, use ESC.H to initiate an enquire/acknowledge handshake. The following example shows the ESC.M and ESC.N instructions used with ESC.H.

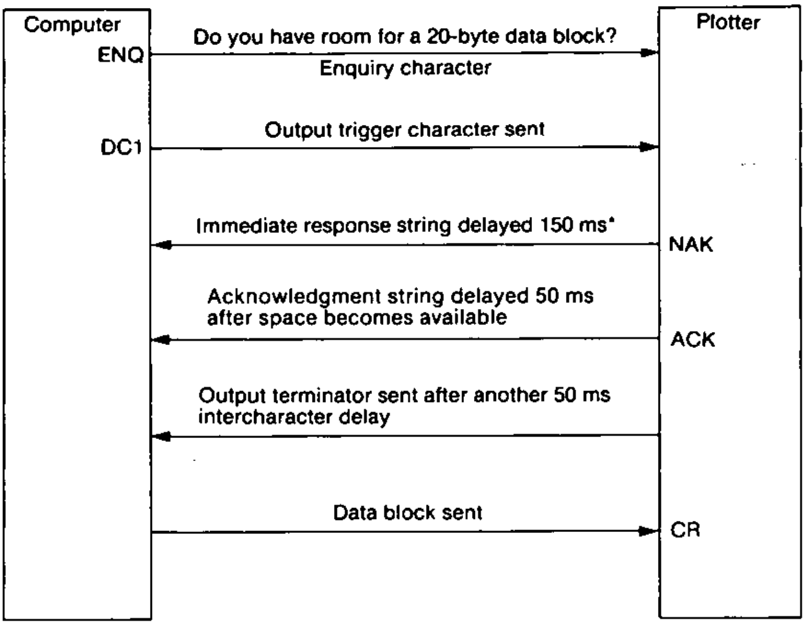
```
ESC.M100;17;0;13:
ESC.N50;21:
ESC.H20;5;6:
```

These instructions specify the parameter values as:

Turnaround delay	= 100
Output trigger character	= 17 (ASCII **DC1**)
Echo terminate character	= none
Output terminate character	= 13 (ASCII **CR**)
Intercharacter delay	= 50 milliseconds

Immediate response string	= 21 (ASCII **NAK**)
Data block size	= 20 bytes
Enquiry character	= 5 (ASCII **ENQ**)
Acknowledgment string	= 6 (ASCII **ACK**)

The following diagram illustrates the data exchange this handshake would implement.


Computer
Plotter
ENQ
Do you have room for a 20-byte data block?
Enquiry character
DC1
Output trigger character sent
Immediate response string delayed 150 ms*
NAK
Acknowledgment string delayed 50 ms after space becomes available
ACK
Output terminator sent after another 50 ms intercharacter delay
Data block sent
CR

*Total of 100 ms turnaround delay and 50 ms intercharacter delay.

Example — Using ESC . I to Initiate Enquire/Acknowledge Handshake (Mode 2)

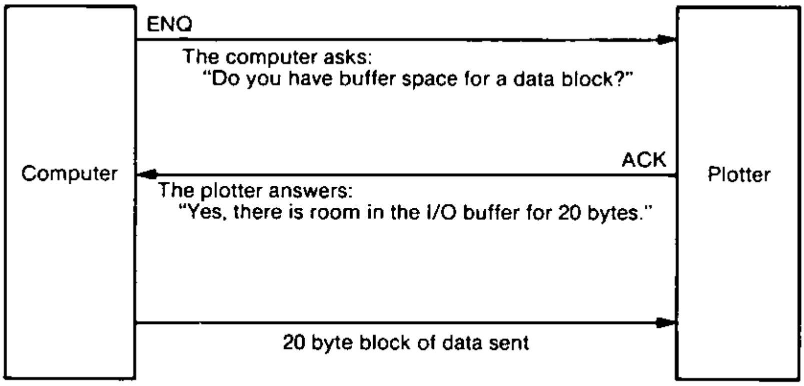
If your computer requires only the turnaround delay be included with the handshake characters, you can initiate the simplest form of an enquire/acknowledge handshake (mode 2) with ESC . I. This usually is true when the handshake is part of the operating system, and not part of the program.

```
ESC.I20;5;6:
```

This instruction specifies the parameter values as:

Block size	= 20 bytes
Enquiry character	= 5 (ASCII **ENQ**)
Acknowledgment string	= 6 (ASCII **ACK**)

The following diagram illustrates the data exchange this handshake would implement.


ENQ
The computer asks:
"Do you have buffer space for a data block?"
ACK
Computer
Plotter
The plotter answers:
"Yes, there is room in the I/O buffer for 20 bytes."
20 byte block of data sent

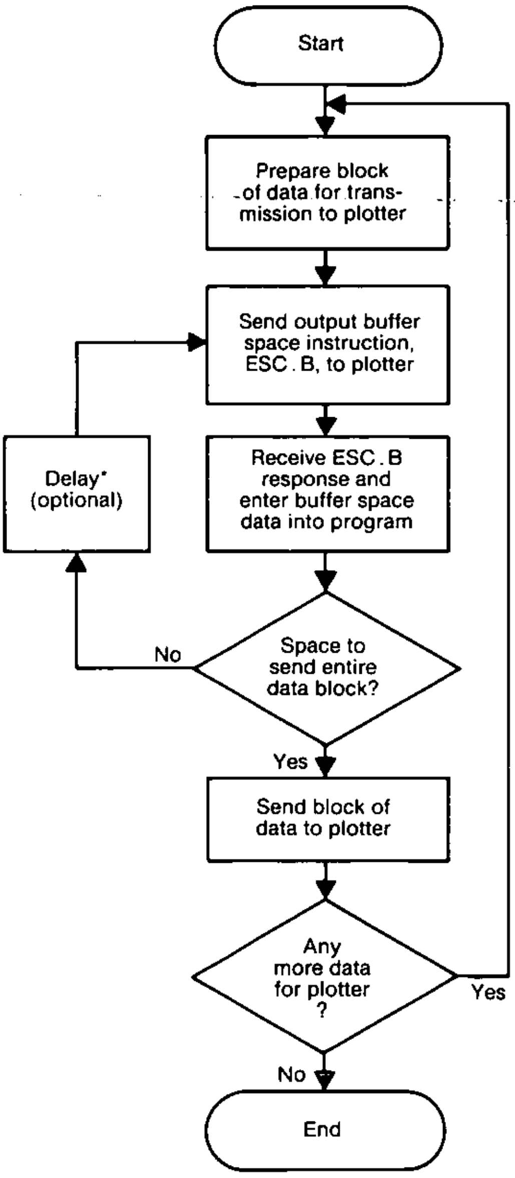
Software Checking Handshake

When your computer system cannot use any of the previously described handshakes, you must add lines to your program to ensure that data transferred between the computer and plotter is not lost. In the previous section, we talked about one type of software handshake using the enquire/acknowledge principle (mode 1: ESC . H). This is the preferred and more efficient of the two methods of software handshake. The second method of software handshake uses the output buffer space instruction, ESC . B, as described below.

Initiating a Software Checking Handshake Using Device-Control Instructions

The following flowchart illustrates how a typical software checking handshake works within a program. Use the output buffer space instruction, ESC . B, in your program to monitor the plotter's buffer space. This instruction is explained in detail, at the end of this chapter.

The ESC . B instruction outputs the number of empty bytes in the plotter's I/O buffer. When the plotter has enough available bytes to accommodate the data, transmit the data block. Repeat this process until all data has been sent. This method can use large amounts of computer and plotter I/O processing time.


Start
Prepare block of data for transmission to plotter
Send output buffer space instruction, ESC . B, to plotter
Receive ESC . B response and enter buffer space data into program
Delay* (optional)
Space to send entire data block?
No
Yes
Send block of data to plotter
Any more data for plotter ?
Yes
No
End

*Insert computer instructions that will temporarily halt the program, such as a WAIT statement or a FOR . . . NEXT loop.

Software Checking Handshake

You can use one or both of the following techniques to reduce the inquiries about the availability of I/O buffer space.

1. Count the number of bytes sent to the plotter. Delay the initial inquiry about available I/O buffer space until enough data has been sent to fill the buffer once.
2. After a negative reply, wait a short time before again asking how much I/O buffer space is available.

Determining Parameter Requirements for a Software Checking Handshake

To match the requirements of your computer system, the following parameters may be specified for a software checking handshake. Check your computer's documentation to determine which of the following parameters are required.

Parameter	Device-Control Instruction Used
Turnaround delay	ESC.M
Output trigger character	ESC.M
Echo terminate character	ESC.M
Output terminate character	ESC.M
Output initiator	ESC.M
Intercharacter delay	ESC.N

Example — Using ESC.B to Initiate a Software Checking Handshake

In addition to using ESC.B, this example uses the ESC.M and ESC.N instructions to establish a turnaround delay and an intercharacter delay. All other parameters assume their default values because they are omitted. Place the ESC.B instruction *after* ESC.M and ESC.N.

```
ESC.M250:
ESC.N50:
ESC.B
```

These instructions specify the parameter values as:

Turnaround delay	= 250 milliseconds
Intercharacter delay	= 50 milliseconds

Device-Control Instructions Used in RS-232-C Interfacing and Handshaking

You may need to modify the format that the plotter uses for outputting data in order to match the requirements of your system, depending on your application. The following section explains each of the device-control instructions used to format output and control data transfer.

Review Chapter 12 for device-control instruction syntax.

Set Handshake Mode Instruction, ESC . P

USES: The ESC . P instruction simplifies selecting a handshake method. Use this instruction to select one of three standard handshakes.

SYNTAX: ESC . P (handshake method) :

DEFAULT: ESC . P : sets hardwire handshake (parameter 3)

EXPLANATION: The ESC . P instruction selects hardwire, Xon-Xoff, or enquire/acknowlege handshake. (Use ESC . @, H, I, M, and/or N to select a handshake with parameter values other than those listed on the next page.)

The integers 0 through 3 are used to specify a handshake method with predefined parameters as follows:

Parameter Value and Handshake Method				
0 None	**1 Xon-Xoff**	**2 Enq/Ack**	**3 Hardwire (default)**	**Predefined Handshake Parameters Established**
0	50	0	0	Turnaround Delay
0	0	17	0	Output Trigger Character
0	10	10	0	Echo Terminate Character
13	13	13	13	Output Terminator
0	2	2	0	Handshake Mode
20*	20*	20*	20*	Block (Record) Size
60**	60**	60**	60**	Logical I/O Buffer Size
0	10	0	0	Intercharacter Delay
0	0	0	0	Immediate Response String
0	17	0	0	Xon Trigger Character
0	19	0	0	Xoff Trigger Character
0	0	5	0	Enquiry Character
0	0	6	0	Acknowledgment String
20*	20*	20*	20*	Threshold Level

*80 with cartridge

**1024 with cartridge (if physical I/O buffer has not been reduced — refer to the *Allocate Configurable Memory Instruction, ESC . T,* in Chapter 12).

Set Output Mode Instruction, ESC . M

USES: The ESC . M instruction establishes parameters for the plotter's communication format. Use this instruction to establish a turnaround delay, output trigger character, echo terminate character, and an output initiator character. Also use it to change the output terminator from its default value, carriage return.

SYNTAX: ESC . M (turnaround delay) ; (output trigger character) ; (echo terminate character) ; (output terminator) ; (output initiator character) :

Parameter	Format	Range	Default
turnaround delay	real	0 to 255 without cartridge	0
		0 to 65 535 with cartridge	0
output trigger character	ASCII	0 to 126*	0
echo terminate character	ASCII	0 to 126*	0
output terminator(s)	ASCII	0 to 127**	13 (**CR**)
output initiator character	ASCII	0 to 127	0

*The values 5 and 27 are not allowed.
**The value 0 terminates the string.

EXPLANATION: A description of the instruction's parameters follows. Remember to use the decimal code of the desired ASCII character(s) in parameters two through five.

1. **Turnaround Delay.** The turnaround delay postpones the plotter's transmission of data until the computer is ready to receive and process it. To specify a turnaround delay, use a number within the range shown in the preceding table. The actual number of milliseconds the plotter will use as a turnaround delay is 1.1 × the parameter specified.

   If you set an intercharacter delay using ESC . N, that value is added to the value of the turnaround delay.

2. **Output Trigger Character.** When an output trigger character is used, it is the last character output by the computer when making a request of the plotter. This character tells the plotter, "It is O.K. to respond to my request now." The **DC1** character (decimal code 19) is often used for the output trigger character.

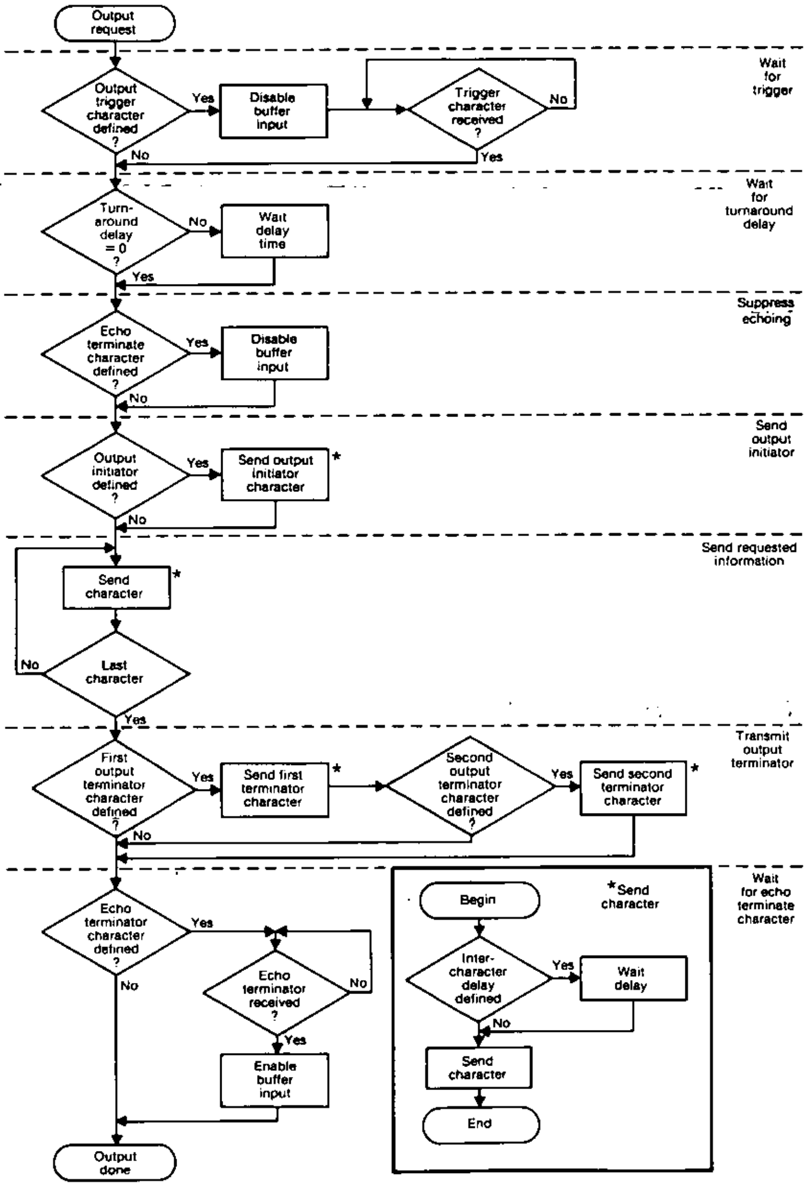
3. **Echo Terminate Character.** The echo terminate character tells the plotter that the computer will echo all responses and that this echoed data should be ignored (the plotter's I/O buffer should be closed) until an echo terminate character is received. To determine if you need an echo terminate character, check your system documentation to see if your computer echoes data. If your computer does not echo the plotter's response, specify this parameter as zero or omit it.

When the plotter receives the echo terminate character, it re-opens its I/O buffer to accept data from the computer. If the computer echoes the plotter's response without appending a character, then the plotter's output terminator (described in the next paragraph) should be used as the echo terminate character. The **LF** character (decimal code 10) is often used for the echo terminate character.

4. **Output Terminator.** The output terminator is a one- or two-character terminator that the computer requires the plotter to send at the end of each response to a data request. The output terminator tells the computer, "This completes my transmission." The character **CR** (decimal code 13) is the default output terminate character.

5. **Output Initiator.** The output initiator character tells the computer, "This starts my transmission." It is a one-character initiator that is sent by the plotter at the beginning of any output response. In order to specify an output initiator, the output terminator must consist of two characters, or the second character must be set to 0 or omitted (by entering only a semicolon). The character **STX** (decimal code 2) is generally used for the output initiator.

The flowchart on the next page depicts a plotter output request.


Output request
Output trigger character defined ?
Yes
Disable buffer input
Trigger character received ?
No
Yes
No
Wait for trigger
Turn-around delay = 0 ?
No
Wait delay time
Yes
Wait for turnaround delay
Echo terminate character defined ?
Yes
Disable buffer input
No
Suppress echoing
Output initiator defined ?
Yes
Send output initiator character *
No
Send output initiator
Send character *
Last character
No
Yes
Send requested information
First output terminator character defined ?
Yes
Send first terminator character *
Second output terminator character defined ?
Yes
Send second terminator character *
No
Transmit output terminator
Echo terminator character defined ?
Yes
Echo terminator received ?
No
Yes
Enable buffer input
No
Output done
Wait for echo terminate character
*Send character
Begin
Inter-character delay defined
Yes
Wait delay
No
Send character
End

Output Request Flowchart

Set Extended Output and Handshake Mode Instruction, ESC . N

USES: The ESC . N instruction establishes parameters for the plotter's communication format. Use this instruction to specify an intercharacter delay, an immediate response string for the enquire/acknowledge handshake, or the Xoff trigger character(s) for the Xon-Xoff handshake.

SYNTAX: Xon-Xoff: **ESC** . N (intercharacter delay) ; (Xoff trigger character(s)) :

Enquire/acknowledge: **ESC** . N (intercharacter delay) ; (immediate response string) :

Parameter	Format	Range	Default
intercharacter delay	real	0 to 255 without cartridge 0 to 65 535 with cartridge	0
Xoff trigger character(s) (Xon-Xoff) or	ASCII	0 to127	0 (no character)
immediate response string (Enquire/acknowledge)	ASCII	0 to 127	0 (no response)

EXPLANATION: A description of the instruction's parameters follows.

1. **Intercharacter Delay.** The 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 ESC . M) before the first character is sent by the plotter, and is also inserted before each subsequent character in a string being sent to the computer.

   To specify an intercharacter delay, use a number within the range shown in the preceding table. The actual number of milliseconds the plotter will use as an intercharacter delay is 1.1 $\times$ the parameter specified.

2. **Xoff Trigger Character.** In an Xon-Xoff handshake, the plotter sends the Xoff trigger character to signal the computer to temporarily stop sending data while the plotter processes what it has already received. The **DC1** character (decimal code 19) is generally used for the Xoff trigger character.

   or

   Immediate Response String. In an enquire/acknowledge handshake, some systems require an immediate response from the plotter, acknowledging the enquiry from the computer. Systems of this type include computers that transmit data to the plotter after a certain time interval, but before receiving a go-ahead signal from the plotter. The plotter transmits the immediate response string immediately after receipt of an enquiry character. This tells the computer, "Wait, I am checking my buffer space." The **NAK** character (decimal code 21) is often used for this character.

   NOTE: If you do *not* have a cartridge installed, you can only use one character for the Xoff trigger character or immediate response string. With a cartridge installed, you can use up to 10 characters, separated by semicolons. ■

Set Handshake Mode 1 Instruction, ESC . H

USES: The ESC . H instruction is useful for setting up a handshake when the handshaking is part of the application program. Use this instruction to configure the plotter for enquire/acknowledge handshake when the computer requires that all the plotter's output (including the handshake response) is sent in accordance with the parameters set in the ESC . M and ESC . N instructions.

SYNTAX: **ESC** . H (data block size) ; (enquiry character) ; (acknowledgment string) :

Parameter	Format	Range	Default
data block size	real	0 to 60 bytes–without cartridge	20 bytes
		0 to 1974 bytes with cartridge	80 bytes
enquiry character	ASCII	0 to 126*	0 (no character)
acknowledgment string	ASCII	0 to 127	0 (no character)

*Excluding 27.

EXPLANATION: A description of the parameters is given next.

1. **Data Block Size.** Data block size specifies the maximum size of each block of data that the plotter will receive from the computer.

2. **Enquiry Character.** In an enquire/acknowledge handshake, the computer sends an enquiry charcter which asks the plotter if it has room for a block of data. The character **ENQ** (decimal code 5) is typically used as the enquiry character.

3. **Acknowledgment String.** In an enquire/acknowledge handshake, the plotter responds to the enquiry character with the acknowledgment string when there is sufficient room in its I/O buffer for a block of data. The character **ACK** (decimal code 6) is typically used. Avoid using characters which have special meaning to the computer.

   NOTE: If you do *not* have a cartridge installed, you can only use one character for the acknowledgement string. With a cartridge installed, you can use up to 10 characters, separated by semicolons. ■

Set Handshake Mode 2 Instruction, ESC . I

USES: The ESC . I instruction is used when the computer does not expect handshake responses to be sent using the output parameters defined by ESC . M and ESC . N. This is often true when the handshake protocol is part of the operating system. It is used with the enquire/acknowledge or Xon-Xoff handshake to establish handshake protocol.

SYNTAX: Xon-Xoff: **ESC** . I (Xoff threshold level) ; (omitted parameter) ; (Xon trigger character(s)) :

Enquire/acknowledge: **ESC** . I (data block size) ; (enquiry character) ; (acknowledgment string) :

Parameter	Format	Range	Default
Xoff threshold level (Xon-Xoff)	real	0 to 60 bytes without cartridge	20 bytes without cartridge
or		0 to 1974 with cartridge	80 bytes with cartridge
data block size (Enquire/acknowledge)	ASCII	0 to 80 bytes without cartridge	20 bytes
		0 to 1974 bytes with cartridge	80 bytes
enquiry character	ASCII	0 to 126*	0 (no character)
Xon trigger character(s) (Xon-Xoff)	ASCII	0 to 127	0 (no character)
or acknowledgment string (Enquire/acknowledge)	ASCII	0 to 127	0 (no character)

*Except 27.

1. **Xoff Threshold Level.** In an Xon-Xoff handshake, subtract the value you specify in this parameter from the size of the I/O buffer. This is the point at which the plotter will send the Xoff trigger character to the computer, telling it to stop sending data. With most computers, 10 is a suitable value. For maximum efficiency, specify the smallest possible value that will ensure that the I/O buffer will not overflow.

or

Data Block Size. Data block size specifies the maximum size of each block of data that the plotter will receive from the computer.

2. **Enquiry Character.** In an enquire/acknowledge handshake, the computer sends an enquiry character which asks the plotter if it has room in its I/O buffer for a block of data. The character ENQ (decimal code 5) is typically used as the enquiry character*.

   If you are using an Xon-Xoff handshake, omit the parameter or specify the NULL character (decimal code 0).

3. **Xon Trigger Character(s).** In an Xon-Xoff handshake, the plotter sends the Xon trigger character to signal the computer to resume sending data when there is sufficient room in the I/O buffer. The DC1 character (decimal code 17) is generally used for the Xon trigger character.

   or

   Acknowledgment String. In an enquire/acknowledge handshake, the plotter responds to the enquiry character with the acknowledgment string when there is sufficient room in its I/O buffer for data. The character ACK (decimal code 6) is often used for the acknowledgment string.

   NOTE: If you do *not* have a cartridge installed, you can only use one character for the Xon trigger character or acknowledgment string. With a cartridge installed, you can use up to 10 characters, separated by semicolons. ■

*If your computer is configured to send an ENQ when it is ready to send data to the plotter, the plotter will automatically respond with ACK when it receives ENQ, unless the ENQ has been defined as the enquiry character. This "dummy handshake" does not prevent I/O buffer overflow.

Output Buffer Space Instruction, ESC.B

USES: The ESC.B instruction outputs the space currently in the I/O buffer available for data. You can use this instruction in a software checking handshake to interrogate the plotter regarding available I/O buffer space. If you have a cartridge installed, you can use this instruction to confirm the size of the I/O buffer when you have reallocated buffer space.

SYNTAX: ESC.B

RESPONSE: available buffer space [TERM]

EXPLANATION: The ESC.B instruction outputs the number of unused bytes remaining in the I/O buffer. When you turn the plotter on the output response is 60 (1024 with cartridge). The response can be as low as 0 (if either the ESC.T or ESC.@ instruction has reduced the I/O buffer), or as large as 1974 (if both the ESC.T and ESC.@ instructions have increased the logical I/O buffer and the cartridge is installed).

Output Extended Error Instruction, ESC.E

USES: The ESC.E instruction outputs a number that defines any I/O-related error. Use this instruction in program debugging to identify the errors that occur. If you have a cartridge installed, you can use this instruction to perform block I/O error checking. For an example of ESC.E used in I/O error checking, refer to *Block Mode*, later in this chapter.

SYNTAX: ESC.E

RESPONSE: error number [TERM]

EXPLANATION: The ESC.E instruction outputs the number of the I/O error. The I/O error is output as an integer in the range of 0 or 10–16, as described in the following table.

Error No.	Meaning
0	No I/O error has occurred.
10	New output has been generated before previous output was finished being transmitted. In an RS-232-C configuration, the previous output will continue normally and the new output will be ignored (thus causing the error).
11	Invalid character received after first two characters (**ESC**.) in a device-control instruction.
12	Invalid character received while parsing a device-control instruction. The parameter containing the invalid character 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 termination) or the next **ESC** character (abnormal termination) is received.
15	A framing error or parity error has been detected.
16	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.

Reset Instruction, ESC . R

USES: The ESC . R instruction resets certain I/O conditions to the states that exist when the plotter is first turned on. Use this instruction in an initialization sequence when starting a new program.

SYNTAX: **ESC** . R

EXPLANATION: The ESC . R instruction aborts any currently executing device-control instruction and any partially parsed HP-GL instruction. This instruction also resets the parser and clears the I/O buffer (and polygon buffer if a cartridge is installed).

In addition, ESC . R defaults the size allocation of the I/O buffer (and polygon buffer if a cartridge is installed).

Executing ESC . R is equivalent to executing this four-instruction sequence:

ESC . J ESC . K ESC . T : ESC . P :

For more detailed information on what each of these instructions does, refer to their individual descriptions.

NOTE: After executing the ESC . R instrucion, execute an ESC . L. This ensures that all conditions have been reset before any subsequent instruction can be parsed. In this application of the ESC . L instruction, the numerical value of the buffer size is not important. However, your program should read the output response before sending more data, in order to avoid potential errors. Refer to the hints under the ESC . T instruction, earlier in this chapter, for more information. ■

Data Transmission Modes

With a cartridge installed, the RS-232-C version of the plotter has two modes of data transmission: normal mode and block mode. Without a cartridge, you can only use normal mode. Both modes are explained below.

Normal Mode

The plotter is in normal mode when you turn it on. **If you are using a software package, leave the plotter in normal mode unless instructed to change it.** In normal mode, all HP-GL instructions are put in a buffer where they are parsed and executed in order. Device-control instructions (ESC . E, ESC . B, etc.) are not buffered but are executed immediately.

Normal mode is the default mode, but you can also initiate it using one of the parameters of the ESC . @ instruction. Refer to the *Set Plotter Configuration Instruction, ESC . @*, in Chapter 12, for details.

Block Mode

If you have installed a cartridge in your plotter, you can also use block mode. Block mode is used to detect transmission errors. Consider using block mode only if you are using a modem or are in a situation where there is likely to be noise on the data transmission lines. Initiate block mode using the ESC.@ device-control instruction in a program.

If an error is detected in block mode, the block of data will not be executed. This allows you to retransmit the data, preventing garbled data from causing plotting errors.

When you turn on block mode, the buffer is cleared. Subsequently, all characters *except* device-control instructions* and handshake characters are checked for errors. If there are no errors, the data is processed by the plotter.

Block mode has no effect on the type of handshake used or on the handshake parameters. If the number of characters in the buffer exceeds the logical buffer size, error 16 will occur (buffer overflow). Your handshake should prevent this from occurring.

Using the ESC.@ and ESC.E Instructions for Block I/O Error Checking

The diagram on the next page illustrates block I/O error checking; an explanation follows.

*Device-control instructions are executed immediately.

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 or framing error. At this point, the plotter discards the block because an error occurred.
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.

Enable block mode by setting bit 4 of the second parameter of the ESC.@ instruction to logic state "1" (decimal value 16), as explained under The *Set Plotter Configuration Instruction, ESC.@*, in Chapter 12. Then begin sending data blocks, following each with the ESC.E instruction. For each data block, if the response to ESC.E is 0 (indicating no error), the data block is executed normally and a new data block can be sent. If the response to ESC.E is within the range of 10 through 16, an error has occurred and the plotter will discard the entire data block. Then you can retransmit the data block. Be sure to send an ESC.E instruction to clear out the buffer before disabling block mode.

NOTE: Using the ESC.L instruction when in block mode may disrupt communication. If you use an ESC.L instruction while in block mode, use 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. ■

Automatic Modem Disconnection Modes

Two modes are available for automatic disconnection at the end of an RS-232-C/CCITT V.24 session conducted over phone lines. The two modes are switched/datex-line disconnection and leased-line disconnection. When active, disconnection modes disable hardwire handshake. To leave either mode, turn the plotter off and on again.

Switched/Datex-Line Disconnection Mode

In this mode, the CTS and DSR (CB and CC) lines (described in Appendix A) control the DTR 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 left cursor control key (left arrow) while turning on the plotter. The plotter remains in this mode until turned off.

Leased-Line Disconnection Mode

In this mode, the CTS, DSR, and DCD (CB, CC, and CF) lines (described in Appendix A) 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 right cursor control key (right arrow) while turning on the plotter. The plotter remains in this mode until turned off.

CHAPTER 14

CHAPTER 14

HP-IB (IEEE-488) Interfacing

What You'll Learn in This Chapter

This chapter is for owners of HP ColorPro plotters with the Hewlett-Packard Interface (HP-IB). In this chapter you will learn:

- how to use the HP-IB addressing technique
- which HP-IB functions are implemented on the plotter
- how to use serial and parallel polling

Appendix A contains program examples for a variety of computers. The program line that addresses the plotter is highlighted. Use this line in all of your application programs. *In most cases, this will be all the information you need to use HP-IB interfacing.* These program examples also show you how to send and receive data.

Read the rest of this chapter if your computer is not listed in Appendix A, or if you need to understand the HP-IB addressing technique. This chapter explains the HP-IB functions, and their implementation on the plotter.

What Is the HP-IB?

The Hewlett-Packard Interface Bus (HP-IB) provides for mechanical, electrical, timing, and data compatibility between all devices adhering to the ANSI/IEEE-488 standard. For most applications, you will probably only need to understand how to use the addressing concept, described next in this chapter. However, if you want to know more about how the HP-IB works, refer to the ANSI/IEEE 488-1978 standard. Another helpful document is Hewlett-Packard's *Tutorial Description of the HP-IB* (Part No. 5952-0156).

Addressing the Plotter

The HP-IB uses an addressing technique to ensure that each device on the bus (interconnected by HP-IB cables) receives only the data intended for it. Using this addressing technique, devices can be instructed to talk (send) or listen (receive). More than one device can listen at the same time, but only one device can be designated as a talker at any given time.

There are basically two modes of addressing the plotter: addressable and listen-only mode. In addressable mode, the plotter can function as a talker or as a listener, depending on the commands it receives from the computer. In listen-only mode, the plotter hears all activity on the bus, but it cannot talk.

Addressable Mode

The addressing technique on HP desktop computers requires assigning a "select code" to the HP-IB interface, and an "address code" to the plotter. Appendix A lists program examples for several personal computers. The highlighted program line is the line that addresses the plotter. *All of your programs must include this addressing information.*

The plotter is set to an address of 05 at the factory. Using the rear-panel switches, you can set your plotter to any one of 31 different HP-IB addresses, ranging from 0 through 30, or listen-only mode (described in the following section). Each HP-IB interface can have as many as 15 devices connected to it, set to different specific address codes.

HP-IB Address Settings

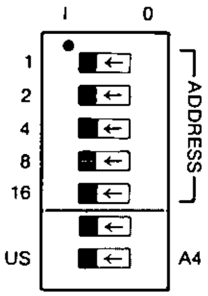
Address Switch Settings					Address Codes		Address Characters		
16	**8**	**4**	**2**	**1**	**Decimal**	**Octal**	**Listen**	**Talk**	
0	0	0	0	0	0	0	SP	@	
0	0	0	0	1	1	1	!	A	
0	0	0	1	0	2	2	“	B	
0	0	0	1	1	3	3	#	C	
0	0	1	0	0	4	4	$	D	
0	0	1	0	1	5	5	%	E	Preset
0	0	1	1	0	6	6	&	F	
0	0	1	1	1	7	7	’	G	
0	1	0	0	0	8	10	(	H	
0	1	0	0	1	9	11	)	I	
0	1	0	1	0	10	12	*	J	
0	1	0	1	1	11	13	+	K	
0	1	1	0	0	12	14	,	L	
0	1	1	0	1	13	15	—	M	
0	1	1	1	0	14	16	.	N	
0	1	1	1	1	15	17	/	O	
1	0	0	0	0	16	20	0	P	
1	0	0	0	1	17	21	1	Q	
1	0	0	1	0	18	22	2	R	
1	0	0	1	1	19	23	3	S	
1	0	1	0	0	20	24	4	T	
1	0	1	0	1	21	25	5	U	Reserved for HP desktop computer address
1	0	1	1	0	22	26	6	V	
1	0	1	1	1	23	27	7	W	
1	1	0	0	0	24	30	8	X	
1	1	0	0	1	25	31	9	Y	
1	1	0	1	0	26	32	:	Z	
1	1	0	1	1	27	33	;	[	
1	1	1	0	0	28	34	<	\	
1	1	1	0	1	29	35	=	]	
1	1	1	1	0	30	36	>	<	
1	1	1	1	1	31	37	?	-	Sets listen-only mode

NOTE: When using your plotter with an HP desktop computer, do not use address 21; it is reserved for the desktop computer's address. ■

HP-IB Interfacing

Listen-Only Mode

To activate listen-only mode set all the rear-panel address switches to 1, as shown in the following illustration. In listen-only mode, the plotter does not have an address, but listens to all data transmitted on the bus. The plotter cannot then be placed in a talker-active state.



Listen-only mode is useful in a system that has no controller but has a dedicated talker (such as a tape drive or other mass storage unit) transmitting information to the plotter.

When several plotters are connected to the HP-IB, you can plot on all plotters simultaneously by setting all but one to listen-only mode. Use the one plotter that isn't in listen-only mode to do the talking for all of the plotters.

Notes on Addressing Protocol

Some computer systems can use high-level languages (such as BASIC, FORTRAN, Pascal, and COBOL) with high-level input/output (I/O) statements. In this case, the addressing procedure (unlisten, talk, listen) is taken care of by the computer's internal operating system and need not be of concern to you. With these high-level I/O statements, you may not be able to control some of the other bus functions. If your system fits this description, you do not need to continue reading this chapter.

Some computers must use low-level I/O statements to address devices on the HP-IB bus. If your computer uses such statements, you'll need to direct the talking, listening, and unlistening activities as described in the following section. In order to communicate effectively with the plotter, it is important that you understand the HP-IB addressing protocol of your computer. Therefore, you may wish to review this aspect of your computer.

Controlling Addressing Sequences

One of the first things you must consider when *directly* controlling the HP-IB is addressing. Following is a typical addressing sequence.

<Unlisten Command> <Talk Address> <Listen Addresses>

This sequence is made up of three major parts which serve the following purposes:

1. The unlisten command is the universal bus command with a character code of **?**. It unaddresses all listeners. After the unlisten command is transmitted, no active listeners remain on the bus.
2. The talk address designates the device that is to talk. A new talk address automatically unaddresses the previous talker.
3. The listen addresses designate one or more devices that are to listen. A listen address adds the designated device as listener along with other addressed listeners.

This addressing sequence simply directs who is to talk to whom. The commands (unlisten, talk, listen) are implemented 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.

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 over the data lines:

?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.

Refer to the preceding section called *Addressable Mode* for a table of ASCII characters and their decimal and octal equivalent values.

Interface Functions

Interface functions provide the physical capability to communicate via the HP-IB. There are 10 interface functions, plus two special cases of the controller function. All HP-IB devices do not implement all functions. The following table lists the functions implemented on the HP ColorPro. A dash (—) indicates that the plotter does *not* implement that function.

HP-IB Interface Functions

Mnemonic	**Interface Function Name**	**Plotter Implementation**
SH	Source Handshake	SH1
AH	Acceptor Handshake	AH1
T	Talker (or TE = Extended Talker)*	T6
L	Listener (or LE = Extended Listener)*	L3
SR	Service Request	SR1
RL	Remote Local	—
PP	Parallel Poll	PP1, PP2 or PP0**
DC	Device Clear	DC1
DT	Device Trigger	—
C	Any Controller	—
C_N	A Specific Controller (for example: C_A, C_B . . .)	—
C_S	The System Controller	—

*Extended Talkers and Listeners use a two-byte address. Otherwise, they are the same as Talker and Listener.

**PP1 (if address $\geqslant$ 8), PP2 (if address $<$ 8), PP0 (if listen-only mode).

Serial and Parallel Polling

Serial and parallel polling are the processes used by the computer to determine the state of devices (such as plotters and printers) on the HP-IB bus. Polling can be very useful in digitizing applications.

Serial Polling

Refer to your computer's documentation to determine whether or not your system has serial poll capability, and for the necessary (enable/disable) commands. In a serial poll, the computer polls the devices on the bus one at a time, in sequence. 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.

The condition(s) you define with the S-mask parameter of the IM instruction determine which condition(s) will trigger a service request. When a condition is met (e.g., digitized point available), the plotter sends a service request to the computer; this sets bit 6 of the status byte to a logical "1".

Use an interrupt routine to halt the program when a service request is received. Then, send a serial poll to determine which device on the line needs 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 condition (defined by the S-mask of the IM instruction) occurs again.

Parallel Polling

Refer to your computer's documentation to determine if your system has parallel poll capability and for the necessary (enable/disable) commands. If your system can implement parallel polling, this may be the fastest method of determining which device on the bus needs service. In one operation, the parallel poll determines which device (with parallel poll capability) needs service. The plotter cannot respond to a parallel poll in listen-only mode.

Use the P-mask parameter of the IM instruction to enable the plotter's parallel poll response, and to specify which condition will trigger a positive response. After receiving a positive response, use the OS instruction to obtain the status of the plotter.

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. This is called remote configuration.

Plotter 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	← Preset
6	1	2	2	
7	0	1	1	

APPENDIX A

Interfacing Supplementary

Plotter/Computer Interconnection Programs

This section contains simple test programs you can use to verify communication between your plotter and computer. These programs also appear in Chapter 3 of your plotter's Operating Manual. In this section, we have highlighted the program lines that establish interfacing and/or handshake conditions and, in some cases, computer-specific instructions that channel information to the plotter. When you write programs, use these lines *in the order shown* to ensure communication between your plotter and computer.

To use this section, find the program appropriate to your computer and interface. Computers are listed in alphabetic order. If you have followed the step-by-step instructions in the Operating Manual, entered the program correctly, and the interface is working, the plotter will print "7440A COMMUNICATION OK" when you run the program.

NOTE: In order to read data from the plotter, your computer system must be able to accept input. The input in this program example is the plotter identification number "7440A". If your computer is unable to read input from the plotter using BASIC, the words *Output Only* will appear in the heading and the plotter will print "COMMUNICATION OK" when you run the program. ■

Apple IIc Computer
(RS-232-C Interface)

```
10 PRINT CHR$(4); "PR#2"
20 PRINT CHR$(1); "10B"
30 PRINT CHR$(4); "IN#2"
40 PRINT CHR$(27) + ".M50;63;13;13:"
50 PRINT "IN;OI;"
60 INPUT ID$
70 PRINT "SP1;PA500,500;"
80 PRINT "LB"+ID$+" COMMUNCIATION OK" + CHR$(3)
90 PRINT "PA0,0;SP0;"
100 PRINT CHR$(4); "PR#0"
110 PRINT CHR$(4); "IN#0"
120 END
```

Program Explanation

10	directs computer output to the plotter at port 2
20	reassigns computer baud rate to 2400
30	opens computer input port
40	sets a turnaround delay of 50 ms, output trigger character of 63 (ASCII **?**), and echo terminate character and output terminator of 13 (ASCII **CR**)
50	initializes plotter; directs plotter to output its identification
60	computer reads identification information
70	selects pen 1; moves to the starting label position
80	labels plotter identification and "COMMUNICATION OK"
90	moves paper for viewing of plotted output; returns the pen to the carousel
100	closes computer output port
110	closes computer input port

Apple IIe or II Plus Computer
(RS-232-C Interface)

```
10 PR#2 : IN#2
20 PRINT CHR$(27) + ".M50;63;13;13:"
30 PRINT "IN;OI;"
40 INPUT ID$
50 PRINT "SP1;PA500,500;"
60 PRINT "LB";ID$;" COMMUNICATION OK";CHR$(3)
70 PRINT "PA0,0;SP0;"
80 PR#0 : IN#0
90 END
```

Program Explanation

10 directs computer output to the plotter at port 2; prepares computer to accept input from port 2

20 sets a turnaround delay of 50 ms, output trigger character of 63 (ASCII ?), and an echo terminator character and output terminator of 13 (ASCII **CR**)

30 initializes plotter; directs plotter to output its identification

40 computer reads identification information

50 plotter selects pen 1; moves to the starting label position

60 labels plotter identification and the message "COMMUNICATION OK"

70 moves paper for viewing of plotted output; returns the pen to the carousel

80 returns output to computer terminal; prepares computer to accept input from the keyboard

Apple /// Computer
(RS-232-C Interface)

```
10 OPEN #1, ".RS232"
20 PRINT #1, "IN;OI;"
30 INPUT #1, ID$
40 PRINT #1, "SP1;PA500,500;"
50 PRINT #1, "LB"+ID$+" COMMUNICATION OK"+CHR$(3)
60 PRINT #1, "PA0,0;SP0;"
70 CLOSE #1
80 END
```

Program Explanation

10 directs computer output to the plotter at port 1; establishes RS-232-C interface

20 initializes plotter; directs plotter to output its identification

30 computer reads identification information

40 selects pen 1; moves to the starting label position

50 labels plotter identification and "COMMUNICATION OK"

60 moves paper for viewing of plotted output; returns the pen to the carousel

70 closes computer port 1

AT&T Personal Computer 6300
(RS-232-C Interface)

```
10 OPEN "COM1:9600,N,8,1,RS,CS65535,DS,CD" AS #1
20 PRINT #1, "IN;OI;"
30 INPUT #1, ID$
40 PRINT #1, "SP1;PA500,500;"
50 PRINT #1, "LB"+ID$+" COMMUNICATION OK"+CHR$(3)
60 PRINT #1, "PA0,0;SP0;"
70 END
```

Program Explanation

10 sets the following RS-232-C interface conditions: 9600 baud rate, parity off, 8 data bits, 1 stop bit, hardwire handshake; directs computer output to the plotter at the COM1: port; "#1" designates file 1 as the file which will hold plotter data

20 initializes plotter; directs plotter to output its identification

30 computer reads identification information

40 selects pen 1; moves to the starting pen position

50 labels plotter identification and "COMMUNICATION OK"

60 moves paper for viewing of plotted output; returns the pen to the carousel

Compaq Personal Computer
(RS-232-C Interface)

```
10 OPEN "COM1:9600,S,7,1,RS,CS65535,DS,CD" AS #1
20 PRINT #1, "IN;OI;"
30 INPUT #1, ID$
40 PRINT #1, "SP1;PA500,500;"
50 PRINT #1, "LB"+ID$+" COMMUNICATION OK"+CHR$(3)
60 PRINT #1, "PA0,0;SP0;"
70 END
```

Program Explanation

10 sets the following RS-232-C interface conditions: 9600 baud rate, parity off, 7 data bits, 1 stop bit, hardwire handshake; directs computer output to the plotter at the COM1: port; "#1" designates file 1 as the file which will hold plotter data

20 initializes plotter; directs plotter to output its identification

30 computer reads identification information

40 selects pen 1; moves to the starting pen position

50 labels plotter identification and "COMMUNICATION OK"

60 moves paper for viewing of plotted output; returns the pen to the carousel

DEC Rainbow 100 Computer (RS-232-C Interface)

Output Only

```
10 LPRINT "IN;"
20 LPRINT CHR$(27)+".I10;0;17:"
30 LPRINT CHR$(27)+".N;19:"
40 LPRINT "SP1;PA500,500;"
50 LPRINT "LBCOMMUNICATION OK"+CHR$(3)
60 LPRINT "PA0,0;SP0;"
70 END
```

Program Explanation

NOTE: The LPRINT statement directs computer output to the plotter at the LinePRINTer port of the computer. ■

10 initializes plotter; directs plotter to output its identification

20 initiates an Xon-Xoff handshake with an Xoff threshold level of 10 bytes and an Xon trigger character of 17 (ASCII **DC1**)

30 sets an Xoff trigger charactger of 19 (ASCII **DC3**)

40 selects pen 1; moves to the starting label position

50 labels "COMMUNICATION OK"

60 moves paper for viewing of plotted output; returns the pen to the carousel

DEC Professional 350 Computer
(RS-232-C Interface)

```
10 OPEN 'LP:' FOR OUTPUT AS FILE #1
20 OPEN 'LP:' FOR INPUT AS FILE #2
30 PRINT #1, CHR$(27)+".M0;0;13;13:"
40 PRINT #1, "IN;OI;"
50 INPUT #2, ID$
60 PRINT #1, "SP1;PA500,500;"
70 PRINT #1, "LB"+ID$+" COMMUNICATION OK"+CHR$(3)
80 PRINT #1, "PA0,0;SP0;"
90 CLOSE #1 \ CLOSE #2
100 END
```

Program Explanation

10 directs computer output to the plotter at port 'LP:' as file #1

20 directs plotter input to the computer at port 'LP:' as file #2

30 sets an echo terminator character and an output terminator of 13 (ASCII **CR**)

40 initializes plotter; directs plotter to output its identification

50 computer reads identification information

60 selects pen 1; moves to the starting label position

70 labels plotter identification and "COMMUNICATION OK"

80 moves paper for viewing of plotted output; returns the pen to the carousel

90 closes files #1 and #2

HP PORTABLE Computer (HP-IB Interface) — *Output Only*

```
10 OPEN "O",1,"PLT"
20 PRINT #1,"IN;SP1;PA500,500;"
30 PRINT #1,"LBCOMMUNICATION OK"+CHR$(3)
40 PRINT #1,"PA0,0;SP0;"
50 END
```

Program Explanation

10 opens the "PLT" port; "O" specifies the port will be used for output; "1" opens file #1

20 initializes the plotter; selects pen 1; moves to the starting label position

30 labels "COMMUNICATION OK"

40 moves paper for viewing of plotted output; returns the pen to the carousel

HP PORTABLE Computer (RS-232-C Interface) — *Output Only*

```
10 OPEN "O",1,"PLT"
20 PRINT #1, "IN;SP1;PA500,500;"
30 PRINT #1, "LBCOMMUNICATION OK"+CHR$(3)
40 PRINT #1, "PA0,0;SP0;"
50 END
```

Program Explanation

10 opens the "PLT" port; "O" specifies the port will be used for output; "1" opens file #1

20 initializes the plotter; selects pen 1; moves to the starting label position

30 labels "COMMUNICATION OK"

40 moves paper for viewing of plotted output; returns the pen to the carousel

HP Series 80 Personal Computer
(HP-IB Interface)

```
10 OUTPUT 705 ;"IN;OI;"
20 ENTER 705 ; I$
30 OUTPUT 705 ;"SP1;PA500,500;"
40 OUTPUT 705 ;"LB"&I$&" COMMUNICATION OK"&CHR$(3)
50 OUTPUT 705 ;"PA0,0;SP0;"
60 END
```

NOTE: Series 80 computers can use an extended version of BASIC with graphic capabilities comparable to HP-GL. Refer to your computer's programming documentation for specific instructions. ■

Program Explanation

10	directs output to the plotter at select code 7, address 5; initializes plotter
20	computer reads identification information
30	selects pen 1; moves to the starting label position
40	labels plotter identification and "COMMUNICATION OK"
50	moves paper for viewing of plotted output; returns the pen to the carousel

HP Series 200 Personal Technical Computer
(HP-IB Interface)

```
10 OUTPUT 705 ;"IN;OI;"
20 ENTER 705 ; ID$
30 OUTPUT 705 ;"SP1;PA500,500;"
40 OUTPUT 705 ;"LB"&ID$&" COMMUNICATION OK"&CHR$(3)
50 OUTPUT 705 ;"PA0,0;SP0;"
60 END
```

NOTE: Series 200 computers can use an extended version of BASIC with graphic capabilities comparable to HP-GL. Refer to your computer's programming documentation for specific instructions. ■

Program Explanation

10	directs output to the plotter at select code 7, address 5; initializes plotter; directs plotter to output its identification
20	computer reads identification information
30	selects pen 1; moves to the starting label position
40	labels plotter identification and "COMMUNICATION OK"
50	moves paper for viewing of plotted output; returns the pen to the carousel

HP Touchscreen Personal Computer (HP 150) (HP-IB Interface) *Output Only*

```
10 OPEN "O",1,"PLT"
20 PRINT #1,"IN;SP1;PA500,500;"
30 PRINT #1,"LBCOMMUNICATION OK"+CHR$(3)
40 PRINT #1,"PA0,0;SP0;"
50 END
```

Program Explanation

10 opens the "PLT" port; "O" specifies the port will be used for output; "1" opens file #1

20 initializes plotter; selects pen 1; moves to the starting label position

30 labels "COMMUNICATION OK"

40 moves paper for viewing of plotted output; returns the pen to the carousel

Interfacing Suppl.

HP Touchscreen Personal Computer (HP 150) (RS-232-C Interface)

In the following program, notice the way the ports are opened and closed to accept and read data from the plotter (lines 10, 30, 40, and 60). This technique will be helpful when you use output instructions.

```
10 OPEN "O",1,"PLT"
20 PRINT #1, "IN;OI;"
30 CLOSE #1
40 OPEN "I",2,"PLT"
50 INPUT #2, ID$
60 OPEN "O",1,"PLT"
70 PRINT #1, "SP1;PA500,500;"
80 PRINT #1, "LB";ID$;" COMMUNICATION OK"+CHR$(3)
90 PRINT #1, "PA0,0;SP0;"
100 END
```

NOTE: If you are using GW™-BASIC and an RS-232-C interface, this program will not run without error. We recommend that you use the program example shown for the HP Touchscreen Personal Computer (HP 150) with an *HP-IB* interface. It is not necessary to change your RS-232-C interface conditions. ■

Program Explanation

10 opens the "PLT" port; "O" specifies the port will be used for output; "1" opens file #1

20 initializes plotter; directs plotter to output its identification to the computer

30 closes the output port

40 opens the "PLT" port; "I" specifies the port will be used for input; "2" opens file #2

50 computer reads identification information

60 opens the "PLT" port; "O" specifies the port will be used for output; "1" opens file #1

70 selects pen 1; moves to the starting label position

80 labels plotter identification and "COMMUNICATION OK"

90 moves paper for viewing of plotted output; returns the pen to the carousel

IBM Personal Computer (PC and PC-XT)
(RS-232-C Interface)

```
10 OPEN "COM1:9600,S,7,1,RS,CS65535,DS,CD" AS #1
20 PRINT #1, "IN;OI;"
30 INPUT #1, ID$
40 PRINT #1, "SP1;PA500,500;"
50 PRINT #1, "LB"+ID$+" COMMUNICATION OK"+CHR$(3)
60 PRINT #1, "PA0,0;SP0;"
70 END
```

Program Explanation

10 sets the following RS-232-C interface conditions: 9600 baud rate, parity off, 7 data bits, 1 stop bit, and hardwire handshake; directs computer output to the plotter at the COM1: port; "#1" designates file 1 as the file which will hold plotter data

20 initializes the plotter; directs plotter to output its identification

30 computer reads identification information

40 selects pen 1; moves to the starting label position

50 labels plotter identification and "COMMUNICATION OK"

60 moves paper for viewing of plotted output; returns the pen to the carousel

IBM AT Computer
(RS-232-C Interface)

```
10 OPEN "COM1:9600,S,7,1,RS,CS65535,DS,CD" AS #1
20 PRINT #1, "IN;OI;"
30 INPUT #1, ID$
40 PRINT #1, "SP1;PA500,500;"
50 PRINT #1, "LB"+ID$+" COMMUNICATION OK"+CHR$(3)
60 PRINT #1, "PA0,0;SP0;"
70 END
```

Program Explanation

10 sets the following RS-232-C interface conditions: 9600 baud rate, parity off, 7 data bits, 1 stop bit, and hardwire handshake; directs computer output to the plotter at the COM1: port; "#1" designates file 1 as the file which will hold plotter data

20 initializes the plotter; directs plotter to output its identification

30 computer reads identification information

40 selects pen 1; moves to the starting label position

50 labels plotter identification and "COMMUNICATION OK"

60 moves paper for viewing of plotted output; returns the pen to the carousel

Olivetti M24 Computer
(RS-232-C Interface)

```
10 OPEN "COM1:9600,S,7,1,RS,CS65535,DS,CD" AS #1
20 PRINT #1, "IN;OI;"
30 INPUT #1, ID$
40 PRINT #1, "SP1;PA500,500;"
50 PRINT #1, "LB"+ID$+" COMMUNICATION OK"+CHR$(3)
60 PRINT #1, "PA0,0;SP0;"
70 END
```

Program Explanation

10 sets the following RS-232-C interface conditions: 9600 baud rate, parity off, 7 data bits, 1 stop bit, and hardwire handshake; directs computer output to the plotter at the COM1: port; "#1" designates file 1 as the file which will hold plotter data

20 initializes the plotter; directs plotter to output its identification

30 computer reads identification information

40 selects pen 1; moves to the starting label position

50 labels plotter identification and "COMMUNICATION OK"

60 moves paper for viewing of plotted output; returns the pen to the carousel

Sirius 1 Computer ***Output Only***
(RS-232-C Interface)

```
10 OPEN "O",#1,"AUX"
20 PRINT #1, "IN;SP1;PA500,500;"
30 PRINT #1, "LB COMMUNICATION OK";CHR$(3)
40 PRINT #1, "PA0,0;SP0;"
50 END
```

Program Explanation

10 opens the "AUX" port; "O" specifies the port will be used for output; "#1" opens file #1

20 initializes plotter; selects pen 1; moves to the starting label position

30 labels "COMMUNICATION OK"

40 moves paper for viewing of plotted output; returns the pen to the carousel

Quick Reference of Handshake Types

Following is a quick reference and summary of the RS-232-C instructions required for each handshake method. For more detailed information on interfacing and handshake methods and how to choose the best method for your system, refer to the Hewlett-Packard Plotter Note No. 6 (Part No. 5953-9770).

Handled by Operating System			**Handled by Software**	
Hardwire	**Xon-Xoff**	**ENQ/ACK (Mode 2)**	**ENQ/ACK (Mode 1)**	**Buffer Checking**
ESC.P or ESC.@	ESC.P or ESC.N and ESC.I	ESC.P or ESC.I	— ESC.M and ESC.H	— ESC.M and ESC.B

Transmission Errors

Transmission errors occur when communication between the computer and plotter is incomplete or does not conform to what is expected or required by either device.

Transmission errors include:

- Framing Error — The plotter does not detect a valid stop bit at the end of every character.
- Parity Error — The plotter does not detect the expected parity (odd or even). This error is inhibited if parity is set to "off".
- Buffer Overflow Error — The plotter receives more bytes of data than it has space for in the I/O buffer.

When the plotter detects a framing or parity error, it sets error 15. This error generally indicates that the communication problem is hardware-related (e.g., wrong parity selection, incompatible or incorrectly set baud rates, etc.) If a parity error occurs, the bad character is replaced with the ASCII delete character DEL (decimal code 127).

When the plotter detects an I/O buffer overflow, it sets error 16 and the last character received is replaced with the DEL character. The HP-GL data that caused the overflow will be lost. Error 16 generally indicates an improperly established handshake protocol.

You can determine the error number using the ESC . E instruction in your program. A complete list of RS-232-C errors is included with the discussion of the ESC . E instruction in Chapter 12.

Mechanical Interface and Connector Pin Allocations

RS-232-C/CCITT V.24 Interface Implementation

The plotter interfaces to the RS-232-C communications lines through a standard 25-pin connector. The plotter is compatible with RS-232-C and CCITT V.24 protocols.

When a hardwire handshake method is used, the Data Terminal Ready (DTR) line (pin 20) is used to signal whether space is available in the logical I/O buffer for more data.

Connector Pin Allocations

Details of the connector pin allocations, including pin numbers, signal directions, and signal levels for the RS-232-C, CCITT V.24 interface specifications, are shown in the following table. Pins not listed are not used or connected.

RS-232-C/CCITT V.24 Interface Pin Allocations

Pin No.	Function	RS-232-C	CCITT V.24	Signal Direction and Level
1	Protective Ground	AA	(none)	Not applicable

2	Transmitted Data (TD)	BA	103	Data from plotter
				High = SPACE = "0" = +12 V
				Low = MARK = "1" = −12 V

3	Received Data (RD)	BB	104	Data to plotter
				High = SPACE = "0" = +3 V to +25 V
				Low = MARK = "1" = −3 V to −25 V

4	Request to Send (RTS)	CA	105	Signal from plotter
				High = ON = +12 V
				Low = OFF = −12 V

5	Clear to Send (CTS)	CB	106	Signal to plotter
				High = ON = +3 V to +25 V
				Low = OFF = −3 V to −25 V

6	Data Set Ready (DSR)	CC	107	Signal to plotter
				High = ON = +3 V to +25 V
				Low = OFF = −3 V to −25 V

7	Signal Ground (SGND)	AB	102	Not applicable

8	Data Carrier Detect (DCD)	CF	109	Signal to plotter
				High = ON = +3 V to +25 V
				Low = OFF = −3 V to −25 V

20	Data Terminal Ready (DTR)	CD	108.2	Signal from plotter
				High = ON = +12 V
				Low = OFF = −12 V

APPENDIX B

Reference Material

ASCII Character Codes

Numbers are often used as a code to represent not only values, but also alphanumeric characters such as "A" or "," or "x" or "2". One of the most common computer codes used is ASCII*. ASCII is an eight-bit code, containing seven data bits and one parity bit. The plotter uses ASCII for I/O operations. The parity bit here is set to 0. For example:

Character	ASCII Binary Code	ASCII Decimal Code
A	01000001	65
B	01000010	66
?	00111111	63

Character Sets and ASCII Codes

The plotter has five character sets, named in the following table. All characters in each set are shown at the end of the section.

*American Standard Code for Information Interchange

Standard Plotter		
Set Number	Name	ISO Registration Number
0	ANSI ASCII	006
1	HP 9825 HPL Character Set	—
2	French/German	—
3	Scandinavian	—
4	Spanish/Latin American	—

When the plotter is equipped with the graphics enhancement cartridge, 14 additional character sets are available. These sets are named in the following table. All characters are shown at the end of this section.

Graphics Enhancement Cartridge		
Set Number	Name	ISO Registration Number
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 Norwegian, Version 1	060
33	ISO German	021
34	ISO French	025
35	ISO British	004
36	ISO Italian	015
37	ISO Spanish	017
38	ISO Portuguese	016
39	ISO Norwegian, Version 2	061

The plotter's reactions to nonprinting ASCII control characters (decimal codes 0–32) while in label mode are shown in the following table. These reactions are true regardless of the character set being 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)
14	SI	Shift-In (Select Standard Character Set)
16	DLE	NOP
17	DC1	NOP
18	DC2	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)

Decimal Code	ASCII Character	All Sets
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's character sets are shown on the following pages. All printing ASCII characters and their decimal codes (33–126) are listed in each character set.

NOTE: Each of the shaded symbols is automatically backspaced one character before it is drawn. Therefore, when an accented letter is required, the letter should be entered first, followed by the backspaced character. ■

Printing Characters

Decimal			Set										Set							
Code	0	1	2	3	4	6	7	8	9	30	31	32	33	34	35	36	37	38	39
33	!	!	!	!	!	!	À	｡	!	!	!	!	!	!	!	!	!	!	!
34	"	"	"	"	"	"	Â	｢	"	"	"	"	"	"	"	"	"	"	"
35	#	#	£	£	¿	#	È	｣	#	#	#	#	#	£	£	£	£	#	§
36	$	$	$	$	$	$	Ê	､	¤	¤	¤	$	$	$	$	$	$	$	$
37	%	%	%	%	%	%	Ë	･	%	%	%	%	%	%	%	%	%	%	%
38	&	&	&	&	&	&	Î	ｦ	&	&	&	&	&	&	&	&	&	&	&
39	'	'	□	'	□	'	Ï	ｧ	'	'	'	'	'	'	'	'	'	'	'
40	(	(	(	(	(	(	´	ｨ	(	(	(	(	(	(	(	(	(	(	(
41	)	)	)	)	)	)	`	ｩ	)	)	)	)	)	)	)	)	)	)	)
42	*	*	*	*	*	*	ˆ	ｪ	*	*	*	*	*	*	*	*	*	*	*
43	+	+	+	+	+	+	¨	ｫ	+	+	+	+	+	+	+	+	+	+	+
44	,	,	,	,	,	,	~	ｬ	,	,	,	,	,	,	,	,	,	,	,
45	-	-	-	-	-	-	Ù	ｭ	-	-	-	-	-	-	-	-	-	-	-
46	.	.	.	.	.	.	Û	ｮ	.	.	.	.	.	.	.	.	.	.	.
47	/	/	/	/	/	/	£	ｯ	/	/	/	/	/	/	/	/	/	/	/
48	0	0	0	0	0	0	¯	ｰ	0	0	0	0	0	0	0	0	0	0	0
49	1	1	1	1	1	1		ｱ	1	1	1	1	1	1	1	1	1	1	1
50	2	2	2	2	2	2		ｲ	2	2	2	2	2	2	2	2	2	2	2
51	3	3	3	3	3	3	°	ｳ	3	3	3	3	3	3	3	3	3	3	3
52	4	4	4	4	4	4	Ç	ｴ	4	4	4	4	4	4	4	4	4	4	4
53	5	5	5	5	5	5	ç	ｵ	5	5	5	5	5	5	5	5	5	5	5
54	6	6	6	6	6	6	Ñ	ｶ	6	6	6	6	6	6	6	6	6	6	6
55	7	7	7	7	7	7	ñ	ｷ	7	7	7	7	7	7	7	7	7	7	7
56	8	8	8	8	8	8	¡	ｸ	8	8	8	8	8	8	8	8	8	8	8
57	9	9	9	9	9	9	¿	ｹ	9	9	9	9	9	9	9	9	9	9	9
58	:	:	:	:	:	:	¤	ｺ	:	:	:	:	:	:	:	:	:	:	:
59	;	;	;	;	;	;	£	ｻ	;	;	;	;	;	;	;	;	;	;	;
60	<	<	<	<	<	<	¥	ｼ	<	<	<	<	<	<	<	<	<	<	<
61	=	=	=	=	=	=	§	ｽ	=	=	=	=	=	=	=	=	=	=	=
62	>	>	>	>	>	>	ƒ	ｾ	>	>	>	>	>	>	>	>	>	>	>
63	?	?	?	?	?	?	¢	ｿ	?	?	?	?	?	?	?	?	?	?	?
64	@	@	@	@	@	@	â	ﾀ	@	@	É	@	§	à	@	§	§	§	@

(Table continues)

Reference Material

Printing Characters (Continued)

Decimal Code	Set 0	1	2	3	4	Set 6	7	8	9	30	31	32	33	34	35	36	37	38	39
65	A	A	A	A	A	A	ê	ﾁ	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
67	C	C	C	C	C	C	û	ﾃ	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
69	E	E	E	E	E	E	é	ﾅ	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
71	G	G	G	G	G	G	ú	ﾇ	G	G	G	G	G	G	G	G	G	G	G
72	H	H	H	H	H	H	à	ﾈ	H	H	H	H	H	H	H	H	H	H	H
73	I	I	I	I	I	I	è	ﾉ	I	I	I	I	I	I	I	I	I	I	I
74	J	J	J	J	J	J	ò	ﾊ	J	J	J	J	J	J	J	J	J	J	J
75	K	K	K	K	K	K	ù	ﾋ	K	K	K	K	K	K	K	K	K	K	K
76	L	L	L	L	L	L	ä	ﾌ	L	L	L	L	L	L	L	L	L	L	L
77	M	M	M	M	M	M	ë	ﾍ	M	M	M	M	M	M	M	M	M	M	M
78	N	N	N	N	N	N	ö	ﾎ	N	N	N	N	N	N	N	N	N	N	N
79	O	O	O	O	O	O	ü	ﾏ	O	O	O	O	O	O	O	O	O	O	O
80	P	P	P	P	P	P	Å	ﾐ	P	P	P	P	P	P	P	P	P	P	P
81	Q	Q	Q	Q	Q	Q	î	ﾑ	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q
82	R	R	R	R	R	R	Ø	ﾒ	R	R	R	R	R	R	R	R	R	R	R
83	S	S	S	S	S	S	Æ	ﾓ	S	S	S	S	S	S	S	S	S	S	S
84	T	T	T	T	T	T	å	ﾔ	T	T	T	T	T	T	T	T	T	T	T
85	U	U	U	U	U	U	í	ﾕ	U	U	U	U	U	U	U	U	U	U	U
86	V	V	V	V	V	V	ø	ﾖ	V	V	V	V	V	V	V	V	V	V	V
87	W	W	W	W	W	W	æ	ﾗ	W	W	W	W	W	W	W	W	W	W	W
88	X	X	X	X	X	X	Ä	ﾘ	X	X	X	X	X	X	X	X	X	X	X
89	Y	Y	Y	Y	Y	Y	ì	ﾙ	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
90	Z	Z	Z	Z	Z	Z	Ö	ﾚ	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z	Z
91	[	[	[	Ø	[	[	Ü	ﾛ	[	Ä	Ä	Æ	Ä	°	[	°	¡	Ã	Æ
92	\	√	ç	Æ	¡	¥	É	ﾜ	\	Ö	Ö	Ø	Ö	ç	\	ç	Ñ	Ç	Ø
93	]	]	]	ø	]	]	ï	ﾝ	]	Å	Å	Å	Ü	§	]	é	¿	Õ	Å
94	^	↑	^	æ	^	^	ß	ﾞ	^	^	Ü	^	^	^	^	^	^	^	^
95	_	_	_	_	_	_	Ô	ﾟ	_	_	_	_	_	_	_	_	_	_	_
96	`	`	`	`	`	`	Á		`	`	é	`	`	`	`	ù	`	`	`

(Table continues)

Printing Characters (Continued)

Decimal Code	Set 0	1	2	3	4	Set 6	7	8	9	30	31	32	33	34	35	36	37	38	39
97	a	a	a	a	a	a	Ã		a	a	a	a	a	a	a	a	a	a	a
98	b	b	b	b	b	b	ã		b	b	b	b	b	b	b	b	b	b	b
99	c	c	c	c	c	c	Đ		c	c	c	c	c	c	c	c	c	c	c
100	d	d	d	d	d	d	đ		d	d	d	d	d	d	d	d	d	d	d
101	e	e	e	e	e	e	Í		e	e	e	e	e	e	e	e	e	e	e
102	f	f	f	f	f	f	Ì		f	f	f	f	f	f	f	f	f	f	f
103	g	g	g	g	g	g	Ó		g	g	g	g	g	g	g	g	g	g	g
104	h	h	h	h	h	h	Ò		h	h	h	h	h	h	h	h	h	h	h
105	i	i	i	i	i	i	Õ		i	i	i	i	i	i	i	i	i	i	i
106	j	j	j	j	j	j	õ		j	j	j	j	j	j	j	j	j	j	j
107	k	k	k	k	k	k	Š		k	k	k	k	k	k	k	k	k	k	k
108	l	l	l	l	l	l	š		l	l	l	l	l	l	l	l	l	l	l
109	m	m	m	m	m	m	Ú		m	m	m	m	m	m	m	m	m	m	m
110	n	n	n	n	n	n	Ÿ		n	n	n	n	n	n	n	n	n	n	n
111	o	o	o	o	o	o	ÿ		o	o	o	o	o	o	o	o	o	o	o
112	p	p	p	p	p	p	Þ		p	p	p	p	p	p	p	p	p	p	p
113	q	q	q	q	q	q	þ		q	q	q	q	q	q	q	q	q	q	q
114	r	r	r	r	r	r			r	r	r	r	r	r	r	r	r	r	r
115	s	s	s	s	s	s			s	s	s	s	s	s	s	s	s	s	s
116	t	t	t	t	t	t			t	t	t	t	t	t	t	t	t	t	t
117	u	u	u	u	u	u			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
119	w	w	w	w	w	w	¼		w	w	w	w	w	w	w	w	w	w	w
120	x	x	x	x	x	x	½		x	x	x	x	x	x	x	x	x	x	x
121	y	y	y	y	y	y	ª		y	y	y	y	y	y	y	y	y	y	y
122	z	z	z	z	z	z	º		z	z	z	z	z	z	z	z	z	z	z
123	{	₶	˜	¯	~	{	«		{	ä	ä	æ	ä	é	{	à	˙	ä	æ
124	|	⊦	˙	˙	~	|	■		|	ö	ö	ø	ö	ù	|	ò	ñ	ç	ø
125	}	–	˜	¯	~	}	»		}	å	å	å	ü	è	}	è	ç	ö	å
126	~	~	˙	˙	~	~	±		¯	¯	ü	¯	ß	˙	¯	ì	˜	˙	|

Computer Museum

Default Conditions Established by the DF Instruction

The following table shows default conditions established by the DF instruction. Additional conditions are listed after table.

Default Conditions

Function	Equivalent Instructions	Conditions
Plotting mode	PA;	Absolute
Line type	LT;	Solid line
Line pattern length	LT;	4% of the diagonal distance between P1 and P2
Input window	IW;	Set to current hard-clip limits
Relative character direction	DR1,0;	Horizontal
Relative character size	SR;	Width =0.75% of $(P2_X - P1_X)$ Height = 1.5% of $(P2_Y - P1_Y)$
Standard character set	CS0;	Set 0
Alternate character set	CA0;	Set 0
Character set selected	SS;	Standard
Character slant	SL0;	0 degrees
Label terminator	DT**ETX**;	**ETX** (ASCII decimal code 3)
Symbol mode	SM;	Off
Tick length	TL;	tp = tn = 0.5% or $\lvert P2_X - P1_X \rvert$ for Y-tick and 0.5% of $\lvert P2_Y - P1_Y \rvert$ for X-tick
Mask value	IM 223,0,0;	Recognizes all defined errors except 6
Digitize clear	DC;	Off

(Table continues)

Reference Material

Default Conditions (Continued)

Function	Equivalent Instructions	Conditions
Scale	SC;	User-unit scaling off
Pen velocity	VS;	40 cm/s (15.7 in./s)
Chord angle	—	5 degrees
Fill type	FT;	Type 1, bidirectional solid fill
Fill spacing	FT;	1% of diagonal distance between P1 and P2
Fill angle	FT;	0 degrees
Pen thickness	PT;	0.3 mm
Polygon mode	PM0,PM2;	Polygon buffer cleared

When DF is executed, the carriage-return point for labeling instructions is updated to the current pen position.

Absolute character size defined with the SI instruction defaults to the same as *SR;*.

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

- locations of P1 and P2
- current pen, its position, and up/down status
- 90-degree rotation
- generated errors (not cleared)

Conditions Established by the IN Instruction

The initialize instruction sets the plotter to the same conditions as the default instruction, DF, and sets these additional conditions:

- raises the pen

- sets P1 and P2 to default conditions

  $P1_x = 250$ $P2_x = 10250$
  $P1_y = 279$ $P2_y = 7479$

- cancels 90-degree rotation
- sets bit position 3 of the status word to 1 (to indicate the plotter has been initialized)
- clears any HP-GL or RS-232-C error condition

No Operation (NOP) Instructions

In order to maintain software compatibility with other HP plotters, the HP ColorPro recognizes the following device-control instructions as no operation (NOP) instructions. They are ignored and no error is generated.

ESC.(or ESC.Y
ESC.) or ESC.Z

Scaling without Using the SC Instruction

The plotter movements are in terms of plotter units where a plotter unit = 0.025 mm. While the plotter can be scaled into user units using the SC instruction, it might be more convenient for you to write programs where numbers to be plotted are in some units other than plotter units or conventional user units. If so, you can use the following equations in your program to convert these "other units" into plotter units for the plotter:

$$X_{scaled} = \left[\frac{P2_X - P1_X}{U2_X - U1_X}\right] A_X + P1_X - U1_X\left[\frac{P2_X - P1_X}{U2_X - U1_X}\right]$$

$$Y_{scaled} = \left[\frac{P2_Y - P1_Y}{U2_Y - U1_Y}\right] A_Y + P1_Y - U1_Y\left[\frac{P2_Y - P1_Y}{U2_Y - U1_Y}\right]$$

where: A_X is the X-coordinate of the desired point in user units,

A_Y is the Y-coordinate of the desired point in user units,

$P1_X$ is the X-coordinate of P1 in plotter units,

$P1_Y$ is the Y-coordinate of P1 in plotter units,

$P2_X$ is the X-coordinate of P2 in plotter units,

$P2_Y$ is the Y-coordinate of P2 in plotter units,

$U1_X$ is the X-coordinate of P1 in user units,

$U1_Y$ is the Y-coordinate of P1 in user units,

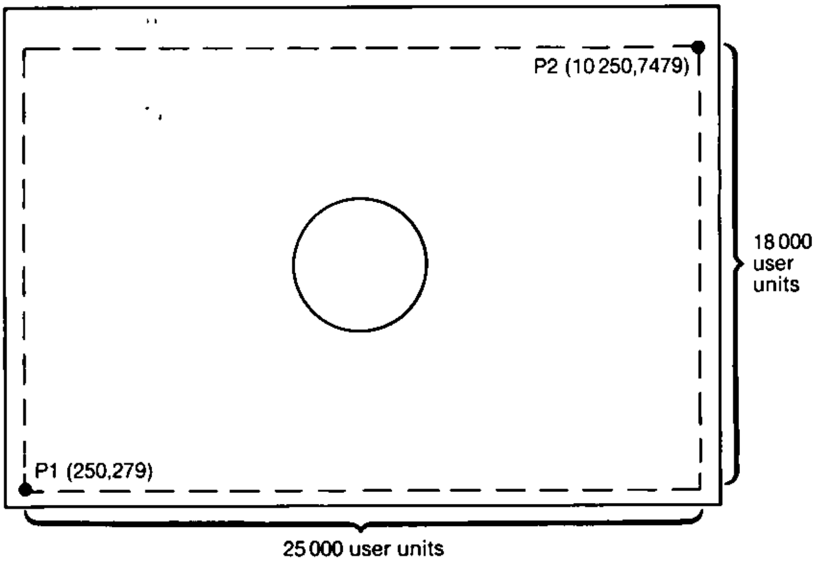
$U2_X$ is the X-coordinate of P2 in user units, and

$U2_Y$ is the Y-coordinate of P2 in user units.

To demonstrate the use of the scaling equations, let's go through an example.

Problem

Scale the P1/P2 area (P1 = 250,279, and P2 = 10 250,7479 plotter units) into new user units where P1 = 0,0 and P2 = 25 000,18 000. At the center point (X = 12 500 and Y = 9000 user units), draw a circle with radius 2500 as shown below.



Reference Material

Solution

1. Recall that the equations of a circle are:

$$X = R \cos t$$
$$Y = R \sin t$$
where $0 \leqslant t \leqslant 2\pi$

2. Since we are to plot relative to a point that is not at the origin, an offset X_O, Y_O must be added to the circle equations. The offset in user units is:

$$X_O = 12\,500$$
$$Y_O = 9000$$

3. The desired circle equations are then:

$$A_X = 2500 \cos t + 12\,500$$
$$A_Y = 2500 \sin t + 9000$$

4. Determine the user scale:

$$X = 0 \text{ to } 25\,000$$
$$Y = 0 \text{ to } 18\,000$$

therefore

$$U1_X = 0$$
$$U1_Y = 0$$
$$U2_X = 25\,000$$
$$U2_Y = 18\,000$$

5. Determine the values for P1 and P2 which are set using the IN instruction:

$$P1 = 250\,,\,279$$
$$P2 = 10\,250\,,\,7479$$

therefore

$$P1_X = 250$$
$$P1_Y = 279$$
$$P2_X = 10\,250$$
$$P2_Y = 7479$$

6. Solving for X and Y, determine the equivalent plotter-unit values for the circle equations:

$$X = \left[\frac{P2_X - P1_X}{U2_X - U1_X}\right] A_X + P1_X - U1_X \left[\frac{P2_X - P1_X}{U2_X - U1_X}\right]$$

$$= \left[\frac{10\,250 - 250}{25\,000 - 0}\right] (2500 \cos t + 12\,500) + 250 - 0 \left[\frac{10\,250 - 250}{25\,000 - 0}\right]$$

$$= 0.4\,(2500 \cos t + 12\,500) + 250 - 0$$

$$= 1000 \cos t + 5250$$

$$Y = \left[\frac{P2_Y - P1_Y}{U2_Y - U1_Y}\right] A_Y + P1_Y - U1_Y \left[\frac{P2_Y - P1_Y}{U2_Y - U1_Y}\right]$$

$$+ \left[\frac{7479 - 279}{18\,000 - 0}\right] (2500 \sin t + 9000) + 279 - 0 \left[\frac{7479 - 279}{18\,000 - 0}\right]$$

$$= 0.4\,(2500 \sin t + 9000) + 279 - 0$$

$$= 1000 \sin t + 3879$$

7. The following program will plot the required circle using the equivalent plotter-unit values, and the default P1 and P2.

```
10  REM  Insert configuration statement here
20  PRINT #1, "IN;IP250,279,10250,7479;SP1;"
30  PI=3.141593
40  FOR T=0 TO 2*PI+PI/20 STEP PI/20
50    X=1000*COS(T)+5250
60    Y=1000*SIN(T)+3879
70    PRINT #1, "PA";X;",";Y;";PD;"
80  NEXT T
90  PRINT #1, "SP0;"
100 END
```

Notes

APPENDIX C

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 10). To read a device-control error number in your program, use the ESC . E instruction (presented in Chapter 12).

The following tables list the error messages and their meanings.

HP-GL Errors

Error No.	Meaning	Usual Plotter Reaction	Possible Cause
1	Instruction not recognized	Ignores the instruction	A mnemonic is incorrect or missing; an alphabetic character was specified in a parameter when a numeric character was expected.*

*Check for typographical errors in the mnemonic and the parameters (a common mistake is to type O for 0, or vice versa). Also, check to be sure instructions are not being terminated prematurely. For example, if you are using an HP-IB configuration, check to be sure the computer is not inserting line feeds (LF) in the middle of a string of instructions. If an instruction is terminated after the mnemonic and before the parameters, the subsequent receipt of the parameters without a mnemonic will cause error 1.

(Table continues)

HP-GL Errors (Continued)

Error No.	Meaning	Usual Plotter Reaction	Possible Cause
2	Wrong number of parameters	If too few parameters, ignores the instruction. If too many parameters, executes the instruction with the correct number of parameters and ignores the rest.	Too few or too many parameters; an incomplete X,Y coordinate pair.
3	Bad parameter	Ignores the instruction	A parameter is out-of-range.**
4	Unused	Unused	Unused
5	Unknown character set	Ignores the instruction	A set other than 0–4 (or 6–9 and 30–39 with a cartridge) has been designated or selected.
6	Position overflow	Ignores the instruction	A single label is so long that it exceeds the plotter's numeric range.
7	Buffer overflow	Executes the data that fits in the polygon buffer and ignores the data that overflows	The polygon buffer does not have enough space allocated.***

**Check each parameter range for the instructions that are suspected of causing the error. Parameter ranges are listed at the beginning of each instruction's discussion throughout this manual, as well as in Appendix D.

***Check buffer allocations. Refer to the ESC . T and the ESC . S instructions in Chapter 12 for information on the plotter's buffers.

Device-Control Errors

Error No.	Meaning	Possible Cause
0	None	No I/O error has occurred.
10	Invalid output request	New output has been generated 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	Invalid byte following **ESC.**	Invalid character received after first two characters (**ESC.**) in a device-control instruction.
12	Invalid byte in device-control instruction	Invalid character received while parsing a device-control instruction. The parameter containing the invalid character and all following parameters are defaulted.
13	Out-of-range 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. **NOTE:** The receipt of a character other than another parameter, a semicolon, or a colon will result in error 12 overwriting error 14. ■
15	Error in I/O transmission	A framing error or parity error has been detected.
16	I/O buffer overflow	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.

Notes

APPENDIX D

Instruction Summary

HP-GL Instructions

This section lists the formal syntax and parameter ranges for HP-GL instructions, in alphabetical order of the instruction's mnemonic. For a complete description of each instruction, refer to the indicated page number.

The semicolon is included as the terminator, although *both* the semicolon and the next mnemonic are valid terminators. (In an HP-IB configuration, you can also use a line-feed character [LF] as a terminator.)

[TERM] means the terminator sent by the plotter at the end of an output response. It is [CR LF] in an HP-IB configuration, and [CR] (or as set by the ESC . M instruction) in an RS-232-C configuration.

AA Arc Absolute Instruction Page 7-7

AA X,Y, arc angle (, chord angle);

Parameter	Format	Range	Default
X,Y coordinates	real	−32 768.0000 to 32 767.9999 current units	none
arc angle	real	−32 768.0000 to 32 767.9999 degrees	none
chord angle	real	0.1 to 180 degrees*	5 degrees

*This is the practical range. The allowable range is −32 768.0000 to 32 767.9999.

AR Arc Relative Instruction Page 7-7

AR X,Y, arc angle (, chord angle);

Parameter	Format	Range	Default
X,Y coordinates	real	−32 768.0000 to 32 767.9999 current units	none
arc angle	real	−32 768.0000 to 32 767.9999 degrees	none
chord angle	real	0.1 to 180 degrees*	5 degrees

*This is the practical range. The allowable range is −32 768.0000 to 32 767.9999.

CA Designate Alternate Character Set Instruction

Page 6-43

CA set;
or
CA;

Parameter	Format	Range	Default
set	integer	0 to 4 without cartridge	0
		0 to 4, 6 to 9, 30 to 39 with cartridge	0

CI Circle Instruction

Page 7-2

CI radius (,chord angle);

Parameter	Format	Range	Default
radius	real	−32 768.0000 to 32 767.9999 current units	none
chord angle	real	0.1 to 180 degrees*	5 degrees

*This is the practical range. The allowable range is −32 768.0000 to 32 767.9999 degrees.

CP Character Plot Instruction Page 6-32

CP spaces, lines;
or
CP;

Parameter	Format	Range	Default
spaces	real	−128.0000 to 127.9999	none
lines	real	−128.0000 to 127.9999	none

CS Designate Standard Character Set Instruction Page 6-42

CS set;
or
CS;

Parameter	Format	Range	Default
set	integer	0 to 4 without cartridge	0
		0 to 4, 6 to 9, 30 to 39 with cartridge	0

DC Digitize Clear Instruction Page 11-4

DC;

DF Default Instruction Page 3-9

DF;

See table in Appendix B or Chapter 3.

DI Absolute Direction Instruction Page 6-22

DI run, rise;
or
DI;

Parameter	Format	Range	Default
run (or cos θ)	real	−128.0000 to 127.9999	1
rise (or sin θ)	real	−128.0000 to 127.9999	0

DP Digitize Point Instruction Page 11-3

DP;

DR Relative Direction Instruction Page 6-27

DR run, rise;
or
DR;

Parameter	Format	Range	Default
run (or cos θ)	real	−128.0000 to 127.9999	1% of $P2_X - P1_X$
rise (or sin θ)	real	−128.0000 to 127.9999	0% of $P2_Y - P1_Y$

DT Define Label Terminator Instruction Page 6-9

DT label terminator **ETX**;

Parameter	Format	Range	Default
label terminator	label	any character except **NULL** (decimal code 0)	**ETX** (decimal code 3)

EA Edge Rectangle Absolute Instruction Page 7-24

EA X,Y;

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

*If scaling is on, the range may be smaller.

EP Edge Polygon Instruction Page 7-43

EP;

ER Edge Rectangle Relative Instruction Page 7-30

ER X,Y;

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

*If scaling is on, the range may be smaller.

EW Edge Wedge Instruction Page 7-17

EW radius, start angle, sweep angle, (, chord angle);

Parameters	Format	Range	Default
radius	real	−32 768.0000 to 32 767.9999 degrees	none
start angle	real	−32 768.0000 to 32 767.9999 degrees	none
sweep angle	real	−32.768.0000 to 32 767.9999 degrees	none
chord angle	real	0.1 to 180 degrees*	5 degrees

*This is the practical range. The allowable range is −32 768.0000 to 32 767.9999.

FP Fill Polygon Instruction Page 7-44

FP;

FT Fill Type Instruction Page 7-12

FT type (, spacing (, angle));
or
FT;

Parameter	Format	Range	Default
fill type	integer	1 to 4	1
spacing	real	1 to 32 767.9999 current units	depends on fill type
angle	real	−32 768.0000 to 32 767.9999 degrees (module 360)	0 degrees

IM Input Mask Instruction Page 10-4

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

*Error numbers 1, 2, 3, 4, 5, and 7 will be reported.

IN Initialize Instruction Page 3-11

IN;

IP Input P1 and P2 Instruction Page 3-12

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

Parameter	Format	Range	Default
X,Y coordinates	integer	−32 768 to 32 767 plotter units	$P1_X = 250$ $P1_Y = 279$ $P2_X = 10250$ $P2_Y = 7479$

IW Input Window

Page 9-4

IW $X_{LL}, Y_{LL}, X_{UR}, Y_{UR}$;
or
IW;

Parameter	Format	Range	Default
X,Y coordinates	integer	−32 768 to 32 767	A4-size 0,0,10 900,7650 A-size 0,0,10 300,7650

LB Label Instruction

Page 6-2

LB c . . . c *term*

(where *term* is **ETX** (decimal code 3) or the label terminator defined by the DT instruction)

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

LT Line Type Instruction

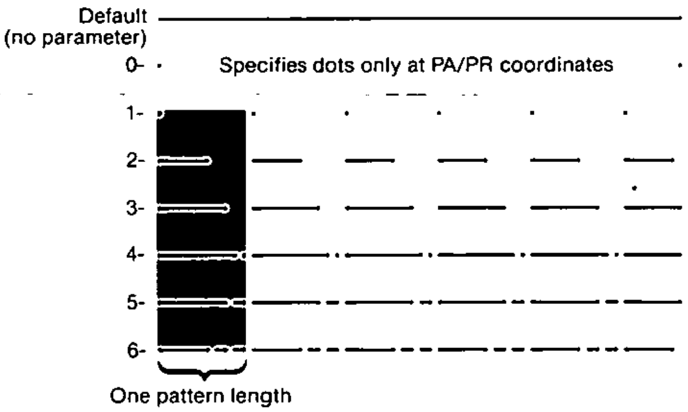
Page 5-8

LT pattern number (, pattern length);
or
LT;

Parameter	Format	Range	Default
pattern number	real	0 to 6*	solid line
pattern length	real	0 to 100*	4% of the diagonal distance between P1 and P2

*This is a practical range. The allowable range is −128.0000 to 127.9999.

The line types for each pattern number are shown below.



OA Output Actual Position and Pen Status Instruction Page 10-12

OA;

Response: X,Y,P [TERM] — integers, in ASCII.

X,Y — in plotter units within current hard-clip limits.

P — 0, pen up or 1, pen down.

OC Output Commanded Position and Pen Status Intruction Page 10-13

OC;

Response: X,Y,P [TERM] — two decimals and one integer, in ASCII.

X,Y — in current units, −32 768.0000 to 32 767.9999.

P — 0, pen up or 1, pen down.

OD Output Digitized Point and Pen Status Instruction — Page 11-3

OD;

Response: X,Y,P [TERM] — integers, in ASCII.

X,Y — in plotter units within current hard-clip limits.

P — 0, pen up or 1, pen down.

OE Output Error Instruction — Page 10-7

OE;

Response: Error number [TERM] — a positive ASCII integer, 0 to 7. (Refer to Appendix C.)

OF Output Factors Instruction — Page 10-14

OF;

Response: 40,40 [TERM] — integers, in ASCII.

OH Output Hard-Clip Limits Instruction — Page 9-8

OH;

Response: X_{LL}, Y_{LL}, X_{UR}, Y_{UR} [TERM] — ASCII integers representing plotter units.

OI Output Identification Instruction — Page 10-9

OI;

Response: 7440A [TERM] — ASCII string, five characters.

OO Output Options Instruction — Page 10-10

OO;

Response: without a cartridge — 0,1,0,0,0,0,0,0 [TERM]

With a cartridge — 0,1,0,0,1,1,0,1 [TERM]

OP Output P1 and P2 Instruction Page 9-9

OP;

Response: $P1_X, P1_Y, P2_X, P2_Y$ [TERM] — ASCII integers representing plotter units.

OS Output Status Instruction Page 10-10

OS;

Response: Status [TERM] — integer in ASCII, 0 to 255. Power-on status, 26.

OW Output Window Instruction Page 9-7

OW;

Response: $X_{LL}, Y_{LL}, X_{UR}, Y_{UR}$ [TERM] — integers, in ASCII.

X,Y — in plotter units.

PA Plot Absolute Instruction Page 4-4

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

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

*If scaling is on, the range may be smaller.

PD Pen Down Instruction

Page 4-3

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

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

*If scaling is on, the range may be smaller.

PM Polygon Mode Instruction

Page 7-34

PM n;

Parameter	Format	Range	Default
n	integer	0 to 2	0

PR Plot Relative Instruction

Page 4-4

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

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

*If scaling is on, the range may be smaller.

PT Pen Thickness Instruction Page 7-15

PT pen thickness;
or
PT;

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

PU Pen Up Instruction Page 4-3

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

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

*If scaling is on, the range may be smaller.

RA Fill Rectangle Absolute Instruction Page 7-24

RA X,Y;

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

*If scaling is on, the range may be smaller.

RO Rotate Coordinate System Instruction Page 9-10

RO n;
or
RO;

Parameter	Format	Range	Default
n	integer	0 or 90 degrees	0 degrees

RR Fill Rectangle Relative Instruction Page 7-30

RR X,Y;

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

*If scaling is on, the range may be smaller.

SA Select Alternate Character Set Instruction Page 6-45

SA;

SC Scale Instruction Page 3-14

SC X_{min}, X_{max}, Y_{min}, Y_{max};
or
SC;

Parameter	Format	Range	Default
X- and Y- ranges	integer	−32 768 to 32 767 user units	none

SI Absolute Character Size Instruction Page 6-14

SI width, height;
or
SI;

Parameter	Format	Range	Default
width	real	−128.0000 to 127.9999	0.187 cm
height	real	−128.0000 to 127.9999	0.269 cm

SL Character Slant Instruction Page 6-19

SL tan θ;
or
SL;

Parameter	Format	Range	Default
tangent θ	real	−128.0000 to 127.9999	0 (no slant)

SM Symbol Mode Instruction Page 5-6

SM character;
or
SM;

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

*The semicolon (decimal code 59) is an HP-GL terminator and cannot be used as a symbol. Use it only to cancel symbol mode (*SM;*).

SP Select Pen Instruction Page 4-2

SP pen number;
or
SP;

Parameter	Format	Range	Default
pen number	integer	0 to 8	none

SR Relative Character Size Instruction Page 6-16

SR width, height;
or
SR;

Parameter	Format	Range	Default
width	real	−128.0000 to 127.9999	0.75% of $\lvert P2_X - P1_X \rvert$
height	real	−128.0000 to 127.9999	1.5% of $\lvert P2_Y - P1_Y \rvert$

SS Select Standard Character Set Instruction Page 6-44

SS;

TL Tick Length Instruction Page 5-3

TL tp (,tn);
or
TL;

Parameter	Format	Range	Default
tp and tn	real	−128.0000 to 127.9999 percentage	0.5% of $\lvert P2_X - P1_X \rvert$ and of $\lvert P2_Y - P1_Y \rvert$

UC User-Defined Character Instruction Page 6-46

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

Parameter	Format	Range	Default
pen control	integer	99 to 127.9999 = pen down −99 to −128.0000 = pen up	pen up
X- and Y-increments	real	−98.9999 to 98.9999	none

VS Velocity Select Instruction Page 4-9

VS pen speed;
or
VS;

Parameter	Format	Range	Default
pen speed	real	0 to 40*	40 cm/s (1.2 g acceleration)

*This is the practical range. The allowable range is 0 to 127.9999.

WG Fill Wedge Instruction Page 7-17

WG radius, start angle, sweep angle, (,chord angle);

Parameters	Format	Range	Default
radius	real	−32 768.0000 to 32 767.9999 degrees	none
start angle	real	−32 768.0000 to 32 767.9999 degrees	none
sweep angle	real	−32.768.0000 to 32 767.9999 degrees	none
chord angle	real	0.1 to 180 degrees*	5 degrees

*This is the practical range. The allowable range is −32 768.0000 to 32 767.9999.

XT X-Tick Instruction Page 5-2

XT;

YT Y-Tick Instruction Page 5-2

YT;

Device-Control Instructions

This section lists the formal syntax for device-control instructions in alphabetical order of the escape sequence. In order to use device-control instructions, you must be using an RS-232-C/ CCITT V.24 interface. Refer to the indicated page number for details.

Set Plotter Configuration   Page 12-8

ESC.@ [(logical I/O buffer size);(I/O conditions)]:

Parameter	Format	Range	Default
logical I/O buffer size*	real	1 to 1974	1024 or size of physical I/O buffer
I/O conditions	real	0 to 31	hardwire handshake, normal mode

*This parameter can only be used if the cartridge is installed.

Output Identification   Page 12-13

ESC.A

Response: plotter model number (ASCII string), firmware revision level (integer).

Output Buffer Space   Page 12-11, 13-32

ESC.B

Response: available buffer space [TERM] — 0 to 60 bytes without cartridge, 0 to 1024 with cartridge.

Output Extended Error   Page 12-14, 13-32

ESC.E

Response: I/O error number — 0 (no error) or 10 to 16.

Set Handshake Mode 1

Page 13-28

ESC.H [(data block size);(enquiry character);(acknowledgment string)]:

Parameter	Format	Range	Default
data block size	real	0 to 70 bytes without cartridge	20 bytes
		0 to 1974 bytes with cartridge	80 bytes
enquiry character	ASCII	0 to 126*	0 (no character)
acknowledgment string	ASCII	0 to 127	0 (no character)

*Excluding 27.

Set Handshake Mode 2

Page 13-30

ESC.I [(data block size/Xoff threshold level);(enquiry character);(Xon trigger character/acknowledgment string)]:

Parameter	Format	Range	Default
Xoff threshold level (Xon-Xoff)	real	0 to 60 bytes	20 bytes without cartridge
or			80 bytes with cartridge
data block size (Enquire/acknowledge)	ASCII	0 to 80 bytes without cartridge	20 bytes
		0 to 1974 bytes with cartridge	80 bytes
enquiry character	ASCII	0 to 126*	0 (no character)
Xon trigger character(s) (Xon-Xoff)	ASCII	0 to 127	0 (no character)
or acknowledgment string (Enquire/acknowledge)	ASCII	0 to 127	0 (no character)

*Except 27.

Abort Device Control
Page 12-17

ESC.J

Abort Graphics
Page 12-17

ESC.K

Output Buffer Size When Empty
Page 12-11

ESC.L

Response: logical I/O buffer size [TERM] — 1 to 60 bytes without cartridge, 1 to 1024 bytes with cartridge (up to 1974 if buffer size altered using ESC . T).

Set Output Mode
Page 13-23

ESC.M [(turnaround delay) ; (output trigger character) ; (echo terminate character) ; (output terminator) ; (output initiator)] :

Parameter	Format	Range	Default
turnaround delay	real	0 to 255 without cartridge	0
		0 to 65 535 with cartridge	0
output trigger character	ASCII	0 to 126*	0
echo terminate character	ASCII	0 to 126*	0
output terminator(s)	ASCII	0 to 127**	13 (**CR**)
output initiator character	ASCII	0 to 127	0

*The values 5 and 27 are not allowed.
**The value 0 terminates the string.

Set Extended Output and Handshake Mode Page 13-27

ESC . N [(intercharacter delay) ; (Xoff trigger character/ immediate response string)] :

Parameter	Format	Range	Default
intercharacter delay	real	0 to 255 without cartridge 0 to 65 535 with cartridge	0
Xoff trigger character(s) (Xon-Xoff) or	ASCII	0 to 127	0 (no character)
immediate response string (Enquire/acknowledge)	ASCII	0 to 127	0 (no response)

Output Extended Status Page 12-15

ESC . O

Response: plotter operating status [TERM]

Set Handshake Mode Page 13-22

ESC . P (handshake method) :

Parameter	Format	Range	Default
handshake method	integer	0 to 3	hardwire

Reset Page 13-33

ESC . R

NOTE: Should be followed by ESC . L. ■

Output Configurable Memory Size Page 12-12

ESC.S (buffer):

Parameter	Format	Range	Default
buffer	real	0 to 2	0

Response: buffer size [TERM] — 0 to 1974 bytes.

Allocate Configurable Memory Page 12-5

ESC.T [(I/O buffer size);(polygon buffer size)]:

Parameter	Format	Range	Default
physical I/O buffer	real	2 to 1974 bytes	1024
polygon buffer	real	0 to 1972 bytes	950

Notes

Glossary

Absolute Plotting — Plotting to a point whose location is specified with respect to the fixed origin (0,0).

Acceleration — The rate at which the pen attains its maximum velocity (measured in g's or meters per second squared).

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.

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.

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: hardwire, Xon-Xoff, enquire-acknowledge, and software checking.

Hard-Clip Limits — That part of the plotting area beyond which the pen physically cannot move; the mechanical limits of the plotter.

Hatch — A fill type made of parallel lines.

HP-GL — Hewlett-Packard Graphics Language; the graphics instruction set Hewlett-Packard plotters understand.

HP-IB — Abbreviation for Hewlett-Packard's Interface Bus. Hewlett-Packard's version of IEEE Standard 488-1978 for interfacing programmable devices (e.g., computers, plotters, and printers).

IEEE 488-1978 Interface — A parallel interface standardized by EIA* Standard 488-1978.

Initialize — To set plotter conditions to known default values.

*Electronic Industries Association.

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.

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

Subject Index (Continued)

Subject Index (Continued)

Subject Index (Continued)

HEWLETT PACKARD

PART NO. 07440-90001
MICROFICHE NO. 07440-90051

AUGUST 198
PRINTED IN U.S.