

NP-10

USER'S MANUAL

NOT INTENDED FOR SALE

PN 80820191

Federal Communications Commission Radio Frequency Interference Statement

This equipment generates and uses radio frequency energy and if not installed and used properly, that is, in strict accordance with the manufacturer's instructions, may cause interference to radio and television reception. It has been type tested and found to comply with the limits for a Class B computing device in accordance with the specifications in Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference in a residential installation. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient the receiving antenna
- Relocate the computer with respect to the receiver
- Move the computer away from the receiver
- Plug the computer into a different outlet so that computer and receiver are on different branch circuits.

If necessary, the user should consult the dealer or an experienced radio/television technician for additional suggestions. The user may find the following booklet, prepared by the Federal Communications Commission helpful: "How to Identify and Resolve Radio-TV Interference Problems." This booklet is available from the U.S. Government Printing Office, Washington, D.C., 20402, Stock No. 004-000-00345-4.

For compliance with Federal Noise Interference Standard, this equipment requires a shielded cable.

This statement will be applied only for the printers marketed in U.S.A.

Self Declaration

Radio interferences regarding this equipment has been eliminated according to Vfg 1046/1984 announced by the DBP.

DBP has been informed about the introduction of this special equipment and has been conceded the right to examine the whole series.

It is upon the responsibility of the user to assume that his own assembled system is in accordance with the technical regulations under Vfg 1046/1984.

To observe FTZ-regulations it is necessary, to establish all connections to the printer with shielded cable.

The equipment may only be opened by qualified service representatives.

This statement will be applied only for the printers marketed in West Germany.

Trademark Acknowledgement

NP-10: Star Micronics Co., Ltd.

Apple II: Apple computer Inc.

IBM Personal Computer, IBM PC: International Business Machines Corp.

Microsoft BASIC: Microsoft Corporation

NOTICE

- All rights reserved. Reproduction of any part of this manual in any form whatsoever, without STAR's express permission is forbidden.
- The contents of this manual are subject to change without notice.
- All efforts have been made to ensure the accuracy of the contents of this manual at the time of going to press. However, should any errors be detected, STAR would be greatly appreciate being informed of them.
- The above notwithstanding, STAR can assume no responsibility for any errors in this manual.

©Copyright 1986 Star Micronics Co., Ltd.

Table of Contents

Chapter 1	Setting Up Your Printer	1
	Locating the printer	
	Check the carton contents	
	Removing the printer cover	
	Removing the packing tube	
	Installing the ribbon cartridge	
Chapter 2	Getting to Know Your Printer	7
	Controls and parts of the printer	
	Parts of the printer	
	Controls and indicators	
	Extra functions	
	Other controls	
	Selecting and loading paper	
	Loading single sheets	
	Loading sprocket-feed paper	
	Adjusting the print head	
	Testing the printer	
	Connecting the printer	
Chapter 3	Printing with BASIC	19
	Some basics of BASIC	
	First steps	
	ASCII codes and the CHR\$ function	
	Control codes	
	The escape codes	
	A note on command syntax	
	Some special kinds of text	
	Near Letter Quality characters	
	Italic printing	
	Underlining	
	Superscripts and subscripts	
	Changing the print pitch	
	Expanded print	
	Condensed print	
	Proportional printing	
	Making words stand out	
	Mixing print modes	

Chapter 4 Formatting Text 37

- Lines and line spacing
 - Starting a new line
 - Reverse line feeds
 - Changing the line spacing
 - Moving down the page without a carriage return
- Page control
 - Form feed
 - Reverse form feed
 - Changing the page length
- Top and bottom margins
- Setting left and right margins
- Horizontal and vertical tabs
 - Horizontal tabs
 - One-time horizontal tabs
 - Vertical tabs
 - Vertical tab channels
- Centering and aligning text

Chapter 5 Special Features of the Printer 59

- Now hear this
- Resetting the printer
- Taking the printer off line
- Printing the bottom of the sheet
- Backspace, delete, and cancel text
- Printing zeroes
- Immediate-print
- Adjusting the width of space between characters
- Uni-directional printing
- The seven bit dilemma
- Block graphics characters and special symbols
- International character sets
- Printing characters in the control code area
- Printing BIG characters
- The macro control code
- Reading a hex dump

Chapter 6	Download Characters and Dot Graphics	79
	Dot matrix printing	
	Designing your own characters	
	Defining proportional characters	
	Designing your own NLQ characters	
	User-designed dot graphics	
	Higher resolution graphics	
	More graphics programming tips	
	Redefining alternate graphics codes	
	9-pin graphics mode	
Chapter 7	Caring for Your Printer	95
	Cleaning the printer	
	Replacing the ribbon	
	Replacing the print head	
Appendix A	DIP Switch Settings	101
	Switch functions	
Appendix B	ASCII Code Conversion Chart	105
Appendix C	Character Code Table	107
Appendix D	Function Codes	115
	Commands to control print style	
	Font style controls	
	Font pitch controls	
	Special print modes	
	Controlling the vertical print position	
	Line feed and reverse line feed controls	
	Form feed and related commands	
	Top/bottom margins and vertical tabs	
	Controlling the horizontal print position	
	Download character commands	
	Dot graphics commands	
	Macro instruction commands	
	Other commands	
Appendix E	Command Summary in Numeric Order	161

Appendix F	Technical Specifications	165
Appendix G	The Parallel Interface	169
	Functions of the Connector Signals	
Index		173

CHAPTER 1

SETTING UP YOUR

PRINTER

Subjects covered in Chapter 1 include—

- **Choosing a suitable place for your printer**
- **Unpacking**
- **Setting up**

LOCATING THE PRINTER

Give some thought to the best place to put the printer. Both the printer and computer should be used in normal home surroundings. For best performance, we recommend:

- Use the printer on a flat surface.
- Keep it out of direct sunlight and away from heat-producing appliances.
- Use it only in temperatures where you are comfortable.
- Avoid locations with dust, grease, or high humidity.
- Supply it “clean” electricity. Don’t connect it to the same circuit used by large, noise-producing appliances (such as refrigerators).
- Make sure the line voltage is within 10% of the voltage specified on the identification plate.

CHECK THE CARTON CONTENTS

Open the carton and check each item in the box against Figure 1-1 to make sure that you have everything (there should be four items).

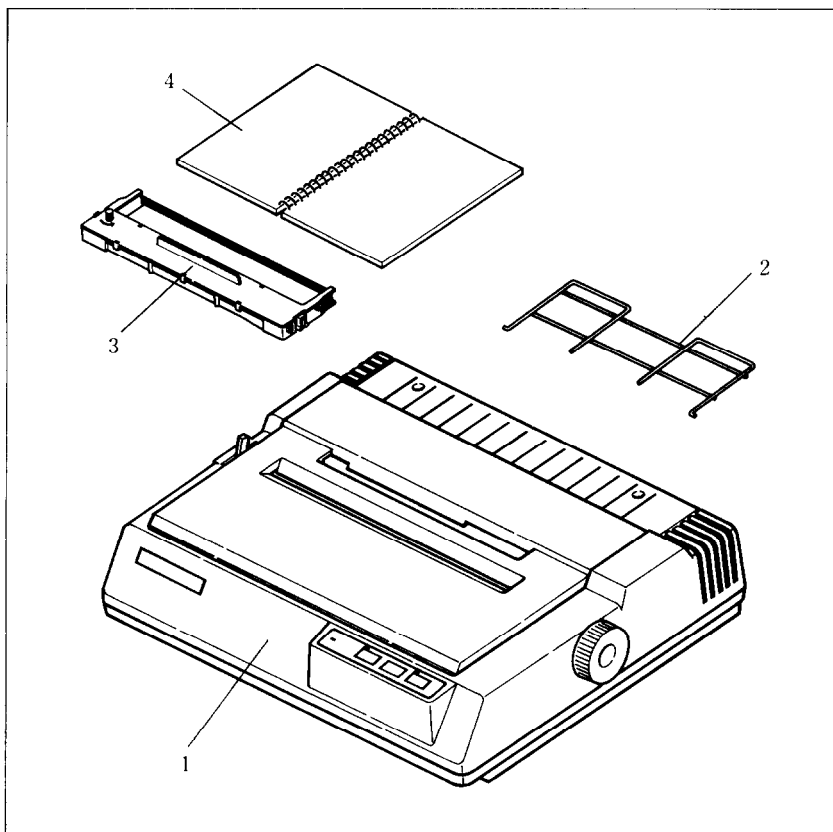


Figure 1-1. Check to make sure you have all four items: 1) Printer, 2) Paper separator, 3) Ribbon cartridge, and 4) User's manual.

■ Removing the printer cover

The printer's cover is important for two reasons — it keeps dust and dirt away from the printer's delicate mechanism, and it absorbs nearly all of the printer's operating sounds. Don't take off the cover unless you have to change the ribbon or make an adjustment.

To remove the cover, lift up the back of the cover to disengage the two tabs at the front, then lift it off the rest of the way. To replace the cover, slide the tabs in at the front and lower it into place. Figure 1-2 shows the proper position and movement for both removing and replacing the cover.

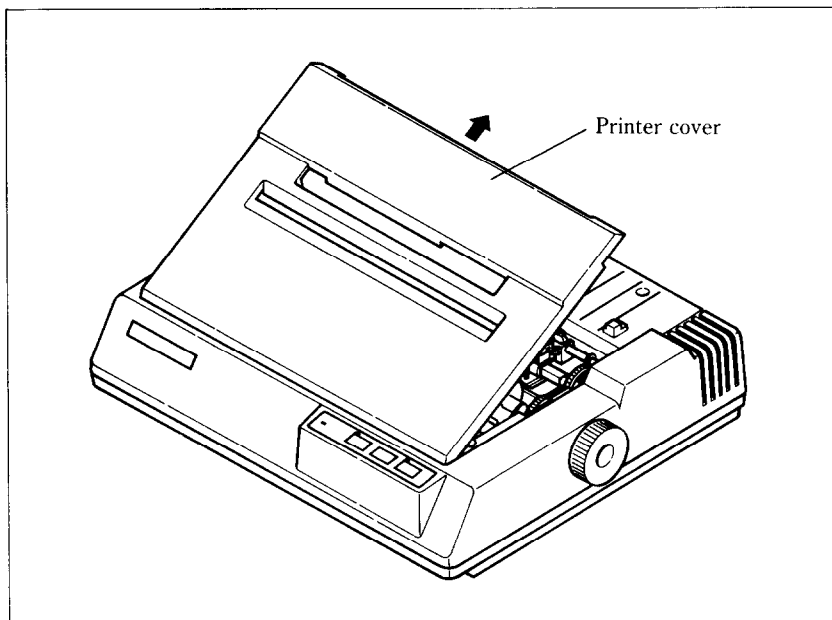


Figure 1-2. Remove the printer cover by lifting carefully.

■ Removing the packing tube

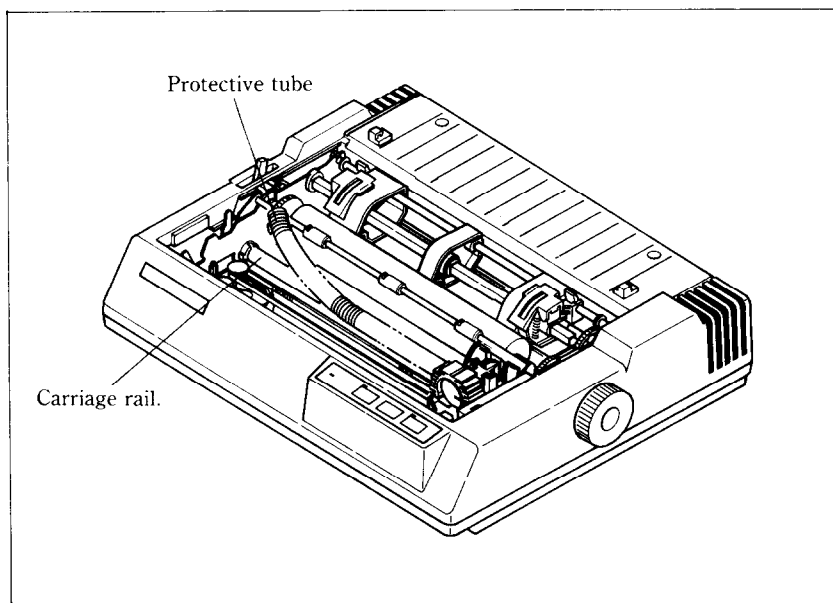


Figure 1-3. Remove the protective tube from the carriage rail.

The printer is shipped with a protective spiral tube to keep the print head from being damaged during shipping. This tube must be removed before use. First, take off the printer cover. The protective tube is split along its bottom side to allow easy removal. Grasp one end of the spiral tube and firmly pull it upward and away from the carriage rail.

■ Installing the ribbon cartridge

This printer uses a neat, easy-to-change ribbon cartridge with automatic threading. To fit or change the ribbon cartridge:

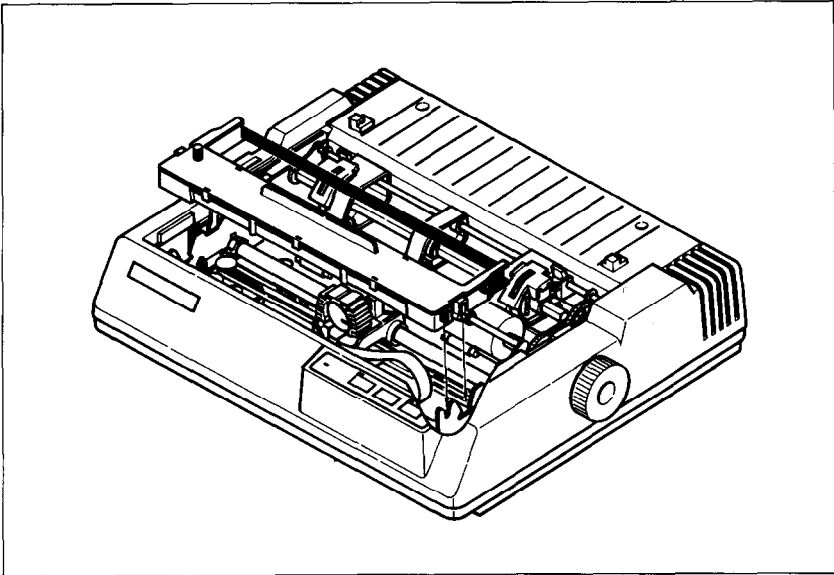


Figure 1-4. Press the cartridge into place until the holding springs snap into place.

1. Turn *off* the power and remove the printer cover.
2. Slide the print head gently to the center of the printer.
Warning: The print head gets hot during operation, so let it cool off before you touch it.
3. Set the release lever to either single sheet (■) or sprocket-feed paper (■).

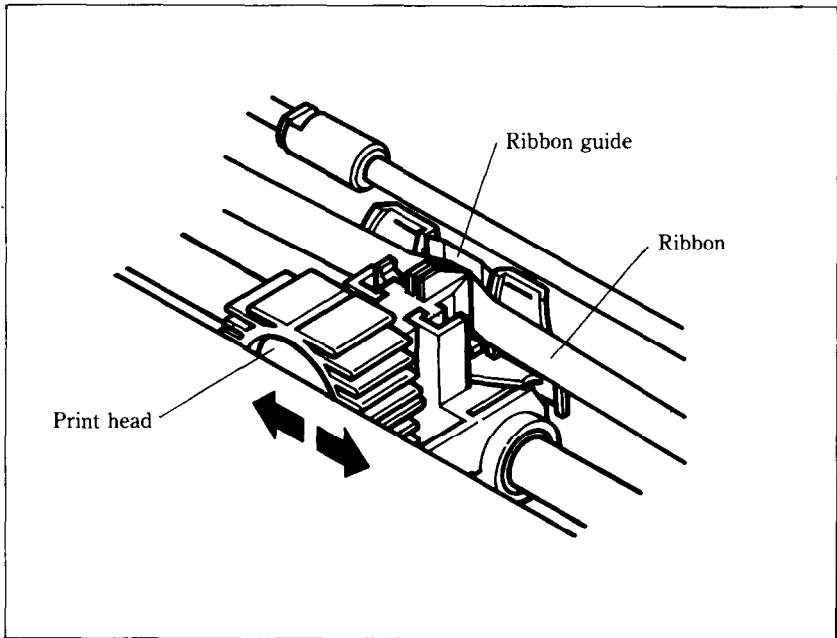


Figure 1-5. To thread the ribbon, slide the print head carriage manually to the right or left, and the ribbon will slip down by itself into its proper position.

4. With the ribbon facing away from you, use the guide holders as a fulcrum, and lightly press the cartridge down until the two holder springs snap shut to hold the cartridge firmly in place. See Figure 1-4.
5. Check that the cartridge fits so that the drive pins engage the cartridge teeth, by turning the ribbon advance knob. Advance the ribbon to take up any slack.
6. Gently slide the print head carriage manually all the way to your right or left until the ribbon automatically slips down into its proper place between the print head and the silver ribbon guide.
7. Replace the printer cover.

MEMO

CHAPTER 2

GETTING TO KNOW YOUR PRINTER

Subjects covered in Chapter 2 include—

- **Parts of the printer — what they're for and how to use them**
- **Paper selection and loading**
- **Adjustment**
- **Testing printer operation**

CONTROLS AND PARTS OF THE PRINTER

■ **Parts of the printer**

PRINTER COVER — This protects the ribbon and the print head from dust and dirt, and cuts down the sound of the printer.

PAPER SEPARATOR — This metal guide is used with sprocket-feed paper.

POWER CORD — This cord connects the printer to its power source, usually a wall outlet. It's located at the left rear of the printer.

PRINT HEAD — This is the device that does the actual printing. Like the strike lever in a typewriter, tiny pins in the print head hit the paper through a ribbon.

TRACTOR FEED UNIT — The drive gear and sprockets of the tractor feed unit move sprocket-feed paper through the printer.

PLATEN — This is the rubber cylinder that carries paper to the print head.

INTERFACE CONNECTOR — On the back of the printer, this connector is used to connect your computer to the printer.

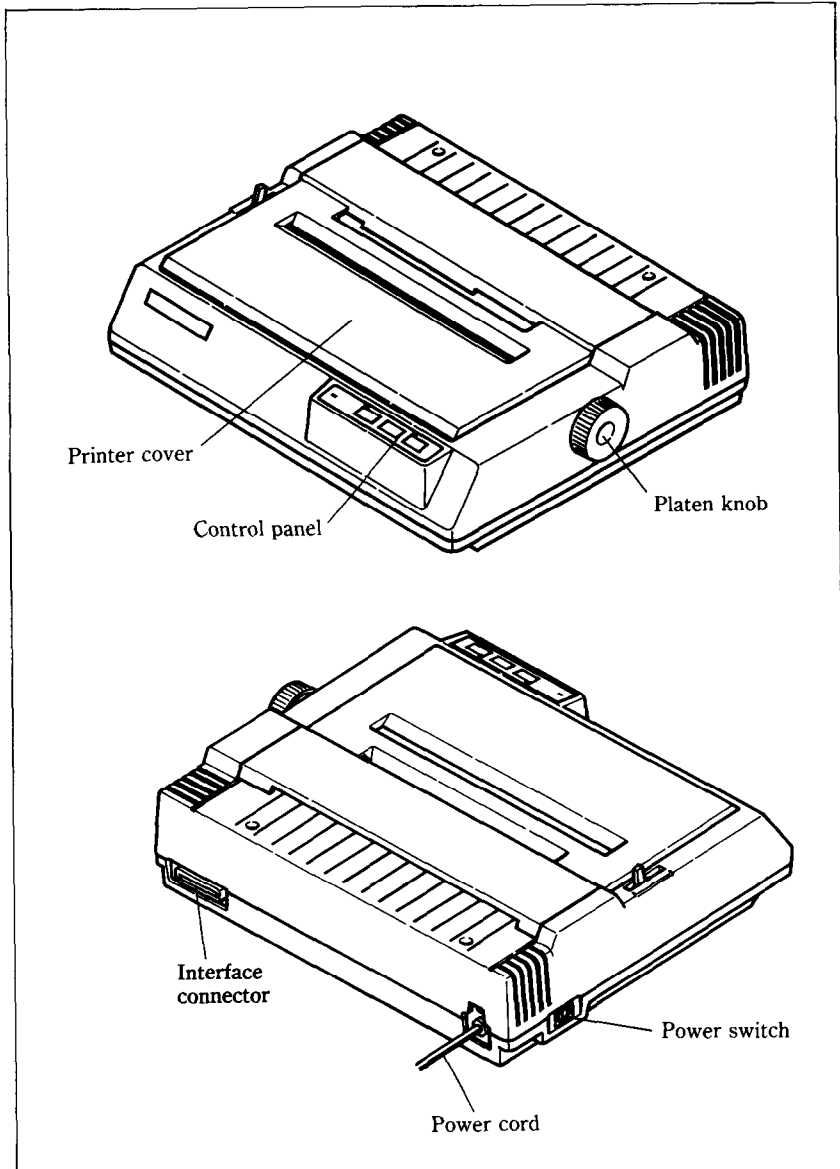


Figure 2-1. Front and rear views of the printer.

■ Controls and indicators

Now let's go over the controls and indicators of the printer, starting with the control panel on the right. There are three indicators and three keys on the control panel.

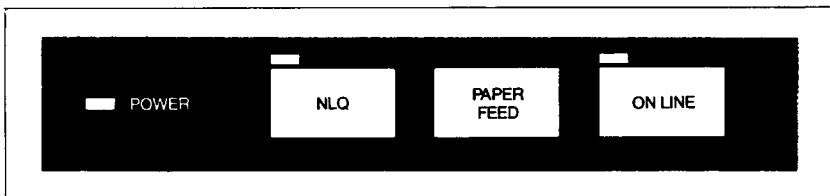


Figure 2-2. The control panel.

POWER INDICATOR – Glows orange when the power is *on*. This indicator flickers when the printer runs out of paper and when certain printer malfunctions occur.

NLQ KEY and INDICATOR – Changes the character style (draft or NLQ) when the On Line indicator is off, and glows green when the NLQ mode is selected.

PAPER FEED KEY – Advances the paper one line at a time when the On Line indicator is off. If you hold the key down, you get consecutive line feeds. If you push the On Line key while holding this key, you can advance the paper to the top of the next page or a new form.

ON LINE KEY and INDICATOR – Glows green when the printer is ready to receive data from your computer (on line). When the printer is off line, it sends a signal to the computer indicating that it cannot accept data. Pressing the On Line key switches the printer between on line and off line.

■ Extra functions

There are seven more functions that are not directly specified on the control panel. You can access four of them by pressing one particular key when you turn on the power; three others are activated by pressing two control keys at once.

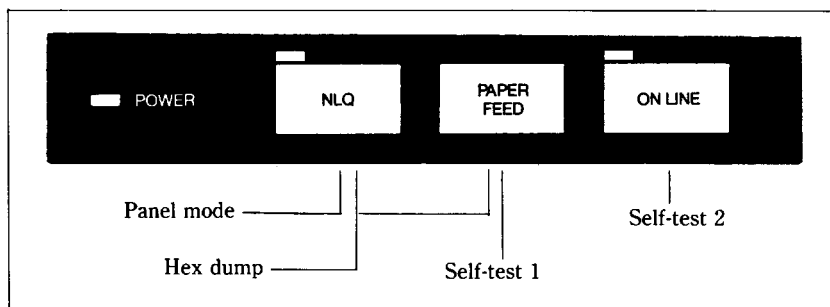


Figure 2-3. Extra functions while turning on the printer.

We'll explain about the "Panel" mode, which is powerful function of the printer. Some commercial software does not support defining your own selections. In this case, use the "Panel" mode.

Selection of the Panel Mode at power-on ensures that your choices remain in effect until you turn the power off. This means that the printer will ignore any codes sent by the software that you are using. The Panel Mode allows selection of the character quality (Draft or NLQ).

If you want to set the Panel mode, press the NLQ key on the control panel when you turn on the power. After a second, you will hear a beep, and the ON LINE indicator will glow. If you want to change the character, press the ON LINE key, press the NLQ key once again to make your choice, and press the ON LINE key again.

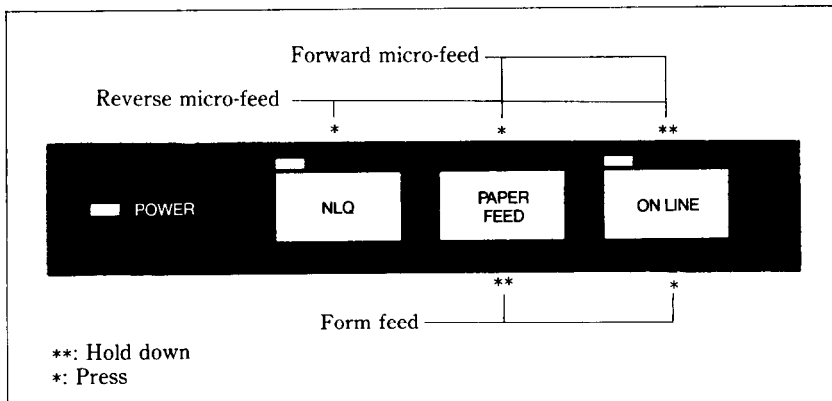


Figure 2-4. You can set many functions by the combinations of the control panel keys while in the Off Line mode.

When you want to align the print start position, you can set it by the micro-feed operation with the control panel, instead of turning the platen knob manually.

1. Set the Off Line mode by pressing the On Line key.
2. While holding down the On Line key, press one of the following keys.

Paper Feed key — Forward micro-feed

NLQ key — Reverse micro-feed

3. When you can set the print start position, release the Paper Feed key or the NLQ key first, then release the On Line key.

■ Other controls

There are other controls, not connected to the control panel. Some of the more important ones are:

POWER SWITCH — The switch that turns the printer on and off is at the back, on the left side.

PLATEN KNOB — This knob is at the middle of the right side and lets you turn the platen by hand, like a typewriter.

Caution: Turn the platen knob only with the Power switch *off*. Turning the knob with the power on could damage the drive gears.

RELEASE LEVER — The release lever is on top of the printer, near the left rear corner. It controls how strongly the paper is held against the platen. The release lever has four positions: the top setting is used for inserting paper, the next is for single sheets, the third is for sprocket-feed paper, and the bottom one is used when adjusting the paper.





PAPER BAIL — The bail is the movable bar that holds the paper against the platen. Its position depends on the setting of the release lever.

DIP SWITCHES — There is one set of eight switches that is used in interfacing the printer to your particular computer. There are also switches to set page length, and international character set. See Appendix A for a complete list and explanation.

SELECTING AND LOADING PAPER

Your printer handles two basic types of paper — single sheets (stationery, envelopes, multipart business forms, etc.) and continuous paper (fan-fold perforated paper).

The release lever controls the pressure of the paper against the platen. It has four settings:

-  — The top setting is used when you want to load paper in the printer.
-  — The second setting is for normal single sheets.
-  — The third is for sprocket-feed paper.
-  — The bottom one is used when you want to release the paper completely to adjust it.

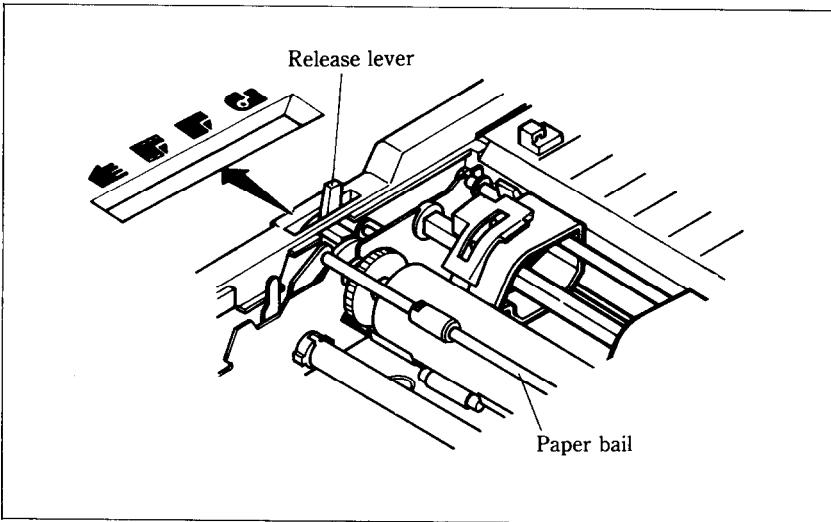


Figure 2-5. Use friction feed for single sheets and tractor feed for continuous paper.

The paper bail holds the paper against the platen according to the setting of the release lever. The bail is opened when the release lever is in the top position, closed when the lever is in the second or third positions, and opened by the printer when the lever is in the bottom position.

■ Loading single sheets

Instead of feeding the paper in *manually* by turning the platen knob, you can use the release lever for automatic feeding (the printer must be turned *on*). Single sheets from 5.5 to 8.5 inches wide can be used:

1. Flip the clamp Levers backward and move the sprockets on the tractor feed unit all the way to the ends.

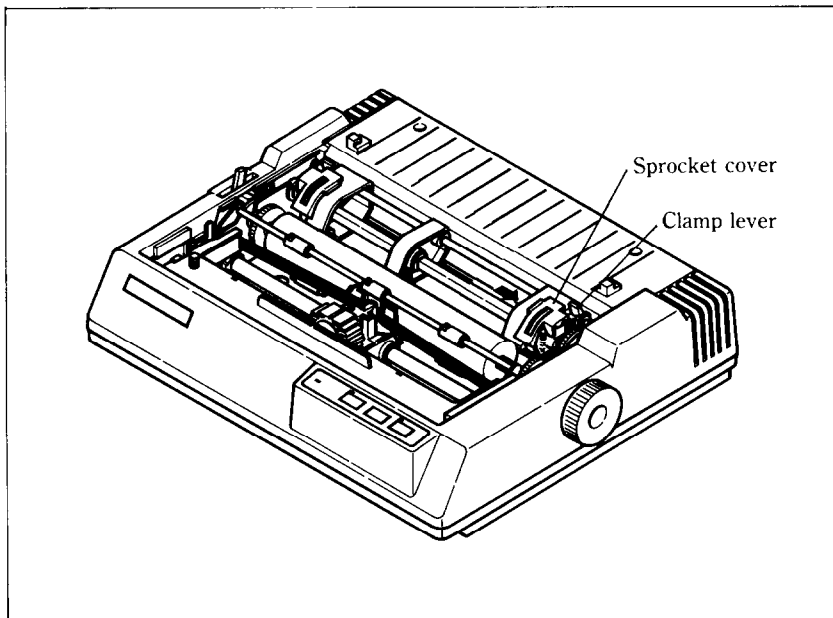


Figure 2-6. Move the sprockets on the tractor feed unit all the way to the ends.

2. Install the printer cover on the printer.
3. Set the release lever for single sheets — the second position from the top.
4. Insert a sheet of paper to the paper chute on the printer cover.
5. Turn on the power switch. (The Power indicator will blink because there is no paper.)
6. Now set the release lever to the auto-feed setting — the top position. The printer will advance the paper automatically.
7. When the paper stops, set the release lever for single sheets again.

To align paper that is not in straight —

- Set the release lever to the adjustment setting.
- Straighten the paper and adjust it for the margins you want.
- Move the release lever back to its original setting.

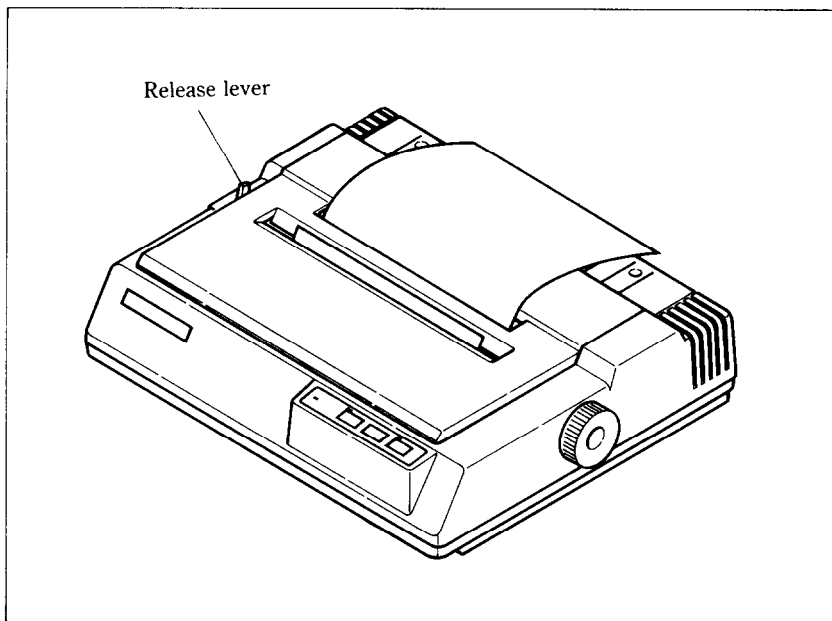


Figure 2-7. You can insert a sheet of paper automatically by using the release lever.

■ Loading sprocket-feed paper

This is standard computer paper, with the holes along the sides and perforations between the sheets. It is also called sprocket, punched, fan-fold, or just plain "computer paper". It can be as narrow as 4.5 inches, and up to 10 inches wide.

1. Remove the printer cover.
2. Place a stack of fan-fold paper behind the printer.
3. Turn on the printer and set the release lever for sprocket-feed paper.
4. Open the sprocket covers, on the right and left sprocket units, as shown in Figure 2-8.
5. Flip the clamp levers backward. This allows the two sprocket units to move freely right and left so you can align them with the holes in the paper.
6. Feed the end of the paper into the slot next to the platen cover plate.
7. Fit the holes in the paper over the sprocket pins so the paper is even, and clamp the sprocket units in place.

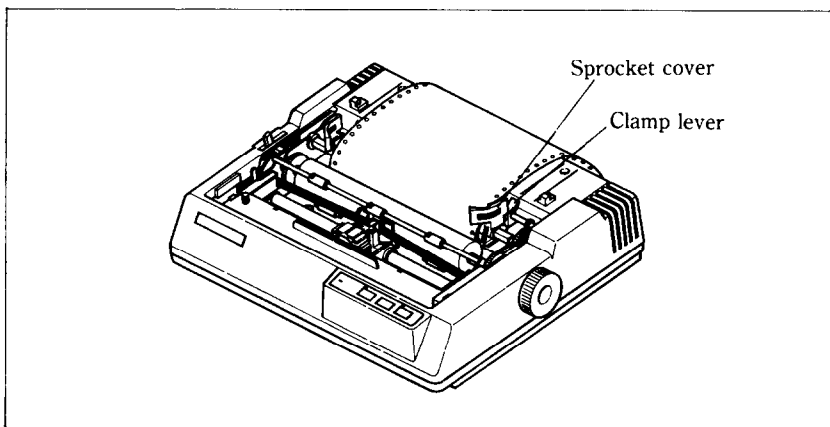


Figure 2-8. Open the sprocket covers to expose the sprocket teeth.

8. Check that the paper is still positioned correctly, and close the sprocket covers.
9. Set the release lever to the auto-feed setting. The printer will advance the paper automatically.
10. When the paper stops, set the release lever for sprocket-feed paper again.
11. Install the paper separator on the printer, as shown in Figure 2-9. Then, install the printer cover.

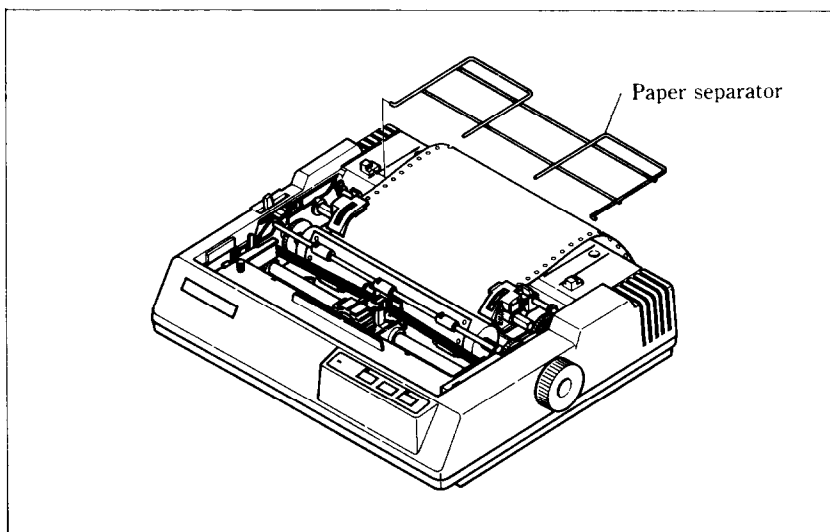


Figure 2-9. Ready to load the sprocket-feed paper.

ADJUSTING THE PRINT HEAD

The distance between the print head and the platen must be adjusted to accommodate papers of different thicknesses. To make this adjustment, move the adjustment lever, which is under the printer cover and immediately in front of the release lever (Figure 2-10). Pulling the adjustment lever towards you will widen the gap; pushing it away from you will narrow the gap.

There are four positions; you can feel the lever clicking into the various notches. The first step (shown) is the one most commonly used for single sheets of paper.

Try different positions until you get the best printing results.

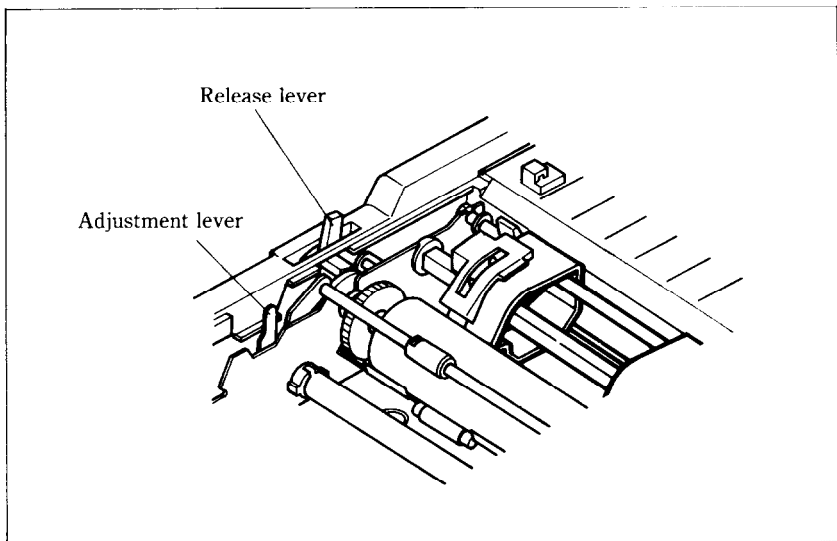


Figure 2-10. The adjustment lever allows for different thicknesses of paper.

TESTING THE PRINTER

Your printer has two built-in programs that printout sample lines of letters, numbers, and other characters — to show you that everything's in good working order. Self-test 1 is the long

CONNECTING THE PRINTER

Now that you've assembled and tested your printer, all that remains is to connect it to your computer as follows:

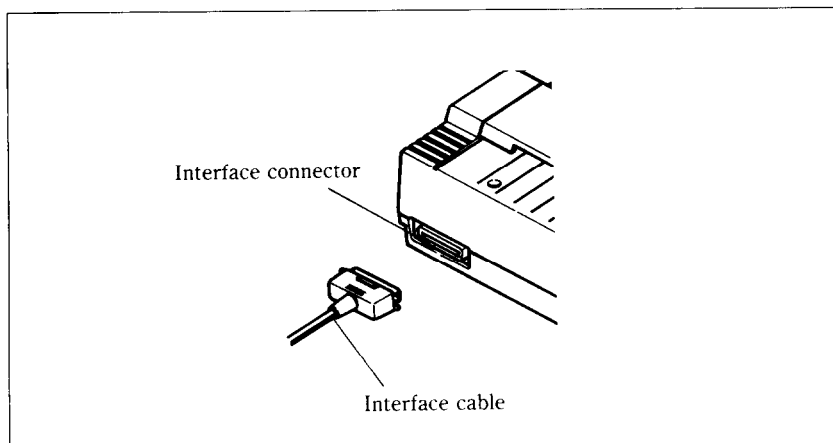


Figure 2-12. Connect the cable like this.

1. Make sure that both your computer and printer are turned off.
2. Connect the printer end of the interface cable to the connector socket at the right rear of the printer as shown in Figure 2-12.
3. Connect the other end of the cable to your computer as described in the computer manual.

CHAPTER 3

PRINTING WITH BASIC

Subjects covered in Chapter 3 include—

- **Listing BASIC programs on the printer**
- **How a program prints things**
- **Control codes, escape codes, and command syntax**
- **Near letter quality (NLQ) characters**
- **Fixed and proportional character spacing**
- **Special printing—**
 - Printing in italics**
 - Underlining**
 - Superscripts and subscripts**
 - Boldface and emphasized text**
 - Mixing print modes**

To show you how to control your printer from a program, we have chosen BASIC because it is easy to learn and easy to use. Also, more personal computer users program in BASIC than in any other language.

The rest of this manual will show you a little BASIC — just enough for you to use your printer. We're not going to try to make you an expert programmer, though, only get you started. There are many excellent books that will teach you BASIC, so if you discover that you like to program you should have no trouble learning more about it.

SOME BASICS OF BASIC

■ First steps

The first things that a beginner learns to do are to list a program and to print a character string. Certainly these are the easiest operations one can do, but even they may depend on what computer you have. In Microsoft BASIC, we can list all the steps in a program by entering LIST. This lists them on the CRT screen; if we want to print them on a printer, we prefix the command with an L (enter LLIST).

The Microsoft BASIC command for outputting information is PRINT. Like the LIST command, this displays the information on the CRT screen so we have to add an L (\rightarrow LPRINT) if we want to use the printer. Just put whatever you want to print between quotes and after LPRINT (anything enclosed in quotes is called a character string). For example, we would use LPRINT "Hello!" to output "Hello!" to the printer. We'll see later how to LPRINT more than just character strings.

We started with Microsoft BASIC because it is the most widely used version of BASIC. The programs in this manual are written in Microsoft BASIC so they should run on most computers. But if strange things happen when you try to run a program, check the BASIC manual that came with your computer.

Let's consider Apple II computers for a minute. These popular computers use their own brand of BASIC. To use an Apple II, enter the following—

```
PR#1          PR#1
LIST          PRINT "Hello!"
PR#0          PR#0
```

The PR#1 tells the Apple to send everything to the printer, the LIST or PRINT command sends it, and the PR#0 returns output to the screen.

Now that we know how to address the printer, let's try listing a BASIC program. We will load a program into memory ready to program printer operation — just as soon as we learn a little bit about the ASCII codes.

■ ASCII codes and the CHR\$ function

You can talk to your computer in BASIC, but your computer and your printer talk to each other in what are known as ASCII codes. In the ASCII code, each number from 0 to 255 has a particular meaning — 36, for example, makes the printer print a dollar sign. Some numbers cause the printer to do other things, too. For instance, sending a 7 sounds the printer's bell.

Taken together, these numbers and their meanings make up the ASCII code (pronounced *ask-key*), which stands for the *American Standard Code for Information Interchange*. There are ASCII codes for all the letters of the alphabet (upper case and lower case), 0 to 9, most punctuation marks, and some (but not all) of the functions of the printer.

There are a number of different ways to represent an ASCII code, depending on how you are using it. For example, the ASCII codes for the upper-case letter "A" are 65 (decimal) or &H41 (hexadecimal). Or you can just call it "A". Appendix B shows all of the ASCII codes.

BASIC uses the CHR\$ function to represent ASCII characters and many functions. To print the letter "A" we would enter LPRINT CHR\$(65). To make the printer's bell sound, we would enter LPRINT CHR\$(7). In general, we print a character by entering LPRINT CHR\$(*ASCII code*) to the printer.

We can also use hex ASCII codes. Although we use only decimal ASCII codes in this manual, in some applications it will help if you understand what a hex code is. "Hex" is short for hexadecimal and refers to a base-16 number (the numbers we use in everyday life are base 10). Since the hex system needs 16 digits, it uses the numerals 0 through 9 and also the letters A through F. In BASIC programs, you can always tell that a number is in hexadecimal by the "&H" immediately preceding it. The ASCII code for the letter "A" (65 in decimal) is &H41 in hex.

■ Control codes

ASCII codes with values of 32 or less do not correspond to keys on the keyboard. These codes control many of the printer's functions, so we call them control codes. To enter a control code from the keyboard, we have to press two keys at the same time

— the “control” key and one other. The other key determines what code is sent — pressing the control (CTRL) key and A sends ASCII code 1, CTRL B sends ASCII code 2, and so on.

Your printer has a lot of control codes to let you do some useful things. Let’s try one that we’ve mentioned several times already:

```
10 ' Demo of ASCII code
20 LPRINT CHR$(7)
30 END
RUN
```

That’s the printer’s bell (we call it that even though it sounds like a buzzer). We’ll learn more about it in later — we just wanted to show you a control code that would get your attention right away!

There are four common ways of referring to a control code: the name of the code or its abbreviation, the decimal ASCII value, the hexadecimal ASCII value, and the “CTRL-” value. For example, the ASCII code that causes the printer to advance the paper one line is decimal 10. This code may be referred to by any of the following:

line feed	— the name of the code
<LF>	— its abbreviation
ASCII 10	— its decimal value
ASCII &H0A	— its hexadecimal value (the &H signifies hex)
CHR\$(10)	— the way it’s used in BASIC
CTRL-J	— the way you send it from a keyboard

Of course, most of the time we don’t need to bother with these. The computer is smart enough to know that when we press the “A” key we want to print the letter “A” — it takes care of all the intermediate steps.

Appendix B is a table that shows the various names for each code so you can convert back and forth. The microcomputer world is not very consistent in describing ASCII codes, so it’s important that you have a basic knowledge of them.

■ The escape codes

Back when the ASCII system was set up, computer equipment was relatively simple and thirty-three control codes were considered sufficient at the time. The American Standards people realized that eventually more control codes would be needed, so they included the escape (ESC) code to allow almost any number of additional codes to be defined when they became necessary.

ESC allows us to “escape” from the ordinary set of control codes so we can specify additional functions and other information needed for a printer function. In this manual, we’ll write the ESC code inside broken brackets, like this — `<ESC>`.

`<ESC>` — decimal 27 — is always followed by at least one other number; it is never used alone. The whole series of related numbers is called an escape sequence.

■ A note on command syntax

Because the readers of this manual will be running such a wide variety of applications on so many different computers, we can’t show the exact way of sending codes to the printer for each one of them. Instead, as we introduce you to each new command, we will show the commands as in this example:

```
<ESC> “W” 1
```

This command turns on expanded printing. `<ESC>`, as we mentioned earlier, is the escape code (which is ASCII code 27). A letter or number in quotes (such as the “W” above) means that the character should be sent to the printer, but without the quotes. In our example, you should send a capital W following the escape code. In BASIC, you could do this in a couple of ways: by sending the character itself (e.g. `LPRINT “W”;`), or by using the `CHR$` function to send the ASCII code for the character (e.g. `LPRINT CHR$(87);`).

Many of printer commands end with a 1 or 0. When shown as in the above example (i.e., no quotes and no “`CHR$`”), you can use either ASCII code 1 (i.e., `CHR$(1)`) or the character “1” (which is ASCII code 49). The same principle applies to commands ending with 0.

So for our example above, any of these BASIC statements will have the same result:

```
LPRINT CHR$(27);CHR$(87);CHR$(1)
LPRINT CHR$(27);CHR$(87);CHR$(49)
LPRINT CHR$(27);"w";CHR$(1)
LPRINT CHR$(27);"w";CHR$(49)
LPRINT CHR$(27);"w1"
```

Even though there are many commands that require the use of ASCII code 0 (the $\langle \text{NUL} \rangle$ character), the number "0" (zero or ASCII code 48) cannot be substituted. In these cases, instead of an unadorned 0 we will show CHR\$(0) each time these commands are referenced.

That's it for the basics. You are now ready to learn how to use the many features of your printer.

SOME SPECIAL KINDS OF TEXT

If you looked carefully at your printer's self test, you probably noticed that it can print in italics. But that's not all! Your printer can underline characters, print superscripts and subscripts, and perhaps most exciting, print near letter quality characters.

■ Near Letter Quality characters

This printer's Near Letter Quality (sometimes abbreviated as NLQ) character set is ideal for correspondence and other important printing, as it produces very high quality characters. Normally, your printer prints draft quality characters. This is adequate for most work and has the fastest printing speed. For the final printout, try NLQ. The program below shows an example.

```
10 ' Demo of NLQ character set
20 LPRINT CHR$(27);"x1";
30 LPRINT "This line shows NEAR LETTER QUALITY!"
40 LPRINT CHR$(27);"x0";
50 LPRINT "This line shows standard print."
```

In this program, line 20 selects NLQ characters with `<ESC> "x" 1` command. Line 30 prints a sample before line 40 switches printer back to draft printing with an `<ESC> "x" 0`. When you run the program you should get this:

```
This line shows NEAR LETTER QUALITY!
This line shows standard print.
```

Now turn off the power switch. After you turn on the power switch while holding the NLQ key down on the control panel, set the NLQ mode with the NLQ key while in the Off Line mode. Then run the program again. You should get like this:

```
This line shows NEAR LETTER QUALITY!
This line shows standard print.
```

As you can see, now the printout only contains NLQ characters. This is because we set the printer to the "Panel" mode at power on, so the printer ignored the control codes to set or cancel NLQ characters. This is a very useful way to print NLQ characters if you are using a packaged software which does not support the NLQ character mode.

Table 3-1
Near letter quality commands

Function	Control code
Near letter quality ON	<code><ESC> "x" 1</code>
Near letter quality OFF	<code><ESC> "x" 0</code>

■ Italic printing

Italic letters are letters that are slanted to the right. Your printer can print all of draft characters in *italic* as well as the roman (standard) letters you are accustomed to. Italics can be used to give extra emphasis to certain words. The commands to turn italic on and off are shown in Table 3-2.

Table 3-2
Italic commands

Function	Control code
Italic ON	<ESC> "4"
Italic OFF	<ESC> "5"

Use this program to see italic characters:

```
10 ' Demo of italic and roman
20 LPRINT CHR$(27);"4";
30 LPRINT "This line is in ITALIC characters."
40 LPRINT CHR$(27);"5";
50 LPRINT "This line is in ROMAN characters."
```

Here is what you should get:

```
This line is in ITALIC characters.
This line is in ROMAN characters.
```

In this program, line 20 turns italic on with <ESC> "4", and line 40 turns italic off with <ESC> "5".

■ Underlining

Not only can your printer print all styles of printing in both roman and italic, but it can underline them too. The control codes are shown in Table 3-3.

Table 3-3
Underline commands

Function	Control code
Underline ON	<ESC> "-" 1
Underline OFF	<ESC> "-" 0

It's that simple. Let's try it with this program:

```
10 ' Demo of underlining
20 LPRINT CHR$(27);"-1";
30 LPRINT "This phrase is UNDERLINED;" ;
40 LPRINT CHR$(27);"-0";
50 LPRINT " this is not."
```


It should come out like this:

This phrase is UNDERLINED; this is not.

In this program underline is turned on in line 20 with $\langle \text{ESC} \rangle$ “-” 1, and then off in line 40 with $\langle \text{ESC} \rangle$ “-” 0. There’s a new aspect to this program, though. The semicolons at the end of the first three lines told BASIC that those lines were to be continued. Therefore, BASIC didn’t send a carriage return and line feed at the end of those lines. We just did this to illustrate that all these control codes can be used in the middle of a line. It’s easy to underline or *italicize* only part of a line.

■ Superscripts and subscripts

Your printer can print in two different heights of characters. The smaller characters are called *superscripts* and *subscripts* and are half the height of normal characters. *Superscripts* print evenly with the tops of regular printing while *subscripts* print evenly with the bottom of regular printing. They are frequently used to reference footnotes, and in mathematical formulas.

Table 3-4 has the codes for using superscripts and subscripts.

Table 3-4
Superscripts and subscripts commands

Function	Control code
Superscript ON	$\langle \text{ESC} \rangle$ “S” 0
Subscript ON	$\langle \text{ESC} \rangle$ “S” 1
Super and subscript OFF	$\langle \text{ESC} \rangle$ “T”

Try this program to see them work:

```

10 ' Demo of superscripts and subscripts
20 LPRINT "Look! ";
30 LPRINT CHR$(27);"S0";
40 LPRINT "SUPERSCRIPTS ";
50 LPRINT CHR$(27);"T";
60 LPRINT "& ";
70 LPRINT CHR$(27);"S1";
80 LPRINT "SUBSCRIPTS ";

```

```
90 LPRINT CHR$(27);"T";
100 LPRINT "on one line."
```

Look! **SUPERSCRIPTS** & **SUBSCRIPTS** on one line.

Here line 30 turns on superscripts with $\langle \text{ESC} \rangle$ "S" 0. It's turned off in line 50 with $\langle \text{ESC} \rangle$ "T". Then between printing text, subscripts are turned on in line 70 with $\langle \text{ESC} \rangle$ "S" 1, and finally off in line 90. Again, everything prints on one line because of the semicolons.

CHANGING THE PRINT PITCH

In "printer talk," the number of characters that can be printed in one inch is called the print pitch or character pitch. Normally, your printer is set for 10 characters per inch, which is called *pica* (and is the same as the pica pitch on most typewriters). This works out to 80 characters per line.

You can also print 12 character per inch (*elite pitch*). This gives you 96 characters per line.

You can set these pitches by software as shown in the table below.

Table 3-5
Print pitch commands

Pitch	Characters/inch	Control code
Pica	10	$\langle \text{ESC} \rangle$ "P"
Elite	12	$\langle \text{ESC} \rangle$ "M"

Try this program to see how these two pitches work. Be sure to set the printer to draft mode.

```
10 ' Demo of pica and elite pitches
20 LPRINT CHR$(27);"M";
30 LPRINT "This line is ELITE pitch."
40 LPRINT CHR$(27);"P";
50 LPRINT "This line is PICA pitch (normal)."
```

When you run this program you should get this:

```
This line is ELITE pitch.  
This line is PICA pitch (normal).
```

Line 20 turns on elite pitch with `<ESC> "M"`. Line 30 prints the line at 12 characters per inch. The `<ESC> "P"` in line 40 resets the printer to pica pitch and line 50 prints the line in pica pitch.

■ Expanded print

Each of the print pitches can be enlarged to twice its normal width. This is called expanded print. Try this program to see how it works:

```
10 ' Demo of expanded print  
20 LPRINT "Demonstration of ";  
30 LPRINT CHR$(14);  
40 LPRINT "EXPANDED";  
50 LPRINT CHR$(20);  
60 LPRINT " printing."  
70 LPRINT "Notice that ";  
80 LPRINT CHR$(14);  
90 LPRINT "EXPANDED mode"  
100 LPRINT "automatically turns off at the end of  
    a line."
```

```
Demonstration of EXPANDED printing.  
Notice that EXPANDED mode  
automatically turns off at the end of a line.
```

Expanded print set with `CHR$(14)` is automatically cancelled at the end of the line. This is convenient in many applications, such as for one line titles. Note that you didn't need to put an `<ESC>` in front of the `CHR$(14)`, although `<ESC> CHR$(14)` works just the same.

You can also cancel one line expanded print *before* a carriage return with `CHR$(20)`, as done in line 50.

Sometimes you may wish to stay in expanded print for more than one line. Change your program to this:

```
10 ' Demo of permanent expanded mode
20 LPRINT CHR$(27);"w1";
30 LPRINT "Permanent expanded"
40 LPRINT "mode stays on until"
50 LPRINT "it is ";
60 LPRINT CHR$(27);"w0";
70 LPRINT "turned off."
```

Now the results look like this:

```
Permanent expanded
mode stays on until
it is turned off.
```

When you turn on expanded print with $\langle \text{ESC} \rangle$ "W" 1 it stays on until you turn it off with $\langle \text{ESC} \rangle$ "W" 0.

Table 3-6
Expanded print commands

Function	Control code
One line expanded ON	CHR\$(14) or $\langle \text{ESC} \rangle$ CHR\$(14)
One line expanded OFF	CHR\$(20)
Expanded ON	$\langle \text{ESC} \rangle$ "W" 1
Expanded OFF	$\langle \text{ESC} \rangle$ "W" 0

■ Condensed print

Each of the print pitches also can be condensed to its normal width. This is called condensed print. Try this program to see how it works:

```
10 ' Demo of condensed print
20 LPRINT "Demonstration of ";
30 LPRINT CHR$(15);
40 LPRINT "CONDENSED";
50 LPRINT CHR$(18);
60 LPRINT " printing."
```

Demonstration of CONDENSED printing.

Condensed print set with CHR\$(15) stays on until you turn it off with CHR\$(18). Note that you don't need to put an <ESC> in front of the CHR\$(15), although <ESC> CHR\$(15) works just the same.

Table 3-7
Condensed print commands

Function	Control code
Condensed ON	CHR\$(15) or <ESC> CHR\$(15)
Condensed OFF	CHR\$(18)

By combining expanded print and condensed print with the two pitches, this printer has eight different character widths available.

Enter this program to see how the print pitches, expanded print and condensed print can be combined:

```

10 ' Demo of various print pitches
20 LPRINT CHR$(15);
30 LPRINT CHR$(27);"M";
40 LPRINT "This line is CONDENSED ELITE pitch."
50 LPRINT CHR$(27);"P";
60 LPRINT "This line is CONDENSED PICA pitch."
70 LPRINT CHR$(18);
80 LPRINT CHR$(27);"M";
90 LPRINT "This line is NORMAL ELITE pitch."
100 LPRINT CHR$(27);"P";
110 LPRINT "This line is NORMAL PICA pitch."
120 LPRINT CHR$(27);"W1";
130 LPRINT CHR$(15);
140 LPRINT CHR$(27);"M";
150 LPRINT "This line is EXPANDED CONDENSED ELITE."
160 LPRINT CHR$(27);"P";
170 LPRINT "This line is EXPANDED CONDENSED PICA."
180 LPRINT CHR$(18);
190 LPRINT CHR$(27);"M";
200 LPRINT "This is EXPANDED ELITE."
210 LPRINT CHR$(27);"P";

```

```

220 LPRINT "This is EXPANDED PICA."
230 LPRINT CHR$(27);"WO"
240 END

```

Here's what you should get from this program:

```

This line is CONDENSED ELITE pitch.
This line is CONDENSED PICA pitch.
This line is NORMAL ELITE pitch.
This line is NORMAL PICA pitch.
This line is EXPANDED CONDENSED ELITE.
This line is EXPANDED CONDENSED PICA.
THIS IS EXPANDED ELITE.
THIS IS EXPANDED PICA.

```

■ Proportional printing

Have you ever looked closely at the printing in books and magazines? Doesn't it look nice? The main reason is that each character is given an amount of space proportional to its actual width. A typewriter (and most printers), on the other hand, give every character the same amount of space, no matter how wide it is. (Pica pitch, for example, gives a "w" and an "i" 1/10 of an inch each. Looking at these letters you see that a "w" is two or three times as wide as an "i".)

Well, you too can enjoy professional-looking proportional printing. You can turn proportional printing on and off with the following commands.

Table 3-8
Proportional commands

Function	Control code
Proportional ON	<ESC> "p" 1
Proportional OFF	<ESC> "p" 0

Try this program to see how the proportional spacing works.

```

10 ' Demo of proportional printing
20 LPRINT CHR$(27);"M";
30 LPRINT "This line is NORMAL ELITE printing."

```

```
40 LPRINT CHR$(27);"p1";
50 LPRINT "This line is PROPORTIONAL ELITE."
60 LPRINT CHR$(27);"P";
70 LPRINT "This line is PROPORTIONAL PICA."
80 LPRINT CHR$(27);"p0";
90 LPRINT "This line is NORMAL PICA printing."
100 END
```

When you run this program you should get this:

```
This line is NORMAL ELITE printing.
This line is PROPORTIONAL ELITE.
This line is PROPORTIONAL PICA.
This line is NORMAL PICA printing.
```

Line 20 selects the elite pitch and line 40 turns on the proportional printing with $\langle \text{ESC} \rangle$ "p" 1. Line 50 prints a line with proportional elite pitch. Then, line 60 selects the pica pitch, so that line 70 prints a line with proportional pica pitch. Finally, line 80 resets the proportional printing and line 90 prints a line in normal pica pitch.

MAKING WORDS STAND OUT

Your printer has good print density when it's just printing normally. But sometimes you may want something to stand out from the rest of the page. This printer provides two ways to do this: boldface and emphasized print. Both of these go over the characters twice, but they use slightly different methods to darken the characters. Let's try them and see what the difference is.

The following table shows the control codes for getting into and out of boldface and emphasized modes.

Table 3-9
Print emphasis commands

Function	Control code
Boldface ON	<ESC> "G"
Boldface OFF	<ESC> "H"
Emphasized ON	<ESC> "E"
Emphasized OFF	<ESC> "F"

Try them now with this little program:

```
10 ' Demo of boldface and emphasized
20 LPRINT CHR$(27);"G";
30 LPRINT "This line is BOLDFACE printing."
40 LPRINT CHR$(27);"E";
50 LPRINT "This line is BOLDFACE and EMPHASIZED."
60 LPRINT CHR$(27);"H";
70 LPRINT "This line is EMPHASIZED printing."
80 LPRINT CHR$(27);"F";
90 LPRINT "This line is NORMAL printing."
```

Run this program. The results will look like this:

```
This line is BOLDFACE printing.
This line is BOLDFACE and EMPHASIZED.
This line is EMPHASIZED printing.
This line is NORMAL printing.
```

Line 20 turns on boldface with <ESC> "G" and line 30 prints a line of text. In line 40 emphasized is turned on with <ESC> "E". Line 50 prints a line of text in boldface *and* emphasized. Line 60 then turns boldface off with <ESC> "H" so that line 70 can print in emphasized only. Finally, line 80 turns emphasized off, so your printer is set for normal printing.

Look closely at the different lines of printing. In the line of boldface printing each character has been printed twice, and the paper was moved up slightly the second time they were printed. In emphasized printing, the characters are moved slightly to the right the second time the printer prints. The second line combined both of these so that each character was printed 4 times.

MIXING PRINT MODES

We have learned how to use the various print modes individually and together. Now we'll see how to combine them more efficiently.

You have at your disposal a unique command that lets you choose any valid combination of print modes and pitch. This is the Master Print mode command. It looks like this:

<ESC> “!” *n*

Here, the value of *n* defines the print style to be selected. The value of *n* can range from 0 to 255, which is the range of values that can be stored in one eight-bit byte. If you look at each bit in this byte, you'll find that each one represents a printing style variation. Adding the binary values of the selected bits gives the value of *n* for a particular combination of print styles.

Table 3-10 shows the decimal values of the bits in the Master Print byte. To calculate the value *n* for a particular combination of printing styles, just add the decimal values of the features that you want to combine.

Table 3-10
Values of mixing print styles for Master Print

Bit	Print style	Decimal value
1	Elite print	1
2	Proportional print	2
3	Condensed print	4
4	Emphasized print	8
5	Boldface print	16
6	Expanded print	32
7	Italics	64
8	Underline	128

For example, if you want to select elite expanded boldface print, you would calculate the value of *n* like this:

Elite	1
Boldface	16
Expanded	32
<i>n</i> =	<u>49</u>

The command would look like this:

```
CHR$(27);"!";CHR$(49)
```

To better understand the way the print modes work, consider that each mode except pica (pica is the default) has a separate switch that can be turned on and off via software. Once the switch is on, it stays on until turned off. However, when two modes that conflict are turned on at the same time, the printer must choose which one to use.

For example, suppose you turn on both Elite and Emphasized modes. These cannot combine, so the printer must make a choice; in this case, the printer chooses Elite.

Summary notes

- 1) Pica is the default pitch and is active when Elite is turned off.
- 2) When two modes conflict, the one of lesser priority bit-wise is cancelled. For example, Condensed and Emphasized can not be printed at the same time, printing is Emphasized.
- 3) Elite cancels Emphasized.
- 4) Underline, Expanded and Italics modes combine with any print modes.
- 5) Emphasized will not mix with Elite or Condensed.

CHAPTER 4

FORMATTING TEXT

Subjects covered in Chapter 4 include—

- **The carriage return and line feed**
- **The amount of space between lines**
- **Moving to the next page**
- **The number of lines on a printed page**
- **Horizontal and vertical tabs**
- **Setting margins — left, right, top and bottom**
- **Centering and aligning**

Chapter 3 showed us the basic techniques for using the printer. Now we're ready for the more advanced ones. We'll concentrate on changing the appearance of the page to suit our needs.

LINES AND LINE SPACING

■ Starting a new line

Up until now the only time we have thought about printing on a new line is when we *didn't* want it to happen. We learned that putting a semicolon (;) at the end of a BASIC line will *not* end the line of printing. So somehow, the computer tells the printer when to end one line and start another.

There are two codes that are used to end one line and start another. They are *carriage return* (CHR\$(13)) and *line feed* (CHR\$(10)). Like the escape codes, they have been given abbreviations which you'll find many texts (including this one): <CR> and <LF>. The codes are simple, but their action is a little confusing (especially with BASIC). Let's begin with the carriage

return. Each time the printer receives a CHR\$(13) it returns the print head to the left margin. It does not advance the paper (if DIP switch 1-3 is on; see below).

Line feed is more complicated. Each time the printer receives a CHR\$(10) it both advances the paper one line and returns the print head to the left margin, ready to start a new line.

Now to add a little confusion — most (but not all) versions of BASIC add a line feed (CHR\$(10)) to every carriage return (CHR\$(13)) they receive. If your version of BASIC doesn't do this, then you should turn DIP switch 1-3 off so that your printer will add the line feed for you. When you have DIP switch 1-3 off the printer will do the same thing when it receives a carriage return as it does when it receives a line feed.

If you find that your printer double spaces when it should single space, then you probably need to turn DIP switch 1-3 on.

■ Reverse line feeds

Your printer can move the paper up or down. The unique tractor design allows the paper to be fed in either direction without jamming. This allows you to move around the page at will. You can use this feature to print several columns of text side by side, or to print a graph and then move back up and insert descriptive legends. As you experiment you're bound to come up with more uses!

The simplest form of reverse paper feeding is a reverse line feed. The code is <ESC> <LF>, which causes the paper to move down (in effect, moving the printing *up*) one line. A "line" used in a reverse line feed is the same size as a line in a regular line feed (this is normally 1/6 inch). When you change the line spacing (which you'll read about next), both forward and reverse line feeds change.

Table 4-1
Line feed commands

Function	Control code
Return print head to left margin	CHR\$(13)
Advance paper one line	CHR\$(10)
Reverse paper one line	<ESC> CHR\$(10)

■ Changing the line spacing

When you turn your printer on, the line spacing is set to 6 lines per inch. This is fine for most printing applications, but when you want something different, the printer makes it easy to set the line spacing to whatever you want.

Try this program to see how easy it is to change the line spacing:

```

10 ' Demo of line spacing
20 FOR I=1 TO 20
30 IF I=13 THEN 60
40 LPRINT CHR$(27);"A";CHR$(I);
50 LPRINT "THIS LINE SPACING IS SET TO";I
60 NEXT I
70 LPRINT "LINE SPACING IS SET TO 1/6 INCH
(NORMAL). "
80 LPRINT CHR$(27);"2"
90 END

```

This is what you will get:

```

THIS LINE SPACING IS SET TO 1
THIS LINE SPACING IS SET TO 2
THIS LINE SPACING IS SET TO 3
THIS LINE SPACING IS SET TO 4
THIS LINE SPACING IS SET TO 5
THIS LINE SPACING IS SET TO 6
THIS LINE SPACING IS SET TO 7
THIS LINE SPACING IS SET TO 8
THIS LINE SPACING IS SET TO 9
THIS LINE SPACING IS SET TO 10
THIS LINE SPACING IS SET TO 11
THIS LINE SPACING IS SET TO 12
THIS LINE SPACING IS SET TO 14
THIS LINE SPACING IS SET TO 15
THIS LINE SPACING IS SET TO 16
THIS LINE SPACING IS SET TO 17
THIS LINE SPACING IS SET TO 18
THIS LINE SPACING IS SET TO 19
THIS LINE SPACING IS SET TO 20
LINE SPACING IS SET TO 1/6 INCH (NORMAL).

```

Line 40 changes the line spacing. The command $\langle \text{ESC} \rangle$ "A" CHR\$(n) changes the line spacing to $n/72$ of an inch. The loop that is started in line 20 increases the value of n (the variable I in this program) each time it is executed. So the line spacing increases as the program continues. Line 30 just shortcuts the loop when $I = 13$, since BASIC won't let us send CHR\$(13) without adding an unwanted CHR\$(10) to it. Finally, the $\langle \text{ESC} \rangle$ "2" in line 80 resets the line spacing to 6 lines per inch. This is a shortcut that is the same as $\langle \text{ESC} \rangle$ "A" CHR\$(12).

When you run this program with the DIP switch 1-6 off (IBM mode), you cannot get the printout as shown above.

The $\langle \text{ESC} \rangle$ "A" CHR\$(n) command in IBM mode only defines the line spacing as $n/72$ of an inch; the $\langle \text{ESC} \rangle$ "2" command changes the line spacing to the amount defined by the previous $\langle \text{ESC} \rangle$ "A".

So, you need to change the following lines to the previous program as shown below for the IBM mode:

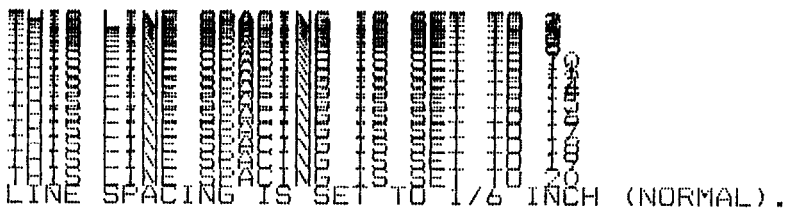
```
40 LPRINT CHR$(27);"A";CHR$(I);CHR$(27);"2";  
80 LPRINT CHR$(27);"A";CHR$(12);CHR$(27);"2"
```

You may wonder why $1/72$ of an inch was chosen as the increment for the line spacing command. There's a good reason: the dots that the printer makes are $1/72$ inch apart. So this means that you can vary the line spacing in increments as fine as one dot — unless you want finer spacing, in which case you may use one third dot spacing.

The $\langle \text{ESC} \rangle$ "3" CHR\$(n) command sets the line spacing in increments of $1/216$ inch. Change line 40 in your program to this:

```
40 LPRINT CHR$(27);"3";CHR$(I);
```

and run the program again. Now the results will look like this:



The program works just the same as before, but the line spacings are just one-third what they were. This is because $\langle \text{ESC} \rangle "3" \text{CHR}\(n) sets the line spacing to $n/216$ inch.

Table 4-2 shows all the line spacing commands, including several "shortcut" commands for commonly used line spacings.

Table 4-2
Line spacing commands

Function	Control code
Set line spacing to 1/8 inch	$\langle \text{ESC} \rangle "0"$
Set line spacing to 7/72 inch	$\langle \text{ESC} \rangle "1"$
Set line spacing to 1/6 inch or use $\langle \text{ESC} \rangle "A"$ definition	$\langle \text{ESC} \rangle "2"$
Set or define line spacing to $n/72$ inch	$\langle \text{ESC} \rangle "A" \text{CHR}\(n)
Set line spacing to $n/216$ inch	$\langle \text{ESC} \rangle "3" \text{CHR}\(n)
One-time line feed of $n/216$ inch	$\langle \text{ESC} \rangle "J" \text{CHR}\(n)
One-time reverse line feed of $n/216$ inch	$\langle \text{ESC} \rangle "j" \text{CHR}\(n)

■ Moving down the page without a carriage return

So far, all the commands that move the paper also move the print head back to the left margin. Normally this is what you want. Sometimes, though, you may wish to move down the page without moving the printhead back to the left margin. The following commands do just that.

The $\langle \text{ESC} \rangle "J" \text{CHR}\(n) command causes the printer to make one line feed of $n/216$ inch, but does not *change* the setting of the line spacing. Try this program to see how it works:

```
10 ' Demo of one-time line feeds
20 LPRINT "LINE NUMBER 1."
```

```
30 LPRINT "LINE NUMBER 2.";
40 ' One-time line feed
50 LPRINT CHR$(27);"J";CHR$(100);
60 LPRINT "LINE NUMBER 3."
70 LPRINT "LINE NUMBER 4."
80 END
```

Here is what your printer will produce:

```
LINE NUMBER 1.
LINE NUMBER 2.

                LINE NUMBER 3.
LINE NUMBER 4.
```

The $\langle \text{ESC} \rangle$ "J" $\text{CHR}\$(100)$ in line 50 changes the spacing to 100/216 inches for one line only without moving the printhead. The rest of the lines printed with the normal line spacing. Notice that both line 30 and line 50 end with semicolons. This prevents the normal line feed from occurring.

The $\langle \text{ESC} \rangle$ "j" $\text{CHR}\$(n)$ command works the same way except that the paper moves in the opposite direction. Try this simple change to your program to see the difference.

```
40 ' One-time reverse line feed
50 LPRINT CHR$(27);"j";CHR$(100);

                LINE NUMBER 3.
LINE NUMBER 4.
LINE NUMBER 1.
LINE NUMBER 2.
```


PAGE CONTROL

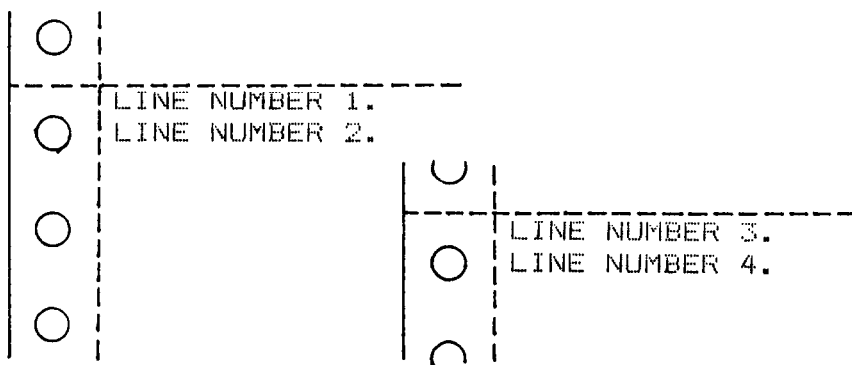
Now that we have seen how to control line spacing, we can go on to page control — positioning the printing on the page and adjusting the paper length.

■ Form feed

The simplest forms control code is the *form feed*. Form feed (or `<FF>`) is `CHR$(12)` and causes the printer to move the paper to the top of the next sheet. Try it by changing lines 40 and 50 to this:

```
40 ' Form feed
50 LPRINT CHR$(12);
```

Before you run the program, turn your printer off and adjust the paper so that the top of the sheet is even with the top of the ribbon guide on the print head, then turn the printer back on. If you don't remember how to do this, review Chapter 2. When you run the program, the results will look like this:



The form feed (`CHR$(12)`) in line 50 caused the printer to move to the top of a new page before printing the last two lines.

■ Reverse form feed

Just as your printer can perform a reverse line feed, it can do a reverse form feed. This code moves the paper so that the print head is positioned at the top of the current page. This can be used, for example, to print text in a multi-column magazine format; print the first column, then reverse form feed back to the top of the page to start the second column. The code for reverse form feed is easy to remember: `<ESC> <FF>`.

Table 4-3
Form feed commands

Function	Control code
Advance paper to top of next page	<code>CHR\$(12)</code>
Reverse paper to top of current page	<code><ESC> CHR\$(12)</code>

■ Changing the page length

You may have some computer forms that are not 11 inches high. That's no problem, because you can tell your printer how high the forms are that you are using. There are two commands for doing this, shown in this table.

Table 4-4
Form length control

Function	Control code
Set the page length to <i>n</i> lines	<code><ESC> "C" CHR\$(<i>n</i>)</code>
Set the page length to <i>n</i> inches	<code><ESC> "C" CHR\$(0) CHR\$(<i>n</i>)</code>

Let's set up a 7 inch high form length, which is typical of many computer checks. The following program will do it.

```
10 ' Demo of variable form lengths
20 LPRINT CHR$(27);"C";CHR$(0);CHR$(7);
30 LPRINT "PAY TO THE ORDER OF:"
40 LPRINT CHR$(12);
50 LPRINT "PAY TO THE ORDER OF:"
```

This program should print "PAY TO THE ORDER OF:" twice, and they should be 7 inches apart. Line 20 sets the form length to 7 inches. After line 30 prints, line 40 sends a form feed advance the paper to the top of the next form. Line 50 then prints its message.

After you have run this program, turn off the printer and adjust the top of form position. When you turn the printer back on the page length will reset to its normal setting (usually 11 inches).

TOP AND BOTTOM MARGINS

Some programs don't keep track of where they are printing on the page. This causes problems when you get to the bottom of a page because these programs just keep on printing, right over the perforation. This makes it very hard to read, especially if a line happens to fall right on the perforation.

The printer has a solution to this predicament. This printer can keep track of the position on the page, and advance the paper so that you won't print too near the perforation. There are two commands to do this. One controls the space at the top of the page (top margin) and the other controls the space at the bottom of the page (bottom margin). The control codes are given in the following table:

Table 4-5
Top and bottom margin commands

Function	Control code
Set top margin to n lines	$\langle \text{ESC} \rangle$ "r" CHR\$(n)
Set bottom margin to n lines	$\langle \text{ESC} \rangle$ "N" CHR\$(n)
Cancel top and bottom margins	$\langle \text{ESC} \rangle$ "O"

In both cases the value of n tells your printer how many lines to skip, although there is a slight difference in the usage. When you set the top margin with $\langle \text{ESC} \rangle$ "r" CHR\$(n), the value of n tells the printer what line to start printing on. When you set the bottom margin with $\langle \text{ESC} \rangle$ "N" CHR\$(n), the value of n tells

the printer how many blank lines should be left at the bottom of the page.

Let's try a simple application to see how these margins work. Enter this program, which will print 150 lines *without* top and bottom margins.

```
10 ' Demo of top and bottom margins
60 LPRINT CHR$(12);      :' Form feed
70 FOR I=1 TO 150
80 LPRINT "THIS IS LINE";I
90 NEXT I
110 LPRINT CHR$(12)      :' Form feed
120 END
```

When you run this program it will print 150 lines right down the page and across the perforations. When it's done line 110 sends a form feed to advance to the top of the next page.

Now add the following lines to your program. (Don't forget the semicolons or you won't get quite the same results that we did.)

```
20 ' Leave 6 blank lines at the bottom of page
30 LPRINT CHR$(27);"N";CHR$(6);
40 ' Start top of page at line 6
50 LPRINT CHR$(27);"r";CHR$(6);
100 LPRINT CHR$(27);"O";  : ' Clear top and
    bottom margins
```

Now when you run the program, your printer skip the first six lines and the last six lines on each page. Always send a form feed after setting the top margin, or it will not work on the first page printed. That's because the top margin only takes effect after a form feed.

Line 50 sets the top margin, line 30 sets the bottom margin, and line 100 clears both margins when we are done.

THIS IS LINE 1
THIS IS LINE 2
THIS IS LINE 3
THIS IS LINE 4
THIS IS LINE 5
THIS IS LINE 6
THIS IS LINE 7
THIS IS LINE 8
THIS IS LINE 9
THIS IS LINE 10

THIS IS LINE 49
THIS IS LINE 50
THIS IS LINE 51
THIS IS LINE 52
THIS IS LINE 53
THIS IS LINE 54
THIS IS LINE 55

THIS IS LINE 56
THIS IS LINE 57
THIS IS LINE 58
THIS IS LINE 59
THIS IS LINE 60
THIS IS LINE 61
THIS IS LINE 62
THIS IS LINE 63
THIS IS LINE 64

THIS IS LINE 103
THIS IS LINE 104
THIS IS LINE 105
THIS IS LINE 106
THIS IS LINE 107
THIS IS LINE 108
THIS IS LINE 109
THIS IS LINE 110

THIS IS LINE 111
THIS IS LINE 112
THIS IS LINE 113
THIS IS LINE 114
THIS IS LINE 115
THIS IS LINE 116
THIS IS LINE 117
THIS IS LINE 118
THIS IS LINE 119
THIS IS LINE 120
THIS IS LINE 121

When you want to reset the margins to the default values, you have two choices. You can either turn the printer off and back on, or you can set margin values equal to the default values. This means that you should set a left margin of 0 and right margin of 80 in pica pitch.

If you change the pitch of your printing after you set your margins, the margins will not change. They stay at the same place on the page. So if you set the margins to give you 65 columns of printing when you are using pica type, and you change to elite type, you will have room for more than 65 columns of elite printing between the margins.

HORIZONTAL AND VERTICAL TABS

Suppose you need to move across the page to a certain position several times in a document. It's not much fun to type in space after space. You don't have to — you can "tab" your way across the page.

Your printer's tabs are like those on a typewriter, but much more powerful. You have both horizontal and vertical tabs which can be used for both text and graphics — and they're really handy for indenting paragraphs and making tables.

■ Horizontal tabs

Horizontal tabs are set automatically every eight positions. To move the print head to the next tab position, send CHR\$(9).

Try this program to see how the default tabs work.

```
10 ' Demo of horizontal tabbing
20 LPRINT "ONE";CHR$(9);"TWO";CHR$(9);"THREE";
  CHR$(9);"FOUR"
```

Here's what you should get —

```
ONE      TWO      THREE     FOUR
```

Even though the words are different length, they are spaced out evenly by the horizontal tabs.

Now add the following line to your program to set different horizontal tabs:

```
15 LPRINT CHR$(27);"D";CHR$(7);CHR$(14);CHR$(21);
   CHR$(0)
```

⟨ESC⟩ “D” is the command to begin setting horizontal tabs. It must be followed by characters representing the positions where you want the tabs set. In our program we are setting tabs in columns 7, 14, and 21. The final CHR\$(0) ends the string of tabs. In fact, any character that is not greater than the previous one will stop setting tabs. This means that you must put all your tab values in order, from least to greatest, or they won’t all get set.

When you run the program now it produces this:

```
ONE      TWO      THREE     FOUR
```

The words are now closer together, but still evenly spaced. To reset the default tabs, use the following program:

```
LPRINT CHR$(27);"e0";CHR$(8)
```

This command sets the horizontal tabs to every eight columns.

Table 4-7
Horizontal tab commands

Function	Control code
Advance to next tab position	CHR\$(9)
Set tabs at $n1$, $n2$, etc.	⟨ESC⟩ “D” CHR\$($n1$) CHR\$($n2$) CHR\$(0)
Set tabs every n columns	⟨ESC⟩ “e” 0 CHR\$(n)

■ One-time horizontal tabs

Suppose you need to move to a position across the page, but you only need to do it once. It doesn't make much sense to set up a tab to use only one time. There is an easier way.

The solution is called a *one-time* tab. Table 4-8 shows the three commands.

Table 4-8
One-time horizontal tabs

Function	Control code
One-time horizontal tab of n spaces	$\langle \text{ESC} \rangle "f" 0 \text{CHR}\(n)
Absolute horizontal tab	$\langle \text{ESC} \rangle "$" \text{CHR}\$(n1) \text{CHR}\$(n2)$
Relative horizontal tab	$\langle \text{ESC} \rangle "\backslash" \text{CHR}\$(n1) \text{CHR}\$(n2)$

The first command in Table 4-8 moves the print head n columns to the right. It has the same effect as sending n spaces to the printer.

The absolute horizontal tab command moves the print head to a specified position on the page. The position that you want the print head to move to (measured from the left margin) is specified by the values of $n1$ and $n2$ using the formula $n1 + n2 \times 256$.

Try this program to see how this works.

```

10 ' Demo of absolute horizontal tabs
20 FOR I=60 TO 70 STEP 2
30 LPRINT I;"++++";
40 LPRINT CHR$(27);"$";CHR$(I);CHR$(0);
50 LPRINT "Horizontal tab."
60 NEXT I
70 END

```

In this program, the print head is positioned before the "H" in "Horizontal" is printed.

```

60 +++++ Horizontal tab.
62 +++++ Horizontal tab.
64 +++++ Horizontal tab.
66 +++++ Horizontal tab.
68 +++++ Horizontal tab.
70 +++++ Horizontal tab.

```

The relative horizontal tab command can move the print head right from the current position with NLQ characters. The formula for calculating how much the print head moves is the same as in the absolute horizontal tab command.

Try this program to see how this works.

```

10 ' Demo of relative horizontal tabs
20 LPRINT CHR$(27);"x1";
30 FOR I=1 TO 3
40 LPRINT "Relative";
50 LPRINT CHR$(27);"\";CHR$(I*20);CHR$(0);
60 LPRINT "Horizontal";
70 LPRINT CHR$(27);"\";CHR$(I*10);CHR$(0);
80 LPRINT "Tab."
90 NEXT I
100 LPRINT CHR$(27);"x0"
110 END

```

```

Relative Horizontal Tab.
Relative Horizontal Tab.
Relative Horizontal Tab.

```

■ Vertical tabs

Vertical tabs have the same kinds of uses that horizontal tabs do — they just work in the other direction. Horizontal tabs allow you to reach a specific column on the page no matter where you start from. Vertical tabs are the same. If you have a vertical tab set at line 20, a *vertical tab* (or <VT>) will move you to line 20 whether you start from line 5 or line 19.

Vertical tabs are *not* set at the power-on default. If you send a CHR\$(11) which is the ASCII code for <VT> before we have set

up tabs, the command advances the paper only one line. Enter this program to see how this works.

```
10 ' Demo of vertical tabs
40 LPRINT CHR$(11);"FIRST TAB."
50 LPRINT CHR$(11);"SECOND TAB."
60 LPRINT CHR$(11);"THIRD TAB."
70 LPRINT CHR$(11);"FOURTH TAB."
```

Now, let's set some vertical tabs of our own. Add these lines to the program:

```
20 LPRINT CHR$(27);"B";CHR$(10);CHR$(15);
30 LPRINT CHR$(25);CHR$(30);CHR$(0);
```

FIRST TAB.

SECOND TAB.

THIRD TAB.

FOURTH TAB.

<ESC> "B" is the command to set vertical tabs. Like the horizontal tab setting command, tab positions must be defined

in ascending order. Our example sets vertical tabs at lines 10, 15, 25 and 30. Then the CHR\$(11) in each of the following lines advances the paper to the next vertical tab.

Add one more line to the program to demonstrate one more feature of vertical tabs.

```
80 LPRINT CHR$(11);"FIFTH TAB."
```

Now when you run the program the first page looks just like before, but line 80 sends one more $\langle VT \rangle$ than there are tabs. This doesn't confuse the printer — it advances the paper to the *next* tab position, which happens to be the first tab position on the next page.

There's a one-time vertical tab command that works just like the one-time horizontal tab command. Try this program to see how these commands work.

```
10 ' Demo of print positions
20 LPRINT "* 1st Line, 1st Column."
30 LPRINT CHR$(27);"f1";CHR$(3);
40 LPRINT "* 5th Line, 1st Column."
50 LPRINT CHR$(27);"f0";CHR$(7);
60 LPRINT "* 6th Line, 7th Column."
70 LPRINT CHR$(27);"f1";CHR$(2);
80 LPRINT CHR$(27);"f0";CHR$(5);
90 LPRINT "* 9th Line, 5th Column."
100 END
```

In this program, line 30 sends a one-time vertical tab with $\langle ESC \rangle$ "f" 1 command. Line 50 sends a one-time horizontal tab with $\langle ESC \rangle$ "f" 0 command. Finally, line 70 and line 80 send one-time vertical tab and one-time horizontal tab, so the line 90 prints text at the specified position.

* 1st Line, 1st Column.

* 5th Line, 1st Column.

* 6th Line, 7th Column.

* 9th Line, 5th Column.

Table 4-9
Vertical tab commands

Function	Control code
Advance paper to next tab position	CHR\$(11)
Set vertical tabs at $n1$, $n2$, etc.	<ESC> "B" CHR\$($n1$) CHR\$($n2$) CHR\$(0)
Set vertical tabs every n lines	<ESC> "e" 1 CHR\$(n)
One-time vertical tab	<ESC> "f" 1 CHR\$(n)

■ Vertical tab channels

Vertical tab channels are especially helpful in two situations. The first occurs when you are writing a program to accompany a preprinted form that can accommodate various types of responses. The second occurs when you create a multipage form or report with different vertical tabs on each page.

Table 4-10
Vertical tab channel commands

Function	Control code
Set vertical tabs at $n1$, $n2$, etc. as channel $n0$	<ESC> "b" CHR\$($n0$) CHR\$($n1$) CHR\$($n2$) CHR\$(0)
Select vertical channel $n0$	<ESC> "p" CHR\$($n0$)

You can store up to eight channels of tab stops. They are numbered from 0 to 7. If you have already stored a set using <ESC> "B" command, your printer has labelled it as channel 0.

Try this program how to see the vertical tab channels work.

10 ' Demo of vertical tab channels

```
20 LPRINT CHR$(27);"b";CHR$(1);CHR$(10);CHR$(20);
   CHR$(0);
30 LPRINT CHR$(27);"b";CHR$(2);CHR$(15);CHR$(25);
   CHR$(0);
40 LPRINT CHR$(27);"b";CHR$(3);CHR$(17);CHR$(28);
   CHR$(0);
50 ' Use vertical tab channels
60 FOR I=1 TO 3
70 LPRINT "TOP OF FORM"
80 LPRINT CHR$(27);"/";CHR$(I);
90 LPRINT CHR$(11);
100 LPRINT "1ST TAB OF CHANNEL";I
110 LPRINT CHR$(11);
120 LPRINT "2ND TAB OF CHANNEL";I
130 LPRINT CHR$(12);
140 NEXT I
150 LPRINT CHR$(27);"@"
160 END
```

When you run this program the printout should look like this.

TOP OF FORM	TOP OF FORM	TOP OF FORM
1ST TAB OF CHANNEL 1	1ST TAB OF CHANNEL 2	1ST TAB OF CHANNEL 3
2ND TAB OF CHANNEL 1	2ND TAB OF CHANNEL 2	2ND TAB OF CHANNEL 3

In this program we set tabs at 10 and 20 in channel 1 in line 20. In line 30 we set tabs 15 and 25 in channel 2, and in line 40 we set tabs 17 and 28 in channel 3.

Because the channels are stored, you must make the printer to recall one before you use it, so we used `<ESC> "P" CHR$(n0)` in line 80.

CENTERING AND ALIGNING TEXT

Text can be arranged in any of three formats: left aligned or left-justified (normal printing with "ragged" right margin), centered between the margins, or right aligned (right-justified). These are selected by the following commands:

Table 4-11
Alignment commands

Function	Control code
Left-justified printing	<code><ESC> "a" CHR\$(0)</code>
Centered printing	<code><ESC> "a" CHR\$(1)</code>
Right-justified printing	<code><ESC> "a" CHR\$(2)</code>

Try this program to see how easy it is.

```

10 ' Demo of aligning and centering
20 LPRINT CHR$(27);"1";CHR$(20);
30 LPRINT CHR$(27);"Q";CHR$(60);
40 LPRINT CHR$(27);"a";CHR$(0);
50 LPRINT "THIS LINE IS LEFT-ALIGNED."
60 LPRINT CHR$(27);"a";CHR$(1);
70 LPRINT "THIS LINE IS CENTERED."
80 LPRINT CHR$(27);"a";CHR$(2);
90 LPRINT "THIS LINE IS RIGHT-ALIGNED."
100 LPRINT CHR$(27);"@"
110 END

```

When you run this program, you should get like this:

```

THIS LINE IS LEFT-ALIGNED.
      THIS LINE IS CENTERED.
                THIS LINE RIGHT-ALIGNED.

```

MEMO

CHAPTER 5

SPECIAL FEATURES OF THE PRINTER

Subjects covered in Chapter 5 include—

- **Printer's bell**
- **Master reset**
- **Uni-directional printing**
- **International character sets**
- **Printing BIG characters**
- **The optional sheet feeder**
- **Macro instruction**
- **Reading a hex dump**

In the previous chapters we have learned about several groups of control codes. In this chapter we will look at more control codes. These codes don't fit neatly into any of the groups that we have looked at so far, but they add a lot of useful features to your printer.

■ **Now hear this**

You may have heard the printer's *bell* if you have ever run out of paper. And you may have wondered why it's called a bell when it *beeps* instead of ringing! It's a long story that goes back to the early days of computers, when teletype machines were used for computer terminals. These mechanical marvels were equipped with a bell that could be heard for blocks. This bell was used to signal the operator that something needed attention. The code that the computer sent to the teletype machine to ring the bell was, reasonably enough, called a *bell code*. The name *bell code* is still with us, even if the bell has changed to a beeper, and a lot of people still call the beeper a bell, even if it doesn't sound like one.

The code to sound the “bell” is CHR\$(7), which is ASCII code 7 or <BEL>. Any time your printer receives this code it will sound the bell for a quarter of a second. This can be used to remind the operator to change the paper or to make other adjustments to the printer.

You can try this by typing:

```
LPRINT CHR$(7)
```

■ Resetting the printer

Up to now when we wanted to reset the printer to the power on condition we have had to either turn the printer off and on again, or to send the specific codes that reset the particular features. There is an easier way. The control code <ESC> “@” will reset all of the printer’s features to the power on condition (as determined by the DIP switches), with two exceptions. The exceptions are that <ESC> “@” will not erase any characters that you have stored in the printer’s RAM memory (Chapter 6 tells you how to create your own characters), and it won’t erase a macro if you have one stored in the printer’s RAM (this chapter will tell you how to create a macro).

In addition, if you set the “Panel” mode by the control panel settings at power on, these functions will remain with this control code.

■ Taking the printer off line

You know how to put your printer, *off line* with the On Line key on the control panel. The printer has another *off line* state that can be controlled from the computer. When you turn the printer *off line* from the computer, the printer will ignore anything that you send it, except for the code to go *on line* again. CHR\$(19) is the code to turn the printer off line; CHR\$(17) turns the printer back to on line status.

■ Printing the bottom of the sheet

Sometimes when you are using individual sheets of paper you may want to print near the bottom of a sheet. The paper-out detector usually stops the printer when you are about 1 inch from the bottom of the sheet. This is notify you if you are run-

ning out of continuous paper.

Your printer has the ability to print right to the bottom of the sheet. You can disable the paper-out detector so that it doesn't stop the printer. This will allow you to print to the end of the sheet, and even beyond if you are not careful. The codes to control the paper-out detector, along with the other codes that we have just looked at are in the following table.

Table 5-1
Some miscellaneous commands

Function	Control code
Sound bell	CHR\$(7)
Master rest	<ESC> "@"
Off line	CHR\$(19)
On line	CHR\$(17)
Paper-out detector off	<ESC> "8"
Paper-out detector on	<ESC> "9"
Move print head back one space	CHR\$(8)
Delete last character sent	CHR\$(127)
Cancel text in print buffer	CHR\$(24)
Print "slash zero"	<ESC> "~" 1
Print "normal zero"	<ESC> "~" 0
Immediate-print on	<ESC> "i" 1
Immediate-print off	<ESC> "i" 0
Add <i>n</i> dot spaces between characters	<ESC> CHR\$(32) CHR\$(<i>n</i>)

■ Backspace, delete, and cancel text

Backspace (CHR\$(8)) "backs up" the print head so that you can print two characters right on top of each other. Each time your printer receives a backspace it moves the print head one character to the left, instead of to the right. You can *strike over* multiple letters by sending more than one backspace code.

Delete (CHR\$(127)) also "backs up" one character, but then it "erases" the previous character (it's erased from your printer's buffer, not from the paper).

Cancel text (CHR\$(24)) deletes all the text in the print buffer; that is, in the line before the delete text command. Since your printer prints one line of text at a time, only that line will be deleted.

The following program shows how these codes works.

```

10 LPRINT "BACKSPACE DOES NOT";
20 LPRINT CHR$(8);CHR$(8);CHR$(8);
30 LPRINT "=== WORK."
40 LPRINT "DELETE DOES NOT";
50 LPRINT CHR$(127);CHR$(127);CHR$(127);
60 LPRINT "WORK."
70 LPRINT "CANCEL LINE";
80 LPRINT CHR$(24);
90 LPRINT "DOES NOT WORK."

```

Here is what this program will print:

```

BACKSPACE DOES NOT WORK.
DELETE DOES WORK.
DOES NOT WORK.

```

The backspace codes in line 20 move the print head a total of three spaces to the left so that the first part of line 30 will overprint the word "NOT". The delete codes in line 50 "erase" the three letters in the word "NOT" so that it doesn't even print.

In line 80, CHR\$(24) deletes the words in line 70. The semicolon at the end of line 70 prevents a line feed from causing that line to print before the printer receives the CHR\$(24) code. The text in line 90 prints as it normally would because it is after CHR\$(24).

■ Printing zeroes

There are two types of zeroes. There is the one we use every day — 0 — and this is what your printer will print as its default.

The other type is used almost exclusively in computers and engineering. It is called the "slash zero" and is written like this — Ø. The line through the number prevents you from misreading it as the letter "O". Before high-quality printers were available, this was a good idea but there is no special need for it today (although you may want to use the slash zero for a special effect).

■ Immediate-print

This printer can print at the rate of 100 characters per second.

But it will also print more slowly at the speed of your typing. In the *immediate-print* mode, the print head prints one character at a time, as you send it. The printer also moves the paper up so that you can see the current line and then down to continue printing.

You can turn the immediate-print mode on with $\langle \text{ESC} \rangle$ "i" 1. Before looking at it, let's review the normal operation of the print buffer. Enter this program.

```
20 A$="" : INPUT " TYPE A CHARACTER ",A$
30 IF A$="" THEN 50
40 LPRINT A$; : GOTO 20
50 LPRINT : LPRINT CHR$(27);"@"
```

Now type several characters, and after each press the RETURN key. The printer simply puts the characters into its buffer while it waits for a carriage return code. (In this program the RETURN key doesn't send a carriage return code.) To end this program and print the contents of the buffer, press RETURN alone.

Now add this line:

```
10 LPRINT CHR$(27);"i1";
```

And RUN the program. Your printer will now respond to your typing — immediately.

When you are finished, press RETURN alone.

■ Adjusting the width of space between characters

This printer provides a command that adjusts the space between all the characters that it prints when the NLQ character mode is on while the DIP switch 1-6 is set on. The $\langle \text{ESC} \rangle$ CHR\$(32) CHR\$(n) command adds blank space between the characters. The units of space that are added vary in 1/240-inch units.

Try this program to see how this works.

```
10 ' Demo of adjusting spaces
20 LPRINT CHR$(27);"x1";
```

```

30 FOR I=10 TO 1 STEP -2
40 LPRINT CHR$(27);" ";CHR$(I);
50 LPRINT "This line is added";I;"spaces."
60 NEXT I
70 LPRINT CHR$(27);" ";CHR$(0);
80 LPRINT "This line is Normal space."
90 END

```

```

This line is added 10 spaces.
This line is added 8 spaces.
This line is added 6 spaces.
This line is added 4 spaces.
This line is added 2 spaces.
This line is Normal space.

```

This command can be used to produce micro-justification, which is a method of justifying lines by increasing the space between each character.

■ Uni-directional printing

Uni-directional printing means *printing in one direction only*. Your printer normally prints when the print head is moving in both directions. But once in a while you may have an application such as making table where you are more concerned with how the vertical lines align than with how fast it prints. The printer lets you make this choice. The table below shows the commands for controlling how the printer prints.

Table 5-2
Printing direction commands

Function	Control code
Print in one direction	<ESC> "U" 1
Print in both directions	<ESC> "U" 0
One time print in one direction	<ESC> "<"

Try this program to see the difference that printing in one direction makes.

```

10 ' Demo of uni-directional printing
20 LPRINT CHR$(27);"1";

```

```
30 FOR I=1 TO 10
40 LPRINT "|"
50 NEXT I
60 LPRINT : LPRINT
70 LPRINT CHR$(27);"U1";
80 FOR I=1 TO 10
90 LPRINT "|"
100 NEXT I
110 LPRINT CHR$(12);CHR$(27);"@"
```

Here is what you will get. The top line is printed bi-directionally, and the bottom is printed uni-directionally. You will have to look hard because there isn't much difference, but the bottom vertical lines are perfectly aligned.

|

Let's analyze the program. Line 20 defines the line spacing to 7/72 of an inch so that the characters that we print will touch top to bottom. Lines 30 ~ 50 print 10 vertical line characters. Then line 70 sets uni-directional printing and the vertical lines are printed again. Finally line 110 sends a form feed to advance the paper to the top of a new page, and then uses the master reset to restore the printer to the power on condition.

You can also set the printer to print in one direction for one line only by using the `<ESC> "<"` command. This command immediately moves the print head to the left margin and then prints the remainder of the line from left to right.

■ The seven bit dilemma

Some computers (not the IBM-PC fortunately!) don't have the capability to send eight bits on their parallel interface, but can only send seven bits. This would make it impossible for these computers to use this printer's block graphics characters and special symbols if our engineers hadn't thought of a solution. (All of these characters have ASCII codes greater than 127 which means that the eighth bit must be on to use them.) The solution lies in the three control codes given in the following table:

Table 5-3
Eighth bit controls

Function	Control code
Turn the eighth bit ON	<ESC> ">"
Turn the eighth bit OFF	<ESC> "="
Accept the eighth bit "as is" from the computer	<ESC> "#"

■ Block graphics characters and special symbols

Besides the upper and lower case letters and symbols that we are by now familiar with, your printer has a whole different set of characters that are for special uses. These characters include block graphics for drawing forms and graphs, and special symbols for mathematical, engineering and professional uses. The special characters are included in two character sets. The character set you normally use with IBM mode is called character set #1. The special characters are printed out when you send ASCII codes 160~255 to the printer.

Your printer also offers character set #2 which is almost the same as character set #1 except for the addition of ASCII codes 3~6, 20, 21, 26, and 128~159. Character set #2 is selected with <ESC>"6"; to go back to character set #1, use <ESC>"7".

You can also specify the power-on default character set by setting DIP switch 1-4 on for character set #1 and off for character set #2. The following program will print out all of the graphics characters available:


```

10 LPRINT CHR$(27);"0";
20 LPRINT CHR$(27);"6";
30 FOR J=3 TO 6
40 LPRINT " ";J;CHR$(J);CHR$(9);
50 NEXT J
60 LPRINT " 20 ";CHR$(20)
70 LPRINT
80 LPRINT " 21 ";CHR$(21);CHR$(9);
90 LPRINT " 26 ";
100 LPRINT CHR$(26)
120 LPRINT
130 FOR J=128 TO 254 STEP 5
140 FOR I=J TO J+4
150 IF I>254 THEN 170
160 LPRINT I;CHR$(I);CHR$(9);
170 NEXT I
180 LPRINT :LPRINT
190 NEXT J

```

Figure 5-1 shows what this program will print. If your chart doesn't look like this because it has regular letters and numbers instead of the special symbols, then your computer is only using seven bits. You can get the correct printout by changing line 160 to this:

```

160 LPRINT I;CHR$(27);">";CHR$(I);CHR$(27);"=";
    CHR$(9);

```

A note for the IBM-PC users:

When you run this program, you cannot get the right-pointed arrow (CHR\$(26)) with the IBM-PC computers. This is because the IBM-PC does not send this code to the printer.

There is a solution to avoid this problem. Change lines 100 and 120 to the either set of the following lists.

```

100 O=INP(&H379) : IF O<128 THEN 100
110 OUT &H378,26 : OUT &H37A,5 : OUT &H37A,4
120 LPRINT : LPRINT

```

```

100 O=INP(&H3BD) : IF O<128 THEN 100
110 OUT &H3BC,26 : OUT &H3BE,5 : OUT &H3BE,4
120 LPRINT : LPRINT

```

3 ♡	4 ♠	5 ♣	6 ♠	20 ♠
21 ♠	26 ♠			
128 ♣	129 ü	130 é	131 ä	132 ä
133 ä	134 ä	135 y	136 ë	137 ë
138 é	139 i	140 i	141 i	142 Å
143 Å	144 é	145 æ	146 Æ	147 ö
148 ö	149 ö	150 ü	151 ü	152 y
153 ö	154 ü	155 ø	156 £	157 ¥
158 R	159 F	160 Å	161 i	162 ó
163 ú	164 ñ	165 ñ	166 @	167 @
168 ÷	169 -	170 -	171 ½	172 ¼
173 i	174 *	175 *	176	177
178	179	180	181	182
183	184 ¶	185 ¶	186	187 ¶
188	189	190 ¶	191 ¶	192 L
193 ⊥	194 T	195 †	196 -	197 †
198 †	199	200 ⊥	201 ¶	202 ⊥
203 ¶	204 ¶	205 =	206 ¶	207 ⊥
208 ⊥	209 ¶	210 ¶	211 ⊥	212 ⊥
213 F	214 ¶	215 ¶	216 †	217 J
218 Γ	219	220	221	222
223	224 α	225 β	226 Γ	227 π
228 Σ	229 σ	230 μ	231 τ	232 ø
233 θ	234 ω	235 δ	236 ω	237 ø
238 €	239 ∅	240 ≡	241 ±	242 ≥
243 ≤	244 ∫	245 J	246 ÷	247 ≈
248 °	249 ·	250 -	251 √	252 ∅
253 ²	254 ■			

Figure 5-1. Character set #2

So how are all of these strange characters used? Here is a short program that demonstrate how the graphics characters can be combined to create a figure: the 5 of clubs.

```

10 LPRINT CHR$(27);"6";
20 LPRINT CHR$(218);CHR$(196);CHR$(196);
   CHR$(196);CHR$(196);CHR$(196);CHR$(191)
30 LPRINT CHR$(179);CHR$(53);CHR$(32);CHR$(32);
   CHR$(32);CHR$(32);CHR$(179)
40 LPRINT CHR$(179);CHR$(32);CHR$(5);CHR$(32);
   CHR$(5);CHR$(32);CHR$(179)
50 LPRINT CHR$(179);CHR$(32);CHR$(32);CHR$(5);
   CHR$(32);CHR$(32);CHR$(179)
60 LPRINT CHR$(179);CHR$(32);CHR$(5);CHR$(32);
   CHR$(5);CHR$(32);CHR$(179)
70 LPRINT CHR$(179);CHR$(32);CHR$(32);CHR$(32);
   CHR$(32);CHR$(53);CHR$(179)
80 LPRINT CHR$(192);CHR$(196);CHR$(196);
   CHR$(196);CHR$(196);CHR$(196);CHR$(217)
90 LPRINT CHR$(27);"7"

```

If you have a 7-bit interface, add the following lines to the program given above:

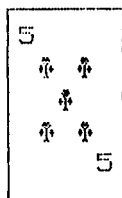
```

15 LPRINT CHR$(27);">";
95 LPRINT CHR$(27);"="

```

In this program line 10 selects character set #2 so that all the graphics characters can be used (including the "club" symbol). Then lines 20~80 print the 5 of clubs figure. Line 90 cancels character set #2 (which is the same as selecting character set #1).

Here is what this program prints:



■ International character sets

Your printer changes languages by changing 12 characters that are different in the different languages. These sets of characters are called *international character sets* — and there are 11 sets in all. The control codes to select the international character sets are given in Table 5-4.

Table 5-4
International character set commands

Country	Control code
U.S.A.	<ESC> "R" CHR\$(0)
France	<ESC> "R" CHR\$(1)
Germany	<ESC> "R" CHR\$(2)
England	<ESC> "R" CHR\$(3)
Denmark type I	<ESC> "R" CHR\$(4)
Sweden	<ESC> "R" CHR\$(5)
Italy	<ESC> "R" CHR\$(6)
Spain	<ESC> "R" CHR\$(7)
Japan	<ESC> "R" CHR\$(8)
Norway	<ESC> "R" CHR\$(9)
Denmark type II	<ESC> "R" CHR\$(10)

The characters that change are shown beneath their ASCII code in Table 5-5.

Table 5-5
International character sets

Country	35	36	64	91	92	93	94	96	123	124	125	126
U.S.A.	#	\$	@	[\]	^	'	{		}	~
France	#	\$	ã	°	ç	û	^	'	ë	û	ë	..
Germany	#	\$	§	Ä	Ü	Ü	^	'	ä	ö	ü	ß
England	£	\$	@	[\]	^	'	{		}	~
Denmark type I	#	\$	@	ŕ	ø	Å	^	'	æ	ø	å	~
Sweden	#	¥	É	Ä	Ü	Å	Ü	ë	ä	ö	å	Ü
Italy	#	\$	@	°	\	é	^	ü	ä	ö	ë	ì
Spain	₧	\$	@	í	ñ	¿	^	'	ñ	}	}	~
Japan	#	\$	@	[¥]	^	'	{		}	~
Norway	#	¥	É	ŕ	ø	Å	Ü	ë	æ	ø	å	Ü
Denmark type II	#	\$	É	ŕ	ø	Å	Ü	ë	æ	ø	å	Ü


```

40 LPRINT CHR$(I);
50 NEXT I
60 LPRINT CHR$(16);CHR$(17);
70 FOR I=21 TO 31
80 LPRINT CHR$(I);
90 NEXT I
100 LPRINT
110 FOR I=128 TO 134
120 LPRINT CHR$(I);
130 NEXT I
140 LPRINT CHR$(144);CHR$(145);
150 FOR I=149 TO 159
160 LPRINT CHR$(I);
170 NEXT I
180 LPRINT CHR$(27);"IO"
190 END

```

When you run this program you should get like this:

```

aèùòì ° £$βø " ÅÖÜäéé¥
aèùòì ° £$βø " ÅÖÜäéé¥

```

Table 5-6
Control code area commands

Function	Control code
Printable code area expansion	<ESC> "6"
Control code area expansion	<ESC> "7"
Select undefined codes as characters	<ESC> "I" 1
Cancel undefined codes as characters	<ESC> "I" 0

■ Printing BIG characters

You can even enlarge your character sets for attention-grabbing headings or special effects. There are six commands you can use. Everything following any of them will be enlarged as shown below, until the cancel code is entered.

Table 5-7
Big character commands

Function	Control code
Double-high enlarged print	<ESC> "h" CHR\$(1)
Quad-high enlarged print	<ESC> "h" CHR\$(2)
Double-high lower-half enlarged print	<ESC> "h" CHR\$(3)
Double-high upper-half enlarged print	<ESC> "h" CHR\$(4)
Quad-high lower-half enlarged print	<ESC> "h" CHR\$(5)
Quad-high upper-half enlarged print	<ESC> "h" CHR\$(6)
Cancel enlarged print	<ESC> "h" CHR\$(0)

Try this program to see the big characters.

```

10 ' Demo of BIG characters
20 FOR I=1 TO 6
30 LPRINT "THIS IS ";
40 LPRINT CHR$(27);"h";CHR$(I);
50 LPRINT "BIG";
60 LPRINT CHR$(27);"h";CHR$(0);
70 LPRINT " CHARACTER." : LPRINT
80 IF I=2 THEN LPRINT
90 NEXT I
100 END

```

When you run this program, you will get like this:

```

THIS IS  BIG  CHARACTER.

THIS IS  BIG  CHARACTER.

THIS IS  L L L  CHARACTER.

THIS IS  D T C  CHARACTER.

THIS IS  L L L  CHARACTER.

THIS IS  D T C  CHARACTER.

```

As you can see, when the big character command is used, the baseline for each character does not align. If you want to align the baseline, try this program:

```

10 ' Demo of aligning BIG characters
20 LPRINT "THIS IS ";
30 LPRINT CHR$(27);"j";CHR$(21);
40 LPRINT CHR$(27);"h";CHR$(1);
50 LPRINT "DOUBLE";
60 LPRINT CHR$(27);"h";CHR$(0);
70 LPRINT CHR$(27);"J";CHR$(18);
80 LPRINT " SIZED PRINTING."
90 LPRINT :LPRINT :LPRINT
100 LPRINT "THIS IS ";
110 LPRINT CHR$(27);"j";CHR$(63);
120 LPRINT CHR$(27);"h";CHR$(2);
130 LPRINT "QUAD";
140 LPRINT CHR$(27);"h";CHR$(0);
150 LPRINT CHR$(27);"J";CHR$(63);
160 LPRINT " SIZED PRINTING." :LPRINT :LPRINT
170 END

```

When you run this program, you will get this:

THIS IS DOUBLE SIZED PRINTING.

THIS IS QUAD SIZED PRINTING.

■ The macro control code

The last of our group of miscellaneous codes is definitely not the least. It is a *user-defined* control code, called a *macro* control code. The term *macro* is from the jargonese *macro-instruction* which refers to an instruction that “calls,” or uses a group of normal instructions. In computer programming macro-instructions (which are similar to subroutines) save programmers a lot of time and effort. Your printer’s macro can save you a lot of time and effort also.

Here is how the printer's macro works. You *define a macro* by telling the printer what normal control codes are to be included in the macro. Then you can use the macro any time that you want and the printer will do all the things that you included in the macro definition. You can include up to 16 codes in a single macro. You can even use the macro to store a frequently used word or phrase. There are two control codes for the macro: one to define it, and one to use it. They are given in the Table 5-8.

Table 5-8
Macro instruction commands

Function	Control code
Define macro	<ESC> "+" ... (<i>codes you include</i>) ... CHR\$(30)
Use macro	<ESC> "+" CHR\$(1)

To see how this works we can build a macro that will reset the printing style to normal, no matter what style it may be to start with. The following program will define a macro to do this.

```

10 LPRINT CHR$(27);"+";           :'Start macro
20 LPRINT CHR$(27);"h";CHR$(0);:'Big character off
30 LPRINT CHR$(27);"!";CHR$(0);:'Select normal pica
40 LPRINT CHR$(27);"T";           :'Super & subscripts
   off
50 LPRINT CHR$(27);"a";CHR$(0);:'Left-aligned
   printing
60 LPRINT CHR$(30)                :'End macro

```

As the comments in the program shown, we started to define the macro in line 10. Line 20 cancels big character printing. Line 30 sets normal pica, and this command also cancels proportional pitch, condensed print, expanded print, boldface, emphasized, italics, and underlining. Line 40 cancels superscripts and subscripts, and line 50 sets left-justified printing. Then, line 60 ends the macro definition. This printer will remember this macro until the power is turned off or until a new macro is defined. A macro can hold up to 16 bytes (characters) of information. The one that we defined contains eleven.

Now that you have defined a macro, let's see how to use it. This program will print one line using several printing features. Then it "calls" the macro in line 60. When line 80 prints the style is "plain vanilla" because the macro has reset it.

```
10 LPRINT CHR$(27);"Q";CHR$(40);
20 LPRINT CHR$(27);"a";CHR$(2);
30 LPRINT CHR$(27);"-1";
40 LPRINT CHR$(27);"h";CHR$(1);
50 LPRINT "TESTING ABCD"
60 LPRINT CHR$(27);"+";CHR$(1);
70 LPRINT "TESTING ABCD"
80 END
```

TESTING ABCD

TESTING ABCD

■ Reading a hex dump

The BASIC in some computers changes ASCII codes before they send them to the printer. If you run into problem because of this, try this hex dump to check the ASCII codes.

First turn off the printer. Hold down both the NLQ key and Paper Feed key and turn on the printer, then run the following program.

```
10 FOR I=0 TO 255
20 LPRINT CHR$(I);
30 NEXT I
40 LPRINT
50 END
```

If your system passes the codes directly to the printer without changing them, you will get a printout like this. (You can print out the last remaining line in the print buffer by putting the printer off line with the On Line key.)

```

00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F .....
10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F .....
20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F .....
30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F 0123456789:;=>?
40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F @ABCDEFGHIJKLMNO
50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F PQRSTUWXYZ[\]^_
60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F `abcdeghijklmno
70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 7E 7F pqrstuvwxyz{|}~.
80 81 82 83 84 85 86 87 88 89 8A 8B 8C 8D 8E 8F .....
90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F .....
A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF .....
B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF .....
C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF .....
D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 DA DB DC DD DE DF .....
E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF .....
F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF .....
0D 0A ..

```

Most BASICs, however, are not quite that straight forward. For example, the IBM-PC prints the following:

```

00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E .....
0F 10 11 12 13 14 15 16 17 18 19 1B 1C 1D 1E 1F .....
20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F .....
30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F 0123456789:;=>?
40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F @ABCDEFGHIJKLMNO
50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F PQRSTUWXYZ[\]^_
60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F `abcdeghijklmno
70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 7E 7F pqrstuvwxyz{|}~.
80 81 82 83 84 85 86 87 88 89 8A 8B 8C 8D 8E 8F .....
90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F .....
A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF .....
B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF .....
C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF .....
D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 DA DB DC DD DE DF .....
E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF .....
F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF .....
0D 0A ..

```

Notice that your printer receives decimal code 13 (hex 0D) together with hex 0A, which is really decimal 10. In addition, your printer does not receive decimal code 26 (hex 1A).

Your printer prints 16 hex numbers per line, printing the characters on the right hand side. If it receives less than 16, it sits in a holding pattern, awaiting more data. Taking the printer off line dumps the characters to the paper.

To debug a program quickly, just use the hex dump capability. Appendix B will help you translate the hex codes to ASCII equivalents.

MEMO

CHAPTER 6

DOWNLOAD

CHARACTERS

AND DOT GRAPHICS

Subjects covered in Chapter 6 include—

- **Designing and printing your own characters**
- **User-designed graphics**

Using the procedures outlined so far, all of the printer's standard functions can be utilized. There are two additional user-defined functions available: designing and printing your own characters, and designing and printing your own dot graphics.

These two functions can be used to design and print any symbol, or any graphic shape. This allows you to make your own special symbols, or print a company logo. Some understanding of dot matrix technology will be useful.

DOT MATRIX PRINTING

Dot matrix printers are named because each character is formed from a group of dots. The print head comprises nine wires or pins stacked vertically. For any standard character, up to seven of the pins are used at one time; for capital letters, the top seven are used, for standard height lower-case characters, the center five pins, and for "descending" characters (such as "p" or "g"), the bottom seven pins are used. The print head moves across the paper and at any instant prints a vertical series of dots; looking at Figure 6-1 will reveal that the letter "p" consists of five vertical columns of dots.

Any pin is fired by only one number, and any group of pins is

fired by a different number. One byte of data (a decimal number from 0 to 255), can thus represent a unique combination of dots. For example, if the data representing “1” was sent to the print head, only the bottom pin would fire; if “2” was sent, only the second pin from the bottom would fire, and if “3” was sent, the two bottom pins would fire, and so on. Columns of these dots side by side make all the characters.

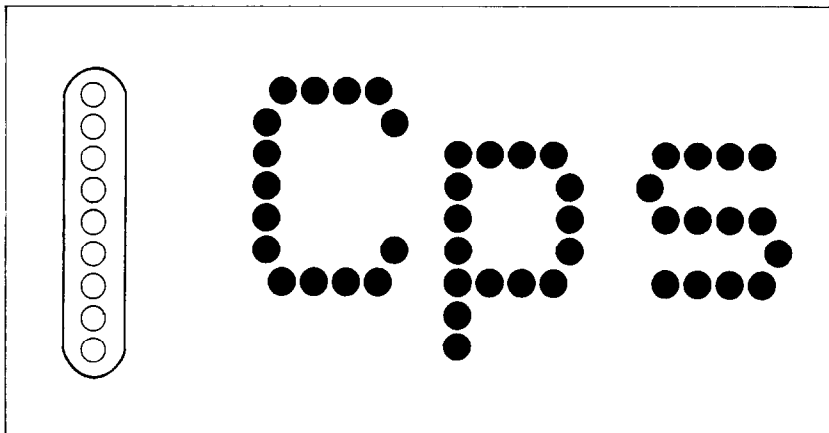


Figure 6-1. As the print head moves across the page, each of the pins prints a row of dots.

DESIGNING YOUR OWN CHARACTERS

Standard characters are permanently stored in the printer's ROM (Read Only Memory), but characters you design are *downloaded* and stored in RAM (Random Access Memory) for use.

Designing and printing your own characters has two requirements: firstly, designing the shape of the character, calculating the data necessary to make the shape, and sending that data to the printer, and secondary, sending the command to print the downloaded characters instead of the standard characters. There are a number of design constraints for download draft quality characters:

- The matrix or grid on which you design the characters is

six boxes wide by nine boxes high.

- Horizontally, dots may be centered in a box, or may straddle a line, making the actual character grid 11 dots wide by 9 dots high. Vertically, dots can only be centered in a box. See Figure 6-2.
- The minimum width of a character is five dots.
- Unlike standard characters that are restricted to a height of seven dots, user-defined characters may use eight dots vertically.
- Dots cannot overlap — that is, you may not have a dot inside a box next to one that is on a line.
- You may define any position between ASCII 32 and 127.

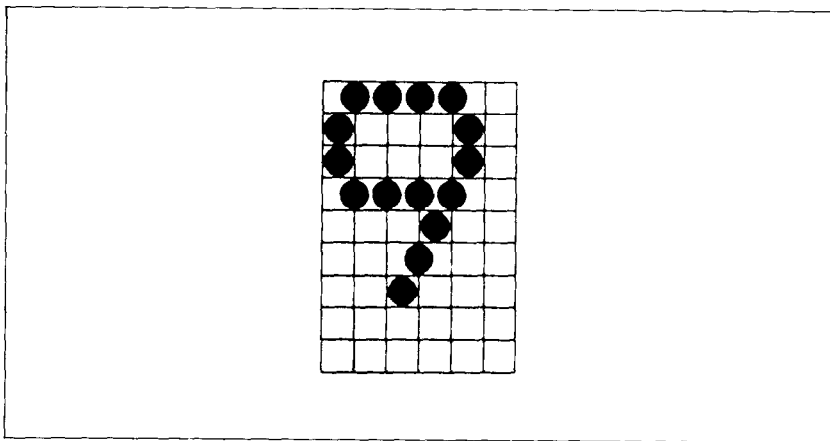


Figure 6-2. Dots can be inside boxes or straddle the vertical lines of the grid.

Photocopy the grid in Figure 6-3 to help design your new characters. We will use a tiny representation of a chemist's flask for our example. We decided that our flask will *not* be a “descender” (printing below the baseline of standard characters), so a figure “1” is written next to Descender on the grid. If your character is a descender, write a “0” next to Descender.

Next we calculate the *vertical* numerical values of the columns of dots, and enter them underneath the grid. For example, looking at Figure 6-4, we see that in the left-most column there is only one dot, and it is sitting in the “2” box. Thus its vertical

value is 2. In the next column, there is a dot straddling the vertical line in the "1" box, and another in the "4" box; its value is recorded as 5.

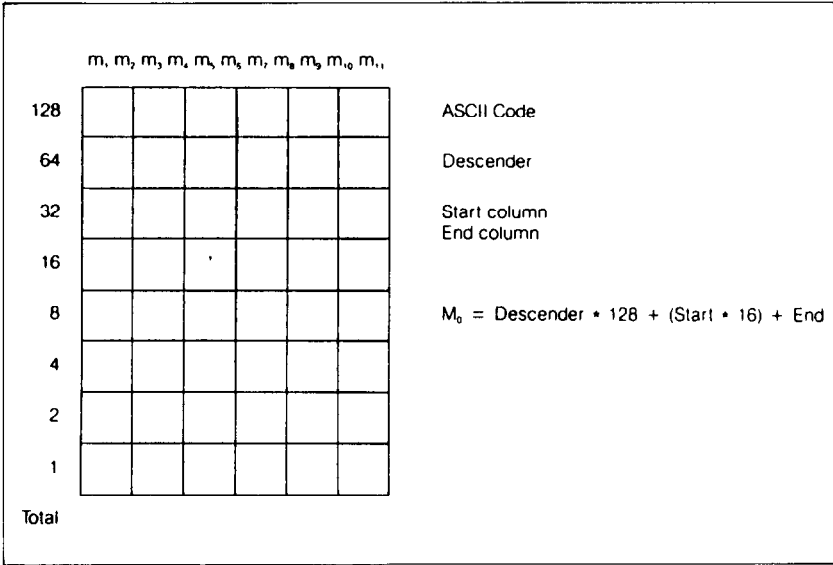


Figure 6-3. Use this grid (or one similar to it) to define your own draft characters.

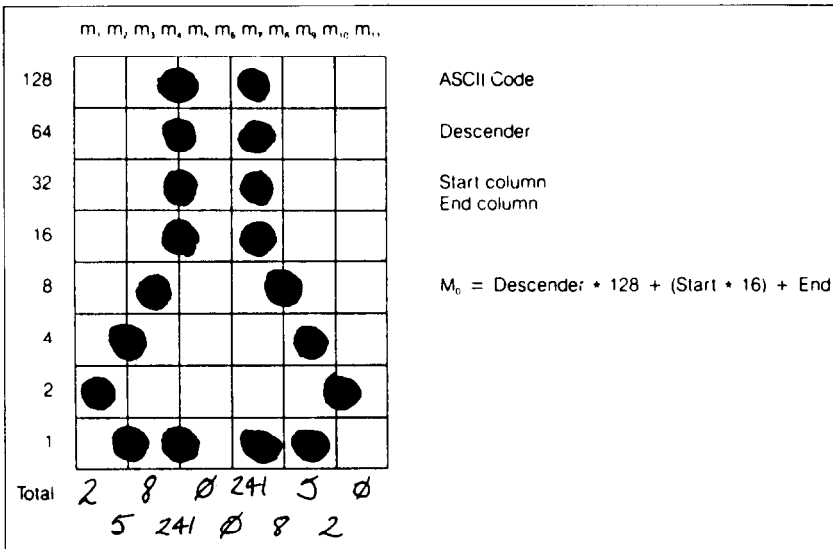


Figure 6-4. Add the values of the dots in each column and write the sum of each column at the bottom.

You will see the letters *m1*, *m2*, *m3* etc., are above your values. When you define the character, you will substitute your value for each of these letters.

Before you complete the definition, you will need to decide exactly where in the standard ASCII set you want to place your characters. After downloading, you access your new character by sending the code for the character you replaced. For example, we decided to assign our flask to the “<” character. You are not restricted in your choice of ASCII codes, so little-used codes may be replaced by your own more useful characters. The complete command syntax for a download character is:

```
<ESC> "&" CHR$(0) n1 n2 m0 m1 m2 m3 m4 m5 m6 m7 m8 m9
m10 m11
```

Where <ESC> is the escape code (CHR\$(27)), the “&” is the ampersand (CHR\$(38)) followed by CHR\$(0), and *n1* and *n2* specify the ASCII codes you want the new characters to occupy. *n1* specifies the beginning of the range of ASCII codes to be defined, and *n2* the end; if you only want to define one character, *n1* and *n2* will be the same.

m0 is called the attribute byte, and describes two attributes: the top or high order bit describes whether the character is a descender, and the last seven bits describe proportional widths. Our character will occupy from column 1 to column 11, so the proportional data is 11. See Figure 6-5. For further detail about proportional width of characters, see the following section.

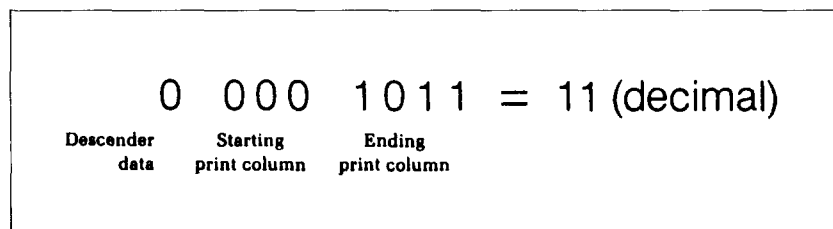


Figure 6-5. The attribute byte (*m0*) for our flask character.

Even though there are no dots in the last column, we define to column 11 to leave some space between the characters.

m1 to *m11* are the column data we defined along the bottom of

the grid; *m1* the left-most column value, *m2* the next, and so on. You must define eleven data, even if the last few data are 0. For the flask, the complete command will be:

```
CHR$(27);CHR$(38);CHR$(0);CHR$(60);CHR$(60);
CHR$(139);CHR$(2);CHR$(5);CHR$(8);CHR$(241);CHR$(0);
CHR$(0);CHR$(241);CHR$(8);CHR$(5);CHR$(2);CHR$(0)
```

Put this command into the appropriate syntax for the language you are using. Before you run the program and send the data to the printer, remember to check that DIP switch 1-1 is set to off.

The last requirement is the command to make the printer use the downloaded characters stored in RAM instead of the standard character set stored in ROM. The command is:

```
<ESC> “%” n
```

Where *n* is either “1” (CHR\$(49), to select), or “0” (CHR\$(48), to cancel). If you want your new characters to be included in a standard character set, you may copy the standard set stored in ROM into the download RAM, which will save you switching back and forth between the download area and the standard ROM. If you wish to use this option, make sure that you copy the standard characters into RAM **before** you download your own, because the last-copied characters replace previously copied ones. The command to copy the standard characters into the download RAM area is:

```
<ESC> “:” CHR$(0) CHR$(0) CHR$(0)
```

To recap, the order of commands if you wish to use various user-defined characters with the standard set is:

- Select the draft characters : <ESC> “x” 0
- Copy standard character set to RAM : <ESC> “:”
CHR\$(0) CHR\$(0) CHR\$(0)
- Define characters, and send data to printer : <ESC> “&”
CHR\$(0)

- Tell printer to use download character set : $\langle \text{ESC} \rangle$ "%"
1

To demonstrate how to use the download characters, let's use the "flask" character and some other user-defined characters to print a small graph. This program will do just that:

```

10 LPRINT CHR$(27);":";CHR$(0);CHR$(0);CHR$(0);
20 LPRINT CHR$(27);"&";CHR$(0);CHR$(60);CHR$(62);
30 FOR N=60 TO 62
40 FOR M=0 TO 11
50 READ MM
60 LPRINT CHR$(MM);
70 NEXT M
80 NEXT N
90 LPRINT
100 DATA 139, 2, 5, 8,241, 0, 0,241, 8, 5,
      2, 0
110 DATA 139,124, 0, 66, 4, 64, 36, 16, 2, 16,
      12, 0
120 DATA 139, 46, 16, 2, 60, 0, 48, 0, 48, 0,
      48, 0
130 LPRINT CHR$(27);"D";CHR$(11);CHR$(0)
140 LPRINT CHR$(27);"h";CHR$(1);
150 LPRINT " U.S. EXPORTS"
160 LPRINT CHR$(27);"h";CHR$(0);
170 LPRINT CHR$(27);"%1";
180 LPRINT "AUTOS";CHR$(9);
190 FOR I=.4 TO 9.3 STEP .4
200 LPRINT CHR$(61);
210 NEXT I
220 LPRINT
230 LPRINT "CHEMICALS";CHR$(9);
240 FOR I=.4 TO 8.7 STEP .4
250 LPRINT CHR$(60);
260 NEXT I
270 LPRINT
280 LPRINT "GUNS";CHR$(9);
290 FOR I=.4 TO 1.4 STEP .4
300 LPRINT CHR$(62);
310 NEXT I

```


DESIGNING YOUR OWN NLQ CHARACTERS

As you have probably noticed, NLQ characters are printed by two passes of the print head. Half of the character is printed on the first pass, and the remainder on the second pass. The paper is rolled up half a dot height in between passes to let the print head print dots on the second pass that overlap the previous dots, in order to fill in the spaces and produce denser characters.

Fundamentally, the process to define and print downloaded NLQ characters is the same as for draft characters, except that you must design the character with the two passes of the print head in mind.

Use the grid in Figure 6-6 to help plot the data. Due to the two-pass process, NLQ characters can contain up to 16 dots vertically. *m1* to *m11* are the data for the first pass, and *m12* to *m22* are the data for the second pass.

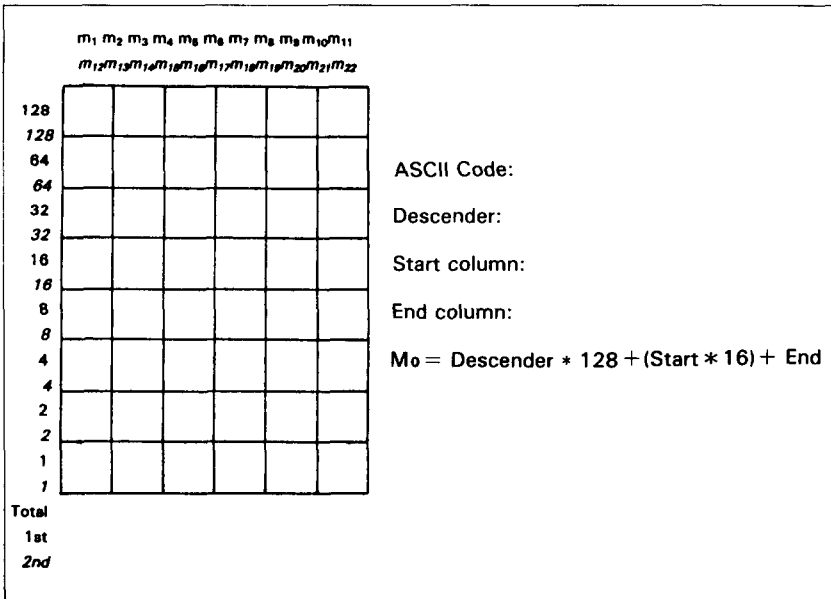


Figure 6-6. Use this grid (or one similar to it) to define your own NLQ characters.

To make the character data easier to calculate, put the first pass dots inside the boxes on the grid when you are designing

your characters, and the second pass dots on the lines. Remember that adjacent dots may not overlap on the same pass; in other words, if one dot is in a box, its adjacent dot on the same pass may not be on a line. Therefore, to make smooth characters, put the dots needed to overlap any spaces in a character in the appropriate places in the second pass grid.

If we replot our flask for NLQ, the grid data will look like Figure 6-7.

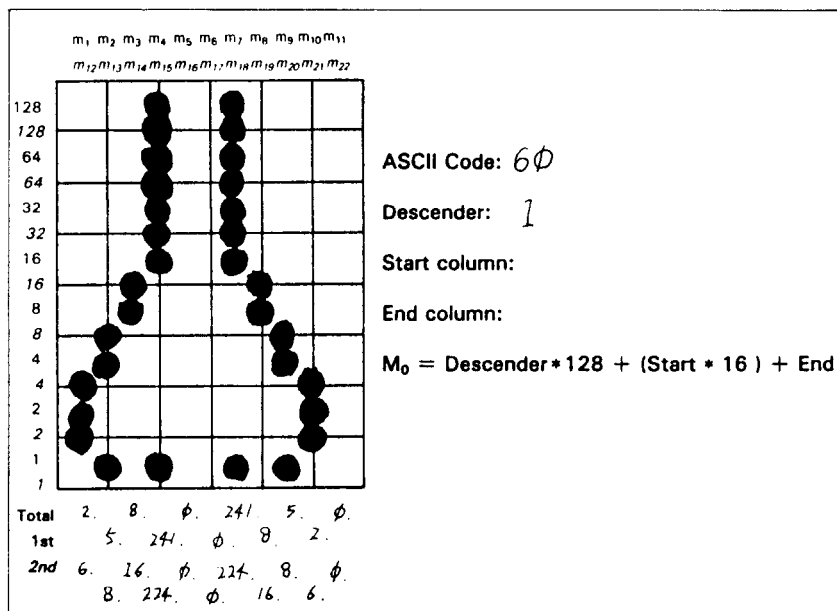


Figure 6-7. Add the values of the dots in the boxes first, and then add the values on the lines.

The defining process is the same as for draft characters, except that you must select NLQ characters by using $\langle \text{ESC} \rangle$ "x" 1 first to select the NLQ mode, and you must define 22 data (m_1 to m_{22}). If you wish to print your NLQ characters with the standard character set, the remarks regarding the simultaneous use of standard and download characters in the draft character section apply equally to NLQ characters.

To demonstrate how to use the NLQ download characters, try this program.

```
10 LPRINT CHR$(27);"x1";
20 LPRINT CHR$(27);":";CHR$(0);CHR$(0);CHR$(0);
30 LPRINT CHR$(27);"&";CHR$(0);CHR$(60);CHR$(62);
40 FOR N=60 TO 62
50 FOR M=0 TO 22
60 READ MM
70 LPRINT CHR$(MM);
80 NEXT M
90 NEXT N
100 LPRINT
110 DATA 139, 2, 5, 8,241, 0, 0,241, 8, 5,
    2, 0
120 DATA 6, 8, 16,224, 0, 0,224, 16, 8, 6,
    0
130 DATA 139,124, 0, 66, 4, 64, 36, 16, 2, 16,
    12, 0
140 DATA 120, 4, 0, 4, 0, 96, 4, 0, 4, 8,
    0
150 DATA 139, 46, 16, 2, 60, 0, 48, 0, 48, 0,
    48, 0
160 DATA 92, 32, 4, 60, 0, 48, 0, 48, 0, 48,
    0
170 LPRINT CHR$(27);"D";CHR$(11);CHR$(0)
180 LPRINT CHR$(27);"h";CHR$(1);
190 LPRINT " U.S. EXPORTS"
200 LPRINT CHR$(27);"h";CHR$(0);
210 LPRINT CHR$(27);"%1";
220 LPRINT "AUTOS";CHR$(9);
230 FOR I=.4 TO 9.3 STEP .4
240 LPRINT CHR$(61);
250 NEXT I
260 LPRINT
270 LPRINT "CHEMICALS";CHR$(9);
280 FOR I=.4 TO 8.7 STEP .4
290 LPRINT CHR$(60);
300 NEXT I
310 LPRINT
320 LPRINT "GUNS";CHR$(9);
330 FOR I=.4 TO 1.4 STEP .4
340 LPRINT CHR$(62);
350 NEXT I
360 LPRINT
```

```

370 LPRINT CHR$(9);"+--";
380 SCALE$="--+--"
390 FOR I=2 TO 8 STEP 2
400 LPRINT SCALE$;
410 NEXT I
420 LPRINT "--+"
430 LPRINT CHR$(9);" ";
440 FOR I=2 TO 8 STEP 2
450 LPRINT " ";I;
460 NEXT I
470 LPRINT CHR$(27);"%0"
480 LPRINT CHR$(27);"x0";
490 LPRINT CHR$(27);"s0";
500 LPRINT CHR$(9);"MILLIONS OF DOLLARS"
510 LPRINT CHR$(27);"T"
520 END

```

U.S. EXPORTS

```

AUTOS      @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
CHEMICALS  ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
GUNS       - - - -
           +-----+-----+-----+-----+-----+
                   2     4     6     8
           MILLIONS OF DOLLARS

```

USER-DESIGNED DOT GRAPHICS

Designing and printing graphics is similar to the procedures for characters, except that the printed width can range from one to 1920 dots. Selection of the appropriate line spacing will allow rows of dots to link seamlessly, and in this way you may design and print a whole page of data. In the dot graphics mode, you directly address all dot positions on the page if desired. The graphics mode can print as many as 240 dots per inch compared with the standard resolution of 60 dots per inch (dpi).

The command to print normal density (60 dpi horizontally, 72 dpi vertically) uses this format:

```
<ESC> "*" CHR$(0) n1 n2 m1 m2 .....
```


⟨ESC⟩ “*” specifies the graphics mode, the CHR\$(0) specifies normal density. Other densities are described later. Any number of graphics data may follow; the *n1* and *n2* specify the number of bytes.

To figure the value of *n1* and *n2*, you need to work out how wide your image will be. Because you are limited to the largest number that can be sent in one byte (255), the formula to work out *n1* and *n2* is:

if the number of columns is X,
then $n1 = X \text{ MOD } 256$, and $n2 = \text{INT}(X/256)$

Table 6-1 may make the calculation easier.

Table 6-1
Calculating *n1* and *n2*.

If the number of columns, x, ranges from:	Then <i>n1</i> is:	and <i>n2</i> is:
1 to 255	x	0
256 to 511	x-256	1
512 to 767	x-512	2
768 to 1023	x-768	3
1024 to 1279	x-1024	4
1280 to 1535	x-1280	5
1536 to 1791	x-1536	6
1792 to 1920	x-1792	7

m1, *m2* and so on are specified as before, that is, any desired combination of pins will be fired by just one number. Because the top eight pins are used in the dot graphics mode, numbers between 0 and 255 (in other words, one byte) will fire a unique combination of pins to produce any pattern of dots. Thus *m1*, *m2* and so on are the data bytes of the command.

To plot large graphics, use graph paper and draw a line every eight rows. Draw your graphics shape, and then calculate the values of each vertical column of eight dots; each value becomes one data byte. When drawing large shapes, the data byte CHR\$(255) (all pins firing, or a solid block of dots) is common.

A line spacing of 8/72 inch (using the ⟨ESC⟩ “A” CHR\$(*n*), where *n*=8), will ensure that successive rows of dots will join perfectly.

HIGHER RESOLUTION GRAPHICS

There are seven graphics modes you can use; they are summarized in Table 6-2.

Table 6-2

Function	Control code
Normal density (60 dots/inch)	$\langle \text{ESC} \rangle "*" \text{CHR}\$(0) n1 n2 m1 m2 \dots$
Double density (120 dots/inch)	$\langle \text{ESC} \rangle "*" \text{CHR}\$(1) n1 n2 m1 m2 \dots$
Double density/double speed	$\langle \text{ESC} \rangle "*" \text{CHR}\$(2) n1 n2 m1 m2 \dots$
Quadruple density (240 dots/inch)	$\langle \text{ESC} \rangle "*" \text{CHR}\$(3) n1 n2 m1 m2 \dots$
Semi-double density graphics (80 dots/inch)	$\langle \text{ESC} \rangle "*" \text{CHR}\$(4) n1 n2 m1 m2 \dots$
Plotter graphics (72 dots/inch)	$\langle \text{ESC} \rangle "*" \text{CHR}\$(5) n1 n2 m1 m2 \dots$
CRT graphics (90 dots/inch)	$\langle \text{ESC} \rangle "*" \text{CHR}\$(6) n1 n2 m1 m2 \dots$

You will notice that the higher the density of the graphic mode, the more points you will need to plot. Higher density modes also take longer to print. The double density double speed mode ignores every second column of dots to give you a quick printout for debugging purposes; you can then change to true double density graphics for the final printout.

If your software cannot access the $\langle \text{ESC} \rangle "*" \text{CHR}\(n) command, there are individual letter escape commands for four of the modes. See Table 6-3.

Table 6-3
Alternative graphics commands

Density	Single command	Individual command
Noram1	$\langle \text{ESC} \rangle "*" \text{CHR}\$(0) n1 n2 m1 m2 \dots$	$\langle \text{ESC} \rangle "K" n1 n2 m1 m2 \dots$
Double	$\langle \text{ESC} \rangle "*" \text{CHR}\$(1) n1 n2 m1 m2 \dots$	$\langle \text{ESC} \rangle "L" n1 n2 m1 m2 \dots$
Double with double-speed	$\langle \text{ESC} \rangle "*" \text{CHR}\$(2) n1 n2 m1 m2 \dots$	$\langle \text{ESC} \rangle "Y" n1 n2 m1 m2 \dots$
Quadruple	$\langle \text{ESC} \rangle "*" \text{CHR}\$(3) n1 n2 m1 m2 \dots$	$\langle \text{ESC} \rangle "Z" n1 n2 m1 m2 \dots$
Semi-double	$\langle \text{ESC} \rangle "*" \text{CHR}\$(4) n1 n2 m1 m2 \dots$	none
Plotter	$\langle \text{ESC} \rangle "*" \text{CHR}\$(5) n1 n2 m1 m2 \dots$	none
CRT	$\langle \text{ESC} \rangle "*" \text{CHR}\$(6) n1 n2 m1 m2 \dots$	none

MORE GRAPHICS PROGRAMMING TIPS

At the end of this chapter, we'll discuss two modes that the printer offers to help you solve potential graphics problems. A redefining code allows you to change the density for graphics programs that use one of the four alternate codes. The 9-pin graphics mode allows you to use all nine pins on each line and thus speed up screen dumps.

Table 6-4
More graphics commands

Function	Control code
Redefine the graphics mode	<code><ESC>"?" n0 n1</code>
9-pin graphics	<code><ESC>"^" n0 n1 n2 m1 m2</code>

■ Redefining alternate graphics codes

Your printer provides a command to redefine one of the alternate graphics modes — K, L, Y, or Z — so that it represents any other of the seven graphics modes. The command is `<ESC>"?" n0 n1`, where *n0* is one of the four letters, "K", "L", "Y", or "Z", and *n1* is one of the numbers used with the `<ESC>"*` command, 0 to 6. There are several instances in which you may use this sequence.

The first occurs if you have written a program to be printed in one graphics mode and now want to print it in another. If you have used concatenation to store your graphics command in one short character string, that will not be difficult. You can simply change the mode number or alternate code in the definition of the character string.

A second time you can make good use of the redefining code occurs when you want to change a program in which you have not concatenated the graphics codes. Using the `<ESC>"?"` sequence allows you to change every instance of your graphics command by entering only one line.

■ 9-pin graphics mode

In the early part of this chapter, we said that the bottom pin of the print head is not normally used in the graphics modes. That's because most computers communicate with parallel-type

peripheral devices using eight data lines. When the peripheral is a printer, each data line corresponds to one pin on the print head. Thus each byte sent will fire up to eight pins.

But the printer has 9 pins available. So how do you fire the ninth pin with only 8 data lines? In fact, do you really want to bother with just one extra pin? Well, for such graphics-intensive applications as screen dumps, printing 9 pins at a time can speed up the process considerably. For this purpose, your printer has a special 9-pin graphics mode (it won't, however, work with 7-bit interface systems). In this mode the printer takes 2 bytes to fire all 9 pins as shown in Figure 6-8.

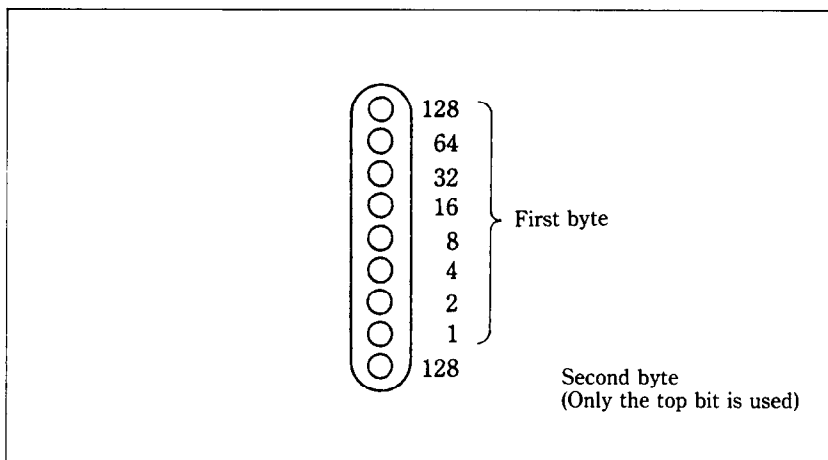


Figure 6-8. The printer needs 2 bytes to fire all 9 pins in the 9-pin graphics mode.

In addition, you can select the print density by the value of *n0*. When *n0* is 0 the normal density is selected, and when *n0* is 1 the double density is selected.

Since computers are faster than printers, there is no significant time loss in printing a single line of graphics with 9 pins. You get 9 dots per line in about the same time as you get 8 dots in the other graphics modes.