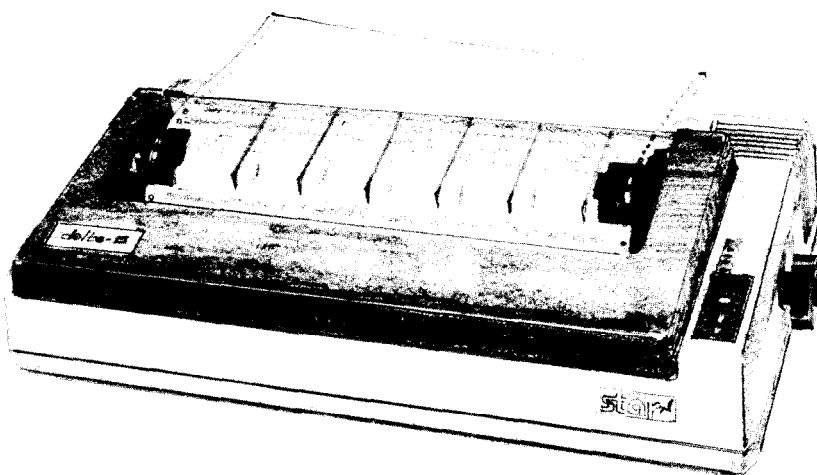


Delta User's Manual



starTM
MICRONICS • INC

THE POWER BEHIND THE PRINTED WORD.

NOT INTENDED FOR SALE

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.

A note about the programs in this manual:

This manual contains several programs that help to demonstrate the versatility of the Delta printers. Star Micronics has made every effort to insure that the programs are functional and accurate. However, Star Micronics cannot guarantee their accuracy or suitability to any particular application.

Trademark Acknowledgement

Delta-10, Delta-15, grafstar, Universal/Atari Parallel Interface, Universal/Commodore Parallel Interface: Star Micronics

Apple, Apple II, Apple II + , Apple IIe, Applesoft: Apple Computer Inc.

Atari 400, Atari 800, Atari 850: Atari Inc., a Warner Communications Company

Commodore, VIC-20, C-64: Commodore Business Machines, Inc.

Compaq: Compaq Computer Corporation

CP/M: Digital Research

EasyWriter: Information Unlimited Software, Inc.

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

Kaypro: Kaypro Computer Corporation

Microsoft BASIC: Microsoft Corporation

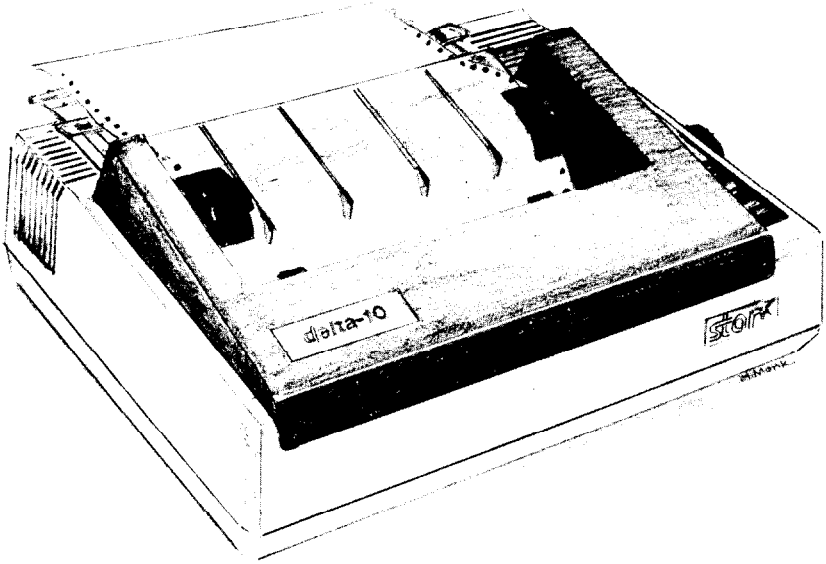
Osborne 1: Osborne Computer Corporation

SuperCalc: Sorcim Corporation

TRS-80: Radio Shack, a division of Tandy Corporation

WordStar: MicroPro International Corporation

©Copyright 1983 Star Micronics, Inc.



A Special Message to the New Owner

Your new Delta printer and this manual are both setting new standards for the computer printer industry — and you're part of it! Congratulations, and welcome aboard!

First, about this manual. It's another first in our industry — the first to be truly written not just for the person who does his own programming, but for the first-time user or anyone else who prefers to leave the programming to others, and simply inserts his store-bought programs (software) into his computer/printer system. Someone very much like you, perhaps . . .

You'll find using this manual easy and pleasant. We've gone to great lengths to make it so, as it's master-minded by solid experts in the arcane art of computer science, and written by equally proficient practitioners in the art of Plain English!

As a first example, look over the Table of Contents and you'll see what we mean. Whether you're a greenhorn or a wizard, everybody will find what they need to know to fulfill their expectations. We suggest that each new owner/user, before you even unpack the box, read or at least scan Chapters 1 and 2 — "A Closer Look" and "Getting Started with Delta" — as well as Appendix A, "Unpacking and Installation." Now you can unpack the box and start putting things together.

When you're ready to connect up your computer to your Delta, look at Appendices B through G for directions applying to your make of computer. Remember, Delta has both serial and parallel interfaces, so there's nothing extra to buy!

Which leads naturally to a few words of praise for some other special features that make the new Delta so satisfactory to own. Features like the high speed 160-character per second printout, the capability to design your own characters, do your own plotting, your own infinite variety of dot graphics patterns and densities. You'll have a ball! For you, Chapters 3 through 8 are a must, and of course everybody should look at Chapter 10 which tells how to maintain your Delta for a long and carefree life.

We'll end this as we began, with congratulations for your wise buying, and a most cordial welcome to the wonderful world of Delta printing. . . fast, clean and beautiful!

Table of Contents

Chapter 1	A Closer Look	1
	Components and Controls	
	Paper Selection and Loading	
	Loading single sheets	
	Loading roll paper	
	Loading sprocket-feed paper	
	Bottom feeding Delta-15	
	Ribbon Installation	
	Adjusting the Gap	
	Self-Test	
Chapter 2	Getting Started With Delta	17
	Using Commercial Software	
	First, some terminology	
	Using Delta with SuperCalc	
	Using Delta with word processors	
	Using this book without learning BASIC	
	Some Basics about BASIC	
	Establishing communications	
	The CHR\$ function	
	Control codes	
	The escape code	
	Some problem codes	
Chapter 3	Printing Text With Delta	29
	Changing the print pitch	
	Expanded print	
	Making Delta print darker	
	Some Special Kinds of Text	
	Italic printing	
	Underlining	
	Superscripts and subscripts	
	Mixing modes	
	Summary	
Chapter 4	Line Spacing and Forms Control	43
	Starting New Lines	

	Changing Line Spacing	
	Moving down the page without a carriage return	
	Forms Controls	
	Form feed	
	Changing the Page Length	
	Top and Bottom Margins	
	Summary	
Chapter 5	Formatting Your Output	55
	A one-shot tab command	
	Setting Left and Right Margins	
	Using Vertical Tabs	
	A one-shot vertical tab command	
	Summary	
Chapter 6	Special Features of the Delta Printer	61
	Now hear this	
	Initializing Delta	
	Putting Delta to sleep	
	Printing to the bottom of the sheet	
	Unidirectional printing	
	Backspace and delete	
	The seven bit dilemma	
	Block graphics characters and special symbols	
	International character sets	
	The macro control code	
	Summary	
Chapter 7	Creating Your Own Characters	73
	Dot Matrix Printing	
	The Print Matrix	
	Defining Your Own Characters	
	Rule 1: Download characters are seven dots high	
	Rule 2: Dots cannot overlap	
	Add up each column of dots	
	Assigning a value to your character	
	Download character definition command	
	Printing Download Characters	
	Proportional Characters	
	Defining proportional characters	
	Printing proportional characters	
	Connecting characters	

	Mixing Print Modes with Download Characters	
	A Utility Program	
	Summary	
Chapter 8	Printing With Dot Graphics	99
	Comparing dot graphics with download characters	
	Using the Dot Graphics Commands	
	Specifying the number of columns of dots	
	Specifying the graphics data	
	Combining text and graphics	
	Printing a Design or Logo	
	Plotting with Delta	
	How the program works	
	Using Delta for business graphics	
	High Resolution Graphics	
	If You Have Problems with BASIC	
	Summary	
Chapter 9	Getting It All Together	119
Chapter 10	Maintenance	123
	Cleaning Delta	
	Removing the Upper Case	
	Replacing a Fuse	
	Replacing the Print Head	
Appendix A	Setting Up Delta	131
	Where shall we put it?	
	What have we here?	
	Removing the shipping screws	
	Removing the packing from inside the printer	
	Installing the platen knob	
	Removing the tractor unit	
	Attaching the paper separator and paper guide	
	Installing the ribbon	
	Installing the printer cover	
	Connecting Delta to your computer	
Appendix B	IBM Personal Computer and Compaq	139
	Recommended DIP switch settings for IBM-PC	
	BASIC programming	

	Listing programs Program listings	
Appendix C	Apple II Computers	143
	Setting the switches Applesoft BASIC Listing programs Program listings Chart program Special character chart program Macro program Bridge hand program Numeral program Download utility program Delta plot program Pie chart program	
Appendix D	TRS-80 Computers	161
	Recommended DIP switch settings for TRS-80 TRS-80 BASIC Chart program Special character chart program Macro program Bridge hand program Numeral program Download utility program Delta plot program Pie chart program	
Appendix E	Osborne, Kaypro and CP/M Computers	177
	Setting the switches Using MBASIC Program listings Chart program Special character chart program Macro program Bridge hand program Numeral program Download utility program Delta plot program Pie chart program	
Appendix F	Atari 400/800 Computers	193
	Setting the switches Using Atari BASIC Listing programs	

	Program listings	
	Chart program	
	Special character chart program	
	Macro program	
	Bridge hand program	
	Numeral program	
	Download utility program	
	Delta plot program	
	Pie chart program	
Appendix G	Commodore VIC-20 and C-64 Computers	211
	Setting the switches	
	Using Commodore BASIC	
	Listing programs	
	Program listings	
	Chart program	
	Special character chart program	
	Macro program	
	Bridge hand program	
	Numeral program	
	Download utility program	
	Delta plot program	
	Pie chart program	
Appendix H	DIP Switch Settings	227
	Switch Functions	
Appendix I	ASCII Codes	233
Appendix J	Character Style Charts	239
Appendix K	Function Code Reference	251
	Commands to Control Print Style	
	Font style controls	
	Font pitch controls	
	Special print modes	
	Commands to Control Vertical Position of Print Head	
	Line feed controls	
	Form feed controls	
	Vertical tabs	
	Commands to Control Horizontal Position of Print Head	
	Download Character Commands	
	Commands to Control Graphics	
	Macro Instruction Commands	
	Other Commands	

Appendix L	Command Summary in Numeric Order	279
Appendix M	ASCII Code Conversion Chart	283
Appendix N	Technical Specifications	291
Appendix O	The Parallel Interface	293
	Functions of the Connector Signals	
Appendix P	Serial Interface Specifications	297
	Configuring the Serial Interface	
	Delta's Serial Protocols	
	Serial busy protocols	
	XON/XOFF protocol	
	ACK protocol	
Index		303
Consumer Response		309
DIP Switch Quick Reference		311
Warranty	Inside back cover	
Command Quick Reference	Inside back cover	

Table of Tables

Table 3-1	Print pitch commands	30
Table 3-2	Expanded print commands	32
Table 3-3	Print emphasis commands	33
Table 3-4	Italic commands	34
Table 3-5	Underline commands	35
Table 3-6	Superscript and subscript commands	36
Table 4-1	Line spacing commands	46
Table 4-2	Form length commands	49
Table 4-3	Top and bottom margin commands	50
Table 5-1	Left and right margin commands	57
Table 6-1	Bell commands	62
Table 6-2	Some miscellaneous commands	63
Table 6-3	Printing direction	64
Table 6-4	Eighth bit controls	66
Table 6-5	International character set commands	68
Table 6-6	International character sets	69
Table 6-7	Macro instruction commands	69
Table 7-1	Download character definition commands	88
Table 7-2	Download character printing commands	91
Table 7-3	Mixing download characters with various print modes	95
Table 8-1	Calculating n1 and n2 for graphics	101
Table 8-2	Dot graphics commands	115
Table B-1	IBM serial switch settings	141
Table H-1	DIP switch settings	228
Table H-2	International character sets	231
Table O-1	Delta parallel interface	295
Table P-1	Serial interface pin functions	298
Table P-2	DIP switch 3	298
Table P-3	Handshaking protocols	300
Table P-4	Data transfer rates	300



Chapter 1

A Closer Look

In this chapter, we'll introduce you to your Delta printer. We'll cover:

- Components and controls
- Paper selection and loading
- Adjusting the gap—for different paper thickness
- Self-test—print-out of available characters

Components and Controls

First, the components. You saw most of these when you unpacked your printer. Now we'll give you a brief explanation of

what they do. (For directions on how to set up Delta, see Appendix A.)

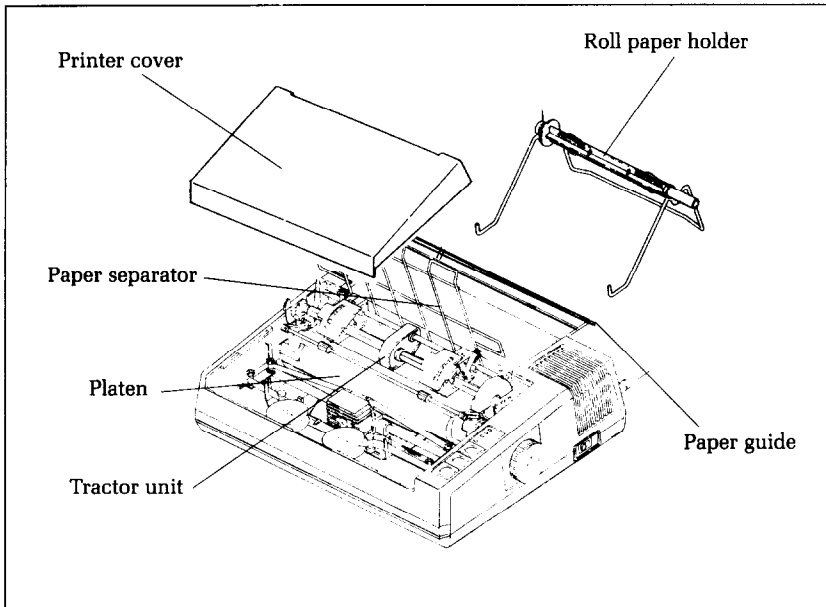


Figure 1-1. For instructions on attaching the various components, see Appendix A.

Printer cover — protects ribbon and print head from dust and dirt — and also reduces the sound level.

Paper separator and paper guide — used with roll paper and sprocket-feed paper.

Roll paper holder and holder shaft — used only with roll paper.

Tractor unit — feeds sprocket-feed paper with its drive gear and sprocket units.

Platen — this is the rubber cylinder that carries paper to the print head.

Now let's take a tour around the controls. You'll find that all of the operating controls are on the right side of the printer.

On/off power switch — towards the backside. This turns on the electricity to your machine.

Platen knob — middle, right side. Lets you manually turn the platen, just like a typewriter.

CAUTION: Turn this knob only with power switch off. Turning it with the power on could damage the platen drive gears.

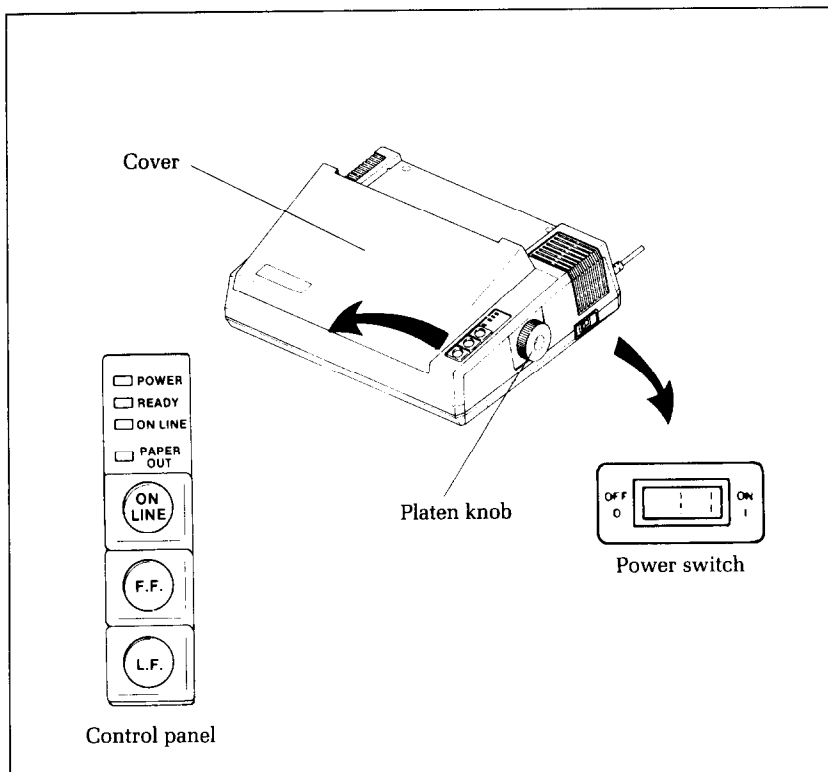


Figure 1-2. All of Delta's controls are on the right side.

Control panel — on top right corner. These three buttons and four “status” lamps are your day-to-day operational controls. Here's what they do:

Power lamp — glows green when the power is on.

Ready lamp — glows green when the printer is ready to accept data. This light flickers during transmission. Don't worry about the flicker; it's normal!

On Line lamp — glows green when data transmission is possible.

Paper Out lamp — glows red when printer is out of paper and stops printing.

On Line button — lets you change the “mode” from on-line to off-line. When it's on-line, the printer can receive data from the computer. When it's off-line, you can advance the paper with the form feed and line feed buttons.

F.F. button — stands for “Form Feed.” When you're off-line you can tap this button and advance the paper to the top of a new page or “form.”

L.F. button — stands for “Line Feed.” When you’re off-line this allows you to advance the paper one line at a time. If you hold the button down, you’ll get multiple line feeds, one after the other.

Around the backside are some important components and connectors. From right to left, they are:

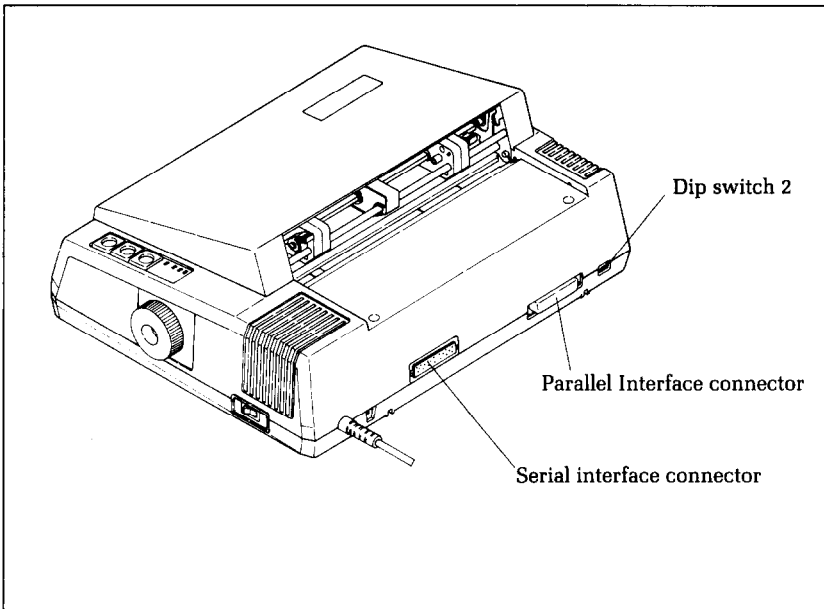


Figure 1-3. Around on the back of Delta you’ll find the interface connectors.

DIP switches — primarily, these switches are used in interfacing the Delta printer to your particular brand of computer. See the appendix for directions on doing this.

Parallel interface connector — the place where you “hook up” your computer to the Delta so they are “interfacing” and thus able to communicate with each other.

Serial interface connector — this interface allows you to connect Delta with a computer using serial communications.

Power cord — you know what it is for, don’t you? It furnishes the electrical power to run the printer.

Paper Selection and Loading

That’s it for components and connectors. The next thing we’ll look at is the variety of papers available for Delta, and how to load

them, ready to print. For starters, Delta can handle single sheets—whether standard-size stationery, envelopes, multi-part carbonless business forms, or almost any other kind of individual sheets. You can also print on continuous paper—either in rolls or fan-folded perforated paper.

Here's a good place to spend a minute talking about the release lever, which you'll be using often. This lever controls the pressure of the paper against the platen. It has two settings — "F" and "T".

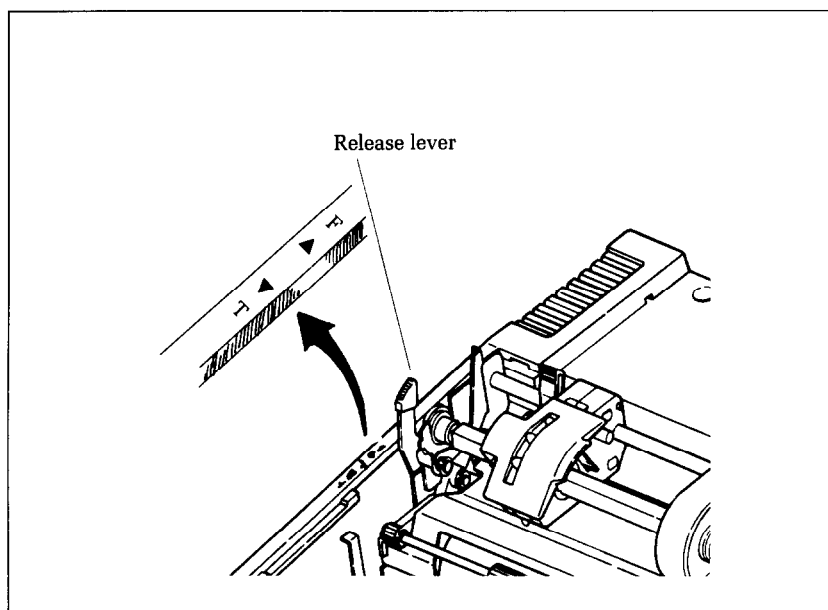


Figure 1-4. The paper release lever has two settings: "F" for friction feed and "T" for tractor feed.

The "F" setting stands for "Friction Feed" and this setting is always used when running single sheets or roll paper. The "T" position stands for "Tractor Feed" and is used only with sprocket-feed paper. "F" tightens the pressure of the paper against the platen, while "T" loosens this pressure, so it's easier to move the paper around.

Loading single sheets

Paper width must always be between 8 and 10 inches (8 and 15 inches for the Delta-15), and paper thickness between .07 mm and .10 mm.

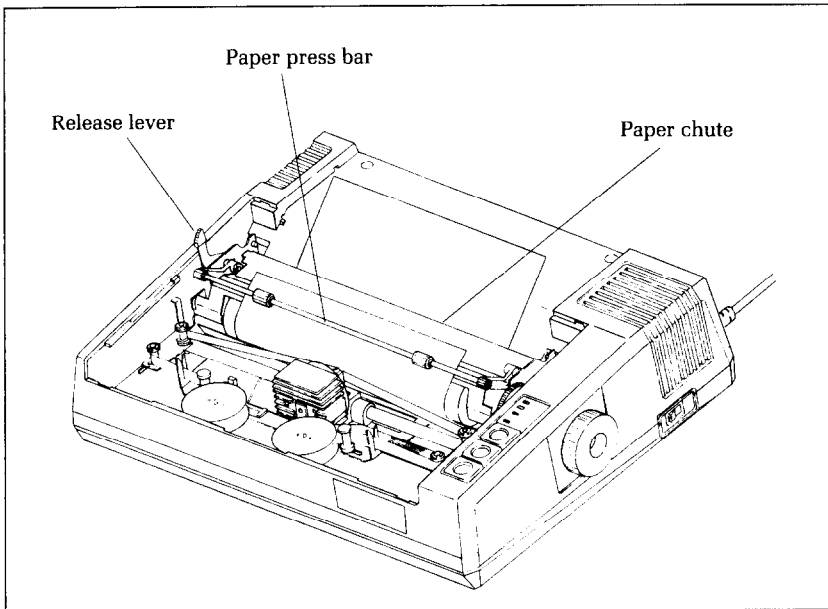


Figure 1-5. Inserting a single sheet of paper can be done "under power" with the line feed button.

Now, instead of rolling the paper in *manually* by turning the platen knob, we're going to use the L.F. button, with the power switch turned on. (This means we'll have to use the "F" (friction feed) position of the release lever.) Remember what we told you about that L.F. (line feed) button? This allows you to advance the paper one line at a time, and if you hold the button down, you'll get multiple line feeds, one after the other.

OK? Now let's start.

1. Remove the printer cover and tractor unit (you can leave the paper guide and paper separator on if you have installed them).
2. Turn the power switch on.
3. Lift up the paper press bar.
4. Set the release lever to the "F" position.
5. Insert the sheet from the back side of the platen (between the paper chute and the platen cover plate).
6. Press the ON LINE button until the ON LINE light goes off.
7. Press the L.F. button to roll the paper in until it appears on the front side of the platen, about where you want the first line to start printing.

NOTE: To straighten paper (if it's in crooked):

- Move the release lever to "T" position.
 - Position the sheet where you want it, moving right or left if necessary to get the paper located between the margins of the printing area.
 - Move release lever back to "F" position.
8. Push the paper press bar back to its original position, flush against the paper.
 9. Replace the printer cover.
 10. Presto! You're ready to start printing!

Loading roll paper

Roll paper, like single sheets, is fed into the printer by "friction feed," using the platen as motive power. Thus, when using roll paper, you must first remove the tractor unit. However, you will need the three components of paper separator, paper guide, and roll paper holder in place. Appendix A tells you how to install the first two. We'll explain here how to attach the roll paper holder and shaft.

The paper holder is (surprise!) the rack that holds the roll of paper. It is inserted into the two holes that you'll find in the back of the printer. (On the Delta-15, the holder attaches the same way, but instead of at the middle, it's over to one side, away from the electrical power cord. The roll of paper is placed on the holder shaft and mounted on the wire rack holder as shown in Figure 1-6.

Roll paper specs are the same for both Delta-10 and Delta-15 (8½" wide, .07 to .10 mm thickness, and maximum 5" diameter roll).

Let's start to load the Delta. It's done almost the same way as loading single sheets, except that the "single sheet" in this case is quite long!

1. Remove the printer cover and tractor unit.
2. Turn the power switch on.
3. Lift up the paper press bar.
4. Set the release lever in the "F" position (Figure 1-5).
5. Pull the paper separator upright (Figure 1-7).
6. Load paper roll onto wire rack holder, so that the paper unrolls toward the printer from the *bottom* of the roll.
7. Unroll some paper, and pass it above the paper guide and beneath the up-ended paper separator.
8. Insert the end of the roll into the paper chute, located at the back side of the platen.
9. Press the ON LINE button until the ON LINE light is off.

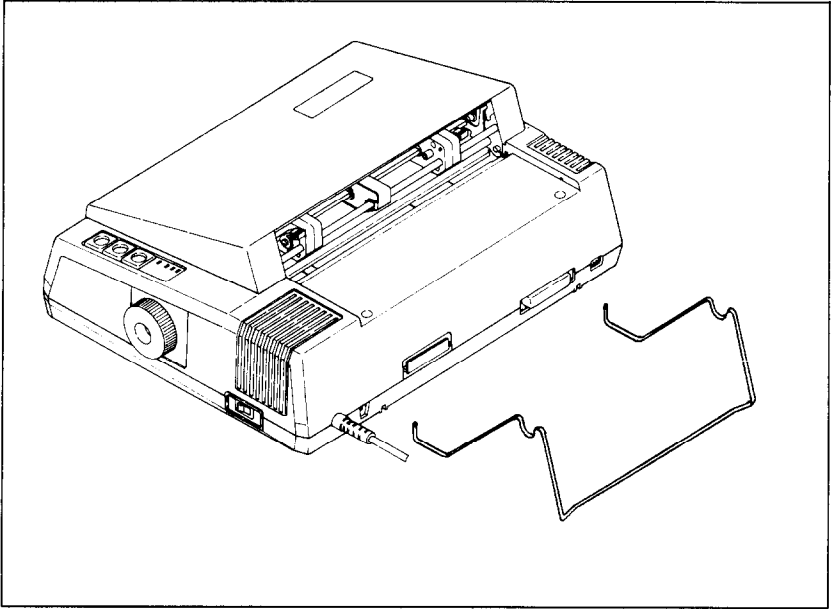


Figure 1-6. The roll paper holder is attached to the back of Delta.

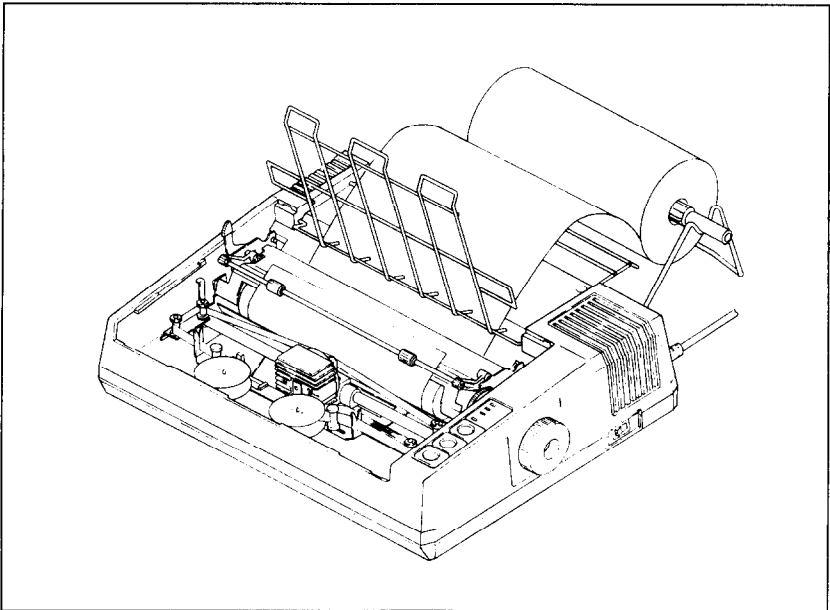


Figure 1-7. Inserting roll paper into Delta is similar to loading single sheets.

10. Press the L.F. button to move the paper in until the leading edge appears on the front side of the platen, about where you want the first line to start printing.
NOTE: To straighten roll paper (if it's in crooked):
 - Move the release lever to the "T" position.
 - Position the sheet where you want it, moving it right or left if necessary to get the paper located between the margins of the printing area.
 - Move the release lever back to "F" position.
11. Push the paper press bar back to its original position, flush against the paper.
12. Replace the printer cover.
13. Presto! You're ready to start printing!

Loading sprocket-feed paper

This is the familiar perforated paper, with the holes along both sides, also called sprocket, punched, fan-fold, or just plain "computer paper." It can be as narrow as 3", and up to 10" wide (5" to 15½" on Delta-15).

To use this kind of paper, you'll need to install the tractor unit, with its two "sprocket" wheels to carry the paper along.

To install the tractor, identify the two "snap levers" shown in Figure 1-8. At the same time, identify the two "stoppers," nickel-plated bars over which the hooked or cut-out bottom edge of the tractor frame fits.

OK? Now pick up the tractor unit. While depressing the two snap levers, guide it down to the two stoppers; when the hooks slide over the stopper bars, let go of the snap levers to lock it in place.

Next, if you haven't already, install the paper separator and paper guide (see Appendix A), and we're ready to start loading.

1. Turn the power switch off and remove the printer cover.
2. Pull the release lever (on left side) to position "T".
3. Raise the paper press bar; lift the paper separator upright.
4. Place the stack of fan-fold paper behind the printer.
5. Open the tractor covers, atop the right and left sprocket units, as shown in Figure 1-9.
6. Flip the clamp levers forward. This allows the two sprocket units to move freely right and left, so you can align them with the holes in the paper.
7. Pick up the top sheet, and feed it between the paper chute and platen cover plate.
8. Push the paper down and forward, so it wraps around the platen.

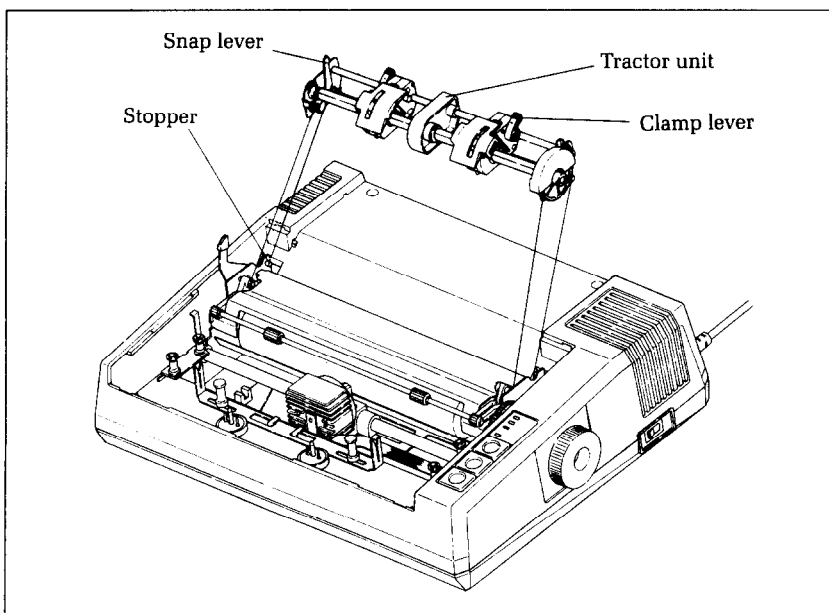


Figure 1-8. Replace the tractor unit by placing the hooks against the stoppers and lower the front into place while holding the snap levers.

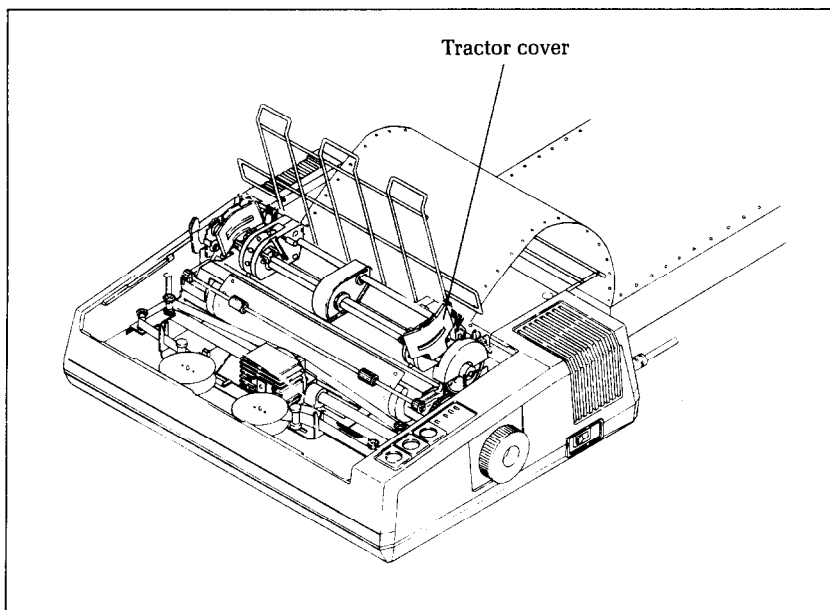


Figure 1-9. Open the tractor covers to expose the sprocket teeth.

9. Return the paper separator to its original flat position.
10. Pull the paper up, past the sprocket units.
11. When holes fit snugly over the nubby teeth in both sprockets, close the tractor covers and snap the clamp levers back into their locked position (Figure 1-10).
12. With the platen knob, roll the paper up or down until the correct "start-print" position is reached. You do this by lining up the horizontal perforation (where you tear apart individual sheets) with the top of the ribbon guide (as shown in Figure 1-11).
13. Now you're ready to roll! Replace the printer cover and turn the power switch on. Rapid printing!

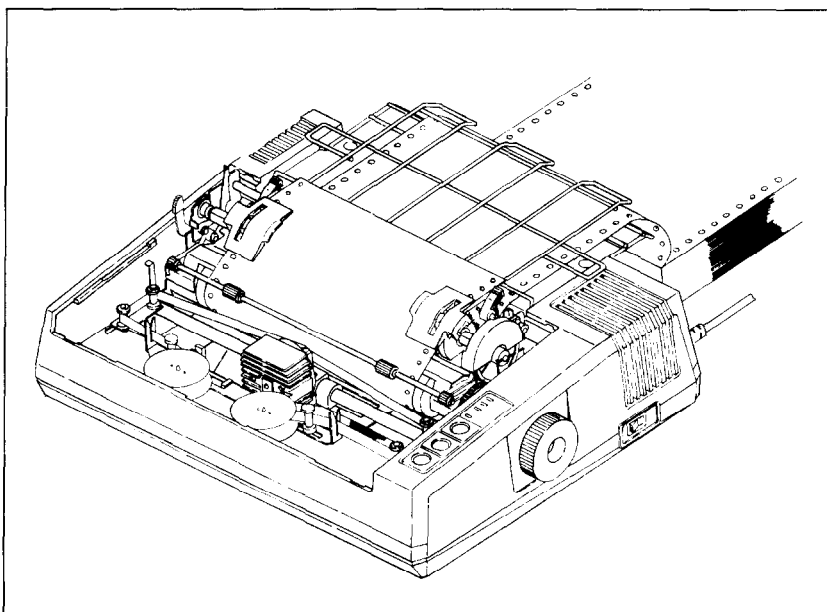


Figure 1-10. Delta ready to run with sprocket-feed paper.

Bottom feeding Delta-15

The Delta-15 can be loaded with sprocket paper in two different ways—either from the back, as with Delta-10, or through a slot in the bottom. To load Delta-15 from the back, follow the steps shown in the previous section. But for loading through the bottom slot, you position the Delta-15 above the stack of fan-fold paper, with the paper being fed up through the bottom of the printer and on out the back.

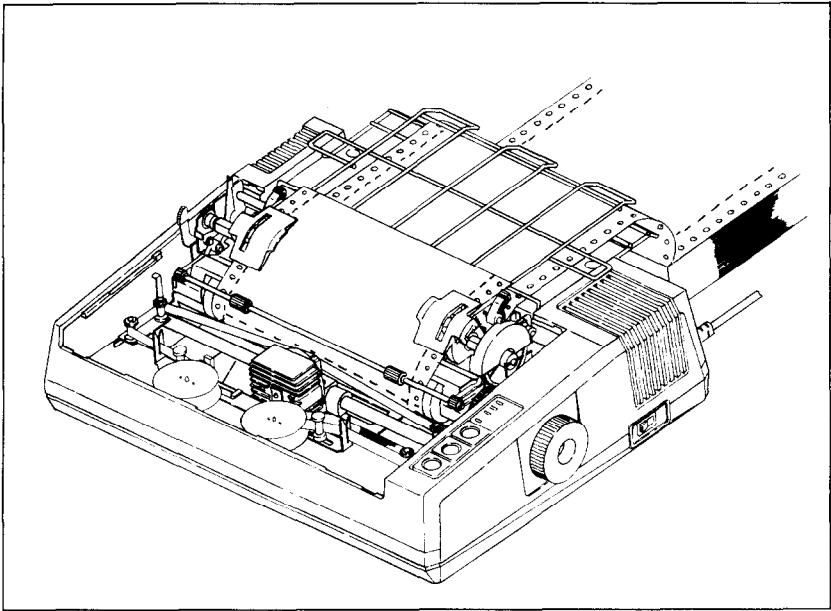


Figure 1-11. The perforation should be lined up with the top of the ribbon guide.

To use Delta-15 this way, you'll need to install the tractor unit, the paper separator, and the paper guide if you haven't already. If you're unsure how to do it, see Appendix A and Figure 1-8.

The steps for bottom loading Delta-15:

1. Turn the power switch off and remove the printer cover.
2. Pull the release lever to position "T" (Figure 1-4).
3. Raise the paper press bar.
4. Place the stack of sprocket-feed paper below the printer, ideally on a specially-built printer table with a built-in slot.
5. Open the tractor covers, right and left (Figure 1-9).
6. Flip the clamp levers forward. This allows the two sprocket units to move freely right and left, so you can align them with the holes in the paper.
7. Pick up the first "sheet" and lift it up and through the slot in the bottom of the Delta-15.
8. Push the paper up to the front of the platen roller.
9. Feed the top sheet inside the paper press bar and past the platen, high enough so you can grip the paper from above the printer.
10. Pull the paper up past the sprocket wheels.
11. When the holes fit snugly over the nubby teeth, close tractor covers and snap the clamp levers back into the locked positions.

12. With the platen knob, roll the paper up or down until the correct "start-print" position is reached. This position is achieved by lining up the horizontal perforation with the top of the ribbon guide as shown in Figure 1-11.
13. Now we're ready to roll — replace the printer cover, and turn on the power switch. Speedy printing!

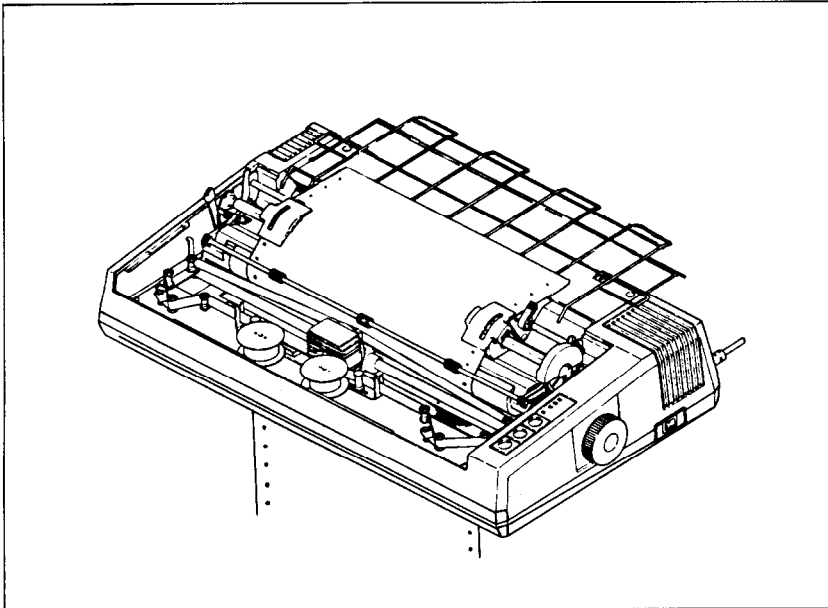


Figure 1-12. Delta-15 can be loaded from the back like Delta-10 or from the bottom, as shown here.

Ribbon Installation

Installing the ink ribbon with its two spools is described in detail in Appendix A. Just follow the diagrams.

Adjusting the Gap

What's the gap? The gap is the space between the print head and the platen. Adjusting the gap is simply adjusting the printer to take different thicknesses of paper.

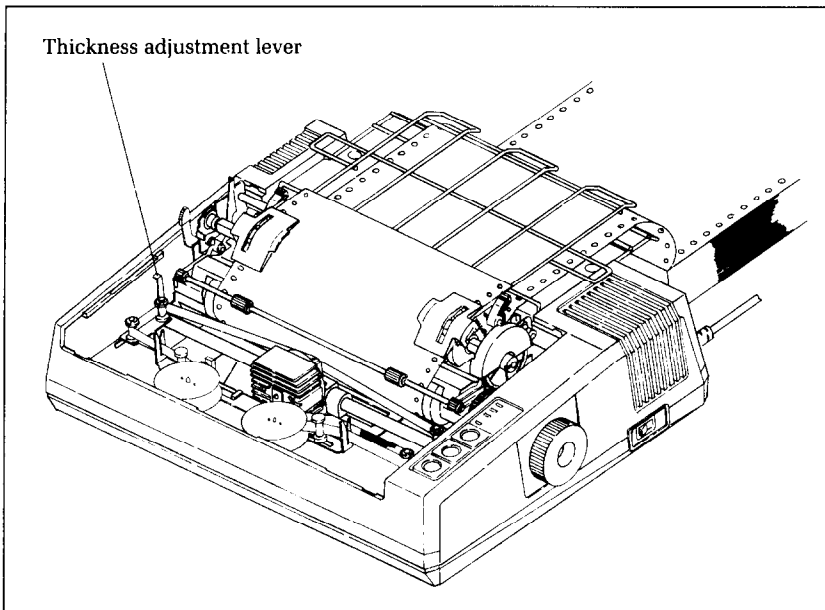


Figure 1-13. Adjusting the print head gap lever allows you to obtain optimum print quality on paper ranging from .07 mm to .28 mm thick—even 3-part carbonless sets.

To make the adjustment, move the “thickness adjustment lever” which is immediately in front of the “release lever” shown in Figure 1-13. Pulling the thickness adjustment lever towards you will widen the gap; pushing it away from you will narrow the gap.

Five positions are available; you can feel the lever clicking into the various notches. The second step (illustrated) is the one most commonly used for single sheets of paper. The lever is straight up in this position.

You shouldn't encounter any difficulty in getting the right gap setting to fit your paper. If necessary, experiment; you'll soon find the best position for the paper you're using.

Self-Test

The “self-test” is a trial run of your beautiful new machine. Delta carries a built-in program that prints out sample lines of letters, numbers, and other characters—to show you that everything's in good working order. It also serves as a display of all the characters available in the Delta. And, finally it's a “warm-up” that permits you to check your installation of ribbon and paper, and the adjustment of the print head gap.

Best of all, you don't have to wait another minute—you can print the self-test without hooking up the Delta to your computer! It's as simple as 1, 2, 3 . . .

1. Plug the printer's power cord into a 120 VAC outlet.
2. Insert a sheet of paper.
3. While holding down the L.F. button, turn the power switch on.

Surprised you, didn't it? How did you like that blinding speed — 160 characters per second! And the amazing array of type faces, symbols, and graphics! The sample print-out contains characters in the following sizes and type faces, all of them stored in the printer's permanent memory.

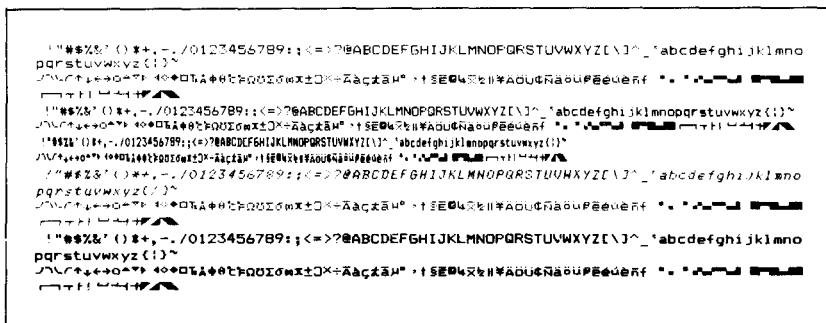
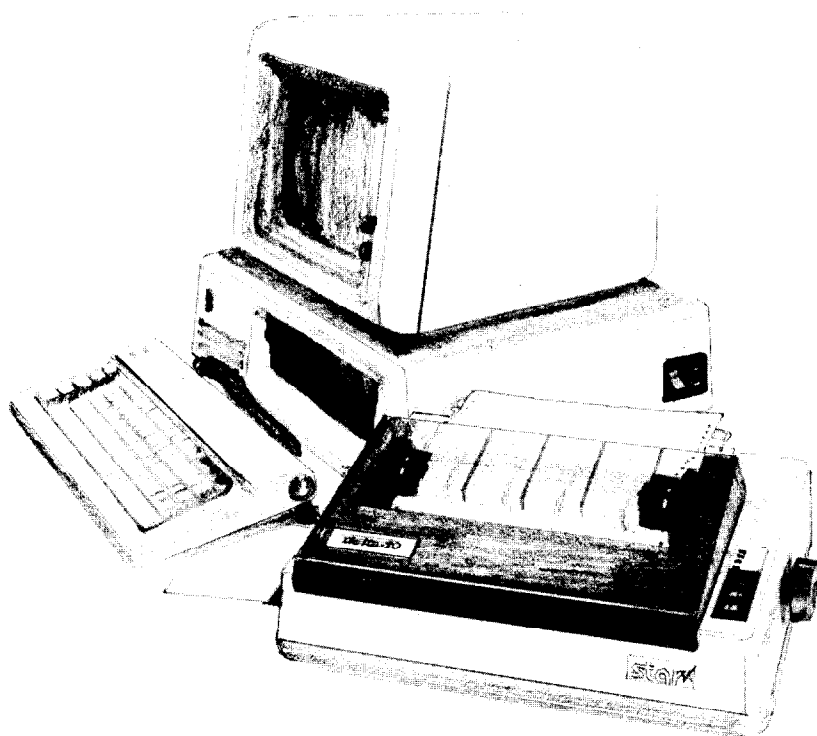


Figure 1-14. The self-test gives a hint of what's to come.

1. Standard pica type — 10 characters per inch
2. Standard elite type — 12 characters per inch
3. Condensed type — 17 characters per inch
4. Italic pica style — 10 characters per inch
5. Emphasized pica — 10 characters per inch

What next? Chapter 2 takes up the timely subject of “Communicating with Delta.” Now you'll learn how to make your computer put your printer through its many paces.



Chapter 2

Getting Started With Delta

You have assembled and tested your printer, and seen a quick sample of Delta's capabilities in the self-test. Now it's time to do what you bought Delta to do: print information from your computer.

But first you need to connect Delta to your computer. Figure 2-1 shows where the cables connect, but there's more that you need to know. Complete instructions for connecting Delta to many popular computers are given in the appendix. Find the appendix that covers your computer and follow the instructions for connecting Delta and for setting the DIP switches. If your computer isn't listed in the appendix, then ask your Star dealer which computer that is listed is most like yours. If none of the listed computers are similar to yours, then your Star dealer will give you

advice on connecting Delta to your computer.

When everything is connected, come back here and we will check it out!

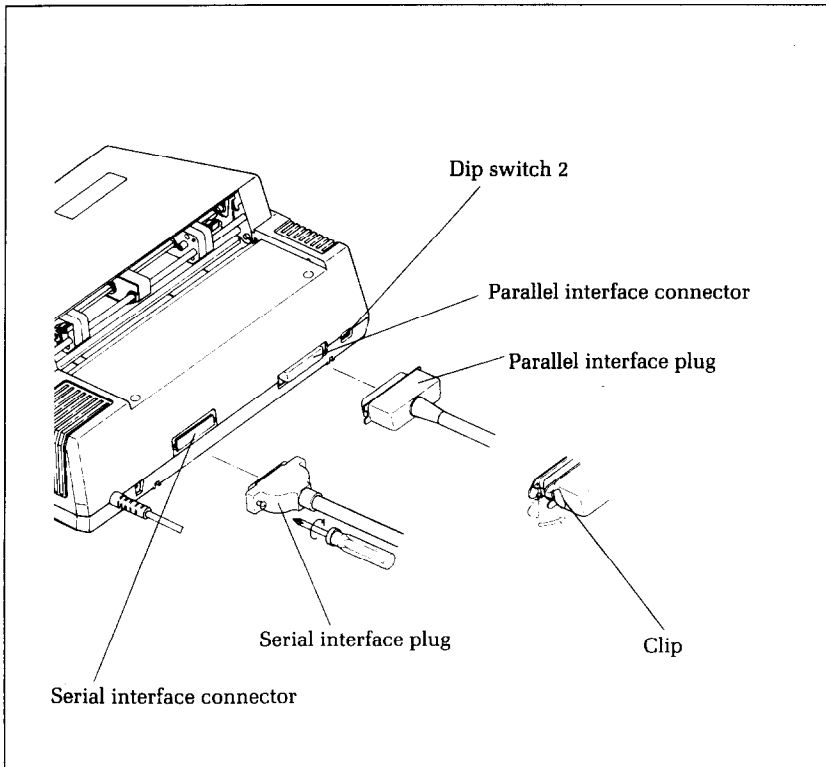


Figure 2-1. Delta has both serial and parallel interfaces.

Using Commercial Software

Many of you purchased Delta to use with commercial software. You made a good choice because Delta is compatible with most commercial programs, from word processing programs to spreadsheet programs to accounting programs.

Many of these programs have a routine for describing your printer. These routines are often in "installation programs". They typically give you a choice of printers or printer types to pick from. Some typical descriptions that you might pick for Delta are: "TTY type printer with backspace", "IBM-dot matrix printer", "Centronics-type printer", "Dot matrix ASCII printer". Delta should work fine with any of these descriptions.

Many of these lists of printers are not very clear, and may not include anything that you think describes Delta. If you can't decide which description best fits Delta, we recommend that you narrow the list to two or three choices (you can quickly eliminate all the daisy-wheel printer types) and then experiment. You won't hurt anything if you guess wrong; it just won't work right. This should quickly tell you if your guess is right. If all else fails, though, your Star dealer will be happy to give you some advice.

Some programs don't ask you what kind of printer you have, but instead they ask some questions about what your printer can do. Here are the answers to the "most asked" questions. Delta can do a "backspace". Delta can do a "hardware form feed".

With these questions answered, you are ready to start printing. Read the manual that came with your commercial software to see how to make it send information for Delta to print. This is all you need to know to use Delta as a regular printer. But Delta isn't just a regular printer. Delta has many capabilities that your commercial software isn't aware of. A little later we will see what it takes to use some of Delta's advanced features with commercial software.

First, some terminology

Delta knows what to print because it knows how to interpret the codes that the computer sends to it. These codes are numbers that the computer sends to Delta. Both the computer and Delta know the meaning of these codes because they are a set of standard codes used by almost all microcomputers. This set of codes is the *American Standard Code for Information Interchange*, which is usually referred to as ASCII (pronounced ask-key). There are ASCII codes for all the letters of the alphabet, both lower case and capital, the numbers from 0 to 9, most punctuation marks, and some (but not all) of Delta's functions.

ASCII codes are referred to in several different ways, depending on the way they are used. Some times these codes are treated as regular numbers. For example, the letter "A" is represented by the number 65 in ASCII. Appendix M shows all of the ASCII codes.

In BASIC, ASCII codes are used in the CHR\$ function. This function is used to print the character that is represented by the number in the CHR\$ function. The BASIC statement PRINT CHR\$(65) will print an "A" on the terminal.

In some other programming languages, ASCII codes are referred to by their hex value. "Hex" is short for hexadecimal which is a base-16 number system. (Our usual numbers are base-

10) Since hex needs 16 digits, it uses the numbers 0 through 9 and then it uses the letters A through F for digits. The ASCII code for the letter "A" is 41 in HEX.

Of course, most of the time we don't even need to think about this code system. Our computers are smart enough to know that when we press the "A" key on our keyboard we want to print the letter "A". The computer takes care of all the rest.

But there are a number of ASCII codes that don't have keys on the keyboard. The most important of these codes are the codes that have ASCII values below 32. These codes control many of Delta's functions. Even though there aren't keys for these codes, most keyboards can send these codes. It's done by holding down the "control" key (many times marked CTRL) and simultaneously pressing a letter key. The particular letter key that is pressed determines what code is sent. Control and A sends ASCII code 1, control and B sends ASCII code 2, and so on. Because of the way they are created, these codes are often referred to as "control-A" etc.

So there are four common ways of referring to the same set of codes: the character or name of the code, the decimal ASCII value, the hexadecimal ASCII value, and the "control-" value.

For example, the code that causes Delta to advance the paper one line is ASCII 10 (decimal). This code is commonly referred to by all the following names:

line feed	— its name
<LF>	— the abbreviation of its name
ASCII 10	— its decimal value
ASCII 0AH	— its hex value (the H signifies hex)
CHR\$(10)	— the way it's used in BASIC
control-J	— the way you send it from a keyboard.

There's a chart in Appendix M that shows these side-by-side so that you can convert back and forth.

The reason that we are telling you all this about ASCII codes is that people are not very consistent about how they describe ASCII codes. We are going to help you use Delta with commercial software, but we don't know what its documentation is going to call the various codes. So if you know all the different things that the codes might be called it will be easier to figure out what it is trying to tell you.

Now, armed with the knowledge of what to look for, you can delve into the manuals of your commercial software and dig out the secrets of how to send "control codes" to your printer. When

you find the method that your program uses, then you can shop through this manual to find the function that you want to use. By translating the codes from the system that we use, to the system that your commercial software uses, you should be able to use many of Delta's advanced features. It may help, however if we look at a couple of examples.

Using Delta with SuperCalc

SuperCalc is typical of the many spreadsheet programs that are now available. It has the capability of using several of the advanced features of Delta. Perhaps the most often used feature with spreadsheet programs is compressed printing. Let's see how to use compressed printing with SuperCalc.

In SuperCalc, the /Output command provides output to the printer. One of the options of the /Output command is S(etup). This option provides you with a menu of functions to configure SuperCalc to match your printer. You can change the number of characters that SuperCalc will print on a line and the number of lines that will print on a page. You should be sure that these values match your printer. Delta-10's print 80 characters per line of pica type, or 136 characters of condensed type. Delta-15 can print 136 characters per line of pica type, or 233 characters per line of condensed type. One of the other options on this menu is "send setup codes to printer". This is how we tell Delta that we want to use condensed print. The code to switch Delta into condensed print is ASCII 15, or control-O. So to switch on condensed type, use the /Output command and, after selecting D(isplay) and entering the range to print, select the S(etup) option, and the S(etup)—"Manual setup codes" sub option. Then, at the prompt that says "Enter codes (CR when done)", type control-O. Remember, to enter control-O you hold down the CTRL key while you press the O key (That's the letter Oh, not the number zero). Then just press return and select P(rint) to print your report.

You only need to go through this procedure once each time you use SuperCalc because Delta will stay in compressed print until it's turned off or reset.

You might also wish to use some of Delta's other features with SuperCalc. Find the code for the feature you wish to use in Appendix K and use the same procedure given here. Remember that Appendix M can be used to translate between the different names for the codes.

Using Delta with word processors

Not many word processing programs recognize the advanced

features of printers like Delta. They usually provide for some method of making bold characters and underlining. But Delta can do much more than that. The people that write word processing programs do, however, know that there are a lot of different printers on the market, and so they usually, (but not always) provide a way of sending special codes to a printer. We will study one example of this to see how a typical word processor handles it. Once you understand the concept you should be able to use your program manual to figure out how your word processor does it.

The program that we will study is the EasyWriter word processor for the IBM Personal Computer. This uses a fairly typical method of handling special codes. Generally, word processing programs don't want you to put non-printing codes in the file. They "know" that they won't print anything, and so they "protect" you by not letting you use them. But the non-printing codes are the ones that you need to use Delta's features. So EasyWriter provides a way to override this protection. If you precede a special code with a "control-O" then EasyWriter will accept the next non-printing code.

Let's look at a specific example. Suppose you want to print the title of a book in italic. The code sequence to select italic type is Escape 4 (that's two separate characters). Entering the 4 is no problem; it's a printing character so EasyWriter won't object (although in this case it's not going to print). The Escape, however, is a non-printing character so it requires special handling. To enter the Escape code first enter control-O (hold the Ctrl key while you press the letter O). Then press the Esc key. The Escape character shows on the screen as a left pointing arrow. Now just type the number 4 and you're done.

When you want to end the italic, you need to enter Escape 5. Use the same procedure: enter control-O, Esc, and then 5.

You can use many of Delta's features this way. Find the codes that you need in Appendix K, and then if necessary, use Appendix M to translate the codes into the form your word processor uses.

A note to WordStar users: WordStar is probably the most popular word processing program in the world. But it provides no way to enter special printer control codes from the keyboard. WordStar does, however, provide you with a way to use some of Delta's advanced features. WordStar has four special commands that you can use to access Delta's features. These are called "user printer controls" and are control-P Q, control-P W, control-P E, and control-P R. You might use two of these to turn italic on and off and the other two for some other function. The process of setting up these codes is called "patching" and is done with the

install program that comes with WordStar. The procedure is fairly involved, but it is explained in the WordStar manual. If you have trouble figuring it out, ask for assistance where you bought WordStar.

Using this book without learning BASIC

Throughout most of this book we will be teaching you how to use Delta's features using the BASIC programming language in our examples. This is because it is easy to communicate with Delta from BASIC and because, despite its shortcomings, BASIC is the nearest thing to a universal language among users of personal computers. But it's not the only way to communicate with Delta, as we have already seen. Even if you don't know BASIC, you can learn how to use Delta's features by reading on. When you find a function that you want to use, just apply what you already know about translating from one name for codes to another. The examples will still show you how the commands are used, even if you are not using BASIC.

Some Basics About BASIC

Probably the simplest thing to do with your printer in BASIC is to list a program on the printer. But in this world of proliferating microcomputers even this presents a problem. It seems that every computer uses a different system of communicating with the printer. We are going to tell you about some of the more common ways, and hope that between this and your computer's BASIC manual you will be able to stay with us.

First on our list is Microsoft BASIC's way of communicating with the printer. They just add an "L" to the beginning of the LIST and PRINT commands, making them LLIST and LPRINT. This method is used by more computers than any other and so we will use it throughout this book, after telling the rest of you how to follow along.

Microsoft BASIC is used by TRS-80 computers, IBM-PC computers, many CP/M computers, and many other computers. (Look in your BASIC manual; it will probably say if it's Microsoft BASIC.)

Next we need to talk about Apple II computers. They have a real simple system. To list a program that you have loaded into

memory, just type:

```
PR#1
LIST
PR#0
```

The PR#1 says “send everything to the printer”, the LIST sends it, and the PR#0 says “Ok, back to the screen now”. (There are some slightly different versions of these commands in Appendix C.)

Some other computers require you to open the printer as a numbered device, and then direct the output to that device. For example, to list a program on the printer with a Commodore C-64 computer you type the following:

```
OPEN4,4
CMD4
LIST
PRINT#4 : CLOSE4
```

This says that the printer is device 4, directs the output to it, lists the program, and finally closes device 4.

The appendix gives more information about listing programs on various computers. Find the appendix that tells how your computer works, and try it.

Now that we all know how our computers address the printer, let's try listing a BASIC program. Load a BASIC program and LLIST it (or however your computer does it).

We've crossed the first major hurdle—learning how to list programs on Delta. Now we are ready to jump into the world of programming with Delta. But first, there are a few fundamentals that we need to cover.

Establishing communications

We've learned something about communicating with our printer. Now we need to adapt what we know to printing in a BASIC program. Generally, computers use about the same procedure for printing in a program as they do to list a program. Again take a few moments to look at the appendix that relates to your computer. We'll continue when you have it all figured out.

Welcome back. Let's try what we learned. Type the following:

```
NEW
1Ø LPRINT "TESTING"
RUN
```

Remember—we use LPRINT; you may have to use something else!

At any rate, you should have the word “TESTING” on your printer. Quite an achievement, isn't it? Let's get done with this simple stuff so that we can go on to something interesting.

The CHR\$ function

We mentioned CHR\$ earlier as one way to express ASCII codes. We are going to use it a lot in communicating with Delta. Delta uses many of the ASCII codes that don't represent letters and numbers. The CHR\$ function gives us an easy way to send these codes to the printer. Try this to see how the CHR\$ function works:

```
NEW
1Ø LPRINT CHR$(68)
RUN
```

That should print a “D” for Delta. If you check the chart in Appendix I you will see that 68 is the ASCII code for “D”.

Control codes

Delta uses many of the non-printing ASCII codes for control codes. These codes perform a function rather than printing a character. Let's try an easy one right now:

```
NEW
1Ø LPRINT CHR$(7)
RUN
```

Who made that noise? That's Delta's bell. We will learn more

about it in Chapter 6. We just wanted to illustrate a code that causes Delta to perform a function.

The escape code

There's one particular ASCII code that we are going to be using more than all the rest. This is ASCII 27, which is called *escape*. In BASIC it's CHR\$(27). With all of Delta's advanced features, there weren't enough single ASCII codes to go around. So *escape* is used to start sequences of control codes that open a wider range of functions to us.

While you must call this code CHR\$(27) in BASIC, we are going to refer to it as <ESC> in this book. This will make it much easier to recognize when we use it.

A typical *escape* code sequence starts with <ESC> which is followed by one or more CHR\$ codes. As an example, the *escape* code sequence to turn on italic print is:

```
<ESC> CHR$(52)
```

In a program, this would look like this:

```
NEW  
10 LPRINT CHR$(27) CHR$(52);  
20 LPRINT "TESTING"  
RUN
```

Try this program, it will print the word *TESTING* in italic.

Some of you fast students may have noticed that CHR\$(52) is the same as "4". That's right, the program will work just as well if line 10 is changed like this:

```
10 LPRINT CHR$(27) "4";
```

That's just another form of the same ASCII code, and it's all the same to Delta.

Here's another shortcut for BASIC programmers: since <ESC> is used so often, assign it to a variable. In a long program, typing ESC\$ is much easier than typing CHR\$(27) each time! Now

our program looks like this:

```
5 ESC$=CHR$(27)
10 LPRINT ESC$ "4";
```

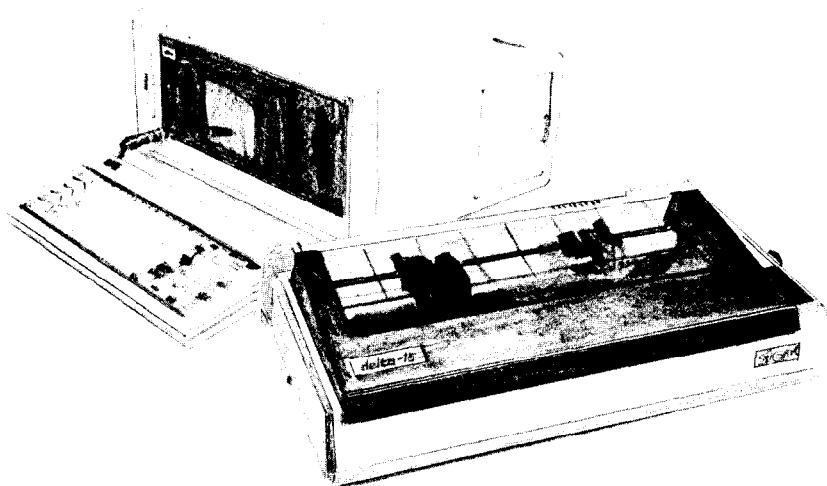
Turn your printer off and back on now, or you will be printing in italic for quite a while!

Some problem codes

Before we go too far we need to mention some codes that may cause you problems. Like most of the subjects in this chapter, we have to be a little vague because of the differences in computers.

Nearly all BASICs change some of the ASCII codes between your BASIC program and your printer. Some turn CHR\$(10) (a line feed) into a CHR\$(13) (a carriage return) before sending it on. Some other problem codes are 0, and 9 through 13. Once again we refer you to the appendix about your computer, where some more specific information awaits.

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



Chapter 3

Printing Text With Delta

Beginning with this chapter we will be exploring all the features of Delta. All our examples will be given in Microsoft BASIC as used by the IBM Personal Computer, but remember that you don't need to know BASIC to use Delta's features. Just use the same ASCII codes as we do in our examples.

If your computer doesn't use Microsoft BASIC, look in the appendix to see what changes you need to make for your BASIC. The appendix tells you how to change the short example programs, and gives complete listings of the longer programs, already converted for your computer.

You have already printed a few lines on your Delta printer. Now it's time to start looking at the many variations of printing style that you have available to you. The first technique that we

will investigate is changing the width of the characters that Delta prints.

Changing the print pitch

In "printer talk," character width is called *pitch*. Normally, Delta prints 10 characters per inch. This is called *pica* pitch because it's the same spacing as a standard pica typewriter.

Delta can also print 12 characters per inch. This is called *elite* pitch because it is the same spacing as an elite typewriter.

Condensed print is approximately 17 characters per inch (actually it's 17.14 characters per inch). Condensed pitch allows you to get 136 columns of printing on an 8½ inch page.

You tell Delta which pitch you want to use with the <ESC> "B" command. The table below shows the three options of this command.

Table 3-1
Print pitch commands

Pitch	Characters/inch	Control code
Pica	10	<ESC> "B" CHR\$(1)
Elite	12	<ESC> "B" CHR\$(2)
Condensed	17	<ESC> "B" CHR\$(3)

Let's see how these three pitches look. Try this program:

```
NEW
10 LPRINT CHR$(27) "B" CHR$(2)
20 LPRINT "THIS IS ELITE PITCH PRINTING"
30 LPRINT CHR$(27) "B" CHR$(3)
40 LPRINT "CONDENSED IS THE NARROWEST PITCH"
50 LPRINT CHR$(27) "B" CHR$(1)
60 LPRINT "NOW WE ARE BACK TO PICA PITCH PRINTING"
```

When you run this program you should get this:

THIS IS ELITE PITCH PRINTING

CONDENSED IS THE NARROWEST PITCH

NOW WE ARE BACK TO PICA PITCH PRINTING

Line 10 turns on elite pitch with `<ESC> "B" CHR$(2)`. Line 20 prints a line at 12 characters per inch. The `<ESC> "B" CHR$(3)` in line 30 changes Delta to condensed pitch and line 40 prints a line in condensed pitch. Line 50 resets Delta to pica pitch and line 60 prints a line in pica pitch.

Pica pitch and condensed pitch can be set with "shortcut" codes. Instead of using `<ESC> "B" CHR$(n)`, you can set them with a single code. `CHR$(18)` sets pica pitch and `CHR$(15)` sets condensed pitch. You can not set elite pitch with a single code.

Expanded print

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

```
NEW
10 LPRINT CHR$(14) "THIS LINE IS EXPANDED"
20 LPRINT "BUT THIS LINE IS NOT"
```

```
THIS LINE IS EXPANDED
BUT THIS LINE IS NOT
```

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

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

```
10 LPRINT CHR$(27) "W" CHR$(1) "THIS LINE IS
EXPANDED"
20 LPRINT "AND SO IS THIS ONE"
30 LPRINT CHR$(27) "W" CHR$(0) "NOW WE'RE BACK TO
NORMAL"
```

Now the results look like this:

```
THIS LINE IS EXPANDED
AND SO IS THIS ONE
NOW WE'RE BACK TO NORMAL
```

When you turn on expanded print with $\langle \text{ESC} \rangle$ "W" CHR\$(1) it stays on until you turn it off with $\langle \text{ESC} \rangle$ "W" CHR\$(0). That's what we added line 30 for.

Table 3-2
Expanded print commands

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

By combining expanded print with the three pitches, Delta has six different character widths available.

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

```

10 LPRINT CHR$(14) "EXPANDED PICA PITCH"
20 LPRINT CHR$(27) "B" CHR$(2)
30 LPRINT CHR$(14) "EXPANDED ELITE PITCH"
40 LPRINT CHR$(27) "B" CHR$(3)
50 LPRINT CHR$(14) "EXPANDED CONDENSED PITCH"
60 LPRINT CHR$(27) "B" CHR$(1)
70 LPRINT "NOW WE ARE BACK TO UNEXPANDED PICA
PRINTING"

```

Here's what you should get from this program:

EXPANDED PICA PITCH

EXPANDED ELITE PITCH

EXPANDED CONDENSED PITCH

NOW WE ARE BACK TO UNEXPANDED PICA PRINTING

Making Delta print darker

Delta has very good print density when it's just printing regularly. But sometimes you may want something to stand out from the rest of the page. Delta provides two ways to do this: double-strike 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 double-strike and emphasized modes.

Table 3-3
Print emphasis commands

Function	Control code
Double-strike ON	<ESC> "G"
Double-strike OFF	<ESC> "H"
Emphasized ON	<ESC> "E"
Emphasized OFF	<ESC> "F"

Try them now with this little program:

```
NEW
10 LPRINT CHR$(27) "G"
20 LPRINT "THIS IS DOUBLE-STRIKE PRINTING"
30 LPRINT CHR$(27) "H" CHR$(27) "E";
40 LPRINT "THIS IS EMPHASIZED PRINTING"
50 LPRINT CHR$(27) "G";
60 LPRINT "AND THIS IS BOTH AT ONCE"
70 LPRINT CHR$(27) "H" CHR$(27) "F"
```

Run this program. The results will look like this:

```
THIS IS DOUBLE-STRIKE PRINTING
THIS IS EMPHASIZED PRINTING
AND THIS IS BOTH AT ONCE
```

Line 10 turns on double-strike with <ESC> "G" and line 20 prints a line of text. In line 30 double-strike is turned off with <ESC> "H" and then emphasized is turned on with <ESC> "E". Line 40 prints a line of emphasized text. Line 50 then turns double-strike back on so that line 60 can print in both at once. Finally, line 70 turns both off, so that Delta is set for normal printing.

Look closely at the different lines of printing. In the line of double-strike printing each character has been printed twice, and they are moved down just slightly the second time they are

printed. In emphasized printing, they are moved slightly to the right the second time Delta prints. The last line combined both of these so that each character was printed 4 times. Now that's pretty nice printing, isn't it?

Some Special Kinds of Text

We're just getting started on the kinds of text that Delta can print. Still to come are italic characters, underlined characters, superscripts and subscripts.

Italic printing

Italic letters are letters that are slanted to the right. Delta can print all the kinds of letters that we have seen so far in *italic* as well as the roman (standard) letters we have been using. Italics can be used to give extra emphasis to certain words. The command codes to turn italic on and off are shown in Table 3-4.

Table 3-4
Italic commands

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

Use this program to see italic characters:

```
NEW
1Ø LPRINT CHR$(27) "4";
2Ø LPRINT "THIS LINE IS PRINTED IN ITALIC"
3Ø LPRINT CHR$(27) "5";
4Ø LPRINT "THIS LINE IS NORMAL PRINTING"
```

Here is what you should get:

```
THIS LINE IS PRINTED IN ITALIC
THIS LINE IS NORMAL PRINTING
```

This program is easy; line 10 turns italic on with $\langle \text{ESC} \rangle$ "4", and line 30 turns it off with $\langle \text{ESC} \rangle$ "5".

Underlining

Not only can Delta print all the styles of printing that we have seen in both roman and italic, but it can underline them too. The control codes are shown in Table 3-5.

Table 3-5
Underline commands

Function	Control code
Underline ON	$\langle \text{ESC} \rangle$ "-" CHR\$(1)
Underline OFF	$\langle \text{ESC} \rangle$ "-" CHR\$(0)

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

```
NEW
10 LPRINT CHR$(27) "-" CHR$(1);
20 LPRINT "THIS IS UNDERLINED";
30 LPRINT CHR$(27) "-" CHR$(0);
40 LPRINT " AND THIS IS NOT"
```

It should come out like this:

```
THIS IS UNDERLINED AND THIS IS NOT
```

In this program underline is turned on in line 10 with $\langle \text{ESC} \rangle$ "-" CHR\$(1), and then off in line 30 with $\langle \text{ESC} \rangle$ "-" CHR\$(0). There's a new little wrinkle in this program, though. It all printed on one line. 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

We have seen how Delta can print in 6 different widths. Delta

can also 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 even with the tops of regular printing while *subscripts* print even with the bottom of regular printing. They are frequently used to reference footnotes, and in mathematical formulas.

Table 3-6 has the codes for using superscripts and subscripts.

Table 3-6
Superscript and subscript commands

Function	Control code
Superscript ON	<ESC> "S" CHR\$(0)
Subscript ON	<ESC> "S" CHR\$(1)
Super & subscript OFF	<ESC> "T"

Try this program to see them work:

```

NEW
10 LPRINT "THIS LINE USES";
20 LPRINT CHR$(27) "S" CHR$(0);
30 LPRINT " SUPERSCRIPTS";
40 LPRINT CHR$(27) "T";
50 LPRINT " AND";
60 LPRINT CHR$(27) "S" CHR$(1);
70 LPRINT " SUBSCRIPTS";
80 LPRINT CHR$(27) "T";
90 LPRINT " BOTH"

```

THIS LINE USES ~~SUPERSCRIPTS~~ AND ~~SUBSCRIPTS~~ BOTH

Here line 20 turns on superscripts with <ESC> "S" CHR\$(0). It's turned off in line 40 with <ESC> "T". Then, between printing text, subscripts are turned on in line 60 with <ESC> "S" CHR\$(1), and finally off in line 80. Again, everything prints on one line because of the semicolons.

Mixing modes

We have learned how to use Delta's many different printing modes individually. Now let's see how we can combine these


```

1140 'SPECIAL PRINT MODES'
1150 EMPHASIZED$ = CHR$(27) + CHR$(69)
1160 NOT.EMPHASIZED$ = CHR$(27) + CHR$(70)
1170 DOUBLE.STRIKE$ = CHR$(27) + CHR$(71)
1180 NOT.DOUBLE.STRIKE$ = CHR$(27) + CHR$(72)
1190 UNDERLINED$ = CHR$(27) + CHR$(45) +
CHR$(1)
1200 NOT.UNDERLINED$ = CHR$(27) + CHR$(45) +
CHR$(0)
1210 SUPERSCRIP$ = CHR$(27) + CHR$(83) +
CHR$(0)
1220 SUBSCRIPT$ = CHR$(27) + CHR$(83) +
CHR$(1)
1230 NOT.SCRIPTED$ = CHR$(27) + CHR$(84)
1240 RESET.ALL$ = NOT.EMPHASIZED$ + NOT.UNDERLINED$
+ NOT.DOUBLE.STRIKE$
1250 RESET.ALL$ = RESET.ALL$ + ROMAN$ + PICA$ +
NOT.ENLARGED$
1260 'CONSTANTS'
1270 TRUE = 1 : FALSE = 0
1280 REGULAR.HEADING$ = STRING$(27,"*") + "REGULAR"
+ STRING$(27,"*")
1290 RETURN
2000 '
2010 '
2020 ' PRINT HEADING '
2030 '
2040 '
2050 LPRINT RESET.ALL$
2060 LPRINT ENLARGED$ " NORMAL ENLARGED "
2070 LPRINT RESET.ALL$;
2080 LPRINT UNDERLINED$;
2090 LPRINT CONDENSED$ "CONDENSED ";
2100 LPRINT ELITE$ " ELITE ";
2110 LPRINT PICA$ " PICA ";
2120 LPRINT CONDENSED$ "CONDENSED ";
2130 LPRINT ELITE$ " ELITE ";
2140 LPRINT PICA$ " PICA "
2150 LPRINT RESET.ALL$
2160 RETURN
3000 '
3010 '
3020 ' PRINT FOUR LINES '
3030 '
3040 '

```


Here is the chart it produces:

NORMAL			ENLARGED		
CONDENSED	ELITE	PICA	CONDENSED	ELITE	PICA
REGULAR					
ABcd**yy	ABcd**yy	ABcd**yy	ABcd	ABcd	ABcd
ABcd**yy	ABcd**yy	ABcd**yy	ABcd	ABcd	ABcd
ABcd**yy	ABcd**yy	ABcd**yy	ABcd	ABcd	ABcd
ABcd**yy	ABcd**yy	ABcd**yy	ABcd	ABcd	ABcd
DOUBLE STRIKE					
ABcd**yy	ABcd**yy	ABcd**yy	ABcd	ABcd	ABcd
ABcd**yy	ABcd**yy	ABcd**yy	ABcd	ABcd	ABcd
ABcd**yy	ABcd**yy	ABcd**yy	ABcd	ABcd	ABcd
ABcd**yy	ABcd**yy	ABcd**yy	ABcd	ABcd	ABcd
EMPHASIZED					
....	ABcd	ABcd
....	ABcd	ABcd
....	ABcd	ABcd
....	ABcd	ABcd
DOUBLE STRIKE & EMPHASIZED					
....	ABcd	ABcd
....	ABcd	ABcd
....	ABcd	ABcd
....	ABcd	ABcd

Summary

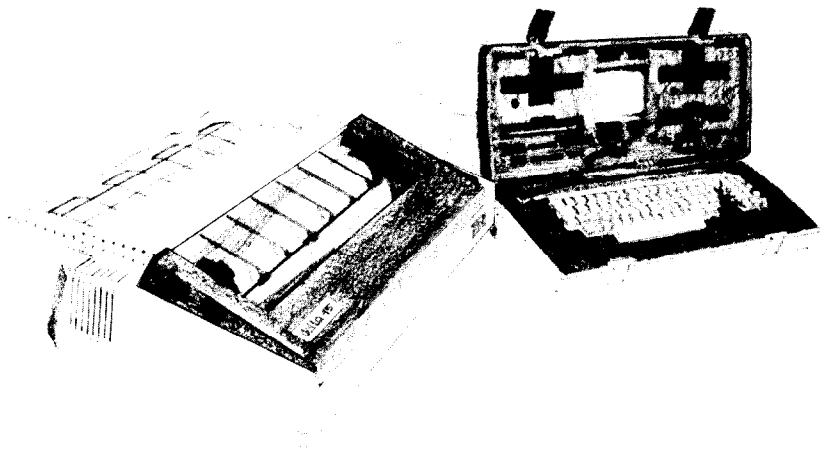
Control code

<ESC> "B" CHR\$(1)
 <ESC> "B" CHR\$(2)
 <ESC> "B" CHR\$(3)
 CHR\$(18)

Function

Sets pica pitch
 Sets elite pitch
 Sets condensed pitch
 Sets pica pitch

CHR\$(15)	Sets condensed pitch
CHR\$(14)	One line expanded
<ESC> CHR\$(14)	One line expanded
<ESC> "W" CHR\$(1)	Expanded on
<ESC> "W" CHR\$(0)	Expanded off
<ESC> "G"	Double-strike on
<ESC> "H"	Double-strike off
<ESC> "E"	Emphasized on
<ESC> "F"	Emphasized off
<ESC> "4"	Italic on
<ESC> "5"	Italic off
<ESC> "-" CHR\$(1)	Underline on
<ESC> "-" CHR\$(0)	Underline off
<ESC> "S" CHR\$(0)	Superscript on
<ESC> "S" CHR\$(1)	Subscript on
<ESC> "T"	Super & subscript off



Chapter 4

Line Spacing and Forms Control

We have learned how to print in many different ways, but so far we haven't looked at how to position the printing on the page. In this chapter we will learn how to change the vertical spacing and the length of the page.

Starting New Lines

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 is telling 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)). The codes are simple, but their action is a little confusing (especially with BASIC). Carriage return is the easiest. Each time that the printer receives a CHR\$(13) it returns the print head to the left margin. It does not advance the paper (if DIP switch 2-4 is off; 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)) that they send. If your version of BASIC doesn't do this, then you should turn DIP switch 2-4 on so that Delta will add the line feed for you. When you have DIP switch 2-4 on 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 2-4 off.

Changing Line Spacing

When you turn Delta on the line spacing is set to 6 lines per inch (or 8 lines per inch if DIP switch 1-5 is off). This is fine for most printing applications, but sometimes you may want something different. Delta makes it easy to set the line spacing to whatever value you want.

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

```
NEW
10 FOR I = 1 TO 25
20 IF I = 13 THEN 50
30 LPRINT CHR$(27) "A" CHR$(I);
40 LPRINT "DELTA HAS VARIABLE LINE SPACING"
50 NEXT
60 LPRINT CHR$(27) "2"
```


mand does not change the setting of the line spacing, but it does cause the printer to make one line feed of $n/144$ inch. Try this program to see how it works:

```
NEW
10 LPRINT "LINE NUMBER 1"
20 LPRINT "LINE NUMBER 2";
30 LPRINT CHR$(27) "J" CHR$(100);
40 LPRINT "LINE NUMBER 3"
50 LPRINT "LINE NUMBER 4"
```

Here is what Delta 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 30 changes the line spacing to $100/144$ for one line only. The rest of the lines are printed with the normal line spacing. Notice that both line 20 and line 30 end with semicolons. This prevents the normal line feed from occurring.

The value of n in all three commands ($\langle \text{ESC} \rangle$ "A", $\langle \text{ESC} \rangle$ "3", and $\langle \text{ESC} \rangle$ "J") can range from 0 to 255. A value of 0 means that there is no line spacing. This allows you to print multiple lines in the same position on the page. This is useful when you want to overprint graphics and text.

Moving down the page without a carriage return

So far, all the commands that move the paper also move the print head to the left margin. And 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 $\langle \text{ESC} \rangle$ "a" $\text{CHR}\$(n)$ command does just that. This command advances the

paper n lines (using whatever the current line spacing is) without moving the printhead. Change line 30 of your program so that it is like this:

```
30 LPRINT CHR$(27) "a" CHR$(3);
```

Now when you run the program the results will look like this:

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

The new line 30 moves the paper up 3 lines, but the printhead doesn't move. Therefore, line 40 prints its message starting in the column that the printhead was left in at the end of line 20.

Forms Controls

We have seen how to control the spacing between lines on a page. Delta also has commands that control the placement of printing on the page, and even adjust for different size pages.

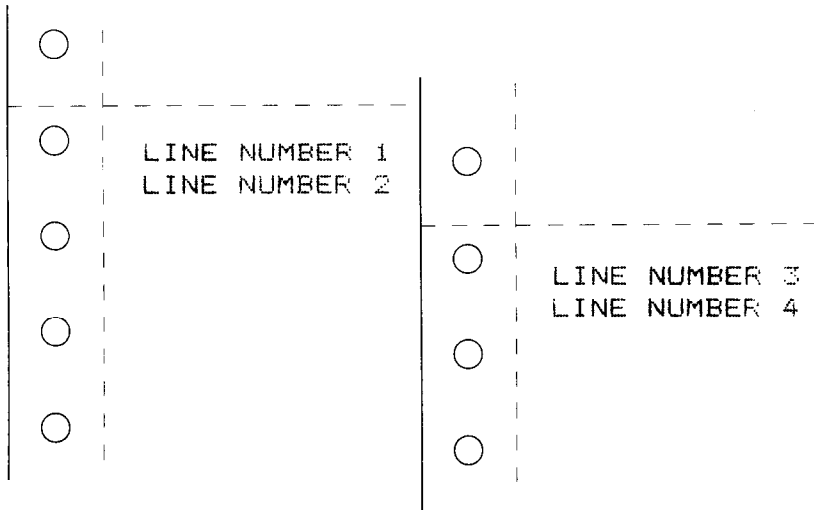
Form feed

The simplest forms control code is the *form feed*. Form feed is `CHR$(12)` and causes the printer to move the paper to the top of the next sheet. Try it by changing line 30 to this:

```
30 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 1. When you

run the program, the results will look like this:



The form feed (CHR\$(12)) in line 30 caused the printer to move to the top of a new page before printing the last two lines.

A note to TRS-80 users: CHR\$(12) is a problem code for the TRS-80. To send a form feed command to Delta you must add 128 to it making it CHR\$(140). Use CHR\$(140) where we use CHR\$(12) in these programs.

Changing the Page Length

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

Table 4-2
Form length commands

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

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

```
NEW
10 LPRINT CHR$(27) "C" CHR$(0) CHR$(7);
20 LPRINT "PAY TO THE ORDER OF:"
30 LPRINT CHR$(12);
40 LPRINT "PAY TO THE ORDER OF:"
```

This program should print "PAY TO THE ORDER OF:" twice, and they should be 7 inches apart. Line 10 sets the form length to 7 inches. After line 20 prints, line 30 sends a form feed to advance the paper to the top of the next form. Line 40 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 be reset to its normal setting (usually 11 inches).

Top and Bottom Margins

Many programs that use a printer don't keep track of where they are printing on the page. This causes a problem 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. And if you separate the pages then you are really in trouble.

Of course Delta has a solution to this predicament. Delta 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 and the other controls the space at the bottom of the page. The control codes are given in the following table.

Table 4-3
Top and bottom margin commands

Function	Control code
Set top margin	<ESC> "R" CHR\$(n)
Set bottom margin	<ESC> "N" CHR\$(n)
Clear top and bottom margins	<ESC> "O"

In both cases the value of *n* tells Delta how many lines to skip, although there is a slight difference in the usage. When you set the top margin with `<ESC> "R" CHR$(n)`, the value of *n* tells Delta what line to start printing on. When you set the bottom margin with `<ESC> "N" CHR$(n)`, the value of *n* tells Delta 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.

```
NEW
30 FOR I = 1 TO 150
40 LPRINT "THIS IS LINE "; I
50 NEXT
70 LPRINT CHR$(12);
```

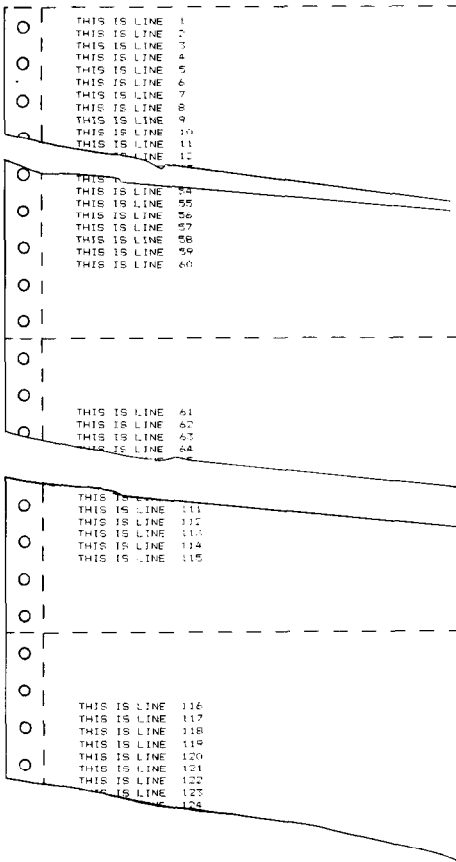
When you run this program it will print 150 lines right down the page and across the perforations. When it's done line 70 sends a form feed to advance the paper to the top of the next page. Look at the lines that have printed near the perforations. Separate the sheets and see if any of the lines have been torn in half. These are the problems that the top and bottom margins will solve.

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.)

```
10 LPRINT CHR$(27) "N" CHR$(6);
20 LPRINT CHR$(27) "R" CHR$(6);
60 LPRINT CHR$(27) "O";
```

Now when you run the program Delta will skip the first six lines and the last six lines on each page (except for the first page, where Delta started printing at the top). That's because the top margin only works after a form feed, and we didn't send Delta a form feed after we set the top margin.

Line 10 sets the top margin, line 20 sets the bottom margin, and line 60 clears both margins when we are done.



Summary

Control code

CHR\$(10)

CHR\$(13)

<ESC> "A" CHR\$(n)

<ESC> "3" CHR\$(n)

<ESC> "0"

<ESC> "1"

<ESC> "2"

<ESC> "j" CHR\$(n)

<ESC> "a" CHR\$(n)

Function

Line feed

Carriage return

Set line spacing to $n/72$ inch

Set line spacing to $n/144$ inch

Set line spacing to $1/8$ inch

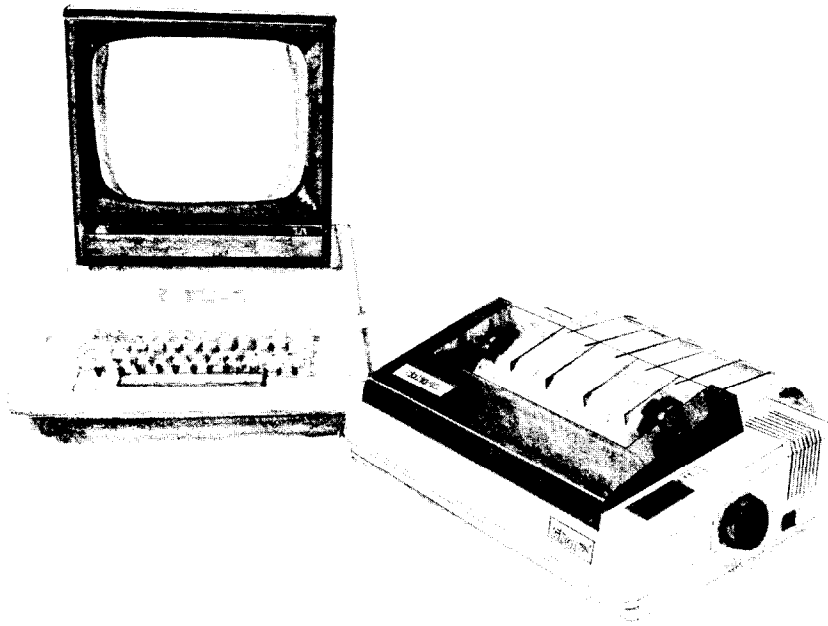
Set line spacing to $7/72$ inch

Set line spacing to $1/6$ inch

One-time line feed of $n/144$ inch

Advance the paper n lines

CHR\$(12)	Form feed
<ESC> "C" CHR\$(n)	Set page length to n lines
<ESC> "C" CHR\$(0) CHR\$(n)	Set page length to n inches
<ESC> "R" CHR\$(n)	Set top margin; start printing on line n
<ESC> "N" CHR\$(n)	Set bottom margin; leave n lines blank
<ESC> "O"	Clear top and bottom margins



Chapter 5

Formatting Your Output

You have probably used the tab and margin features on a typewriter. They make it easier to format the text on a page. Delta also has tabs and margins that you can set. But it goes beyond the capabilities of a typewriter because besides having tabs that go across the page, called *horizontal tabs*, Delta has *vertical tabs* that go down the page. In this chapter we will discover how to use the tabs and margins on Delta.

When you turn Delta on there are horizontal tabs set automatically every ten spaces. If you start counting at column 1 they are at columns 10, 20, 30, 40, etc. It's easy to use these tabs; you just send a CHR\$(9) to Delta and the print head will move to the next tab position. CHR\$(9) is the ASCII code <HT> for *horizontal tab*.

Try this one line program to demonstrate the use of the default horizontal tabs.

```
NEW
2Ø LPRINT "ONE" CHR$(9) "TWO" CHR$(9) "THREE"
      CHR$(9) "FOUR"
```

Here's what will print:

```
      ONE          TWO          THREE          FOUR
```

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

CHR\$(9) is a problem with some computers. Some BASICs convert CHR\$(9) to a group of spaces that act like a sort of pseudo-tab. This is fine if the computer and the printer have the same tab settings, but it doesn't allow us to use our own tab settings on Delta. We can "outsmart" these computers by adding 128 to the ASCII value that we use. Instead of using CHR\$(9), use CHR\$(137) for a tab command. Even this trick won't work for Apple II computers, for they use CHR\$(9) for something else entirely. Apple users can get some help in Appendix C.

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

```
1Ø LPRINT CHR$(27) "D" CHR$(8) CHR$(16) CHR$(24)
      CHR$(Ø)
```

<ESC> "D" is the command to begin setting horizontal tabs. It must be followed by characters representing the positions that you want the tabs set. In our program we are setting tabs in columns 8, 16, and 24. The CHR\$(0) at the end 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. (It also means that a CHR\$(1) is just as good as a CHR\$(0) for ending a group of tabs; some computers have trouble sending CHR\$(0).)

When you run the program now it produces this:

ONE TWO THREE FOUR

The words are now closer together, but still evenly spaced. Turn your printer off and on again to reset the default tabs.

If you set tabs in one pitch, such as pica, and then change the pitch, say to elite, the tab settings will also change. If, for example, the tabs are set every eight spaces, when you change pitch they will still be set every eight spaces, but the spaces will be a different width.

A one-shot tab command

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 must be an easier way—and of course there is.

The solution is called a *one-time tab* and is <ESC> "b" CHR\$(n). This command moves the print head n columns to the right. It has the same effect as sending n spaces to the printer.

Setting Left and Right Margins

Delta's left and right margins work just like a typewriter—once they are set all the printing is done between them. The commands to set the margins are given in the following table:

Table 5-1
Left and right margin commands

Function	Control code
Set left margin at column n	<ESC> "M" CHR\$(n)
Set right margin at column n	<ESC> "Q" CHR\$(n)

Try setting Delta's margins with this program:

```
NEW
10 GOSUB 100
20 LPRINT CHR$(27) "M" CHR$(10);
```

```

30 LPRINT CHR$(27) "Q" CHR$(70)
40 GOSUB 100
50 END
100 FOR I = 1 TO 80
110 LPRINT "X";
120 NEXT I
130 LPRINT
140 RETURN

```

The first thing that this program does is to branch to the subroutine that starts in line 100. This subroutine prints 80 X's in a row. The first time that the subroutine is used, all the X's fit in one line. Then line 20 sets the left margin to 10, and line 30 sets the right margin to 70. Once again the subroutine is used, but this time the X's won't all fit on one line since there is now only room for 61 characters between the margins. (There's room for 61 (instead of 60) characters because you can print in both the first and last column that you name.)

Run the program. The results will look like this:

```

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

```

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 1 and a right margin of 80 on Delta-10 or 136 on Delta-15.

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 then you change to elite type you will have room for more than 65 columns of elite printing between the margins.

Using 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 <VT> (or vertical tab) will move you to line 20 whether you start from line 5 or line 19.

The default vertical tab settings are every six lines. If you send a CHR\$(11), which is the ASCII code for <VT>, before we have set up tabs it will advance the paper to one of these preset tabs. Enter this program to see how this works.

```
NEW
20 LPRINT CHR$(11) "FIRST TAB"
30 LPRINT CHR$(11) "SECOND TAB"
40 LPRINT CHR$(11) "THIRD TAB"
50 LPRINT CHR$(11) "FOURTH TAB"
```

The CHR\$(11) in each line advances the paper to the next vertical tab. The lines should be spaced evenly, six lines apart.

Now let's set some vertical tabs of our own. Add this line to the program:

```
10 LPRINT CHR$(27) "P" CHR$(10) CHR$(20) CHR$(40)
   CHR$(50) CHR$(0);
```

<ESC> "P" 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, 20, 40 and 50. Then the CHR\$(11) in each of the following lines advances the paper to the next vertical tab. Figure 5-1 is what you get.

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

```
60 LPRINT CHR$(11) "FIFTH TAB"
```

Now when you run the program the first page looks just like before, but line 60 sends one more <VT> than there are tabs. This doesn't confuse Delta—it advances the paper to the next tab position which happens to be the first tab position on the next page. That's nice, isn't it?

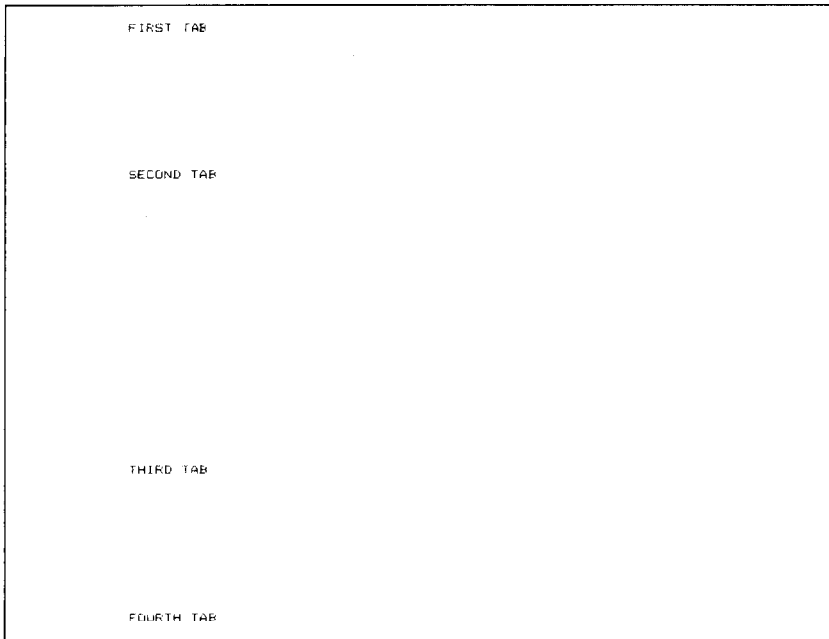


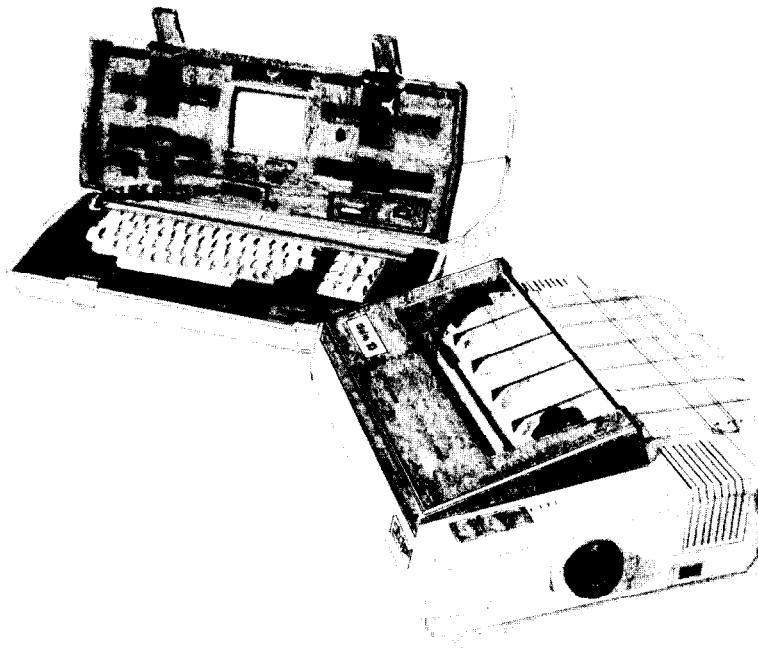
Figure 5-1.

A one-shot vertical tab command

There's a one-time vertical tab command that works just like the one-time horizontal tab command. It is $\langle \text{ESC} \rangle$ "a" CHR\$(n), and it causes the paper to advance n lines. It doesn't change the settings of the vertical tabs.

Summary

Control code	Function
CHR\$(9)	Horizontal tab
$\langle \text{ESC} \rangle$ "D" n1 n2 n3 . . . CHR\$(0)	Set horizontal tabs
$\langle \text{ESC} \rangle$ "b" n	One-time horizontal tab of n spaces
$\langle \text{ESC} \rangle$ "M" n	Set left margin
$\langle \text{ESC} \rangle$ "N" n	Set right margin
CHR\$(11)	Vertical tab
$\langle \text{ESC} \rangle$ "P" n1 n2 n3 . . . CHR\$(0)	Set vertical tabs
$\langle \text{ESC} \rangle$ "a" n	One-time vertical tab of n lines



Chapter 6

Special Features of the Delta Printer

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 groupings that we have studied, but they add a lot of capability to Delta. So here goes.

Now hear this

You may have heard Delta'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 had a bell in them 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*. Well 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. So with our trivia lesson out of the way, let's see how we can "ring the bell."

The code to sound Delta's "bell" is CHR\$(7), which is ASCII code 7 or <BEL>. Any time Delta receives this code it will sound the bell for a quarter of a second. This can be used to remind an operator to change the paper or to make another adjustment to the printer.

You can try this by typing:

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

There are two other codes that affect the bell. One disables the bell, so that Delta will ignore a CHR\$(7), and the other turns the bell back on. All three codes that affect the bell are shown in the following table.

Table 6-1
Bell commands

Function	Control code
Sound bell	CHR\$(7)
Disable bell	<ESC> "Y" CHR\$(0)
Enable bell	<ESC> "Y" CHR\$(1)

Initializing Delta

Up to now when we wanted to reset Delta to the power on condition we have had to either turn the printer off and then 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 Delta's features to the power on condition (as determined by the DIP switches), with two exceptions. Those exceptions are that <ESC> "@" will not erase any characters that you have stored in Delta's RAM memory (Chapter 7 tells you how to create your own characters), and it won't erase the macro if you

have one stored in Delta's RAM (this chapter will tell you how to create a macro).

Putting Delta to sleep

You know how to put Delta *off-line* with the ON LINE button so that you can use the FF and LF buttons. Delta has another *off-line* state that can be controlled from your computer. When you turn Delta *off-line* from your computer, Delta will ignore anything that you send it, except for the code to go *on-line* again. CHR\$(19) is the code to turn Delta *off-line*; CHR\$(17) returns Delta to *on-line* status.

Printing to 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 Delta when you are about 2½ inches from the bottom of the sheet. This is to notify you if you are running out of continuous paper.

Delta 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 learned are in the following table.

Table 6-2
Some miscellaneous commands

Function	Control code
Master reset	<ESC> "@"
<i>Off-line</i>	CHR\$(19)
<i>On-line</i>	CHR\$(17)
<i>Paper-out</i> detector off	<ESC> "8"
<i>Paper-out</i> detector on	<ESC> "9"

Unidirectional printing

Unidirectional printing is a big word that means *printing in one direction only*. Delta normally prints when the printhead is moving in both directions. But once in a while you may have an application where you are more concerned about how the vertical lines align than with how fast it prints. Delta lets you make this choice. The table below shows the commands for controlling how Delta prints.

Table 6-3
Printing direction

Function	Control code
Print in one direction	<ESC> "U" CHR\$(1)
Print in both directions	<ESC> "U" CHR\$(0)

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

```

NEW
10 LPRINT CHR$(27) "A" CHR$(7);
20 FOR I = 1 TO 10
30 LPRINT "|"
40 NEXT I
50 LPRINT : LPRINT
60 LPRINT CHR$(27) "U" CHR$(1);
70 FOR I = 1 TO 10
80 LPRINT "|"
90 NEXT I
100 LPRINT CHR$(12) CHR$(27) "@";

```

Here is what you will get. The top line is printed bidirectionally, and the bottom is printed unidirectionally. You will have to look hard because there isn't much difference.

Let's analyze the program. Line 10 sets the line spacing to 7/72 of an inch so that the characters that we print will touch top to bottom. Lines 20-40 print 10 vertical line characters. Then line 60 sets one-direction printing and the vertical lines are printed again. Finally line 100 sends a form feed to advance the paper to the top of a new page, and then uses the master reset to restore Delta to the power-on condition.

Backspace and delete

Backspace (CHR\$(8)) "backs up" the printhead so that you can print two characters right on top of each other. Each time Delta receives a backspace it moves the printhead 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 Delta's buffer, not from the paper).

The following program shows how these two codes work.

```
NEW
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"
```

Here is what this program will print:

```
BACKSPACE DOES NOT WORK
DELETE DOES WORK
```

The backspace codes in line 20 move the printhead 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.

The seven bit dilemma

Certain computers (most notably the Apple II) don't have the capability to send eight bits on their parallel interface. They can only send seven bits. This would make it impossible for these

computers to use Delta's block graphics characters and special symbols if Star's 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 6-4
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, Delta has a whole different set of characters that are for special uses. These characters include block graphics characters for drawing forms and graphs, and special symbols for mathematical, engineering and professional uses. The following program will print out all of the graphics characters available.

```
NEW
1Ø FOR J = 16Ø TO 255 STEP 8
2Ø FOR I = J TO J + 7
```

160 = √	161 = ∽	162 = ∽	163 = ∽
168 = ◊	169 = ∗	170 = ∽	171 = ∽
176 = ∽	177 = ∽	178 = ∽	179 = ∽
184 = ∽	185 = ∽	186 = ∽	187 = ∽
192 = ∽	193 = ∽	194 = ∽	195 = ∽
200 = ∽	201 = ∽	202 = ∽	203 = ∽
208 = ∽	209 = ∽	210 = ∽	211 = ∽
216 = ∽	217 = ∽	218 = ∽	219 = ∽
224 = ∽	225 = ∽	226 = ∽	227 = ∽
232 = ∽	233 = ∽	234 = ∽	235 = ∽
240 = ∽	241 = ∽	242 = ∽	243 = ∽
248 = ∽	249 = ∽	250 = ∽	251 = ∽

Figure 6-1.

```

30 LPRINT I "=" CHR$(I) CHR$(9);
40 NEXT I : LPRINT : NEXT J

```

Figure 6-1 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 (unless you have set DIP switch 2-3 on by mistake). You can get the correct printout by changing line 30 to this:

```

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

```

So how are all of these strange characters used? Here is a short program that demonstrates how the graphics characters can be combined to create figures. If you have a 7-bit interface, add lines 5 and 70 shown below the main listing.

```

NEW
10 LPRINT CHR$(27) "A" CHR$(6);
20 LPRINT CHR$(235) CHR$(231) CHR$(231) CHR$(236)
30 LPRINT CHR$(233) CHR$(163) CHR$(161) CHR$(234)
40 LPRINT CHR$(233) CHR$(162) CHR$(160) CHR$(234)
50 LPRINT CHR$(237) CHR$(232) CHR$(232) CHR$(238)
60 LPRINT CHR$(27) "2";

```

164 = †	165 = ‡	166 = †	167 = ‡
172 = †	173 = †	174 = †	175 = †
180 = †	181 = †	182 = †	183 = †
188 = †	189 = †	190 = †	191 = †
196 = †	197 = †	198 = †	199 = †
204 = †	205 = †	206 = †	207 = †
212 = †	213 = †	214 = †	215 = †
220 = †	221 = †	222 = †	223 = †
228 = †	229 = †	230 = †	231 = †
236 = †	237 = †	238 = †	239 = †
244 = †	245 = †	246 = †	247 = †
252 = †	253 = †	254 = †	255 = †

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

```
5 LPRINT CHR$(27) ">";
70 LPRINT CHR$(27) "=";
```

In this program line 10 sets the line spacing to 6 dots which is the height of the graphics characters. Then lines 20-50 print the figure, and line 60 resets the line spacing to 1/6 inch. Here is what this program prints:



International character sets

Delta is a multi-lingual printer for it can speak in eight languages! Delta changes languages by changing 11 characters that are different for the different languages. These sets of characters are called *international character sets*. The control codes to select the international character sets are given in the following table.

Table 6-5
International character set commands

Country	Control code
U.S.A.	<ESC> "7" CHR\$(0)
England	<ESC> "7" CHR\$(1)
Germany	<ESC> "7" CHR\$(2)
Denmark	<ESC> "7" CHR\$(3)
France	<ESC> "7" CHR\$(4)
Sweden	<ESC> "7" CHR\$(5)
Italy	<ESC> "7" CHR\$(6)
Spain	<ESC> "7" CHR\$(7)

The characters that change are shown in Table 6-6.

The macro control code

The last of our group of miscellaneous control codes is definitely not the least. It is a *user-defined* control code, called a *macro*

Table 6-6
International character sets

Country	35	64	91	92	93	94	96	123	124	125	126
U.S.A.	#	@	[\]	^	'	{		}	~
England	£	@	[\]	^	'	{		}	~
Germany	#	§	Ä	Ö	Ü	^	'	ä	ö	ü	β
Denmark	#	@	Æ	Ǿ	Å	^	'	æ	ø	å	~
France	£	à	°	ç	§	^	'	é	ù	è	..
Sweden	#	É	Ä	Ö	Å	Ü	é	ä	ö	å	ü
Italy	#	§	°	ç	é	^	ù	à	ò	è	ì
Spain	#	@	ì	Ñ	¿	^	'	..	ñ	}	~

control code. The term *macro* is from the jargonesque *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. Delta’s macro can save you a lot of time and effort also.

Here is how Delta’s macro works. You *define* your macro by telling Delta what normal control codes are to be included in the macro. Then you can use the macro any time that you want and Delta 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 below.

Table 6-7
Macro instruction commands

Function	Control code
Define macro	<ESC> “+” . . . codes you include . . . CHR\$(30)
Use macro	<ESC> “!”

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$(27) "+";           ' START DEFINITION
   OF MACRO
20 LPRINT CHR$(18);              ' PICA
30 LPRINT CHR$(27) "W" CHR$(0);  ' EXPANDED OFF
40 LPRINT CHR$(27) "F";         ' EMPHASIZED OFF
50 LPRINT CHR$(27) "H";         ' DOUBLE-STRIKE OFF
60 LPRINT CHR$(27) "-" CHR$(0);  ' UNDERLINE OFF
70 LPRINT CHR$(27) "T"          ' SUPER & SUBSCRIPTS
   OFF
80 LPRINT CHR$(27) "5";         ' REGULAR PRINT
90 LPRINT CHR$(30);             ' END MACRO
   DEFINITION

```

As the comments in the program listing show this will define a macro that will reset all the print style functions. Delta 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 fifteen.

Now that you have defined a macro, let's see how to use it. This program will print one line using several printing style features. Then it "calls" the macro in line 50. When line 60 prints the style is "plain vanilla" because the macro has reset it.

```

10 LPRINT CHR$(27) "4";           ' ITALIC
20 LPRINT CHR$(27) "G";           ' DOUBLE-STRIKE
30 LPRINT CHR$(27) "W" CHR$(1);  ' EXPANDED
40 LPRINT "TESTING ONE, TWO, THREE"
50 LPRINT CHR$(27) "!";         ' USE THE MACRO
60 LPRINT "TESTING FOUR, FIVE, SIX"

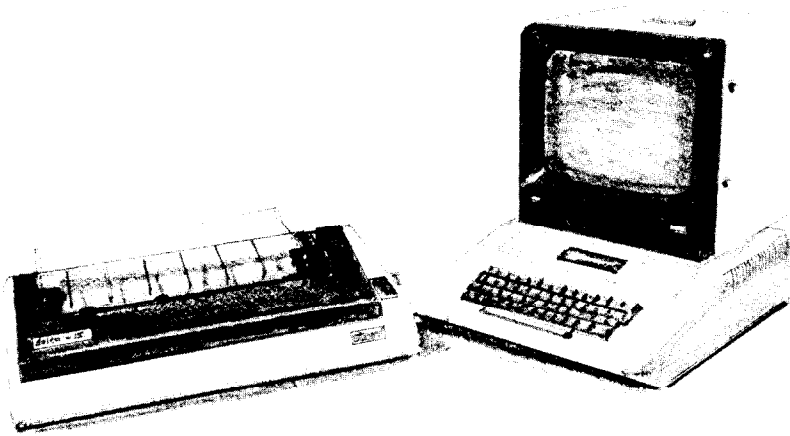
```

TESTING ONE, TWO, THREE
 TESTING FOUR, FIVE, SIX

In this chapter we have learned many different commands that have many different uses. In the next chapter we will make up for this diversity—the whole chapter only covers three commands! But they are some of the most powerful that Delta offers. They give you the ability to create your own characters.

Summary

Control code	Function
CHR\$(7)	Bell
<ESC> "Y" CHR\$(0)	Disable bell
<ESC> "Y" CHR\$(1)	Enable bell
<ESC> "@"	Reset
CHR\$(19)	Off-line
CHR\$(17)	On-line
<ESC> "8"	Paper-out detector off
<ESC> "9"	Paper-out detector on
<ESC> "U" CHR\$(1)	Unidirectional printing
<ESC> "U" CHR\$(0)	Bidirectional printing
CHR\$(8)	Backspace
CHR\$(127)	Delete
<ESC> ">"	Eighth bit on
<ESC> "="	Eighth bit off
<ESC> "#"	Eighth bit as-is
<ESC> "7" n	Select international character set
<ESC> "+" ... CHR\$(30)	Define macro
<ESC> "!"	Use macro



Chapter 7

Creating Your Own Characters

In the previous four chapters of this manual you've learned how to control the Delta printer to give you dozens of different typefaces. By using various combinations of pitches, character weights, and font selections, you can create nearly any effect you want to in text. And with international character sets and the special text and graphics characters described in Chapter 6, you can print almost any character you can think of.

But if "almost any character" isn't good enough for you, then it's a good thing you have a Delta printer! With it you can actually create your own characters. As you'll see in this chapter, *download characters* can be used to print a logo, special characters for foreign languages, scientific and professional applications, or any other specific printing task.

Dot Matrix Printing

In order to create download characters, you'll need some understanding of how dot matrix printers work. They're called "dot matrix" because each character is made up of a group of dots. Look closely at some printed characters produced by your Delta and you will see the dots. Figure 7-1 shows how the letter "A" is formed by printing 17 dots.

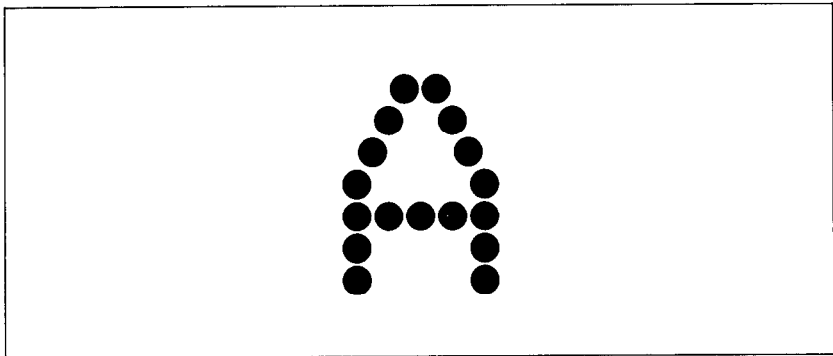


Figure 7-1. The letter "A" is created by printing 17 dots.

The printhead in Delta consists of nine thin wires stacked one atop the other. Figure 7-2 shows an enlarged schematic view of the front of the printhead, showing the ends of the wires and their relationship to the printed characters. As you can see, the capital letters use the top seven wires of the printhead, and the descenders (such as the lower case "g" shown) use the bottom seven pins. As the printhead moves across the page (in either direction—that's what is meant by bi-directional printing) it prints one column of dots at a time. Each time a dot is supposed to print an electromagnet inside the printhead causes the appropriate wire to strike the ribbon (making Delta an *impact printer*).

The Print Matrix

All of the standard characters that Delta prints are formed from patterns of dots that are permanently stored in the printer's ROM (read-only memory). This includes all of the standard ASCII characters, the block graphics and special characters, the international character sets, and the italic characters.

But there is another area of memory in Delta reserved for

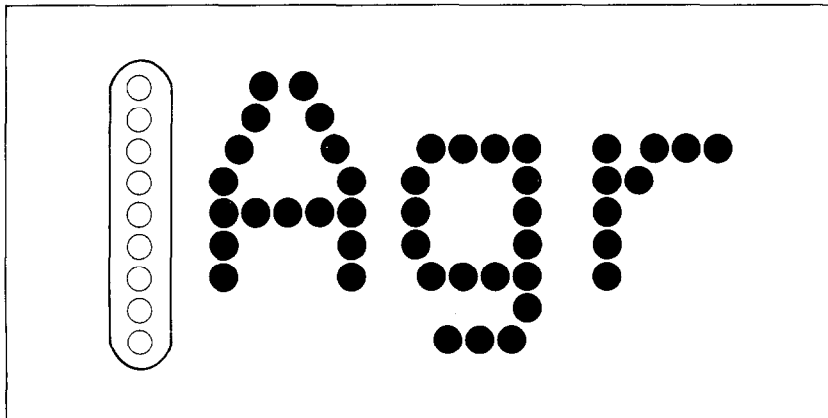


Figure 7-2. As the printhead moves across the page, each of the wires prints one row of dots.

user-defined characters. These are characters that you design and download into Delta. When download characters are defined they are stored in RAM (random access memory), which allows you to define or modify them at any time.

Each of these characters, whether it is from the standard character ROM or in download RAM, is constructed on a grid which is six “boxes” wide by nine “boxes” high. The dots used to print a character can be inside any of the boxes. In addition, a dot can straddle any of the vertical lines. As an example, take a look at the enlarged “9” superimposed on the grid in Figure 7-3. As you can see, some dots are inside the boxes, and some are centered on the vertical lines. This, in effect, makes the character grid 11 dots wide by 9 dots high. To see how the rest of the characters in the standard character ROM are constructed, take a look at Appendix J.

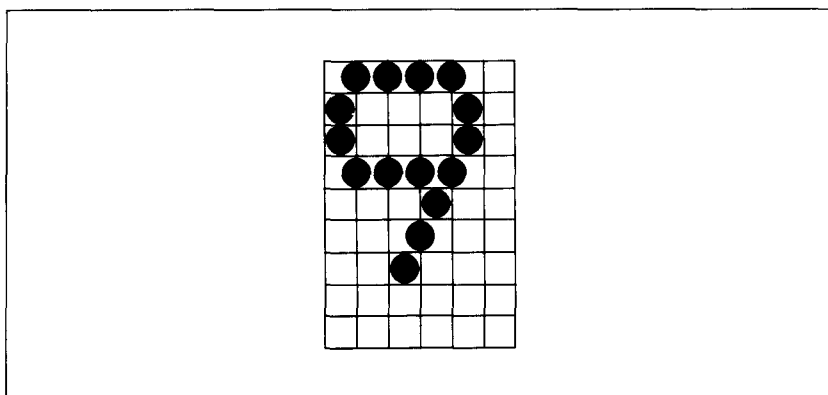


Figure 7-3. Dots can be inside boxes or straddle the vertical lines of the grid.

Defining Your Own Characters

You've seen how the engineers at Star designed their characters by using a grid to lay out the dots. Now you can define characters exactly the same way. Make up some grids (photocopy Figure 7-4 if you wish) and get ready to be creative! (Just in case you are not feeling creative, and to make our explanations a little clearer, we'll be using a heart as an example of a download character. You can see how we've laid it out in Figure 7-5. You'll find this especially useful if you've always wanted to write a bridge column like Charles Goren.)

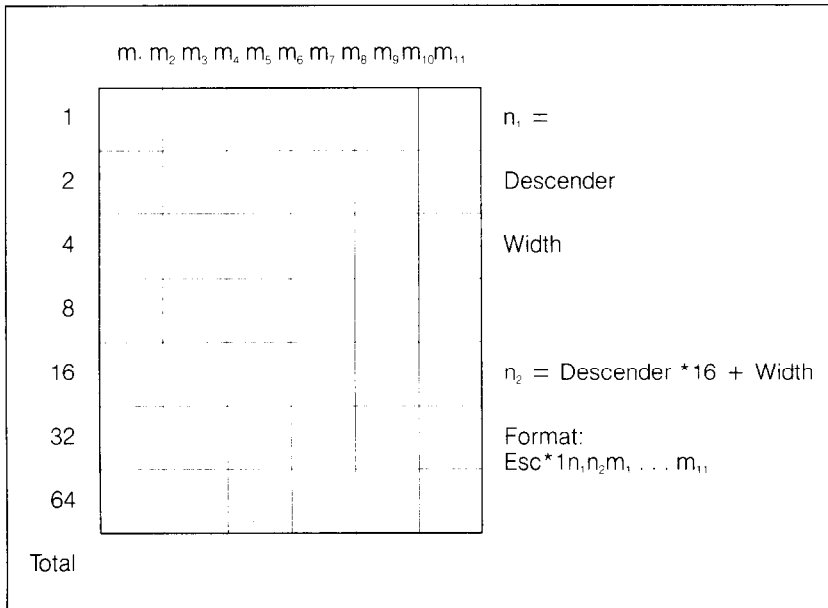


Figure 7-4. Use this grid (or one similar to it) to define your own characters.

You'll notice that Figure 7-4 includes a lot of information around the grid. Don't be intimidated; we'll explain each item as we come to it in our discussion of defining and actually printing download characters. You may have noticed another difference between this grid and the one shown in Figure 7-3: it's only seven boxes high. Which leads us to . . .

Rule 1: Download characters are seven dots high

As you noticed in Figure 7-2, capital letters, most lowercase

letters, and most special characters use only the top seven pins of the printhead. This is also the standard for download characters, so our grid is only seven dots high.

It's also possible to use the bottom seven pins, just as the "g", "p", "q", and "y" of the standard character sets do. These are called descenders (because the bottom of the character descends below the baseline of the rest of the characters).

One bit in the download character definition command is used to tell Delta whether a character is to be treated as a descender or not. We'll get to the command in due time. For now, if your character uses the top seven dots, write in a zero next to the word "Descender" on the layout grid; if it uses the bottom seven dots, write in a one. In our example, we'll want the bottom of the heart to line up with the baseline of the other characters, so it will not be a descender. As shown in Figure 7-5, we've written in a "0" on our grid.

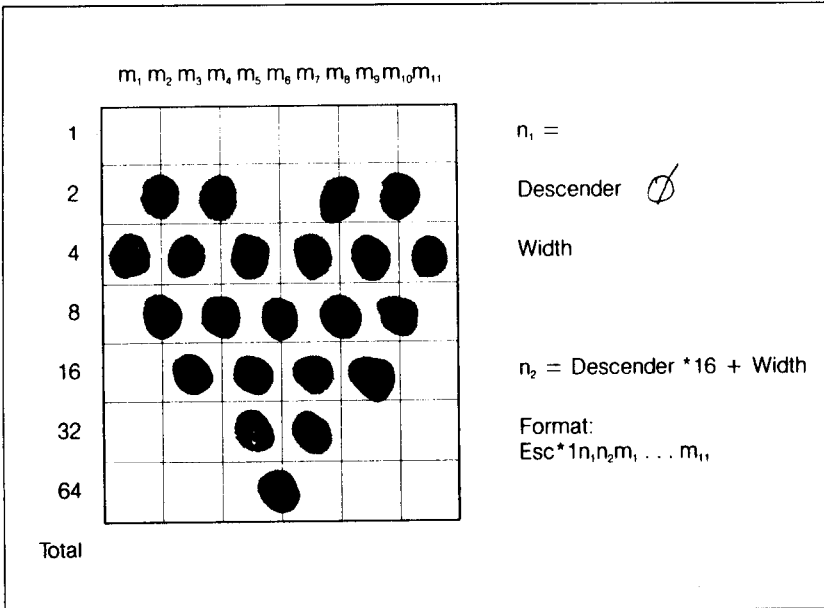


Figure 7-5. We've designed a character and decided that it would not be a descender, hence the "0" written in.

Rule 2: Dots cannot overlap

As you can see in Figure 7-5 our heart will print fairly solid. But, you may ask, why not make it really solid and print all the intermediate dots, as shown in Figure 7-6? Because the dots that straddle the vertical lines in the grid actually overlap those inside

the boxes. If we tried to print overlapping dots, Delta's print head would have to slow down and back up to print both dots—not very efficient! To avoid this inefficiency, Delta will not allow you to define a character like Figure 7-6. (Actually, you can define it, but when it prints, Delta will leave out the overlapping dots, so that it would print like Figure 7-5.)

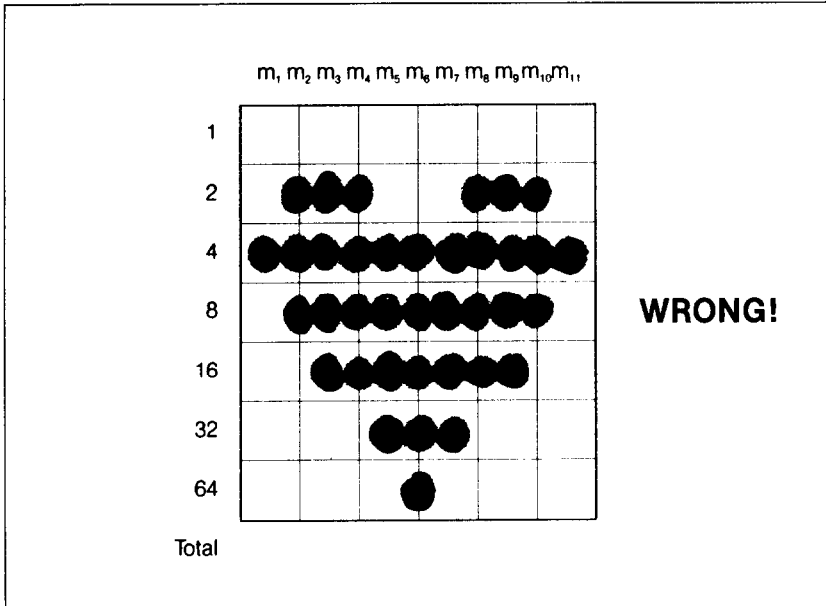


Figure 7-6. Dots cannot overlap; those in immediately adjacent “half columns” will be ignored when the character is printed.

Add up each column of dots

Now it's time to give our creative side a break and get down to some basic arithmetic. That's where the numbers down the left side of the grid come in. Notice that there is a number for each row of dots and that each number is twice the previous number. By making these numbers powers of two we can take any combination of dots in a vertical column and assign them a unique value. Some examples will make this clearer. As shown in Figure 7-7, if we add the numbers for the dots that print in a column, the sum will be a number in the range of 0 to 127. Each number from 0-127 represents a unique combination of dots.

So add up the values of the dots in each column using this system. This way it takes one number to describe each column of dots. In Figure 7-8 we've shown our grid with the sums of the columns filled in across the bottom (see if these agree with your

1		● - 1	● - 1
2	● - 2	● - 2	● - 2
4		● - 4	● - 4
8	● - 8		● - 8
16			● - 16
32	● - 32		● - 32
64		● - 64	● - 64
Sum	42	71	127

Figure 7-7. By adding the values of each dot in a column, you'll get a unique description for any combination of dots.

answers!). Across the top of the grid you've probably noticed the cryptic labeling of each column: m_1, m_2, m_3 , etc. These labels correspond to the labels in the command syntax statement, which we'll get to shortly.

	$m_1, m_2, m_3, m_4, m_5, m_6, m_7, m_8, m_9, m_{10}, m_{11}$	
1		$n_1 =$
2	● ● ● ●	Descender
4	● ● ● ● ● ●	Width
8	● ● ● ● ●	
16	● ● ● ●	$n_2 = \text{Descender} * 16 + \text{Width}$
32	● ●	Format: Esc*1 $n_1, n_2, m_1, \dots, m_{11}$
64	●	
Total	4 20 52 52 20 4 10 10 72 10 10	

Figure 7-8. Add the values of the dots in each column and write the sum of each column at the bottom.

Assigning a value to your character

We've done a pretty thorough job of designing and describing a user-defined character. But the Delta has room for 189 download characters—how does it know *which* user-defined character we want to print? Exactly the same way it knows which standard character we want to print: every character is assigned a unique number.

The standard characters are assigned the ASCII codes—numbers from 0 to 255. For the download character sets there are two banks of characters that can be defined: values from 33 to 126 and 160 to 254. This means that once a character is defined and assigned a value (and the download character set is selected), you can use that character on the printer the same way you would any standard character. You can send the character with the same ASCII value (for instance, if you had assigned your character a code of 66, it would print each time you sent a character "B" to the printer). You can also access the character from a BASIC program with the CHR\$ function—in this case LPRINT CHR\$(66) would print the character.

Except for the limitation that download characters must be assigned values in the range of 33 to 126 or 160 to 254, there are no rules or restrictions on the use of numbers. This means you can use whatever is most convenient for you—perhaps seldom-used keys can be replaced by more useful characters. In our example, we'll assign the heart a value of 72, which is the ASCII value for the letter "H". This way, when we want to print a heart, all we have to do is send the printer an "H"—that's easy to remember!

We could hardly write bridge columns with just a heart, so in Figure 7-9 we've made completed grids for all four card suits. In order to make them easy to use, we've assigned the club a value of 67 (the ASCII value for "C"), the diamond is 68 ("D"), and the spade is 83 ("S"). The information on the grids is now complete (except for proportional width data—a more advanced topic we'll take up shortly).

Download character definition command

You've read through a long explanation of download characters and we haven't even told you the command syntax yet! Now the wait is over. This is the most complex command in the Delta repertoire and now you've got the necessary knowledge to implement it. Here it is:

```
<ESC> "*" CHR$(1) n1 n2 m1 m2 m3 m4 m5 m6 m7 m8 m9 m10 m11
```

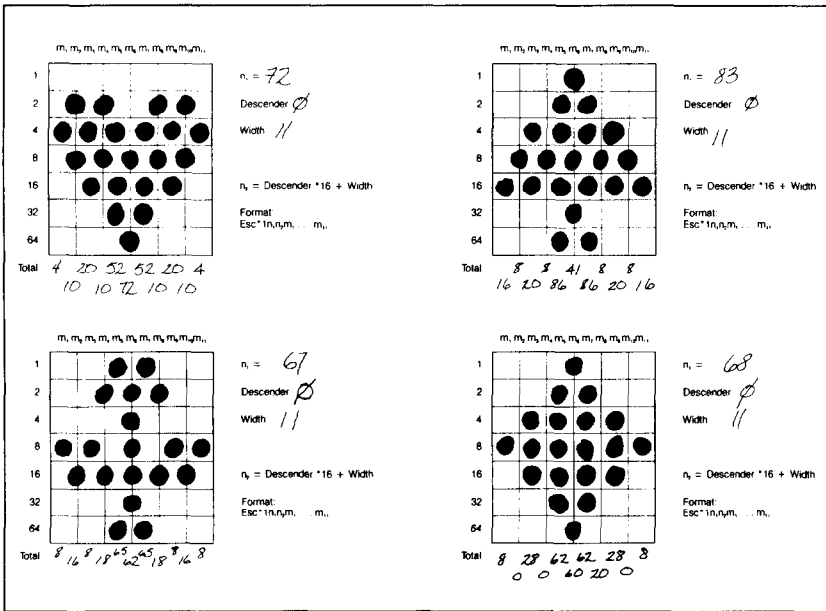


Figure 7-9. Character designs for the four card suits.

Like the other Delta commands, it starts with an <ESC> (CHR\$(27)). The next character is an asterisk (*), which is CHR\$(42), followed by a CHR\$(1).

n1 is the value we assign to the character—in the case of the heart it is CHR\$(72).

n2 is called the *attribute byte*, for it describes two attributes of the character we have designed: descender data and proportional width information. A byte consists of eight bits. In the attribute byte, the first three (high order) bits are unused, the fourth bit is used for the descender data, and the last four bits are used for proportional widths. We'll be discussing proportional character widths in detail later in this chapter; for now, we'll leave it at 11. The descender data was discussed earlier: to use the top seven pins, this bit should be 0; to use the bottom seven pins this bit should be 1. Figure 7-10 shows the bits of the attribute byte as we'll use them for our heart character. Since the descender data is 0, the value of the byte is equal to the value of the proportional data—11. By now you've probably seen an easier way to determine the value of the attribute byte. Instead of translating everything to binary, merely assign the descender data a value of 16 (the value of the fourth bit) if you want descenders, or 0 if you don't want descenders. Then just add the descender data to the proportional width. This way, it's simply a matter of adding two decimal numbers. (In our case, it's 0 + 11 = 11.)

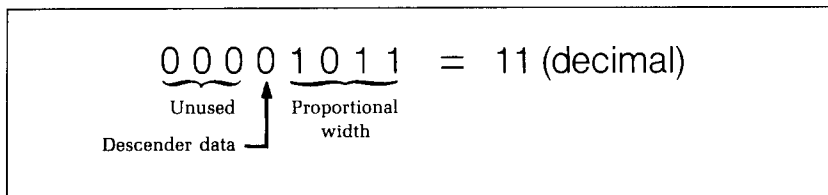


Figure 7-10. The attribute byte (n2) for our heart character.

You'll probably recognize m1. . .m11 from the top of our layout grid. That's right, each column is described by one byte. Now we've got everything we need to download one character to the printer. The complete command for our heart character is shown in Figure 7-11.

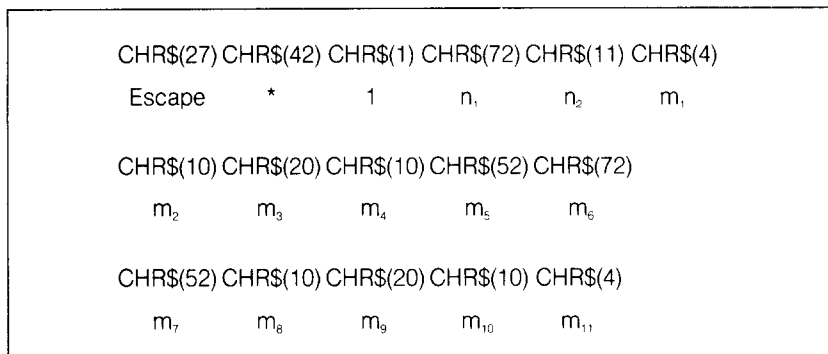


Figure 7-11. This is the complete command to send our heart character to the Delta printer.

Now let's send the information to the printer. The following program will send the character definitions for all four characters to the printer. Enter the program and run it.

```

10 FOR I=1 TO 4
20 LPRINT CHR$(27) "*" CHR$(1);
30 READ N1,N2
40 LPRINT CHR$(N1) CHR$(N2);
50 FOR M=1 TO 11
60 READ M1
70 LPRINT CHR$(M1);
80 NEXT M
90 NEXT

```

```

100 LPRINT
110 DATA 72,11,4,10,20,10,52,72,52,10,20,10,4
120 DATA 83,11,16,8,20,8,86,41,86,8,20,8,16
130 DATA 67,11,8,16,8,18,65,62,65,18,8,16,8
140 DATA 68,11,8,0,28,0,62,65,62,0,28,0,8

```

When you run this program, it looks like nothing happens. That's OK. We'll see why in just a moment. Save this program. We'll need it again shortly.

Printing Download Characters

You've now defined and sent four characters to the Delta. But how do you know that? If you try printing those characters now (type LPRINT "HCDS") you don't get a heart, club, diamond and spade. Instead you get. . .HCDS. That's because the download characters are stored in a different part of Delta's memory. To tell it to look in download character RAM instead of standard character ROM it requires another command:

```
<ESC> "$" CHR$(n)
```

This command is used to select the download character set (if $n=1$) or to select the standard character set (if $n=0$). Let's try it out. Enter this command:

```
LPRINT CHR$(27) "$" CHR$(1) "HCDS"
```

Voila! It should have printed out the four characters we defined. Your printout should look like this:

♥♣♦♠

(If it doesn't, check the last program we ran for errors, then re-run it.)

Let's find out if there are any other characters in the download RAM. Try this program:

```

10 LPRINT CHR$(27) CHR$(36) CHR$(1)
20 FOR I=33 TO 126 : LPRINT CHR$(I); : NEXT I
30 FOR I=160 TO 254 : LPRINT CHR$(I); : NEXT I
40 LPRINT
50 LPRINT CHR$(27) CHR$(36) CHR$(0)

```

Nope! Just four characters in the download set. This is inconvenient for a couple of reasons. First, every time you wanted to use a download character you would have to switch back and forth between character sets. Knowing that you wouldn't want to do that, Delta won't even allow it. Standard characters and download characters cannot be mixed in a line. If you want to use download characters, the command should appear at the beginning of the line. All subsequent characters (even on following lines) are printed with the download set until you return to the standard characters with an `<ESC> "$" CHR$(0)`. (Note that the `<ESC> "$" CHR$(1)` command can be in the middle of a line, and that *entire line* will be printed with the download characters. Likewise, if you select the standard character set anywhere in a line, the *entire line* will be printed with the standard characters. Conflicting commands within a line can cause unpredictable results.)

So does that mean that in order to print something meaningful with our card suits we have to define an entire alphabet? Fear not. The engineers at Star have made it an easy task to use mostly standard characters with just a few special characters thrown in. This command copies all the characters from the standard character ROM into download RAM:

```
<ESC> "*" CHR$(0)
```

Since it will copy *all* characters into the download area, it will wipe out any characters that are already there. So it's important to send this command to the printer before you send any download characters you want to define. With that in mind, add this line to the program we used to send the characters to Delta:

```
5 LPRINT CHR$(27) "*" CHR$(0)
```

Now try the download printout test program again. Your results should look like Figure 7-12. You probably noticed that our

printout test includes the characters with ASCII values from 160 to 254, but nothing prints. The <ESC> "*" CHR\$(0) command copies only the standard ASCII characters (those in the range of 33 to 126) to download RAM; it does not copy any block graphics characters.

```
!"#$%&'()*+,-./0123456789:;<=>?@AB♣♦EF♠♥
IJKLMNOPQR♠TUVWXYZ[\]^_`abcdefghijklmnop
qrstuvwxyz{|}~
```

Figure 7-12. Printout of the download character set, into which all the standard characters have been copied, and the C, D, H, and S have been changed.

To demonstrate how to use these characters, let's use this character set to print a typical bridge hand. This program will do just that:

```
10 'Program to deal bridge hands and print on Delta
20 GOSUB 10000 'Initialize variables
30 GOSUB 20000 'Initialize printer
40 GOSUB 30000 'Deal cards
50 GOSUB 40000 'Print hands
60 END
1000 'Initialize variables
1010 DEFINT A-Z
1020 DIM HAND(4), DECK(52), CARD$(13), SUIT$(3)
1030 CARD$(1)=" 2" : CARD$(2)=" 3" : CARD$(3)=" 4" :
    CARD$(4)=" 5" : CARD$(5)=" 6"
1040 CARD$(6)=" 7" : CARD$(7)=" 8" : CARD$(8)=" 9" :
    CARD$(9)=" 10"
1050 CARD$(10)=" J" : CARD$(11)=" Q" : CARD$(12)="
    K" : CARD$(13)=" A"
1060 SUIT$(0)="S" : SUIT$(1)="H" : SUIT$(2)="D" :
    SUIT$(3)="C"
1070 INPUT "Random number seed";I
1080 RANDOMIZE I
1090 RETURN
2000 'Initialize printer
2010 LPRINT CHR$(27) CHR$(68) CHR$(20) CHR$(40)
    CHR$(0) 'Set tabs
```

```

2020 LPRINT CHR$(27) CHR$(43) CHR$(27) CHR$(36)
      CHR$(0) CHR$(27) CHR$(69) CHR$(30)          'Macro
      instruction is used to select standard
      characters, emphasized
2030 LPRINT CHR$(27) CHR$(42) CHR$(0)
      'Load standard characters in RAM
2040 FOR I=1 TO 4
      'This loop reads data for the four
2050 LPRINT CHR$(27) CHR$(42) CHR$(1);
      'card suit characters and sends it
2060 FOR J=1 TO 13      'to the printer
2070 READ X : LPRINT CHR$(X);
2080 NEXT J
2090 NEXT I
2100 LPRINT
2110 RETURN
2120 DATA 72,11,4,10,20,10,52,72,52,10,20,10,4
2130 DATA 83,11,16,8,20,8,86,41,86,8,20,8,16
2140 DATA 67,11,8,16,8,18,65,62,65,18,8,16,8
2150 DATA 68,11,8,0,28,0,62,65,62,0,28,0,8
3000 'Deal cards
3010 FOR CARD = 1 TO 52
3020 X = INT(RND * 4 + 1)
3030 IF HAND(X)=13 THEN 3020 ELSE HAND(X)=HAND(X)+1
3040 DECK(CARD)=X
3050 NEXT CARD
3060 RETURN
4000 'Print four hands
4010 LPRINT CHR$(27) "!" CHR$(9) "NORTH"
4020 LPRINT CHR$(27) "$" CHR$(1) CHR$(27) CHR$(70);
4030 HAND = 1
4040 FOR SUIT = 0 TO 3
4050 LPRINT CHR$(9);
4060 GOSUB 4300
4070 LPRINT
4080 NEXT SUIT
4090 LPRINT CHR$(27) "!" "WEST" CHR$(9) CHR$(9)
      "EAST"
4100 LPRINT CHR$(27) "$" CHR$(1) CHR$(27) CHR$(70);
4110 FOR SUIT = 0 TO 3
4120 HAND = 2
4130 GOSUB 4300
4140 LPRINT CHR$(9) CHR$(9);
4150 HAND = 3
4160 GOSUB 4300

```

```

4170 LPRINT
4180 NEXT SUIT
4190 LPRINT CHR$(27) "!" CHR$(9) "SOUTH"
4200 LPRINT CHR$(27) "$" CHR$(1) CHR$(27) CHR$(70);
4210 HAND = 4
4220 FOR SUIT = 0 TO 3
4230 LPRINT CHR$(9);
4240 GOSUB 4300
4250 LPRINT
4260 NEXT SUIT
4270 LPRINT CHR$(27) "$" CHR$(0) CHR$(27) CHR$(70)
4280 RETURN
4290 'Print one line
4300 LPRINT SUIT$(SUIT);
4310 FOR CARD = 13 TO 1 STEP -1
4320 IF DECK(SUIT*13+CARD)=HAND THEN LPRINT
      CARD$(CARD);
4330 NEXT CARD
4340 RETURN
    
```

Note that we didn't have to re-enter the download characters, since they were already sent to the printer with the previous program. They will stay with the printer until you download new characters to replace them or turn the printer off. Even the <ESC> "@ " command, which initializes the printer, does not destroy the contents of download RAM.

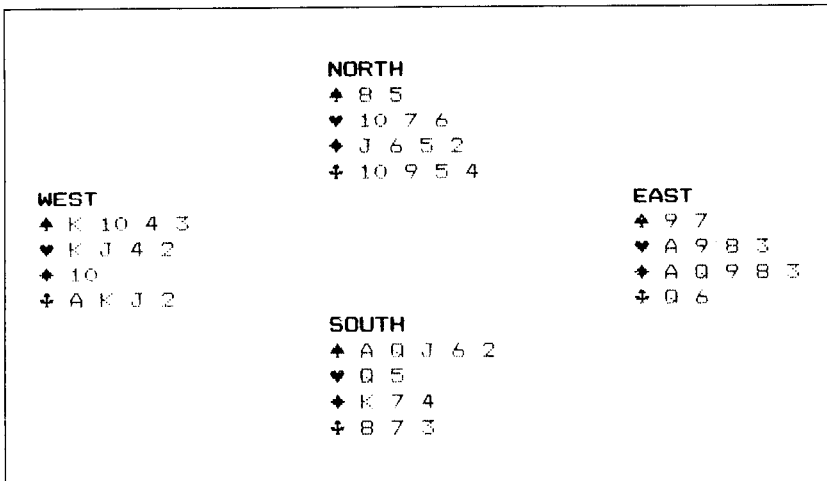


Figure 7-13. The card program shuffles, deals, and prints out a bridge hand.

Table 7-1
Download character definition commands

Function	Control code
Define download character	<ESC> "*" CHR\$(1) n1 n2 m1 . . . m11
Copy ROM to download RAM	<ESC> "*" CHR\$(0)

Proportional Characters

Up until now, all the characters that your Delta has printed have been of a fixed width—either 10, 12, or 17 (or 5, 6 or 8.5 in enlarged mode) characters per inch. Whichever pitch you select, all the characters are the same width. You'll notice though, that in typeset books, such as this one, each character has a slightly different width. For instance, the "i" is quite narrow, and the "W" is very wide. This is more pleasing to the eye and easier to read.

So, if you're going to go to the trouble of designing your own download characters for Delta, you might as well make them pleasing to the eye! Proportional download characters allow you to do just that. As you'll remember from our initial discussion of download character definition, part of the attribute byte is for proportional width data. We skipped over that, with the promise of describing it later. Well now is the time!

Defining proportional characters

Except for the actual width, defining characters for proportional printing is exactly the same as defining normal width download characters. Characters can range from 4 to 11 dots wide. This means that characters can be as narrow as one-third the normal width. The examples in Figure 7-14 show characters of different widths. These characters are defined in the program that follows.

```

10 DATA 77,11,1,126,1,2,4,8,4,2,1,126,1
20 DATA 105,4,64,61,64,0,0,0,0,0,0,0
30 DATA 112,23,127,0,17,0,17,14,0,0,0,0,0
40 DATA 115,6,8,84,0,84,32,0,0,0,0,0
50 DATA -1
60 READ CHR
70 IF CHR < 0 THEN 150
80 READ CODE
90 LPRINT CHR$(27) "*" CHR$(1) CHR$(CHR) CHR$(CODE) ;

```


One thing to remember about defining proportional characters: a character cannot be wider than the specified width. That seems obvious enough! For example, if you specify a width of 6 for a character, the seventh through eleventh columns of dots (if you specified any) will not print. You must, however, send information (even if it is 0) for those columns when you define a character; Delta expects eleven characters following the $\langle \text{ESC} \rangle$ “*” CHR\$(1) n1 n2 sequence.

In most cases, the width you select should actually be one dot *wider* than the number of columns that the character actually occupies. This is so that there will be a space (of one dot) between characters when you print them. If you specify a width which is exactly the same as the number of columns in the character definition, the characters will touch when they print (this is sometimes desirable—for border characters or for large download characters that are more than eleven dots wide).

Printing proportional characters

Printing with proportional download characters is much like using normal width download characters: one command is used to select the download set or the standard character set. Here's the command:

```
 $\langle \text{ESC} \rangle$  "X" CHR$(n)
```

If n is 1, then the download character set is selected, and proportional widths are used. If n is 0, the standard character set is selected.

It should be noted that it is possible to use the same character definitions for either normal width or proportional download characters (if a valid proportional width is included in the attribute byte). The only difference is the way they are accessed: $\langle \text{ESC} \rangle$ “\$” CHR\$(1) for normal width or $\langle \text{ESC} \rangle$ “X” CHR\$(1) for proportional width. The two commands work independently of each other, so that $\langle \text{ESC} \rangle$ “\$” CHR\$(0) will *not* turn off proportional download characters, and $\langle \text{ESC} \rangle$ “X” CHR\$(0) will *not* turn off normal width download characters. If you have selected both normal and proportional download characters, proportional will print until you send the printer an $\langle \text{ESC} \rangle$ “X” CHR\$(0). The printer will then continue to print with normal width download characters (rather than returning to the standard character set) until you send an $\langle \text{ESC} \rangle$ “\$” CHR\$(0). This can lead to confusion if you have accidentally specified both types of download characters.

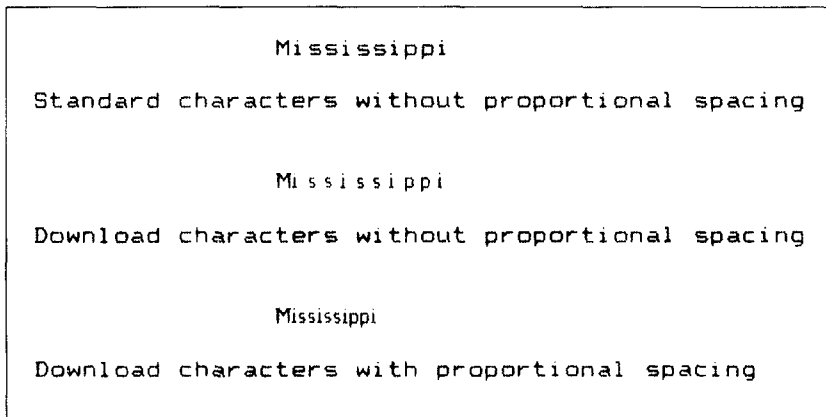


Figure 7-15. This printout shows the same text, printed with the same download characters, in both normal and proportional widths.

Table 7-2
Download character printing commands

Function	Control code
Normal download characters ON	<ESC> "\$" CHR\$(1)
Normal download characters OFF	<ESC> "\$" CHR\$(0)
Proportional download characters ON	<ESC> "X" CHR\$(1)
Proportional download characters OFF	<ESC> "X" CHR\$(0)

Connecting characters

As we noted earlier, it's possible to connect proportional width characters. This can be useful for creating logos or other characters which are larger than one normal character. It also makes it possible to create connecting scripts, like handwriting. The trick to this is to specify the width in the attribute byte to be exactly the same as the number of columns of dots that the character (or partial character) occupies. And, if you change the vertical spacing to 7/72" (use the <ESC> "1" command), you can make characters connect vertically. This allows you to make very large characters indeed!

In the program that follows, we've used this technique to create some large numbers. Each digit is actually made up of four characters—two horizontally by two vertically. This means, of course, that you must define and print four characters for each finished digit. We assigned the upper left quadrant of each digit to ASCII codes from 160 to 169, the upper right quadrant to codes 170 to 179, and so on. Figure 7-16 shows how one digit is defined, and Figure 7-17 shows the final output of our program.

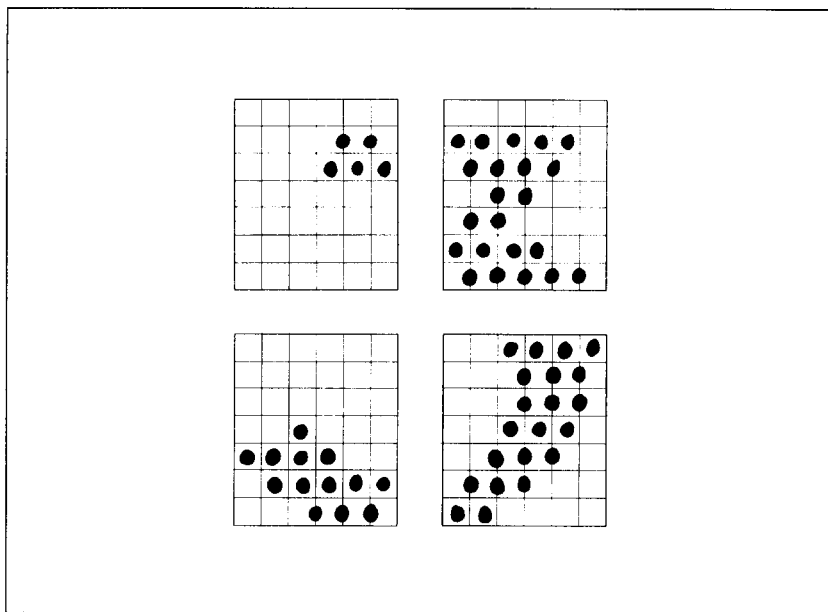


Figure 7-16. Each digit is made up of four individual characters.

```

10 'Program to define and print numerals
20 'Each numeral is made up of 4 characters (2 wide
   x 2 high)
30 DEF.DOWN.CHAR$ = CHR$(27)+CHR$(42)+CHR$(1)
40 DOWN.CHAR.PROP$ = CHR$(27)+CHR$(88)+CHR$(1)
50 NOT.DOWN.CHAR.PROP$ = CHR$(27)+CHR$(88)+CHR$(0)
60 LINE.7$ = CHR$(27)+CHR$(49) : LINE.12$ =
   CHR$(27)+CHR$(50)
70 FOR N1 = 160 TO 200      '4 ASCII CHARS/NUMERAL
80 LPRINT DEF.DOWN.CHAR$;
90 LPRINT CHR$(N1);
100 READ N2
110 LPRINT CHR$(N2);
120 FOR S = 1 TO 11
130 READ MS
140 LPRINT CHR$(MS);
150 NEXT S
160 NEXT N1
170 '
180 ASCII = 160             'START OF DOWN CHARACTERS
190 FOR NUM = 0 TO 9       'NUMERALS 0 THRU 9
200 NUMERAL.TOP$(NUM) = CHR$(ASCII + 0) +
   CHR$(ASCII + 1)

```



```
210 NUMERAL.BOT$(NUM) = CHR$(ASCII + 2) +
    CHR$(ASCII + 3)
220 ASCII = ASCII + 4
230 NEXT NUM
240 BLANK$ = CHR$(200)
250 LPRINT DOWN.CHAR.PROP$; LINE.7$
260 FOR NUM = 0 TO 9
270 LPRINT NUMERAL.TOP$(NUM);BLANK$;
280 NEXT NUM
290 LPRINT
300 FOR NUM = 0 TO 9
310 LPRINT NUMERAL.BOT$(NUM);BLANK$;
320 NEXT NUM
330 LPRINT NOT.DOWN.CHAR.PROP$; LINE.12$
340 'ZERO
350 DATA 11,0,96,16,104,16,44,30,14,0,2,1
360 DATA 11,2,1,2,1,6,8,38,88,32,88,32
370 DATA 11,3,12,19,12,51,0,96,0,96,0,96
380 DATA 11,0,32,0,48,0,28,3,12,3,4,3
390 'ONE
400 DATA 11,0,0,0,0,0,4,0,4,0,4,126
410 DATA 9,12,114,12,114,12,2,0,0,0,0,0
420 DATA 11,64,0,64,0,64,0,64,32,80,47,80
430 DATA 9,47,80,47,64,0,64,0,64,0,0,0
440 ' TWO
450 DATA 11,0,0,0,0,0,12,16,14,0,6,0
460 DATA 11,3,0,3,0,70,56,70,56,4,24,0
470 DATA 11,64,0,64,32,64,32,80,32,80,40,64
480 DATA 11,44,64,38,65,34,65,32,80,32,88,0
490 ' THREE
500 DATA 11,0,0,0,0,0,0,4,2,4,2,4
510 DATA 11,34,84,34,92,34,76,34,68,2,64,0
520 DATA 11,16,0,48,0,56,64,48,64,32,64,32
530 DATA 11,64,32,64,48,9,54,9,22,9,6,1
540 ' FOUR
550 DATA 11,0,0,0,0,0,0,64,36,88,32,16
560 DATA 11,0,0,64,32,64,56,64,60,2,12,0
570 DATA 11,0,8,4,10,5,10,5,8,4,72,4
580 DATA 11,88,38,89,38,89,6,73,4,8,6,0
590 ' FIVE
600 DATA 11,0,0,0,0,64,32,84,50,76,34,68
610 DATA 10,34,68,34,68,34,68,2,68,2,0,0
620 DATA 10,0,32,24,101,24,97,0,64,0,64,0
```

```

63Ø DATA 11,64,Ø,96,1,48,15,48,15,16,15,Ø
64Ø ' SIX
65Ø DATA 11,Ø,96,Ø,112,Ø,12Ø,Ø,92,Ø,1Ø2,Ø
66Ø DATA 11,98,Ø,98,Ø,98,Ø,7Ø,Ø,14,Ø,6
67Ø DATA 11,7,8,23,8,55,8,99,Ø,65,Ø,64
68Ø DATA 11,Ø,96,Ø,112,1,62,1,3Ø,1,14,Ø
69Ø ' SEVEN
7ØØ DATA 11,Ø,16,8,6,8,6,8,6,8,6,8
71Ø DATA 9,7Ø,8,1Ø2,8,54,8,6,Ø,2,Ø,Ø
72Ø DATA 11,Ø,64,Ø,96,Ø,12Ø,Ø,124,Ø,3Ø,1
73Ø DATA 9,6,1,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø
74Ø ' EIGHT
75Ø DATA 11,Ø,Ø,Ø,Ø,24,36,24,1Ø2,24,1Ø2,Ø
76Ø DATA 11,67,Ø,67,Ø,99,28,34,28,34,28,Ø
77Ø DATA 11,12,18,44,19,1Ø8,19,96,1,64,Ø,64
78Ø DATA 11,Ø,96,1,112,15,48,15,16,14,Ø,Ø
79Ø ' NINE
8ØØ DATA 11,Ø,Ø,12Ø,4,12Ø,6,12Ø,6,Ø,3,Ø
81Ø DATA 11,3,Ø,3,Ø,67,4,123,4,122,4,12Ø
82Ø DATA 11,48,Ø,56,Ø,113,Ø,99,Ø,99,Ø,99
83Ø DATA 11,Ø,115,Ø,57,Ø,31,Ø,15,Ø,7,Ø
84Ø ' SPACE
85Ø DATA 11,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø

```

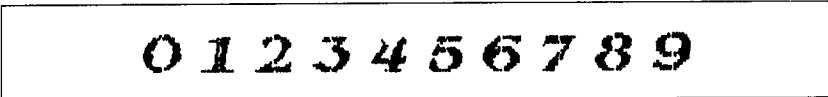


Figure 7-17. The output for characters like this must be carefully planned.

Mixing Print Modes with Download Characters

It's possible to get even more printing effects by combining download characters with the various print modes available with Delta. Most of the commands that you learned in Chapter 3 work with normal width download characters as well as standard characters. A few of them will work with proportional download characters as well. Table 7-3 summarizes the various print modes and their compatibility with download characters.

Table 7-3
Mixing download characters with various print modes

	Normal width (Escape \$)	Proportional (Escape X)
Standard Characters	*	*
Italic	-	-
Pica	*	*
Elite	*	-
Condensed	*	-
Expanded	*	-
Double-strike	*	-
Emphasized	*	-
Underline	*	*
Super/subscript	*	-

A Utility Program

If you've followed along this far you've probably become pretty proficient at designing download characters. And even the addition is getting easier! But this is a good computer application—Computer Aided Design (CAD) for download characters. The program below allows you to design and edit characters on the screen. You can make changes (no erasing!) until it's the way you like it, and then the program makes the necessary calculations and sends the character to Delta.

```

10 DIM Z(8,12),MM(11)
20 CLS:GOSUB 660
30 CS$=CHR$(16)+CHR$(17):SC$=STRING$(2,219):BIT=0
40 A$=INKEY$:IF A$="" THEN 40
50 IF A$=CHR$(27) THEN COLOR 7,0:CLS:END
60 IF A$="P" OR A$="p" THEN GOSUB 680:GOTO 40
70 IF A$="e" OR A$="E" THEN CLS:GOSUB 90:GOSUB
  260:GOTO 40
80 BEEP:GOTO 40
90 X=1:Y=1:G=1:H=1 : REM **** THIS SUBROUTINE
  DRAWS THE MATRIX ****
100 FOR I=1 TO 11:MM(I)=0:NEXT I
110 J=2:FOR I=10 TO 20:LOCATE 2,I+J :J=J+2:PRINT
  "M";:NEXT I
120 J=1:FOR I=10 TO 20:LOCATE 3,I+J :J=J+2:PRINT
  I-9;:NEXT I

```

```

130 P1=1:M$=CHR$(179)+
    STRING$(2,32):N$=STRING$(2,196)+
    CHR$(197):L$=STRING$(2,196)+CHR$(193)
140 LOCATE 4,10:PRINT CHR$(218);CHR$(196);
150 FOR I=1 TO 10:PRINT
    CHR$(196);CHR$(194);CHR$(196); :NEXT I
160 PRINT CHR$(196);CHR$(191):LOCATE 5,10:FOR K=1 TO
    12:PRINT M$;:NEXT K:PRINT
170 FOR J=1 TO 6:LOCATE 5+P1,10:P1=P1+1:PRINT
    CHR$(195);
180 FOR K=1 TO 10:PRINT N$;:NEXT K
190 PRINT CHR$(196);CHR$(196);CHR$(180):LOCATE
    5+P1,10:P1=P1+1
200 FOR K=1 TO 12:PRINT M$;:NEXT K
210 PRINT:NEXT J:LOCATE 18,10:PRINT CHR$(192);
220 FOR I=1 TO 10:PRINT L$;:NEXT I
230 PRINT CHR$(196);CHR$(196);CHR$(217)
240 FOR I=0 TO 6:LOCATE 5+I*2,6:PRINT 2^I;:NEXT I
250 RETURN : REM **** END OF MATRIX SUBROUTINE
    ****
260 REM **** SINGLE CHARACTER INPUT @ EDIT LEVEL
    ****
270 LOCATE 5,11:PRINT CS$;:GOSUB 590
280 A$=INKEY$:IF A$="" THEN 280
290 B$=RIGHT$(A$,1)
300 IF B$=CHR$(75) THEN GOSUB 390:GOTO 370
310 IF B$=CHR$(77) THEN GOSUB 410:GOTO 370
320 IF B$=CHR$(80) THEN GOSUB 430:GOTO 370
330 IF B$=CHR$(72) THEN GOSUB 450:GOTO 370
340 IF B$=CHR$(82) THEN GOSUB 470:GOTO 370
350 IF B$=CHR$(83) THEN GOSUB 490:GOTO 370
360 IF B$=CHR$(79) THEN GOSUB 500:GOTO 380
370 GOTO 280
380 RETURN : REM **** END OF INPUT ****
390 GOSUB 920:Y=Y-3:H=H-1:IF Y<1 THEN BEEP:Y=1:H=1
400 GOSUB 950:RETURN
410 GOSUB 920:Y=Y+3:H=H+1:IF Y>31 THEN
    BEEP:Y=31:H=11
420 GOSUB 950:RETURN
430 GOSUB 920:X=X+2:G=G+1:IF X>13 THEN BEEP:X=13:G=7
440 GOSUB 950:RETURN
450 GOSUB 920:X=X-2:G=G-1:IF X<1 THEN BEEP:X=1:G=1
460 GOSUB 950:RETURN
470 IF Z(G,H-1)=1 OR Z(G,H+1)=1 THEN BEEP:RETURN

```

```
480 Z(G,H)=1:COLOR 31,1:LOCATE X+4,Y+10:PRINT
    SC$;:COLOR 7,0:RETURN
490 Z(G,H)=0:COLOR 7,0:LOCATE X+4,Y+10:PRINT
    CS$;:COLOR 7,0:RETURN
500 REM **** GET OUT OF EDIT MODE ****
510 FOR I=2 TO 10:LOCATE I,55:PRINT
    STRING$(20,32);:NEXT I
520 IF Z(G,H)=1 THEN LOCATE X+4,Y+10:COLOR 7,0:PRINT
    SC$;:GOTO 540
530 IF Z(G,H)=0 THEN LOCATE X+4,Y+10:COLOR 7,0:PRINT
    " ";
540 REM **** PRINT THE COLUMN - VALUES ****
550 FOR I=1 TO 11:FOR J=1 TO 7
560 MM(I)=MM(I)+Z(J,I)*2^(J-1):NEXT J:NEXT I
570 J=0:FOR I=1 TO 11:LOCATE 19,10+J:PRINT
    RIGHT$(STR$(MM(I)),3);:J=J+3:NEXT I
580 GOSUB 660 :RETURN
590 REM **** DISPLAY MENU FOR EDIT MODE ****
600 LOCATE 2,55:PRINT "cursor movement";
610 LOCATE 4,60:PRINT CHR$(24);:LOCATE 5,58:PRINT
    CHR$(27);" ";
620 PRINT CHR$(26);:LOCATE 6,60:PRINT CHR$(25)
630 LOCATE 8,55:PRINT "<ins>  insert";
640 LOCATE 9,55:PRINT "<del>  delete";
650 LOCATE 10,55:PRINT "<end>  exit edit";:RETURN
660 FOR I=1 TO 7:FOR J=1 TO 11:Z(I,J)=0:NEXT J:NEXT
    I
670 LOCATE 24,2:PRINT "E) EDIT    P) PRINTER    <ESC>
    ) END ";:RETURN
680 REM **** PRINT MODE ****
690 LOCATE 20,5:INPUT "NORMAL OR PROPORTIONAL    (N/
    P) -> ";AN$
700 IF AN$="N" THEN PR=0:GOTO 750
710 IF AN$="P" THEN GOTO 730
720 BEEP:GOTO 690
730 LOCATE 21,5:INPUT "ENTER THE PROPORTIONAL DATA
    (4-11) -> ";PR
740 IF PR<4 OR PR>11 THEN 730
750 LOCATE 22,5:INPUT "IF SHIFTED DOWN ENTER  1 ELSE
    ENTER 0  -> ";SH
760 IF SH<0 OR SH>1 THEN BEEP:GOTO 750
770 LOCATE 23,5:INPUT "ENTER YOUR ASCII CODE (33-126
    OR 160-254) -> ";AS
780 IF (AS<32 AND AS>126) OR (AS<160 AND AS>254)
    THEN 770
```

```

790 FOR I=20 TO 23:LOCATE I,5:PRINT
  STRING$(55,32);:NEXT I
800 IF SH=1 THEN SH=16 ELSE SH=0
810 N1=AS:N2=PR+SH
820 FOR I=1 TO 11:MM$=MM$+CHR$(MM(I)):NEXT I
830 LPRINT
  CHR$(27);"*";CHR$(1);CHR$(N1);CHR$(N2);MM$
840 IF AN$="N" THEN LPRINT CHR$(27);"$";CHR$(1)
  :GOTO 860
850 LPRINT CHR$(27);"X";CHR$(1)
860 FOR I=1 TO 20:LPRINT CHR$(N1);" ";:NEXT I:LPRINT
870 LPRINT CHR$(14);:FOR I=1 TO 10:LPRINT CHR$(N1);"
  ";;NEXT I:LPRINT CHR$(20)
880 LPRINT CHR$(15);:FOR I=1 TO 20:LPRINT CHR$(N1);"
  ";;NEXT I:LPRINT CHR$(18)
890 IF AN$="N" THEN LPRINT CHR$(27);"$";CHR$(0):GOTO
  910
900 LPRINT CHR$(27);"X";CHR$(0)
910 LPRINT CHR$(27);"@":MM$="":RETURN :REM **** END
  OF PRINT MODE ****
920 IF Z(G,H)=0 THEN LOCATE X+4,Y+10:PRINT " ";
930 IF Z(G,H)=1 THEN LOCATE X+4,Y+10:COLOR 7,0:PRINT
  SC$;
940 RETURN
950 IF Z(G,H)=1 THEN COLOR 31,1: LOCATE
  X+4,Y+10:PRINT CS$;; COLOR 7,0
960 IF Z(G,H)=0 THEN COLOR 7,0: LOCATE
  X+4,Y+10:PRINT CS$;; COLOR 7,0
970 RETURN

```

Summary

Control code	Function
<ESC> "*" CHR\$(1) n1 n2 m1 . . . m11	Defines download character into RAM
<ESC> "*" CHR\$(0)	Copies fonts in ROM into download RAM
<ESC> "X" CHR\$(1)	Selects the download character set and uses proportional spacing
<ESC> "X" CHR\$(0)	Cancel proportional download character set
<ESC> "\$" CHR\$(1)	Selects the download character set and uses normal spacing
<ESC> "\$" CHR\$(0)	Cancel normal download character set

Free Manuals Download Website

<http://myh66.com>

<http://usermanuals.us>

<http://www.somanuals.com>

<http://www.4manuals.cc>

<http://www.manual-lib.com>

<http://www.404manual.com>

<http://www.luxmanual.com>

<http://aubethermostatmanual.com>

Golf course search by state

<http://golfingnear.com>

Email search by domain

<http://emailbydomain.com>

Auto manuals search

<http://auto.somanuals.com>

TV manuals search

<http://tv.somanuals.com>