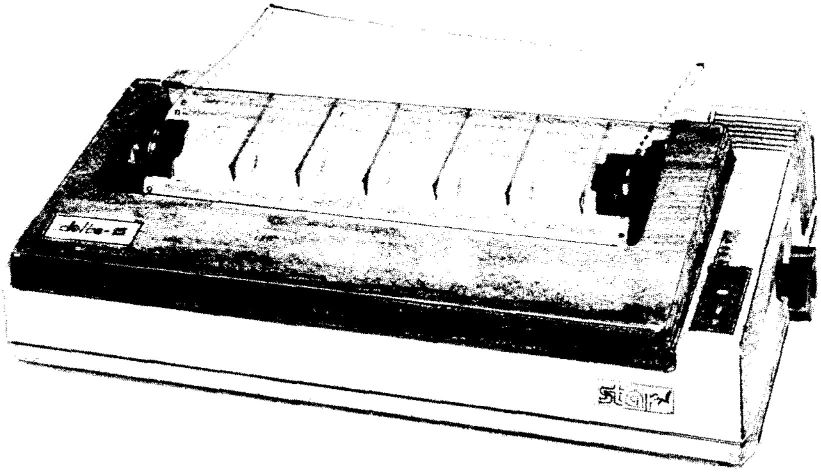
Delta User's Manual



star™
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

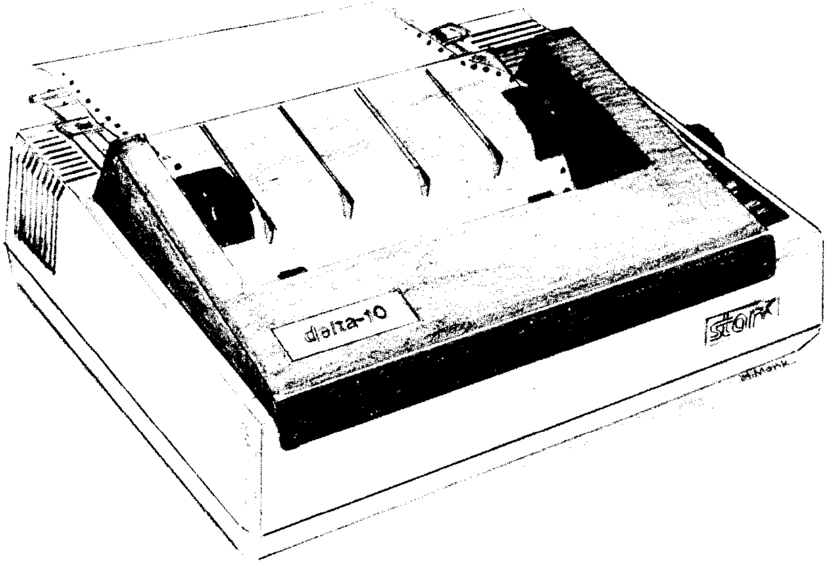
Microsoft BASIC: Microsoft Corporation

Osborne 1: Osborne Computer Corporation

SuperCalc: Sorcim Corporation

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

WordStar: MicroPro International Corporation



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 wi:h 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.)

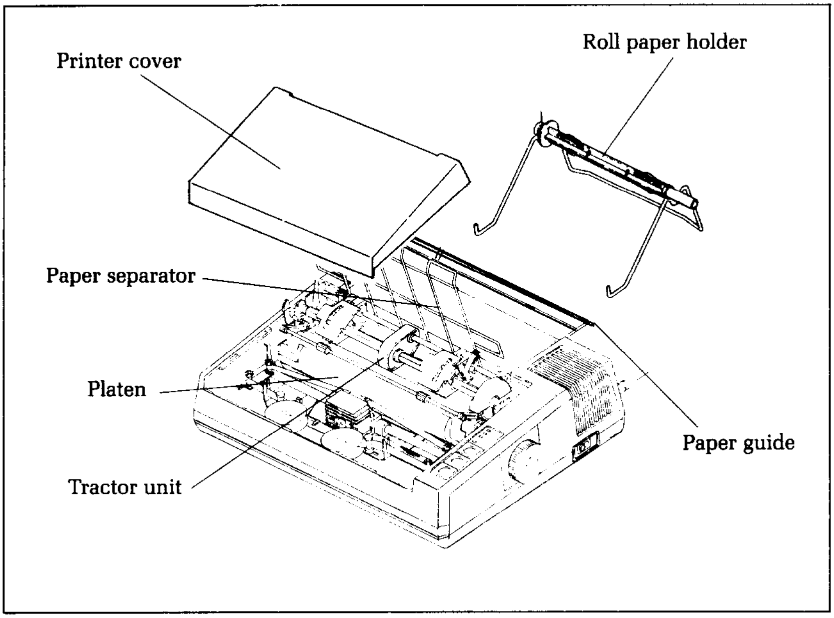


Figure 1-1. *For instructions on attaching the various components, see Appendix A.*

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.

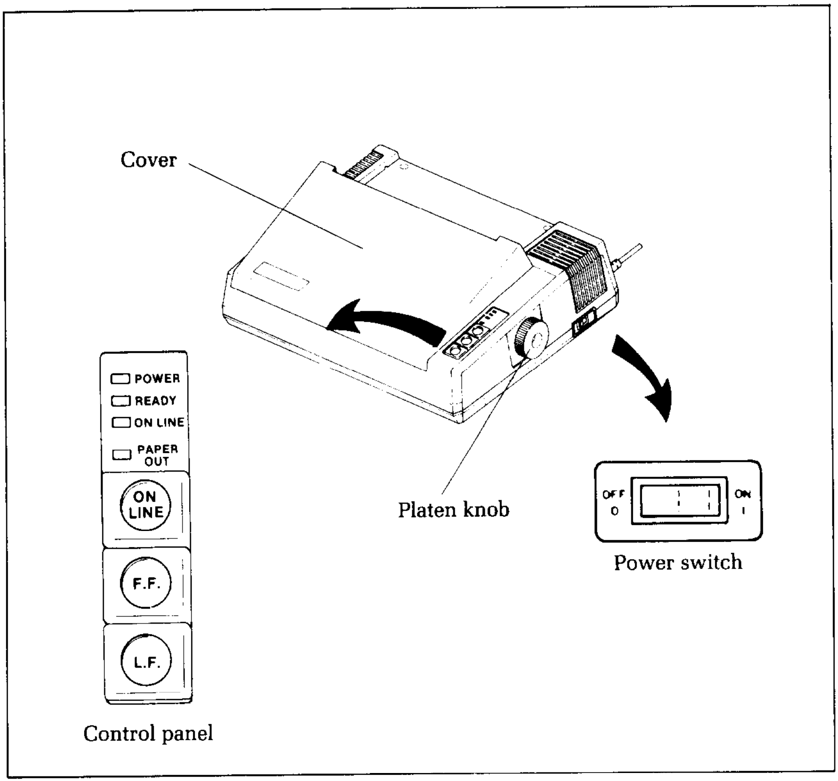


Figure 1-2. *All of Delta's controls are on the right side.*

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:

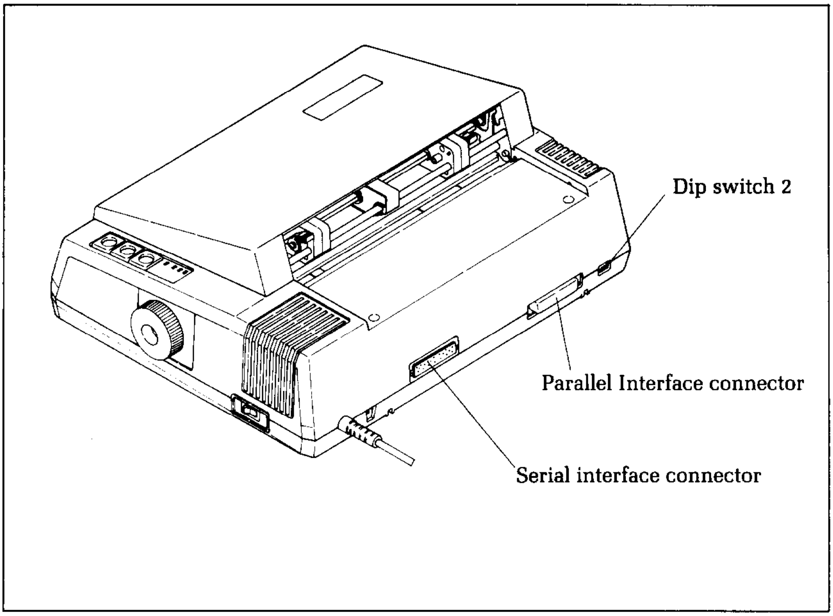


Figure 1-3. *Around on the back of Delta you'll find the interface connectors.*

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 fanfolded 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".

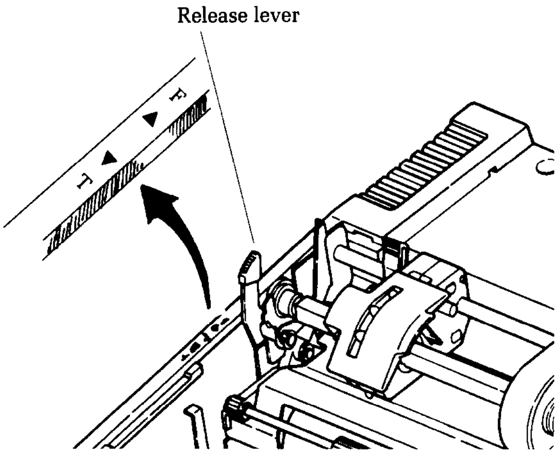


Figure 1-4. *The paper release lever has two settings: "F" for friction feed and "T" for tractor feed.*

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.

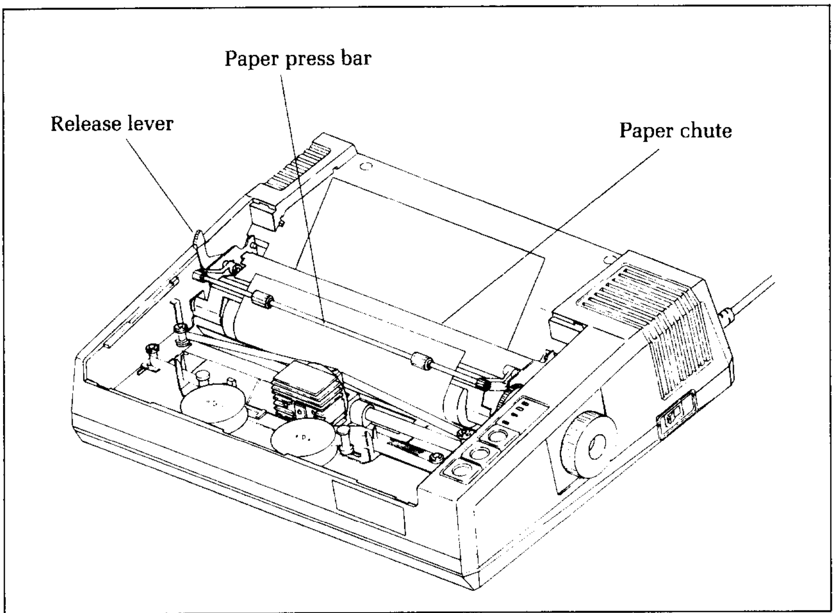


Figure 1-5. *Inserting a single sheet of paper can be done "under power" with the line feed button.*

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.

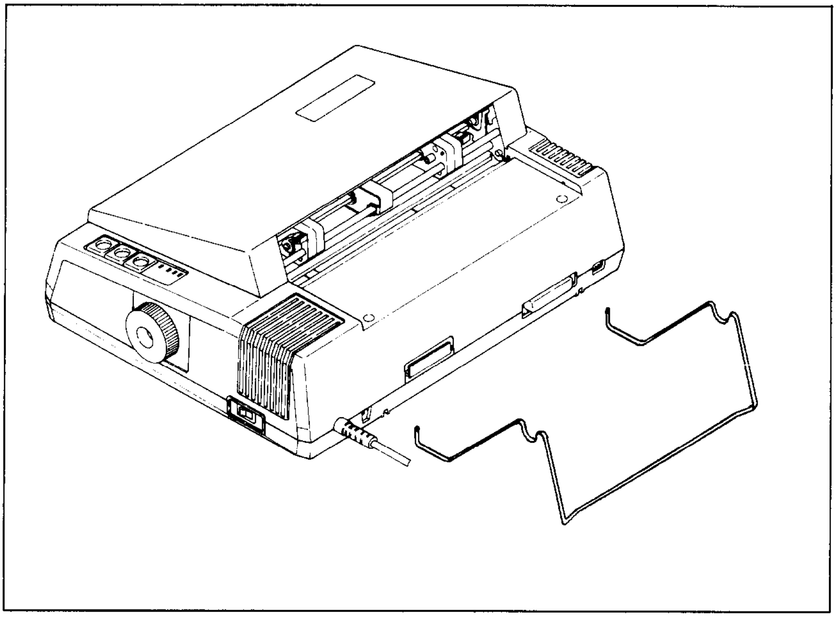


Figure 1-6. *The roll paper holder is attached to the back of Delta.*

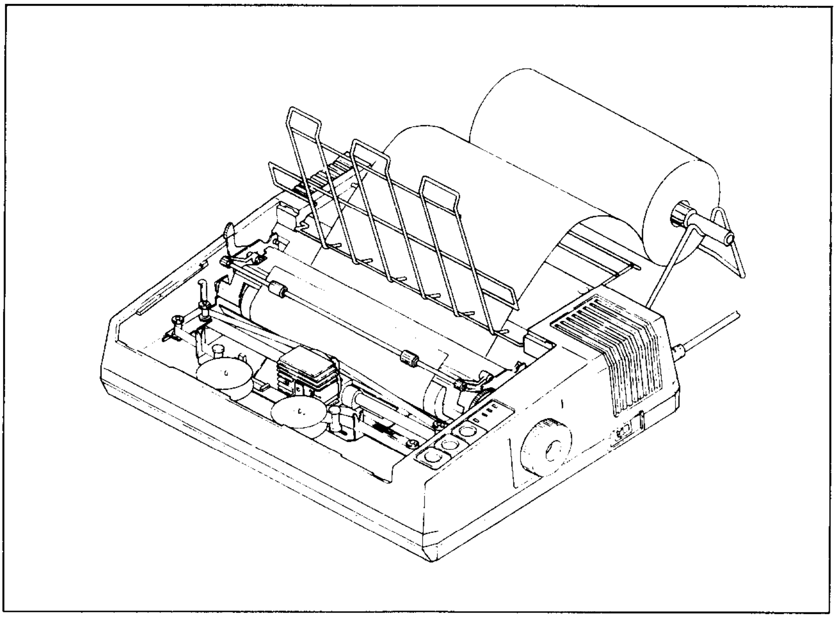


Figure 1-7. *Inserting roll paper into Delta is similar to loading single sheets.*

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.

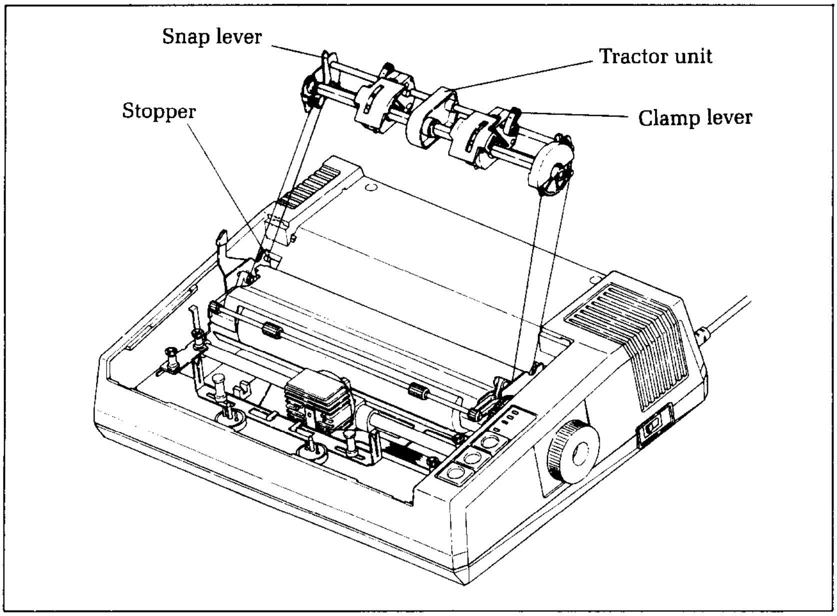


Figure 1-8. *Replace the tractor unit by placing the hooks against the stoppers and lower the front into place while holding the snap levers.*

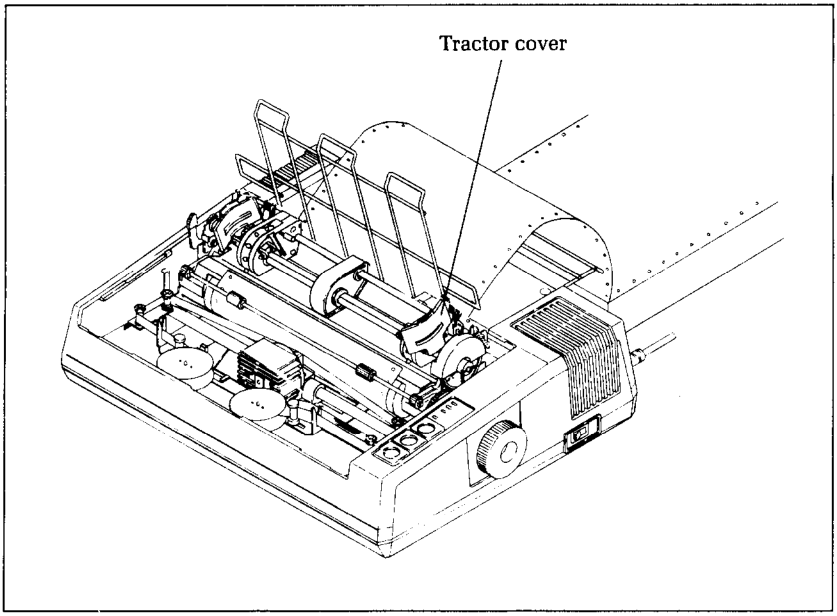


Figure 1-9. *Open the tractor covers to expose the sprocket teeth.*

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!

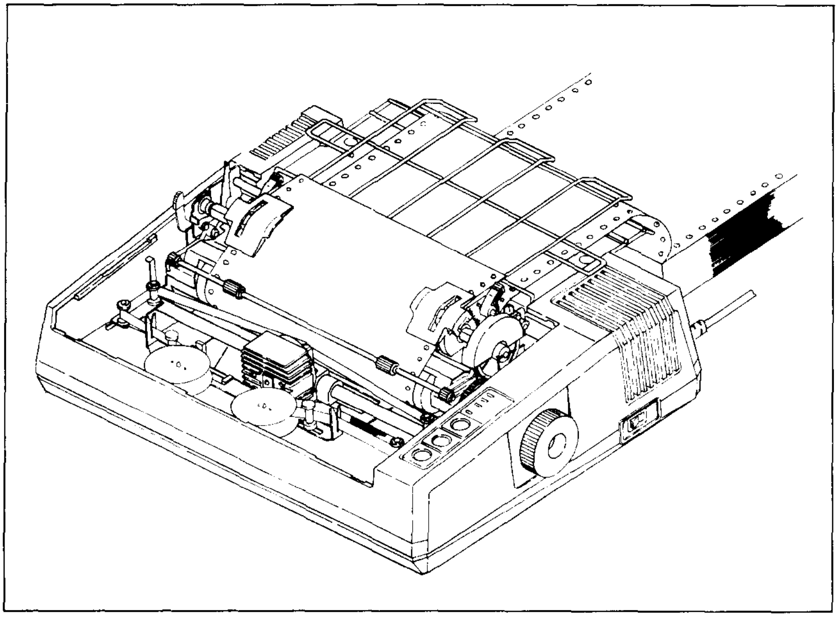


Figure 1-10. *Delta ready to run with sprocket-feed paper.*

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.

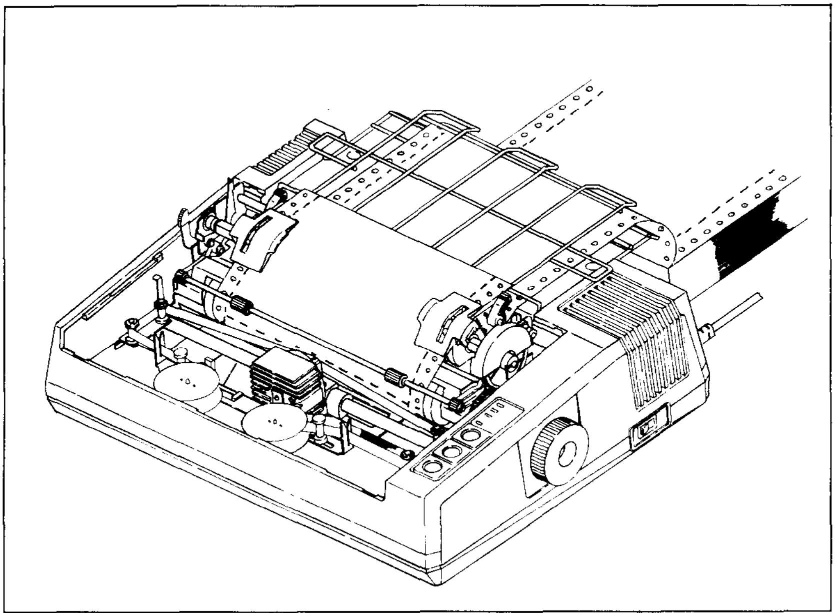


Figure 1-11. *The perforation should be lined up with the top of the ribbon guide.*

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!

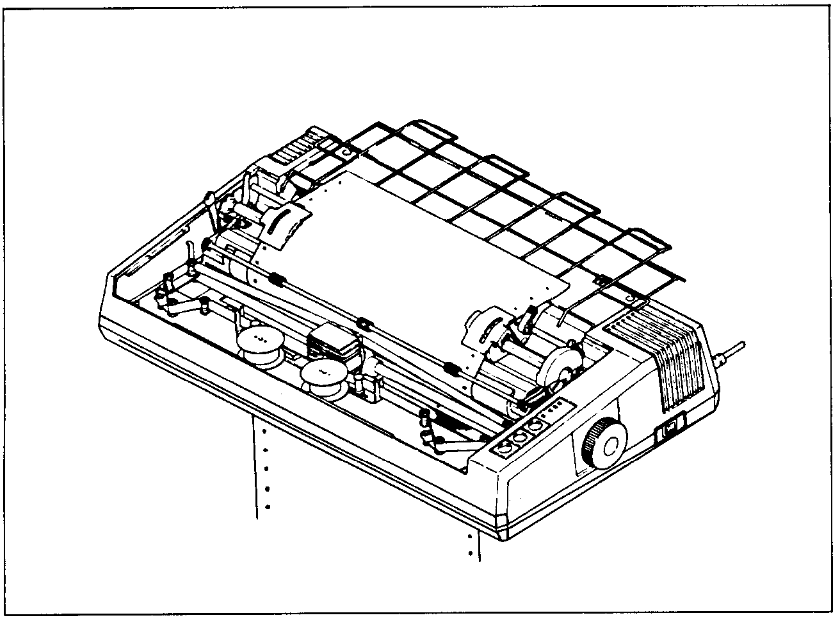


Figure 1-12. *Delta-15 can be loaded from the back like Delta-10 or from the bottom, as shown here.*

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.

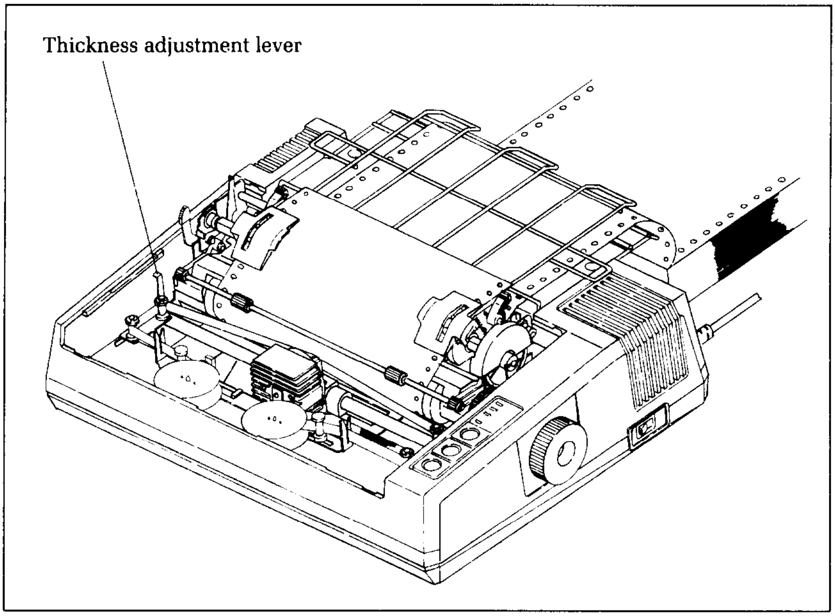


Figure 1-13. *Adjusting the print head gap lever allows you to obtain optimum print quality on paper ranging from .07 mm to .28 mm thick—even 3-part carbonless sets.*

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.

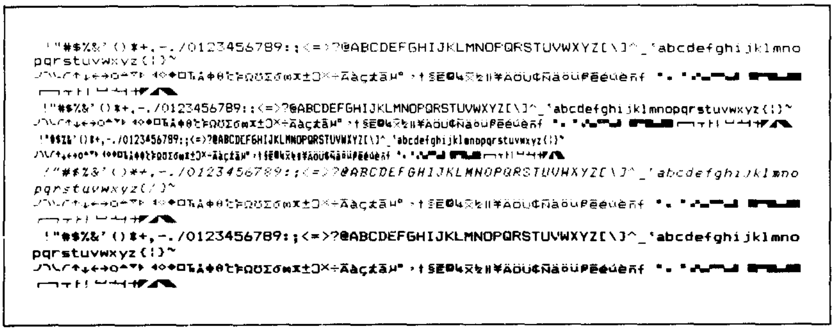
Self-Test

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.

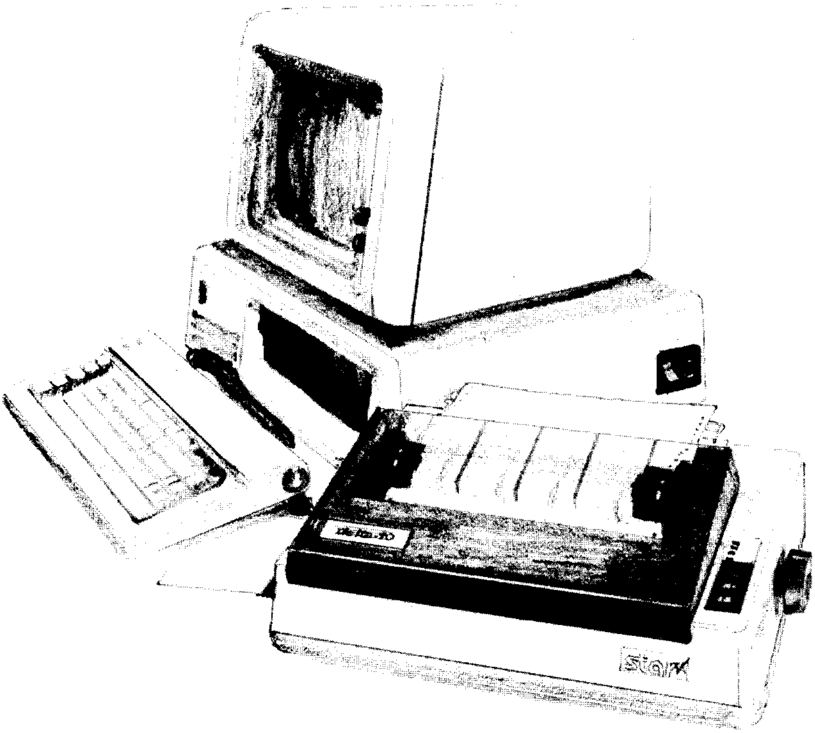


```
 !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmno
pqrstuvwxyz{|}~
↑↓←→○▲▼▶ ◀◇◆□ ÅΦθ ΩÜΣσ∞±⊃×÷Ãàç ã ° †§Ē ¥ÄÖÜ¢Ñäöü éèùèñƒ
 !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~
↑↓←→○▲▼▶ ◀◇◆□ ÅΦθ ΩÜΣσ∞±⊃×÷Ãàç ã ° †§Ē ¥ÄÖÜ¢Ñäöü éèùèñƒ
 !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~
↑↓←→○▲▼▶ ◀◇◆□ ÅΦθ ΩÜΣσ∞±⊃×÷Ãàç ã ° †§Ē ¥ÄÖÜ¢Ñäöü éèùèñƒ
 !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmno
pqrstuvwxyz{|}~
↑↓←→○▲▼▶ ◀◇◆□ ÅΦθ ΩÜΣσ∞±⊃×÷Ãàç ã ° †§Ē ¥ÄÖÜ¢Ñäöü éèùèñƒ
 !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmno
pqrstuvwxyz{|}~
↑↓←→○▲▼▶ ◀◇◆□ ÅΦθ ΩÜΣσ∞±⊃×÷Ãàç ã ° †§Ē ¥ÄÖÜ¢Ñäöü éèùèñƒ
```

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!

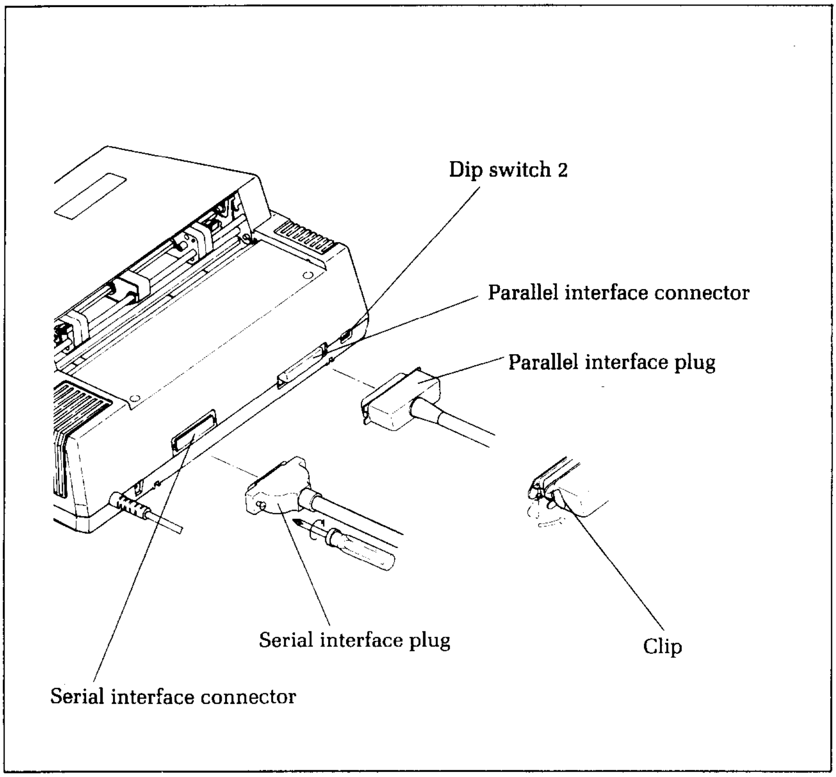


Figure 2-1. *Delta has both serial and parallel interfaces.*

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#Ø
```

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
10 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
10 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
10 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
1Ø LPRINT CHR$(27) CHR$(52);
2Ø 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:

```
1Ø 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)
1Ø LPRINT ESC$ "4";
```

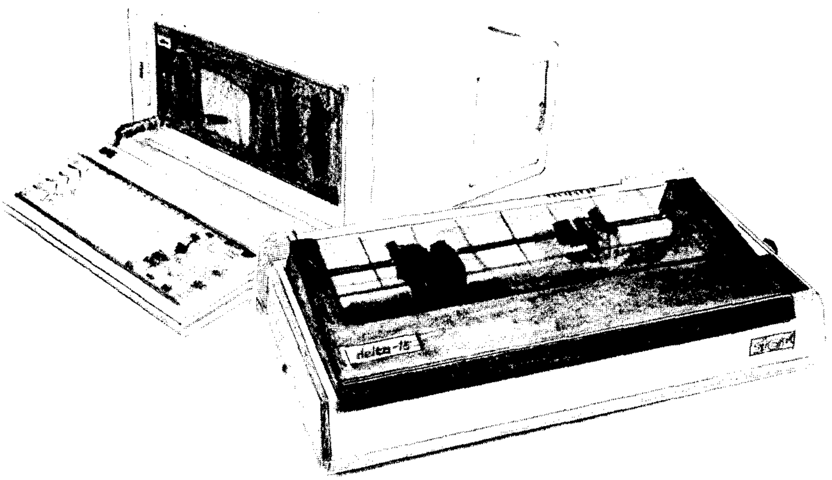
Turn your printer off and back on now, or you will be printing in italic for quite a while!

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.

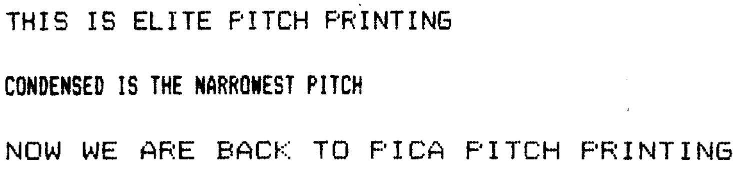
Table 3-1
Print pitch commands

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
1Ø LPRINT CHR$(27) "B" CHR$(2)
2Ø LPRINT "THIS IS ELITE PITCH PRINTING"
3Ø LPRINT CHR$(27) "B" CHR$(3)
4Ø LPRINT "CONDENSED IS THE NARROWEST PITCH"
5Ø LPRINT CHR$(27) "B" CHR$(1)
6Ø 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
1Ø LPRINT CHR$(14) "THIS LINE IS EXPANDED"
2Ø 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:

```
1Ø LPRINT CHR$(27) "W" CHR$(1) "THIS LINE IS
   EXPANDED"
2Ø LPRINT "AND SO IS THIS ONE"
3Ø LPRINT CHR$(27) "W" CHR$(Ø) "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 ⟨ESC⟩ "W" CHR$(1) it stays on until you turn it off with ⟨ESC⟩ "W" CHR$(0). That's what we added line 30 for.

Table 3-2
Expanded print commands

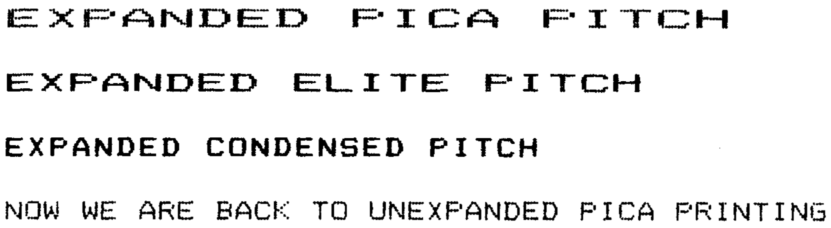
Function	Control code
One line expanded	CHR$(14)
Expanded ON	⟨ESC⟩ "W" CHR$(1)
Expanded OFF	⟨ESC⟩ "W" CHR$(0)

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:

```
1Ø LPRINT CHR$(14) "EXPANDED PICA PITCH"
2Ø LPRINT CHR$(27) "B" CHR$(2)
3Ø LPRINT CHR$(14) "EXPANDED ELITE PITCH"
4Ø LPRINT CHR$(27) "B" CHR$(3)
5Ø LPRINT CHR$(14) "EXPANDED CONDENSED PITCH"
6Ø LPRINT CHR$(27) "B" CHR$(1)
7Ø 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.

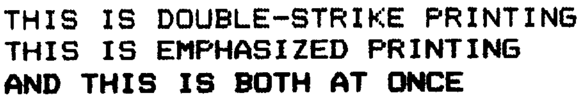
Table 3-3
Print emphasis commands

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
1Ø LPRINT CHR$(27) "G"
2Ø LPRINT "THIS IS DOUBLE-STRIKE PRINTING"
3Ø LPRINT CHR$(27) "H" CHR$(27) "E";
4Ø LPRINT "THIS IS EMPHASIZED PRINTING"
5Ø LPRINT CHR$(27) "G";
6Ø LPRINT "AND THIS IS BOTH AT ONCE"
7Ø 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.

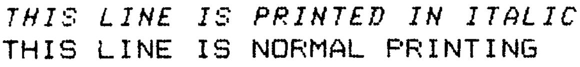
Table 3-4
Italic commands

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 ⟨ESC⟩ “4”, and line 30 turns it off with ⟨ESC⟩ “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.

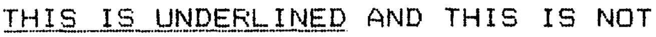
Table 3-5
Underline commands

Function	**Control code**
Underline ON	⟨ESC⟩ “ - ” CHR$(1)
Underline OFF	⟨ESC⟩ “ - ” CHR$(0)

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

```
NEW
1Ø LPRINT CHR$(27) "-" CHR$(1);
2Ø LPRINT "THIS IS UNDERLINED";
3Ø LPRINT CHR$(27) "-" CHR$(Ø);
4Ø 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 ⟨ESC⟩ “ - ” CHR$(1), and then off in line 30 with ⟨ESC⟩ “ - ” CHR$(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.

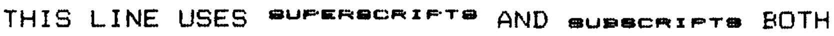
Table 3-6 has the codes for using superscripts and subscripts.

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
1Ø LPRINT "THIS LINE USES";
2Ø LPRINT CHR$(27) "S" CHR$(Ø);
3Ø LPRINT " SUPERSCRIPTS";
4Ø LPRINT CHR$(27) "T";
5Ø LPRINT " AND";
6Ø LPRINT CHR$(27) "S" CHR$(1);
7Ø LPRINT " SUBSCRIPTS";
8Ø LPRINT CHR$(27) "T";
9Ø 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 modes for even more printing effects. Condensed, italic, double-strike, underlined subscripts are something that you are probably just itching to print!

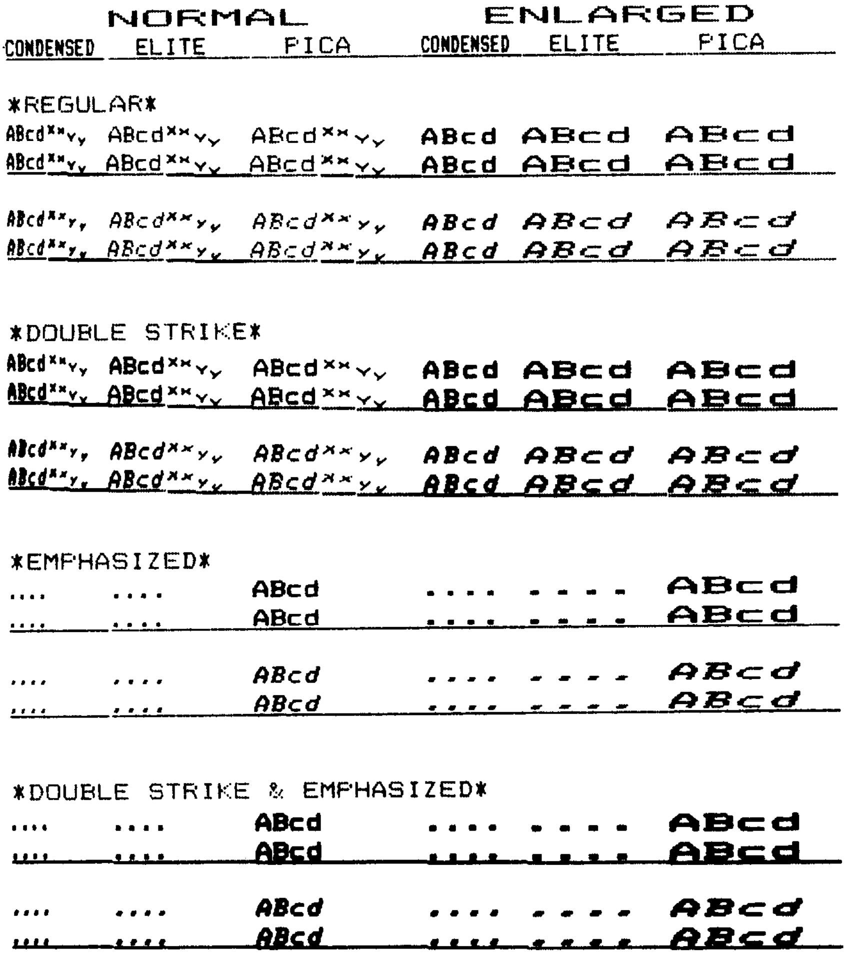
There are 288 "theoretical" combinations of the modes that we have learned. Of these, a mere 112 will work! (Some combinations, like expanded superscripts, just don't work.) Instead of trying to list all the combinations that work, we have a program that prints a chart showing all the combinations. There is a sample of each of the 112 possible combinations on the chart. (The dots just indicate the few combinations that *don't* work.) Enter the following program and run it to make your own chart.

```
1ØØ WIDTH "LPT1:", 255   'CANCEL AUTO CR & LF AFTER
   8Ø CHAR'S
11Ø GOSUB 1ØØØ  'BUILD INSTRUCTION SET REQUIRED
12Ø GOSUB 2ØØØ  'PRINT HEADING
13Ø LPRINT "*REGULAR*"
14Ø GOSUB 3ØØØ  'PRINT FOUR LINES REGULAR
15Ø LPRINT  "*DOUBLE STRIKE*"
16Ø LPRINT DOUBLE.STRIKE$;
17Ø GOSUB 3ØØØ  'PRINT FOUR LINES DOUBLE STRIKE
18Ø LPRINT "*EMPHASIZED*"
19Ø EMPHASIZED = TRUE
2ØØ GOSUB 3ØØØ  'PRINT FOUR LINES EMPHASIZED
21Ø LPRINT "*DOUBLE STRIKE & EMPHASIZED*"
22Ø LPRINT DOUBLE.STRIKE$ EMPHASIZED$;
23Ø GOSUB 3ØØØ   'PRINT DOUBLE STRIKE & EMPHASIZED
24Ø END
25Ø '******************************************
26Ø '******************************************
1ØØØ '''''''''''''''''''''''''''''''''''''''''''
1Ø1Ø '     '
1Ø2Ø '        BUILD INSTRUCTION SET    '
1Ø3Ø '     '
1Ø4Ø '''''''''''''''''''''''''''''''''''''''''''
1Ø5Ø ''''''FONT (CHARACTER) STYLES''''''''''''''
1Ø6Ø ITALIC$ = CHR$(27) + CHR$(52)
1Ø7Ø ROMAN$  = CHR$(27) + CHR$(53)
1Ø8Ø ''''''''''''PITCH MODES''''''''''''''''''''
1Ø9Ø     ENLARGED$  = CHR$(27) + CHR$(87) + CHR$(1)
11ØØ NOT.ENLARGED$  = CHR$(27) + CHR$(87) + CHR$(Ø)
111Ø     PICA$      = CHR$(27) + CHR$(66) + CHR$(1)
112Ø     ELITE$     = CHR$(27) + CHR$(66) + CHR$(2)
113Ø     CONDENSED$ = CHR$(27) + CHR$(66) + CHR$(3)
```

```
114Ø ''''''''SPECIAL PRINT MODES''''''''''''''''
115Ø     EMPHASIZED$     = CHR$(27) + CHR$(69)
116Ø NOT.EMPHASIZED$     = CHR$(27) + CHR$(7Ø)
117Ø     DOUBLE.STRIKE$ = CHR$(27) + CHR$(71)
118Ø NOT.DOUBLE.STRIKE$ = CHR$(27) + CHR$(72)
119Ø     UNDERLINED$     = CHR$(27) + CHR$(45) +
   CHR$(1)
12ØØ NOT.UNDERLINED$     = CHR$(27) + CHR$(45) +
   CHR$(Ø)
121Ø     SUPERSCRIPT$    = CHR$(27) + CHR$(83) +
   CHR$(Ø)
122Ø     SUBSCRIPT$      = CHR$(27) + CHR$(83) +
   CHR$(1)
123Ø NOT.SCRIPTED$       = CHR$(27) + CHR$(84)
124Ø RESET.ALL$ = NOT.EMPHASIZED$ + NOT.UNDERLINED$
   + NOT.DOUBLE.STRIKE$
125Ø RESET.ALL$ = RESET.ALL$ + ROMAN$ + PICA$ +
   NOT.ENLARGED$
126Ø ''''''''''''''''CONSTANTS''''''''''''''''''
127Ø TRUE = 1 :FALSE = Ø
128Ø REGULAR.HEADING$ = STRING$(27,"*") + "REGULAR"
   + STRING$(27,"*")
129Ø RETURN
2ØØØ ''''''''''''''''''''''''''''''''''''''''''''
2Ø1Ø '      '
2Ø2Ø '       PRINT HEADING    '
2Ø3Ø '      '
2Ø4Ø ''''''''''''''''''''''''''''''''''''''''''''
2Ø5Ø LPRINT RESET.ALL$
2Ø6Ø LPRINT ENLARGED$ "   NORMAL     ENLARGED  "
2Ø7Ø LPRINT RESET.ALL$;
2Ø8Ø LPRINT UNDERLINED$;
2Ø9Ø LPRINT CONDENSED$ "CONDENSED ";
21ØØ LPRINT ELITE$     "  ELITE   ";
211Ø LPRINT PICA$      "  PICA    ";
212Ø LPRINT CONDENSED$ "CONDENSED ";
213Ø LPRINT ELITE$     "  ELITE   ";
214Ø LPRINT PICA$      "  PICA    "
215Ø LPRINT RESET.ALL$
216Ø RETURN
3ØØØ ''''''''''''''''''''''''''''''''''''''''''''
3Ø1Ø '     '
3Ø2Ø '       PRINT FOUR LINES         '
3Ø3Ø '     '
3Ø4Ø ''''''''''''''''''''''''''''''''''''''''''''
```

```
3050 ITALICS    = FALSE       :LPRINT ROMAN$;
3060 UNDERLINED = FALSE       :LPRINT
   NOT.UNDERLINED$;
3070 ENLARGED   = FALSE       :LPRINT NOT.ENLARGED$;
3080 PICA       = FALSE
3090 '''''''''''''PRINT PITCHES'''''''''''''''''
3100 LPRINT CONDENSED$;
3110 GOSUB 3500 'PRINT AS REQUIRED
3120 LPRINT ELITE$;
3130 GOSUB 3500 'PRINT AS REQUIRED
3140 LPRINT PICA$; :PICA = TRUE
3150 GOSUB 3500 'PRINT AS REQUIRED
3160 '''''''''SEE WHAT HAS BEEN DONE''''''''''''
3170 IF ENLARGED = TRUE THEN LPRINT :GOTO 3190
   'CHECK IF UNDERLINED
3180 LPRINT ENLARGED$; :ENLARGED = TRUE :GOTO 3080
   'ELSE CONTINUE LINE
3190 IF UNDERLINED = TRUE THEN LPRINT :GOTO 3210
   'CHECK IF ITALICS
3200 LPRINT UNDERLINED$;  :UNDERLINED = TRUE :GOTO
   3070   'ELSE PRINT UNDERLINE
3210 IF ITALICS = TRUE THEN LPRINT RESET.ALL$
   :RETURN
3220 LPRINT ITALIC$;  :ITALICS = TRUE  :GOTO 3060
   'REPEAT WITH ITALICS
3500 ''''''''''''''''''''''''''''''''''''''''''''
3510 '      '
3520 '         PRINT AS REQUIRED        '
3530 '      '
3540 ''''''''''''''''''''''''''''''''''''''''''''
3550 BLANK$ = STRING$(6,32)   :FOUR.DOT$ = "...."
3560 IF EMPHASIZED = FALSE THEN LPRINT "ABcd";
   :GOTO 3610
3570 IF PICA = FALSE THEN LPRINT FOUR.DOT$;
   :GOTO 3590
3580 LPRINT EMPHASIZED$ "ABcd" ;
3590 IF ENLARGED = TRUE THEN LPRINT " ";
   :ELSE LPRINT BLANK$;
3600 RETURN
3610 '''''''''''EMPHASIZED = FALSE''''''''''''''
3620 IF ENLARGED = TRUE THEN LPRINT " ";
   :RETURN
3630 LPRINT SUPERSCRIPT$; "Xx";
3640 LPRINT SUBSCRIPT$;   "Yy  ";
3650 LPRINT NOT.SCRIPTED$;
3660 RETURN
```

Here is the chart it produces:

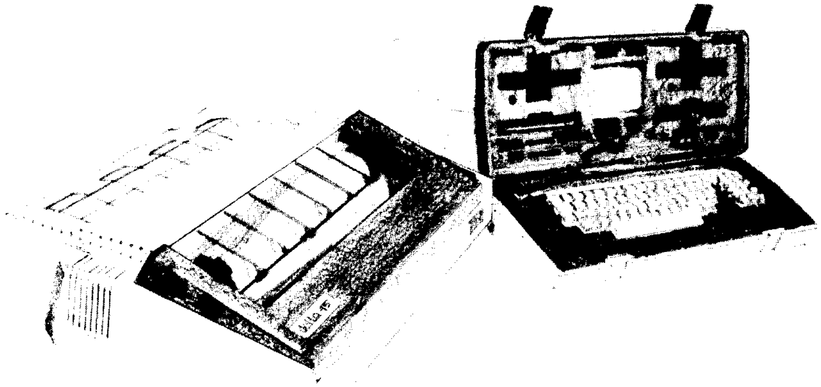


```
             NORMAL                              ENLARGED
CONDENSED   ELITE        PICA          CONDENSED   ELITE       PICA

*REGULAR*
ABcdxxyy    ABcdxxyy     ABcdxxyy      ABcd        ABcd        ABcd
ABcdxxyy    ABcdxxyy     ABcdxxyy      ABcd        ABcd        ABcd

ABcdxxyy    ABcdxxyy     ABcdxxyy      ABcd        ABcd        ABcd
ABcdxxyy    ABcdxxyy     ABcdxxyy      ABcd        ABcd        ABcd

*DOUBLE STRIKE*
ABcdxxyy    ABcdxxyy     ABcdxxyy      ABcd        ABcd        ABcd
ABcdxxyy    ABcdxxyy     ABcdxxyy      ABcd        ABcd        ABcd

ABcdxxyy    ABcdxxyy     ABcdxxyy      ABcd        ABcd        ABcd
ABcdxxyy    ABcdxxyy     ABcdxxyy      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	**Function**
〈ESC〉 "B" CHR$(1)	Sets pica pitch
〈ESC〉 "B" CHR$(2)	Sets elite pitch
〈ESC〉 "B" CHR$(3)	Sets condensed pitch
CHR$(18)	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.

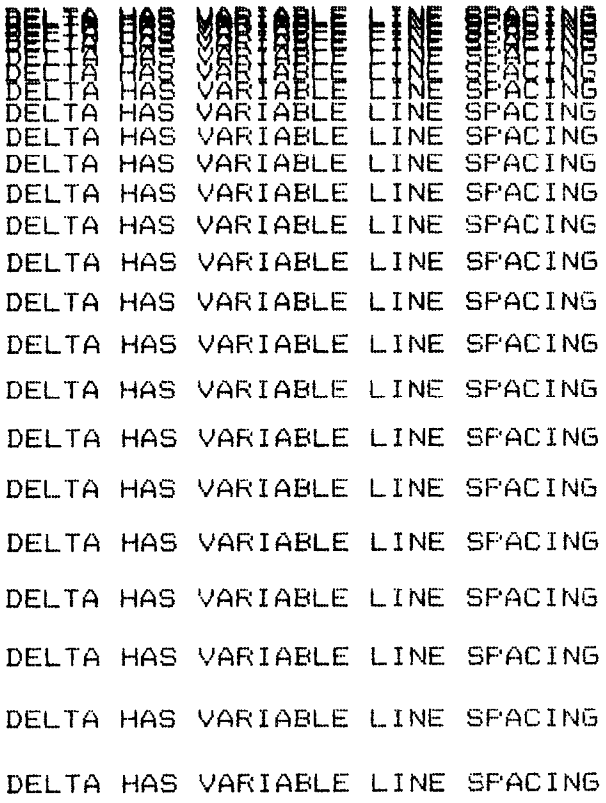
Changing Line Spacing

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"
```

This is what you will get:



```
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
```

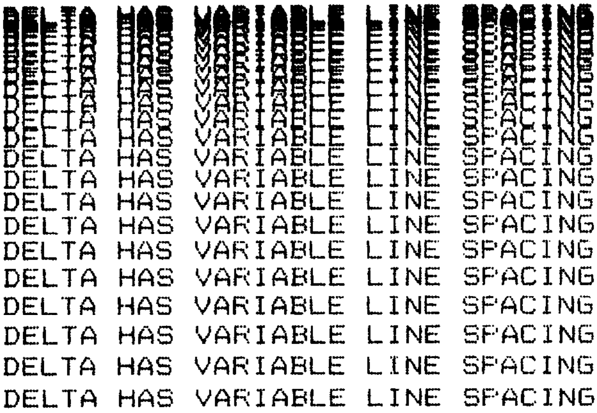
Line 30 changes the line spacing. The command ⟨ESC⟩ "A" CHR$(n) changes the line spacing to *n*/72 of an inch. The loop that is started in line 10 increases the value of *n* (the variable I in the program) each time it is executed. So the line spacing increases as the program continues. Line 20 just shortcuts the loop when I = 13, since BASIC won't let us send CHR$(13) without adding an unwanted CHR$(10) to it. Finally, the ⟨ESC⟩ "2" in line 60 resets the line spacing to 6 lines per inch. This is a shortcut that is the same as ⟨ESC⟩ "A" CHR$(12).

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

The ⟨ESC⟩ "3" CHR$(n) command sets the line spacing in increments of 1/144 inch. Change line 30 in your program so it is like this:

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

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



```
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
DELTA HAS VARIABLE LINE SPACING
```

The program works just the same as before, but the line spacings are just half what they were. This is because ⟨ESC⟩ "3" CHR$(n) sets the line spacing to *n*/144 inch.

Here are all the line spacing commands, including several "shortcut" commands for commonly used line spacings.

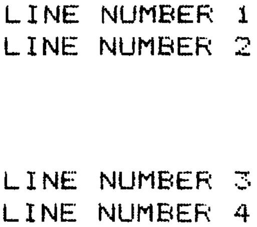
Table 4-1
Line spacing commands

Function	**Control code**
Set line spacing to *n*/72 inch	⟨ESC⟩ "A" CHR$(*n*)
Set line spacing to *n*/144 inch	⟨ESC⟩ "3" CHR$(n)
Set line spacing to 1/8 inch	⟨ESC⟩ "0"
Set line spacing to 7/72 inch	⟨ESC⟩ "1"
Set line spacing to 1/6 inch	⟨ESC⟩ "2"
One-time line feed of *n*/144 inch	⟨ESC⟩ "J" CHR$(*n*)

Let's take a look at the last command in the table, which gives a one-time line feed of *n*/144 inch. The ⟨ESC⟩ "J" CHR$(*n*) command 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
1Ø LPRINT "LINE NUMBER 1"
2Ø LPRINT "LINE NUMBER 2";
3Ø LPRINT CHR$(27) "J" CHR$(1ØØ);
4Ø LPRINT "LINE NUMBER 3"
5Ø LPRINT "LINE NUMBER 4"
```

Here is what Delta will produce:



```
LINE NUMBER 1
LINE NUMBER 2



LINE NUMBER 3
LINE NUMBER 4
```

The ⟨ESC⟩ "J" 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 (⟨ESC⟩ "A", ⟨ESC⟩ "3", and ⟨ESC⟩ "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 ⟨ESC⟩ "a" 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

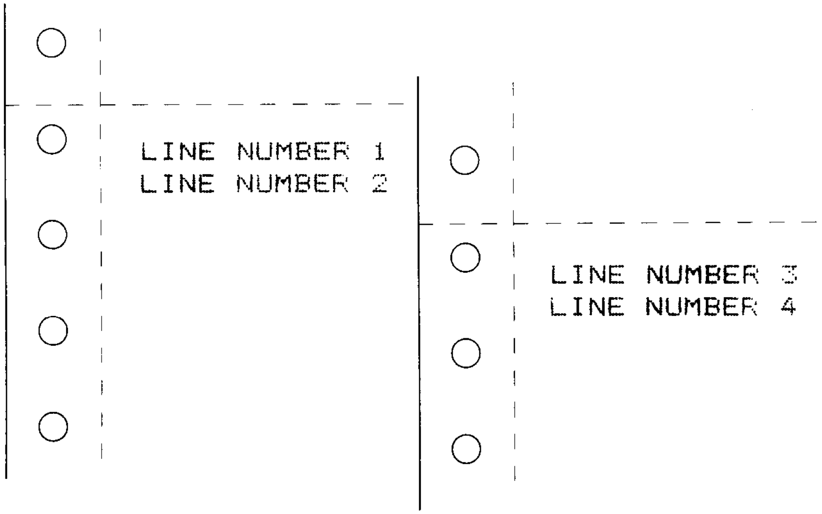
We have seen how to control the spacing between lines on a page. Delta also has commands that control the placement of printing on the page, and even adjust for different size pages.

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:



```
LINE NUMBER 1
LINE NUMBER 2
```

```
LINE NUMBER 3
LINE NUMBER 4
```

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 n lines	⟨ESC⟩ "C" CHR$(n)
Set the page length to *n* inches	⟨ESC⟩ "C" CHR$(0) CHR$(n)

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

```
NEW
1Ø LPRINT CHR$(27) "C" CHR$(Ø) CHR$(7);
2Ø LPRINT "PAY TO THE ORDER OF:"
3Ø LPRINT CHR$(12);
4Ø 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
3Ø FOR I = 1 TO 15Ø
4Ø LPRINT "THIS IS LINE "; I
5Ø NEXT
7Ø LPRINT CHR$(12);
```

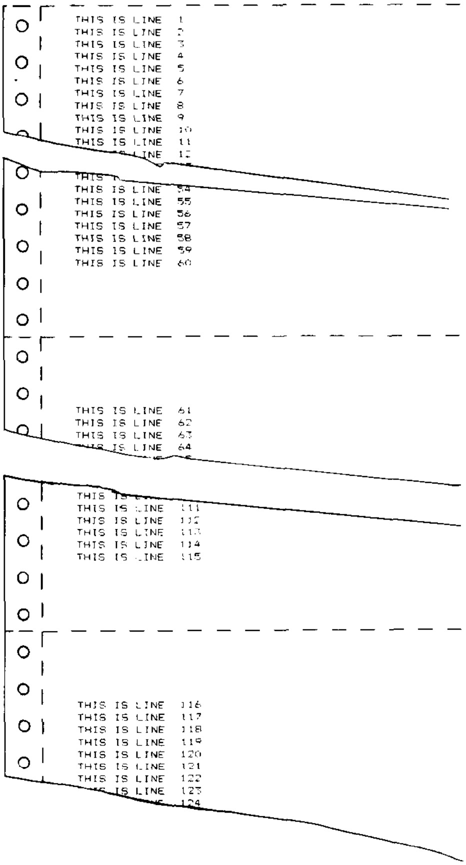
When you run this program it will print 150 lines right down the page and across the perforations. When it's done line 70 sends a form feed to advance the paper to the top of the next page. Look at the lines that have printed near the perforations. Separate the sheets and see if any of the lines have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

```
1Ø LPRINT CHR$(27) "N" CHR$(6);
2Ø LPRINT CHR$(27) "R" CHR$(6);
6Ø 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.

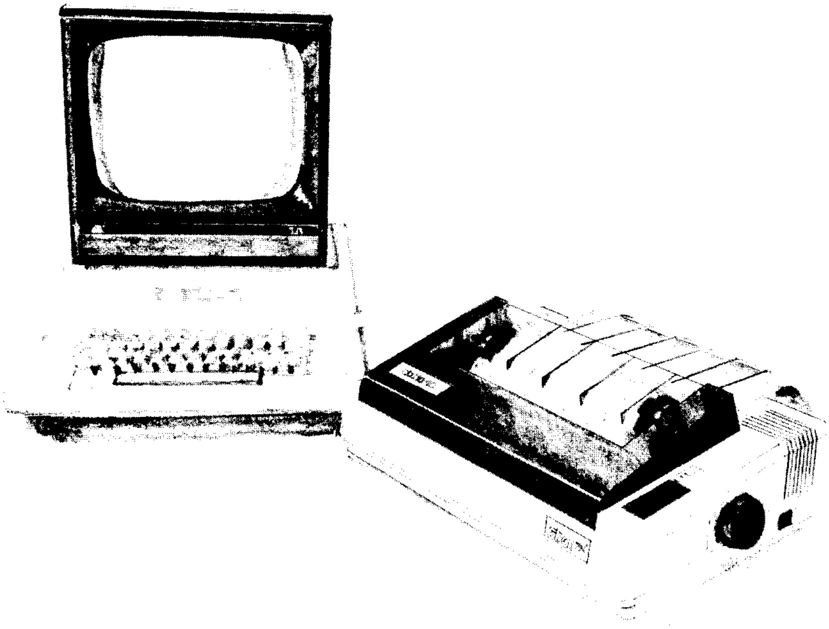


```
THIS IS LINE  1
THIS IS LINE  2
THIS IS LINE  3
THIS IS LINE  4
THIS IS LINE  5
THIS IS LINE  6
THIS IS LINE  7
THIS IS LINE  8
THIS IS LINE  9
THIS IS LINE  10
THIS IS LINE  11
        LINE  12
THIS
THIS IS LINE  54
THIS IS LINE  55
THIS IS LINE  56
THIS IS LINE  57
THIS IS LINE  58
THIS IS LINE  59
THIS IS LINE  60

THIS IS LINE  61
THIS IS LINE  62
THIS IS LINE  63
     IS LINE  64
THIS
THIS IS LINE  111
THIS IS LINE  112
THIS IS LINE  113
THIS IS LINE  114
THIS IS LINE  115

THIS IS LINE  116
THIS IS LINE  117
THIS IS LINE  118
THIS IS LINE  119
THIS IS LINE  120
THIS IS LINE  121
THIS IS LINE  122
     IS LINE  123
              124
```

Summary

Control code	**Function**
CHR$(10)	Line feed
CHR$(13)	Carriage return
⟨ESC⟩ "A" CHR$(n)	Set line spacing to n/72 inch
⟨ESC⟩ "3" CHR$(n)	Set line spacing to n/144 inch
⟨ESC⟩ "0"	Set line spacing to 1/8 inch
⟨ESC⟩ "1"	Set line spacing to 7/72 inch
⟨ESC⟩ "2"	Set line spacing to 1/6 inch
⟨ESC⟩ "J" CHR$(n)	One-time line feed of n/144 inch
⟨ESC⟩ "a" CHR$(n)	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 *pseudotab*. 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*)

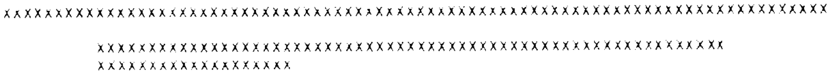
Try setting Delta's margins with this program:

```
NEW
1Ø GOSUB 1ØØ
2Ø LPRINT CHR$(27) "M" CHR$(1Ø);
```

```
3Ø LPRINT CHR$(27) "Q" CHR$(7Ø)
4Ø GOSUB 1ØØ
5Ø END
1ØØ FOR I = 1 TO 8Ø
11Ø LPRINT "X";
12Ø NEXT I
13Ø LPRINT
14Ø 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:



```
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

         XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
         XXXXXXXXXXXXXXXXXXX
```

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
2Ø LPRINT CHR$(11) "FIRST TAB"
3Ø LPRINT CHR$(11) "SECOND TAB"
4Ø LPRINT CHR$(11) "THIRD TAB"
5Ø LPRINT CHR$(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:

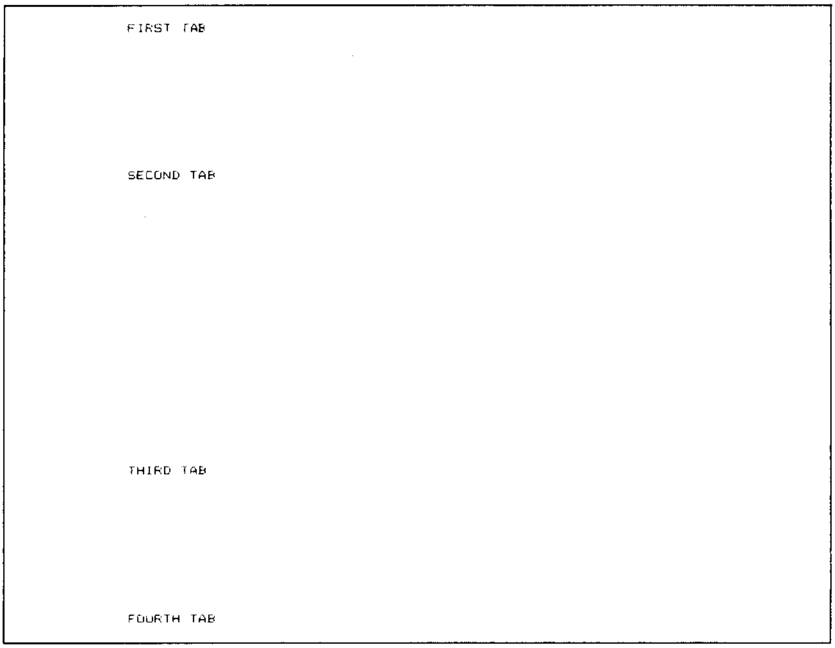
```
1Ø LPRINT CHR$(27) "P" CHR$(1Ø) CHR$(2Ø) CHR$(4Ø)
   CHR$(5Ø) CHR$(Ø);
```

⟨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.

```
6Ø 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?



FIRST TAB

SECOND TAB

THIRD TAB

FOURTH TAB

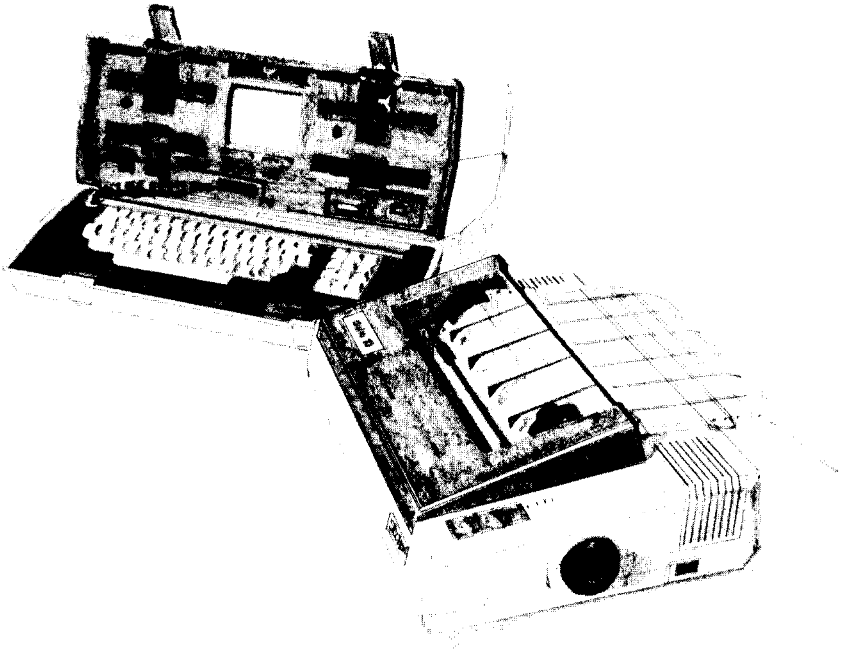
Figure 5-1.

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 ⟨ESC⟩ "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$(9)	Horizontal tab
⟨ESC⟩ "D" *n1 n2 n3* . . . CHR$(0)	Set horizontal tabs
⟨ESC⟩ "b" *n*	One-time horizontal tab of *n* spaces
⟨ESC⟩ "M" *n*	Set left margin
⟨ESC⟩ "N" *n*	Set right margin
CHR$(11)	Vertical tab
⟨ESC⟩ "P" *n1 n2 n3* . . . CHR$(0)	Set vertical tabs
⟨ESC⟩ "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⟩ "@"
Off-line	CHR$(19)
On-line	CHR$(17)
Paper-out detector off	⟨ESC⟩ "8"
Paper-out 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
1Ø LPRINT CHR$(27) "A" CHR$(7);
2Ø FOR I = 1 TO 1Ø
3Ø LPRINT "¦"
4Ø NEXT I
5Ø LPRINT : LPRINT
6Ø LPRINT CHR$(27) "U" CHR$(1);
7Ø FOR I = 1 TO 1Ø
8Ø LPRINT "¦"
9Ø NEXT I
1ØØ 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

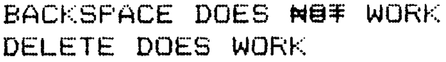
Backspace (CHR$(8)) "backs up" the printhead so that you can print two characters right on top of each other. Each time Delta receives a backspace it moves the printhead one character to the left, instead of to the right. You can *strike over* multiple letters by sending more than one backspace code.

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
1Ø LPRINT "BACKSPACE DOES NOT";
2Ø LPRINT CHR$(8) CHR$(8) CHR$(8);
3Ø LPRINT "=== WORK"
4Ø LPRINT "DELETE DOES NOT";
5Ø LPRINT CHR$(127) CHR$(127) CHR$(127);
6Ø 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.

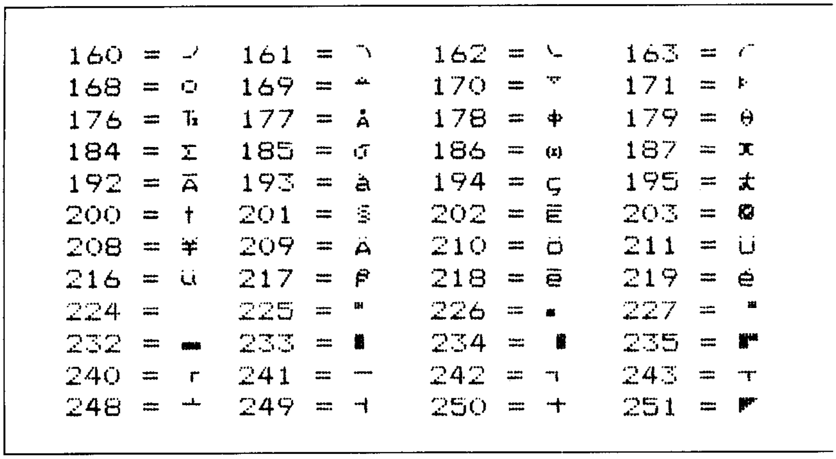
Table 6-4
Eighth bit controls

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 = Å   178 = ⊕   179 = θ
184 = Σ   185 = σ   186 = ⊗   187 = π
192 = Ā   193 = à   194 = ç   195 = ±
200 = †   201 = §   202 = Ē   203 = ©
208 = ¥   209 = Ä   210 = Ö   211 = Ü
216 = ü   217 = ₧   218 = ë   219 = é
224 =     225 = ▘   226 = ▖   227 = ▝
232 = ▄   233 = ▌   234 = ▐   235 = ▛
240 = ┌   241 = ─   242 = ┐   243 = ┬
248 = ┴   249 = ┤   250 = ┼   251 = ▜
```

Figure 6-1.

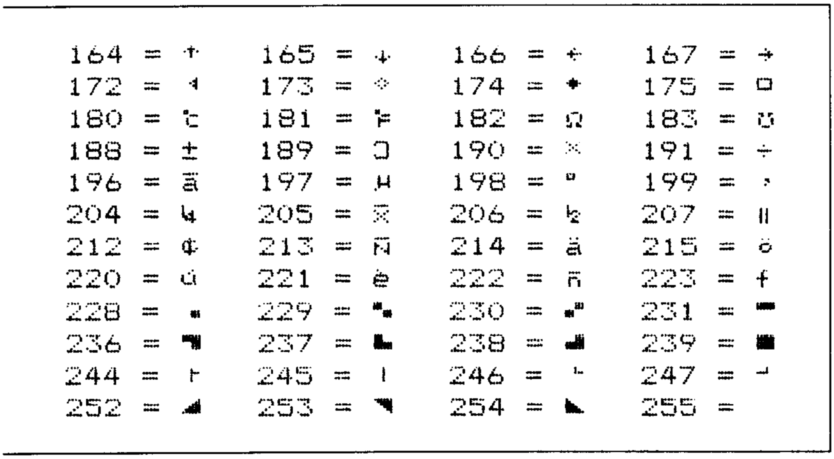
```
30 LPRINT I "= " CHR$(I) CHR$(9);
40 NEXT I : LPRINT : NEXT J
```

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";
```

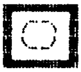


```
164 = ↑     165 = ↓     166 = ←     167 = →
172 = ◂     173 = ◇     174 = ◆     175 = □
180 = ℃     181 = ℉     182 = Ω     183 = ℧
188 = ±     189 = ⊐     190 = ×     191 = ÷
196 = ã     197 = µ     198 = °     199 = ·
204 = ¼     205 = ☒     206 = ½     207 = ‖
212 = ¢     213 = Ñ     214 = ä     215 = ö
220 = ü     221 = ė     222 = ñ     223 = ƒ
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$(27) ">";
7Ø LPRINT CHR$(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	Ø	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	#	@	¡	Ñ	¿	^	‘	¨	ñ	}	~

control code. The term *macro* is from the jargonese *macro-instruction* which refers to an instruction that "calls," or uses a group of normal instructions. In computer programming macro-instructions (which are similar to subroutines) save programmers a lot of time and effort. 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〉 "!"

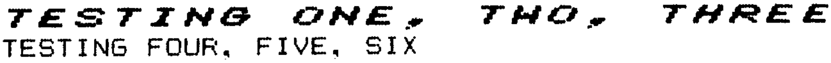
To see how this works we can build a macro that will reset the printing style to normal, no matter what style it may be to start with. The following program will define a macro to do this.

```
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"
```

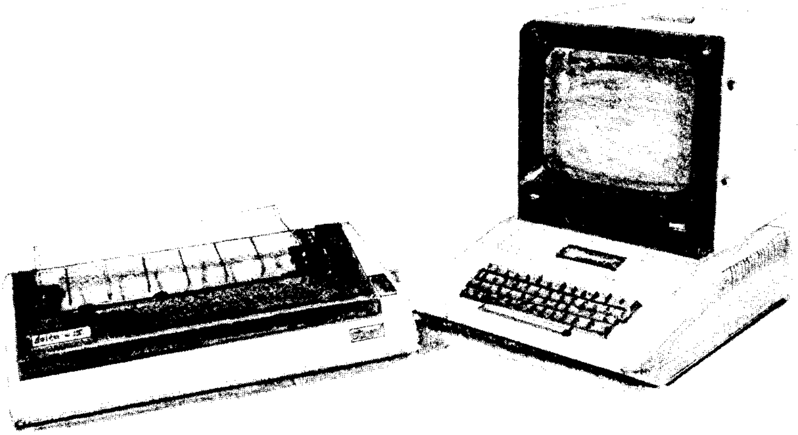


```
TESTING ONE, TWO, THREE
TESTING FOUR, FIVE, SIX
```

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.

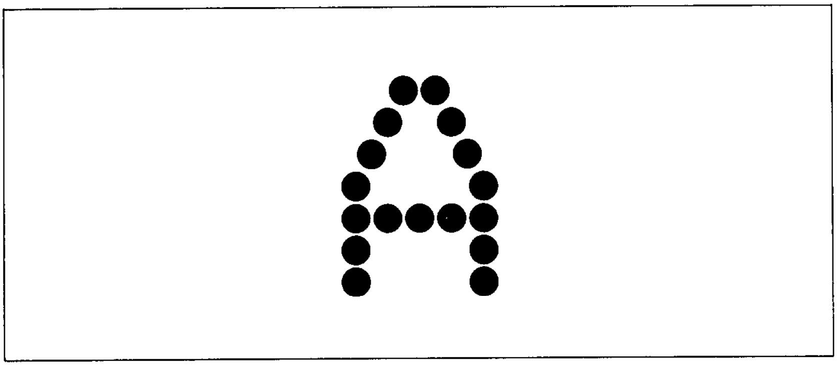


Figure 7-1. *The letter "A" is created by printing 17 dots.*

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

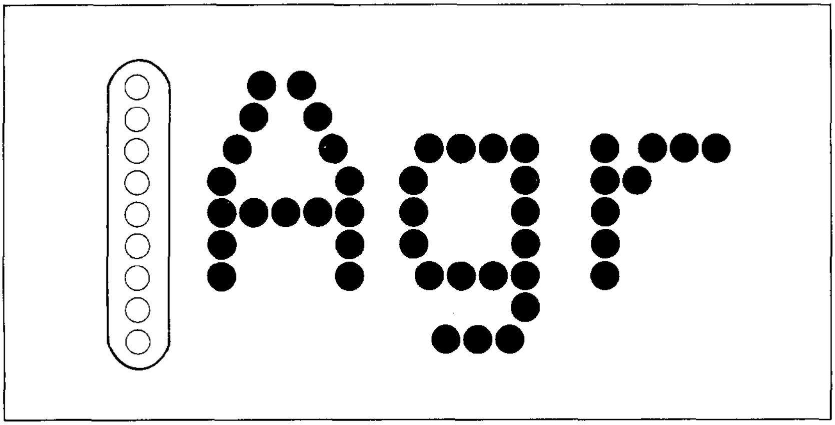


Figure 7-2. *As the printhead moves across the page, each of the wires prints one row of dots.*

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.

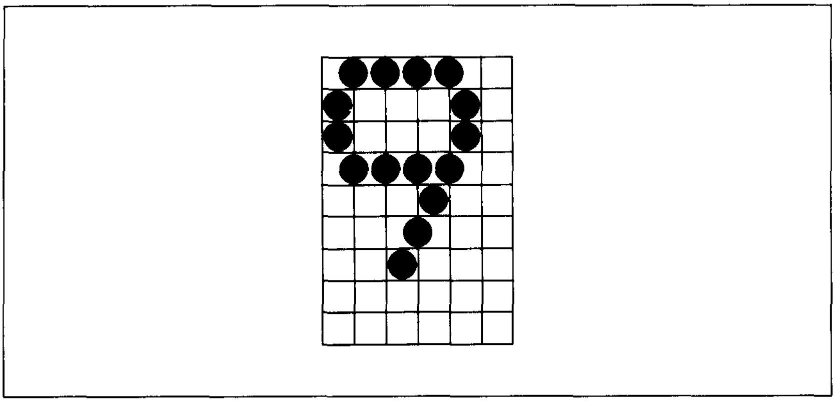


Figure 7-3. *Dots can be inside boxes or straddle the vertical lines of the grid.*

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.)

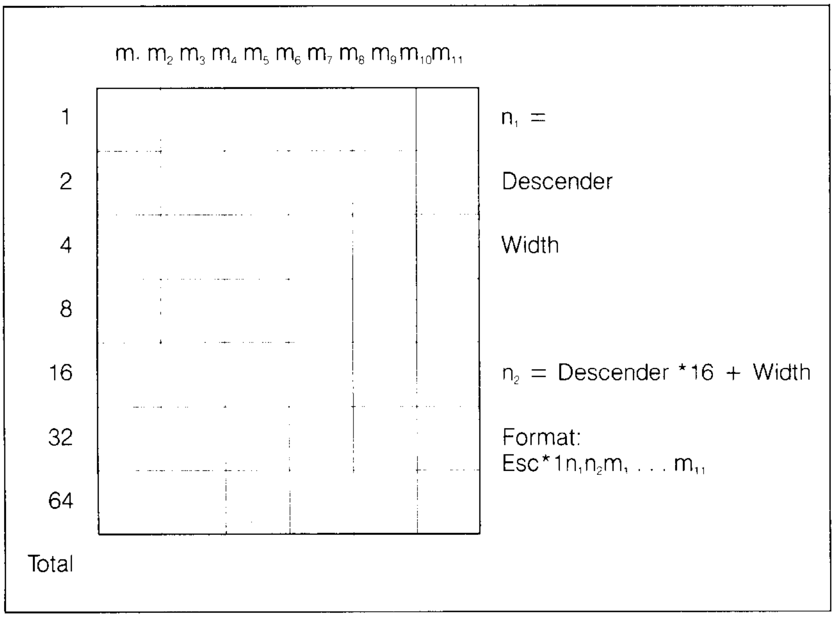

$m. m_2 m_3 m_4 m_5 m_6 m_7 m_8 m_9 m_{10} m_{11}$
1
2
4
8
16
32
64
Total
n_1 =
Descender
Width
n_2 = Descender *16 + Width
Format:
Esc*1$n_1 n_2 m_1$ . . . m_{11}

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.

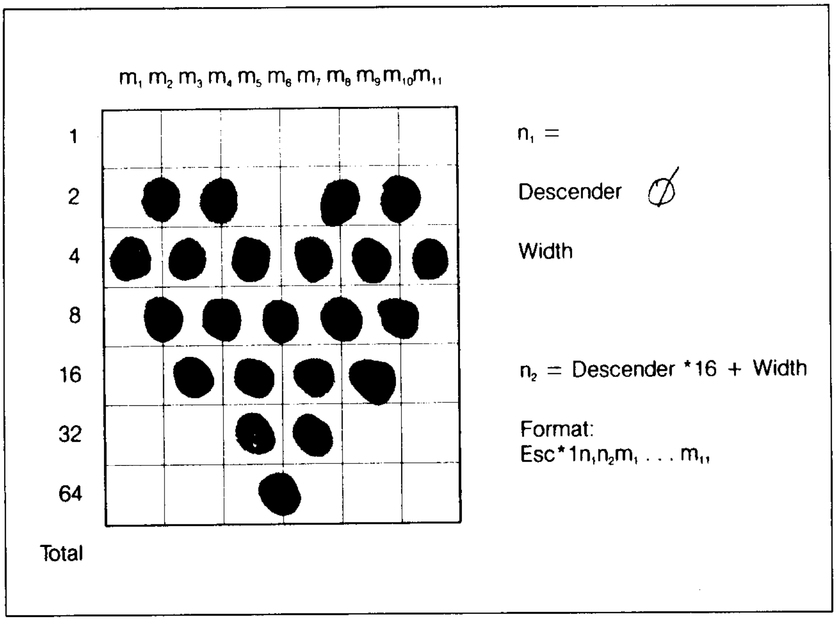

m_1 m_2 m_3 m_4 m_5 m_6 m_7 m_8 m_9 m_{10} m_{11}
1
2
4
8
16
32
64
Total
n_1 =
Descender Ø
Width
n_2 = Descender *16 + Width
Format:
Esc*1$n_1n_2m_1$ . . . m_{11}

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.)

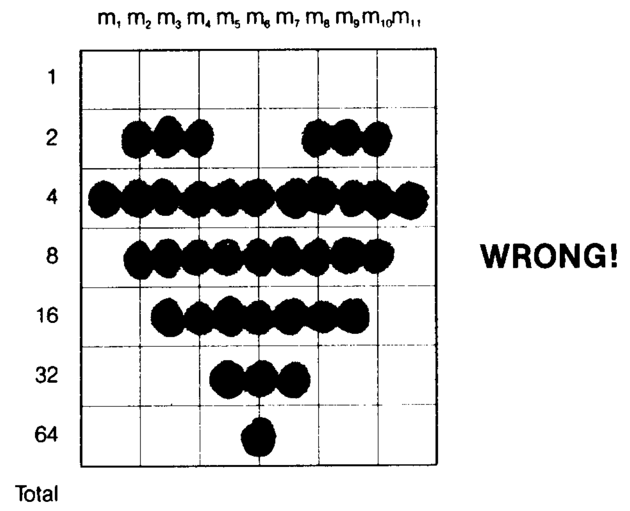

m_1 m_2 m_3 m_4 m_5 m_6 m_7 m_8 m_9 m_{10} m_{11}
1
2
4
8
16
32
64
Total
WRONG!

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

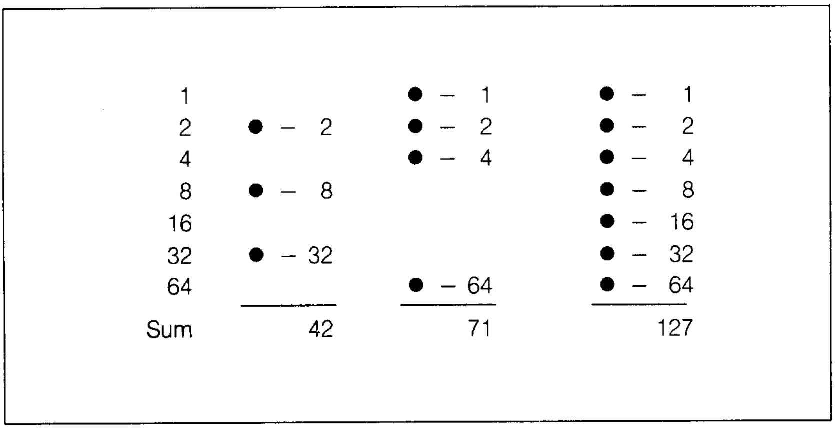


Figure 7-7. *By adding the values of each dot in a column, you'll get a unique description for any combination of dots.*

answers!). Across the top of the grid you've probably noticed the cryptic labeling of each column: *m1, m2, m3*, etc. These labels correspond to the labels in the command syntax statement, which we'll get to shortly.

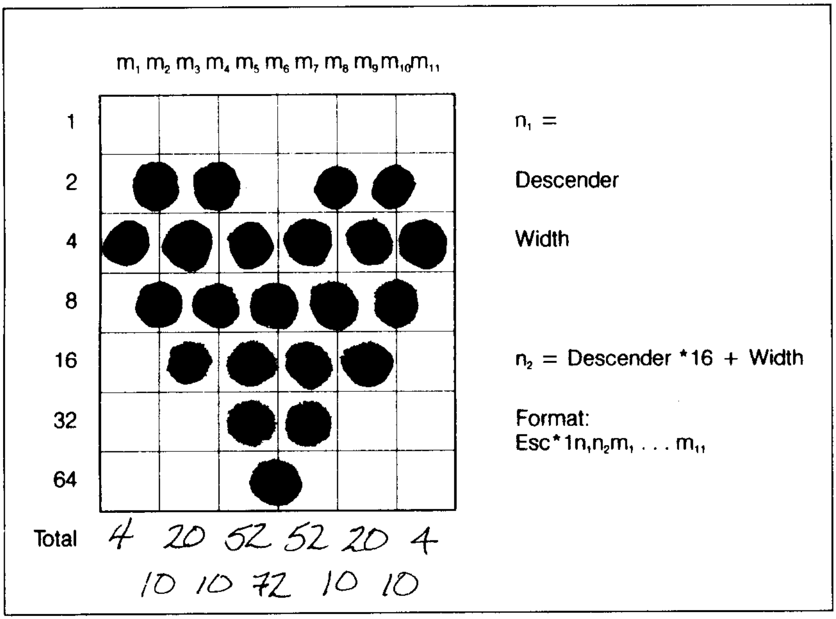


Figure 7-8. *Add the values of the dots in each column and write the sum of each column at the bottom.*

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) n1n2m1m2m3m4m5m6m7m8m9m10m11
```

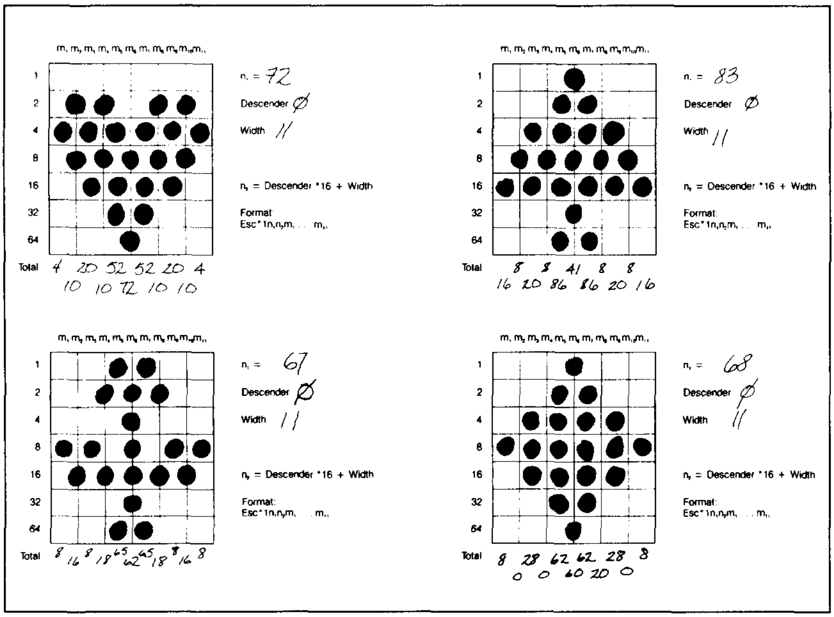


Figure 7-9. *Character designs for the four card suits.*

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.)

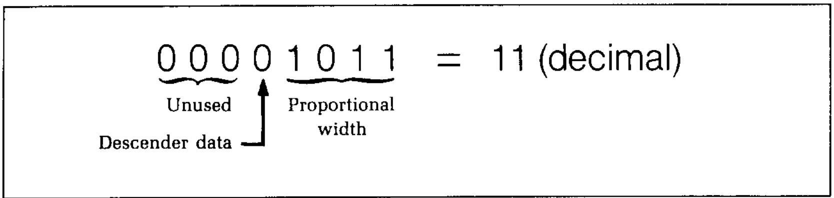


Figure 7-10. *The attribute byte (n2) for our heart character.*

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.

CHR$(27)	CHR$(42)	CHR$(1)	CHR$(72)	CHR$(11)	CHR$(4)
Escape	*	1	n_1	n_2	m_1
CHR$(10)	CHR$(20)	CHR$(10)	CHR$(52)	CHR$(72)	
m_2	m_3	m_4	m_5	m_6	
CHR$(52)	CHR$(10)	CHR$(20)	CHR$(10)	CHR$(4)	
m_7	m_8	m_9	m_{10}	m_{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.

```
1Ø FOR I=1 TO 4
2Ø LPRINT CHR$(27) "*" CHR$(1);
3Ø READ N1,N2
4Ø LPRINT CHR$(N1) CHR$(N2);
5Ø FOR M=1 TO 11
6Ø READ M1
7Ø LPRINT CHR$(M1);
8Ø NEXT M
9Ø NEXT
```

```
1ØØ LPRINT
11Ø DATA 72,11,4,1Ø,2Ø,1Ø,52,72,52,1Ø,2Ø,1Ø,4
12Ø DATA 83,11,16,8,2Ø,8,86,41,86,8,2Ø,8,16
13Ø DATA 67,11,8,16,8,18,65,62,65,18,8,16,8
14Ø DATA 68,11,8,Ø,28,Ø,62,65,62,Ø,28,Ø,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 rerun 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.)

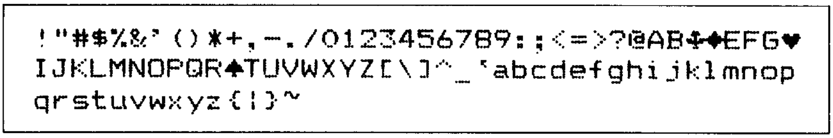
So does that mean that in order to print something meaningful with our card suits we have to define an entire alphabet? Fear not. The engineers at Star have made it an easy task to use mostly standard characters with just a few special characters thrown in. This command copies all the characters from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

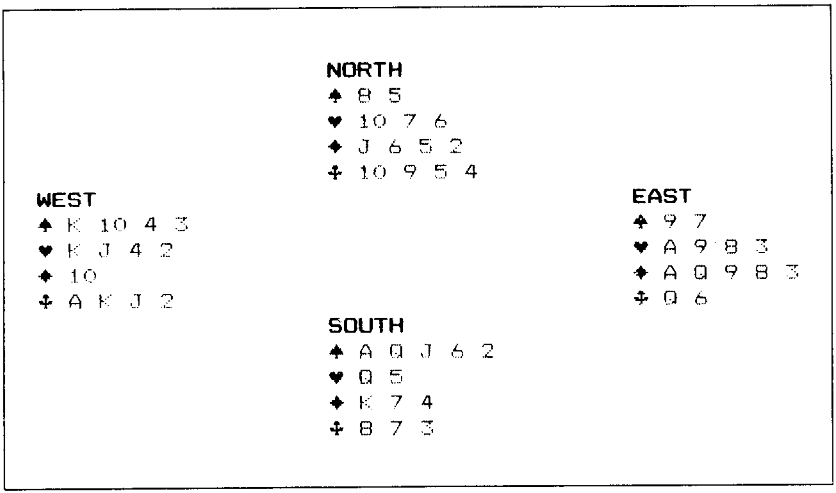
To demonstrate how to use these characters, let's use this character set to print a typical bridge hand. This program will do just that:

```
1Ø 'Program to deal bridge hands and print on Delta
2Ø GOSUB 1ØØØ          'Initialize variables
3Ø GOSUB 2ØØØ          'Initialize printer
4Ø GOSUB 3ØØØ          'Deal cards
5Ø GOSUB 4ØØØ          'Print hands
6Ø END
1ØØØ 'Initialize variables
1Ø1Ø DEFINT A-Z
1Ø2Ø DIM HAND(4), DECK(52), CARD$(13), SUIT$(3)
1Ø3Ø CARD$(1)=" 2" : CARD$(2)=" 3" : CARD$(3)=" 4" :
   CARD$(4)=" 5" : CARD$(5)=" 6"
1Ø4Ø CARD$(6)=" 7" : CARD$(7)=" 8" : CARD$(8)=" 9" :
   CARD$(9)=" 1Ø"
1Ø5Ø CARD$(1Ø)=" J" : CARD$(11)=" Q" : CARD$(12)="
   K" : CARD$(13)=" A"
1Ø6Ø SUIT$(Ø)="S" : SUIT$(1)="H" : SUIT$(2)="D" :
   SUIT$(3)="C"
1Ø7Ø INPUT "Random number seed";I
1Ø8Ø RANDOMIZE I
1Ø9Ø RETURN
2ØØØ 'Initialize printer
2Ø1Ø LPRINT CHR$(27) CHR$(68) CHR$(2Ø) CHR$(4Ø)
   CHR$(Ø)    'Set tabs
```

```
2Ø2Ø LPRINT CHR$(27) CHR$(43) CHR$(27) CHR$(36)
   CHR$(Ø) CHR$(27) CHR$(69) CHR$(3Ø)       'Macro
   instruction is used to select standard
   characters, emphasized
2Ø3Ø LPRINT CHR$(27) CHR$(42) CHR$(Ø)
   'Load standard characters in RAM
2Ø4Ø FOR I=1 TO 4
   'This loop reads data for the four
2Ø5Ø LPRINT CHR$(27) CHR$(42) CHR$(1);
   'card suit characters and sends it
2Ø6Ø FOR J=1 TO 13    'to the printer
2Ø7Ø READ X : LPRINT CHR$(X);
2Ø8Ø NEXT J
2Ø9Ø NEXT I
21ØØ LPRINT
211Ø RETURN
212Ø DATA 72,11,4,1Ø,2Ø,1Ø,52,72,52,1Ø,2Ø,1Ø,4
213Ø DATA 83,11,16,8,2Ø,8,86,41,86,8,2Ø,8,16
214Ø DATA 67,11,8,16,8,18,65,62,65,18,8,16,8
215Ø DATA 68,11,8,Ø,28,Ø,62,65,62,Ø,28,Ø,8
3ØØØ 'Deal cards
3Ø1Ø FOR CARD = 1 TO 52
3Ø2Ø X = INT(RND * 4 + 1)
3Ø3Ø IF HAND(X)=13 THEN 3Ø2Ø ELSE HAND(X)=HAND(X)+1
3Ø4Ø DECK(CARD)=X
3Ø5Ø NEXT CARD
3Ø6Ø RETURN
4ØØØ 'Print four hands
4Ø1Ø LPRINT CHR$(27) "!" CHR$(9) "NORTH"
4Ø2Ø LPRINT CHR$(27) "$" CHR$(1) CHR$(27) CHR$(7Ø);
4Ø3Ø HAND = 1
4Ø4Ø FOR SUIT = Ø TO 3
4Ø5Ø LPRINT CHR$(9);
4Ø6Ø GOSUB 43ØØ
4Ø7Ø LPRINT
4Ø8Ø NEXT SUIT
4Ø9Ø LPRINT CHR$(27) "!" "WEST" CHR$(9) CHR$(9)
   "EAST"
41ØØ LPRINT CHR$(27) "$" CHR$(1) CHR$(27) CHR$(7Ø);
411Ø FOR SUIT = Ø TO 3
412Ø HAND = 2
413Ø GOSUB 43ØØ
414Ø LPRINT CHR$(9) CHR$(9);
415Ø HAND = 3
416Ø GOSUB 43ØØ
```

```
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.



```
                    NORTH
                    ♠ 8 5
                    ♥ 10 7 6
                    ♦ J 6 5 2
                    ♣ 10 9 5 4
WEST                                        EAST
♠ K 10 4 3                                  ♠ 9 7
♥ K J 4 2                                   ♥ A 9 8 3
♦ 10                                        ♦ A Q 9 8 3
♣ A K J 2                                   ♣ Q 6
                    SOUTH
                    ♠ A Q J 6 2
                    ♥ Q 5
                    ♦ K 7 4
                    ♣ 8 7 3
```

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.

```
1Ø DATA  77,11,1,126,1,2,4,8,4,2,1,126,1
2Ø DATA 1Ø5,4,64,61,64,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø
3Ø DATA 112,23,127,Ø,17,Ø,17,14,Ø,Ø,Ø,Ø,Ø
4Ø DATA 115,6,8,84,Ø,84,32,Ø,Ø,Ø,Ø,Ø,Ø
5Ø DATA -1
6Ø READ CHR
7Ø IF CHR < Ø THEN 15Ø
8Ø READ CODE
9Ø LPRINT CHR$(27) "*" CHR$(1) CHR$(CHR) CHR$(CODE) ;
```

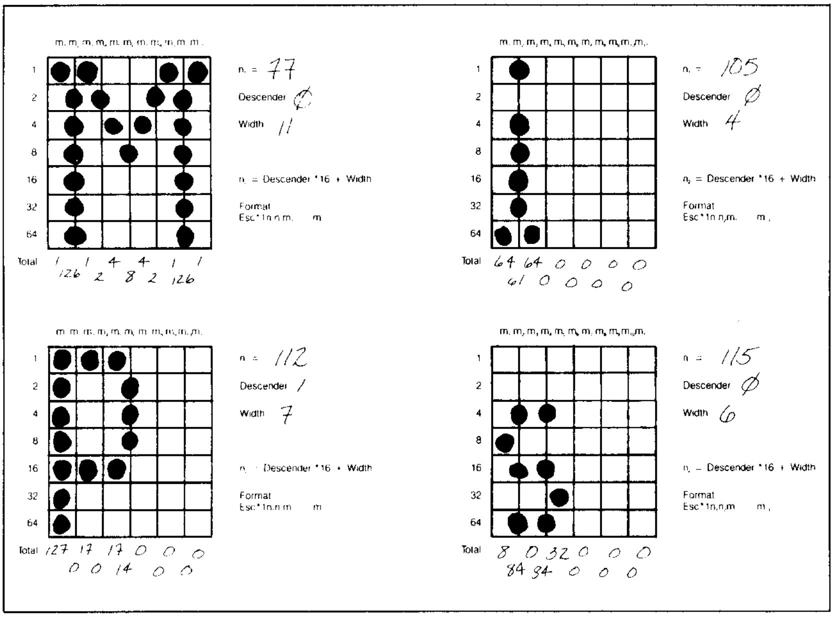


Figure 7-14. *These download characters are defined as proportional characters.*

```
100 FOR I = 1 TO 11
110 READ BITS
120 LPRINT CHR$(BITS) ;
130 NEXT I
140 GOTO 60
150 LPRINT "      Mississippi"
155 LPRINT
160 LPRINT "Standard characters without proportional
   spacing"
170 LPRINT
175 LPRINT
180 LPRINT CHR$(27) "$" CHR$(1) "      Mississippi"
190 LPRINT CHR$(27) "$" CHR$(0)
200 LPRINT "Download characters without proportional
   spacing"
210 LPRINT
215 LPRINT
220 LPRINT CHR$(27) "X" CHR$(1) "      Mississippi"
230 LPRINT CHR$(27) "X" CHR$(0)
240 LPRINT "Download characters with proportional
   spacing"
```

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 ⟨ESC⟩ "*" 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).

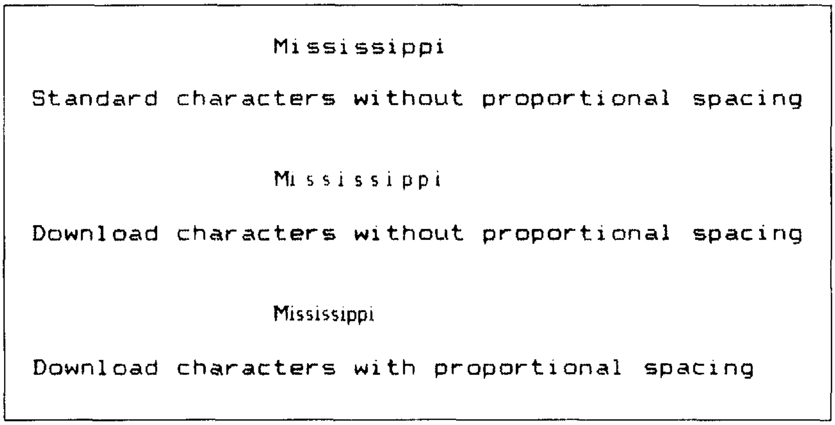
Printing proportional characters

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:

```
    ⟨ESC⟩ "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: ⟨ESC⟩ "$" CHR$(1) for normal width or ⟨ESC⟩ "X" CHR$(1) for proportional width. The two commands work independently of each other, so that ⟨ESC⟩ "$" CHR$(0) will *not* turn off proportional download characters, and ⟨ESC⟩ "X" CHR$(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 ⟨ESC⟩ "X" CHR$(0). The printer will then continue to print with normal width download characters (rather than returning to the standard character set) until you send an ⟨ESC⟩ "$" CHR$(0). This can lead to confusion if you have accidentally specified both types of download characters.



```
Mississippi

Standard characters without proportional spacing

Mississippi

Download characters without proportional spacing

Mississippi

Download characters with proportional spacing
```

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.

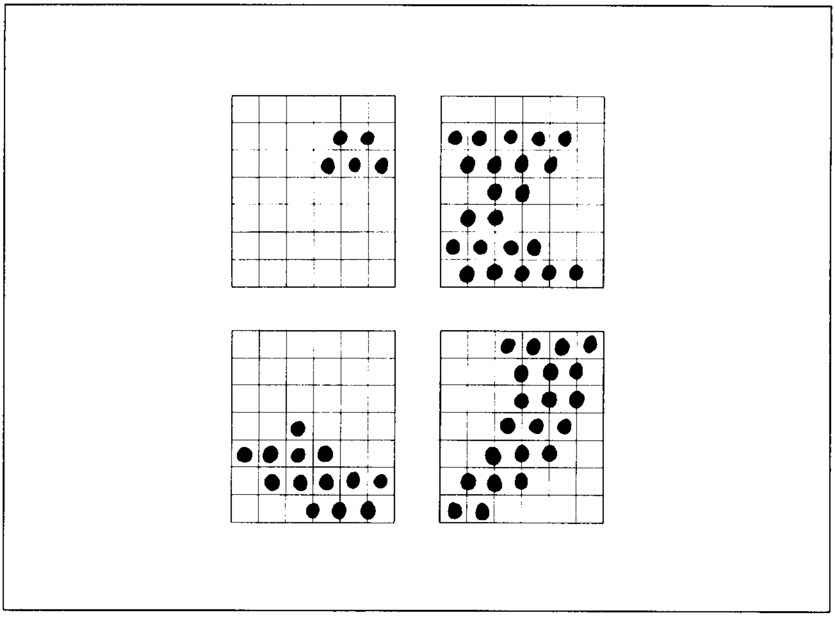


Figure 7-16. *Each digit is made up of four individual characters.*

```
1Ø 'Program to define and print numerals
2Ø 'Each numeral is made up of 4 characters (2 wide
   x 2 high)
3Ø DEF.DOWN.CHAR$ = CHR$(27)+CHR$(42)+CHR$(1)
4Ø DOWN.CHAR.PROP$ = CHR$(27)+CHR$(88)+CHR$(1)
5Ø NOT.DOWN.CHAR.PROP$ = CHR$(27)+CHR$(88)+CHR$(Ø)
6Ø LINE.7$ = CHR$(27)+CHR$(49) : LINE.12$ =
   CHR$(27)+CHR$(5Ø)
7Ø FOR N1 = 16Ø TO 2ØØ     '4 ASCII CHARS/NUMERAL
8Ø LPRINT DEF.DOWN.CHAR$;
9Ø LPRINT CHR$(N1);
1ØØ READ N2
11Ø LPRINT CHR$(N2);
12Ø FOR S = 1 TO 11
13Ø READ MS
14Ø LPRINT CHR$(MS);
15Ø NEXT S
16Ø NEXT N1
17Ø '
18Ø ASCII = 16Ø             'START OF DOWN CHARACTERS
19Ø FOR NUM = Ø TO 9        'NUMERALS Ø THRU 9
2ØØ NUMERAL.TOP$(NUM) = CHR$(ASCII + Ø) +
   CHR$(ASCII + 1)
```

```
21Ø NUMERAL.BOT$(NUM) = CHR$(ASCII + 2) +
   CHR$(ASCII + 3)
22Ø ASCII = ASCII + 4
23Ø NEXT NUM
24Ø BLANK$ = CHR$(2ØØ)
25Ø LPRINT DOWN.CHAR.PROP$; LINE.7$
26Ø FOR NUM = Ø TO 9
27Ø LPRINT NUMERAL.TOP$(NUM);BLANK$;
28Ø NEXT NUM
29Ø LPRINT
3ØØ FOR NUM = Ø TO 9
31Ø LPRINT NUMERAL.BOT$(NUM);BLANK$;
32Ø NEXT NUM
33Ø LPRINT NOT.DOWN.CHAR.PROP$; LINE.12$
34Ø 'ZERO
35Ø DATA 11,Ø,96,16,1Ø4,16,44,3Ø,14,Ø,2,1
36Ø DATA 11,2,1,2,1,6,8,38,88,32,88,32
37Ø DATA 11,3,12,19,12,51,Ø,96,Ø,96,Ø,96
38Ø DATA 11,Ø,32,Ø,48,Ø,28,3,12,3,4,3
39Ø 'ONE
4ØØ DATA 11,Ø,Ø,Ø,Ø,Ø,4,Ø,4,Ø,4,126
41Ø DATA 9,12,114,12,114,12,2,Ø,Ø,Ø,Ø,Ø
42Ø DATA 11,64,Ø,64,Ø,64,Ø,64,32,8Ø,47,8Ø
43Ø DATA 9,47,8Ø,47,64,Ø,64,Ø,64,Ø,Ø,Ø
44Ø ' TWO
45Ø DATA 11,Ø,Ø,Ø,Ø,Ø,12,16,14,Ø,6,Ø
46Ø DATA 11,3,Ø,3,Ø,7Ø,56,7Ø,56,4,24,Ø
47Ø DATA 11,64,Ø,64,32,64,32,8Ø,32,8Ø,4Ø,64
48Ø DATA 11,44,64,38,65,34,65,32,8Ø,32,88,Ø
49Ø ' THREE
5ØØ DATA 11,Ø,Ø,Ø,Ø,Ø,Ø,4,2,4,2,4
51Ø DATA 11,34,84,34,92,34,76,34,68,2,64,Ø
52Ø DATA 11,16,Ø,48,Ø,56,64,48,64,32,64,32
53Ø DATA 11,64,32,64,48,9,54,9,22,9,6,1
54Ø ' FOUR
55Ø DATA 11,Ø,Ø,Ø,Ø,Ø,Ø,64,36,88,32,16
56Ø DATA 11,Ø,Ø,64,32,64,56,64,6Ø,2,12,Ø
57Ø DATA 11,Ø,8,4,1Ø,5,1Ø,5,8,4,72,4
58Ø DATA 11,88,38,89,38,89,6,73,4,8,6,Ø
59Ø ' FIVE
6ØØ DATA 11,Ø,Ø,Ø,Ø,64,32,84,5Ø,76,34,68
61Ø DATA 1Ø,34,68,34,68,34,68,2,68,2,Ø,Ø
62Ø DATA 1Ø,Ø,32,24,1Ø1,24,97,Ø,64,Ø,64,Ø
```

```
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
```

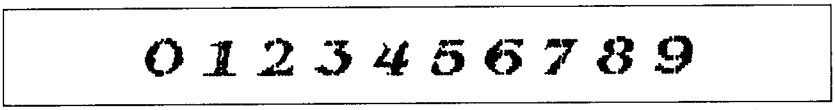


Figure 7-17. *The output for characters like this must be carefully planned.*

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
```

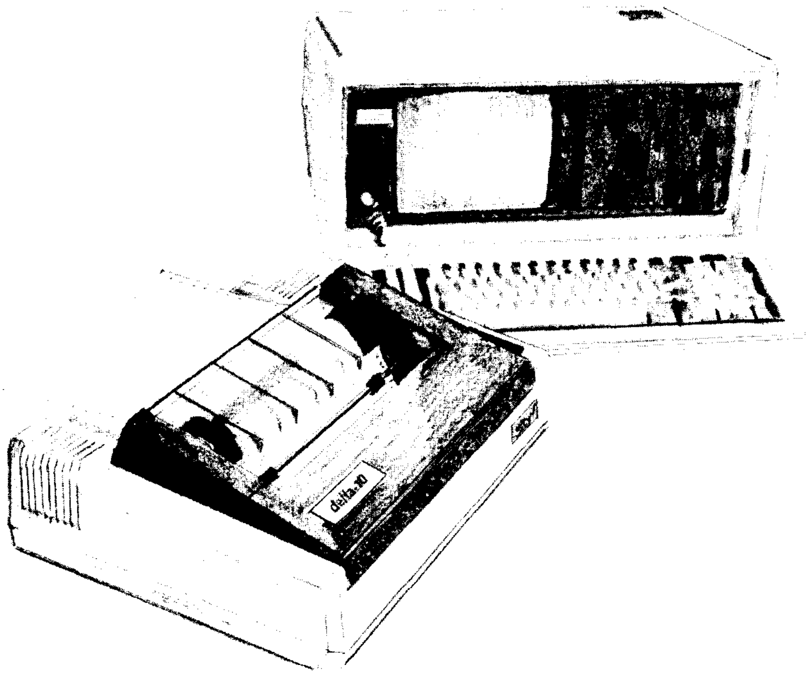
```
13Ø P1=1:M$=CHR$(179)+
   STRING$(2,32):N$=STRING$(2,196)+
   CHR$(197):L$=STRING$(2,196)+CHR$(193)
14Ø LOCATE 4,1Ø:PRINT CHR$(218);CHR$(196);
15Ø FOR I=1 TO 1Ø:PRINT
   CHR$(196);CHR$(194);CHR$(196); :NEXT I
16Ø PRINT CHR$(196);CHR$(191):LOCATE 5,1Ø:FOR K=1 TO
   12:PRINT M$;:NEXT K:PRINT
17Ø FOR J=1 TO 6:LOCATE 5+P1,1Ø:P1=P1+1:PRINT
   CHR$(195);
18Ø FOR K=1 TO 1Ø:PRINT N$;:NEXT K
19Ø PRINT CHR$(196);CHR$(196);CHR$(18Ø):LOCATE
   5+P1,1Ø:P1=P1+1
2ØØ FOR K=1 TO 12:PRINT M$;:NEXT K
21Ø PRINT:NEXT J:LOCATE 18,1Ø:PRINT CHR$(192);
22Ø FOR I=1 TO 1Ø:PRINT L$;:NEXT I
23Ø PRINT CHR$(196);CHR$(196);CHR$(217)
24Ø FOR I=Ø TO 6:LOCATE  5+I*2,6:PRINT 2^I;:NEXT I
25Ø RETURN :  REM  **** END OF MATRIX SUBROUTINE
   ****
26Ø REM  **** SINGLE CHARACTER INPUT @ EDIT LEVEL
   ****
27Ø LOCATE 5,11:PRINT CS$;:GOSUB 59Ø
28Ø A$=INKEY$:IF A$="" THEN 28Ø
29Ø B$=RIGHT$(A$,1)
3ØØ IF B$=CHR$(75) THEN GOSUB 39Ø:GOTO 37Ø
31Ø IF B$=CHR$(77) THEN GOSUB 41Ø:GOTO 37Ø
32Ø IF B$=CHR$(8Ø) THEN GOSUB 43Ø:GOTO 37Ø
33Ø IF B$=CHR$(72) THEN GOSUB 45Ø:GOTO 37Ø
34Ø IF B$=CHR$(82) THEN GOSUB 47Ø:GOTO 37Ø
35Ø IF B$=CHR$(83) THEN GOSUB 49Ø:GOTO 37Ø
36Ø IF B$=CHR$(79) THEN GOSUB 5ØØ:GOTO 38Ø
37Ø GOTO 28Ø
38Ø RETURN : REM  **** END OF INPUT ****
39Ø GOSUB 92Ø:Y=Y-3:H=H-1:IF Y<1 THEN BEEP:Y=1:H=1
4ØØ GOSUB 95Ø:RETURN
41Ø GOSUB 92Ø:Y=Y+3:H=H+1:IF Y>31 THEN
   BEEP:Y=31:H=11
42Ø GOSUB 95Ø:RETURN
43Ø GOSUB 92Ø:X=X+2:G=G+1:IF X>13 THEN BEEP:X=13:G=7
44Ø GOSUB 95Ø:RETURN
45Ø GOSUB 92Ø:X=X-2:G=G-1:IF X<1 THEN BEEP:X=1:G=1
46Ø GOSUB 95Ø:RETURN
47Ø 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$(55,32);:NEXT I
8ØØ IF SH=1 THEN SH=16 ELSE SH=Ø
81Ø N1=AS:N2=PR+SH
82Ø FOR I=1 TO 11:MM$=MM$+CHR$(MM(I)):NEXT I
83Ø LPRINT
   CHR$(27);"*";CHR$(1);CHR$(N1);CHR$(N2);MM$
84Ø IF AN$="N" THEN LPRINT CHR$(27);"$";CHR$(1)
   :GOTO 86Ø
85Ø LPRINT CHR$(27);"X";CHR$(1)
86Ø FOR I=1 TO 2Ø:LPRINT CHR$(N1);" ";:NEXT I:LPRINT
87Ø LPRINT CHR$(14);:FOR I=1 TO 1Ø:LPRINT CHR$(N1);"
   ";:NEXT I:LPRINT CHR$(2Ø)
88Ø LPRINT CHR$(15);:FOR I=1 TO 2Ø:LPRINT CHR$(N1);"
   ";:NEXT I:LPRINT CHR$(18)
89Ø IF AN$="N" THEN LPRINT CHR$(27);"$";CHR$(Ø):GOTO
   91Ø
9ØØ LPRINT CHR$(27);"X";CHR$(Ø)
91Ø LPRINT CHR$(27);"@":MM$="":RETURN :REM **** END
   OF PRINT MODE ****
92Ø IF Z(G,H)=Ø THEN LOCATE X+4,Y+1Ø:PRINT "  ";
93Ø IF Z(G,H)=1 THEN LOCATE X+4,Y+1Ø:COLOR 7,Ø:PRINT
   SC$;
94Ø RETURN
95Ø IF Z(G,H)=1 THEN COLOR 31,1: LOCATE
   X+4,Y+1Ø:PRINT CS$;: COLOR 7,Ø
96Ø IF Z(G,H)=Ø THEN COLOR 7,Ø: LOCATE
   X+4,Y+1Ø:PRINT CS$;: COLOR 7,Ø
97Ø 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)	Cancels proportional download character set
⟨ESC⟩ "$" CHR$(1)	Selects the download character set and uses normal spacing
⟨ESC⟩ "$" CHR$(0)	Cancels normal download character set



Chapter 8

Printing Dot Graphics

In Chapter 7 you were introduced to a form of computer graphics; you were able to actually define characters dot by dot. In this chapter you'll learn to use the same principles to make Delta print whole pages of dot graphics! We'll show you how to use dot graphics to create "super download characters." In addition, you'll see how your Delta printer can be used as a graphics plotter. This can have some practical business applications as well as create some terrific computer art!

Comparing Dot Graphics with Download Characters

A good understanding of dot graphics requires an understanding of how dot matrix printers work; you may want to review the first few pages of Chapter 7. The principles for dot graphics are the same as those for download characters.

There are some differences in the way they are implemented however. While download commands can be used to define a character between four and eleven columns of dots wide, dot graphics commands can be used to define a shape as narrow as one column of dots wide or as wide as 3264 dots on a Delta-15!

There is no "descender data" with dot graphics; graphics images are always printed with the top seven or eight pins of the print head, depending on whether you have a 7-bit or 8-bit interface (if you're not sure which type of interface your computer has, check the appendix for your computer).

So when do you use graphics and when do you use download characters? Practically anything you can do with graphics you can do with download characters, and vice versa. A clever programmer could actually plot a mathematical curve using download characters or use strings of graphics data as user-defined characters. But why do it the hard way? There are several instances when dot graphics is clearly the best way to approach the problem:

- If the graphic image to be printed is wider than 11 dots or higher than 7 dots
- If an image is to be printed just one time, as opposed to a frequently used "text" character
- If you want higher resolution (Delta can print as many as 240 dots per inch in dot graphics mode; text mode, which includes download characters, prints 60 dots per inch)

Using the Dot Graphics Commands

The command to print normal density (60 dots per inch horizontal; 72 dots per inch vertical) dot graphics uses this format:

```
<ESC> "K" n1 n2 m1 m2. . .
```

Just like many of the other codes you have learned, the command starts with an escape sequence (<ESC> "K" in this case). But unlike Delta's other codes there can be any number of graphics data bytes following the command. That's where *n1* and *n2* come in; they are used to tell Delta how many bytes of graphics data to expect.

Specifying the number of columns of dots

To figure the values of *n1* and *n2*, you'll need to figure out how wide your graphic image will be (remember that there are 60 columns of dots per inch in normal density). Then comes the fun part: converting one number (the number of columns of dots) into two! Why is it necessary to use two numbers to tell Delta the number of graphics codes to expect? Because the largest number we can send in one byte (that's what the BASIC CHR$( ) function sends: one byte) is 255. And with normal density graphics it's possible to have a graphics image as wide as 480 dots on Delta-10 or 816 dots on Delta-15. So to figure out how many columns of graphics data to expect, Delta multiplies *n2* by 256 and adds the value of *n1*. If you divide the number of columns by 256, then *n2* is the quotient and *n1* is the remainder (why not let your computer figure it out for you: if the number of columns is assigned to variable X, then N1 = X MOD 256 and N2 = INT(X/256)). Table 8-1 might make things even easier.

Table 8-1
Calculating n1 and n2

If the number of columns, x, ranges from:	then *n1* is:	and *n2* is:
1 to 255	x	0
256 to 511	x - 256	1
512 to 767	x - 512	2
768 to 1023	x - 768	3
1024 to 1279	x - 1024	4
1280 to 1535	x - 1280	5
1536 to 1791	x - 1536	6
1792 to 2047	x - 1792	7
2048 to 2303	x - 2048	8
2304 to 2559	x - 2304	9
2560 to 2815	x - 2560	10
2816 to 3071	x - 2816	11
3072 to 3264	x - 3072	12

Specifying the graphics data

Now that we've told Delta data how much data to expect, we better figure out how to send that information! Just as you do with download characters, with dot graphics you have control over the firing of every single pin on Delta's print head. In Figure 8-1, you can see that we've labeled each pin on the print head with a number, as we did with download characters (you should note one important difference: this time the *top* pin has the highest value; for download character definitions it is the bottom pin). And specifying pins to fire is done in the same way: to fire the second pin from the top, for instance, send a CHR$(64). Firing several pins at once is done in a similar fashion. For example, to print the first, third, and fourth dots, add their values (128 + 32 + 16) to send this total: CHR$(176).

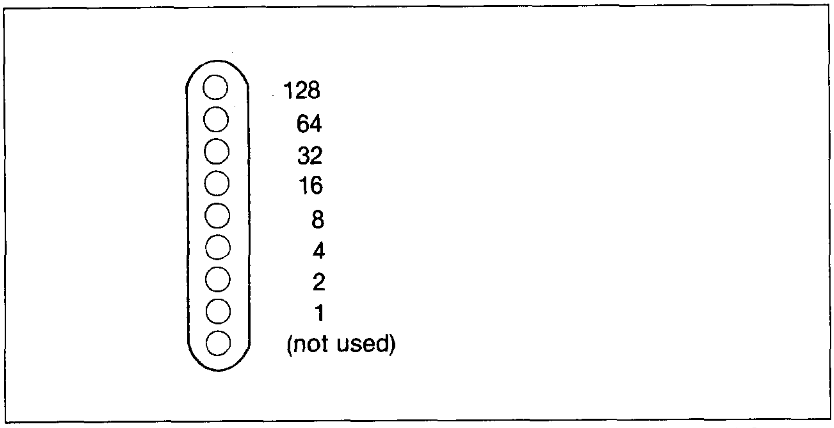


Figure 8-1. *Starting with the most significant bit at the top, each pin of the print head is assigned a value which is a power of two. Note that for 7-bit computers, the top pin has a value of 64, and the bottom two pins are unused.*

A short program should demonstrate how to implement the graphics command. The program below gave us this printout:

```
1Ø 'PRINT GRAPHIC PATTERN
2Ø WIDTH "LPT1:",255
3Ø LPRINT CHR$(27) CHR$(75) CHR$(94) CHR$(1);
4Ø FOR I = 1 TO 25
5Ø FOR J = Ø TO 6
6Ø LPRINT CHR$(2^J);
7Ø NEXT J
8Ø FOR J = 6 TO Ø STEP -1
9Ø LPRINT CHR$(2^J);
1ØØ NEXT J
11Ø NEXT I
```

```
12Ø WIDTH "LPT1:",8Ø
13Ø LPRINT
```

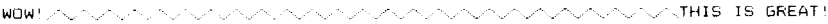
In line 30 we've selected normal density graphics and said that 350 characters of graphics data would follow (94 + (1 * 256) = 350). The loop between lines 40 and 110 is repeated 25 times; this is what gives us the "zigzag" effect. The loop from line 50 to line 70 creates the lines that slope up; the loop between lines 80 and 100 prints the downward sloping lines. This is an example of plotting a very simple mathematical function to create a design. Later in this chapter we'll show something more complex.

Combining text and graphics

It's also possible to mix text and graphics in one line. This can be useful for labeling charts or graphs, or even inserting fancy graphics in text. Try adding these lines to our program:

```
25 LPRINT "WOW!";
115 LPRINT "THIS IS GREAT!";
```

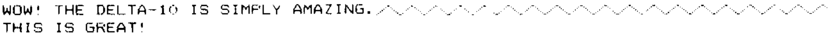
Now if you run the program you should get a printout that looks like this:



```
WOW!                                                              THIS IS GREAT!
```

But there is one thing to be careful of: all graphics data must print on the same line. The graphics command is turned off at the end of each line, even if you have specified that more graphics codes follow. To see what we mean, change line 25 as shown and run the program.

```
25 LPRINT "WOW! THE DELTA-10 IS SIMPLY AMAZING.";
```



```
WOW! THE DELTA-10 IS SIMPLY AMAZING.
THIS IS GREAT!
```

(To get the same effect, Delta-15 users should change two program lines:

```
30 LPRINT CHR$(27) CHR$(75) CHR$(188) CHR$(2);
40 FOR I = 1 TO 50
```

This will make the zigzag pattern long enough to go off the page.)

As you can see, Delta printed graphics up to the end of the line, then ignored the rest of the graphics data and returned to normal text on the next line.

Printing a Design or Logo

Since you control the firing of every pin, you can print nearly anything with Delta that you can draw (and probably better, if you're like most computer users!). This can be used for creating "computer art" or drawing maps. Or, as we'll show you here, you can use dot graphics to print your logo at the top of each letter you print.

Designing an image to print with dot graphics is much like designing download characters. The best way to start is to lay out your image on graph paper. Since you can print eight rows (seven with a 7-bit interface) of dots with each pass of the print head, draw a heavy horizontal line every eight rows on your graph paper. And it may be helpful to write the dot values (128, 64, 32, etc.) down the left side of each row. Then after you've filled in the "dots" that you want to print, it's time to get out the old calculator again! Just as you did with download characters, add up the values of each column of dots; this makes up one byte.

In the program below, we've taken the logo graphics information and put it into BASIC DATA statements. The program itself is short and simple. The loop starting at line 160 reads the data statements into a string array variable called LOGO$. In line 230 we change the line spacing to 8/72 inch so that the lines of graphics data will connect vertically. The actual printing is done in the loop between lines 250 and 280; line 260 sends the graphics control code to Delta and line 270 sends one line of graphics data.

The printout from the program is shown right below the program.

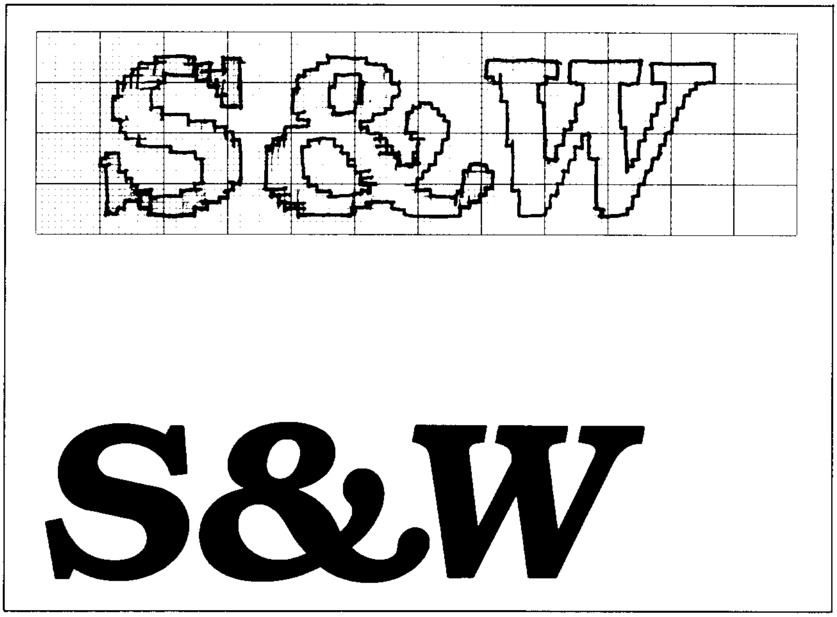


Figure 8-2. *By laying out the logo on graph paper, you can calculate all of the graphics data.*

```
1ØØ ' PRINT S&W LOGO
11Ø LINE.8$ = CHR$(27)+CHR$(65)+CHR$(8)  'Set line
   spacing to 8 dots
12Ø LINE.12$ = CHR$(27)+CHR$(5Ø)         'Set line
   spacing to 1/6"
13Ø GRAPHIC$ = CHR$(27)+CHR$(75)         'Select dot
   graphics
14Ø DIM LOGO$(4)
15Ø WIDTH "LPT1:",255
16Ø ' READ DATA
17Ø FOR ROW = 1 TO 4
18Ø FOR COLUMN = 1 TO 1ØØ
19Ø READ P
2ØØ LOGO$(ROW) = LOGO$(ROW) + CHR$(P)
21Ø NEXT COLUMN
22Ø NEXT ROW
23Ø ' PRINT LOGO
24Ø LPRINT LINE.8$;
25Ø FOR ROW = 1 TO 4
26Ø LPRINT GRAPHIC$;CHR$(1ØØ);CHR$(Ø);
27Ø LPRINT LOGO$(ROW)
28Ø NEXT ROW
```

```
290 LPRINT LINE.12$
300 'ROW 1
310 DATA Ø,Ø,Ø,Ø,1,3,7,7,7,15,14,14,14,
   14,14,7,7,3,3,15
320 DATA 15,15,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,
   Ø,1,3,3,7,7,15,14,14,14
330 DATA 14,15,7,7,7,3,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø
340 DATA Ø,6,7,7,7,7,7,7,7,7,6,6,Ø,Ø,7,7,7,7,7,7
350 DATA 7,7,7,7,7,Ø,Ø,7,7,7,7,7,7,7,7,7,7,Ø,Ø,Ø
360 ' ROW 2
370 DATA Ø,Ø,6Ø,255,255,255,255,255,143,15,7,7,7,7,
   3,3,3,131,193,241
380 DATA 24Ø,24Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,1,121,253,
   253,255,255,255,143,7,7,7
390 DATA 31,253,252,248,248,24Ø,192,Ø,7,15,
   31,31,15,7,3,Ø,Ø,Ø,Ø,Ø
400 DATA Ø,Ø,Ø,224,255,255,255,255,255,31,Ø,
   Ø,Ø,1,3,31,255,255,255,255
410 DATA 255,255,1,Ø,Ø,Ø,1,7,31,255,252,24Ø,192,
   128,Ø,Ø,Ø,Ø,Ø,Ø
420 'ROW 3
430 DATA Ø,31,31,3,129,128,192,192,192,192
440 DATA 192,224,224,224,224,24Ø,255,255,255,255
450 DATA 255,127,Ø,Ø,Ø,Ø,63,127,255,255
460 DATA 255,255,193,128,128,128,128,192,224,24Ø
470 DATA 252,255,255,255,127,63,31,7,7,31
480 DATA 254,252,248,224,128,Ø,Ø,3,7,7,7,3,Ø,Ø,
   192,255,255,255,255,255
490 DATA 15,15,63,252,24Ø,192,Ø,24Ø,255,255
500 DATA 255,255,255,7,15,127,252,24Ø,192,Ø,Ø,
   Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø
510 'ROW 4
520 DATA Ø,248,248,24Ø,224,224,112,112,56,56
530 DATA 56,56,56,12Ø,12Ø,24Ø,24Ø,224,224,192,128,
   Ø,Ø,Ø,Ø,Ø,192,224,24Ø,24Ø
540 DATA 24Ø,248,248,248,12Ø,12Ø,56,56,56,56
550 DATA 48,112,224,224,224,224,24Ø,24Ø,248,248
560 DATA 12Ø,12Ø,56,56,56,56,12Ø,24Ø,224,224
570 DATA 192,128,Ø,Ø,Ø,128,248,248,248,248,24Ø,
   192,Ø,Ø,Ø,Ø,Ø,Ø,24Ø,248
580 DATA 248,248,248,24Ø,192,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,
   Ø,Ø,Ø,Ø,Ø,Ø,Ø
```

Plotting with Delta

This section of the manual gets into more serious BASIC programming just because it's required in order to have the computer act as a plotter driver. Don't be intimidated; while it's beyond the scope of this manual to teach BASIC, if you try the examples and take it slowly you should be doing some fancy plotting of your own before you know it.

If designing and calculating dot graphics images by laying them out on graph paper seems too tedious to you, then let the computer do the work for you! With your computer doing the calculations and Delta plotting the output, you can come up with some terrific business graphs, charts, and mathematical function plots.

The best way to do this is to set up an array in memory. This is your "graph paper." The first thing to do is to determine how big you want your output to be; this will determine the size of your array. (If you have grandiose plans to fill an entire page with plotter output, you better have lots of memory in your computer. With 60 dots per inch horizontally and 72 dots per inch vertically, it takes at least 540 bytes of memory for each square inch of plotted area. That doesn't sound so bad—but an area 8 inches square requires over 32K!)

Your array should be two-dimensional (just like graph paper) where one dimension will be the number of columns of dots and the other dimension is the number of printing lines (remember that you can have up to eight rows of dots per printed line).

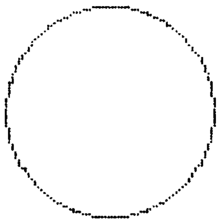
Here's a program that will use calculated-shape graphics to plot a circle. As you'll see, by changing a few lines it can be used to plot virtually any shape.

```
1ØØ ' DELTA-PLOT
1ØØØ ' Set program constants
1Ø1Ø MAXCOL% = 75          : MAXROW% = 14
1Ø2Ø DIM BIT%(MAXCOL%,MAXROW%)
1Ø3Ø MASK%(1) = 64         : MASK%(4) = 8
1Ø4Ø MASK%(2) = 32         : MASK%(5) = 4
1Ø5Ø MASK%(3) = 16         : MASK%(6) = 2
1Ø6Ø LX = 2Ø   : LY = 2Ø
```

```
1Ø7Ø LXFAC = 72/LX      : LYFAC = 87/LY
2ØØØ ' Plot curve
2Ø1Ø RAD = 9
2Ø2Ø X1 = 19   : Y1 = 1Ø
2Ø3Ø FOR ANG% = Ø TO 36Ø STEP 1Ø
2Ø4Ø RANG = ANG%*6.28/36Ø
2Ø5Ø X2 = RAD*COS(RANG)+1Ø  : Y2 = RAD*SIN(RANG)+1Ø
2Ø6Ø GOSUB 4ØØØ
2Ø7Ø NEXT ANG%
3ØØØ ' Send bit image map to printer
3Ø1Ø LPRINT CHR$(27) "A" CHR$(6)
3Ø2Ø FOR ROW% = Ø TO MAXROW%
3Ø3Ø A$ = ""
3Ø4Ø LPRINT CHR$(27) "K" CHR$(MAXCOL%) CHR$(Ø);
3Ø5Ø FOR COL% = 1 TO MAXCOL%
3Ø6Ø A$ = A$ + CHR$(BIT%(COL%,ROW%))
3Ø7Ø NEXT COL%
3Ø8Ø LPRINT A$ " "
3Ø9Ø NEXT ROW%
31ØØ LPRINT CHR$(27) "2"
311Ø END
4ØØØ ' Draw a line from X1,Y1 to X2,Y2
4Ø1Ø XL = X2 - X1        : YL = Y2 - Y1
4Ø2Ø NX = ABS(XL*LXFAC) : NY = ABS(YL*LYFAC)
4Ø3Ø IF NX < NY THEN NX = NY
4Ø4Ø NS% = INT(NX+1)
4Ø5Ø DX = XL/NS%         : DY = YL/NS%
4Ø6Ø FOR I% = 1 TO NS%
4Ø7Ø X1 = X1 + DX        : Y1 = Y1 + DY
4Ø8Ø GOSUB 5ØØØ
4Ø9Ø NEXT I%
41ØØ RETURN
5ØØØ ' Plot a point at X1,Y1
5Ø1Ø XX = X1 * LXFAC     : YY = Y1 * LYFAC
5Ø2Ø COL% = INT(XX) + 1
5Ø3Ø ROW% = INT(YY/6)
5Ø4Ø XIT% = INT(YY - ROW% * 6)+1
5Ø5Ø BIT%(COL%,ROW%) = BIT%(COL%,ROW%) OR
   MASK%(XIT%)
5Ø6Ø RETURN
```

How the program works

In the program above, we've created an array called BIT%, which is dimensioned in line 1020. You'll note that instead of



using numeric constants to dimension the array, we used the variables MAXCOL% and MAXROW%. This way, if your computer has enough memory and you want to plot a larger image, all you need to change are the values in line 1010. The array MASK% contains the values of the dots. (In order to make this program run on the most computers, we're using only six pins for graphics. With many computers, you can use all eight available pins.) In lines 1060 and 1070 we've defined some other variables you'll be interested in: LX, LXFAC, LY, and LYFAC are used as scaling factors. By changing these values, you can change the size of your printed image or even distort it (you can, for example, make our circle print as an ellipse). Experiment a little bit!

The main calculations for plotting the image are done starting at program line 2000. This is where you put the formulas that you want to plot. By changing just the lines between 2000 and 3000 (with some creative mathematics!) you can plot any function—limited only by your imagination. Some examples are shown at the end of this section.

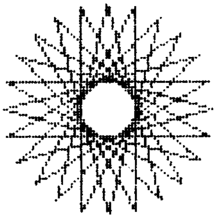
What the program section starting at 2000 actually does is to calculate starting and ending points for a line (in our circle the "lines" are very short—sometimes the starting and ending points are the same). The coordinates of the starting point of the line are assigned to variables X1 and Y1. The line ends at point X2,Y2. When these coordinates have been calculated, a subroutine call is made to line 4000. This subroutine calculates the coordinates of individual points along that line.

After these coordinates have been determined, the subroutine at line 5000 is called. This routine turns "on" an individual dot in our array called BIT%. (Keep in mind that no printing has been done yet; the computer is still drawing the image on its "graph paper" in memory.) The way an individual dot is turned on is using the logical OR function in line 5050.

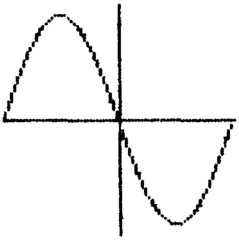
When all the points have been plotted in memory, printing begins at line 3000. We first set the line spacing to 6/72 inch using the ⟨ESC⟩ "A" command. This is so that there are no gaps between rows of dots. Then the loop from line 3020 to line 3090 prints the dot graphics image one line (which is six dots high) at a time. The variable A$ is used to build a string of all the columns of BIT% in a given row.

As you can see, by taking the program in small pieces and analyzing it, graphics programming does not have to be difficult. If you want to try some other plots, try these (replace lines between 2000 and 3000 with the lines below). The printouts from each program are shown below the listing.

```
2000 ' Plot curve
2010 RAD = 9
2020 FOR ANG% = 0 TO 360 STEP 15
2030 RANG = ANG%*6.28/360
2040 RANG2 = (ANG%+150)*6.28/360
2050 X1 = RAD*COS(RANG)+10   : Y1 = RAD*SIN(RANG)+10
2060 X2 = RAD*COS(RANG2)+10  : Y2 =
   RAD*SIN(RANG2)+10
2070 GOSUB 4000
2080 NEXT ANG%
```



```
2000 ' Plot curve
2010 X1 = 0 : Y1 = 10 : X2 = 20 : Y2 = 10 : GOSUB
   4000
2020 X1 = 10 : Y1 = 0 : X2 = 10 : Y2 = 20 : GOSUB
   4000
2030 X1 = 0 : Y1 = 10 : FOR X2 = 0 TO 20 STEP .2
2040 Y2 = 10 - 9 * SIN(3.14159  * X2/10) : GOSUB
   4000
2050 NEXT X2
```



Using Delta for business graphics

You don't have to be a mathematician, scientist, or computer hacker/artist to use Delta's graphics capabilities. It can be used for business graphics too—line graphs, bar charts, pie charts, and more! There are many commercially available graphics programs that support Delta's graphics. And, of course, you can write your own. To get you started, we've written a program that prints a pie chart. Here it is:

```
100 ' PIECHART
110 ESC$ = CHR$(27)     : LF$ =CHR$(10)
120 FF$ = CHR$(12)      : VTAB$ = CHR$(11)
130 EMPHASIZED$ = ESC$ + "E"    : NOT.EMPHASIZED$ =
   ESC$ + "F"
1000 ' Set program constants
1010 OPEN "LPT1:" AS #1 : WIDTH #1,255
1020 DIM BIT%(190,36),A$(36),PCT%(25),TEXT$(42),
   PIECETEXT$(25)
1030 MASK%(1) = 64      : MASK%(4) = 8
1040 MASK%(2) = 32      : MASK%(5) = 4
1050 MASK%(3) = 16      : MASK%(6) = 2
1060 LX = 20            : LY = 20
1070 LXFAC = 190/LX     : LYFAC = 216/LY
1080 FOR I= 1 TO 42
1090 TEXT$(I) = SPACE$(80)
1100 NEXT I
1110 GOSUB 7000
2000 ' Plot curve
2010 RAD = 9
2020 X1 = 19            : Y1 = 10
2030 PRINT " ";
2040 FOR ANG% = 0 TO 360 STEP 5
2050 RANG = ANG%*6.28/360
2060 X2 = RAD*COS(RANG)+10  : Y2 = RAD*SIN(RANG)+10
2070 GOSUB 4000
```

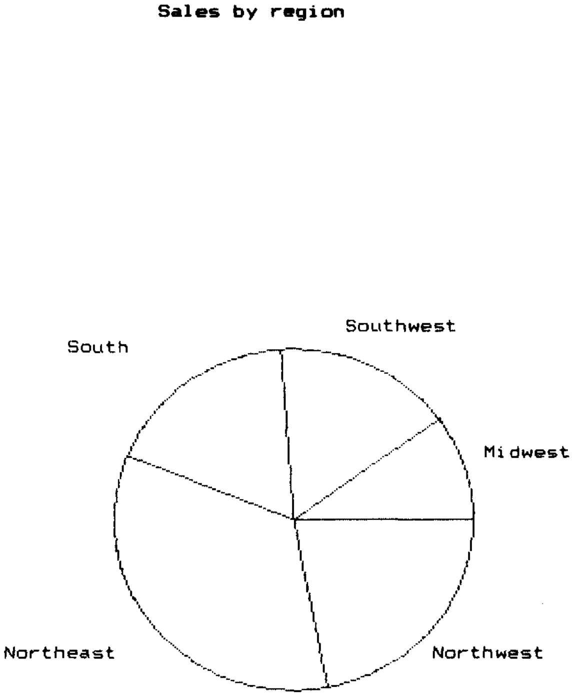
```
2Ø8Ø NEXT ANG%
2Ø9Ø FOR PIECE% = 1 TO NUMBER.PIECES%
21ØØ X1 = 1Ø                : Y1 = 1Ø
211Ø TOTAL.PCT%=TOTAL.PCT%+PCT%(PIECE%)
212Ø ANG%=36Ø*TOTAL.PCT%*.Ø1
213Ø RANG = ANG%*6.28/36Ø
214Ø X2 = RAD*COS(RANG)+1Ø  : Y2 = RAD*SIN(RANG)+1Ø
215Ø GOSUB 4ØØØ
216Ø GOSUB 6ØØØ
217Ø NEXT PIECE%
3ØØØ ' Send bit image map to printer
3Ø1Ø PRINT
3Ø2Ø FOR ROW% = Ø TO 35
3Ø3Ø A$(ROW%) = ""
3Ø4Ø FOR COL% = 1 TO 19Ø
3Ø5Ø A$(ROW%) = A$(ROW%) + CHR$(BIT%(COL%,ROW%))
3Ø6Ø NEXT COL%
3Ø7Ø PRINT CHR$(176);CHR$(176);
3Ø8Ø NEXT ROW%
3Ø9Ø PRINT
3Ø91 PRINT#1,SPACE$(4Ø-LEN(TITLE$)/2);EMPHASIZED$;
   TITLE$;NOT.EMPHASIZED$;LF$
31ØØ PRINT#1,VTAB$;VTAB$;VTAB$
311Ø PRINT#1,ESC$;"A";CHR$(6)
312Ø PRINT#1,TEXT$(1);LF$;TEXT$(2);LF$;TEXT$(3);LF$
313Ø FOR ROW% = Ø TO 35
314Ø PRINT#1,"          ";ESC$;"K";
   CHR$(19Ø);CHR$(Ø);
315Ø PRINT#1,A$(ROW%)
316Ø PRINT#1,TEXT$(ROW%+4);LF$
317Ø PRINT CHR$(176);CHR$(176);
318Ø NEXT ROW%
319Ø PRINT#1,TEXT$(4Ø);LF$
32ØØ PRINT#1,TEXT$(41);LF$
321Ø PRINT#1,TEXT$(42);LF$
322Ø PRINT#1,ESC$;"2";FF$
323Ø END
4ØØØ ' Draw a line from X1,Y1 to X2,Y2
4Ø1Ø XL = X2 - X1       : YL = Y2 - Y1
4Ø2Ø NX = ABS(XL*LXFAC) : NY = ABS(YL*LYFAC)
4Ø3Ø IF NX < NY THEN NX = NY
4Ø4Ø NS% = INT(NX+1)
4Ø5Ø DX = XL/NS%        : DY = YL/NS%
4Ø6Ø FOR I% = 1 TO NS%
4Ø7Ø X1 = X1 + DX       : Y1 = Y1 + DY
```

```
4080 GOSUB 5000
4090 NEXT I%
4100 PRINT CHR$(29);CHR$(205);CHR$(175);
4110 RETURN
5000 ' Plot a point at X1,Y1
5010 XX = X1 * LXFAC    : YY = Y1 * LYFAC
5020 COL% = INT(XX) + 1
5030 ROW% = INT(YY/6)
5040 XIT% = INT(YY - ROW% * 6)+1
5050 BIT%(COL%,ROW%) = BIT%(COL%,ROW%) OR
   MASK%(XIT%)
5060 RETURN
6000 ' Place text fields in the proper location
6010 MIDANG%=(ANG%+PREVANG%)/2
6020 RANG = MIDANG%*6.28/360
6030 X3 = INT(20*SIN(RANG))  : Y3 =
   INT(22*COS(RANG))
6040 X4 = 22 + X3      : Y4 = 40 + Y3
6050 IF MIDANG%>270 OR MIDANG<90 THEN
   MID$(TEXT$(X4),Y4) = PIECETEXT$(PIECE%) ELSE
   MID$(TEXT$(X4),Y4-LEN(PIECETEXT$(PIECE%)))=
   PIECETEXT$(PIECE%)
6060 PREVANG%=ANG%
6070 RETURN
7000 ' Accept Data from Screen
7010 CLS: PRINT : PRINT : PRINT :
7020 INPUT "ENTER TITLE FOR CHART: ",TITLE$
7030 AMT.SOFAR%=0       : AMT.LEFT%=100
7040 FOR I=1 TO 24
7050 CLS: PRINT "      ENTER PARAMETERS FOR
   PIE-CHART"
7060 PRINT "  TOTAL SO FAR :   ";
7070 PRINT USING "###";AMT.SOFAR%
7080 PRINT "  TOTAL REMAINING: ";
7090 PRINT USING "###";AMT.LEFT%
7100 PRINT :PRINT :PRINT :PRINT
7110 INPUT "ENTER PERCENTAGE FOR FIELD:   ",PCT%(I)
7120 IF PCT%(I)>AMT.LEFT% OR PCT%(I)=0 THEN
   PCT%(I)=AMT.LEFT%
7130 AMT.LEFT%=AMT.LEFT%-PCT%(I)
7140 AMT.SOFAR%=AMT.SOFAR%+PCT%(I)
7150 PRINT :PRINT
7160 INPUT "ENTER DESCRIPTION OF FIELD:   ",
   PIECETEXT$(I)
```

```
7170 IF LEN(PIECETEXT$(I))>15 THEN PRINT "FIELD TOO
   LONG - 15 CHAR. MAX": GOTO 7160
7180 IF AMT.LEFT%=0 GOTO 7200
7190 NEXT I
7200 NUMBER.PIECES%=I
7210 IF NUMBER.PIECES%=1 THEN 7030
7220 CLS
7230 RETURN
```

You should recognize many sections of code from the plotting program. We've just expanded on that program framework to include routines for inputting data to be graphed (starts at line 7000) and placing labels next to the pie chart (starts at line 6000).

The output from our program is shown below.



High Resolution Graphics

Up until now all of the dot graphics printing we have done has been with Delta's normal density mode. This can give you some pretty sharp images at great speed. Sometimes though, you may want to create an image with even higher resolution. Delta has four density modes you can use; they're summarized in Table 8-2.

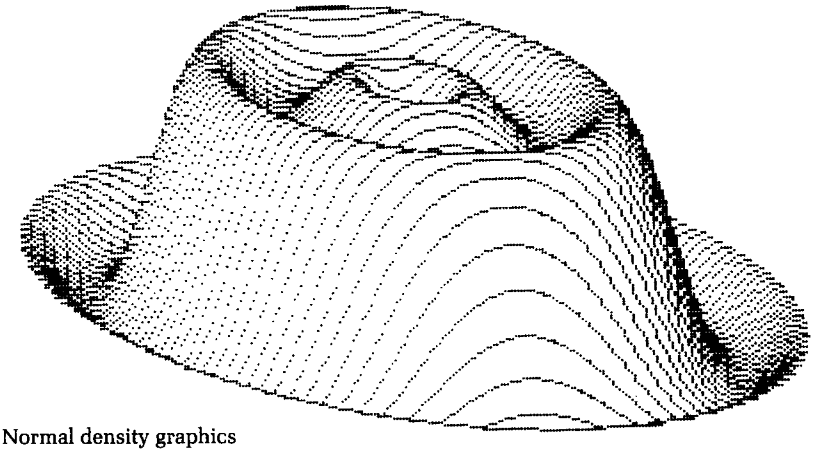
Table 8-2
Dot graphics commands

Function	**Control code**
Normal density (60 dots/inch)	⟨ESC⟩ "K" *n1 n2 m1 m2* . . .
Double density (120 dots/inch)	⟨ESC⟩ "L" *n1 n2 m1 m2* . . .
Double density/double speed	⟨ESC⟩ "y" *n1 n2 m1 m2* . . .
Quadruple density (240 dots/inch)	⟨ESC⟩ "z" *n1 n2 m1 m2* . . .

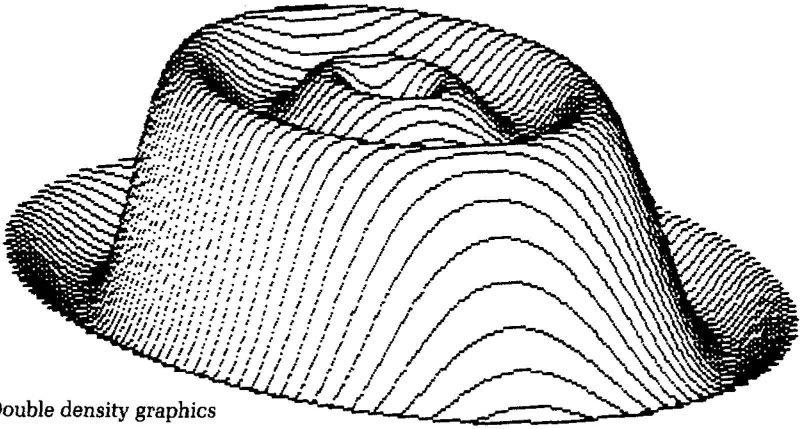
Note: If your computer does not support lowercase characters, use CHR$(121) and CHR$(122) for "y" and "z", respectively.

The command syntax for all of the commands is the same—just as you have learned it for the ⟨ESC⟩ "K" (normal density) command. The number of columns to be printed is *n1* + 256∗*n2*.

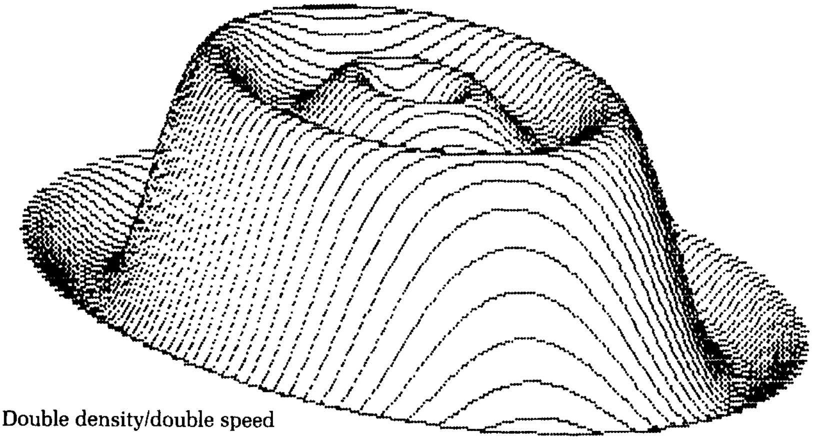
So what do these different modes do? On the following pages are actual size reproductions of printouts of the same image printed in each of the four different densities. They were all printed using the plotting program in this chapter (with a rather complex set of formulas starting at line 2000!).



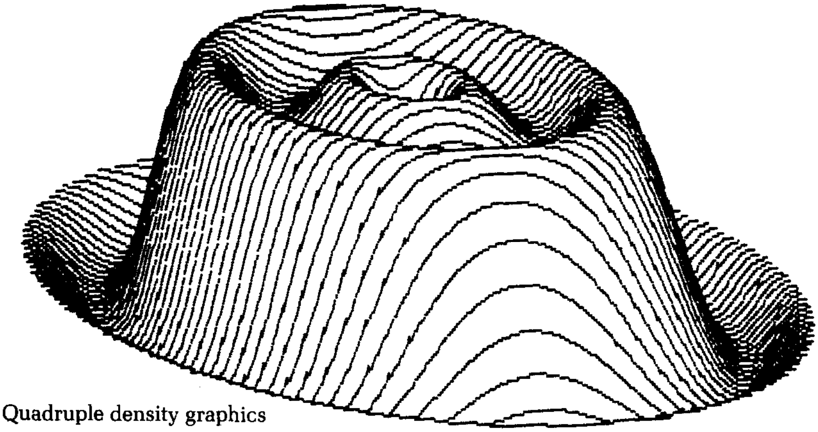
Normal density graphics



Double density graphics



Double density/double speed



Quadruple density graphics

So if quadruple density looks so great, why not use it all the time? Let's try an experiment on your printer which will show just how the different density modes work. Using the first program in this chapter, change line 30 to try each of the different modes. Just change the CHR$(75) (which is the ASCII code for "K") to "L", "y", and "z" in turn. Your printouts should look something like this:

⟨ESC⟩"L"

⟨ESC⟩"y"

⟨ESC⟩"z"

As you can see, the different modes seem to condense the printed image. So, to get the same image in a higher density mode, you must plot more points. This requires twice as much memory for your array, twice as much computing time, and twice as much printing time (but the results may be worth it!).

Star's engineers have given programmers a unique shortcut for program development though—double density double speed mode. Although this mode requires just as much memory and computing time as double density, it prints at the same speed as normal density graphics. Amazing, you say? Well, it is—until you know the secret. Every other column of dots is ignored, so the output is actually the same as normal density graphics. The advantage is that you can write and debug your programs at double speed, then change to double density graphics for terrific output.

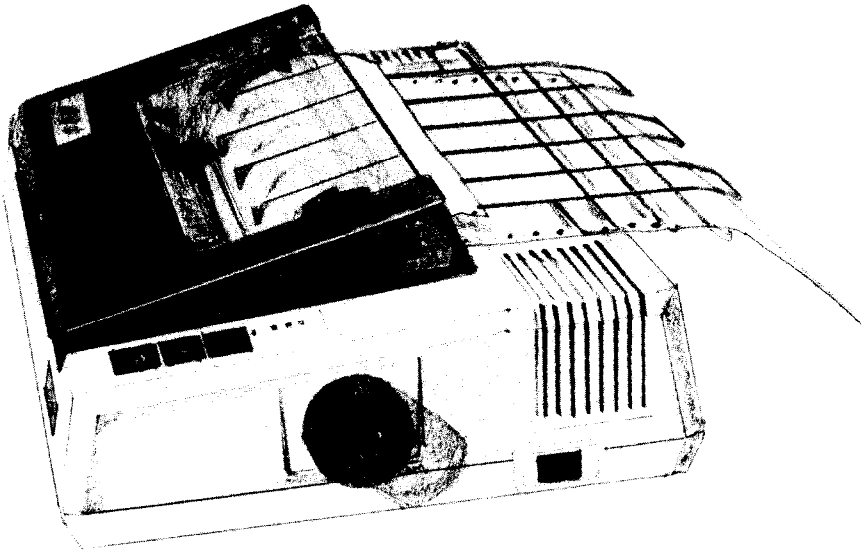
If You Have Problems with BASIC

You may write some graphics programs that look just right in the listing, but the printouts aren't quite what you expected. A common problem is that the BASIC interpreter in your computer is inserting a few of its own codes. For instance, if your program generates a CHR$(13) as valid graphics data, BASIC may follow it with a CHR$(10). Another problem arises with certain computers that replace horizontal tabs (CHR$(9)) with a series of spaces (CHR$(32)). A possible solution to these problems is to not use the bottom dot (which has a value of 1). This way, you will never produce an odd number, hence, you will never have a CHR$(13) or CHR$(9). (This is why we used only six pins in our plotting program.)

That's one solution to one problem. You'll find more of each (with specific information for *your* computer) in the appropriate appendix.

Summary

Control code	**Function**
⟨ESC⟩ "K" *n1 n2 m1 m2* . . .	Print *n1* + 256∗*n2* columns of normal density graphics
⟨ESC⟩ "L" *n1 n2 m1 m2* . . .	Print double density graphics
⟨ESC⟩ "y" *n1 n2 m1 m2* . . .	Print double density graphics at double speed
⟨ESC⟩ "z" *n1 n2 m1 m2* . . .	Print quadruple density graphics



Chapter 9

Getting it All Together

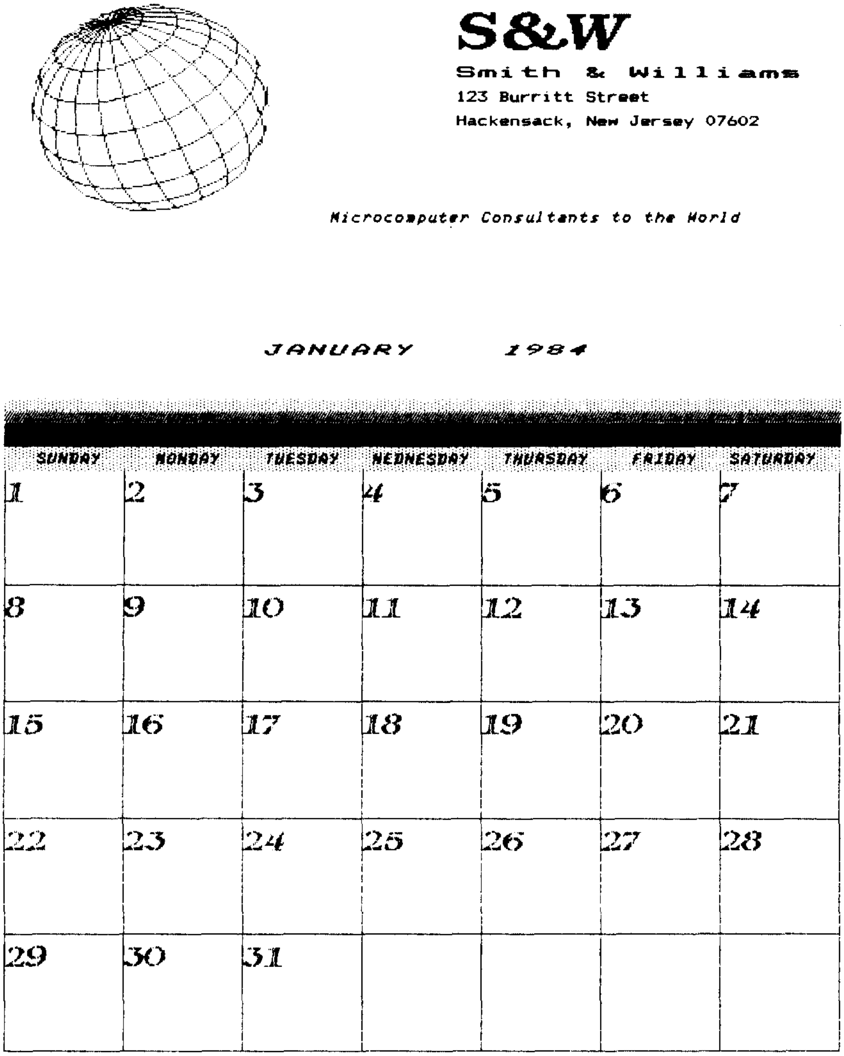
You have now seen how all of Delta's functions work. You surely have some good ideas of ways to use some of Delta's many capabilities. With all the different printing styles available you won't run out of variations for quite a while. And if there is a style of printing that Delta doesn't have built-in, you can develop your own with Delta's download characters.

Four different modes of graphics provide you with limitless ways to create pictures with Delta. You have learned how to create both stored-data and calculated shapes using Delta's graphics capabilities.

The calendar on the following page is a demonstration of just some of Delta's abilities. You will recognize many of the things that you have learned while reading this book. The globe was created the same way as the calculated-shape graphics that you learned about. The S&W logo was used to illustrate stored-data graphics. And you are sure to recognize many of the print styles used at the top of the calendar. Those shaded bars are created by using different graphics densities.

The numbers in the calendar itself are the ones that you created with download characters. And the lines creating the boxes are made with Delta's block graphics characters.

Delta's flexibility in line spacing and its ability to mix many types of printing on one line make it possible to create complex forms like this calendar. With Delta's advanced features it's easy to create a business form or letterhead, and fill it out at the same time. That's a productive printer.



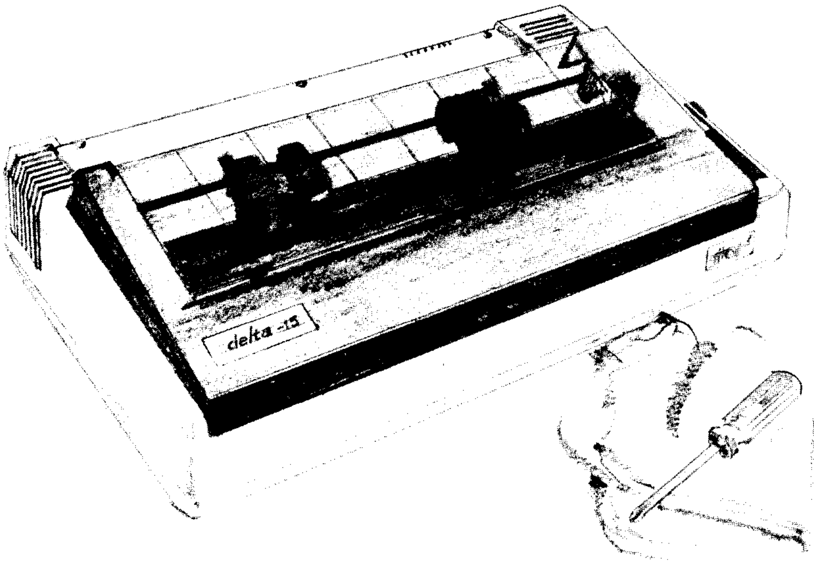
S&W

Smith & Williams
123 Burritt Street
Hackensack, New Jersey 07602

Microcomputer Consultants to the World

JANUARY 1984

SUNDAY	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY	SATURDAY
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				



Chapter 10

Maintenance

As almost any good mechanic will tell you, dust and heat are the prime enemies of any mechanism, and Delta is no exception. The best maintenance is *preventive.* So, to start with, we hope you've found a clean, dust-free location with a comfortable temperature range for your computer printer system, *and* that you'll keep the printer's dust-cover where it belongs — in its place atop the printer! Appendix A gives you further tips on locating Delta.

Cleaning Delta

The second rule for long life is *periodic cleaning.* Both inside and outside of the case respond gratefully to periodic cleaning with a damp rag and alcohol. Do this whenever the case appears to be getting dirty, always being careful to avoid dripping alcohol on the printer mechanism. To clean the *inside* of dust and paper lint, use a nice soft brush, after first removing the printer's upper case in the following manner.

Removing the Upper Case

First, turn the power switch *off* and unplug Delta.

Remove the tractor unit.

Remove the platen knob.

Next, remove the two Phillips screws, per Figure 10-1. (You'll find *three* screws on the larger Delta-15.)

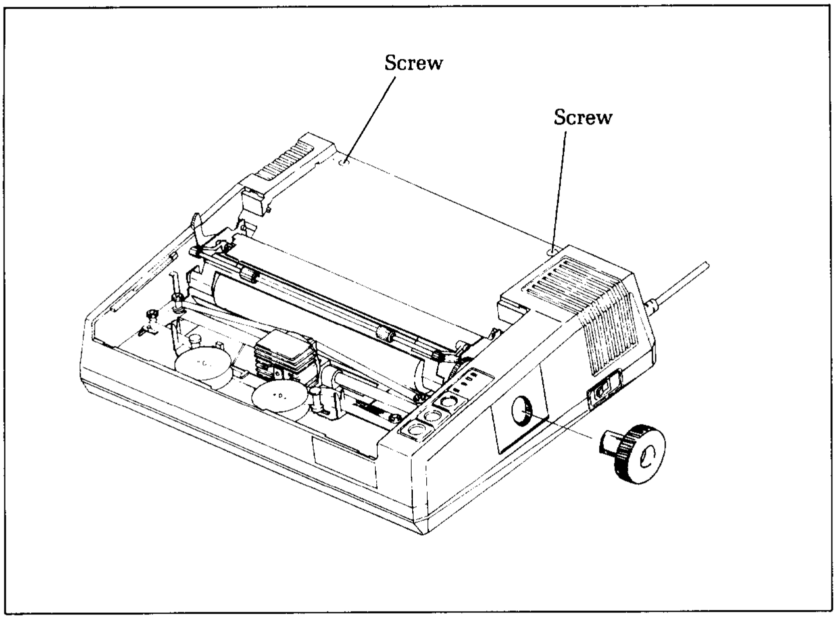


Figure 10-1. *To remove Delta's upper case, remove the screws.*

Lift up the upper case from the back; then "pop out" the front of the case by pushing outward with your thumbs where the plastic tabs fit into the bottom case.

Then stand the upper case on its end, right next to the printer so the short colored wires attached to the control panel remain connected. It's a good idea to keep the case from falling over by setting the corner of the printer on the edge of the case as shown in Figure 10-2.

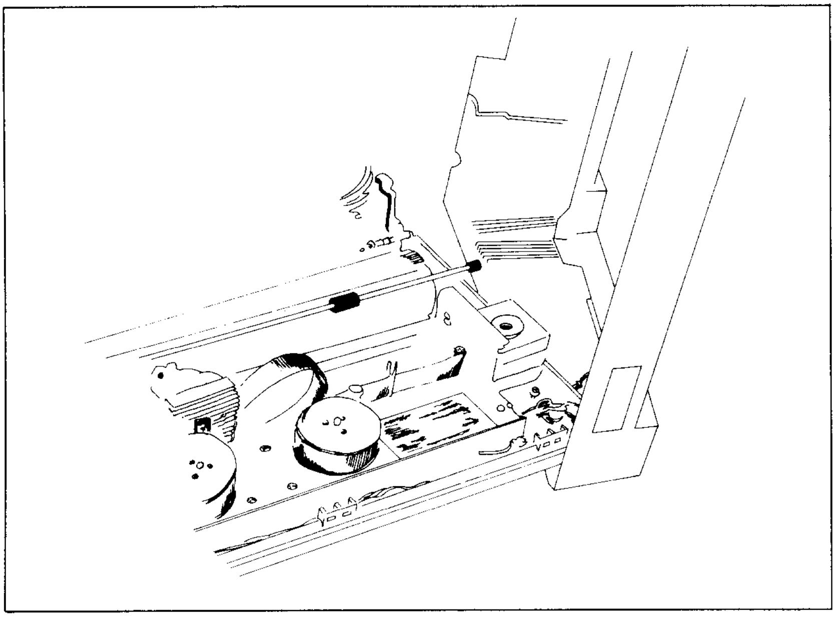


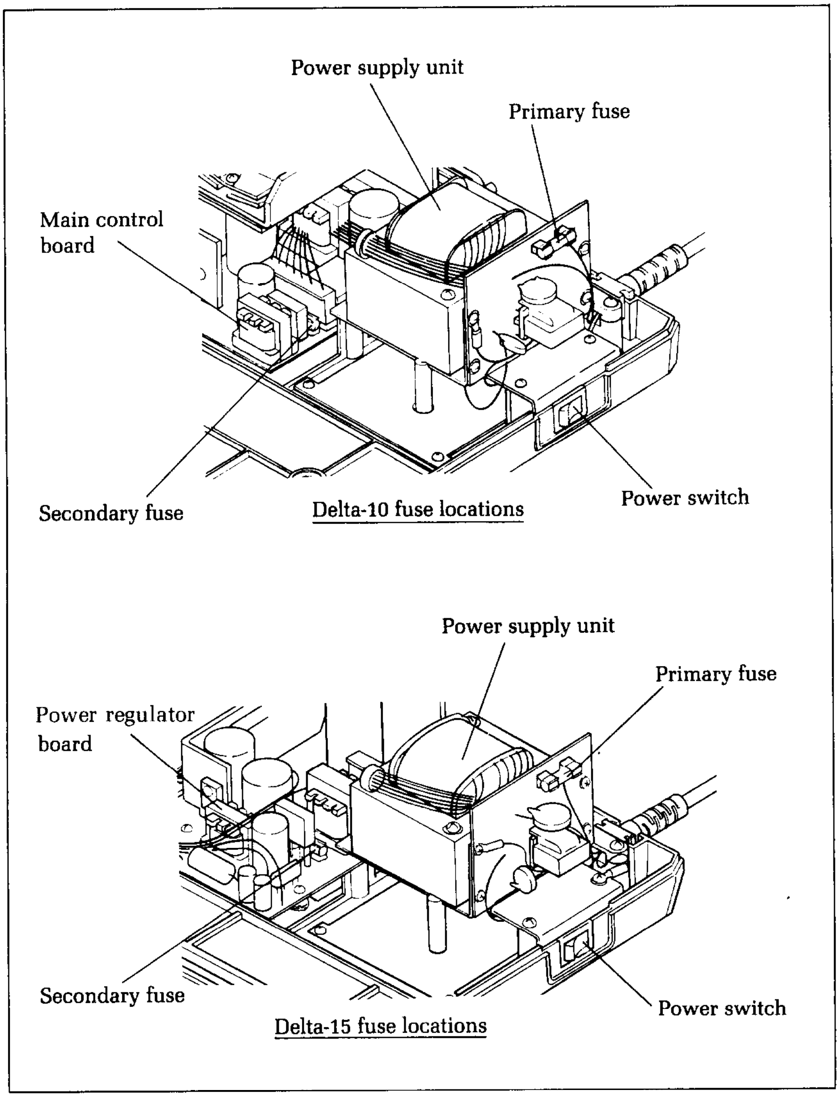
Figure 10-2. *Tilt the cover up so that you don't have to disconnect the control panel wires.*

NOTE TO THE UNWARY: When cleaning the inside of Delta, be *very* careful not to bend or injure any of those tiny wires or other little gizmos that lie exposed and defenseless against a heavy-handed touch.

Besides the periodic cleanings, the only other maintenance you'll encounter will be changing the ink ribbon (see Appendix A for these directions), and replacing an occasional blown fuse or replacement of the print head after a long period of use.

Replacing a Fuse

How can you tell when you've blown a fuse? Well, when the printer won't operate and the power lamp on the control panel isn't lit, even though you're sure the power switch is on and the printer is plugged in — it's likely a blown fuse.



Delta-10 fuse locations

Delta-15 fuse locations

Figure 10-3. *The secondary fuses are a little tricky to reach.*

To check out the fuses (there are two), turn the power switch *off*, and disconnect the power plug. Then remove the platen knob and the upper case, as described in the preceding section.

When the case is off, check for the correct fuse locations on Figure 10-3. Remove the primary fuse first, held in its clamps near the power switch. The fuse is a commonly used type, with a metal strip suspended in a glass and metal case. If the strip is broken, the fuse is blown.

Replace this fuse with a 2A/125V slow-blow type fuse; then test-run the printer. Still immobile? Check the secondary fuse, which is still further inside the "works," as shown in Figure 10-3. If *that's* blown, replace it with a 3A/125V slow-blow type fuse.

Then if the printer still isn't working, better call on your Delta dealer for help.

Incidentally, it makes it easier to remove and replace the fuses if you have a long, flat screwdriver and some needlenose pliers handy. The fuses are small, and the secondary fuse is sort of hard to get at without these particular tools. (But be careful not to crush the glass fuses with the pliers.)

We've found the following technique works pretty well for replacing the secondary fuse. Using the long flat screwdriver, pop one end of the fuse loose from its holding clamp and stand it on end. Then grab the loose end with the needle nose pliers and pull it out. To insert the new fuse, lower it into place with the pliers, so that it lies flat against the two holding clamps. Then use the long screwdriver to push each metal end of the fuse into its adjacent holding clamp.

Replacing the Print Head

As mentioned earlier, the print head has a long life, printing perhaps 100,000,000 characters before it wears out. You'll know when that happens when the printout is too faint for your taste even after replacing the ink ribbon.

WARNING: The print head gets hot during operation, so let it cool off for awhile, if necessary, to avoid burning your fingers.

To replace the print head, start by removing the printer cover and the ink ribbon. Then unplug the print head cable (see Figure 10-4) while holding down the head cable board. Next, remove the two screws and washers which fasten the print head. Then place the new print head in position, and attach it with the same two screws. Apply "screw lock," (an adhesive available at hardware stores) to the heads of the screws. Finally, insert the head cable, with the printed side up, tightly into the head connector on the cable board. Be sure that it's a good solid connection, or it could cause problems.

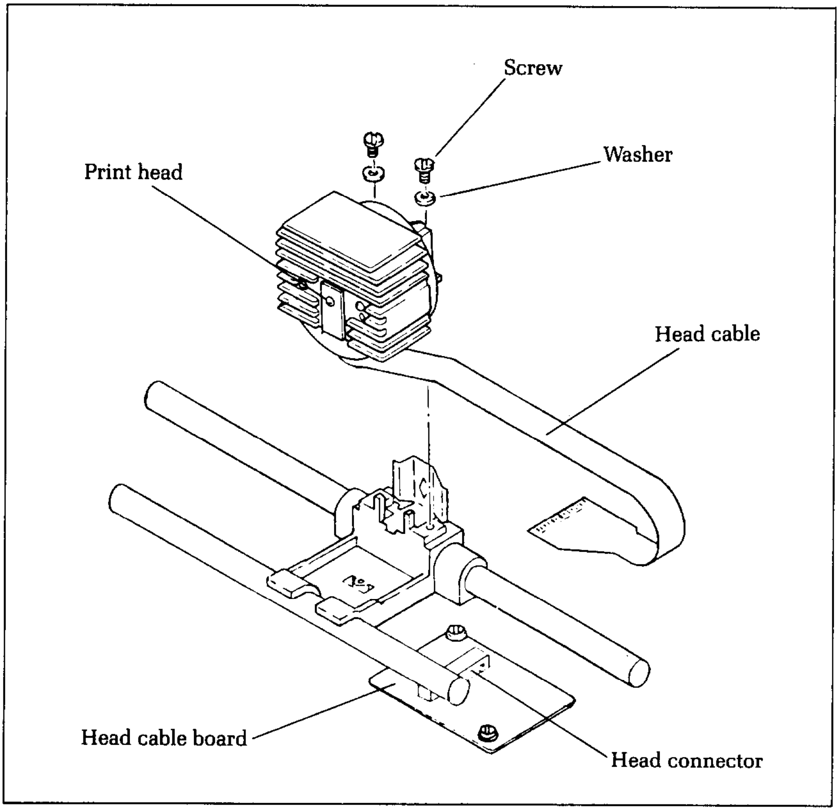
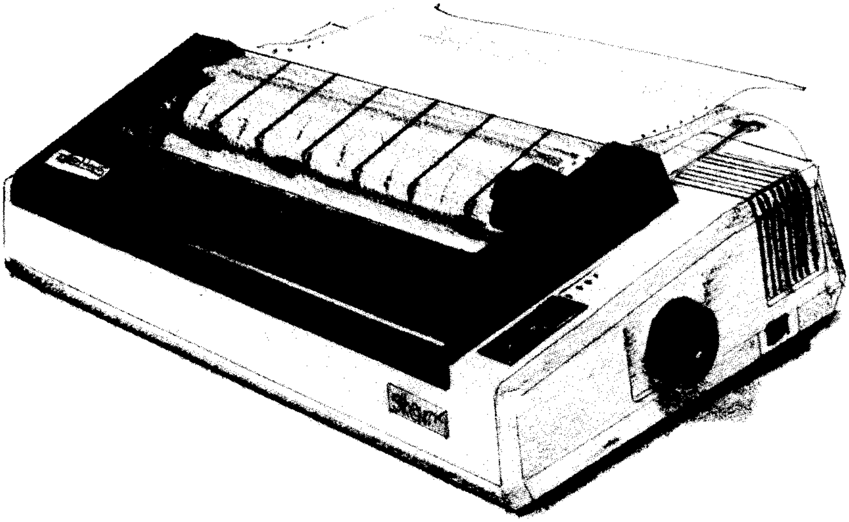


Figure 10-4. *Replacement of Delta's print head is simple.*

Back to perfect printing!



Appendix

Appendix A

Setting Up Delta

In this appendix, we'll show you how to unpack your new Delta printer, set it up in the right location, and get it ready for you to load it with paper and start printing! But first . . .

Where shall we put it?

Before you do anything else, give some thought to where you'll be using your printer. Obviously, it will be somewhere near your computer. And both computer and printer will lead longer, healthier lives if they like their environment. For instance, we recommend . . .

- Placing the printer on a flat surface
- Keeping it out of direct sunlight and away from heat-producing appliances
- Using it only in temperatures where you are comfortable
- Avoiding areas with a lot of dust, grease, or humidity
- Giving it "clean" electricity. Don't connect it to the same circuit as large, noise-producing motors
- Providing the right voltage, which is not over 10% more or less than 120 volts AC.

WARNING: Extremely high or low voltage can damage your printer.

What have we here?

Let's take a look at what's in the box. Take it slow and easy, and check each item in the box against Figure A-1. There should be exactly 11 items. One important item is the printer registration and warranty card. Now is the time to fill it in and mail it. It's a good warranty, and you'll like the protection it gives you.

Let's move on to the next step . . .

Removing the shipping screws

There are two shipping screws on the bottom of the printer,

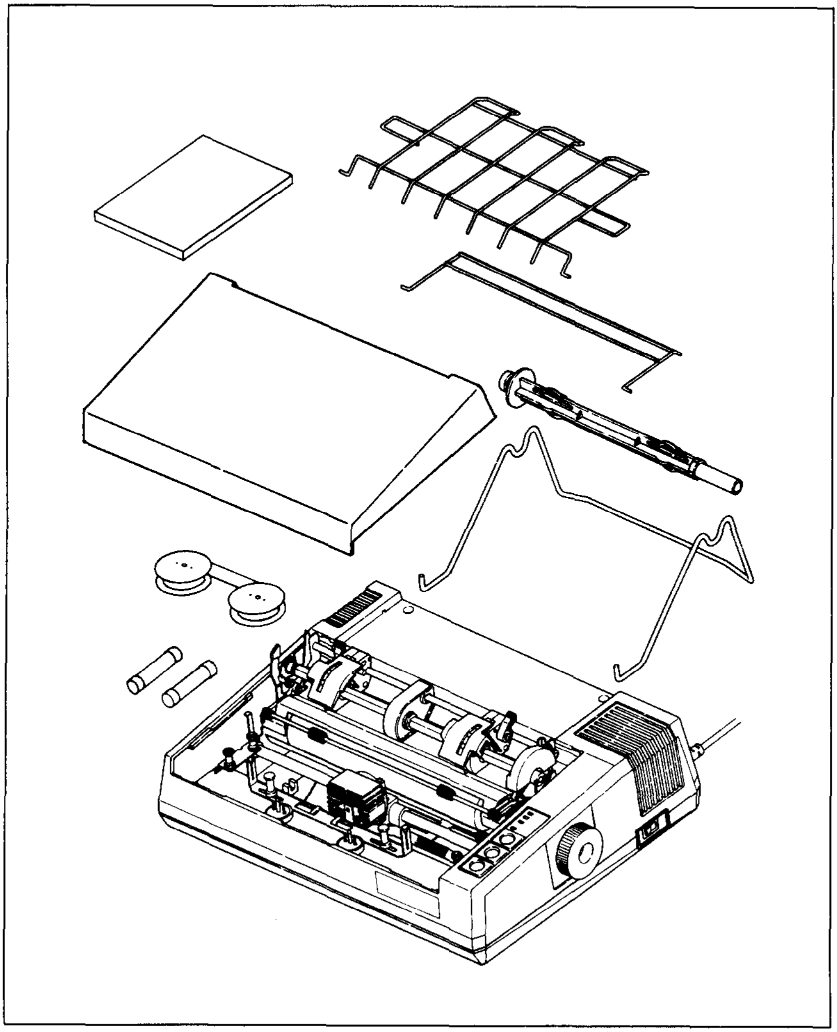


Figure A-1. *Inside the carton you should find: 1) printer, 2) printer cover, 3) paper guide, 4) paper separator, 5) roll paper holder, 6) roll paper shaft, 7) ink ribbon, 8) platen knob, 9) spare fuses, 10) User's Manual, 11) warranty card.*

used to hold the internal chassis to the external frame during shipping. To get at these, carefully place the printer upside down on a soft surface like a foam cushion. Remove the two screws with a Phillips screwdriver as shown in Figure A-2.

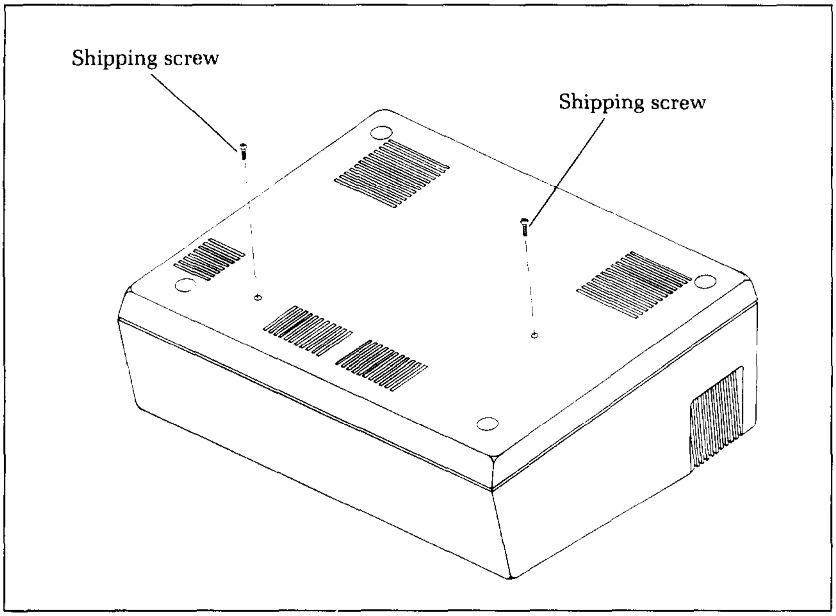


Figure A-2. *Remove the shipping screws before using Delta.*

Removing the packing from inside the printer

Remove three cardboard pieces: a large flat piece protecting the print head, and two smaller pieces stuck in at either end of the platen (the rubber cylinder that feeds the paper through the printer).

You'll want to save the screws, along with the rest of the packing material and the shipping box, in case you ever have to ship the printer. Tape the screws somewhere on the box or packing. (You *did* fill in that registration card, didn't you?)

Installing the platen knob

This is the knob that turns the rubber platen cylinder. It fits into the hole on the right side of the printer case. Just match the odd-shaped hole in the knob with the same shape on the shaft you'll see inside the hole in the case, and press on firmly. Give the knob a few twirls to see that it's turning the platen easily and smoothly.

Removing the tractor unit

The tractor unit, shown in Figure A-4, comes mounted on the printer during shipment. It is used only with sprocket-feed paper. When other papers are used, such as single sheets or roll paper,

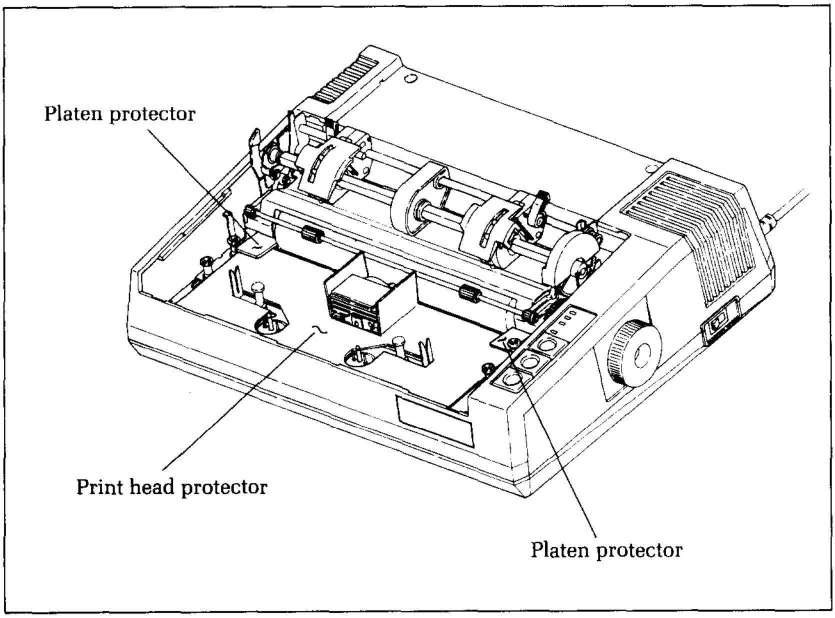


Figure A-3. *There are three pieces of cardboard to remove.*

the unit should be removed, in the following manner:

Remove the printer cover (if attached).

Identify the "snap levers" as shown in Figure A-4.

Pull both snap levers forward, and at the same time . . .

Rock the tractor unit up and towards you about half an inch.

Now lift the tractor up and away from the printer.

Up to this point, we've been clearing the decks for action, so to speak. Only two more things are left to do before we can start printing. They are, 1) attach the paper separator and paper guide racks, and 2) install the ink ribbon. Actually, if you're planning to print on single sheets only, you won't need to use the paper separator and paper guide, which are designed expressly to guide continuous paper (roll or sprocket-feed) through the printer.

Attaching the paper separator and paper guide

First, identify the paper separator (the large metal rack), shown in Figure A-5. Insert one protruding end into the hole shown in the diagram, then gently bend the other protruding end until it snaps into the opposite hole. Follow the same procedure with the smaller metal rack, which is the paper guide.

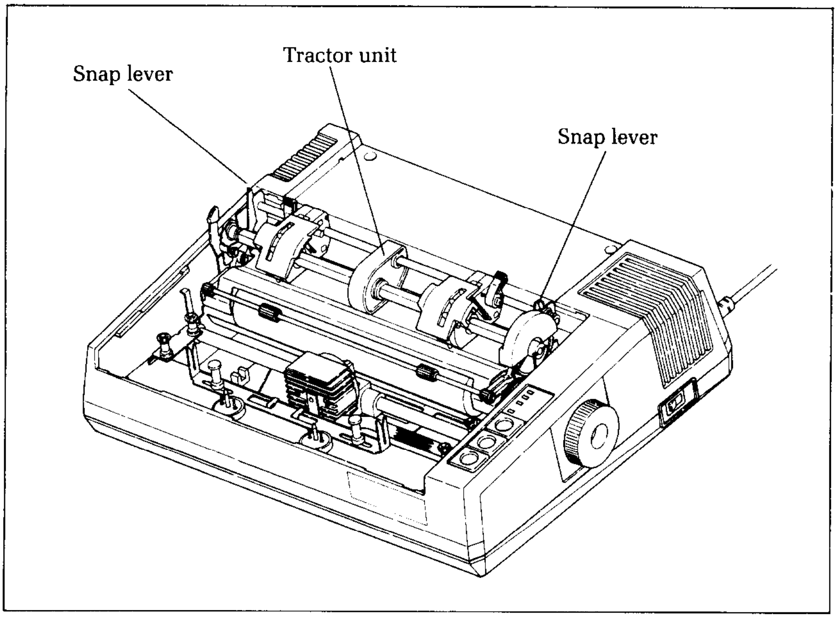


Figure A-4. *Remove the tractor by pulling the snap levers towards you and tilting the tractor unit back.*

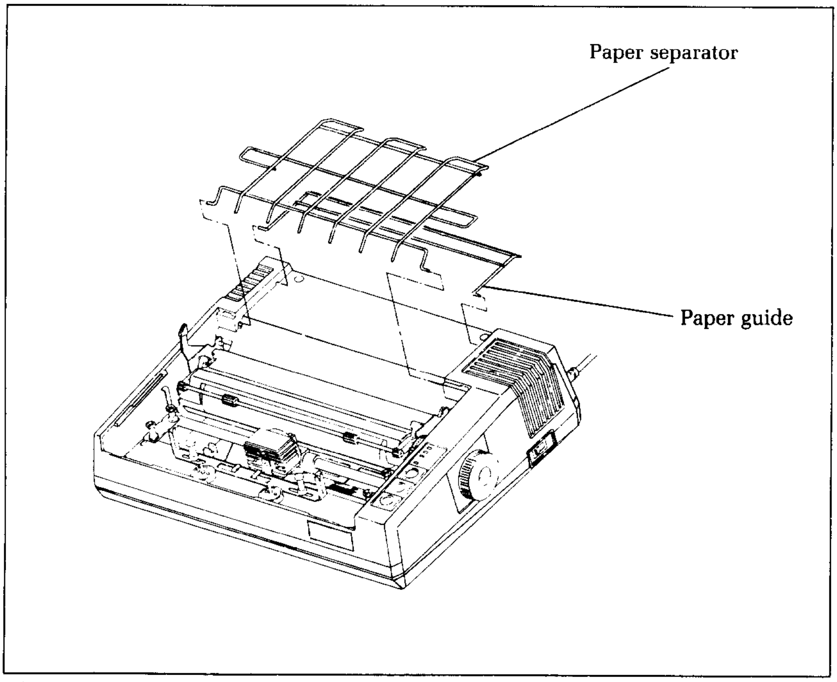


Figure A-5. *Attach the paper guide and paper separator.*

Important news: If you get these in upside down, they won't work. So take another sharp look at Figure A-5 before we pass on to the final act—installing the ink ribbon.

(NOTE: If you're wondering about the wire roll paper holder and holder shaft, we'll explain these in Chapter 1, where we discuss the whole subject of paper selection, paper feed, and related topics.)

Installing the ribbon

Telling you how to set the ribbon is like writing a set of instructions on how to tie your shoelaces. It takes a lot longer to describe it than it does to do it! So, you'll be smart to study the several figures shown here; they'll tell you all you really need to know.

Nevertheless, if you feel better following written instructions, read on . . .

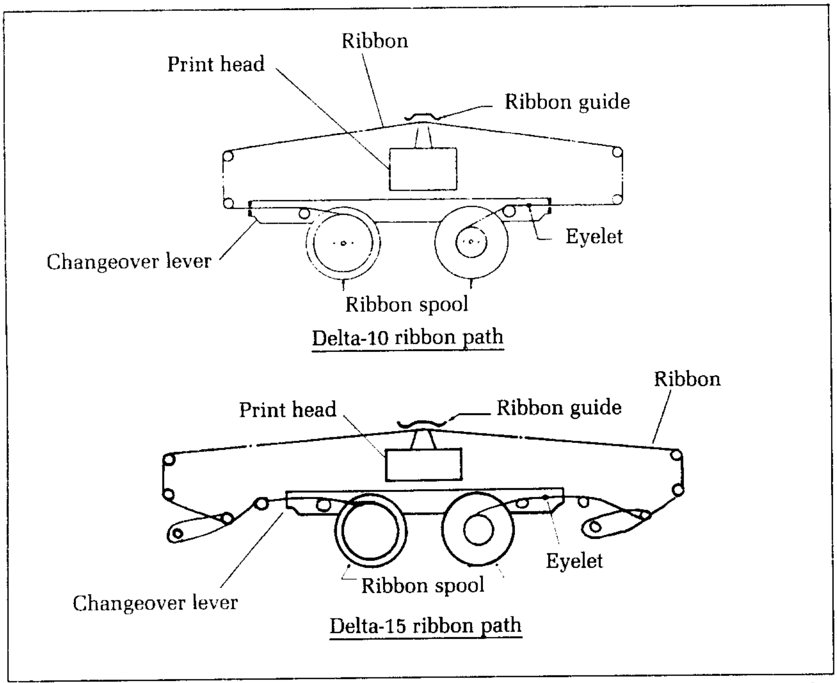


Figure A-6. *You'll find this diagram of the ink ribbon path inside your Delta for easy reference when you change ribbons.*

After looking carefully at the Figure A-6, begin by turning the power *off*, and removing the printer cover. Then slide the print

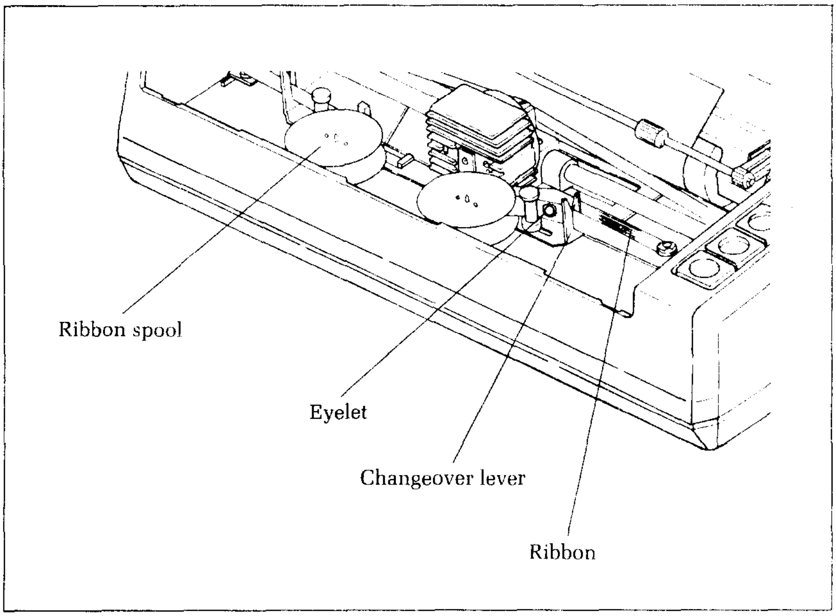


Figure A-7. *The eyelet should be between the ribbon spool and the changeover lever.*

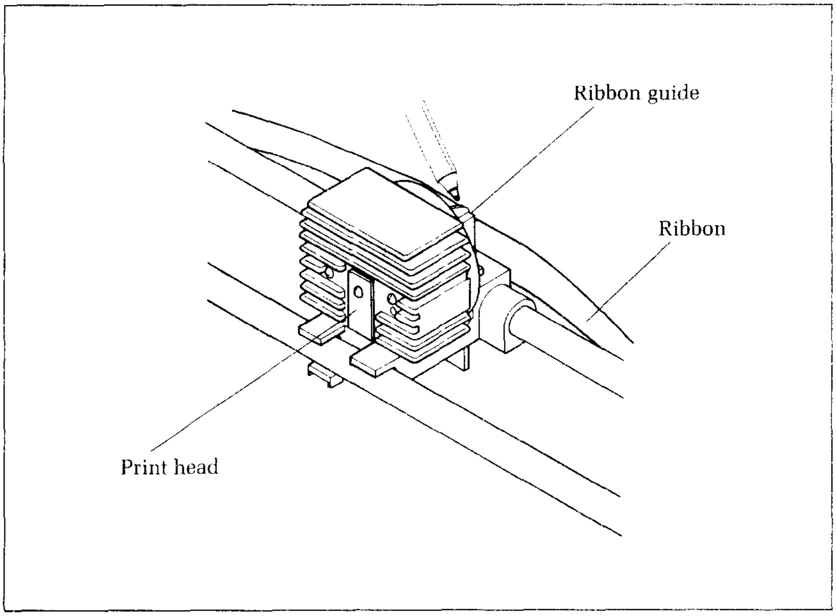


Figure A-8. *You may want to use a ballpoint pen to press the ribbon guide towards the platen and work the ribbon down beside it.*

head gently to the center of the printer. Next, set one of the ribbon spools on a ribbon spool post, being careful to have it wind/unwind exactly as shown in the figures. The spool should fit so that the drive pins engage the spool.

Next, you start threading the ribbon. The only tricky part is around the print head, so pay particular attention to Figure A-8. The ribbon should fit in the slot between the print head and the ribbon guide. Use a ballpoint pen to work the ribbon down between the print head and the ribbon guide.

The other thing to watch is the position of the eyelet near the end of the ribbon. This works exactly like a typewriter ribbon eyelet; it can't get by the slotted guide, which causes the ribbon to automatically reverse its direction.

Finally, set the other spool snugly on the opposite spool holder; then turn the spools by hand four or five turns in each direction to verify that everything is properly set and ready to roll.

Installing the printer cover

Now that you've completed the steps described in this Appendix A, you may want to mount the printer cover in place to keep dust and dirt away. It's a simple procedure. Merely fit the two tabs at the back edge of the cover into the two slots right next to the two holes where you fastened the wire rack paper separator. Then drop the other end down gently until it sits firmly on the printer. That's all!

To remove the cover just reverse the process: lift up the front and pull it out of the slots at the back.

Connecting Delta to your computer

To complete the installation, you'll need to connect Delta to your computer. In Appendixes B through G, we've described the procedure, including specific guidelines for making connections with several of the most popular computers used by Delta owners.

Appendix B

IBM Personal Computer and Compaq Computer

Both the IBM Personal Computer and the Compaq computer function the same when connected to Delta. We will discuss the IBM-PC, knowing that all we say works just as well for the Compaq.

Delta can connect to either a serial or a parallel interface in the IBM-PC or IBM-XT computers. IBM calls a parallel interface a "Parallel Printer Adapter", and they call a serial interface an "Asynchronous Communications Adapter."

You only need a cable to connect Delta to your IBM-PC. Your Delta dealer can furnish this cable, or you can use a standard IBM-PC parallel printer cable for the parallel interface.

We recommend that you set the DIP switches in Delta as shown below when connecting it to an IBM-PC parallel interface.

Recommended DIP Switch Settings for IBM-PC

Switch	Setting	Function
1-1	ON	11 inch page size
1-2	ON	Normal print density
1-3	ON	10 CPI pitch
1-4	ON	Normal characters
1-5	ON	1/6 inch line feed
1-6	ON	U.S.A. Character set
1-7	ON	
1-8	ON	
2-1	ON	Paper-out detector active
2-2	OFF	Parallel interface
2-3	OFF	8-bit interface
2-4	OFF	No auto line feed

IBM-PC Parallel Cable

Delta		IBM-PC Parallel	
Pin No.	**Function**	**Pin No.**	**Function**
1	STROBE	1	STROBE
2	D1	2	D0
3	D2	3	D1
4	D3	4	D2
5	D4	5	D3
6	D5	6	D4
7	D6	7	D5
8	D7	8	D6
9	D8	9	D7
10	ACK	10	ACK
11	BUSY	11	BUSY
12	PAPER END	12	PAPER END
13	SELECTED	13	SELECT
16	GROUND	18-25	GROUND
31	RESET	16	RESET
32	ERROR	15	ERROR

The IBM-PC expects its printer to be connected to the parallel interface. If you are using the serial interface, then you will need to instruct your computer to send information to the serial interface instead of to the parallel interface. This is done with the MODE command. You must use the following two commands each time you turn on your computer.

```
MODE COM1:48,N,8,1,P
MODE LPT1:=COM1:
```

The first "sets up" the asynchronous adapter to match the settings of DIP switch 3 in Delta. The second re-directs printer output to the serial port. The switches on DIP switch 3 must be set as shown below to use this MODE command. (The IBM-DOS manual tells you how to create a different MODE command for different DIP switch settings.) You can put these two MODE commands into a file named AUTOEXEC.BAT and it will execute automatically each time you start your computer.

Table B-1
Serial switch settings

Switch	Setting	Function
3-1	OFF	8 data bits
3-2	OFF	No parity
3-3	ON	Serial busy, block mode
3-4	OFF	
3-5	either	Parity
3-6	ON	4800 baud
3-7	ON	
3-8	OFF	

The serial cable shown below will work with DIP switch 3 set as shown above to connect Delta to a serial interface on the IBM.

IBM-PC Serial Cable

Delta			**IBM-PC**	
Pin No.	**Function**		**Pin No.**	**Function**
2	TRANSMIT DATA	——	3	RECEIVE DATA
3	RECEIVE DATA	——	2	TRANSMIT DATA
4	REQUEST TO SEND	——	5	CLEAR TO SEND
5	CLEAR TO SEND	——	4	REQUEST TO SEND
7	SIGNAL GROUND	——	7	SIGNAL GROUND
8	CARRIER DETECT	——	4	REQUEST TO SEND
20	DATA TERMINAL READY	——	6	DATA SET READY

BASIC programing

All the programs in this book are written in the BASIC used by the IBM-PC. That makes it easy to do the things that we show you. But when you start writing your own programs there are several things that you should know.

IBM BASIC defaults to a printer width of 80. This means that it will automatically insert a carriage return and line feed after every 80 characters. If you want to print lines longer than 80 characters you will need to change the width of the printer. If you set the printer width to 255, then the IBM will *never* insert a line feed and carriage return, unless you start a new line. (This is what you want usually.) To set the width of the printer to 255, use this statement:

```
1ØØ WIDTH "LPT1:", 255
```

IBM BASIC has one other little trick that will mess up your graphics if you let it. IBM BASIC is very insistent about adding a line feed to a carriage return. This is fine if you are printing text, but if an ASCII 13 pops up in the middle of your graphics printout, IBM BASIC will *still* add a line feed to it. This will put strange things in the middle of your graphics, and leave you with extra characters at the end of your line.

There is an easy way to avoid this problem. You just open the printer as a random file. The following program shows how this is done.

```
1Ø OPEN "LPT1:" AS #1              ' RANDOM ACCESS
2Ø WIDTH #1, 255                   ' SET WIDTH TO 255
3Ø PRINT #1, "TESTING"             ' PRINT A LINE
4Ø PRINT #1, CHR$(1Ø)              ' ADD YOUR OWN LF
```

Listing programs

To list programs on the IBM-PC, use the LLIST command. This directs the listing to the printer instead of the screen.

Program listings

There are no program listings given here for the IBM-PC because all the programs in the book are written for the IBM-PC.

Appendix C

Apple II Computers

Apple II computers require an interface board (mounted inside the Apple II) and a cable to run Delta. Star recommends that you use the **grafstar**™ interface for the Apple II, II+, and IIe, by Star Micronics. It comes complete with a cable and is easily installed. A unique feature of the **grafstar**™ makes it possible to do some fancy dot graphics programming.

You can, of course, use many of the available parallel interface boards for the Apple II, and an appropriate cable.

Setting the switches

We recommend that you set the DIP switches in Delta as shown below when connecting it to an Apple II.

Recommended DIP Switch Settings for Apple

Switch	Setting	Function
1-1	ON	11 inch page size
1-2	ON	Normal print density
1-3	ON	10 CPI pitch
1-4	ON	Normal characters
1-5	ON	1/6 inch line feed
1-6	ON	U.S.A. Character set
1-7	ON	
1-8	ON	
2-1	ON	Paper-out detector active
2-2	OFF	Parallel interface
2-3	ON	7-bit interface
2-4	OFF	No auto line feed

Applesoft BASIC

The Apple II computer, using Applesoft BASIC, does not

Apple Parallel Cable

Delta		Apple Board	
Pin No.	**Function**	**Pin No.**	**Function**
25	SIG GND	1	SIG GND
26	SIG GND	2	SIG GND
27	SIG GND	3	SIG GND
1	STROBE	4	STROBE
28	SIG GND	5	N/C
2	DATA1	6	DATA1
3	DATA2	7	DATA2
4	DATA3	8	DATA3
5	DATA4	9	DATA4
6	DATA5	10	DATA5
7	DATA6	11	DATA6
8	DATA7	12	DATA7
9	DATA8	13	DATA8
10	ACK	14	ACK
29	SIG GND	15	SIG GND

have different types of PRINT statements for the screen and printer. You must add commands to your programs that direct the output of the PRINT statements to the printer. To direct output to the printer (with the interface board in slot #1) you must use the PR# 1 command. Depending on the version of Applesoft BASIC that you are using this command can take various forms. It is usually one of the following:

```
1Ø PR# 1
or
1Ø PRINT "<Ctrl-D>PR#1"
or
1Ø PRINT CHR$(4) "PR#1"
```

To return output to the screen, the command is PR#0, in the same form that works for PR# 1.

To allow line lengths longer than the Apple II usually uses you must add the following statement to your programs:

```
2Ø PRINT CHR$(9) "255N"
```

This allows lines of any length to be sent to the printer and is especially important for dot graphics. (The number 255 in the BASIC statement above could be replaced by any number from 0 to 255 and would set the line length to that value.)

Two codes are a particular problem on the Apple II: CHR$(7) and CHR$(9). Try to avoid using these in dot graphics programs.

The Apple II computer uses CHR$(9) as a printer initialization code. It won't send it on to the printer. There is a way to bypass this problem, however. You can change the printer initialization code to a value other than CHR$(9) like this:

```
PR#1
PRINT CHR$(9); CHR$(1)
```

This makes CHR$(1) the printer initialization code (and transfers the problems to *that* code) and allows you to use Delta's tabs.

There is one more way to sneak problem codes past the Apple II's operating system and that's to poke the codes directly to the output port. To send ASCII code 9, for example, you could do this:

```
1ØØ N = 9
11Ø IF PEEK(496Ø1)>127 THEN 11Ø
12Ø POKE 49296,N
```

Line 110 checks the printer's status, and when it's ok, line 120 pokes the code to the printer.

Listing programs

To make a listing of your BASIC programs on Delta from your Apple II computer you must take the following steps:

1. Be sure that the program that you wish to list is in the memory of the Apple II.
2. Direct the output to the printer by typing PR#1.
3. Type LIST to start the listing.
4. When the listing is finished, type PR#0 to redirect the output to the screen.

Program listings

Following are program listings in Applesoft BASIC for the main programs used in the tutorial section of this book. The only modifications that you might have to make are to the PR#1 and PR#0 commands as discussed above.

Chart program

```
100 PR# 1
110 PRINT CHR$ (9);"255N"
120 GOSUB 1000: GOSUB 2000
130 PRINT "*REGULAR*"
140 GOSUB 3000
150 PRINT "*DOUBLE STRIKE*"
160 PRINT DS$;
170 GOSUB 3000
180 PRINT "*EMPHASIZED*"
190 EM = TRUE
200 GOSUB 3000
210 PRINT "*DOUBLE STRIKE & EMPHASIZED*"
220 PRINT DS$;EM$;
230 GOSUB 3000
240 END
1000 REM
1010 IT$ = CHR$ (27) + CHR$ (52)
1020 RO$ = CHR$ (27) + CHR$ (53)
1030 EN$ = CHR$ (27) + "W" + CHR$ (1)
1040 CE$ = CHR$ (27) + "W" + CHR$ (0)
1050 PI$ = CHR$ (27) + "B" + CHR$ (1)
1060 EL$ = CHR$ (27) + "B" + CHR$ (2)
1070 CO$ = CHR$ (27) + "B" + CHR$ (3)
1080 EM$ = CHR$ (27) + "E"
1090 CM$ = CHR$ (27) + "F"
1100 DS$ = CHR$ (27) + "G"
1110 CD$ = CHR$ (27) + "H"
1120 UN$ = CHR$ (27) + "-" + CHR$ (1)
1130 CU$ = CHR$ (27) + "-" + CHR$ (0)
1140 SP$ = CHR$ (27) + "S" + CHR$ (0)
1150 SB$ = CHR$ (27) + "S" + CHR$ (1)
1160 CS$ = CHR$ (27) + "T"
1170 RA$ = CHR$ (27) + "@"
1180 TRUE = 1:FALSE = 0
1190  RETURN
2000  REM
2050  PRINT RA$
2060  PRINT EN$;" NORMAL    ENLARGED  "
2070  PRINT RA$;
2080  PRINT UN$;
2090  PRINT CO$;"CONDENSED ";
```

```
2100  PRINT EL$;"  ELITE   ";
2110  PRINT PI$;"  PICA    ";
2120  PRINT CO$;"CONDENSED ";
2130  PRINT EL$;"  ELITE   ";
2140  PRINT PI$;"  PICA    ";
2150  PRINT RA$
2160  RETURN
3000  REM
3050 IT = FALSE: PRINT RO$;
3060 UN = FALSE: PRINT CU$;
3070 EN = FALSE: PRINT CE$;
3080 PI = FALSE:
3100  PRINT CO$;
3110  GOSUB 3500:
3120  PRINT EL$;
3130  GOSUB 3500
3140  PRINT PI$;:PI = TRUE
3150  GOSUB 3500
3170  IF EN = TRUE THEN PRINT : GOTO 3190
3180  PRINT EN$;:EN = TRUE: GOTO 3080
3190  IF UN = TRUE THEN PRINT : GOTO 3210
3200  PRINT UN$;:UN = TRUE: GOTO 3070
3210  IF IT = TRUE THEN  PRINT RA$: RETURN
3220  PRINT IT$;:IT = TRUE: GOTO 3060
3500  REM
3550 BL$="      ":FD$="...."
3560  IF EM = FALSE THEN  PRINT "AB" + CHR$ (99) +
   CHR$ (100);: GOTO 3610
3570  IF PI = FALSE THEN  PRINT FD$;: GOTO 3590
3580  PRINT EM$;"AB"; CHR$ (99) + CHR$ (100);
3590  IF EN = TRUE THEN  PRINT " ";: RETURN
3600  IF EN = FALSE THEN  PRINT BL$;: RETURN
3610  REM
3620  IF EN = TRUE THEN  PRINT " ";: RETURN
3630  PRINT SP$;"X" + CHR$ (120);
3640  PRINT SB$;"Y" + CHR$ (121) + " ";
3650  PRINT CS$;
3660  RETURN
```

Special character chart program

```
10 L$ =  CHR$ (27) + ">"
20 Z$ =  CHR$ (27) + "="
25  PR# 1
27  PRINT  CHR$ (9);"255N"
30  FOR J = 160 TO 255 STEP 8
```

```
40  FOR I = J TO J + 7
50  PRINT I;"= ";L$; CHR$ (I);Z$;"  ";
60  NEXT I: PRINT : NEXT J
70  PR# 0
80  END
```

Macro program

```
10  PR# 1
15  PRINT  CHR$ (9);"255N"
20  PRINT  CHR$ (27);"+";
30  PRINT  CHR$ (18);
40  PRINT  CHR$ (27);"W"; CHR$ (0);
50  PRINT  CHR$ (27);"F";
60  PRINT  CHR$ (27);"H";
70  PRINT  CHR$ (27);"-"; CHR$ (0);
80  PRINT  CHR$ (27);"T";
90  PRINT  CHR$ (27);"5";
100  PRINT  CHR$ (30)
110  PR# 0
120  END
```

Bridge hand program

```
10  HOME
20  GOSUB 1000
30  GOSUB 2000
40  GOSUB 3000
50  GOSUB 4000
60  END
1000  REM
1010  DIM HA(4),DE(52),CA$(13),SU$(3)
1020 CA$(1) = " 2":CA$(2) = " 3":CA$(3) = " 4":
   CA$(4) = " 5" : CA$(5) = " 6"
1030 CA$(6) = " 7":CA$(7) = " 8":CA$(8) = " 9":
   CA$(9) = " 10"
1040 CA$(10) = " J":CA$(11) = " Q":CA$(12) = " K":
   CA$(13) = " A"
1050 SU$(0) = "S":SU$(1) = "H":SU$(2) = "D":
   SU$(3) = "C"
1080  RETURN
2000  REM
2010  PR# 1
2015  PRINT  CHR$ (9); CHR$ (25)
2016  PRINT  CHR$ (25);"255N"
```

```
2Ø2Ø  PRINT  CHR$ (27); CHR$ (68); CHR$ (2Ø);
   CHR$ (4Ø); CHR$ (Ø)
2Ø3Ø  PRINT  CHR$ (27); CHR$ (43); CHR$ (27);
   CHR$ (36); CHR$ (Ø); CHR$ (27); CHR$ (69);
   CHR$ (3Ø)
2Ø4Ø  PRINT  CHR$ (27); CHR$ (42); CHR$ (Ø)
2Ø45  FOR I = 1 TO 4
2Ø5Ø  PRINT  CHR$ (27); CHR$ (42); CHR$ (1);
2Ø6Ø  FOR J = 1 TO 13
2Ø7Ø  READ X: PRINT  CHR$ (X);
2Ø8Ø  NEXT J
2Ø9Ø  NEXT I
21ØØ  PRINT
211Ø  RETURN
212Ø  DATA  72,11,4,1Ø,2Ø,1Ø,52,72,52,1Ø,2Ø,1Ø,4
213Ø  DATA  83,11,16,8,2Ø,8,86,41,86,8,2Ø,8,16
214Ø  DATA  67,11,8,16,8,18,65,62,65,18,8,16,8
215Ø  DATA  68,11,8,Ø,28,Ø,62,65,62,Ø,28,Ø,8
3ØØØ  REM
3Ø1Ø  FOR CA = 1 TO 52
3Ø2Ø X = INT ( RND (1) * 4 + 1)
3Ø3Ø  IF HA(X) = 13 THEN 3Ø2Ø:
3Ø35 HA(X) = HA(X) + 1
3Ø4Ø DE(CA) = X
3Ø5Ø  NEXT CA
3Ø6Ø  RETURN
4ØØØ  REM
4Ø1Ø  PR# 1
4Ø12  PRINT CHR$ (9); CHR$ (25)
4Ø14  PRINT CHR$ (25);"255N":
4Ø15  PRINT CHR$ (27);"!"; CHR$ (9);"NORTH"
4Ø2Ø  PRINT CHR$ (27);"$"; CHR$ (1); CHR$ (27);
   CHR$ (7Ø);
4Ø3Ø HA = 1
4Ø4Ø  FOR SU = Ø TO 3
4Ø5Ø  PRINT CHR$ (9);
4Ø6Ø  GOSUB 43ØØ
4Ø7Ø  PRINT
4Ø8Ø  NEXT SU
4Ø9Ø  PRINT CHR$ (27);"!";"WEST"; CHR$(9);
   CHR$ (9);"EAST"
41ØØ  PRINT CHR$ (27);"$"; CHR$ (1); CHR$ (27);
   CHR$ (7Ø);
411Ø  FOR SU = Ø TO 3
```

```
4120 HA = 2
4130  GOSUB 4300
4140  PRINT  CHR$ (9); CHR$ (9);
4150 HA = 3
4160  GOSUB 4300
4170  PRINT
4180  NEXT SU
4190  PRINT CHR$ (27);"!"; CHR$ (9);"SOUTH"
4200  PRINT CHR$ (27);"$"; CHR$ (1); CHR$ (27);
   CHR$ (70);
4210 HA = 4
4220  FOR SU = 0 TO 3
4230  PRINT CHR$  (9);
4240  GOSUB 4300
4250  PRINT
4260  NEXT SU
4270  PRINT CHR$ (27);"$"; CHR$ (0); CHR$ (27);
   CHR$ (70)
4280  RETURN
4300  PRINT SU$(SU);
4310  FOR CA = 13 TO 1 STEP  - 1
4320 H1 = HA: IF DE(SU * 13 + CA) = H1 THEN  PRINT
   CA$(CA);
4330  NEXT CA
4340  RETURN
```

Numeral program

```
10  REM
20  PR# 1
25  PRINT  CHR$ (9);"255N"
27  PRINT  CHR$ (9); CHR$ (25)
30 DC$ =  CHR$ (27) +  CHR$ (42) +  CHR$ (1)
40 DP$ =  CHR$ (27) +  CHR$ (88) +  CHR$ (1)
50 CP$ =  CHR$ (27) +  CHR$ (88) +  CHR$ (0)
60 L7$ =  CHR$ (27) + "1":L12$ =  CHR$ (27) +
   CHR$ (50)
70  FOR N1 = 33 TO 73
80  PRINT DC$;
90  PRINT  CHR$ (N1);
100 READ N2
110  PRINT  CHR$ (N2);
120  FOR S = 1 TO 11
130  READ MS
140  PRINT  CHR$ (MS);
150   NEXT S
```

```
16Ø  NEXT N1
18Ø AS = 33
19Ø  FOR NU = Ø TO 9
21Ø NT$(NU) =  CHR$ (AS + Ø) +  CHR$ (AS + 1)
22Ø NB$(NU) =  CHR$ (AS + 2) +  CHR$ (AS + 3)
225 AS = AS + 4
23Ø  NEXT NU
24Ø BL$ =  CHR$ (73)
25Ø  PRINT DP$;L7$
26Ø  FOR NU = Ø TO 9
27Ø  PRINT NT$(NU);BL$;
28Ø  NEXT NU
29Ø  PRINT
3ØØ  FOR NU = Ø TO 9
31Ø  PRINT NB$(NU);BL$;
32Ø  NEXT NU
33Ø  PRINT CP$;L12$
34Ø  REM  ZERO
35Ø  DATA  11,Ø,96,16,1Ø4,16,44,3Ø,14,Ø,2,1
36Ø  DATA  11,2,1,2,1,6,8,38,88,32,88,32
37Ø  DATA  11,3,12,19,12,51,Ø,96,Ø,96,Ø,96
38Ø  DATA  11,Ø,32,Ø,48,Ø,28,3,12,3,4,3
39Ø  REM  ONE
4ØØ  DATA  11,Ø,Ø,Ø,Ø,Ø,4,Ø,4,Ø,4,126
41Ø  DATA  9,12,114,12,114,12,2,Ø,Ø,Ø,Ø,Ø
42Ø  DATA  11,64,Ø,64,Ø,64,Ø,64,32,8Ø,47,8Ø
43Ø  DATA  9,47,8Ø,47,64,Ø,64,Ø,64,Ø,Ø,Ø
44Ø  REM  TWO
45Ø  DATA  11,Ø,Ø,Ø,Ø,Ø,12,16,14,Ø,6,Ø
46Ø  DATA  11,3,Ø,3,Ø,7Ø,56,7Ø,56,4,24,Ø
47Ø  DATA  11,64,Ø,64,32,64,32,8Ø,32,8Ø,4Ø,64
48Ø  DATA  11,44,64,38,65,34,65,32,8Ø,32,88,Ø
49Ø  REM  THREE
5ØØ  DATA  11,Ø,Ø,Ø,Ø,Ø,Ø,4,2,4,2,4
51Ø  DATA  11,34,84,34,92,34,76,34,68,2,64,Ø
52Ø  DATA  11,16,Ø,48,Ø,56,64,48,64,32,64,32
53Ø  DATA  11,64,32,64,48,9,54,9,22,9,6,1
54Ø  REM  FOUR
55Ø  DATA  11,Ø,Ø,Ø,Ø,Ø,Ø,64,36,88,32,16
56Ø  DATA  11,Ø,Ø,64,32,64,56,64,6Ø,2,12,Ø
57Ø  DATA  11,Ø,8,4,1Ø,5,1Ø,5,8,4,72,4
58Ø  DATA  11,88,38,89,38,89,6,73,4,8,6,Ø
59Ø  REM  FIVE
6ØØ  DATA  11,Ø,Ø,Ø,Ø,64,32,84,5Ø,76,34,68
61Ø  DATA  1Ø,34,68,34,68,34,68,2,68,2,Ø,Ø
```

```
62Ø  DATA  1Ø,Ø,32,24,1Ø1,24,97,Ø,64,Ø,64,Ø
63Ø  DATA  11,64,Ø,96,1,48,15,48,15,16,15,Ø
64Ø  REM  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   11,6,8,23,8,55,8,99,Ø,65,Ø,64
68Ø  DATA  11,Ø,96,Ø,112,1,62,1,3Ø,1,14,Ø
69Ø  REM  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  REM  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  REM  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,6,Ø
84Ø  REM  SPACE
85Ø  DATA  11,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø
```

Download utility program

```
1Ø  DIM Z(8,12),MM(11)
15 CS$ = "*":SC$ = "@":ST$ =
   "                                        "
2Ø  HOME : GOSUB 66Ø
3Ø  VTAB 24: HTAB 34
4Ø  GET A$
5Ø  IF A$ = "P" THEN  GOSUB 68Ø: GOTO 3Ø
6Ø  IF A$ = "E" THEN  HOME : GOSUB 9ØØ: GOSUB 26Ø:
   GOTO 3Ø
7Ø  IF A$ =  CHR$ (27) THEN  HOME : END
8Ø  PRINT  CHR$ (7);: GOTO 3Ø
26Ø  REM
265  FOR I = 1 TO 11:MM(I) = Ø: NEXT I
27Ø  VTAB 3: HTAB 6: PRINT CS$;
28Ø  GET A$
29Ø  IF A$ = "J" THEN  GOSUB 39Ø: GOTO 37Ø
3ØØ  IF A$ = "K" THEN  GOSUB 41Ø: GOTO 37Ø
31Ø  IF A$ = "M" THEN  GOSUB 43Ø: GOTO 37Ø
```

```
32Ø  IF A$ = "I" THEN  GOSUB 45Ø: GOTO 37Ø
33Ø  IF A$ =  CHR$ (13) THEN  GOSUB 47Ø: GOTO 37Ø
34Ø  IF A$ =  CHR$ (32) THEN  GOSUB 49Ø: GOTO 37Ø
35Ø  IF A$ =  CHR$ (27) THEN  GOSUB 5ØØ: GOTO 38Ø
37Ø  GOTO 28Ø
38Ø  RETURN
39Ø  GOSUB 1ØØØ:Y = Y - 2:H = H - 1: IF Y < 1 THEN
   PRINT  CHR$ (7);:Y = 1:H = 1
4ØØ  GOSUB 1Ø5Ø: RETURN
41Ø  GOSUB 1ØØØ:Y = Y + 2:H = H + 1: IF Y >21 THEN
   PRINT  CHR$ (7);:Y = 21:H = 11
42Ø  GOSUB 1Ø5Ø: RETURN
43Ø  GOSUB 1ØØØ:X = X + 2:G = G + 1: IF X > 13 THEN
   PRINT  CHR$ (7);:X = 13:G = 7
44Ø  GOSUB 1Ø5Ø: RETURN
45Ø  GOSUB 1ØØØ:X = X - 2:G = G - 1: IF X < 1 THEN
   PRINT  CHR$ (7);:X = 1:G = 1
46Ø  GOSUB 1Ø5Ø: RETURN
47Ø  IF Z(G,H - 1) = 1 OR Z(G,H + 1) = 1 THEN  PRINT
   CHR$ (7);: RETURN
48Ø Z(G,H) = 1: INVERSE : VTAB X + 2: HTAB Y + 5:
   PRINT SC$;: NORMAL : RETURN
49Ø Z(G,H) = Ø: NORMAL : VTAB X + 2: HTAB Y + 5:
   PRINT CS$;: RETURN
5ØØ  REM
51Ø  FOR I = 2 TO 12: VTAB I: HTAB 29: PRINT
   "            ";: NEXT I
52Ø  IF Z(G,H) = 1 THEN  VTAB X + 2: HTAB Y + 5:
   NORMAL : PRINT SC$;: GOTO 54Ø
53Ø  IF Z(G,H) = Ø THEN  VTAB X + 2: HTAB Y + 5:
   NORMAL : PRINT " ";
54Ø  REM
55Ø  FOR I = 1 TO 11: FOR J = 1 TO 7
56Ø MM(I) = MM(I) + Z(J,I) * 2 ^ (J - 1): NEXT J:
   NEXT I
567  FOR I = 1 TO 11:S$ = STR$ (MM(I)): FOR J = 1 TO
   LEN (S$)
568  VTAB J + 16: HTAB I * 2 + 4: PRINT
   MID$ (S$,J,1);: NEXT J: NEXT I
58Ø  GOSUB 66Ø: RETURN
66Ø  FOR I = 1 TO 7: FOR J = 1 TO 11:Z(I,J) = Ø:
   NEXT J: NEXT I
67Ø  VTAB 24: HTAB 2: PRINT "E)DIT  P)RINTER  <ESC>)
   END ";: RETURN
68Ø  REM
```

```
69Ø  VTAB 21: HTAB 2: INPUT "NORMAL OR PROPORTIONAL
   (N/P) --> ";AN$
7ØØ  IF AN$ = "N" THEN PR = Ø: GOTO 75Ø
71Ø  IF AN$ = "P" THEN 73Ø
72Ø  PRINT  CHR$ (7);: GOTO 69Ø
73Ø  VTAB 21: HTAB 2: PRINT ST$;: VTAB 21: HTAB 2:
735  INPUT "PROPORTIONAL DATA (4-11) -->";PR
74Ø  IF PR < 4 OR PR > 11 THEN 73Ø
75Ø  VTAB 21: HTAB 2: PRINT ST$;: VTAB 21: HTAB 2:
755  INPUT "SHIFTED DOWN 1 ELSE ENTER Ø --> ";SH
76Ø  IF SH < Ø OR SH > 1 THEN PRINT  CHR$ (7);: GOTO
   75Ø
77Ø  VTAB 21: HTAB 2: PRINT ST$;: VTAB 21: HTAB 2:
775  INPUT "ENTER YOUR ASCII (33-126) --> ";AS
78Ø  IF AS < 33 OR AS > 126 THEN 77Ø
785  VTAB 21: HTAB 2: PRINT ST$;: VTAB 23: HTAB 38
79Ø  IF SH = 1 THEN SH = 16
8ØØ N1 = AS:N2 = PR + SH
81Ø  FOR I = 1 TO 11:MM$ = MM$ +  CHR$ (MM(I)):
   NEXT I
815  PR# 1
816  PRINT  CHR$ (9);"255N"
82Ø  PRINT  CHR$ (27);"*"; CHR$ (1); CHR$ (N1);
   CHR$ (N2);MM$
825  IF AN$ = "N" THEN  PRINT  CHR$ (27);"$";
   CHR$ (1): GOTO 83Ø
827  PRINT  CHR$ (27);"X"; CHR$ (1)
83Ø  FOR I = 1 TO 2Ø: PRINT  CHR$ (N1);" ";: NEXT I:
   PRINT
84Ø  PRINT  CHR$ (14);: FOR I = 1 TO 1Ø: PRINT
   CHR$ (N1);" ";: NEXT I: PRINT
85Ø  PRINT  CHR$ (15);: FOR I = 1 TO 2Ø: PRINT
   CHR$ (N1);" ";: NEXT I: PRINT  CHR$ (18)
86Ø  IF AN$ = "N" THEN  PRINT  CHR$ (27);"$"
   CHR$ (Ø): GOTO 87Ø
865  PRINT  CHR$ (27);"X"; CHR$ (Ø)
87Ø  PRINT  CHR$ (27);"@";:MM$ = "": RETURN
9ØØ X = 1:Y = 1:G = 1:H = 1
9Ø1  HOME
9Ø2  FOR I = 2 TO 16 STEP 2: VTAB I: HTAB 5: FOR J =
   1 TO 23: PRINT "-";: NEXT J: PRINT : NEXT I
9Ø4  FOR J = 3 TO 16 STEP 2: VTAB J: FOR I = 5 TO 27
   STEP 2: HTAB I: PRINT "!";: NEXT I: PRINT :
   NEXT J
9Ø5 K = 1: VTAB 1: HTAB 5
```

```
9Ø6  FOR K = 1 TO 11: PRINT K;" ";: NEXT K
9Ø7 K = Ø
9Ø8  FOR V = 3 TO 15 STEP 2: VTAB V: HTAB 2:
   PRINT 2 ^ K:K = K + 1: NEXT V
91Ø  VTAB 3: HTAB 3Ø: PRINT "CURSER "
912  VTAB 4: HTAB 29: PRINT "MOVEMENT"
913  VTAB 5: HTAB 29: PRINT "--------"
914  VTAB 6: HTAB 29: PRINT "<I> UP "
916  VTAB 7: HTAB 29: PRINT "<M> DOWN"
918  VTAB 8: HTAB 29: PRINT "<J> LEFT"
92Ø  VTAB 9: HTAB 29: PRINT "<K> RIGHT"
922  VTAB 1Ø: HTAB 29: PRINT "<RET> INSERT"
924  VTAB 11: HTAB 29: PRINT "<SPACE> DEL"
925  VTAB 12: HTAB 29: PRINT "<ESC> EXIT"
926  RETURN
1ØØØ  IF Z(G,H) = Ø THEN  VTAB X + 2: HTAB Y + 5:
   PRINT " ";
1Ø1Ø  IF Z(G,H) = 1 THEN  VTAB X + 2: HTAB Y + 5:
   PRINT SC$;
1Ø2Ø  RETURN
1Ø5Ø  IF Z(G,H) = 1 THEN  INVERSE : VTAB X + 2:
   HTAB Y + 5: PRINT CS$;: NORMAL
1Ø6Ø  IF Z(G,H) = Ø THEN  NORMAL : VTAB X + 2:
   HTAB Y + 5: PRINT CS$;: NORMAL
1Ø7Ø  RETURN
```

Delta Plot program

```
4  HOME : PRINT " ": PRINT " "
5  PRINT " ": PRINT " "
6  PRINT "THIS PROGRAM TAKES ABOUT"
7  PRINT "1 MINUTE  TO RUN. PLEASE"
8  PRINT " TURN ON YOUR PRINTER AND"
9  PRINT "STAND BY................"
1Ø A = 24576
2Ø  FOR I = A TO A + 12
3Ø  READ B
35  POKE I,B
4Ø  NEXT I
5Ø  DATA  32,74,255,165,25Ø,5,251
6Ø  DATA  133,252,32,63,255,96
1ØØ  REM  DELTA-PLOT
11Ø  DIM BIT%(75,14)
1ØØØ  REM  SET PROGRAM CONSTANTS
1Ø1Ø MASK%(1) = 64:MASK%(4) = 8
1Ø2Ø MASK%(2) = 32:MASK%(5) = 4
```

```
1Ø3Ø MASK%(3) = 16:MASK%(6) = 2
1Ø4Ø LX = 2Ø:LY = 2Ø
1Ø5Ø XFAC = 72 / LX:YFAC = 87 / LY
2ØØØ  REM  PLOT CURVE
2Ø1Ø RAD = 9
2Ø2Ø X1 = 19:Y1 = 1Ø
2Ø3Ø  FOR ANG = Ø TO 36Ø STEP 1Ø
2Ø4Ø R1 = ANG * 6.28 / 36Ø
2Ø5Ø X2 = RAD *  COS (R1) + 1Ø:Y2 = RAD *  SIN (R1)
   + 1Ø
2Ø6Ø  GOSUB 4ØØØ
2Ø7Ø  NEXT ANG
3ØØØ  REM  SEND BIT IMAGE MAP TO PRINTER
3ØØ5  PR# 1
3ØØ6  PRINT  CHR$ (9);"ØN"
3Ø1Ø  PRINT  CHR$ (27);"A"; CHR$ (6)
3Ø2Ø  FOR ROW = Ø TO 14
3Ø22 A$ = ""
3Ø5Ø  FOR COL = 1 TO 75
3Ø6Ø A$ = A$ +  CHR$ (BIT%(COL,ROW))
3Ø7Ø  NEXT COL
3Ø8Ø  PRINT  CHR$ (27);"K"; CHR$ (75); CHR$ (Ø);A$
3Ø9Ø  NEXT ROW
31ØØ  PRINT
31Ø2  PR# Ø
323Ø  END
4ØØØ  REM   DRAW A LINE FROM X1,Y1 TO X2,Y2
4Ø1Ø XL = X2 - X1:YL = Y2 - Y1
4Ø2Ø NX = ABS (XL * XFAC):NY = ABS (YL * YFAC)
4Ø3Ø  IF NX < NY THEN NX = NY
4Ø4Ø NS% = INT (NX + 1)
4Ø5Ø DX = XL / NS%:DY = YL / NS%
4Ø6Ø  FOR I = 1 TO NS%
4Ø7Ø X1 = X1 + DX:Y1 = Y1 + DY
4Ø8Ø  GOSUB 5ØØØ
4Ø9Ø  NEXT I
41ØØ  RETURN
5ØØØ  REM  PLOT A POINT AT X1,Y1
5Ø1Ø XX = X1 * XFAC:YY = Y1 * YFAC
5Ø2Ø COL = INT (XX) + 1
5Ø3Ø ROW = INT (YY / 6)
5Ø4Ø XIT% =  INT (YY - (6 * ROW)) + 1
5Ø42  POKE 25Ø,BIT%(COL,ROW)
5Ø44  POKE 251,MASK%(XIT%)
5Ø46  CALL 24576
```

```
5Ø5Ø BIT%(COL,ROW) = PEEK (252)
5Ø6Ø  RETURN
```

Pie chart program

```
4  HOME
5  PRINT "                  PLEASE STAND BY"
1Ø A = 768
2Ø  FOR I = A TO A + 12
3Ø  READ B
35  POKE I,B
4Ø  NEXT I
5Ø  DATA  32,74,255,165,25Ø,5,251
6Ø  DATA  133,252,32,63,255,96
1ØØ  REM  PIECHART
11Ø  DIM BIT%(19Ø,36),A$(36),PCT%(25),TXT$(42),
   PTXT$(25)
12Ø ES$ =  CHR$ (27):LF$ =  CHR$ (1Ø)
13Ø FF$ =  CHR$ (12):VT$ =  CHR$ (11)
14Ø EM$ = ES$ + "E":CE$ = ES$ + "F"
15Ø  FOR I = 1 TO 168:SP$ = SP$ + CHR$ (Ø): NEXT I
1ØØØ  REM SET PROGRAM CONSTANTS
1Ø2Ø MASK%(1) = 64:MASK%(4) = 8
1Ø2Ø MASK%(2) = 32:MASK%(5) = 4
1Ø3Ø MASK%(3) = 16:MASK%(6) = 2
1Ø4Ø LX = 2Ø:LY = 2Ø
1Ø5Ø XFAC = 19Ø / LX:YFAC = 216 / LY
1Ø6Ø  FOR I = 1 TO 42
1Ø7Ø  FOR J = 1 TO 8Ø:TXT$(I) = TXT$(I) + " "
1Ø8Ø  NEXT J: NEXT I
1Ø9Ø  GOSUB 7ØØØ
1Ø92  HOME : PRINT " ": PRINT " "
1Ø94  PRINT " ": PRINT " "
1Ø96  PRINT "THIS PROGRAM TAKES ABOUT"
1Ø97  PRINT "4 MINUTES TO RUN. PLEASE"
1Ø98  PRINT "TURN ON YOUR PRINTER AND"
1Ø99  PRINT "STAND BY................"
2ØØØ  REM  PLOT CURVE
2Ø1Ø RAD = 9
2Ø2Ø X1 = 19:Y1 = 1Ø
2Ø25  PRINT " ";
2Ø3Ø  FOR ANG = Ø TO 36Ø STEP 5
2Ø4Ø R1 = ANG * 6.28 / 36Ø
2Ø5Ø X2 = RAD * COS (R1) + 1Ø:Y2 = RAD * SIN (R1) +
   1Ø
```

```
2060  GOSUB 4000
2070  NEXT ANG
2080  FOR PI = 1 TO NP%
2090 X1 = 10:Y1 = 10
2100 TP% = TP% + PCT%(PI)
2110 ANG = 360 * TP% * .01
2120 R1 = ANG * 6.28 / 360
2130 X2 = RAD *  COS (R1) + 10:Y2 = RAD *  SIN (R1)
   + 10
2140  GOSUB 4000
2150  GOSUB 6000
2160  NEXT PI
3000  REM  SEND BIT IMAGE MAP TO PRINTER
3020  FOR ROW = 0 TO 35
3022 A$(ROW) = ""
3050  FOR COL = 1 TO 190
3060 A$(ROW) = A$(ROW) +  CHR$ (BIT%(COL,ROW))
3070  NEXT COL
3080  NEXT ROW
3090  PR# 1
3100  PRINT  CHR$ (9);"0N"
3110 X = (40 -  LEN (T1$) / 2)
3120  FOR I = 1 TO X: PRINT " ";: NEXT I
3130  PRINT EM$:TI$:CE$:LF$
3140  PRINT VT$;VT$;VT$
3150  PRINT ES$;"A"; CHR$ (3)
3160  PRINT TXT$(1);LF$;TXT$(2);LF$;TXT$(3);LF$
3170  FOR ROW = 0 TO 35
3180  PRINT ES$;"K"; CHR$ (102); CHR$ (1);
   SP$;A$(ROW)
3200  PRINT TXT$(ROW + 4)
3210  NEXT ROW
3220  PRINT TXT$(40);LF$
3230  PRINT TXT$(41);LF$
3240  PRINT TXT$(42);LF$
3250  PRINT ES$;"2";FF$
3255  PR# 0
3257  HOME
3260  END
4000  REM   DRAW A LINE FROM X1,Y1, TO X2,Y2
4010 XL = X2 - X1:YL = Y2 - Y1
4020 NX =  ABS (XL * XFAC):NY = ABS (YL * YFAC)
4030  IF NX < NY THEN NX = NY
4040 NS% =  INT (NX + 1)
4050 DX = XL / NS%:DY = YL / NS%
```

```
4Ø6Ø  FOR I = 1 TO NS%
4Ø7Ø X1 = X1 + DX:Y1 = Y1 + DY
4Ø8Ø  GOSUB 5ØØØ
4Ø9Ø  NEXT I
41ØØ  RETURN
5ØØØ  REM  PLOT A POINT AT X1,Y1
5Ø1Ø XX = X1 * XFAC:YY = Y1 * YFAC
5Ø2Ø COL =  INT (XX) + 1
5Ø3Ø ROW =  INT (YY / 6)
5Ø4Ø XIT% =  INT (YY - (6 * ROW)) + 1
5Ø42  POKE 25Ø,BIT%(COL,ROW)
5Ø44  POKE 251,MASK%(XIT%)
5Ø46  CALL 768
5Ø5Ø BIT%(COL,ROW) = PEEK (252)
5Ø6Ø  RETURN
6ØØØ  REM
6Ø1Ø MA% = (ANG + PA%) / 2
6Ø2Ø R1 = MA% * 6.28 / 36Ø
6Ø3Ø X3 =  INT (2Ø *  SIN (R1)):Y3 =
   INT (22 *  COS (R1))
6Ø4Ø X4 = 22 + X3:Y4 = 4Ø + Y3
6Ø5Ø  IF MA% > 27Ø OR MA% < 9Ø THEN  GOSUB 61ØØ:
   GOTO 6Ø7Ø
6Ø6Ø  GOSUB 62ØØ
6Ø7Ø PA% = ANG
6Ø8Ø  RETURN
61ØØ MM$ = TXT$(X4)
61Ø2 LL$ =  LEFT$ (MM$,Y4)
61Ø4 PP =  LEN (PTXT$(PI))
61Ø6 RR$ =  RIGHT$ (MM$,8Ø - (Y4 + PP))
61Ø8 TXT$(X4) = LL$ + PTXT$(PI) + RR$
611Ø  RETURN
62ØØ MM$ = TXT$(X4)
62Ø2 PP =  LEN (PTXT$(PI))
62Ø4 LL$ =  LEFT$ (MM$,(Y4 - PP))
62Ø6 RR$ =  RIGHT$ (MM$,(8Ø - Y4))
62Ø8 TXT$(X4) = LL$ + PTXT$(PI) + RR$
621Ø  RETURN
7ØØØ  REM
7ØØ5 I = 1
7Ø1Ø  HOME : PRINT : PRINT : PRINT
7Ø2Ø  INPUT "ENTER TITLE FOR CHART ";TI$
7Ø3Ø AS% = Ø:AL% = 1ØØ
7Ø4Ø  HOME
7Ø5Ø  PRINT "TOTAL SO FAR    : ";AS%
```

```
7Ø6Ø  PRINT "TOTAL REMAINING : ";AL%
7Ø7Ø  INPUT "ENTER % FOR FIELD ";PCT%(I)
7Ø8Ø  IF PCT%(I) > AL% OR PCT%(I) = Ø THEN
   PCT%(I) = AL%
7Ø9Ø AL% = AL% - PCT%(I)
71ØØ AS% = AS% + PCT%(I)
711Ø  INPUT "ENTER DESCRIPTION OF FIELD : ";PTXT$(I)
712Ø  IF  LEN (PTXT$(I)) > 15 THEN  PRINT "FIELD TOO
   LONG - 15 CHAR. MAX": GOTO 711Ø
713Ø  IF AL% = Ø THEN  GOTO 72ØØ
714Ø I = I + 1
715Ø  GOTO 7Ø4Ø
72ØØ NP% = I
721Ø  IF NP% = 1 THEN 7Ø4Ø
722Ø  HOME
723Ø  RETURN
```

Appendix D

TRS-80 Computers

All that's required to connect Delta to your TRS-80 is a cable. It is available at your Delta dealer.

When connecting Delta to a TRS-80 we recommend that you set the DIP switches in Delta as shown below.

Recommended DIP Switch Settings for TRS-80

Switch	Setting	Function
1-1	ON	11 inch page size
1-2	ON	Normal print density
1-3	ON	10 CPI pitch
1-4	ON	Normal characters
1-5	ON	1/6 inch line feed
1-6	ON	U.S.A. Character set
1-7	ON	
1-8	ON	
2-1	ON	Paper-out detector active
2-2	OFF	Parallel interface
2-3	OFF	8-bit interface
2-4	ON	Auto line feed

TRS-80 Model I Parallel Cable

Delta		TRS-80 Model I	
Pin No.	**Function**	**Pin No.**	**Function**
1	STROBE	1	STROBE
2	D1	3	D1
3	D2	5	D2
4	D3	7	D3
5	D4	9	D4
6	D5	11	D5
7	D6	13	D6
8	D7	15	D7
9	D8	17	D8
11	BUSY	21	READY

TRS-80 Model II Parallel Cable

Delta		TRS-80 Model II	
Pin No.	**Function**	**Pin No.**	**Function**
1	STROBE	1	STROBE
2	D1	3	D1
3	D2	5	D2
4	D3	7	D3
5	D4	9	D4
6	D5	11	D5
7	D6	13	D6
8	D7	15	D7
9	D8	17	D8
10	ACK	19	ACK
11	BUSY	21	BUSY

TRS-80 BASIC

You may have to initialize your Model II to direct LPRINT statements to the printer. Use the SYSTEM "FORMS" command to do it.

TRS-80 uses another version of Microsoft Basic. Most of the programs in this book will work just as they are, but the TRS-80 does have a few unique "problem codes". They are 0, 10, 11, and 12. None of these are passed properly to the printer.

You can bypass the TRS-80's BASIC and send these codes directly to the printer with the following short routine. The variable N must be set equal to the code that you wish to pass (in our example it's 0).

```
9Ø N = Ø
1ØØ IF PEEK(14312)<>63 THEN 1ØØ
11Ø POKE 14312,N
```

Or you can use this special printer driver that will solve all your problems. Just run this program first, and then any codes sent by a BASIC program will be sent directly to the printer. This program is for the TRS-80 Model III.

```
5 REM  DRIVER FOR TRS-8Ø III
1Ø AD=16571
2Ø FOR I=Ø TO 14
3Ø READ A:POKE AD+I,A
```

```
40 NEXT I
50 POKE 16422,187
60 POKE 16423,64
70 DATA 33,232,55,203,126,32,252,33,17,
   0,57,126,211,251,201
80 END
```

And here is a version for the TRS-80 Model I.

```
5 REM  DRIVER FOR TRS-80 I
10 AD=16571
20 FOR I=0 TO 15
30 READ A:POKE AD+I,A
40 NEXT I
50 POKE 16422,187
60 POKE 16423,64
70 DATA 33,232,55,203,126,32,252,33,17,
   0,57,126,50,232,55,201
80 END
```

Chart program

```
100 CLEAR 1000
110 GOSUB 1000
120 GOSUB 2000
130 LPRINT "*REGULAR*"
140 GOSUB 3000
150 LPRINT  "*DOUBLE STRIKE*"
160 LPRINT DS$;
170 GOSUB 3000
180 LPRINT "*EMPHASIZED*"
190 EM = TRUE
200 GOSUB 3000
210 LPRINT "*DOUBLE STRIKE & EMPHASIZED*"
220 LPRINT DS$ EM$;
230 GOSUB 3000
240 END
1000 REM
1060 IT$ = CHR$(27) + CHR$(52)
1070 RO$  = CHR$(27) + CHR$(53)
1080 REM
1090 EN$  = CHR$(27) + CHR$(87) + CHR$(1)
1100 NW$  = CHR$(27) + CHR$(87) + CHR$(0)
1110 PI$      = CHR$(27) + CHR$(66) + CHR$(1)
```

```
1120 EL$    = CHR$(27) + CHR$(66) + CHR$(2)
1130 CO$ = CHR$(27) + CHR$(66) + CHR$(3)
1140 REM
1150 EM$    = CHR$(27) + CHR$(69)
1160 NE$    = CHR$(27) + CHR$(70)
1170 DS$ = CHR$(27) + CHR$(71)
1180 ND$ = CHR$(27) + CHR$(72)
1190 UN$    = CHR$(27) + CHR$(45) + CHR$(1)
1200 NU$    = CHR$(27) + CHR$(45) + CHR$(0)
1210 SP$   = CHR$(27) + CHR$(83) + CHR$(0)
1220 SB$     = CHR$(27) + CHR$(83) + CHR$(1)
1230 NS$      = CHR$(27) + CHR$(84)
1240 RA$ = NE$+NU$+ND$
1250 RA$ = RA$ + RO$ + PI$ +NW$
1260 REM
1270 TRUE = 1 : FALSE = 0
1290 RETURN
2000 REM
2050 LPRINT RA$
2060 LPRINT EN$ "   NORMAL     ENLARGED  "
2070 LPRINT RA$;
2080 LPRINT UN$;
2090 LPRINT CO$ "CONDENSED ";
2100 LPRINT EL$     "  ELITE   ";
2110 LPRINT PI$      "  PICA    ";
2120 LPRINT CO$ "CONDENSED ";
2130 LPRINT EL$     "  ELITE   ";
2140 LPRINT PI$      "  PICA    "
2150 LPRINT RA$
2160 RETURN
3000 REM
3050 IT    = FALSE       :LPRINT RO$;
3060 UN = FALSE       :LPRINT NU$;
3070 EN   = FALSE       :LPRINT NW$;
3080 PI       = FALSE
3090 REM
3100 LPRINT CO$;
3110 GOSUB 3500 'PRINT AS REQUIRED
3120 LPRINT EL$;
3130 GOSUB 3500 'PRINT AS REQUIRED
3140 LPRINT PI$; :PI = TRUE
3150 GOSUB 3500 'PRINT AS REQUIRED
3160 REM
3170 IF EN = TRUE THEN LPRINT :GOTO 3190
3180 LPRINT EN$; :EN = TRUE :GOTO 3080
```

```
3190 IF UN = TRUE THEN LPRINT :GOTO 3210
3200 LPRINT UN$;  :UN = TRUE :GOTO 3070
3210 IF IT = TRUE THEN LPRINT RA$ :RETURN
3220 LPRINT IT$;  :IT = TRUE  :GOTO 3060
3500 REM
3550 BL$ = STRING$(6,32)   :FD$ = "...."
3560 IF EM = FALSE THEN LPRINT "ABcd";  :GOTO 3610
3570 IF PI = FALSE THEN LPRINT FD$;     :GOTO 3590
3580 LPRINT EM$ "ABcd" ;
3590 IF EN = TRUE THEN LPRINT " ";        :ELSE
   LPRINT BL$;
3600 RETURN
3610 REM
3620 IF EN = TRUE THEN LPRINT " ";         :RETURN
3630 LPRINT SP$; "Xx";
3640 LPRINT SB$;   "Yy  ";
3650 LPRINT NS$;
3660 RETURN
```

Special character chart program

```
10 FOR J = 160 TO 255 STEP 8
20 FOR I = J TO J+ 7
30 LPRINT I "=" CHR$(I) CHR$(9);
40 NEXT I : LPRINT : NEXT J
```

Macro program

```
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$(27); "T";          ' SUPER & SUBSCRIPT
   OFF
80 LPRINT CHR$(27); "5";
90 LPRINT CHR$(30);         ' END MACRO DEFINITION
```

Bridge hand program

```
20 GOSUB 1000
30 GOSUB 2000
40 GOSUB 3000
50 GOSUB 4000
```

```
7Ø END
1ØØØ REM INITIALIZE VARIABLES
1Ø1Ø DIM HA(4), DE(52), CA$(13), SU$(3)
1Ø2Ø CA$(1)=" 2" : CA$(2)=" 3" : CA$(3)=" 4"
1Ø3Ø CA$(4)=" 5" : CA$(5)=" 6" : CA$(6)= " 7"
1Ø4Ø CA$(7)=" 8" : CA$(8)=" 9" : CA$(9)=" 1Ø"
1Ø5Ø CA$(1Ø)=" J" : CA$(11)=" Q" : CA$(12)=" K" :
   CA$(13)=" A"
1Ø6Ø SU$(Ø)="S" : SU$(1)="H" : SU$(2)="D" :
   SU$(3)="C"
1Ø7Ø RETURN
2ØØØ REM INITIALIZE PRINTER
2Ø1Ø LPRINT CHR$(27); CHR$(68); CHR$(2Ø); CHR$(4Ø);
   CHR$(Ø)
2Ø2Ø LPRINT CHR$(27) CHR$(43); CHR$(27) CHR$(36)
   CHR$(Ø); CHR$(27) CHR$(69) CHR$(3Ø)
2Ø3Ø LPRINT CHR$(27) CHR$(42) CHR$(Ø)
2Ø4Ø FOR I=1 TO 4
2Ø5Ø LPRINT CHR$(27); CHR$(42); CHR$(1);
2Ø6Ø FOR J=1 TO 13
2Ø7Ø READ X : LPRINT CHR$(X);
2Ø8Ø NEXT J
2Ø9Ø NEXT I
21ØØ LPRINT
211Ø RETURN
212Ø DATA 72,11,4,1Ø,2Ø,1Ø,52,72,52,1Ø,2Ø,1Ø,4
213Ø DATA 83,11,16,8,2Ø,8,86,41,86,8,2Ø,8,16
214Ø DATA 67,11,8,16,8,18,65,62,65,18,8,16,8
215Ø DATA 68,11,8,Ø,28,Ø,62,65,62,Ø,28,Ø,8
3ØØØ  REM DEAL CARD
3Ø1Ø FOR CA = 1 TO 52
3Ø2Ø X = INT(RND(Ø) * 4 + 1)
3Ø3Ø IF HA(X)=13 THEN 3Ø2Ø
3Ø35 HA(X)=HA(X) + 1
3Ø4Ø DE(CA)=X
3Ø5Ø NEXT CA
3Ø6Ø RETURN
4ØØØ REM PRINT FOUR HANDS
4Ø1Ø LPRINT CHR$(27); "!"; CHR$(9); "NORTH"
4Ø2Ø LPRINT CHR$(27); "$"; CHR$(1); CHR$(27);
   CHR$(7Ø);
4Ø3Ø HA = 1
4Ø4Ø FOR SU = Ø TO 3
4Ø5Ø LPRINT CHR$(9);
4Ø6Ø GOSUB 43ØØ
```

```
4Ø7Ø LPRINT
4Ø8Ø NEXT SU
4Ø9Ø LPRINT CHR$(27); "!"; "WEST"; CHR$(9); CHR$(9);
   "EAST"
41ØØ LPRINT CHR$(27); "$"; CHR$(1); CHR$(27);
   CHR$(7Ø);
411Ø FOR SU = Ø TO 3
412Ø HA = 2
413Ø GOSUB 43ØØ
414Ø LPRINT CHR$(9) CHR$(9);
415Ø HA = 3
416Ø GOSUB 43ØØ
417Ø LPRINT
418Ø NEXT SU
419Ø LPRINT CHR$(27); "!"; CHR$(9); "SOUTH"
42ØØ LPRINT CHR$(27); "$"; CHR$(1); CHR$(27);
   CHR$(7Ø);
421Ø HA = 4
422Ø FOR SU = Ø TO 3
423Ø LPRINT CHR$(9);
424Ø GOSUB 43ØØ
425Ø LPRINT
426Ø NEXT SU
427Ø LPRINT CHR$(27); "$"; CHR$(Ø); CHR$(27);
   CHR$(7Ø)
428Ø RETURN
429Ø REM PRINT ONE LINE
43ØØ LPRINT SU$(SU);
431Ø FOR CA = 13 TO 1 STEP -1
432Ø IF DE(SU*13+CA)=HA THEN LPRINT CA$(CA);
433Ø NEXT CA
434Ø RETURN
```

Numeral program

```
1Ø REM PROGRAM TO DEFINE AND PRINT NUMERAL
2Ø REM EACH NUMERAL IS MADE UP OF 4 CHARACTERS (2
   WIDE X 2 HIGH)
3Ø DD$ = CHR$(27)+CHR$(42)+CHR$(1)
4Ø DP$ = CHR$(27)+CHR$(88)+CHR$(1)
5Ø ND$ = CHR$(27)+CHR$(88)+CHR$(Ø)
6Ø L7$ = CHR$(27)+CHR$(49) : L12$ =
   CHR$(27)+CHR$(5Ø)
7Ø FOR N1 = 16Ø TO 2ØØ
8Ø LPRINT DD$;
```

```
9Ø LPRINT CHR$(N1);
1ØØ READ N2
11Ø LPRINT CHR$(N2);
12Ø FOR S = 1 TO 11
13Ø READ MS
14Ø LPRINT CHR$(MS);
15Ø NEXT S
16Ø NEXT N1
18Ø AS = 16Ø
19Ø FOR NUM = Ø TO 9
2ØØ NT$(NUM) = CHR$(AS + Ø) + CHR$(AS + 1)
21Ø NB$(NUM) = CHR$(AS + 2) + CHR$(AS + 3)
22Ø AS = AS + 4
23Ø NEXT NUM
24Ø BK$ = CHR$(2ØØ)
25Ø LPRINT DP$; L7$
26Ø FOR NUM = Ø TO 9
27Ø LPRINT NT$(NUM);BK$;
28Ø NEXT NUM
29Ø LPRINT
3ØØ FOR NUM = Ø TO 9
31Ø LPRINT NB$(NUM);BK$;
32Ø NEXT NUM
33Ø LPRINT ND$; L12$
34Ø REM ZERO
35Ø DATA 11,Ø,96,16,1Ø4,16,44,3Ø,14,Ø,2,1
36Ø DATA 11,2,1,2,1,6,8,38,88,32,88,32
37Ø DATA 11,3,14,19,14,51,Ø,96,Ø,96,Ø,96
38Ø DATA 11,Ø,32,Ø,48,Ø,28,3,14,3,4,3
39Ø REM ONE
4ØØ DATA 11,Ø,Ø,Ø,Ø,Ø,4,Ø,4,Ø,4,126
41Ø DATA 9,14,114,14,114,14,2,Ø,Ø,Ø,Ø,Ø
42Ø DATA 11,64,Ø,64,Ø,64,Ø,64,32,8Ø,47,8Ø
43Ø DATA 9,47,8Ø,47,64,Ø,64,Ø,64,Ø,Ø,Ø
44Ø REM TWO
45Ø DATA 11,Ø,Ø,Ø,Ø,Ø,14,16,14,Ø,6,Ø
46Ø DATA 11,3,Ø,3,Ø,7Ø,56,7Ø,56,4,24,Ø
47Ø DATA 11,64,Ø,64,32,64,32,8Ø,32,8Ø,4Ø,64
48Ø DATA 11,44,64,38,65,34,65,32,8Ø,32,88,Ø
49Ø REM THREE
5ØØ DATA 11,Ø,Ø,Ø,Ø,Ø,Ø,4,2,4,2,4
51Ø DATA 11,34,84,34,92,34,76,34,68,2,64,Ø
52Ø DATA 11,16,Ø,48,Ø,56,64,48,64,32,64,32
53Ø DATA 11,64,32,64,48,9,54,9,22,9,6,1
54Ø REM FOUR
```

```
55Ø DATA 11,Ø,Ø,Ø,Ø,Ø,Ø,64,36,88,32,16
56Ø DATA 11,Ø,Ø,64,32,64,56,64,6Ø,2,14,Ø
57Ø DATA 11,Ø,8,4,1Ø,5,1Ø,5,8,4,72,4
58Ø DATA 11,88,38,89,38,89,6,73,4,8,6,Ø
59Ø REM FIVE
6ØØ DATA 11,Ø,Ø,Ø,Ø,64,32,84,5Ø,76,34,68
61Ø DATA 1Ø,34,68,34,68,34,68,2,68,2,Ø,Ø
62Ø DATA 1Ø,Ø,32,24,1Ø1,24,97,Ø,64,Ø,64,Ø
63Ø DATA 11,64,Ø,96,1,48,15,48,15,16,15,Ø
64Ø REM 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 REM 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 REM 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 11,14,18,44,19,1Ø8,19,96,1,64,Ø,64
78Ø DATA 11,Ø,96,1,112,15,48,15,16,14,Ø,Ø
79Ø REM 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 REM SPACE
85Ø DATA 11,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø
```

Download utility program

```
4 CLEAR 1ØØØ
5 ME$="E)DIT  P)RINT  Q)UIT
                                 "
6 BL$=STRING$(63," ")
1Ø DIM Z(8,12),MM(11)
2Ø CLS:GOSUB 66Ø
3Ø CS$="@":SC$=CHR$(143):SS$="*"
4Ø A$=INKEY$:IF A$="" THEN 4Ø
5Ø IF A$="Q" THEN CLS:END
6Ø IF A$="P" THEN GOSUB 68Ø:GOTO 4Ø
7Ø IF A$="E" THEN CLS:GOSUB 9Ø:GOSUB 26Ø:GOTO 4Ø
8Ø GOTO 4Ø
```

```
9Ø X=1:Y=1:G=1:H=1:
1ØØ FOR I=1 TO 11:MM(I)=Ø:NEXT I
1Ø5 CLS
12Ø FOR I=Ø TO 7 : PRINT "    ";
125 FOR J=1 TO 11:PRINT "!---";:NEXT J:PRINT "!":IF
   I<7 THEN PRINT:
126 NEXT I
13Ø FOR I=Ø TO 6:PRINT @64*I+64*I+64,2^I;:NEXT I
14Ø PRINT @7Ø,CS$;
15Ø PRINT @18Ø,"<R> RIGHT";
16Ø PRINT @244,"<L> LEFT";
17Ø PRINT @3Ø8,"<U> UP";
18Ø PRINT @372,"<D> DOWN";
19Ø PRINT @436,"<I> INSERT";
2ØØ PRINT @5ØØ,"<C> CLEAR";
21Ø PRINT @564,"<Q> QUIT";
22Ø RETURN
26Ø REM  EDIT LEVEL
27Ø A$=INKEY$:IF A$="" THEN 27Ø
28Ø IF A$="L" THEN GOSUB 39Ø:GOTO 37Ø
29Ø IF A$="R" THEN GOSUB 41Ø:GOTO 37Ø
3ØØ IF A$="D" THEN GOSUB 43Ø:GOTO 37Ø
31Ø IF A$="U" THEN GOSUB 45Ø:GOTO 37Ø
32Ø IF A$="I" THEN GOSUB 47Ø:GOTO 37Ø
33Ø IF A$="C" THEN GOSUB 49Ø:GOTO 37Ø
34Ø IF A$="Q" THEN GOSUB 5ØØ:GOTO 38Ø
37Ø GOTO 27Ø
38Ø RETURN
39Ø GOSUB 92Ø:Y=Y-4:H=H-1:IF Y<1 THEN Y=1:H=1
4ØØ GOSUB 95Ø:RETURN
41Ø GOSUB 92Ø:Y=Y+4:H=H+1:IF Y>41 THEN Y=41:H=11
42Ø GOSUB 95Ø:RETURN
43Ø GOSUB 92Ø:X=X+2:G=G+1:IF X>13 THEN X=13:G=7
44Ø GOSUB 95Ø:RETURN
45Ø GOSUB 92Ø:X=X-2:G=G-1:IF X<1 THEN X=1:G=1
46Ø GOSUB 95Ø:RETURN
47Ø IF Z(G,H-1)=1 OR Z(G,H+1)=1 THEN RETURN
48Ø Z(G,H)=1:PRINT @X*64+Y+5,SS$;:RETURN
49Ø Z(G,H)=Ø:PRINT @X*64+Y+5,CS$;:RETURN
5ØØ REM GET OF EDIT MODE
51Ø IF Z(G,H)=1 THEN PRINT @X*64+Y+5,SC$;
52Ø IF Z(G,H)=Ø THEN PRINT @X*64+Y+Y," ";
524 PRINT @X*64+Y+5, " ";
53Ø FOR I=1 TO 11:FOR J=1 TO 7:
   MM(I)=MM(I)+Z(J,I)*2[(J-1):NEXT J:NEXT I
```

```
54Ø CLS
542 FOR I=1 TO 11:PRINT "M";I;TAB(5);"= ";MM(I):
   NEXT I
55Ø GOSUB 66Ø:RETURN
66Ø FOR I=1 TO 7:FOR J=1 TO 11:Z(I,J)=Ø:NEXT J:
   NEXT I
67Ø PRINT @96Ø,ME$;:RETURN
68Ø REM  PRINT MODE
69Ø PRINT @832,"";:INPUT "NORMAL OR PROPORTIONAL -->
   ";AN$
7ØØ IF AN$="N" THEN PR=Ø:GOTO 75Ø
71Ø IF AN$="P" THEN GOTO 73Ø
72Ø GOSUB 1ØØØ:GOTO 69Ø
73Ø GOSUB 1ØØØ:PRINT @832,"";:INPUT "PROPORTIONAL
   DATA (4-11) --> ";PR
74Ø IF PR<4 OR PR>11 THEN 73Ø
75Ø GOSUB 1ØØØ:PRINT @832,"";:INPUT "IF SHIFTED
   ENTER 1 ELSE ENTER Ø --> ";SH
76Ø IF SH<Ø OR SH>1 THEN 75Ø
77Ø GOSUB 1ØØØ:PRINT @832,"";:INPUT "ASCII  (33-126)
   (16Ø-254) ";AS
78Ø IF (AS<33 OR AS>126) AND (AS<16Ø OR AS>254) THEN
   77Ø
79Ø GOSUB 1ØØØ
8ØØ IF SH=1 THEN SH=16 ELSE SH=Ø
81Ø N1=AS:N2=PR+SH
82Ø FOR I=1 TO 11:MM$=MM$+CHR$(MM(I)):NEXT I
83Ø LPRINT CHR$(27);"*";CHR$(1);CHR$(N1);CHR$(N2);
   MM$
84Ø IF AN$="N" THEN LPRINT CHR$(27);"$";CHR$(1):GOTO
   86Ø
85Ø LPRINT CHR$(27);"X";CHR$(1)
86Ø FOR I=1 TO 2Ø:LPRINT CHR$(N1);" ";:NEXT I:LPRINT
87Ø LPRINT CHR$(14);:FOR I=1 TO 1Ø:LPRINT
   CHR$(N1);:NEXT I:LPRINT
88Ø LPRINT CHR$(15);:FOR I=1 TO 2Ø:LPRINT
   CHR$(N1);:NEXT I:LPRINT
89Ø IF AN$="N" THEN LPRINT CHR$(27);"$";CHR$(Ø):GOTO
   896
894 LPRINT CHR$(27);"X";CHR$(Ø)
896 LPRINT CHR$(27);"@":MM$="":GOSUB 66Ø:RETURN
92Ø IF Z(G,H)=Ø THEN PRINT @X*64+Y+5," ";
93Ø IF Z(G,H)=1 THEN PRINT @X*64+Y+5,SC$;
94Ø RETURN
95Ø IF Z(G,H)=1 THEN PRINT @X*64+Y+5,SS$;
```

```
960 IF Z(G,H)=0 THEN PRINT @X*64+Y+5,CS$;
970 RETURN
1000 PRINT @832,BL$;:RETURN
```

Delta Plot program

```
5 CLEAR 1000
10 CLS:PRINT "":PRINT "":PRINT ""
20 PRINT "THIS PROGRAM TAKES ABOUT 1 MINUTE  TO RUN"
30 PRINT "PLEASE TURN ON YOUR PRINTER AND STAND  BY"
100 DIM BIT%(76,14)
110 MASK%(1)=128     : MASK%(4)=16
120 MASK%(2)=64      : MASK%(5)=8
130 MASK%(3)=32      : MASK%(6)=4
140 LX=20   : LY=20
150 XFAC=72/LX       : YFAC=87/LY
1000 REM PLOT CURVE
1010 RAD=9
1020 X1=19   : Y1=10
1030 FOR ANG=0 TO 360 STEP 10
1040 R1=ANG*6.28/360
1050 X2=RAD*COS(R1)+10  : Y2=RAD*SIN(R1)+10
1060 GOSUB 3000
1070 NEXT ANG
2000 REM
2010 LPRINT CHR$(27) "A" CHR$(6)
2020 FOR ROW=0 TO 14
2030 A$=""
2040 LPRINT CHR$(27);"K";CHR$(75);CHR$(0);
2050 FOR COL=1 TO 75
2060 MM=(BIT%(COL,ROW)
2065 IF MM=12 THEN MM=140
2070 A$=A$ + CHR$(MM)
2090 NEXT COL
2100 LPRINT A$
2110 NEXT ROW
2120 LPRINT CHR$(27) "2"
2130 END
3000 REM DRAW A LINE FROM X1,Y1 TO X2,Y2
3010 XL=X2 - X1       : YL=Y2 - Y1
3020 NX=ABS(XL*XFAC) : NY=ABS(YL*YFAC)
3030 IF NX < NY THEN NX=NY
3040 NS%=INT(NX+1)
3050 DX=XL/NS%        : DY=YL/NS%
3060 FOR I=1 TO NS%
3070 X1=X1 + DX       : Y1=Y1 + DY
```

```
3Ø8Ø GOSUB 4ØØØ
3Ø9Ø NEXT I
31ØØ RETURN
4ØØØ REM PLOT A POINT AT X1,Y1
4Ø1Ø XX=X1 * XFAC    : YY=Y1 * YFAC
4Ø2Ø COL=INT(XX) + 1
4Ø3Ø ROW=INT(YY/6)
4Ø4Ø XIT%=INT(YY-(6*ROW))+1
4Ø5Ø BIT%(COL,ROW)=BIT%(COL,ROW) OR MASK%(XIT%)
4Ø6Ø RETURN
```

Pie chart program

```
4Ø CLS
45 PRINT "            PLEASE STAND BY"
5Ø CLEAR 1ØØØØ
1ØØ ' PIECHART
11Ø ESC$ = CHR$(27)     : LF$ =CHR$(1Ø)
12Ø FF$ = CHR$(12)      : VT$ = CHR$(11)
13Ø EM$ = ESC$ + "E"    : NE$ = ESC$ + "F"
1ØØØ ' Set program constants
1Ø2Ø DIM BIT%(19Ø,36),A$(36),PCT%(25),
   TXT$(42),PXT$(25)
1Ø3Ø MASK%(1) = 128      : MASK%(4) = 16
1Ø4Ø MASK%(2) = 64       : MASK%(5) = 8
1Ø5Ø MASK%(3) = 32       : MASK%(6) = 4
1Ø6Ø LX = 2Ø   : LY = 2Ø
1Ø7Ø XFAC = 19Ø/LX      : YFAC = 216/LY
1Ø8Ø FOR I= 1 TO 42
1Ø85 REM THERE ARE 8Ø SPACES IN TXT$(I)
1Ø9Ø TXT$(I) = "  (80 spaces)  " :
11ØØ NEXT I
111Ø GOSUB 7ØØØ
112Ø CLS:PRINT " THIS PROGRAM TAKES ABOUT 5 MINUTES
   TO RUN"
113Ø PRINT "SO TURN ON YOUR PRINTER AND STAND
   BY....."
2ØØØ ' Plot curve
2Ø1Ø RD = 9
2Ø2Ø X1 = 19   : Y1 = 1Ø
2Ø4Ø FOR ANG% = Ø TO 36Ø STEP 5
2Ø5Ø RANG = ANG%*6.28/36Ø
2Ø6Ø X2 = RD*COS(RANG)+1Ø  : Y2 = RD*SIN(RANG)+1Ø
2Ø7Ø GOSUB 4ØØØ
2Ø8Ø NEXT ANG%
2Ø9Ø FOR PIE = 1 TO NP%
```

```
21ØØ X1 = 1Ø   : Y1 = 1Ø
211Ø TP%=TP%+PCT%(PIE)
212Ø ANG%=36Ø*TP%*.Ø1
213Ø RANG = ANG%*6.28/36Ø
214Ø X2 = RD*COS(RANG)+1Ø  : Y2 = RD*SIN(RANG)+1Ø
215Ø GOSUB 4ØØØ
216Ø GOSUB 6ØØØ
217Ø NEXT PIE
3ØØØ ' SEND BIT IMAGE MAP TO PRINTER
3Ø2Ø FOR ROW% = Ø TO 35
3Ø3Ø A$(ROW%) = ""
3Ø4Ø FOR COL% = 1 TO 19Ø
3Ø45 MM=BIT%(COL%,ROW%)
3Ø47 IF MM=12 THEN MM=14Ø
3Ø5Ø A$(ROW%) = A$(ROW%) + CHR$(MM)
3Ø6Ø NEXT COL%
3Ø8Ø NEXT ROW%
3Ø9Ø LL=LEN (TT$):NN=4Ø-LL/2:FOR I=1 TO NN:LPRINT
   " ";:NEXT I
3Ø91 LPRINT EM$;TI$;NE$;LF$
31ØØ LPRINT  VT$;VT$;VT$
311Ø LPRINT  ESC$;"A";CHR$(3)
312Ø LPRINT TXT$(1);LF$;TXT$(2);LF$;TXT$(3);LF$
313Ø FOR ROW% = Ø TO 35
314Ø LPRINT  "           ";
   ESC$;"K";CHR$(19Ø);CHR$(Ø);
315Ø LPRINT A$(ROW%)
316Ø LPRINT  TXT$(ROW%+4)
318Ø NEXT ROW%
319Ø LPRINT  TXT$(4Ø);LF$
32ØØ LPRINT  TXT$(41);LF$
321Ø LPRINT  TEXT$(42);LF$
322Ø LPRINT  ESC$;"2";FF$
323Ø END
4ØØØ ' DRAW A LINE FROM X1,Y1 TO X2,Y2
4Ø1Ø XL = X2 - X1       : YL = Y2 - Y1
4Ø2Ø NX = ABS(XL*XFAC) : NY = ABS(YL*YFAC)
4Ø3Ø IF NX < NY THEN NX = NY
4Ø4Ø NS% = INT(NX+1)
4Ø5Ø DX = XL/NS%        : DY = YL/NS%
4Ø6Ø FOR I% = 1 TO NS%
4Ø7Ø X1 = X1 + DX       : Y1 = Y1 + DY
4Ø8Ø GOSUB 5ØØØ
4Ø9Ø NEXT I%
411Ø RETURN
```

```
5ØØØ ' PLOT A POINT AT X1,Y1
5Ø1Ø XX = X1 * XFAC    : YY = Y1 * YFAC
5Ø2Ø COL% = INT(XX) + 1
5Ø3Ø ROW% = INT(YY/6)
5Ø4Ø XIT% = INT(YY - ROW% * 6)+1
5Ø5Ø BIT%(COL%,ROW%) = BIT%(COL%,ROW%) OR
   MASK%(XIT%)
5Ø6Ø RETURN
6ØØØ ' Place text fields in the proper location
6Ø1Ø MA%=(ANG%+PA%)/2
6Ø2Ø RANG = MA%*6.28/36Ø
6Ø3Ø X3 = INT(2Ø*SIN(RANG))  :
   Y3 = INT(22*COS(RANG))
6Ø4Ø X4 = 22 + X3      : Y4 = 4Ø + Y3
6Ø5Ø IF MA%>27Ø OR MA%<9Ø THEN MID$(TXT$(X4),Y4) =
   PXT$(PIE) ELSE MID$(TXT$(X4),Y4-
   LEN(PXT$(PIE)))=PXT$(PIE)
6Ø6Ø PA%=ANG%
6Ø7Ø RETURN
7ØØØ ' ACCEPT DATA FROM SCREEN
7Ø1Ø CLS: PRINT : PRINT : PRINT
7Ø2Ø INPUT "ENTER TITLE FOR CHART: ";TT$
7Ø3Ø AS%=Ø       : AL%=1ØØ
7Ø4Ø I=1
7Ø5Ø CLS: PRINT "      ENTER PARAMETERS FOR
   PIE-CHART"
7Ø6Ø PRINT "  TOTAL SO FAR :   ";
7Ø7Ø PRINT USING "###";AS%
7Ø8Ø PRINT "  TOTAL REMAINING: ";
7Ø9Ø PRINT USING "###";AL%
71ØØ PRINT :PRINT :PRINT :PRINT
711Ø INPUT "ENTER PERCENTAGE FOR FIELD:   ";PCT%(I)
712Ø IF PCT%(I)>AL% OR PCT%(I)=Ø THEN PCT%(I)=AL%
713Ø AL%=AL%-PCT%(I)
714Ø AS%=AS%+PCT%(I)
715Ø PRINT :PRINT
716Ø INPUT "ENTER DESCRIPTION OF FIELD:   ";PXT$(I)
717Ø IF LEN(PXT$(I))>15 THEN PRINT "FIELD TOO LONG -
   15 CHAR. MAX": GOTO 716Ø
718Ø IF AL%=Ø GOTO 72ØØ
7185 I=I+1
719Ø GOTO 7Ø5Ø
72ØØ NP%=I
721Ø IF NP%=1 THEN 7Ø3Ø
```

```
7220 CLS
7230 RETURN
```

Appendix E

Osborne, Kaypro and Other CP/M Computers

All that you need to connect Delta to an Osborne 1 or Kaypro computer is a cable. Your Delta dealer can provide the cable that you need.

Setting the switches

When connecting Delta to an Osborne 1, Kaypro, or other CP/M computer, we recommend that you set the DIP switches in Delta as shown below. (Although our chart indicates switch 2-2 set for a parallel interface, a serial interface will work also.)

Recommended DIP Switch Settings for Osborne 1

Switch	Setting	Function
1-1	ON	11 inch page size
1-2	ON	Normal print density
1-3	ON	10 CPI pitch
1-4	ON	Normal characters
1-5	ON	1/6 inch line feed
1-6	ON	U.S.A. Character set
1-7	ON	
1-8	ON	
2-1	ON	Paper-out detector active
2-2	OFF	Parallel interface
2-3	OFF	8-bit interface
2-4	OFF	No auto line feed

When you connect your printer to your Osborne 1 you must use the SETUP program to tell the computer whether Delta is connected to the Osborne 1's serial or parallel interface (either will work).

Osborne 1 Parallel Cable

Delta		Osborne 1	
Pin No.	**Function**	**Pin No.**	**Function**
2	DATA1	1	DATA0
6	DATA5	2	DATA4
3	DATA2	3	DATA1
7	DATA6	4	DATA5
4	DATA3	5	DATA2
8	DATA7	6	DATA6
5	DATA4	7	DATA3
9	DATA8	8	DATA7
1	STROBE	11	STROBE
11	BUSY	15	BUSY
16	SIG GND	16	SIG GND

Kaypro Parallel Cable

Delta		Kaypro	
Pin No.	**Function**	**Pin No.**	**Function**
1	STROBE	1	STROBE
2	DATA1	2	DATA1
3	DATA2	3	DATA2
4	DATA3	4	DATA2
5	DATA4	5	DATA2
6	DATA5	6	DATA2
7	DATA6	7	DATA2
8	DATA7	8	DATA2
9	DATA8	9	DATA8
11	BUSY	11	BUSY
16	SIG GND	16	SIG GND

Using MBASIC

Many CP/M computers use Microsoft BASIC (called MBASIC). If you have a CP/M-80 computer that uses Microsoft BASIC the program listings given here should work with your computer also.

MBASIC is a very close relative of the IBM-Microsoft BASIC used in this book. The only difference is that MBASIC "interprets" CHR$(9) and substitutes a group of spaces to simulate a tab. You can send a horizontal tab to Delta by using CHR$(137) instead of CHR$(9).

Microsoft BASIC uses the "L" prefix on several commands to direct them to the printer. To list programs on the printer, just type LLIST. To direct program output to the printer, use LPRINT in place of PRINT.

Some versions of Microsoft BASIC will add a carriage return and line feed at the end of every 80 (or sometimes 132) characters. To print lines longer than 80 (or 132) characters (as when doing dot graphics) you must define a wider printer width. The following statement will prevent the computer from inserting unwanted codes.

```
1Ø WIDTH LPRINT 255
```

Program listings

The following programs are in Microsoft BASIC for the Osborne 1.

Chart program

```
1ØØ WIDTH LPRINT 255
11Ø GOSUB 1ØØØ
12Ø GOSUB 2ØØØ
13Ø LPRINT "*REGULAR*"
14Ø GOSUB 3ØØØ
15Ø LPRINT "*DOUBLE STRIKE*"
16Ø LPRINT DOUBLE.STRIKE$;
17Ø GOSUB 3ØØØ
18Ø LPRINT "*EMPHASIZED*"
19Ø EMPHASIZED=TRUE
2ØØ GOSUB 3ØØØ
21Ø LPRINT "*DOUBLE STRIKE AND EMPHASIZED*"
22Ø LPRINT DOUBLE.STRIKE$ EMPHASIZED$;
23Ø GOSUB 3ØØØ
24Ø END
1ØØØ REM
1Ø6Ø ITALIC$=CHR$(27) + CHR$(52)
1Ø7Ø ROMAN$=CHR$(27) + CHR$(53)
1Ø9Ø ENLARGED$=CHR$(27) +CHR$(87)+CHR$(1)
11ØØ NOT.ENLARGED$=CHR$(27)+CHR$(87)+CHR$(Ø)
111Ø PICA$=CHR$(27)+CHR$(66)+CHR$(1)
112Ø ELITE$=CHR$(27)+CHR$(66)+CHR$(2)
113Ø CONDENSED$=CHR$(27)+CHR$(66)+CHR$(3)
115Ø EMPHASIZED$=CHR$(27)+CHR$(69)
116Ø NOT.EMPHASIZED$=CHR$(27)+CHR$(70)
117Ø DOUBLE.STRIKE$=CHR$(27)+CHR$(71)
```

```
118Ø NOT.DOUBLE.STRIKE$=CHR$(27)+CHR$(72)
119Ø UNDERLINED$=CHR$(27)+CHR$(45)+CHR$(1)
12ØØ NOT.UNDERLINED$=CHR$(27)+CHR$(45)+CHR$(Ø)
121Ø SUPERSCRIPT$=CHR$(27)+CHR$(83)+CHR$(Ø)
122Ø SUBSCRIPT$=CHR$(27)+CHR$(83)+CHR$(1)
123Ø NOT.SCRIPTED$=CHR$(27)+CHR$(84)
124Ø RESET.ALL$=NOT.EMPHASIZED$+NOT.UNDERLINED$+
   NOT.DOUBLE.STRIKE$
125Ø RESET.ALL$=RESET.ALL$+ROMAN$+PICA$+
   NOT.ENLARGED$
127Ø TRUE=1: FALSE=Ø
128Ø REGULAR.HEADING$=STRING$(27,"*")+"REGULAR"+
   STRING$(27,"*")
129Ø RETURN
2ØØØ REM
2Ø5Ø LPRINT RESET.ALL$
2Ø6Ø LPRINT ENLARGED$"   NORMAL     ENLARGED"
2Ø7Ø LPRINT RESET.ALL$;
2Ø8Ø LPRINT UNDERLINED$;
2Ø9Ø LPRINT CONDENSED$ "CONDENSED ";
21ØØ LPRINT ELITE$     "  ELITE   ";
211Ø LPRINT PICA$      "  PICA    ";
212Ø LPRINT CONDENSED$ "CONDENSED ";
213Ø LPRINT ELITE$     "  ELITE   ";
214Ø LPRINT PICA$"  PICA    ";
215Ø LPRINT RESET.ALL$
216Ø RETURN
3ØØØ REM
3Ø5Ø ITALICS=FALSE:LPRINT ROMAN$;
3Ø6Ø UNDERLINED=FALSE:LPRINT NOT.UNDERLINED$;
3Ø7Ø ENLARGED=FALSE:LPRINT NOT.ENLARGED$;
3Ø8Ø PICA=FALSE
31ØØ LPRINT CONDENSED$;
311Ø GOSUB 35ØØ
312Ø LPRINT ELITE$;
313Ø GOSUB 35ØØ
314Ø LPRINT PICA$;:PICA=TRUE
315Ø GOSUB 35ØØ
317Ø IF ENLARGED=TRUE THEN LPRINT:GOTO 319Ø
318Ø LPRINT ENLARGED$;:ENLARGED=TRUE:GOTO 3Ø8Ø
319Ø IF UNDERLINED=TRUE THEN LPRINT:GOTO 321Ø
32ØØ LPRINT UNDERLINED$;:UNDERLINED=TRUE:GOTO 3Ø7Ø
321Ø IF ITALICS=TRUE THEN LPRINT RESET.ALL$:RETURN
322Ø LPRINT ITALIC$;:ITALICS=TRUE:GOTO 3Ø6Ø
35ØØ REM
```

```
3550 BLANK$=STRING$(6,32):FOUR.DOT$="...."
3560 IF EMPHASIZED=FALSE THEN LPRINT"ABcd";:GOTO
   3610
3570 IF PICA=FALSE THEN LPRINT FOUR.DOT$;:GOTO 3590
3580 LPRINT EMPHASIZED$ "ABcd ";
3590 IF ENLARGED=TRUE THEN LPRINT " ";:ELSE LPRINT
   BLANK$;
3600 RETURN
3610 REM
3620 IF ENLARGED=TRUE THEN LPRINT " ";:RETURN
3630 LPRINT SUPERSCRIPT$; "Xx";
3640 LPRINT SUBSCRIPT$;   "Yy  ";
3650 LPRINT NOT.SCRIPTED$;
3660 RETURN
```

Special character chart program

```
10 FOR J=160 TO 255 STEP 8
20 FOR I=J TO J+7
30 LPRINT I "= "CHR$(I);"  ";
40 NEXT:LPRINT:NEXT
```

Macro program

```
10 LPRINT CHR$(27)"+";
20 LPRINT CHR$(18);
30 LPRINT CHR$(27)"W"CHR$(0);
40 LPRINT CHR$(27)"F";
50 LPRINT CHR$(27)"H";
60 LPRINT CHR$(27) "-" CHR$(0);
70 LPRINT CHR$(27) "T";
80 LPRINT CHR$(27) "5";
90 LPRINT CHR$(30);
```

Bridge hand program

```
10 WIDTH LPRINT 255
20 GOSUB 1000
30 GOSUB 2000
40 GOSUB 3000
50 GOSUB 4000
60 END
1000 REM
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"
```

```
1Ø4Ø CARD$(6)=" 7":CARD$(7)=" 8":CARD$(8)="
   9":CARD$(9)=" 1Ø"
1Ø5Ø CARD$(1Ø)=" J":CARD$(11)=" Q":CARD$(12)="
   K":CARD$(13)=" A"
1Ø6Ø SUIT$(Ø)="S":SUIT$(1)="H":SUIT$(2)="D":
   SUIT$(3)="C"
1Ø7Ø INPUT "Random number seed ";I
1Ø8ØRANDOMIZE I
1Ø9Ø RETURN
2ØØØ REM
2Ø1Ø LPRINT CHR$(27)CHR$(68)CHR$(2Ø)CHR$(4Ø)CHR$(Ø)
2Ø2Ø LPRINT CHR$(27)CHR$(43)CHR$(27)CHR$(36)CHR$(Ø)
   CHR$(27)CHR$(69)CHR$(3Ø)
2Ø3Ø LPRINT CHR$(27)CHR$(42)CHR$(Ø)
2Ø4Ø FOR I=1 TO 4
2Ø5Ø LPRINT CHR$(27)CHR$(42)CHR$(1);
2Ø6Ø FOR J=1 TO 13
2Ø7Ø READ X:LPRINT CHR$(X);
2Ø8Ø NEXT J
2Ø9Ø NEXT I
21ØØ LPRINT
211Ø RETURN
212Ø DATA 72,11,4,1Ø,2Ø,1Ø,52,72,52,1Ø,2Ø,1Ø,4
213Ø DATA 83,11,16,8,2Ø,8,86,41,86,8,2Ø,8,16
214Ø DATA 67,11,8,16,8,18,65,62,65,18,8,16,8
215Ø DATA 68,11,8,Ø,28,Ø,62,65,62,Ø,28,Ø,8
3ØØØ REM
3Ø1Ø FOR CARD=1 TO 52
3Ø2Ø X=INT(RND*4+1)
3Ø3Ø IF HAND(X)=13 THEN 3Ø2Ø ELSE HAND(X)=HAND(X)+1
3Ø4Ø DECK(CARD)=X
3Ø5Ø NEXT CARD
3Ø6Ø RETURN
4ØØØ REM
4Ø1Ø LPRINT CHR$(27) "!" SPC(2Ø) "NORTH"
4Ø2Ø LPRINT CHR$(27) "$" CHR$(1) CHR$(27)CHR$(7Ø);
4Ø3Ø HAND=1
4Ø4Ø FOR SUIT=Ø to 3
4Ø5Ø LPRINT SPC(2Ø);
4Ø6Ø GOSUB 43ØØ
4Ø7Ø LPRINT
4Ø8Ø NEXT SUIT
4Ø9Ø LPRINT CHR$(27) "!" "WEST" SPC(4Ø) "EAST"
41ØØ LPRINT CHR$(27) "$" CHR$(1) CHR$(27)CHR$(7Ø);
411Ø FOR SUIT=Ø TO 3
```

```
4120 HAND=2
413Ø GOSUB 43ØØ
414Ø LPRINT TAB(45);
415Ø HAND =3
416Ø GOSUB 43ØØ
417Ø LPRINT
418Ø NEXT SUIT
419Ø LPRINT CHR$(27) "!" SPC(2Ø) "SOUTH"
42ØØ LPRINT CHR$(27) "$" CHR$(1)CHR$(27)CHR$(7Ø);
421Ø HAND=4
4220 FOR SUIT=Ø TO 3
4230 LPRINT SPC(2Ø);
424Ø GOSUB 43ØØ
425Ø LPRINT
426Ø NEXT SUIT
4270 LPRINT CHR$(27) "$" CHR$(Ø)CHR$(27)CHR$(7Ø)
428Ø RETURN
43ØØ LPRINT SUIT$(SUIT);
431Ø FOR CARD=13 TO 1 STEP -1
432Ø IF DECK(SUIT*13+CARD)=HAND THEN LPRINT
   CARD$(CARD);
433Ø NEXT CARD
434Ø RETURN
```

Numeral program

```
3Ø DEF.DOWN.CHAR$=CHR$(27)+CHR$(42)+CHR$(1)
4Ø DOWN.CHAR.PROP$=CHR$(27)+CHR$(88)+CHR$(1)
5Ø NOT.DOWN.CHAR.PROP$=CHR$(27)+CHR$(88)+CHR$(Ø)
6Ø LINE.7$=CHR$(27)+CHR$(49):LINE.12$=CHR$(27)+
   CHR$(5Ø)
7Ø FOR N1=16Ø TO 2ØØ
8Ø LPRINT DEF.DOWN.CHAR$;
9Ø LPRINT CHR$(N1);
1ØØ READ N2
11Ø LPRINT CHR$(N2);
12Ø FOR S=1 TO 11
13Ø READ MS
14Ø LPRINT CHR$(MS);
15Ø NEXT S
16Ø NEXT N1
18Ø ASCII=16Ø
19Ø FOR NUM=Ø to 9
2ØØ NUMERAL.TOP$(NUM)=CHR$(ASCII+Ø)+CHR$(ASCII+1)
21Ø NUMERAL.BOT$(NUM)=CHR$(ASCII+2)+CHR$(ASCII+3)
22Ø ASCII=ASCII+4
```

```
23Ø NEXT NUM
24Ø BLANK$=CHR$(2ØØ)
25Ø LPRINT DOWN.CHAR.PROP$;LINE.7$
26Ø FOR NUM=Ø TO 9
27Ø LPRINT NUMERAL.TOP$(NUM);BLANK$;
28Ø NEXT NUM
29Ø LPRINT
3ØØ FOR NUM=Ø TO 9
31Ø LPRINT NUMERAL.BOT$(NUM);BLANK$;
32Ø NEXT NUM
33Ø LPRINT NOT.DOWN.CHAR.PROP$;LINE.12$
34Ø REM ZERO
35Ø DATA 11,Ø,96,16,1Ø4,16,44,3Ø,14,Ø,2,1
36Ø DATA 11,2,1,2,1,6,8,38,88,32,88,32
37Ø DATA 11,3,12,19,12,51,Ø,96,Ø,96,Ø,96
38Ø DATA 11,Ø,32,Ø,48,Ø,28,3,12,3,4,3
39Ø REM ONE
4ØØ DATA 11,Ø,Ø,Ø,Ø,Ø,4,Ø,4,Ø,4,126
41Ø DATA 1Ø,12,114,12,114,12,2,Ø,Ø,Ø,Ø,Ø
42Ø DATA 11,64,Ø,64,Ø,64,Ø,64,32,8Ø,47,8Ø
43Ø DATA 1Ø,47,8Ø,47,64,Ø,64,Ø,64,Ø,Ø,Ø
44Ø REM TWO
45Ø DATA 11,Ø,Ø,Ø,Ø,Ø,12,16,14,Ø,6,Ø
46Ø DATA 11,3,Ø,3,Ø,7Ø,56,7Ø,56,4,24,Ø
47Ø DATA 11,64,Ø,64,32,64,32,8Ø,32,8Ø,4Ø,64
48Ø DATA 11,44,64,38,65,34,65,32,8Ø,32,88,Ø
49Ø REM THREE
5ØØ DATA 11,Ø,Ø,Ø,Ø,Ø,Ø,4,2,4,2,4
51Ø DATA 11,34,84,34,92,34,76,34,68,2,64,Ø
52Ø DATA 11,16,Ø,48,Ø,56,64,48,64,32,64,32
53Ø DATA 11,64,32,64,48,1Ø,54,1Ø,22,1Ø,6,1
54Ø REM FOUR
55Ø DATA 11,Ø,Ø,Ø,Ø,Ø,Ø,64,36,88,32,16
56Ø DATA 11,Ø,Ø,64,32,64,56,64,6Ø,2,12,Ø
57Ø DATA 11,Ø,8,4,1Ø,5,1Ø,5,8,4,72,4
58Ø DATA 11,88,38,89,38,89,6,73,4,8,6,Ø
59Ø REM FIVE
6ØØ DATA 11,Ø,Ø,Ø,Ø,64,32,84,5Ø,76,34,68
61Ø DATA 1Ø,34,68,34,68,34,68,2,68,2,Ø,Ø
62Ø DATA 1Ø,Ø,32,24,1Ø1,24,97,Ø,64,Ø,64,Ø
63Ø DATA 11,64,Ø,96,1,48,15,48,15,16,15,Ø
64Ø REM 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
```

```
680 DATA 11,Ø,96,Ø,112,1,62,1,3Ø,1,14,Ø
690 REM SEVEN
7ØØ DATA 11,Ø,16,8,6,8,6,8,6,8,6,8
71Ø DATA 1Ø,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 1Ø,6,1,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø
74Ø REM 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 REM 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 REM SPACE
85Ø DATA 11,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,
```

Download utility program

```
1Ø DIM Z(8,12),MM(11)
15 DEF FNLOCATE$(X,Y)=CHR$(27)+CHR$(61)+CHR$(X)+
   CHR$(Y+32)
2Ø PRINT CHR$(26);:GOSUB 66Ø
3Ø CS$="<>":SC$="[]":BIT=Ø:SS$="OO"
4Ø A$=INKEY$:IF A$="" THEN 4Ø
5Ø IF A$="Q" THEN PRINT CHR$(26):END
6Ø IF A$="P" THEN GOSUB 68Ø: GOTO 4Ø
7Ø IF A$="E" THEN PRINT CHR$(26):GOSUB 9Ø:GOSUB 26Ø:
   GOTO 4Ø
8Ø GOTO 4Ø
9Ø REM  EDIT MODE
95 H=1:G=1:X=1:Y=1
96 FOR I=1 TO 11:MM(I)=Ø:NEXT I
1ØØ PRINT CHR$(26)
11Ø PRINT
12Ø PRINT "     M1 M2 M3 M4 M5 M6 M7 M8 M9 M1Ø M11"
13Ø for I=Ø to 7:PRINT "      ";:FOR J=1 TO 11:
14Ø PRINT "!==";:NEXT J:PRINT "!":IF I<7 THEN PRINT
   2^I
15Ø NEXT I
16Ø PRINT FNLOCATE$(2Ø,1);"R)IGHT L)EFT U)P D)OWN
   C)LEAR Q)UIT"
18Ø RETURN
26Ø REM **** SINGLE CHARACTER INPUT @ EDIT LEVEL****
```

```
27Ø PRINT FNLOCATE$(4,7);:PRINT CS$;:PRINT
   FNLOCATE$(2Ø,4Ø);
28Ø A$=INKEY$: IF A$="" THEN 28Ø
3ØØ IF A$="L" THEN GOSUB 39Ø:GOTO 37Ø
31Ø IF A$="R" THEN GOSUB 41Ø:GOTO 37Ø
32Ø IF A$="D" THEN GOSUB 43Ø:GOTO 37Ø
33Ø IF A$="U" THEN GOSUB 45Ø:GOTO 37Ø
34Ø IF A$="I" THEN GOSUB 47Ø:GOTO 37Ø
35Ø IF A$="C" THEN GOSUB 49Ø:GOTO 37Ø
36Ø IF A$="Q" THEN GOSUB 5ØØ:GOTO 38Ø
37Ø GOTO 28Ø
38Ø RETURN: REM ****END OF INPUT****
39Ø GOSUB 92Ø:Y=Y-3:H=H-1:IF Y<1 THEN Y=1:H=1
4ØØ GOSUB 95Ø : RETURN
41Ø GOSUB 92Ø:Y=Y+3:H=H+1:IF Y>31 THEN Y=31:H=11
42Ø GOSUB 95Ø:RETURN
43Ø GOSUB 92Ø:X=X+2:G=G+1:IF X>13 THEN X=13:G=7
44Ø GOSUB 95Ø:RETURN
45Ø GOSUB 92Ø:X=X-2:G=G-1:IF X<1 THEN X=1:G=1
46Ø GOSUB 95Ø:RETURN
47Ø IF Z(G,H-1)=1 OR Z(G,H+1)=1 THEN RETURN
48Ø Z(G,H)=1:PRINT FNLOCATE$(X+3,Y+6); SS$;:RETURN
49Ø Z(G,H)=Ø:PRINT FNLOCATE$(X+3,Y+6); CS$;:RETURN
5ØØ REM ****GET OUT OF EDIT MODE****
52Ø IF Z(G,H)=1 THEN PRINT FNLOCATE$(X+3,Y+6);
   SC$;:GOTO 54Ø
53Ø IF Z(G,H)=Ø THEN PRINT FNLOCATE$(X+3,Y+6); "  ";
54Ø REM **** PRINT THE COLUMN - VALUES****
55Ø FOR I=1 TO 11: FOR J=1 TO 7
56Ø MM(I)=MM(I)+Z(J,I)*2^(J-1):NEXT J:NEXT I
57Ø J=Ø:FOR I=1 TO 11:PRINT FNLOCATE$(19,6+J);
   RIGHT$(STR$(MM(I)),3);:J=J+3:NEXT I
572 PRINT
575 PRINT  FNLOCATE$(2Ø,1);STRING$(45," ");
58Ø GOSUB 66Ø:RETURN
66Ø FOR I=1 TO 7:FOR J=1 TO 11:Z(I,J)=Ø:NEXT J:NEXT
   I
67Ø PRINT FNLOCATE$(22,2);:PRINT "E) EDIT    P)
   PRINTER    Q) QUIT  ";:RETURN
68Ø REM ****PRINT MODE****
69Ø PRINT FNLOCATE$(2Ø,1);:INPUT "NORMAL OR
   PROPORTIONAL    (N/P)   ";AN$
7ØØ IF AN$="N" THEN PR=Ø:GOTO 75Ø
71Ø IF AN$="P" THEN GOTO 73Ø
72Ø GOTO 69Ø
```

```
73Ø GOSUB 2ØØØ:PRINT FNLOCATE$(2Ø,1);:INPUT "THE
   PROPORTIONAL DATA  (4-11)    "; PR
74Ø IF PR<4 OR PR>11 THEN 73Ø
75Ø GOSUB 2ØØØ:PRINT FNLOCATE$(2Ø,1);:INPUT "IF
   SHIFTED  ENTER  1 ELSE ENTER Ø  ";SH
76Ø IF SH<Ø OR SH>1 THEN GOTO 75Ø
77Ø GOSUB 2ØØØ:PRINT FNLOCATE$(2Ø,1);:INPUT "ASCII
   CODE (33-126 OR 16Ø-254)   ";AS
78Ø IF (AS<33 OR AS>126) AND (AS<16Ø OR AS>254) THEN
   77Ø
785 PRINT
79Ø PRINT FNLOCATE$(2Ø,1);STRING$(5Ø," ")
8ØØ IF SH=1 THEN SH=16 ELSE SH=Ø
81Ø N1=AS:N2=PR+SH
82Ø FOR I=1 TO 11:MM$=MM$+CHR$(MM(I)):NEXT I
83Ø LPRINT
   CHR$(27);"*";CHR$(1);CHR$(N1);CHR$(N2);MM$
84Ø IF AN$="N" THEN LPRINT CHR$(27);"$";CHR$(1):GOTO
   86Ø
85Ø LPRINT CHR$(27);"X";CHR$(1)
86Ø FOR I=1 TO 2Ø:LPRINT CHR$(N1);" ";:NEXT I:LPRINT
87Ø LPRINT CHR$(14);:FOR I=1 TO 1Ø:LPRINT CHR$(N1);"
   ";:NEXT I:LPRINT CHR$(2Ø)
88Ø LPRINT CHR$(15);:FOR I=1 to 2Ø:LPRINT CHR$(N1);"
   ";: NEXT I:LPRINT CHR$(18)
89Ø IF AN$="N" THEN LPRINT CHR$(27);"$";CHR$(Ø):GOTO
   91Ø
9ØØ LPRINT CHR$(27);"X";CHR$(Ø)
91Ø LPRINT CHR$(27);"@":MM$="":RETURN:REM ****END OF
   PRINT MODE****
92Ø IF Z(G,H)=Ø THEN PRINT FNLOCATE$(X+3,Y+6); "   ";
93Ø IF Z(G,H)=1 THEN PRINT FNLOCATE$(X+3,Y+6); SC$;
94Ø RETURN
95Ø IF Z(G,H)=1 THEN PRINT FNLOCATE$(X+3,Y+6); SS$;
96Ø IF Z(G,H)=Ø THEN PRINT FNLOCATE$(X+3,Y+6); CS$;
97Ø RETURN
2ØØØ PRINT FNLOCATE$(2Ø,1);STRING$(5Ø," ")
2Ø1Ø RETURN
```

Delta Plot program

```
1Ø PRINT CHR$(26)
2Ø PRINT "":PRINT "":PRINT ""
3Ø PRINT "THIS PROGRAM TAKES ABOUT TWO"
4Ø PRINT "MINUTES TO RUN, PLEASE TURN"
5Ø PRINT "ON YOUR PRINTER AND STAND BY"
```

```
100 REM DELTA-PLOT
110 DIM BIT%(76,14)
1000 REM SET PROGRAM CONSTANTS
1010 MASK%(1)=128:MASK%(4)=16
1020 MASK%(2)=64 :MASK%(5)=8
1030 MASK%(3)=32 :MASK%(6)=4
1040 LX=20        :LY=20
1050 XFAC=72/LX   :YFAC=87/LY
2000 REM PLOT CURVE
2010 RAD=9
2020 X1=19  :Y1=10
2030 FOR ANG%=0 TO 360 STEP 10
2040 R1=ANG%*6.28/360
2050 X2=RAD*COS(R1)+10 :Y2=RAD*SIN(R1)+10
2060 GOSUB 4000
2070 NEXT ANG%
3000 REM SEND BIT IMAGE MAP TO PRINTER
3010 LPRINT CHR$(27):"A";CHR$(6)
3020 FOR ROW%=0 TO 14
3030 A$=""
3040 LPRINT CHR$(27);"K";CHR$(75);CHR$(0);
3050 FOR COL% = 1 to 75
3060 LPRINT CHR$(BIT%(COL%,ROW%));
3070 NEXT COL%
3080 LPRINT
3090 NEXT ROW%
3100 LPRINT CHR$(27);"A";CHR$(12)
3110 END
4000 REM DRAW A LINE FROM X1,Y1 TO X2,Y2
4010 XL=X2-X1 :YL=Y2-Y1
4020 NX=ABS(XL*XFAC) : NY=ABS(YL*YFAC)
4030 IF NX<NY THEN NX=NY
4040 NS%=INT(NX+1)
4050 DX=XL/NS%   : DY=YL/NS%
4060 FOR I%=1 TO NS%
4070 X1=X1+DX    :   Y1=Y1+DY
4080 GOSUB 5000
4090 NEXT I%
4100 RETURN
5000 REM PLOT A POINT AT X1,Y1
5010 XX=X1*XFAC    :YY=Y1*YFAC
5020 COL%=INT(XX)+1
5030 ROW%=INT(YY/6)
5040 XIT%=INT(YY-ROW%*6)+1
```

```
5050 BIT%(COL%,ROW%)=BIT%(COL%,ROW%) OR MASK%(XIT%)
5060 RETURN
```

Pie chart program

```
110 ESC$=CHR$(27):LF$=CHR$(10)
120 FF$=CHR$(12):VTAB$=CHR$(11)
130 EMPHASIZED$=ESC$="E":NOT.EMPHASIZED$=ESC$+"F"
1020 DIM BIT%(190,36),A$(36),PCT%(25),TEXT$(42),
   PIECETEXT$(25)
1030 MASK%(1)=128:MASK%(4)=16
1040 MASK%(2)=64 :MASK%(5)=8
1050 MASK%(3)=32 :MASK%(6)=4
1060 LX=20:LY=20
1070 LXFAC=190/LX:LYFAC=216/LY
1080 FOR I=1 TO 42
1090 TEXT$(I)=SPACE$(80)
1100 NEXT I
1110 GOSUB 7000
1120 PRINT CHR$(26)
1130 PRINT "THIS PROGRAM TAKES ABOUT FIVE MINUTES"
1140 PRINT "PLEASE HAVE YOUR PRINTER READY AND "
1150 PRINT "STAND BY "
2010 RAD=9
2020 X1=19:Y1=10
2030 PRINT " ";
2040 FOR ANG%=0 TO 360 STEP 5
2050 RANG=ANG%*6.28/360
2060 X2=RAD*COS(RANG)+10:Y2=RAD*SIN(RANG)+10
2070 GOSUB 4000
2080 NEXT ANG%
2090 FOR PIECE%=1 TO NUMBER.PIECES%
2100 X1=10:Y1=10
2110 TOTAL.PCT%=TOTAL.PCT%+PCT%(PIECE%)
2120 ANG%=360*TOTAL.PCT%*.01
2130 RANG=ANG%*6.28/360
2140 X2=RAD*COS(RANG)+10:Y2=RAD*SIN(RANG)+10
2150 GOSUB 4000
2160 GOSUB 6000
2170 NEXT PIECE%
3010 PRINT
3020 FOR ROW%=0 TO 35
3030 A$(ROW%)=""
3040 FOR COL%=1 TO 190
3050 A$(ROW%)=A$(ROW%)+CHR$(BIT%(COL%,ROW%))
3060 NEXT COL%
```

```
3Ø7Ø PRINT CHR$(176);CHR$(176);
3Ø8Ø NEXT ROW%
3Ø9Ø PRINT
3Ø91 LPRINT SPACE$(4Ø-LEN(TITLE$)/
   2);EMPHASIZED$;TITLE$;NOT. EMPHASIZED$;LF$
31ØØ LPRINT VTAB$;VTAB$;VTAB$
311Ø LPRINT ESC$;"A";CHR$(3)
312Ø LPRINT TEXT$(1);LF$;TEXT$(2);LF$;TEXT$(3):LF$
313Ø FOR ROW%=Ø TO 35
314Ø LPRINT "
   "ESC$;"K";CHR$(19Ø)CHR$(Ø);
315Ø LPRINT A$(ROW%)
316Ø LPRINT TEXT$(ROW%+4)
317Ø LPRINT CHR$(176);CHR$(176);
318Ø NEXT ROW%
319Ø LPRINT TEXT$(4Ø);LF$
32ØØ LPRINT TEXT$(41);LF$
321Ø LPRINT TEXT$(42);LF$
322Ø LPRINT ESC$;"2";FF$
323Ø END
4ØØØ '
4Ø1Ø XL=X2-X1:YL=Y2-Y1
4Ø2Ø NX=ABS(XL*LXFAC):NY=ABS(YL*LYFAC)
4Ø3Ø IF NX<NY THEN NX=NY
4Ø4Ø NS%=INT(NX+1)
4Ø5Ø DX=XL/NS%:DY=YL/NS%
4Ø6Ø FOR I%=1 TO NS%
4Ø7Ø X1=X1+DX:Y1=Y1+DY
4Ø8Ø GOSUB 5ØØØ
4Ø9Ø NEXT I%
41ØØ PRINT CHR$(29);CHR$(205);CHR$(175);
411Ø RETURN
5ØØØ '
5Ø1Ø XX=X1*LXFAC:YY=Y1*LYFAC
5Ø2Ø COL%=INT(XX)+1
5Ø3Ø ROW%=INT(YY/6)
5Ø4Ø XIT%=INT(YY-ROW%*6)+1
5Ø5Ø BIT%(COL%,ROW%)=BIT%(COL%,ROW%) OR MASK%(XIT%)
5Ø6Ø RETURN
6ØØØ REM
6Ø1Ø MIDANG%=(ANG%+PREVANG%)/2
6Ø2Ø RANG=MIDANG%*6.28/36Ø
6Ø3Ø X3=INT(2Ø*SIN(RANG)):Y3=INT(22*COS(RANG))
6Ø4Ø X4=22+X3:Y4=4Ø+Y3
```

```
6Ø5Ø IF MIDANG%>27Ø OR MIDANG%<9Ø THEN
   MID$(TEXT$(X4),Y4)=PIECETEXT$(PIECE%) ELSE
   MID$(TEXT$(X4),Y4-LEN(PIECETEXT$(PIECE%)))
   =PIECETEXT$(PIECE%)
6Ø6Ø PREVANG%=ANG%
6Ø7Ø RETURN
7ØØØ '
7Ø1Ø PRINT CHR$(26):PRINT:PRINT:PRINT
7Ø2Ø INPUT"ENTER TITLE FOR CHART: ";TITLE$
7Ø3Ø AMT.SOFAR%=Ø;AMT.LEFT%=1ØØ
7Ø4Ø FOR I=1 TO 24
7Ø5Ø PRINT CHR$(26);"          ENTER PARAMETERS FOR
   PIE CHART"
7Ø6Ø PRINT        "          TOTAL SO FAR";
7Ø7Ø PRINT AMT.SOFAR%
7Ø8Ø PRINT        "          TOTAL REMAINING";
7Ø9Ø PRINT AMT.LEFT%
71ØØ PRINT:PRINT:PRINT:PRINT
711Ø INPUT "ENTER PERCENTAGE FOR FIELD:  ";PCT%(I)
712Ø IF PCT%(I)>AMT.LEFT% OR PCT%(I)=Ø THEN
   PCT%(I)=AMT.LEFT%
713Ø AMT.LEFT%=AMT.LEFT%-PCT%(I)
714Ø AMT.SOFAR%=AMT.SOFAR%+PCT%(I)
715Ø PRINT:PRINT
716Ø INPUT "ENTER DESCRIPTION OF FIELD:
   ";PIECETEXT$(I)
717Ø IF LEN(PIECETEXT$(I))>15 THEN PRINT "FIELD TOO
   LONG - 15 CHAR. MAX":GOTO 716Ø
718Ø IF AMT.LEFT%=Ø THEN 72ØØ
719Ø NEXT I
72ØØ NUMBER.PIECES%=I
721Ø IF NUMBER.PIECES%=Ø THEN 7Ø3Ø
722Ø PRINT CHR$(26)
723Ø RETURN
```

Appendix F

Atari 400/800 Computers

The best way to connect your Atari to Delta is with the **Universal/Atari Parallel Interface** by Star Micronics. It comes complete with its own cable. Or Delta will connect to the Atari 850 interface, using a cable that is available from your dealer.

Setting the switches

We recommend that you set the DIP switches in Delta as shown below when connecting it to an Atari 400 or 800.

Recommended DIP Switch Settings for Atari

Switch	Setting	Function
1-1	ON	11 inch page size
1-2	ON	Normal print density
1-3	ON	10 CPI pitch
1-4	ON	Normal characters
1-5	ON	1/6 inch line feed
1-6	ON	U.S.A. Character set
1-7	ON	
1-8	ON	
2-1	ON	Paper-out detector active
2-2	OFF	Parallel interface
2-3	OFF	8-bit interface
2-4	ON	Auto line feed

Using Atari BASIC

While the Atari computers don't have any real problems in using the full capabilities of Delta, there are a couple of fairly unique things to keep in mind. Atari BASIC requires that *all* strings be dimensioned. The maximum string length is 99 characters, so Atari users must break up their dot graphics strings into 99 character sections.

Atari 850 Interface Module Parallel Cable

Delta		Atari 850	
Pin No.	**Function**	**Pin No.**	**Function**
1	STROBE	1	STROBE
2	DATA1	2	DATA1
3	DATA2	3	DATA2
4	DATA3	4	DATA3
5	DATA4	5	DATA4
6	DATA5	6	DATA5
7	DATA6	7	DATA6
8	DATA7	8	DATA7
16	SIG GND	11	SIG GND
11	BUSY	13	BUSY
9	DATA8	15	DATA8

To join two strings together, as when building a string of dot graphics data, the following format must be used:

```
A$(LEN(A$)+1)=B$
```

(This adds B$ to the end of A$.)

The Atari adds spaces to print lines when you use the LPRINT command. We recommend that you use the PRINT # command instead. You must open the printer as a device first. For example:

```
1Ø OPEN #4,8,Ø,"P"
2Ø PRINT #4;"TESTING"
```

It's a good idea to close the printer when your program is done using it, like this:

```
9Ø CLOSE #4
```

Atari BASIC also requires that you use semicolons between elements in a print statement where most BASICs will accept a space. Your print commands must look like this:

```
40 PRINT CHR$(27); "B"; CHR$(3); "CONDENSED"
```

Listing programs

Listing BASIC programs to Delta from an Atari computer is relatively easy; just add "P:" to the normal LIST command so that it looks like this:

```
LIST "P:"
```

Program listings

The following programs were translated to work with the Atari 400 and 800 computers.

Chart program

```
10 REM ATARI 400 & 800 & 1200XL
20 DIM BL$(6),FD$(4)
30 TRUE=1:FALSE=0
100 OPEN #4,8,0,"P"
120 GOSUB 1000
130 PRINT #4;"*REGULAR*"
140 GOSUB 2000
150 PRINT #4;"*DOUBLE STRIKE*"
160 PRINT #4;CHR$(27);"G";
170 GOSUB 2000
180 PRINT #4;"*EMPHASIZED*":
190 EM=TRUE
200 GOSUB 2000
210 PRINT #4;"*DOUBLE STRIKE & EMPHASIZED*"
220 PRINT #4;CHR$(27);"G";CHR$(27);"E";
230 GOSUB 2000
240 CLOSE #4
250 END
1000 PRINT #4;CHR$(27);"@"
1010 PRINT #4;CHR$(14);"   NORMAL      ENLARGED   "
1020 PRINT #4;CHR$(27);"-";CHR$(1);
1030 PRINT #4;CHR$(27);"B";CHR$(3);"CONDENSED ";
1040 PRINT #4;CHR$(27);"B";CHR$(2);"  ELITE   ";
1050 PRINT #4;CHR$(27);"B";CHR$(1);"  PICA    ";
1060 PRINT #4;CHR$(27);"B";CHR$(3);"CONDENSED ";
1070 PRINT #4;CHR$(27);"B";CHR$(2);"  ELITE   ";
```

```
1Ø8Ø PRINT #4;CHR$(27);"B";CHR$(1);"  PICA    "
1Ø9Ø PRINT #4;CHR$(27);"@":RETURN
2ØØØ IT=FALSE:PRINT #4;CHR$(27);"5";
2Ø1Ø UN=FALSE:PRINT #4;CHR$(27);"-";CHR$(Ø);
2Ø2Ø EN=FALSE:PRINT #4;CHR$(27);"W";CHR$(Ø);
2Ø3Ø PI=FALSE
2Ø4Ø PRINT #4;CHR$(27);"B";CHR$(3);:GOSUB 213Ø
2Ø5Ø PRINT #4;CHR$(27);"B";CHR$(2);:GOSUB 213Ø
2Ø6Ø PRINT #4;CHR$(27);"B";CHR$(1);:PI=TRUE:GOSUB
   213Ø
2Ø7Ø IF EN=TRUE THEN PRINT #4:GOTO 2Ø9Ø
2Ø8Ø PRINT #4;CHR$(27);"W";CHR$(1);:EN=TRUE:GOTO
   2Ø3Ø
2Ø9Ø IF UN=TRUE THEN PRINT #4:GOTO 211Ø
21ØØ PRINT #4;CHR$(27);"-";CHR$(1);:UN=TRUE:GOTO
   2Ø2Ø
211Ø IF IT=TRUE THEN PRINT #4;CHR$(27);"@":RETURN
212Ø PRINT #4;CHR$(27);"4";:IT=TRUE:GOTO 2Ø1Ø
213Ø BL$="      ":FD$="...."
214Ø IF EM=FALSE THEN PRINT #4;"ABcd";:GOTO 219Ø
215Ø IF PI=FALSE THEN PRINT #4;FD$;:GOTO 217Ø
216Ø PRINT #4;"ABcd";
217Ø IF EN=TRUE THEN PRINT #4;" ";:RETURN
218Ø IF EN=FALSE THEN PRINT #4;BL$;:RETURN
219Ø IF EN=TRUE THEN PRINT #4;" ";:RETURN
22ØØ PRINT #4;CHR$(27);"S";CHR$(Ø);"Xx";
221Ø PRINT #4;CHR$(27);"S";CHR$(1);"Yy";"  ";
222Ø PRINT #4;CHR$(27);"T";
223Ø RETURN
```

Special character chart program

```
1Ø REM PRINT SPECIAL CHAR.SET
2Ø OPEN #4,8,Ø, "P"
3Ø FOR J=16Ø to 255 STEP 8
4Ø FOR I=J TO J+7
5Ø PRINT #4;I;"= ";CHR$(I);"   ";
6Ø NEXT I:PRINT #4:NEXT J
7Ø CLOSE #4
```

Macro program

```
5 REM DEFINE MACRO INSTRUCTION
1Ø OPEN #4,8,Ø,"P"
2Ø PRINT #4,CHR$(27);"+";
3Ø PRINT #4,CHR$(18);
```

```
4Ø PRINT #4,CHR$(27);"W";CHR$(Ø);
5Ø PRINT #4,CHR$(27);"F";
6Ø PRINT #4,CHR$(27);"H";
7Ø PRINT #4,CHR$(27);"-";CHR$(Ø);
8Ø PRINT #4,CHR$(27);"T";
9Ø PRINT #4,CHR$(27);"5";
95 PRINT #4,CHR$(30)
```

Bridge hand program

```
1Ø OPEN #4,8,Ø,"P"
2Ø GOSUB 1ØØØ
3Ø GOSUB 2ØØØ
4Ø GOSUB 3ØØØ
5Ø GOSUB 4ØØØ
6Ø CLOSE #4
7Ø END
1ØØØ REM INITIALIZE VARIABLES
1Ø1Ø DIM HA(4),DE(52),CA$(5Ø),SU$(2Ø)
1Ø2Ø SU$="SHDC"
1Ø3Ø CA$="   2  3  4  5  6  7  8  9  10 J  Q  K  A
   "
1Ø35 FOR I=Ø TO 4:HA(I)=Ø:NEXT I
1Ø4Ø RETURN
2ØØØ REM INITIALIZE PRINTER
2Ø1Ø PRINT #4;CHR$(27);CHR$(68);CHR$(2Ø);CHR$(4Ø);
   CHR$(Ø)
2Ø2Ø PRINT #4;CHR$(27);CHR$(43);CHR$(27);CHR$(36);
   CHR$(Ø);
2Ø3Ø PRINT #4;CHR$(27);CHR$(69);CHR$(3Ø)
2Ø35 PRINT #4;CHR$(27);CHR$(42);CHR$(Ø)
2Ø4Ø FOR I=1 TO 4
2Ø5Ø PRINT #4;CHR$(27);CHR$(42);CHR$(1);
2Ø6Ø FOR J=1 TO 13
2Ø7Ø READ X:PRINT #4;CHR$(X);
2Ø8Ø NEXT J
2Ø9Ø NEXT I
21ØØ PRINT #4
211Ø RETURN
212Ø DATA 72,11,4,1Ø,2Ø,1Ø,52,72,52,1Ø,2Ø,1Ø,4
213Ø DATA 83,11,16,8,2Ø,8,86,41,86,8,2Ø,8,16
214Ø DATA 67,11,8,16,8,18,65,62,65,18,8,16,8
215Ø DATA 68,11,8,Ø,28,Ø,62,65,62,Ø,28,Ø,8
3ØØØ REM DEAL CARD
3Ø1Ø FOR CA=1 TO 52
```

```
3Ø2Ø X=INT(RND(Ø)*4+1)
3Ø3Ø IF HA(X)=13 THEN 3Ø2Ø
3Ø35 HA(X)=HA(X)+1
3Ø4Ø DE(CA)=X
3Ø5Ø NEXT CA
3Ø6Ø RETURN
4ØØØ REM PRINT FOUR HANDS
4Ø1Ø PRINT #4;CHR$(27);"!";CHR$(9);"NORTH"
4Ø2Ø PRINT #4;CHR$(27);"$";CHR$(1);CHR$(27);
   CHR$(70);
4Ø3Ø HA=1
4Ø4Ø FOR SU=Ø TO 3
4Ø5Ø PRINT #4;CHR$(9);
4Ø6Ø GOSUB 43ØØ
4Ø7Ø PRINT #4
4Ø8Ø NEXT SU
4Ø9Ø PRINT #4;CHR$(27);"!";"WEST";CHR$(9);CHR$(9);
   "EAST"
41ØØ PRINT #4;CHR$(27);"$";CHR$(1);CHR$(27);
   CHR$(7Ø);
411Ø FOR SU=Ø TO 3
412Ø HA=2
413Ø GOSUB 43ØØ
414Ø PRINT #4;CHR$(9);CHR$(9);
415Ø HA=3
416Ø GOSUB 43ØØ
417Ø PRINT #4
418Ø NEXT SU
419Ø PRINT #4;CHR$(27);"!";CHR$(9);"SOUTH"
42ØØ PRINT #4;CHR$(27);"$";CHR$(1);CHR$(27);
   CHR$(7Ø);
421Ø HA=4
422Ø FOR SU=Ø TO 3
423Ø PRINT #4;CHR$(9);
424Ø GOSUB 43ØØ
425Ø PRINT #4
426Ø NEXT SU
427Ø PRINT #4;CHR$(27);"$";CHR$(Ø);CHR$(27);CHR$(7Ø)
428Ø RETURN
429Ø REM PRINT ONE LINE
43ØØ PRINT #4;SU$(SU+1,SU+1);
431Ø FOR CA=13 TO 1 STEP -1
432Ø IF DE(SU*13+CA)=HA THEN PRINT #4;
   CA$(CA*3,CA*3+2);
```

```
4330 NEXT CA
4340 RETURN
```

Numeral program

```
10 REM PROGRAM TO DEFINE AND PRINT NUMERALS
20 OPEN #4,8,0,"P"
30 FOR N1=160 TO 200
40 PRINT #4;CHR$(27);CHR$(42);CHR$(1);
50 PRINT #4;CHR$(N1);
60 READ N2
70 PRINT #4;CHR$(N2);
80 FOR S=1 TO 11
90 READ MS
100 PRINT#4;CHR$(MS);
110 NEXT S
120 NEXT N1
130 PRINT #4;CHR$(27);CHR$(88);CHR$(1)
135 PRINT #4;CHR$(27);"1"
140 FOR I=160 TO 200 STEP 4
150 PRINT #4;CHR$(I);CHR$(I+1);CHR$(200);
160 NEXT I
165 PRINT #4
170 FOR I=162 TO 200 STEP 4
180 PRINT #4;CHR$(I);CHR$(I+1);CHR$(200);
190 NEXT I
200 PRINT #4;CHR$(27);"@"
210 CLOSE #4
220 END
340 REM ZERO
350 DATA 11,0,96,16,104,16,44,30,14,0,2,1
360 DATA 11,2,1,2,1,6,8,38,88,32,8,32
370 DATA 11,3,12,9,12,51,0,96,0,96,0,96
380 DATA 11,0,32,0,48,0,28,3,12,3,4,3
390 REM 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 REM 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 REM THREE
```

```
500 DATA 11,Ø,Ø,Ø,Ø,Ø,Ø,4,2,4,2,4
510 DATA 11,34,84,34,92,34,76,34,68,2,64,Ø
520 DATA 11,16,Ø,48,Ø,56,64,48,64,32,64,32
530 DATA 11,64,32,64,48,9,54,22,9,6,1
540 REM FOUR
550 DATA 11,Ø,Ø,Ø,Ø,Ø,Ø,64,36,88,32,16
560 DATA 11,Ø,Ø,64,32,64,56,64,6Ø,2,12,Ø
570 DATA 11,Ø,8,4,1Ø,5,1Ø,5,8,4,72,4
580 DATA 11,88,38,89,38,89,6,73,4,8,6,Ø
590 REM FIVE
600 DATA 11,Ø,Ø,Ø,Ø,64,32,84,5Ø,76,34,68
610 DATA 1Ø,34,68,34,68,34,68,2,68,2,Ø,Ø
620 DATA 1Ø,Ø,32,24,1Ø1,24,97,Ø,64,Ø,64,Ø
630 DATA 11,64,Ø,96,1,48,15,48,15,16,15,Ø
640 REM SIX
650 DATA 11,Ø,96,Ø,112,Ø,12Ø,Ø,92,Ø,1Ø2,Ø
660 DATA 11,98,Ø,98,Ø,98,Ø,7Ø,Ø,14,Ø,6
670 DATA 11,7,8,23,8,55,8,99,Ø,65,Ø,64
680 DATA 11,Ø,96,Ø,112,1,62,1,3Ø,1,14,Ø
690 REM SEVEN
700 DATA 11,Ø,16,8,6,8,6,8,6,8,6,8
710 DATA 9,7Ø,8,1Ø2,8,54,8,6,Ø,2,Ø,Ø
720 DATA 11,Ø,64,Ø,96,Ø,12Ø,Ø,124,Ø,3Ø,1
730 DATA 9,6,1,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø
740 REM EIGHT
750 DATA 11,Ø,Ø,Ø,Ø,24,36,24,1Ø2,24,1Ø2,Ø
760 DATA 11,67,Ø,67,Ø,99,28,34,28,34,28,Ø
770 DATA 11,12,18,44,19,1Ø8,19,96,1,64,Ø,64
780 DATA 11,Ø,96,1,112,15,48,15,16,14,Ø,Ø
790 REM NINE
800 DATA 11,Ø,Ø,12Ø,4,12Ø,6,12Ø,6,Ø,3,Ø
810 DATA 11,3,Ø,3,Ø,67,4,123,4,122,4,12Ø
820 DATA 11,48,Ø,56,Ø,113,Ø,99,Ø,99,Ø,99
830 DATA 11,Ø,115,Ø,57,Ø,31,Ø,15,Ø,7,Ø
840 REM SPACE
850 DATA 11,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø
```

Download utility program

```
5 DIM CS$(1),SC$(1),Z(9,13),MM(11),MM$(11),
   SS$(1),BL$(4Ø),SH$(35),PR$(35)
6 DIM PD$(35),AS$(35),AN$(1)
1Ø CS$="@":SC$="*":SS$="O"
15 BL$="  (40 characters)  "
16 AS$="ASCII (33-126)  (16Ø-254) --> "
17 SH$="IF SHIFTED ENTER 1 ELSE Ø --> "
```

```
18 PR$="NORMAL OR PROPORTIONAL --> "
19 PD$="PROPORTIONAL DATA (4-11) --> "
2Ø GRAPHICS Ø:GOSUB 66Ø
3Ø GOSUB 2ØØØ
4Ø IF KEY=47 THEN GRAPHICS Ø:END
5Ø IF KEY=1Ø THEN GOSUB 68Ø:GOTO 3Ø
6Ø IF KEY=42 THEN GOSUB 9ØØ:GOSUB 26Ø:GOTO 3Ø
7Ø GOTO 3Ø
12Ø IF Z(G,H)=Ø THEN POSITION Y+5,X+2:PRINT
   " ";:GOSUB 3ØØØ
13Ø IF Z(G,H)=1 THEN POSITION Y+5,X+2:PRINT SC$;:
   GOSUB 3ØØØ
14Ø RETURN
15Ø IF Z(G,H)=1 THEN POSITION Y+5,X+2:PRINT SS$;:
   GOSUB 3ØØØ
16Ø IF Z(G,H)=Ø THEN POSITION Y+5,X+2:PRINT CS$;:
   GOSUB 3ØØØ
17Ø RETURN
26Ø REM EDIT LEVEL
265 X=1:Y=1:G=1:H=1
267 FOR I=1 TO 11:MM(I)=Ø:NEXT I
268 FOR I=1 TO 12:FOR J=1 TO 8:Z(J,I)=Ø:NEXT J:NEXT
   I
27Ø GOSUB 2ØØØ
28Ø IF KEY=Ø THEN GOSUB 39Ø:GOTO 37Ø
29Ø IF KEY=4Ø THEN GOSUB 41Ø:GOTO 37Ø
3ØØ IF KEY=58 THEN GOSUB 43Ø:GOTO 37Ø
31Ø IF KEY=11 THEN GOSUB 45Ø:GOTO 37Ø
32Ø IF KEY=13 THEN GOSUB 47Ø:GOTO 37Ø
33Ø IF KEY=18 THEN GOSUB 49Ø:GOTO 37Ø
34Ø IF KEY=47 THEN GOSUB 5ØØ:GOTO 38Ø
37Ø GOTO 27Ø
38Ø RETURN
39Ø GOSUB 12Ø:Y=Y-3:H=H-1:IF Y<1 THEN Y=1:H=1
4ØØ GOSUB 15Ø:RETURN
41Ø GOSUB 12Ø:Y=Y+3:H=H+1:IF Y>31 THEN Y=31:H=11
42Ø GOSUB 15Ø:RETURN
43Ø GOSUB 12Ø:X=X+2:G=G+1:IF X>13 THEN X=13:G=7
44Ø GOSUB 15Ø:RETURN
45Ø GOSUB 12Ø:X=X-2:G=G-1:IF X<1 THEN X=1:G=1
46Ø GOSUB 15Ø:RETURN
47Ø IF Z(G,H-1)=1 OR Z(G,H+1)=1 THEN RETURN
48Ø Z(G,H)=1:POSITION Y+5,X+2:PRINT SS$;:GOSUB 3ØØØ:
   RETURN
```

```
49Ø Z(G,H)=Ø:POSITION Y+5,X+2:PRINT CS$;:GOSUB 3ØØØ:
   RETURN
5ØØ REM GET OUT OF EDIT MODE
51Ø IF Z(G,H)=1 THEN POSITION Y+5,X+2:PRINT SC$;:
   GOSUB 3ØØØ
52Ø IF Z(G,H)=Ø THEN POSITION Y+5,X+2:PRINT
   " ";:GOSUB 3ØØØ
525 GOSUB 4ØØØ
53Ø FOR I=1 TO 11:FOR J=1 TO 7:
   MM(I)=MM(I)+Z(J,I)*(INT (2^(J-1))+1)
535 NEXT J:NEXT I
536 GOSUB 5ØØØ
54Ø FOR I=1 TO 11 POSITION 2+I*3,18:PRINT MM(I);:
   NEXT I
55Ø GOSUB 66Ø:RETURN
66Ø FOR I=1 TO 7:FOR J=1 TO 11:Z(I,J)=Ø:NEXT J:NEXT
   I
67Ø POSITION 1,23:PRINT
   "E)DIT P)RINTER Q)UIT              ";
675 RETURN
68Ø REM PRINT MODE
69Ø GOSUB 7ØØØ:PRINT PR$;:INPUT AN$
7ØØ IF AN$="N" THEN PR=Ø:GOTO 75Ø
71Ø IF AN$="P" THEN 73Ø
72Ø GOTO 69Ø
73Ø GOSUB 7ØØØ:PRINT PD$;:INPUT PR
74Ø IF PR<4 OR PR>11 THEN 73Ø
75Ø GOSUB 7ØØØ:PRINT SH$;:INPUT SH
76Ø IF SH<Ø OR SH>1 THEN GOTO 75Ø
77Ø GOSUB 7ØØØ:PRINT AS$;:INPUT AS
78Ø IF (AS<33 OR AS>126) AND (AS<16Ø OR AS>254) THEN
   77Ø
79Ø GOSUB 7ØØØ
8ØØ IF SH=1 THEN SH=16
81Ø N1=AS:N2=PR+SH
82Ø FOR I=1 TO 11:MM$(LEN(MM$)+1)=CHR$(MM(I)):NEXT I
83Ø OPEN #4,8,Ø,"P"
835 PRINT #4,CHR$(27);"*";CHR$(1);CHR$(N1);CHR$(N2);
   MM$
84Ø IF AN$="N" THEN PRINT #4;CHR$(27);"$";CHR$(1):
   GOTO 86Ø
85Ø PRINT #4;CHR$(27);"X";CHR$(1)
86Ø FOR I=1 TO 2Ø:PRINT #4;CHR$(N1);" ";:NEXT I:
   PRINT #4
```

```
870 PRINT #4;CHR$(14);:FOR I=1 TO 10:PRINT #4;
   CHR$(N1);" ";:NEXT I:PRINT #4
88Ø PRINT #4;CHR$(15);:FOR I=1 TO 20:PRINT #4;
   CHR$(N1);" ";:NEXT I:PRINT #4
89Ø IF AN$="N" THEN PRINT #4;CHR$(27);"$";CHR$(Ø):
   GOTO 895
892 PRINT #4;CHR$(27);"X";CHR$(Ø)
895 PRINT #4;CHR$(27);"@":CLOSE #4:GOSUB 66Ø:MM$="":
   RETURN
9ØØ GRAPHICS Ø
9Ø4 PRINT
9Ø5 PRINT "  M1 M2 M3 M4 M5 M6 M7 M8 M9 M1Ø M11"
91Ø FOR I=Ø TO 7:PRINT "  ";:FOR J=1 TO 11
915 PRINT "!--";:NEXT J:PRINT "!":IF I<7 THEN PRINT
   INT(2^I)+1:NEXT I
92Ø PRINT :PRINT :PRINT
93Ø PRINT "R)IGHT L)EFT D)OWN U)P "
94Ø PRINT "I)NSERT   C)LEAR   Q)UIT"
95Ø POSITION 6,3:PRINT CS$;
955 GOSUB 3ØØØ
96Ø RETURN
2ØØØ REM SINGLE CHAR INPUT
2Ø1Ø KEY=PEEK(764):IF KEY=255 THEN 2Ø1Ø
2Ø2Ø POKE 764,255
2Ø3Ø RETURN
3ØØØ POSITION 35,21
3Ø1Ø PRINT " ";
3Ø2Ø RETURN
4ØØØ POSITION 15,18
4Ø1Ø PRINT "PLEASE STAND BY";
4Ø2Ø RETURN
5ØØØ POSITION 1,18
5Ø1Ø FOR I=1 TO 5:PRINT BL$;:NEXT I
5Ø2Ø RETURN
7ØØØ POSITION 1,2Ø
7Ø1Ø PRINT BL$;
7Ø2Ø POSITION 1,2Ø
7Ø3Ø RETURN
```

Delta Plot program

```
2 GRAPHICS Ø
3 PRINT " ":PRINT " ":PRINT " "
4 PRINT " THIS PROGRAM TAKES ABOUT"
5 PRINT " 1 MINUTE  TO RUN, PLEASE"
6 PRINT " TURN ON YOUR PRINTER AND "
```

```
7 PRINT " STAND BY ............."
8 PRINT " ":PRINT " ":PRINT " "
1Ø FOR I=1536 TO 1553
2Ø READ O
3Ø POKE I,O
4Ø NEXT I
5Ø DATA 1Ø4,1Ø4,133,2Ø5,1Ø4,133,2Ø4
6Ø DATA 1Ø4,5,2Ø5,133,213,1Ø4,5,2Ø4
7Ø DATA 133,212,96
1ØØ REM MICRO-PLOT
11Ø DIM M(76,14),MASK(6)
12Ø DIM A$(1ØØ)
13Ø DIM B$(1Ø)
132 FOR I=Ø TO 14
133 FOR J=1 TO 76
134 M(J,I)=Ø
135 NEXT J
136 NEXT I
1ØØØ REM SET PROGRAM CONSTANT
1Ø1Ø MASK(1)=128:MASK(4)=16
1Ø2Ø MASK(2)=64:MASK(5)=8
1Ø3Ø MASK(3)=32:MASK(6)=4
1Ø4Ø LX=2Ø:LY=2Ø
1Ø5Ø XFAC=72/LX:YFAC=87/LY
2ØØØ REM PLOT CURVE
2Ø1Ø LET RAD=9
2Ø2Ø X1=19:Y1=1Ø
2Ø3Ø FOR ANG=Ø TO 36Ø STEP 1Ø
2Ø4Ø R1=ANG*6.28/36Ø
2Ø5Ø X2=RAD*COS(R1)+1Ø
2Ø55 Y2=RAD*SIN(R1)+1Ø
2Ø6Ø GOSUB 4ØØØ
2Ø7Ø NEXT ANG
3ØØØ REM SEND BIT IMAGE TO PRINTER
3ØØ5 OPEN #4,8,Ø,"P"
3Ø1Ø PRINT #4;CHR$(27);"A";CHR$(6)
3Ø2Ø FOR ROW=Ø TO 14
3Ø3Ø A$=""
3Ø4Ø PRINT #4;CHR$(27);"K";CHR$(75);CHR$(Ø);
3Ø5Ø FOR COL=1 TO 75
3Ø54 RE=INT(M(COL,ROW))
3Ø55 B$=CHR$(RE)
3Ø6Ø A$(LEN(A$)+1)=B$
3Ø7Ø NEXT COL
3Ø8Ø PRINT #4;A$;" "
```

```
3090 NEXT ROW
3100 PRINT #4;CHR$(27);"A";CHR$(12)
3110 CLOSE #4
3150 END
4000 REM DRAW A LINE FROM X1,Y1 TO X2,Y2
4010 XL=X2-X1:YL=Y2-Y1
4020 NX=ABS(XL*XFAC):NY=ABS(YL*YFAC)
4030 IF NX<NY THEN NX=NY
4040 NS=INT(NX+1)
4050 DX=XL/NS
4055 DY=YL/NS
4060 FOR I=1 TO NS
4070 X1=X1+DX:Y1=Y1+DY
4080 GOSUB 5000
4090 NEXT I
4100 RETURN
5000 REM PLOT A POINT AT X1,Y1
5010 XX=X1*XFAC:YY=Y1*YFAC
5020 COL=INT(XX)+1
5030 ROW=INT(YY/6)
5040 XIT=INT(YY-ROW*6)+1
5050 A1=M(COL,ROW)
5060 A2=MASK(XIT)
5070 M(COL,ROW)=USR(1536,A1,A2)
5080 RETURN
```

Pie chart program

```
2 GRAPHICS 0
3 PRINT "PLEASE STAND BY"
10 FOR I=1536 TO 1553
20 READ O
30 POKE I,O
40 NEXT I
50 DATA 104,104,133,205,104,133,204
60 DATA 104,5,205,133,213,104,5,204
70 DATA 133,212,96
100 REM MICRO-PLOT
110 DIM M(76,11),MASK(6),BL$(80),N$(99)
120 DIM A$(100),T$(99),TT(20),RR(20),TI$(99)
130 DIM B$(10),PCT(10),T1$(99),T2$(99)
132 FOR I=0 TO 11
133 FOR J=1 TO 76
134 M(J,I)=0
135 NEXT J
136 NEXT I
```

```
140 GOSUB 7000
141 GRAPHICS 0
142 PRINT " ":PRINT " ":PRINT " "
143 PRINT " THIS PROGRAM TAKES ABOUT"
144 PRINT " 2 MINUTES TO RUN, PLEASE"
145 PRINT " TURN ON YOUR PRINTER AND "
146 " STAND BY ............."
147 PRINT " ":PRINT " ":PRINT " "
150 T1$="  (40 characters)  "
160 T1$(LEN(T1$)+1)=T1$
170 T2$=T1$
180 BL$="  (40 characters)  "
190 BL$(LEN(BL$)+1)=BL$
195 FOR I=1 TO 20:TT(I)=1:NEXT I
198 FOR I=1 TO 20:RR(I)=0:NEXT I
1000 REM SET PROGRAM CONSTANT
1010 MASK(1)=128:MASK (4)=16
1020 MASK(2)=64:MASK(5)=8
1030 MASK(3)=32:MASK(6)=4
1040 LX=20:LY=20
1050 XFAC=72/LX:YFAC=75/LY
2000 REM PLOT CURVE
2010 LET RAD=9
2020 X1=19:Y1=10
2030 FOR ANG=0 TO 360 STEP 5
2040 R1=ANG*6.28/360
2050 X2=RAD*COS(R1)+10
2055 Y2=RAD*SIN(R1)+10
2060 GOSUB 4000
2070 NEXT ANG
2080 FOR PI=1 TO NP
2090 X1=10:Y1=10
2100 TP=TP+PCT(PI)
2110 ANG=360*TP*0.01
2120 R1=ANG*6.28/360
2130 X2=RAD*COS(R1)+10:Y2=RAD*SIN(R1)+10
2140 GOSUB 4000
2160 GOSUB 6000
2170 NEXT PI
2180 IF LEN(T1$)<99 THEN T1$(LEN(T1$)+1)=" ":GOTO
   2180
2190 IF LEN(T2$)<99 THEN T2$(LEN(T2$)+1)=" ":GOTO
   2190
3000 REM SEND BIT IMAGE TO PRINTER
3005 OPEN #4,8,0,"P"
```

```
3Ø1Ø PRINT #4;CHR$(27);"A";CHR$(3)
3Ø12 WW=LEN(TI$)
3Ø13 VV=INT((80-WW)/2)
3Ø14 PRINT #4;BL$(1,VV);TI$:FOR I=1 TO 25:PRINT #4:
   NEXT I
3Ø15 PRINT #4;BL$(1,TT(1));T1$(1,9);CHR$(1Ø)
3Ø16 PRINT #4;BL$(1,TT(2));T1$(1Ø,19);CHR$(1Ø)
3Ø17 PRINT #4;BL$(1,TT(3));T1$(2Ø,29);CHR$(1Ø)
3Ø2Ø FOR ROW=Ø TO 11
3Ø3Ø A$=""
3Ø35 PRINT #4;BL$(1,35);
3Ø4Ø PRINT #4;CHR$(27);"K";CHR$(75);CHR$(Ø);
3Ø5Ø FOR COL=1 TO 75
3Ø54 RE=INT(M(COL,ROW))
3Ø55 B$=CHR$(RE)
3Ø6Ø A$(LEN(A$)+1)=B$
3Ø7Ø NEXT COL
3Ø8Ø PRINT #4;A$
3Ø85 IF ROW>5 THEN GOSUB 8ØØØ:GOTO 3Ø9Ø
3Ø86 HH=(ROW+3)*1Ø
3Ø87 PRINT #4;BL$(1,TT(ROW+4));T1$(HH,HH+9)
3Ø9Ø NEXT ROW
3Ø95 PRINT #4;BL$(1,TT(16));T2$(6Ø,69);CHR$(1Ø);
3Ø96 PRINT #4;BL$(1,TT(17));T2$(7Ø,79);CHR$(1Ø);
3Ø97 PRINT #4;BL$(1,TT(18));T2$(8Ø,89)
31ØØ PRINT #4;CHR$(27);"A";CHR$(12)
311Ø CLOSE #4
315Ø END
4ØØØ REM DRAW A LINE FROM X1,Y1 TO Y2,Y2
4Ø1Ø XL=X2-X1:YL=Y2-Y1
4Ø2Ø NX=ABS(XL*XFAC):NY=ABS(YL*YFAC)
4Ø3Ø IF NX<NY THEN NX=NY
4Ø4Ø NS=INT(NX+1)
4Ø5Ø DX=XL/NS
4Ø55 DY=YL/NS
4Ø6Ø FOR I=1 TO NS
4Ø7Ø X1=X1+DX:Y1=Y1+DY
4Ø8Ø GOSUB 5ØØØ
4Ø9Ø NEXT I
41ØØ RETURN
5ØØØ REM PLOT A POINT AT X1,Y1
5Ø1Ø XX=X1*XFAC:YY=Y1*YFAC
5Ø2Ø COL=INT(XX)+1
5Ø3Ø ROW=INT(YY/6)
5Ø4Ø XIT=INT(YY-ROW*6)+1
```

```
5Ø5Ø A1=M(COL,ROW)
5Ø6Ø A2=MASK(XIT)
5Ø7Ø M(COL,ROW)=USR(1536,A1,A2)
5Ø8Ø RETURN
6ØØØ REM
6Ø1Ø MA=(ANG+PA)/2
6Ø2Ø R1=MA*6.28/36Ø
6Ø3Ø X3=INT(8*SIN(R1))
6Ø35 Y3=INT(1Ø*COS(R1))
6Ø4Ø X4=1Ø+X3:Y4=4Ø+Y3
6Ø41 GOSUB 9ØØØ:RR(PI)=X4
6Ø45 IF MA>27Ø OR MA<9Ø THEN TT(X4)=Y4:GOTO 6Ø5Ø
6Ø47 TT(X4)=Y4-1Ø
6Ø5Ø IF X4>9 THEN GOSUB 65ØØ:GOTO 6Ø6Ø
6Ø52 DD=(X4-1)*1Ø+1
6Ø54 DF=(PI-1)*1Ø+1
6Ø56 T1$(DD,DD+9)=T$(DF,DF+9)
6Ø6Ø PA=ANG
6Ø7Ø RETURN
65ØØ X4=X4-9
65Ø2 DD=(X4-1)*1Ø+1
65Ø4 DF=(PI-1)*1Ø+1
65Ø6 T2$(DD,DD+9)=T$(DF,DF+9)
65Ø8 RETURN
7ØØØ GRAPHICS Ø
7ØØ1 PRINT "TITLE CAN BE UP TO 8Ø CHARACTERS LONG"
7ØØ2 PRINT "ENTER TITLE  ";:INPUT TI$
7ØØ4 IF LEN(TI$)>80 THEN TI$=TI$(1,8Ø)
7ØØ5 AS=Ø:AL=1ØØ:FL=9:OO=1
7Ø1Ø GRAPHICS Ø
7Ø2Ø PRINT "YOU CAN HAVE UP TO 9 FIELDS AND EACH
   FIELD CAN BE UP TO NINE CHARACTERS LONG"
7Ø25 IF LEN(T$)<99 THEN T$(LEN(T$)+1)=" ":GOTO 7Ø25
7Ø3Ø PRINT "AMOUNT SO FAR ";AS
7Ø4Ø PRINT "AMOUNT LEFT   ";AL
7Ø5Ø PRINT "FIELD LEFT    ";FL
7Ø6Ø PRINT :PRINT
7Ø7Ø PRINT "FIELD SIZE %  ";:INPUT FS
7Ø8Ø IF FS>AL OR FS=Ø THEN FS=AL
7Ø9Ø AL=AL-FS:AS=AS+FS
71ØØ PRINT "ENTER FIELD NAME ";:INPUT N$
711Ø IF LEN(N$)>9 THEN N$=N$(1,9)
712Ø IF LEN(N$)<9 THEN N$(LEN(N$)+1)=" ":GOTO 712Ø
713Ø PCT(OO)=FS
714Ø TR=(OO-1)*1Ø+1
```

```
715Ø T$(TR,TR+9)=N$
716Ø OO=OO+1
717Ø IF OO>9 THEN PCT(OO-1)=PCT(OO-1)+AL:GOTO 72ØØ
718Ø if AL=Ø THEN 72ØØ
7185 FL=FL-1
719Ø GOTO 7Ø1Ø
72ØØ NP=OO-1
721Ø GRAPHICS Ø
722Ø RETURN
8ØØØ HH=(ROW-6)*1Ø+1
8Ø1Ø PRINT #4;BL$(1,TT(ROW+4));T2$(HH,HH+9)
8Ø2Ø RETURN
9ØØØ FOR I=1 TO PI
9Ø1Ø IF RR(I)=X4 THEN YY=1
9Ø2Ø NEXT I
9Ø25 IF YY=Ø THEN 9Ø8Ø
9Ø3Ø IF YY=1 THEN X4=X4-1
9Ø4Ø IF X4<1 THEN X4=X4+2
9Ø5Ø YY=Ø:GOTO 9ØØØ
9Ø8Ø RETURN
```

Appendix G

Commodore VIC-20 and C-64 Computers

The best way to connect Delta to your Commodore computer is with a **Universal/Commodore Parallel Interface** by Star Micronics. Or you can use many of the other available parallel interface adapters for the Commodore computers.

Setting the switches

We recommend that you set the DIP switches in Delta as shown below when connecting it to a Commodore computer.

Recommended DIP Switch Settings for Commodore VIC-20 and C-64

Switch	Setting	Function
1-1	ON	11 inch page size
1-2	ON	Normal print density
1-3	ON	10 CPI pitch
1-4	ON	Normal characters
1-5	ON	1/6 inch line feed
1-6	ON	U.S.A. Character set
1-7	ON	
1-8	ON	
2-1	ON	Paper-out detector active
2-2	OFF	Parallel interface
2-3	OFF	8-bit interface
2-4	ON	Auto line feed

Using Commodore BASIC

Commodore computers can use the full capabilities of Delta. Commodore BASIC does, however, have a few differences from other BASICs.

Commodore BASIC has no LPRINT statement. You must open the printer as a file and then direct your print statements to that file, like this:

```
    10 OPEN 4,4
    20 PRINT#4, "TESTING"
```

When the program is done printing, you should clear the buffer and close the file like this:

```
    90 PRINT#4 : CLOSE 4
```

Listing programs

To list a program on the Commodore computers you must open the printer as a file and redirect screen output to the printer before issuing the LIST command. The correct sequence looks like this:

```
    OPEN 4,4
    CMD 4
    LIST
```

When you are done listing your program you must close the printer channel to stop sending output to the printer. To do this, type:

```
    PRINT#4 : CLOSE 4
```

Program listings

The following programs have been converted to run on Commodore computers.

Chart program

```
    10 REM  COMMODORE 64 <> DELTA 10
    100 OPEN4,4:CMD4
    110 GOSUB 1000
    120 GOSUB 2000
    130 PRINT "*REGULAR*"
    140 GOSUB 3000
```

```
15Ø PRINT "*DOUBLE STRIKE*"
16Ø PRINT DS$;
17Ø GOSUB 3ØØØ
18Ø PRINT "*EMPHASIZED*"
19Ø EM=TRUE
2ØØ GOSUB 3ØØØ
21Ø PRINT "*DOUBLE STRIKE & EMPHASIZED*"
22Ø PRINT DS$;EM$;
23Ø GOSUB 3ØØØ
24Ø PRINT#4:CLOSE4
25Ø END
1ØØØ REM
1Ø1Ø IT$=CHR$(27)+CHR$(52)
1Ø2Ø RO$=CHR$(27)+CHR$(53)
1Ø3Ø EN$=CHR$(27)+CHR$(87)+CHR$(1)
1Ø4Ø NW$=CHR$(27)+CHR$(87)+CHR$(Ø)
1Ø5Ø PI$=CHR$(27)+CHR$(66)+CHR$(1)
1Ø6Ø EL$=CHR$(27)+CHR$(66)+CHR$(2)
1Ø7Ø CO$=CHR$(27)+CHR$(66)+CHR$(3)
1Ø8Ø EM$=CHR$(27)+CHR$(69)
1Ø9Ø NE$=CHR$(27)+CHR$(7Ø)
11ØØ DS$=CHR$(27)+CHR$(71)
111Ø ND$=CHR$(27)+CHR$(72)
112Ø UN$=CHR$(27)+CHR$(45)+CHR$(1)
113Ø NU$=CHR$(27)+CHR$(45)+CHR$(Ø)
114Ø SP$=CHR$(27)+CHR$(83)+CHR$(Ø)
115Ø SB$=CHR$(27)+CHR$(83)+CHR$(1)
116Ø NS$=CHR$(27)+CHR$(84)
117Ø RA$=NE$+NU$+ND$+RO$+PI$+NW$
118Ø TRUE=1:FALSE=Ø
119ØRETURN
2ØØØ PRINT RA$
2Ø1Ø PRINT EN$;"   NORMAL      ENLARGED   "
2Ø2Ø PRINT RA$;UN$;
2Ø3Ø PRINT CO$;"CONDENSED ";
2Ø4Ø PRINT EL$;"  ELITE   ";
2Ø5Ø PRINT PI$;"  PICA    ";
2Ø6Ø PRINT CO$;"CONDENSED ";
2Ø7Ø PRINT EL$;"  ELITE   ";
2Ø8Ø PRINT PI$;"  PICA    ";RA$
2Ø9Ø RETURN
3ØØØ IT=FALSE:PRINT RO$;
3Ø1Ø UN=FALSE:PRINT NU$;
3Ø2Ø EN=FALSE:PRINT NW$;
3Ø3Ø PI=FALSE:
```

```
3Ø4Ø PRINT CO$;:GOSUB 313Ø
3Ø5Ø PRINT EL$;:GOSUB 313Ø
3Ø6Ø PRINT PI$;:PI=TRUE:GOSUB 313Ø
3Ø7Ø IF EN=TRUE THEN PRINT:GOTO 3Ø9Ø
3Ø8Ø PRINT EN$;:EN=TRUE:GOTO 3Ø3Ø
3Ø9Ø IF UN=TRUE THEN PRINT:GOTO 311Ø
31ØØ PRINT UN$;:UN=TRUE:GOTO 3Ø2Ø
311Ø IF IT=TRUE THEN PRINT RA$:RETURN
312Ø PRINT IT$;:IT=TRUE:GOTO 3Ø1Ø
313Ø BL$="      ":FD$="...."
314Ø IF EM=FALSE THEN PRINT "AB"+CHR$(99)+CHR$(1ØØ);
   :GOTO 319Ø
315Ø IF PI=FALSE THEN PRINT FD$;:GOTO 317Ø
316Ø PRINT "AB";CHR$(99);CHR$(1ØØ);
317Ø IF EN=TRUE THEN PRINT " ";:RETURN
318Ø IF EN=FALSE THEN PRINT BL$;:RETURN
319Ø IF EN=TRUE THEN PRINT " ";:RETURN
32ØØ PRINT SP$;"X";CHR$(12Ø);
321Ø PRINT SB$;"Y";CHR$(121);"  ";
322Ø PRINT NS$;
323Ø RETURN
```

Special character chart program

```
5 REM COMMODORE 64 <> DELTASET
1Ø OPEN4,4:CMD4
2Ø FOR J=16Ø TO 255 STEP 8
3Ø FOR I=J TO J+7
4Ø PRINT I;" =";CHR$(I);CHR$(9);
5Ø NEXT I:PRINT:NEXT J
6Ø PRINT#4:CLOSE4
7Ø END
```

Macro program

```
1Ø REM COMMODORE 64 <> DELTAMACRO
2Ø OPEN4,4:CMD4
3Ø PRINT CHR$(27);"+";
4Ø PRINT CHR$(18);
5Ø PRINT CHR$(27);"W"CHR$(Ø);
6Ø PRINT CHR$(27);"F";
7Ø PRINT CHR$(27);"H";
8Ø PRINT CHR$(27);"-";CHR$(Ø);
9Ø PRINT CHR$(27);"T";
95 PRINT CHR$(27);"5";
```

```
98 PRINT CHR$(30)
99 PRINT#4:CLOSE4:END
```

Bridge hand program

```
10 REM COMMODORE 64 <> DELTA BRIDGE
15 OPEN4,4:CMD4
20 GOSUB 1000
30 GOSUB 2000
40 GOSUB 3000
50 GOSUB 4000
60 PRINT#4:CLOSE4
70 END
1000 REM  INITIALIZE VARIABLES
1010 DIM HA(4),DE(52),CA$(13),SU$(3)
1020 CA$(1)=" 2":CA$(2)=" 3":CA$(3)=" 4"
1030 CA$(4)=" 5":CA$(5)=" 6":CA$(6)=" 7"
1040 CA$(7)=" 8":CA$(8)=" 9":CA$(9)=" 10"
1050 CA$(10)=" J":CA$(11)=" Q":CA$(12)=" K":
   CA$(13)=" A"
1060 SU$(0)="S":SU$(1)="H":SU$(2)="D":SU$(3)="C"
1070 RETURN
2000 REM INITIALIZE PRINTER
2010 PRINT CHR$(27);CHR$(68);CHR$(20);CHR$(40);
   CHR$(0)
2020 PRINT CHR$(27)CHR$(43);CHR$(27)CHR$(36)CHR$(0);
   CHR$(27)CHR$(69)CHR$(30)
2030 PRINT CHR$(27);CHR$(42);CHR$(0)
2040 FOR I=1 TO 4
2050 PRINT CHR$(27);CHR$(42);CHR$(1);
2060 FOR J=1 TO 13
2070 READ X:PRINT CHR$(X);
2080 NEXT J
2090 NEXT I
2100 PRINT
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 REM DEAL CARD
3010 FOR CA=1 TO 52
3020 X=INT(RND(1)*4+1)
3030 IF HA(X)=13 THEN 3020
3035 HA(X)=HA(X)+1
3040 DE(CA)=X
```

```
3Ø5Ø NEXT CA
3Ø6Ø RETURN
4ØØØ REM PRINT FOUR HANDS
4Ø1Ø PRINT CHR$(27);"!";CHR$(9);"NORTH"
4Ø2Ø PRINT CHR$(27);"$";CHR$(1);CHR$(27);CHR$(7Ø);
4Ø3Ø HA=1
4Ø4Ø FOR SU=Ø TO 3
4Ø5Ø PRINT CHR$(9);
4Ø6Ø GOSUB 43ØØ
4Ø7Ø PRINT
4Ø8Ø NEXT SU
4Ø9Ø PRINT CHR$(27);"!";"WEST";CHR$(9);CHR$(9);
   "EAST"
41ØØ PRINT CHR$(27);"$";CHR$(1);CHR$(27);CHR$(7Ø);
411Ø FOR SU=Ø TO 3
412Ø HA=2
413Ø GOSUB 43ØØ
414Ø PRINT CHR$(9)CHR$(9);
415Ø HA=3
416Ø GOSUB 43ØØ
417Ø PRINT
418Ø NEXT SU
419Ø PRINT CHR$(27);"!";CHR$(9);"SOUTH"
42ØØ PRINT CHR$(27);"$";CHR$(1);CHR$(27);CHR$(7Ø);
421Ø HA=4
422Ø FOR SU=Ø TO 3
423Ø PRINT CHR$(9);
424Ø GOSUB 43ØØ
425Ø PRINT
426Ø NEXT SU
427Ø PRINT CHR$(27);"$"CHR$(Ø);CHR$(27);CHR$(7Ø)
428Ø RETURN
429Ø REM PRINT ONE LINE
43ØØ PRINT SU$(SU);
431Ø FOR CA=13 TO 1 STEP -1
432Ø IF DE(SU*13+CA)=HA THEN PRINT CA$(CA);
433Ø NEXT CA
434Ø RETURN
```

Numeral program

```
5 REM COMMODORE 64 <> DELTANUMERAL
1Ø REM PROGRAM TO DEFINE AND PRINT NUMERALS
2Ø REM EACH NUMERAL IS MADE UP OF 4 CHARACTERS (2
   WIDE * 2 HIGH )
25 OPEN4,4:CMD4
```

```
3Ø DD$=CHR$(27) + CHR$(42) + CHR$(1)
4Ø DP$ = CHR$(27) + CHR$(88) + CHR$(1)
5Ø NDP$ = CHR$(27) + CHR$(88) + CHR$(Ø)
6Ø L7$ = CHR$(27) + CHR$(49) : L12$= CHR$(27) +
   CHR$(5Ø)
7Ø FOR N1= 16Ø TO 2ØØ
8Ø PRINT DD$;
9Ø PRINT CHR$(N1);
1ØØ READ N2
11Ø PRINT CHR$(N2);
12Ø FOR S = 1 TO 11
13Ø READ MS
14Ø PRINT CHR$(MS);
15Ø NEXT S
16Ø NEXT N1
17Ø REM
18Ø AS = 16Ø
19Ø FOR NUM = Ø TO 9
2ØØ NT$(NUM)=CHR$(AS + Ø) + CHR$(AS + 1)
21Ø NB$(NUM)=CHR$(AS + 2) + CHR$(AS + 3)
22Ø AS = AS + 4
23Ø NEXT NUM
24Ø BK$= CHR$(2ØØ)
25Ø PRINT DP$;L7$
26Ø FOR NUM = Ø TO 9
27Ø PRINT NT$(NUM);BK$;
28Ø NEXT NUM
29Ø PRINT
3ØØ FOR NUM = Ø TO 9
31Ø PRINT NB$(NUM); BK$;
32Ø NEXT NUM
33Ø PRINT NP$;L12$
335 PRINT#4:CLOSE4
34Ø REM  ZERO
35Ø DATA 11,Ø,96,16,1Ø4,16,44,3Ø,14,Ø,2,1
36Ø DATA 11,2,1,2,1,6,8,38,88,32,88,32
37Ø DATA 11,3,12,19,12,51,Ø,96,Ø,96,Ø,96
38Ø DATA 11,Ø,32,Ø,48,Ø,28,3,12,3,4,3
39Ø REM  ONE
4ØØ DATA 11,Ø,Ø,Ø,Ø,Ø,4,Ø,4,Ø,4,126
41Ø DATA 9,12,114,12,114,12,2,Ø,Ø,Ø,Ø,Ø
42Ø DATA 11,64,Ø,64,Ø,64,Ø,64,32,8Ø,47,8Ø
43Ø DATA 9,47,8Ø,47,64,Ø,64,Ø,64,Ø,Ø,Ø
44Ø REM TWO
45Ø DATA 11,Ø,Ø,Ø,Ø,Ø,12,16,14,Ø,6,Ø
```

```
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 REM 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 REM  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 REM  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
630 DATA 11,64,0,96,1,48,15,48,15,16,15,0
640 REM SIX
650 DATA 11,0,96,0,112,0,120,0,92,0,102,0
660 DATA 11,98,0,98,0,98,0,70,0,14,0,6
670 DATA 11,7,8,23,8,55,8,99,0,65,0,64
680 DATA 11,0,96,0,112,1,62,1,30,1,14,0
690 REM  SEVEN
700 DATA 11,0,16,8,6,8,6,8,6,8,6,8
710 DATA 9,70,8,102,8,54,8,6,0,2,0,0
720 DATA 11,0,64,0,96,0,120,0,124,0,30,1
730 DATA 9,6,1,0,0,0,0,0,0,0,0,0
740 REM EIGHT
750 DATA 11,0,0,0,0,24,36,24,102,24,102,0
760 DATA 11,67,0,67,0,99,28,34,28,34,28,0
770 DATA 11,12,18,44,19,108,19,96,1,64,0,64
780 DATA 11,0,96,1,112,15,48,15,16,14,0,0
790 REM NINE
800 DATA 11,0,0,120,4,120,6,120,6,0,3,0
810 DATA 11,3,0,3,0,67,4,123,4,122,4,120
820 DATA 11,48,0,56,0,113,0,99,0,99,0,99
830 DATA 11,0,115,0,57,0,31,0,15,0,7,0
840 REM SPACE
850 DATA 11,0,0,0,0,0,0,0,0,0,0,0
```

Download utility program

```
4 ED$=" E)DIT P)RINTER Q)UIT                    "
5 POKE 53281,0:POKE 53280,0
6 PRINT CHR$(5)
```

```
7 DD=115Ø
8 Y=Ø:X=Ø
1Ø DIM Z(8,12),MM(11),ML$(11),KK$(11,5)
15 AD=1984
2Ø PRINT CHR$(147):GOSUB 66Ø
3Ø CS$=CHR$(Ø):SC$=CHR$(42):SS$=CHR$(15)
4Ø GET A$:IF A$=""THEN 4Ø
5Ø IF A$="Q" THEN PRINT CHR$(147):END
6Ø IF A$="P" THEN GOSUB 68Ø:GOTO 4Ø
7Ø IF A$="E" THEN GOSUB 9ØØ:GOSUB 26Ø:GOTO 4Ø
8Ø GOTO 4Ø
12Ø IF Z(G,H)=Ø THEN C$=" ":GOSUB 11ØØØ
13Ø IF Z(G,H)=1 THEN C$=SC$:GOSUB 11ØØØ
14Ø RETURN
15Ø IF Z(G,H)=1 THEN C$=SS$:GOSUB 11ØØØ
16Ø IF Z(G,H)=Ø THEN C$=CS$:GOSUB 11ØØØ
17Ø RETURN
26Ø REM  EDIT LEVEL
265 H=1:G=1:X=Ø:Y=Ø
266 FOR I=1 TO 11:ML$(I)="":MM(I)=Ø:   NEXT I
267 FOR I=1 TO 11:FOR J=1 TO 5:KK$(I,J)="":NEXT J:
   NEXT I
27Ø GET A$:IF A$=""THEN 27Ø
3ØØ IF A$="L"THEN GOSUB 39Ø:GOTO 37Ø
31Ø IF A$="R" THEN GOSUB 41Ø:GOTO 37Ø
32Ø IF A$="D" THEN GOSUB 43Ø:GOTO 37Ø
33Ø IF A$="U" THEN GOSUB 45Ø:GOTO 37Ø
34Ø IF A$="I" THEN GOSUB 47Ø:GOTO 37Ø
35Ø IF A$="C" THEN GOSUB 49Ø:GOTO 37Ø
36Ø IF A$="Q" THEN GOSUB 5ØØ:GOTO 38Ø
37Ø GOTO 27Ø
38Ø RETURN
39Ø GOSUB 12Ø:Y=Y-3:H=H-1:IF Y<Ø then Y=Ø:H=1
4ØØ GOSUB 15Ø:RETURN
41Ø GOSUB 12Ø:Y=Y+3:H=H+1:IF Y>3Ø THEN Y=3Ø:H=11
42Ø GOSUB 15Ø:RETURN
43Ø GOSUB 12Ø:X=X+8Ø:G=G+1:IF X>48Ø  THEN X=48Ø :G=7
44Ø GOSUB 15Ø:RETURN
45Ø GOSUB 12Ø:x=x-8Ø:G=G-1:IF X<Ø THEN X=Ø:G=1
46Ø GOSUB 15Ø:RETURN
47Ø IF Z(G,H-1)=1 OR Z(G,H+1)=1 THEN RETURN
48Ø Z(G,H)=1:C$=SS$:GOSUB 11ØØØ:RETURN
49Ø Z(G,H)=Ø:C$=CS$:GOSUB 11ØØØ:RETURN
5ØØ REM  GET OF EDIT MODE
51Ø C$=ED:GOSUB 1ØØØ
```

```
52Ø IF Z(G,H)=1 THEN C$=SC$:GOSUB 11ØØØ:GOTO 54Ø
53Ø IF Z(G,H)=Ø THEN C$=" ":GOSUB 11ØØØ
54Ø REM PRINT COLUMN VALUES
55Ø FOR I=1 TO 11:FOR J=1 TO 7
56Ø MM(I)=MM(I)+Z(J,I)*2^(J-1):NEXT J:NEXT I
57Ø J=Ø:PRINT CHR$(19):FOR I=1 TO 8 :PRINT CHR$(17):
   NEXT I
574 PRINT "    ";
575 FOR I=1 TO 11:ML$(I)=STR$(MM(I)):NEXT I
58Ø FOR I=1 TO 11:FOR J=1 TO LEN(ML$(I)):
   KK$(I,J)=MID$(ML$(I),J,1):NEXT J
585 NEXT I
59Ø FOR I=1 TO 11:D1=17Ø7:FOR J=1 TO LEN(ML$(I))
592 POKE D1+I*3, ASC(KK$(I,J)):D1=D1+4Ø:NEXT J
594 NEXT I
6ØØ GOSUB 66Ø:RETURN
66Ø B$=ED$:GOSUB 1ØØØ
67Ø FOR I=1 TO 7:FOR J=1 TO 11:Z(I,J)=Ø:NEXT J:NEXT
   I:RETURN
68Ø REM PRINT MODE
69Ø PRINT CHR$(19):FOR I=1 TO 22:PRINT CHR$(17);:
   NEXT I
695 INPUT "NORMAL OR PROPORTIONAL (N/P) ";AN$
7ØØ IF AN$="N" THEN PR=Ø:GOTO 75Ø
71Ø IF AN$="P" THEN GOTO 73Ø
72Ø PRINT CHR$(145);:GOTO 695
73Ø GOSUB 12ØØØ:PRINT CHR$(145);
732 INPUT "PROPORTIONAL DATA (4-11) ";PR
74Ø IF PR<4 OR PR>11 THEN 73Ø
75Ø GOSUB 12ØØØ:PRINT CHR$(145);
755 INPUT "SHIFTED ENTER 1 ELSE ENTER Ø  ";SH
76Ø IF SH<Ø OR SH>1 THEN 75Ø
77Ø GOSUB 12ØØØ:PRINT CHR$(145);
775 INPUT "ASCII CODE  (33-126)  (16Ø-254) ";AS
777 IF (AS<33 OR  AS>126) AND (AS<16Ø OR AS>254)
   THEN 77Ø
779 GOSUB 12ØØØ
78Ø IF SH=1 THEN SH=16
79Ø FOR I=1 TO 11:MM$=MM$+CHR$(MM(I)):NEXT I
8ØØ N1=AS:N2=PR+SH
81Ø OPEN4,4:CMD4
82Ø PRINT CHR$(27);"*";CHR$(1);CHR$(N1);CHR$(N2);MM$
83Ø IF AN$="N" THEN PRINT CHR$(27);"$";CHR$(1):GOTO
   85Ø
84Ø PRINT CHR$(27);"X";CHR$(1)
```

```
850 FOR I=1 TO 20:PRINT CHR$(N1);" ";:NEXT I:PRINT
860 PRINT CHR$(14);:FOR I=1 TO 10:PRINT CHR$(N1);"
   ";:NEXT I:PRINT
870 PRINT CHR$(15);:FOR I=1 TO 20:PRINT CHR$(N1);:
   NEXT I:PRINT
880 IF AN$="N" THEN PRINT CHR$(27);"$";CHR$(0);:GOTO
   890
885 PRINT CHR$(27);"X";CHR$(0)
890 PRINT CHR$(27);"@":MM$="":PRINT#4:CLOSE4:RETURN
900 PRINT CHR$(147)
920 PRINT "     M1 M2 M3 M4 M5 M6 M7 M8 M9 M10 M11"
930 FOR I=0 TO 7 :PRINT "     ";:FOR J=1 TO 11
940 PRINT "!--";:NEXT J:PRINT "!":IF I<7 THEN
   PRINT  2^I
950 NEXT I
960 B$="R)GT L)FT U)P D)WN I)NSRT C)LEAR Q)UIT"
970 GOSUB 1000
975 X=0:Y=0:C$=CS$:GOSUB 11000
980 RETURN
1000 FOR I=1 TO LEN(B$)
1010 Z=ASC(MID$(B$,I,1))
1020 IF Z>64 AND Z<91 THEN Z=Z-64
1030 POKE AD+I,Z
1040 NEXT I
1050 RETURN
11000 POKE DD+X+Y,ASC(C$):RETURN
12000 FOR I=1 TO 40:POKE I+1943,32:NEXT I:RETURN
```

Delta Plot program

```
10 REM COMMODORE 64 <> DELTAPLT
50 PRINT CHR$(147)
60 FOR I=1 TO 5:PRINT " ":NEXT I
70 PRINT "THIS  PROGRAM  TAKES ABOUT ONE MINUTE "
80 PRINT "TO RUN, SO PLEASE TURN ON YOUR PRINTER"
90 PRINT "AND STAND BY........................."
100 REM DELTA-PLOT
110 DIM BIT%(76,14)
1000 REM SET PROGRAM CONSTANTS
1010 MASK%(1) = 64   : MASK%(4) = 8
1020 MASK%(2) = 32   : MASK%(5) = 4
1030 MASK%(3) = 16   : MASK%(6) = 2
1040 LX = 20         : LY = 20
1050 XFAC = 72/LX    : YFAC = 87/LY
2000 REM PLOT CURVE
2010 RAD = 9
```

```
2Ø2Ø X1 = 19          : Y1 = 1Ø
2Ø3Ø FOR ANG = Ø TO 36Ø STEP 1Ø
2Ø4Ø R1 = ANG*6.28/36Ø
2Ø5Ø X2 = RAD*COS(R1)+1Ø : Y2 = RAD*SIN(R1)+1Ø
2Ø6Ø GOSUB 4ØØØ
2Ø7Ø NEXT ANG
3ØØØ REM SEND BIT IMAGE MAP TO PRINTER
3ØØ5 OPEN 4,4,5
3Ø1Ø PRINT#4, CHR$(27) "A" CHR$(6)
3Ø2Ø FOR ROW = Ø TO 14
3Ø22 A$=""
3Ø25 PRINT #4,CHR$(27);"K",CHR$(75 );CHR$(Ø);
3Ø5Ø FOR COL = 1 TO 75
3Ø6Ø A$=A$+ CHR$(BIT% (COL, ROW))
3Ø7Ø NEXT COL
3Ø8Ø PRINT#4,A$
3Ø9Ø NEXT ROW
3Ø95 PRINT#4,CHR$(27);"2"
31ØØ PRINT#4 : CLOSE4
311Ø END
4ØØØ REM DRAW A LINE FROM X1,Y1, TO X2,Y2
4Ø1Ø XL = X2 - X1       : YL = Y2 = Y1
4Ø2Ø NX = ABS(XL*XFAC) : NY = ABS(YL*YFAC)
4Ø3Ø IF NX < NY THEN NX = NY
4Ø4Ø NS% = INT(NX+1)
4Ø5Ø DX = XL/NS%        : DY = YL/NS%
4Ø6Ø FOR I=1 TO NS%
4Ø7Ø X1 = X1 + DX       : Y1 = Y1 + DY
4Ø8Ø GOSUB 5ØØØ
4Ø9Ø NEXT I
41ØØ RETURN
5ØØØ REM PLOT A POINT AT X1,Y1
5Ø1Ø XX = X1 * XFAC     : YY = Y1 * YFAC
5Ø2Ø COL% = INT(XX)+1
5Ø3Ø ROW% = INT(YY/6)
5Ø4Ø XIT% = INT(YY - ROW% * 6) + 1
5Ø5Ø BIT%(COL%,ROW%) = BIT%(COL%,ROW%) OR
   MASK%(XIT%)
5Ø6Ø RETURN
```

Pie chart program

```
5Ø PRINT CHR$(147);"PLEASE STAND BY"
1ØØ REM PIECHART
```

```
11Ø DIM BIT%(19Ø,36), A$(36), PCT%(25), TXT$(42),
   PTXT$(25)
12Ø ES$=CHR$(27):LF$=CHR$(1Ø)
13Ø FF$=CHR$(12):VT$=CHR$(11)
14Ø EM$=ES$ + "E":CE$=ES$ + "F"
15Ø FOR I = 1 TO 168:SP$=SP$ + CHR$(Ø): NEXT I
1ØØØ REM SET PROGRAM CONSTANTS
1Ø1Ø MASK%(1)=64:MASK%(4)=8
1Ø2Ø MASK%(2)=32:MASK%(5)=4
1Ø3Ø MASK%(3)=16:MASK%(6)=2
1Ø4Ø LX = 2Ø: LY = 2Ø
1Ø5Ø XFAC=19Ø/LX:YFAC=216/LY
1Ø6Ø FOR I= 1 TO 42
1Ø7Ø FOR J= 1 TO 8Ø:TXT$(I)=TXT$(I) + " "
1Ø8Ø NEXT J:NEXT I
1Ø9Ø GOSUB 7ØØØ
1Ø92 PRINT CHR$(147): PRINT " ":PRINT " "
1Ø94 PRINT " ":PRINT " "
1Ø96 PRINT "THIS PROGRAM TAKES ABOUT"
1Ø97 PRINT "4 MINUTES TO RUN. PLEASE"
1Ø98 PRINT "TURN ON YOUR PRINTER AND"
1Ø99 PRINT "STAND BY................"
2ØØØ REM PLOT CURVE
2Ø1Ø RAD=9
2Ø2Ø X1=19:Y1=1Ø
2Ø25 PRINT " ";
2Ø3Ø FOR ANG=Ø TO 36Ø STEP 5
2Ø4Ø R1=ANG*6.28/36Ø
2Ø5Ø X2=RAD*COS(R1) + 1Ø:Y2=RAD*SIN(R1) + 1Ø
2Ø6Ø GOSUB 4ØØØ
2Ø7Ø NEXT ANG
2Ø8Ø FOR PI= 1 TO NP%
2Ø9Ø X1=1Ø:Y1=1Ø
21ØØ TP%=TP%+PCT%(PI)
211Ø ANG=36Ø*TP%*.Ø1
212Ø R1=ANG*6.28/36Ø
213Ø X2=RAD*COS(R1)+1Ø:Y2=RAD*SIN(R1) + 1Ø
214Ø GOSUB 4ØØØ
215Ø GOSUB 6ØØØ
216Ø NEXT PI
3ØØØ REM SEND BIT IMAGE MAP TO PRINTER
3Ø2Ø FOR ROW= Ø TO 35
3Ø22 A$(ROW)= ""
3Ø5Ø FOR COL= 1 TO 19Ø
3Ø6Ø A$(ROW)=A$(ROW)+CHR$(BIT%(COL,ROW))
```

```
3070 NEXT COL
3080 NEXT ROW
3100 OPEN4,4:CMD4
3110 X=(40-LEN(T$)/2)
3120 FOR I= 1 TO X: PRINT " ";:NEXT I
3130 PRINT EM$;T$ ;CE$;LF$
3140 PRINT VT$;VT$;VT$
3150 PRINT ES$;"A";CHR$(3)
3160 PRINT TXT$(1);LF$;TXT$(2);LF$;TXT$(3);LF$
3170 FOR ROW= 0 TO 35
3180 PRINT ES$;"K";CHR$(102);CHR$(1);SP$;A$(ROW)
3200 PRINT TXT$(ROW + 4)
3210 NEXT ROW
3220 PRINT TXT$(40);LF$
3230 PRINT TXT$(41);LF$
3240 PRINT TXT$(42);LF$
3250 PRINT ES$;"2";FF$
3254 PRINT#4:CLOSE4
3255 PRINT CHR$(147)
3260 END
4000 REM  DRAW A LINE FROM X1,Y1 TO X2,Y2
4010 XL=X2-X1:YL=Y2-Y1
4020 NX=ABS(XL*XFAC):NY=ABS(YL*YFAC)
4030 IF NX<NY THEN NX=NY
4040 NS%=INT(NX + 1)
4050 DX=XL/NS%:DY=YL/NS%
4060 FOR I= 1 TO NS%
4070 X1=X1 + DX:Y1=Y1+DY
4080 GOSUB 5000
4090 NEXT I
4100 RETURN
5000 REM  PLOT A POINT AT X1,Y1
5010 XX=X1*XFAC:YY=Y1*YFAC
5020 COL=INT(XX) + 1
5030 ROW=INT(YY/6)
5040 XIT%=INT(YY-(6*ROW)) + 1
5050 BIT%(COL,ROW)=BIT%(COL,ROW) OR MASK%(XIT%)
5060 RETURN
6000 REM
6010 MA%=(ANG + PA%)/2
6020 R1=MA%*6.28/360
6030 X3=INT(20*SIN(R1)):Y3=INT(22*COS(R1))
6040 X4=22 + X3: Y4= 40 + Y3
6050 IF MA%>270 OR MA%<90 THEN GOSUB 6100: GOTO 6070
6060 GOSUB 6200
```

```
6Ø7Ø PA%=ANG
6Ø8Ø RETURN
61ØØ MM$=TXT$(X4)
61Ø2 LL$=LEFT$(MM$,Y4)
61Ø4 PP=LEN(PTXT$(PI))
61Ø6 RR$=RIGHT$(MM$,8Ø-(Y4+PP))
61Ø8 TXT$(X4)=LL$ + PTXT$(PI) +RR$
611Ø RETURN
62ØØ MM$=TXT$(X4)
62Ø2 PP=LEN (PTXT$(PI))
62Ø4 LL$=LEFT$(MM$,(Y4 - PP))
62Ø6 RR$=RIGHT$(MM$, (8Ø - Y4))
62Ø8 TXT$(X4) = LL$ + PTXT$(PI) +RR$
621Ø RETURN
7ØØØ REM
7ØØ5 I=1
7Ø1Ø PRINT CHR$(147): PRINT:PRINT:PRINT:PRINT
7Ø2Ø INPUT "ENTER TITLE FOR CHART  ";T$
7Ø3Ø AS%=Ø:AL%=1ØØ
7Ø4Ø PRINT CHR$(147)
7Ø5Ø PRINT "TOTAL SO FAR      :  ";AS%
7Ø6Ø PRINT "TOTAL REMAINING   :  ";AL%
7Ø7Ø INPUT "ENTER % FOR FIELD  ";PCT%(I)
7Ø8Ø IF PCT%(I)>AL% OR PCT%(I)=Ø THEN PCT%(I)=AL%
7Ø9Ø AL%=AL%-PCT%(I)
71ØØ AS%=AS%+PCT%(I)
711Ø INPUT "ENTER DESCRIPTION OF FIELD :  ";PTXT$(I)
712Ø IF LEN(PTXT$(I))> 15 THEN PRINT "FIELD TOO LONG
   - 15 CHAR. MAX": GOTO 711Ø
713Ø IF AL%=Ø THEN GOTO 72ØØ
714Ø I=I+1
715Ø GOTO 7Ø4Ø
72ØØ NP%=I
721Ø IF NP%=1 THEN 7Ø4Ø
722Ø PRINT CHR$(147)
723Ø RETURN
```

Appendix H

DIP Switch Settings

The DIP (dual in-line package) switches control some of the functions of Delta. A DIP switch actually contains several individual switches. Delta has two DIP switches with 8 individual switches in them and one DIP switch with 4 individual switches. Figure H-1 is a drawing of a typical DIP switch.

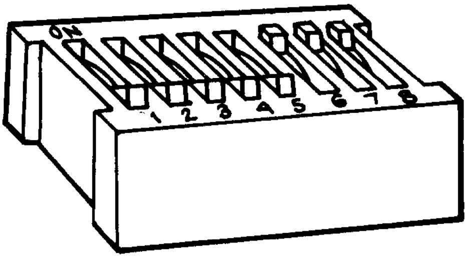


Figure H-1. *A DIP switch is actually a series of several small switches.*

DIP switch 2 is accessible from the rear of the printer, but to get to DIP switch 1 and DIP switch 3 you must remove the upper case. Chapter 10 tells you how to remove it.

Never change the settings of any of the DIP switches when the power is on. Turn off both the printer and your computer.

Table H-1 summarizes the functions of DIP switches 1 and 2. DIP switch 3 controls the serial interface and is covered in Appendix P. The individual switches on DIP switch 1 are called 1-1 through 1-8; those on switch 2 are 2-1 through 2-4.

Table H-1
DIP Switch Settings

Switch	ON	OFF
Switch 1		
1-1	11″ page length	12″ page length
1-2	Normal print	Emphasized print
1-3	10 CPI (pica pitch)	17 CPI (condensed pitch)
1-4	Normal	Italic
1-5	1/6″ line feed	1/8″ line feed
1-6	International character set selection	
1-7		
1-8		
Switch 2		
2-1	Paper-out detector on	Ignore paper-out
2-2	Serial interface	Parallel interface
2-3	7-bit interface	8-bit interface
2-4	Auto LF with CR	LF must be from host

DIP switch 1 controls the default settings for printing functions. It is located inside the case at the left rear. Figure H-2 shows the location of this switch. You must open the case to change the settings of this switch.

DIP switch 2 controls the interface. It can be reached from the back of the printer without opening the case. Figure H-3 shows the location of switch 2.

Switch Functions

Switch	**Function**
1-1	Switch 1-1 sets the default page length for Delta. If switch 1-1 is ON, the page length is set to 11″. When switch 1-1 is OFF the page length is set to 12″. This switch is set ON at the factory.
1-2	This switch selects either normal or emphasized print for the default. If this switch is ON then Delta will print normal type when the power is turned on. If this switch is OFF then Delta will print emphasized type when the power is turned on. This switch is set ON at the factory.

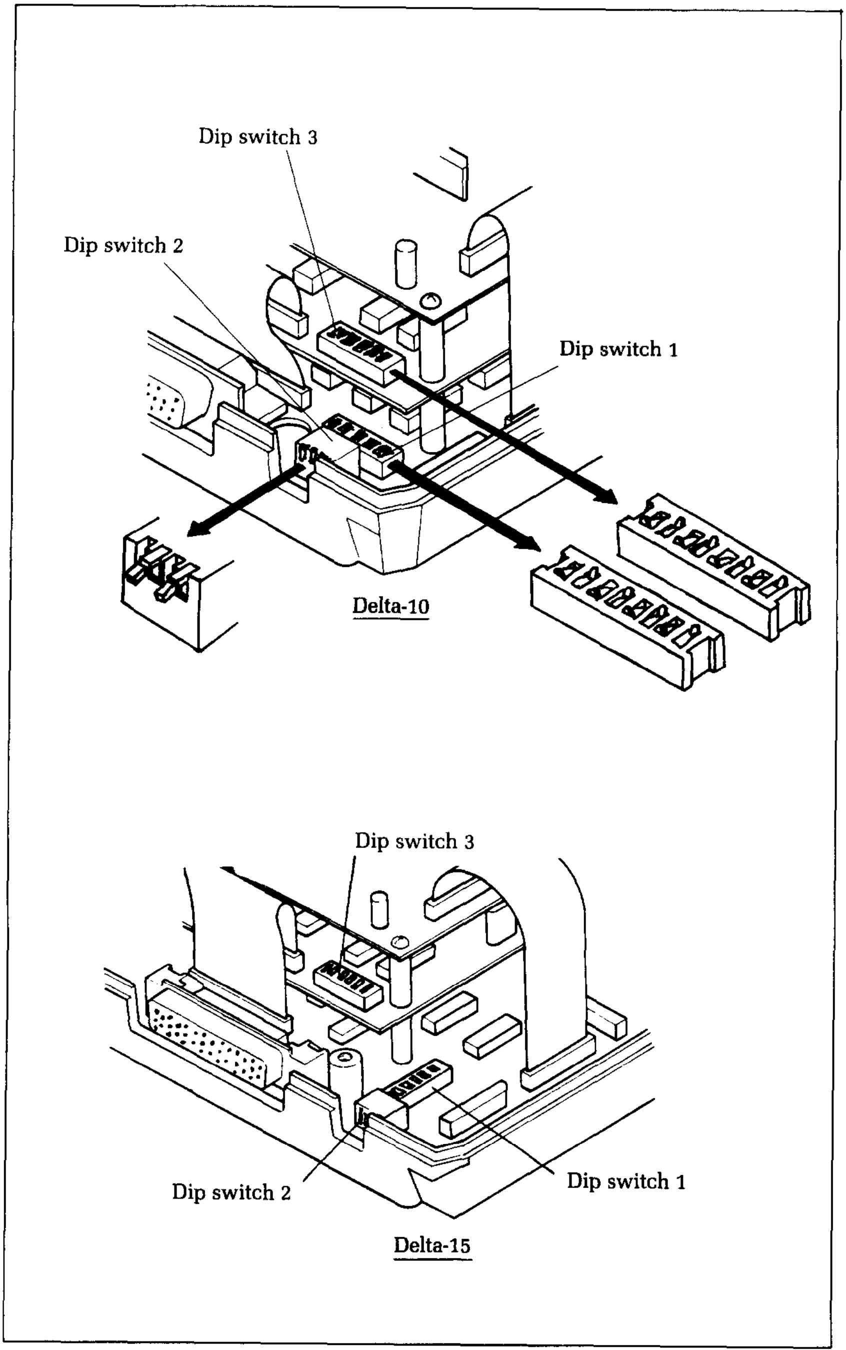


Figure H-2. *DIP switch 1, located inside Delta's case, controls default printing functions*

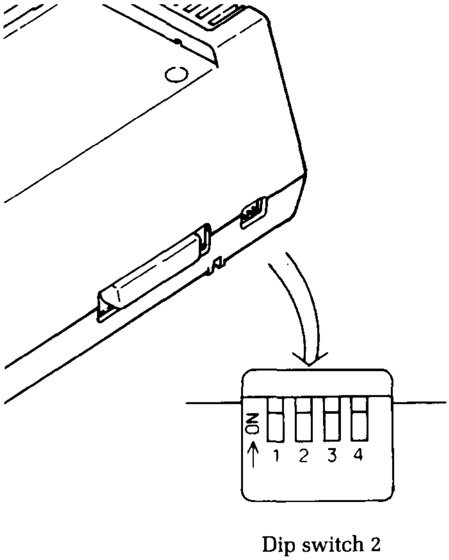


Figure H-3. *DIP switch 2, which controls the interface, is located on the back of the printer.*

1-3 This switch selects the default character pitch. If this switch is ON the default pitch is 10 CPI. If this switch is OFF the default pitch is 17 CPI. This switch is set ON at the factory.

1-4 Switch 1-4 selects the default character style. If this switch is ON then the default character style is normal characters. If this switch is OFF then the default character style is italic. This switch is set ON at the factory.

1-5 This switch sets the default line spacing. When this switch is ON the default line spacing is set to 1/6 inch. This means that Delta will advance the paper 1/6 inch each time it receives a line feed. When this switch is OFF the default line spacing is 1/8 inch. This switch is set ON at the factory.

1-6 - 1-8 These three switches determine the default international character set as shown in Table H-2. These switches are all set ON at the factory.

Table H-2
International character sets

Switch	USA	England	Germany	Denmark	France	Sweden	Italy	Spain
1-6	ON	OFF	ON	OFF	ON	OFF	ON	OFF
1-7	ON	ON	OFF	OFF	ON	ON	OFF	OFF
1-8	ON	ON	ON	ON	OFF	OFF	OFF	OFF

2-1 This switch disables the paper-out sensor. If this switch is ON the printer will signal the computer when it runs out of paper and will stop printing. If this switch is off the printer will ignore the paper-out sensor and will continue printing. This switch is set ON at the factory.

2-2 This switch selects the active interface. Turn this switch ON to use the serial interface. Turn this switch OFF to use the parallel interface. This switch is set OFF at the factory.

2-3 This switch controls the eighth bit of the parallel interface. If this switch is ON the printer will only read the first seven bits on the parallel interface and ignores the eighth bit. If this switch is OFF all eight bits will be read. This switch is set OFF at the factory.

2-4 When this switch is ON, Delta will automatically advance the paper one line every time it receives a carriage return. When this switch is OFF, the computer must send a line feed command every time the paper is to advance. (Most BASICs send a line feed with every carriage return, therefore, this switch should usually be off.) This switch is set OFF at the factory.

Appendix I
ASCII Codes

Standard and Italic Characters

Decimal	Character		Function	Decimal	Character		
0	NUL		End tab settings	47	/	/	
7	BEL		Bell	48	0	0	
8	BS		Backspace	49	1	1	
9	HT		Horizontal tab	50	2	2	
10	LF		Line feed	51	3	3	
11	VT		Vertical tab	52	4	4	
12	FF		Form feed	53	5	5	
13	CR		Carriage return	54	6	6	
14	SO		Expanded print on	55	7	7	
15	SI		Condensed print on	56	8	8	
17	DC1		On line	57	9	9	
18	DC2		Pica pitch	58	:	:	
19	DC3		Off line	59	;	;	
20	DC4		Expanded print off	60	<	<	
27	ESC		Escape	61	=	=	
30	RS		End macro	62	>	>	
32			Space	63	?	?	
33	!	!		64	@	@	*
34	"	"		65	A	A	
35	#	#	*	66	B	B	
36	$	$		67	C	C	
37	%	%		68	D	D	
38	&	&		69	E	E	
39	'	'	Apostrophe	70	F	F	
40	(	(		71	G	G	
41	)	)		72	H	H	
42	*	*		73	I	I	
43	+	+		74	J	J	
44	,	,	Comma	75	K	K	
45	-	-	Hyphen	76	L	L	
46	.	.	Period	77	M	M	

*These characters may be different if you are using an international character set other than the USA set. The characters for each set are shown on the next page.

Decimal	Character		
78	N	N	
79	O	O	
80	P	P	
81	Q	Q	
82	R	R	
83	S	S	
84	T	T	
85	U	U	
86	V	V	
87	W	W	
88	X	X	
89	Y	Y	
90	Z	Z	
91	[	[	*
92	\	\	*
93	]	]	*
94	^	^	*
95	_	_	
96	`	`	*
97	a	a	
98	b	b	
99	c	c	
100	d	d	
101	e	e	
102	f	f	

Decimal	Character		
103	g	g	
104	h	h	
105	i	i	
106	j	j	
107	k	k	
108	l	l	
109	m	m	
110	n	n	
111	o	o	
112	p	p	
113	q	q	
114	r	r	
115	s	s	
116	t	t	
117	u	u	
118	v	v	
119	w	w	
120	x	x	
121	y	y	
122	z	z	
123	{	{	*
124	|	|	*
125	}	}	*
126	~	~	*
127	DEL		Delete

*These characters may be different if you are using an international character set other than the USA set. The characters for each set are shown below.

International Character Sets

Decimal	USA	England	Germany	Denmark	France	Sweden	Italy	Spain
35	#	£	#	#	£	#	#	#
64	@	@	§	@	à	É	§	@
91	[	[	Ä	Æ	°	Ä	°	¡
92	\	\	Ö	Ø	ç	Ö	ç	Ñ
93	]	]	Ü	Å	§	Å	é	¿
94	^	^	^	^	^	Ü	^	^
96	`	`	`	`	`	é	ù	`
123	{	{	ä	æ	é	ä	à	¨
124	|	|	ö	ø	ù	ö	ò	ñ
125	}	}	ü	å	è	å	è	}
126	~	~	ß	~	¨	ü	ì	~

Special Characters

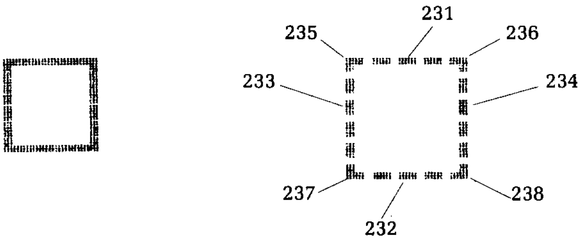
Decimal	Character	Function
135	BEL	Bell
136	BS	Backspace
137	HT	Horizontal tab
138	LF	Line feed
139	VT	Vertical tab
140	FF	Form feed
141	CR	Carriage return
142	SO	Expanded print on
143	SI	Condensed print on
145	DC1	On line
146	DC2	Pica pitch
147	DC3	Off line
148	DC4	Expanded print off
155	ESC	Escape
158	RS	End macro
160	◞	
161	◝	
162	◟	
163	◜	
164	↑	
165	↓	
166	←	
167	→	
168	○	
169	⊥	
170	⊤	
171	⊢	
172	⊣	
173	◇	
174	◆	
175	□	
176	Ts	
177	Å	
178	ф	
179	θ	
180	Ŀ	
181	₣	
182	Ω	
183	℧	
184	Σ	

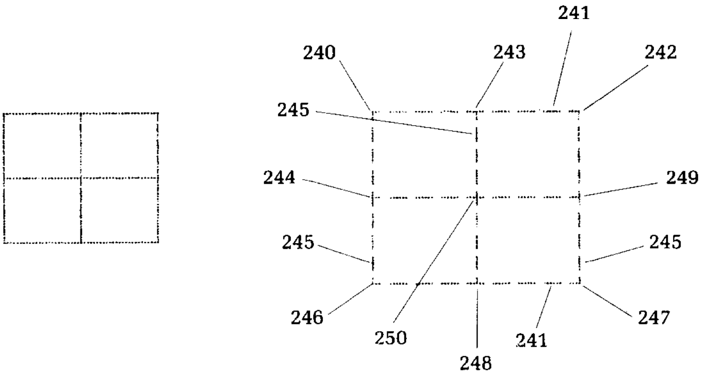
Decimal	Character
185	δ
186	⊗
187	π
188	±
189	⊐
190	×
191	÷
192	Ā
193	à
194	ç
195	ŧ
196	ā
197	µ
198	°
199	ˀ
200	†
201	§
202	Ē
203	©
204	¼
205	⊠
206	½
207	‼
208	¥
209	Ä
210	Ö
211	Ü
212	¢
213	Ñ
214	ä
215	ö
216	ü
217	₧
218	ē
219	é
220	ú
221	è
222	ñ
223	ƒ

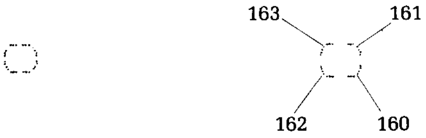
Block Graphics Characters

Decimal	Character
224	Space
225	▘
226	▖
227	▝
228	▗
229	▚
230	▞
231	▀
232	▄
233	▌
234	▐
235	▛
236	▜
237	▙
238	▟
239	█

Decimal	Character
240	┌
241	─
242	┐
243	┬
244	├
245	│
246	└
247	┘
248	┴
249	┤
250	┼
251	┏
252	┛
253	┓
254	┗


231
235
236
233
234
237
238
232

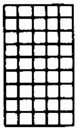
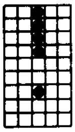
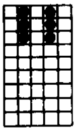
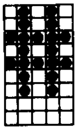

241
240
243
242
245
244
249
245
245
246
250
241
247
248

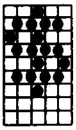
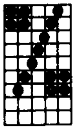
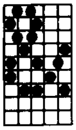
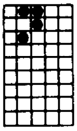

163
161
162
160

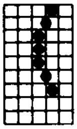
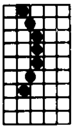
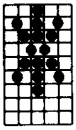
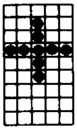
Appendix J

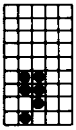
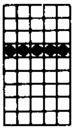
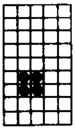
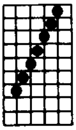
Character Style Charts

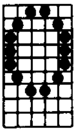
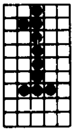
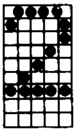
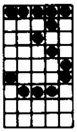
Standard Characters

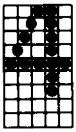
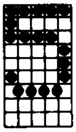
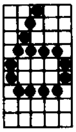
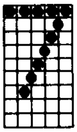
32

33

34

35


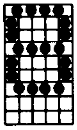
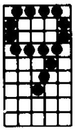
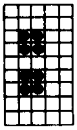
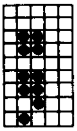
36

37

38

39


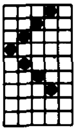
40

41

42

43


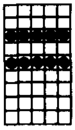
44

45

46

47


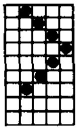
48

49

50

51


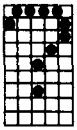
52

53

54

55


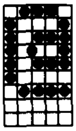
56

57

58

59


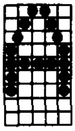
60 

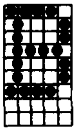
61 

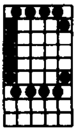
62 

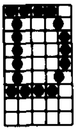
63 

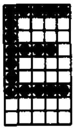
64 

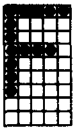
65 

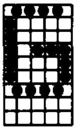
66 

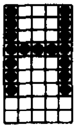
67 

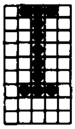
68 

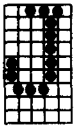
69 

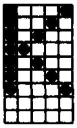
70 

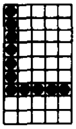
71 

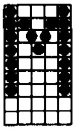
72 

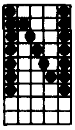
73 

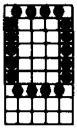
74 

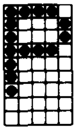
75 

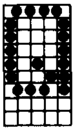
76 

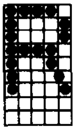
77 

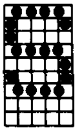
78 

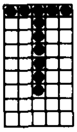
79 

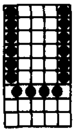
80 

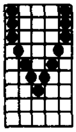
81 

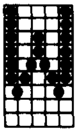
82 

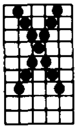
83 

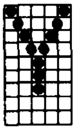
84 

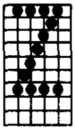
85 

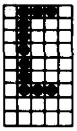
86 

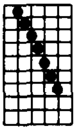
87 

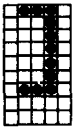
88 

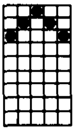
89 

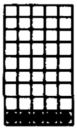
90 

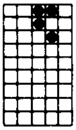
91 

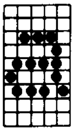
92 

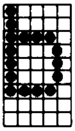
93 

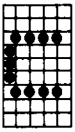
94 

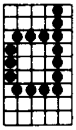
95 

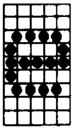
96


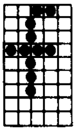
97


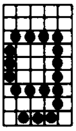
98


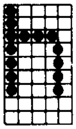
99


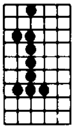
100


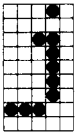
101


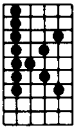
102


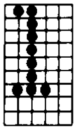
103


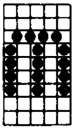
104


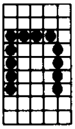
105


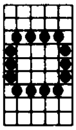
106


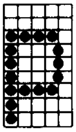
107


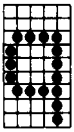
108


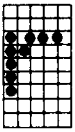
109


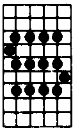
110


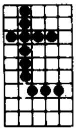
111


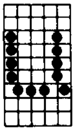
112


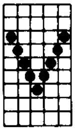
113


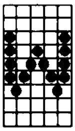
114


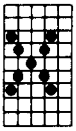
115


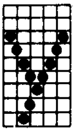
116


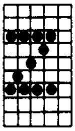
117


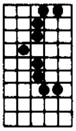
118


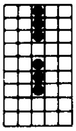
119


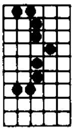
120


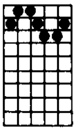
121


122


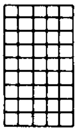
123


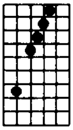
124


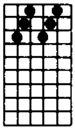
125


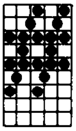
126


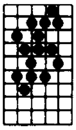
Italic Characters

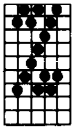
32


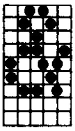
33


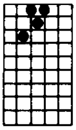
34


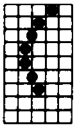
35


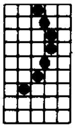
36


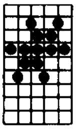
37


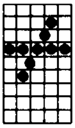
38


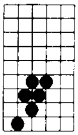
39


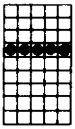
40


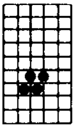
41


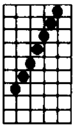
42


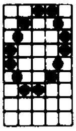
43


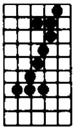
44


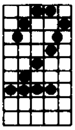
45


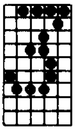
46


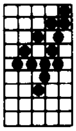
47


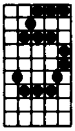
48


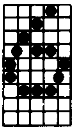
49


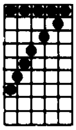
50


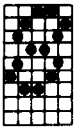
51


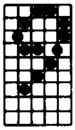
52


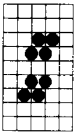
53


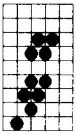
54


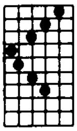
55


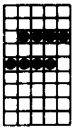
56


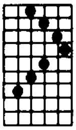
57


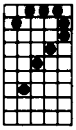
58


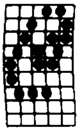
59


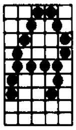
60


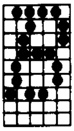
61


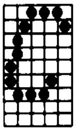
62


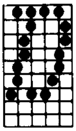
63


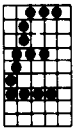
64


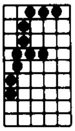
65


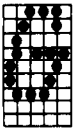
66


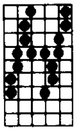
67


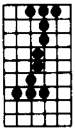
68


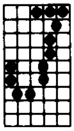
69


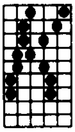
70


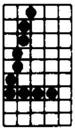
71


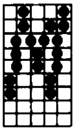
72


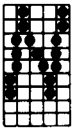
73


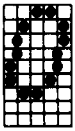
74


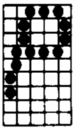
75


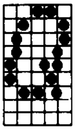
76


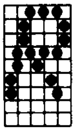
77


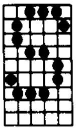
78


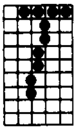
79


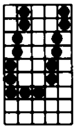
80


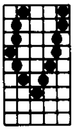
81


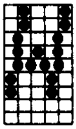
82


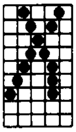
83


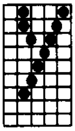
84


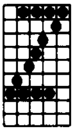
85


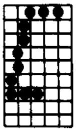
86


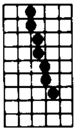
87


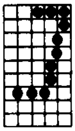
88


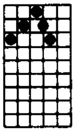
89


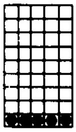
90


91


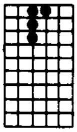
92


93


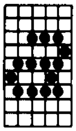
94


95


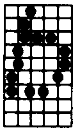
96



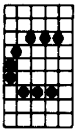
97



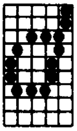
98



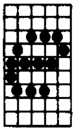
99



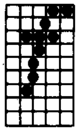
100



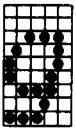
101



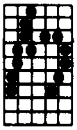
102



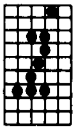
103



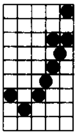
104



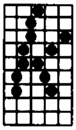
105



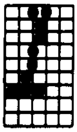
106



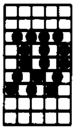
107



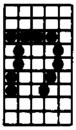
108



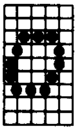
109



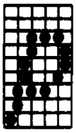
110



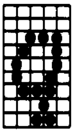
111



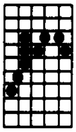
112



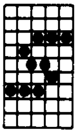
113



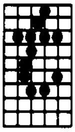
114



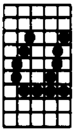
115



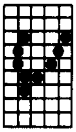
116



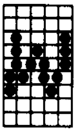
117



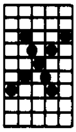
118



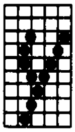
119



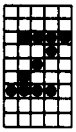
120



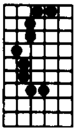
121



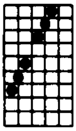
122



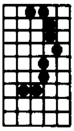
123



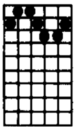
124



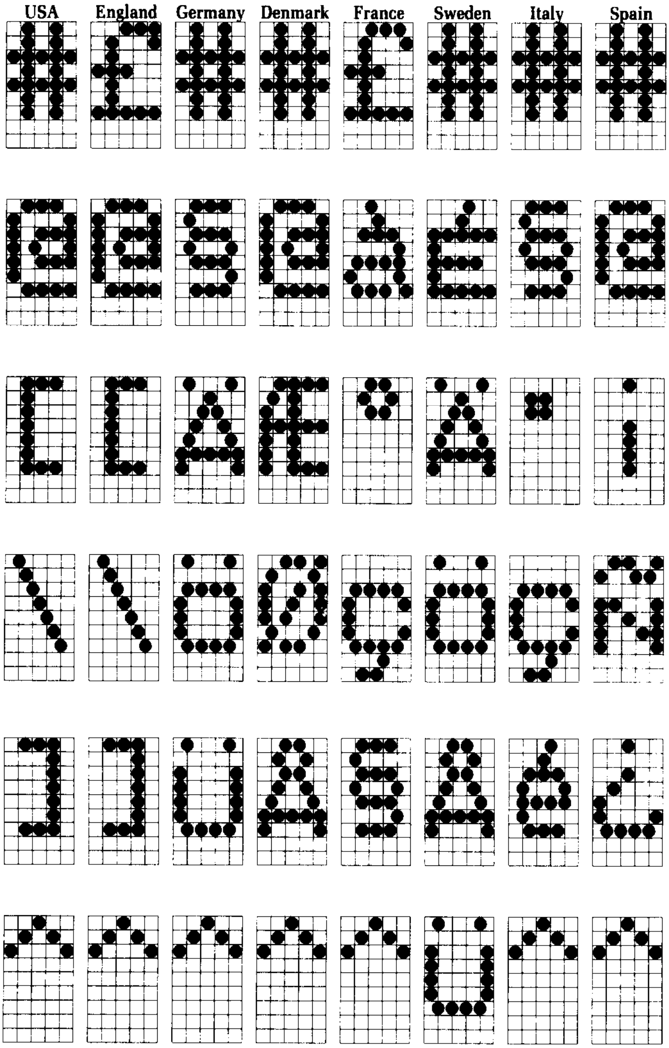
125



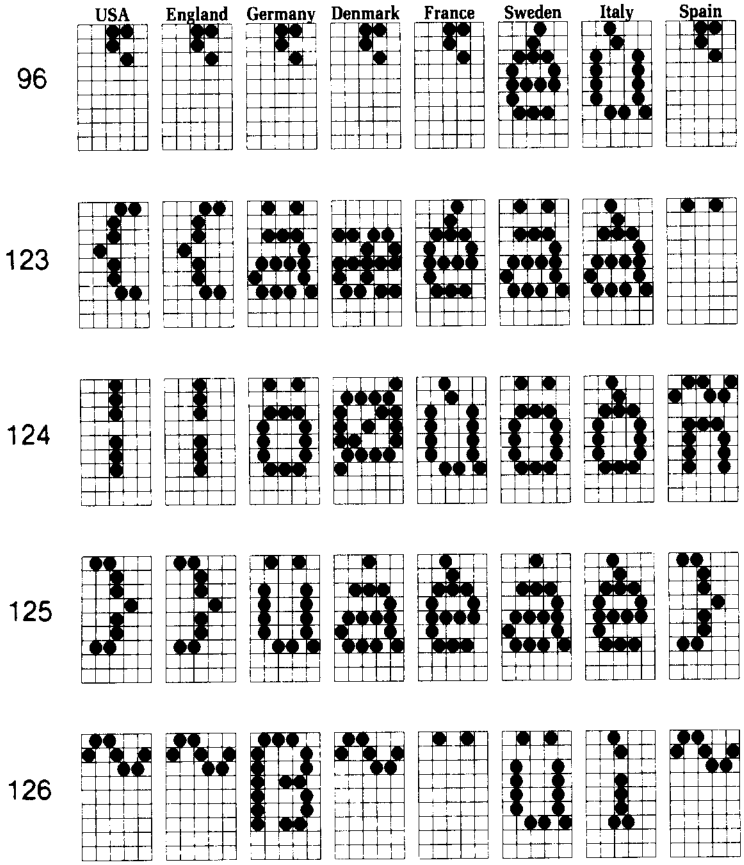
126



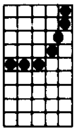
International Characters

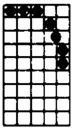


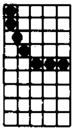
	USA	England	Germany	Denmark	France	Sweden	Italy	Spain
35	#	£	#	#	£	#	#	#
64	@	@	§	@	à	É	§	@
91	[	[	Ä	Æ	°	Ä	°	¡
92	\	\	Ö	Ø	ç	Ö	ç	Ñ
93	]	]	Ü	Å	§	Å	é	¿
94	^	^	^	^	^	Ü	^	^

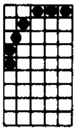

USA
England
Germany
Denmark
France
Sweden
Italy
Spain
96
123
124
125
126

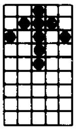
Special Characters

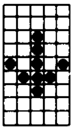
160


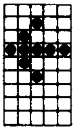
161


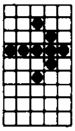
162


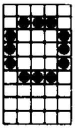
163


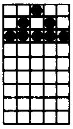
164


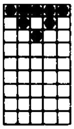
165


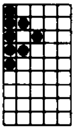
166


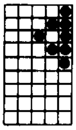
167


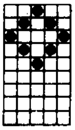
168


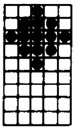
169


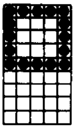
170


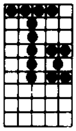
171


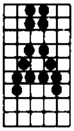
172


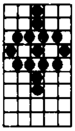
173


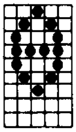
174


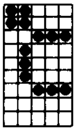
175


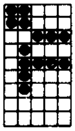
176


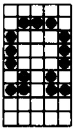
177


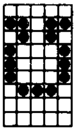
178


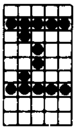
179


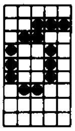
180


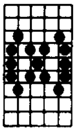
181


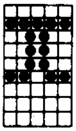
182


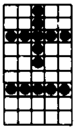
183


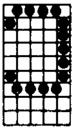
184


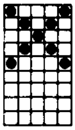
185


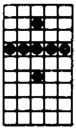
186


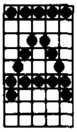
187


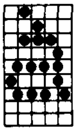
188


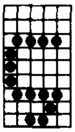
189


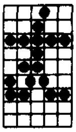
190


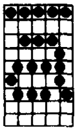
191


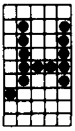
192


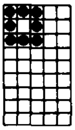
193


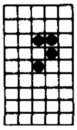
194


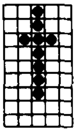
195


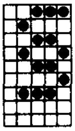
196


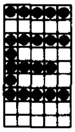
197


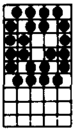
198


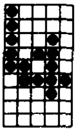
199


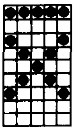
200


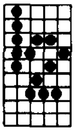
201


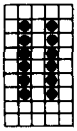
202


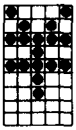
203


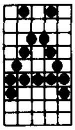
204


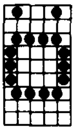
205


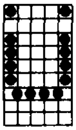
206


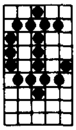
207


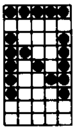
208


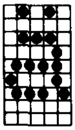
209


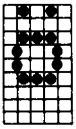
210


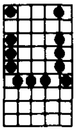
211


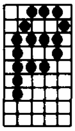
212


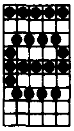
213


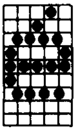
214


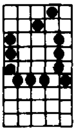
215


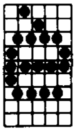
216


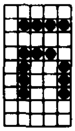
217


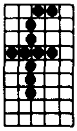
218


219


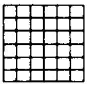
220


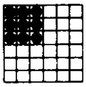
221


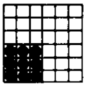
222


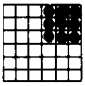
223


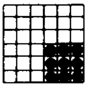
Block Graphics Characters

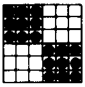
224 

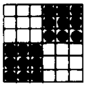
225 

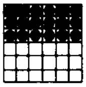
226 

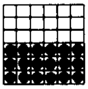
227 

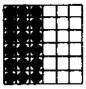
228 

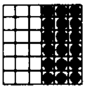
229 

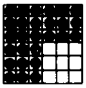
230 

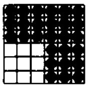
231 

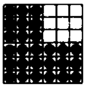
232 

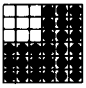
233 

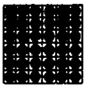
234 

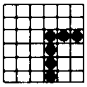
235 

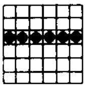
236 

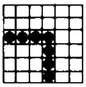
237 

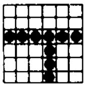
238 

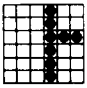
239 

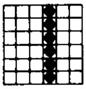
240 

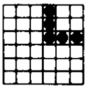
241 

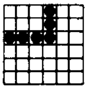
242 

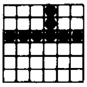
243 

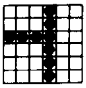
244 

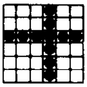
245 

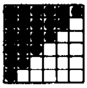
246 

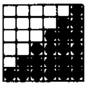
247 

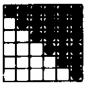
248 

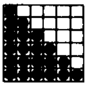
249 

250 

251 

252 

253 

254 

Appendix K

Function Code Reference

The purpose of this Appendix is to provide a quick reference for the various functions available on the Delta-10 and Delta-15. The descriptions of the codes appear in the following format:

PURPOSE:	**Tells what the function code does.**
CODE:	Control code mnemonic
(decimal ASCII)	ASCII decimal equivalent
(hex ASCII)	Hexadecimal equivalent
REMARKS:	Details how the command is used.
REFERENCE:	Tells which chapter of the manual describes the command in greater detail

There are several commands that require that you specify a value (or values) to Delta. In these cases, we have used an italic "*n*" or "*m*" to indicate a variable. You should insert the ASCII code for proper value here.

Commands to Control Print Style

These commands are used to control the font style, the print pitch, and special effects.

Font style controls

PURPOSE: **Select the standard character set.**

CODE:	〈ESC〉	"5"
(decimal ASCII)	27	53
(hex ASCII)	1B	35

REMARKS: This command causes the printer to cancel the italic character set and select instead the standard character set. You can select the standard character set as the power-on default by turning DIP switch 1-4 on.

REFERENCE: Chapter 3

PURPOSE: **Select the italic character set.**

CODE:	〈ESC〉	"4"
(decimal ASCII)	27	52
(hex ASCII)	1B	34

REMARKS: This command selects the italic character set. You can select the italic character set as the power-on default by turning DIP switch 1-4 off.

REFERENCE: Chapter 3

PURPOSE: **Select an international character set.**

CODE:	⟨ESC⟩	"7"	n
(decimal ASCII)	27	55	n
(hex ASCII)	1B	37	n

REMARKS: This command causes the printer to select an international character set determined by the value of n as shown in the table below:

n	Character set
0	U.S.A.
1	England
2	Germany
3	Denmark
4	France
5	Sweden
6	Italy
7	Spain

You can select a particular international character set as a power-on default, by adjusting the settings of DIP switches 1-6, 1-7, and 1-8.

REFERENCE: Chapter 6

Font pitch controls

PURPOSE: **Set the print pitch to pica (10 characters/inch).**

CODE:	⟨ESC⟩	"B"	1
(decimal ASCII)	27	66	1
(hex ASCII)	1B	42	01

REMARKS: This command causes all subsequent printing to be done in pica type. This command also sets the maximum number of print columns to 80 on the Delta-10 and 136 on the Delta-15. You can select pica type as the power-on default by turning DIP switch 1-3 on.

REFERENCE: Chapter 3

PURPOSE: **Set the print pitch to elite (12 characters/inch).**

CODE:	⟨ESC⟩	"B"	2
(decimal ASCII)	27	66	2
(hex ASCII)	1B	42	02

REMARKS: This command causes all subsequent printing to be done in elite type. This command also sets the maximum number of print columns to 96 on the Delta-10 and 163 on the Delta-15.

REFERENCE: Chapter 3

PURPOSE: **Set the print pitch to condensed (17 characters/inch).**

CODE:	⟨ESC⟩	"B"	3
(decimal ASCII)	27	66	3
(hex ASCII)	1B	42	03

REMARKS: This command causes all subsequent printing to be done in condensed type of 17 characters per inch. This command also sets the maximum number of print columns to 136 on the Delta-10 and 233 on the Delta-15. You can select condensed type as the power-on default by turning DIP switch 1-3 off.

REFERENCE: Chapter 3

PURPOSE: **Set the print pitch to pica (10 characters/inch).**

CODE:	⟨DC2⟩
(decimal ASCII)	18
(hex ASCII)	12

REMARKS: This command is the same as ⟨ESC⟩ "B" 1, but can be used in applications where a single-character command is required.

REFERENCE: Chapter 3.

PURPOSE: **Set the print pitch to condensed (17 characters/inch).**

CODE:	⟨SI⟩
(decimal ASCII)	15
(hex ASCII)	0F

REMARKS: This command is the same as ⟨ESC⟩ "B" 3, but can be used in applications where a single-character command is required.

REFERENCE: Chapter 3

PURPOSE: **Set the print pitch to condensed (17 characters/inch).**

CODE:	⟨ESC⟩	⟨SI⟩
(decimal ASCII)	27	15
(hex ASCII)	1B	0F

REMARKS: Same as ⟨SI⟩, above.

PURPOSE: **Set the printer to expanded print.**

CODE:	⟨ESC⟩	"W"	1
(decimal ASCII)	27	87	1
(hex ASCII)	1B	57	01

REMARKS: This command causes all subsequent printing to be in expanded type. The size of the type is determined by the normal type size at the time the command is sent:

	Normal	Expanded
Pica	10 CPI	5 CPI
Elite	12 CPI	6 CPI
Condensed	17 CPI	8.5 CPI

REFERENCE: Chapter 3

PURPOSE: **Set the printer to expanded print for the remainder of the current line.**

CODE:	〈SO〉
(decimal ASCII)	14
(hex ASCII)	0E

REMARKS: This command causes the printer to print expanded characters until a carriage return is sent. The character widths are shown above in the description of the 〈ESC〉 "W" 1 command.

REFERENCE: Chapter 3

PURPOSE: **Set the printer to expanded print for the remainder of the current line.**

CODE:	〈ESC〉	〈SO〉
(decimal ASCII)	27	14
(hex ASCII)	1B	0E

REMARKS: Same as 〈SO〉, above.

REFERENCE: Chapter 3

PURPOSE: **Cancels expanded print.**

CODE:	〈ESC〉	"W"	0
(decimal ASCII)	27	87	0
(hex ASCII)	1B	57	00

REMARKS: This command resets the print size to whatever it was before being set to expanded print.

REFERENCE: Chapter 3

PURPOSE: **Cancels expanded print.**

CODE:	〈DC4〉
(decimal ASCII)	20
(hex ASCII)	14

REMARKS: This command is the same as 〈ESC〉 "W" 0, but can be used in applications where a single-character command is required.

REFERENCE: Chapter 3

Special print modes

PURPOSE: **Select double-strike printing.**

CODE:	⟨ESC⟩	"G"
(decimal ASCII)	27	71
(hex ASCII)	1B	47

REMARKS: This command causes all subsequent characters to be printed in double-strike mode. Double-strike mode causes all characters to be printed once, the paper moved up 1/144 inch, and the characters reprinted. Shifting in and out of double-strike mode on the same line can cause the line to slant slightly.

REFERENCE: Chapter 3

PURPOSE: **Cancel double-strike printing.**

CODE:	⟨ESC⟩	"H"
(decimal ASCII)	27	72
(hex ASCII)	1B	48

REMARKS: This command cancels double-strike printing and returns the printer to normal printing.

REFERENCE: Chapter 3

PURPOSE: **Select emphasized printing.**

CODE:	⟨ESC⟩	"E"
(decimal ASCII)	27	69
(hex ASCII)	1B	45

REMARKS: This command causes all subsequent characters to be printed in emphasized print. Emphasized print can only be used with pica-sized characters, or enlarged pica-sized characters (10 CPI and 5 CPI), and cannot be used with superscripts or subscripts. Emphasized print can, however, be used with double-strike mode to obtain "correspondence quality" printing. You can select emphasized printing as the power-on default by turning DIP switch 1-2 off.

REFERENCE: Chapter 3

PURPOSE:	**Cancel emphasized printing.**		
CODE:	⟨ESC⟩	“F”	
(decimal ASCII)	27	70	
(hex ASCII)	1B	46	
REMARKS:	This command cancels emphasized printing and returns the printer to normal printing. You can select normal printing as the power-on default by turning DIP switch 1-2 on.		
REFERENCE:	Chapter 3		

PURPOSE:	**Select underlining.**		
CODE:	⟨ESC⟩	“-”	1
(decimal ASCII)	27	45	1
(hex ASCII)	1B	2D	01
REMARKS:	This command causes all subsequent characters printed to be automatically underlined. Spaces are also underlined.		
REFERENCE:	Chapter 3		

PURPOSE:	**Cancel underlining.**		
CODE:	⟨ESC⟩	“-”	0
(decimal ASCII)	27	45	0
(hex ASCII)	1B	2D	00
REMARKS:	This command cancels underlining and returns the printer to normal printing.		
REFERENCE:	Chapter 3		

PURPOSE: **Select superscripts.**

CODE:	⟨ESC⟩	“S”	0
(decimal ASCII)	27	83	0
(hex ASCII)	1B	53	00

REMARKS: This command causes all subsequent characters to be printed as superscripts. While in superscript mode, the normal bi-directional printing is cancelled and replaced with uni-directional printing. Printing is also set to double-strike mode. Superscripts may be used in conjunction with the italic font, and in pica, elite, and condensed pitches. It may not, however, be used in conjunction with emphasized or enlarged print.

REFERENCE: Chapter 3

PURPOSE: **Select subscripts.**

CODE:	⟨ESC⟩	“S”	1
(decimal ASCII)	27	83	1
(hex ASCII)	1B	53	01

REMARKS: This command causes all subsequent characters to be printed as subscripts. The same conditions and restrictions apply for subscripts as do for superscripts.

REFERENCE: Chapter 3

PURPOSE: **Cancel superscripts and subscripts.**

CODE:	⟨ESC⟩	“T”
(decimal ASCII)	27	84
(hex ASCII)	1B	54

REMARKS: This command cancels either superscript or subscript mode. It also cancels the uni-directional printing and double-strike which the mode had set.

REFERENCE: Chapter 3

Commands to Control Vertical Position of Print Head

These commands are used to move the paper relative to the location of the print head. By moving the paper up, the print head, in effect, moves down the page.

Line feed controls

PURPOSE:	**Advance the paper one line (Line Feed).**	
CODE:	⟨LF⟩	
(decimal ASCII)	10	
(hex ASCII)	0A	
REMARKS:	The actual distance advanced by the line feed is set either through the setting of DIP switch 1-5 or through various codes which can be sent (see below). When DIP switch 2-4 is "on" a line feed is automatically generated whenever the printer receives a carriage return.	
REFERENCE:	Chapter 4	

PURPOSE:	**Change the line spacing to 1/8 inch.**	
CODE:	⟨ESC⟩	"0"
(decimal ASCII)	27	48
(hex ASCII)	1B	30
REMARKS:	This command sets the distance the paper advances during all subsequent line feeds to 1/8 inch. You can select 1/8 inch line spacing as the power-on default by turning DIP switch 1-5 off.	
REFERENCE:	Chapter 4	

PURPOSE: **Change the line spacing to 7/72 inch.**

CODE:	⟨ESC⟩	“1”
(decimal ASCII)	27	49
(hex ASCII)	1B	31

REMARKS: This command sets the actual distance the paper advances during all subsequent line feeds to 7/72 inch.

REFERENCE: Chapter 4

PURPOSE: **Change the line spacing to 1/6 inch.**

CODE:	⟨ESC⟩	“2”
(decimal ASCII)	27	50
(hex ASCII)	1B	32

REMARKS: This command sets the actual distance the paper advances during all subsequent line feeds to 1/6 inch. You can select 1/6 inch line spacing as the power-on default by turning DIP switch 1-5 on.

REFERENCE: Chapter 4

PURPOSE: **Change the line spacing to *n*/72 inch.**

CODE:	⟨ESC⟩	“A”	*n*
(decimal ASCII)	27	65	*n*
(hex ASCII)	1B	41	*n*

REMARKS: This command sets the distance the paper advances during all subsequent line feeds to *n*/72 inch. The value of *n* must be between 0 and 255.

REFERENCE: Chapter 4

PURPOSE: **Change the line spacing to n/144 inch.**

CODE:	⟨ESC⟩	"3"	n
(decimal ASCII)	27	51	n
(hex ASCII)	1B	33	n

REMARKS: This command sets the actual distance traveled during all subsequent line feeds to be *n*/144 inch. The value of *n* must be between 0 and 255.

REFERENCE: Chapter 4

PURPOSE: **Send a one-time line feed of n/144 inch.**

CODE:	⟨ESC⟩	"J"	n
(decimal ASCII)	27	74	n
(hex ASCII)	1B	4A	n

REMARKS: This command causes the printer to advance the paper *n*/144 inch. It does not change the current value of the line spacing and it does not cause a carriage return. The value of *n* must be between 0 and 255.

REFERENCE: Chapter 4

Form feed controls

PURPOSE: **Advance paper to top of next page (Form Feed).**

CODE:	⟨FF⟩
(decimal ASCII)	12
(hex ASCII)	0C

REMARKS: The actual length of a page ejected by a form feed is set either by the setting of DIP switch 1-1 or through various codes which can be sent (see below).

REFERENCE: Chapter 4

PURPOSE:	**Set page length to *n* lines.**			
CODE:	〈ESC〉	"C"	n	
(decimal ASCII)	27	67	n	
(hex ASCII)	1B	43	n	
REMARKS:	This command sets the length of all subsequent pages to n lines. The value of *n* must be between 1 and 127.			
REFERENCE:	Chapter 4			

PURPOSE:	**Set page length to *n* inches.**			
CODE:	〈ESC〉	"C"	0	n
(decimal ASCII)	27	67	0	n
(hex ASCII)	1B	43	00	n
REMARKS:	This command sets the length of all subsequent pages to n inches. The value of *n* must be between 1 and 32. You can select a power-on default form length of 11 inches or 12 inches by setting DIP switch 1-1.			
REFERENCE:	Chapter 4			

PURPOSE:	**Set the top margin.**			
CODE:	〈ESC〉	"R"	*n*	
(decimal ASCII)	27	82	*n*	
(hex ASCII)	1B	52	*n*	
REMARKS:	This command sets the margin at the top of the page to *n*-1 lines. Printing will start on line n. The default value for *n* upon power on is 1. The value of *n* must be between 1 and 16.			
REFERENCE:	Chapter 4			

PURPOSE: **Set the bottom margin.**

CODE:	⟨ESC⟩	"N"	n
(decimal ASCII)	27	78	n
(hex ASCII)	1B	4E	n

REMARKS: This command sets the margin at the bottom of the page to *n* lines. The printer will automatically execute a form feed when the number of lines left on a page is equal to *n*. The value of *n* must be between 1 and 127. This command is sometimes referred to as "skip-over-perforation."

REFERENCE: Chapter 4

PURPOSE: **Cancel top and bottom margins.**

CODE:	⟨ESC⟩	"O"
(decimal ASCII)	27	79
(hex ASCII)	1B	4F

REMARKS: This command cancels both the top margin set by ⟨ESC⟩ "R" *n* and the bottom margin set by ⟨ESC⟩ "N" *n*.

REFERENCE: Chapter 4

Vertical tabs

PURPOSE: **Advance paper to the next vertical tab position.**

CODE:	⟨VT⟩
(decimal ASCII)	11
(hex ASCII)	0B

REMARKS: This command causes the paper to be advanced to the next vertical tab position, or the top of the next page, whichever it finds first. The vertical tab positions are set upon power on at lines 6, 12, 18, 24, 30, 36, 42, 48, 54, and 60.

REFERENCE: Chapter 5

PURPOSE: **Set vertical tab positions.**

CODE:	〈ESC〉	"P"	*n1 n2 n3...*	0
(decimal ASCII)	27	80	*n1 n2 n3...*	0
(hex ASCII)	1B	50	*n1 n2 n3...*	00

REMARKS: This command cancels all current vertical tab positions and sets those defined at lines *n1*, *n2*, *n3*, etc. The maximum number of vertical tab positions allowed is 20. The ASCII 0 character is used as a command terminator. Each vertical tab position must be between 1 and 255, and they must be specified in ascending order.

REFERENCE: Chapter 5

PURPOSE: **Advance the paper *n* lines.**

CODE:	〈ESC〉	"a"	*n*
(decimal ASCII)	27	97	*n*
(hex ASCII)	1B	61	*n*

REMARKS: This command causes the printer to advance the paper *n* lines. It does not, however, change the current value of the vertical tab positions. The value of *n* must be between 1 and 255.

REFERENCE: Chapter 4

Commands to Control Horizontal Position of Print Head

PURPOSE: **Return print head to home position (Carriage Return).**

CODE:	⟨CR⟩
(decimal ASCII)	13
(hex ASCII)	0D

REMARKS: This command returns the print head to the home position (the left margin). If DIP switch 2-4 has been set on, then this command will also cause a line feed character to be generated after the carriage return, thereby advancing to the beginning of the next print line automatically.

REFERENCE: Chapter 4

PURPOSE: **Set the left print margin.**

CODE:	⟨ESC⟩	"M"	n
(decimal ASCII)	27	77	n
(hex ASCII)	1B	4D	n

REMARKS: This command sets the home position returned to during the execution of all subsequent carriage returns to be print position *n*. The power on default for n is 1. The value of n must be between 1 and 255. For Delta-10 the maximum print position for pica pitch is 80, for elite is 96, and for condensed pitch is 136. For Delta-15 the maximum print position for pica pitch is 136, for elite is 163, and for condensed pitch is 233.

REFERENCE: Chapter 5

PURPOSE: **Set the right print margin.**

CODE:	⟨ESC⟩	"Q"	*n*
(decimal ASCII)	27	81	*n*
(hex ASCII)	1B	51	*n*

REMARKS: This command sets the right hand print margin to print position *n*. After execution of this command, any attempt to print beyond print position *n* will cause the printer to automatically generate a carriage return and a line feed before printing the remainder of the line. The value for *n* must be between 1 and 255.

PURPOSE: **Move the print head to the next horizontal tab position.**

CODE:	⟨HT⟩
(decimal ASCII)	9
(hex ASCII)	09

REMARKS: This command causes the print head to advance to the next horizontal tab position. The horizontal tab positions are set at power-on to print positions 10, 20, 30, etc. (to the maximum print position).

REFERENCE: Chapter 5

PURPOSE: **Set horizontal tab positions.**

CODE:	⟨ESC⟩	"D"	*n1 n2 n3*...	0
(decimal ASCII)	27	68	*n1 n2 n3*...	0
(hex ASCII)	1B	44	*n1 n2 n3*...	00

REMARKS: This command cancels all current horizontal tab positions and sets those defined at print positions *n1*, *n2*, *n3*, etc. The maximum number of horizontal tab positions allowed is 255. The ASCII 0 character is used as a command terminator. Each horizontal tab position must be between 1 and 255, and they must be specified in ascending order.

REFERENCE: Chapter 5

PURPOSE: **Skip *n* print positions.**

CODE:	〈ESC〉	“b”	*n*
(decimal ASCII)	27	98	*n*
(hex ASCII)	1B	62	*n*

REMARKS: This command causes the print head to advance *n* print positions to the right. It does not, however, change the current value of the horizontal tab positions. The value of *n* must be between 1 and 255.

REFERENCE: Chapter 5

PURPOSE: **Move the print head back one print position (backspace).**

CODE:	〈BS〉
(decimal ASCII)	8
(hex ASCII)	08

REMARKS: This command shifts the print head one column to the left. If the print head is at the home position, the command is ignored. This command can be used to overstrike characters.

REFERENCE: Chapter 6

Download Character Commands

PURPOSE: **Define download characters into RAM.**

CODE:	⟨ESC⟩	"*"	1	*n1 n2 m1 m2 m3 m4 m5 m6 m7 m8 m9 m10 m11*
	27	42	1	*n1 n2 m1 m2 m3 m4 m5 m6 m7 m8 m9 m10 m11*
	1B	2A	01	*n1 n2 m1 m2 m3 m4 m5 m6 m7 m8 m9 m10 m11*

REMARKS: This command is used to set up a user-defined character and store it into RAM for later use. RAM is cleared during power down. The value of *n1* is the position in RAM that this character is to occupy. It must be between 33 and 126 or between 160 and 254. That is, it must fall within the range of printable characters. The value of *n2* determines the attributes and width of the character. *m1* thru *m11* determine which dots form the character.

REFERENCE: Chapter 7

PURPOSE: **Copy standard character ROM fonts into RAM.**

CODE:	⟨ESC⟩	"*"	0
(decimal ASCII)	27	42	0
(hex ASCII)	1B	2A	00

REMARKS: This command takes all of the characters in the standard ASCII character (others don't work) and copies them into RAM. This is helpful prior to defining characters in RAM because it allows standard ROM characters to be printed on the same line as download characters.

REFERENCE: Chapter 7

PURPOSE: **Select download character set with proportional spacing.**

CODE:	〈ESC〉	"X"	1
(decimal ASCII)	27	88	1
(hex ASCII)	1B	58	01

REMARKS: This command selects the download character set using the proportional spacing defined in the character attribute data.
NOTE: Download characters *cannot* be mixed with other characters on the same line.

REFERENCE: Chapter 7

PURPOSE: **Cancel download character set with proportional spacing.**

CODE:	〈ESC〉	"X"	0
(decimal ASCII)	27	88	0
(hex ASCII)	1B	58	00

REMARKS: This command cancels the download character set and selects the standard ASCII character set.

REFERENCE: Chapter 7

PURPOSE: **Select download character set with normal spacing.**

CODE:	〈ESC〉	"$"	1
(decimal ASCII)	27	36	1
(hex ASCII)	27	24	01

REMARKS: This command causes the printer to select the download character set using normal spacing and ignoring the proportional width data.
NOTE: Download characters *cannot* be mixed with other characters on the same line.

REFERENCE: Chapter 7

PURPOSE: **Cancel download character set with normal spacing.**

CODE:	⟨ESC⟩	“$”	0
(decimal ASCII)	27	36	0
(hex ASCII)	1B	24	00

REMARKS: This command cancels the download character set and selects the standard ASCII character set.

REFERENCE: Chapter 7

Commands to Control Graphics

PURPOSE: **Print normal-density graphics.**

CODE:	⟨ESC⟩	“K”	*n1 n2 m1 m2 m3...*
(decimal ASCII)	27	75	*n1 n2 m1 m2 m3...*
(hex ASCII)	1B	4B	*n1 n2 m1 m2 m3...*

REMARKS: This command selects 60 dots-per-inch, bit-image graphics mode. The values of *n1* and *n2* represent the number of graphics characters to be printed, where the total number of characters = *n2* times 256 + *n1*. There must be the correct number of graphic characters following *n2*. The ASCII value of these characters determine which pins are fired for each character.

REFERENCE: Chapter 8

PURPOSE: **Print double-density graphics**

CODE:	⟨ESC⟩	"L"	*n1 n2 m1 m2 m3...*
(decimal ASCII)	27	76	*n1 n2 m1 m2 m3...*
(hex ASCII)	1B	4C	*n1 n2 m1 m2 m3...*

REMARKS: This command selects 120 dots-per-inch, column-scan, bit-image graphics mode. The values of *n1* and *n2* are the same as in normal density graphics. There must be the correct number of graphic characters following *n2*. The ASCII value of these characters determine which pins are fired for each character.

REFERENCE: Chapter 8

PURPOSE: **Print double-density graphics with double-speed.**

CODE:	⟨ESC⟩	"y"	*n1 n2 m1 m2 m3...*
(decimal ASCII)	27	121	*n1 n2 m1 m2 m3...*
(hex ASCII)	1B	79	*n1 n2 m1 m2 m3...*

REMARKS: This command selects 120 dots-per-inch, column-scan, bit-image graphics mode with double-speed. The values of *n1* and *n2* are the same as in normal density graphics. There must be the correct number of graphic characters following *n2*. The ASCII value of these characters determine which pins are fired for each character.

REFERENCE: Chapter 8

PURPOSE: **Print quadruple-density graphics.**

CODE:	⟨ESC⟩	"z"	*n1 n2 m1 m2 m3...*
(decimal ASCII)	27	122	*n1 n2 m1 m2 m3...*
(hex ASCII)	1B	7A	*n1 n2 m1 m2 m3...*

REMARKS: This command selects 240 dots-per-inch, column-scan, bit-image graphics mode. The values of *n1* and *n2* are the same as in normal density graphics. There must be the correct number of graphic characters following *n2*. The ASCII value of these characters determine which pins are fired for each character.

REFERENCE: Chapter 8

Macro Instruction Commands

PURPOSE: **Define macro instruction.**

CODE:	⟨ESC⟩	"+"	...	⟨RS⟩
(decimal ASCII)	27	43	...	30
(hex ASCII)	1B	2B	...	1E

REMARKS: This command cancels any existing macro instruction, and replaces it with the instruction defined. The maximum number of characters allowed in the macro instruction is 16. The ⟨RS⟩ character marks the end of the macro definition.

REFERENCE: Chapter 6

PURPOSE: **Execute macro instruction.**

CODE:	⟨ESC⟩	"!"
(decimal ASCII)	27	33
(hex ASCII)	27	21

REMARKS: This command executes a macro instruction that was previously defined.

REFERENCE: Chapter 6

Other Commands

PURPOSE: **Set the value of the eighth data bit to logical 1.**

CODE:	⟨ESC⟩	"⟩"
(decimal ASCII)	27	62
(hex ASCII)	1B	3E

REMARKS: This command forces the eighth data bit of each subsequent character sent to the printer to logical 1. This code allows users with a 7-bit interface to access those characters whose ASCII code is greater than 127. This code should not be used to transmit printer control codes.

REFERENCE: Chapter 6

PURPOSE: **Set the value of the eighth data bit to logical 0.**

CODE:	⟨ESC⟩	"="
(decimal ASCII)	27	61
(hex ASCII)	1B	3D

REMARKS: This command forces the eighth data bit of each subsequent character sent to the printer to logical 0. This code should not be used to transmit printer control codes.

REFERENCE: Chapter 6

PURPOSE: **Accept the value of the eighth data bit as is.**

CODE:	⟨ESC⟩	"#"
(decimal ASCII)	27	35
(hex ASCII)	1B	23

REMARKS: This command cancels either setting of the eighth data bit. The printer will use the value of the eighth data bit that is sent from the computer. This code allows users with only a 7-bit interface to resume normal functions after accessing those characters whose ASCII code is greater than 127.

REFERENCE: Chapter 6

PURPOSE: **Delete the last character sent.**

CODE:	⟨DEL⟩
(decimal ASCII)	127
(hex ASCII)	7F

REMARKS: This command deletes the last character received. This command is ignored if the last character received has already been printed, or if the last character received was all or part of a function code.

REFERENCE: Chapter 6

PURPOSE: **Set printer off line.**

CODE: ⟨DC3⟩
(decimal ASCII) 19
(hex ASCII) 13

REMARKS: This command causes the printer to set itself off line, disregarding all subsequent characters and function codes, with the exception of ⟨DC1⟩, which will return the printer to an on line state. This is not the same as pushing the ON-LINE button. When the ON-LINE light is out the printer will not respond to DC1.

REFERENCE: Chapter 6

PURPOSE: **Set printer on line.**

CODE: ⟨DC1⟩
(decimal ASCII) 17
(hex ASCII) 11

REMARKS: This code resets the printer to an on line state, thus allowing it receive and process all subsequent characters and function codes. This is not the same as pushing the ON-LINE button. When the ON-LINE light is out the printer will not respond to DC1.

REFERENCE: Chapter 6

PURPOSE: **Sound printer bell.**

CODE: ⟨BEL⟩
(decimal ASCII) 7
(hex ASCII) 07

REMARKS: This command causes the printer tone to sound for approximately one-fourth second.

REFERENCE: Chapter 6

PURPOSE:	**Disable the printer bell.**		
CODE:	〈ESC〉	"Y"	0
(decimal ASCII)	27	89	0
(hex ASCII)	1B	59	00
REMARKS:	This command causes the printer to ignore the 〈BEL〉 character.		
REFERENCE:	Chapter 6		

PURPOSE:	**Enable the printer bell.**		
CODE:	〈ESC〉	"Y"	1
(decimal ASCII)	27	89	1
(hex ASCII)	1B	59	01
REMARKS:	This command causes the printer to respond to the 〈BEL〉 character normally by sounding the printer bell.		
REFERENCE:	Chapter 6		

PURPOSE:	**Disable paper-out detector.**	
CODE:	〈ESC〉	"8"
(decimal ASCII)	27	56
(hex ASCII)	1B	38
REMARKS:	This command causes the printer to disregard the signal sent by the paper-out detector. The paper-out signal normally sounds the printer bell and stops printing until paper is inserted and the printer is reset. This command is useful when printing on single sheets of paper because it allows printing to the bottom of the page.	
REFERENCE:	Chapter 6	

PURPOSE:	**Enable paper-out detector.**	
CODE:	〈ESC〉	"9"
(decimal ASCII)	27	57
(hex ASCII)	1B	39
REMARKS:	This command restores the function of the paper-out detector.	
REFERENCE:	Chapter 6	

PURPOSE: **Select uni-directional printing.**

CODE:	⟨ESC⟩	“U”	1
(decimal ASCII)	27	85	1
(hex ASCII)	1B	55	01

REMARKS: This command causes all subsequent lines to be printed in uni-directional printing. Uni-directional printing is useful in printing tables or charts, since it ensures that vertical columns of characters will be in alignment.

REFERENCE: Chapter 6

PURPOSE: **Cancel uni-directional printing.**

CODE:	⟨ESC⟩	“U”	0
(decimal ASCII)	27	85	0
(hex ASCII)	1B	55	00

REMARKS: This command cancels uni-directional printing, and returns to the standard bi-directional printing, which is considerably faster.

REFERENCE: Chapter 6

PURPOSE: **Initialize printer.**

CODE:	⟨ESC⟩	“@”
(decimal ASCII)	27	64
(hex ASCII)	1B	40

REMARKS: This command reinitializes the printer. The print buffer is cleared, and the form length, character pitch, character set, line feed pitch, and international character set are all reset to the values defined by their respective DIP switches.

The main difference between the ⟨ESC⟩ “@” command and turning the printer off and back on is that download character RAM is preserved with this command.

REFERENCE: Chapter 6

Appendix L

Command Summary in Numeric Order

Control code	Function
CHR$(0)	End tab settings
CHR$(7)	Sound bell
CHR$(8)	Backspace
CHR$(9)	Horizontal tab
CHR$(10)	Line feed
CHR$(11)	Vertical tab
CHR$(12)	Form feed
CHR$(13)	Carriage return
CHR$(14)	Expanded print
CHR$(15)	Condensed print
CHR$(17)	On line
CHR$(18)	Pica type
CHR$(19)	Off line
CHR$(20)	Cancel enlarged print
CHR$(27)	Escape **(indicated as 〈ESC〉 below)**
CHR$(30)	End macro instruction definition
CHR$(127)	Delete last character
〈ESC〉 CHR$(14)	Expanded print
〈ESC〉 CHR$(15)	Condensed print
〈ESC〉 “!”	Use macro
〈ESC〉 “#”	Accept eighth bit as is
〈ESC〉 “$” CHR$(0)	Cancel normal download characters
〈ESC〉 “$” CHR$(1)	Use normal download characters
〈ESC〉 “*” CHR$(0)	Copy ROM characters to download RAM
〈ESC〉 “*” CHR$(1) *n1 n2 m1 m2 . . . m11*	Define download character
〈ESC〉 “ + ” . . . CHR$(30)	Define macro
〈ESC〉 “ – ” CHR$(0)	Stop underlining

Control code	Function
⟨ESC⟩ “ – ” CHR$(1)	Start underlining
⟨ESC⟩ “0”	1/8 inch line feed
⟨ESC⟩ “1”	7/72 inch line feed
⟨ESC⟩ “2”	1/6 inch line feed
⟨ESC⟩ “3” *n*	*n*/144 inch line feed
⟨ESC⟩ “4”	Italic print
⟨ESC⟩ “5”	Cancel italic print
⟨ESC⟩ “7” *n*	Select international character set
⟨ESC⟩ “8”	Ignore paper-out signal
⟨ESC⟩ “9”	Enable paper-out signal
⟨ESC⟩ “ = ”	Set eighth bit to 0
⟨ESC⟩ “⟩”	Set eighth bit to 1
⟨ESC⟩ “@”	Reset the printer
⟨ESC⟩ “A” *n*	*n*/72 inch line feed
⟨ESC⟩ “B” CHR$(1)	Pica print
⟨ESC⟩ “B” CHR$(2)	Elite print
⟨ESC⟩ “B” CHR$(3)	Condensed print
⟨ESC⟩ “C” *n*	Set page length to *n* lines
⟨ESC⟩ “C” CHR$(0) *n*	Set page length to *n* inches
⟨ESC⟩ “D” . . . CHR$(0)	Set horizontal tabs
⟨ESC⟩ “E”	Emphasized print
⟨ESC⟩ “F”	Cancel emphasized print
⟨ESC⟩ “G”	Double-strike print
⟨ESC⟩ “H”	Cancel double-strike print
⟨ESC⟩ “J” *n*	Single line feed of *n*/144 inches
⟨ESC⟩ “K” *n1 n2*	Single density graphics
⟨ESC⟩ “L” *n1 n2*	Double density graphics
⟨ESC⟩ “M” *n*	Set left margin at column *n*
⟨ESC⟩ “N” *n*	Set bottom margin at *n* lines
⟨ESC⟩ “O”	Cancel top and bottom margins
⟨ESC⟩ “P” . . . CHR$(0)	Set vertical tabs
⟨ESC⟩ “Q” *n*	Set right margin at column *n*
⟨ESC⟩ “R” *n*	Set top margin at line *n*
⟨ESC⟩ “S” CHR$(0)	Superscript on
⟨ESC⟩ “S” CHR$(1)	Subscript on
⟨ESC⟩ “T”	Cancel super and subscripts
⟨ESC⟩ “U” CHR$(0)	Bidirectional print
⟨ESC⟩ “U” CHR$(1)	Unidirectional print
⟨ESC⟩ “W” CHR$(0)	Cancel enlarged print
⟨ESC⟩ “W” CHR$(1)	Enlarged print
⟨ESC⟩ “X” CHR$(0)	Cancel proportional download characters
⟨ESC⟩ “X” CHR$(1)	Use proportional download characters

Control code	**Function**
⟨ESC⟩ "Y" CHR$(0)	Disable bell
⟨ESC⟩ "Y" CHR$(1)	Enable bell
⟨ESC⟩ "a" *n*	Advance *n* line feeds
⟨ESC⟩ "b" *n*	Tab over *n* columns
⟨ESC⟩ "y" *n1 n2*	Double speed, double density graphics
⟨ESC⟩ "z" *n1 n2*	Quadruple density graphics

Appendix M

ASCII Code Conversion Chart

Standard ASCII Codes

Decimal	Hexadecimal	Binary	Control character	Character
0	00	0000 0000	Ctrl-@	NUL
1	01	0000 0001	Ctrl-A	
2	02	0000 0010	Ctrl-B	
3	03	0000 0011	Ctrl-C	
4	04	0000 0100	Ctrl-D	
5	05	0000 0101	Ctrl-E	
6	06	0000 0110	Ctrl-F	
7	07	0000 0111	Ctrl-G	BEL
8	08	0000 1000	Ctrl-H	BS
9	09	0000 1001	Ctrl-I	HT
10	0A	0000 1010	Ctrl-J	LF
11	0B	0000 1011	Ctrl-K	VT
12	0C	0000 1100	Ctrl-L	FF
13	0D	0000 1101	Ctrl-M	CR
14	0E	0000 1110	Ctrl-N	SO
15	0F	0000 1111	Ctrl-O	SI
16	10	0001 0000	Ctrl-P	
17	11	0001 0001	Ctrl-Q	DC1
18	12	0001 0010	Ctrl-R	DC2
19	13	0001 0011	Ctrl-S	DC3
20	14	0001 0100	Ctrl-T	DC4
21	15	0001 0101	Ctrl-U	
22	16	0001 0110	Ctrl-V	
23	17	0001 0111	Ctrl-W	
24	18	0001 1000	Ctrl-X	
25	19	0001 1001	Ctrl-Y	
26	1A	0001 1010	Ctrl-Z	
27	1B	0001 1011		ESC
28	1C	0001 1100		
29	1D	0001 1101		
30	1E	0001 1110		RS
31	1F	0001 1111		

	Standard ASCII Codes		
Decimal	Hexadecimal	Binary	Character
32	20	0010 0000	SP
33	21	0010 0001	!
34	22	0010 0010	"
35	23	0010 0011	#
36	24	0010 0100	$
37	25	0010 0101	%
38	26	0010 0110	&
39	27	0010 0111	'
40	28	0010 1000	(
41	29	0010 1001	)
42	2A	0010 1010	*
43	2B	0010 1011	+
44	2C	0010 1100	,
45	2D	0010 1101	-
46	2E	0010 1110	.
47	2F	0010 1111	/
48	30	0011 0000	0
49	31	0011 0001	1
50	32	0011 0010	2
51	33	0011 0011	3
52	34	0011 0100	4
53	35	0011 0101	5
54	36	0011 0110	6
55	37	0011 0111	7
56	38	0011 1000	8
57	39	0011 1001	9
58	3A	0011 1010	:
59	3B	0011 1011	;
60	3C	0011 1100	<
61	3D	0011 1101	=
62	3E	0011 1110	>
63	3F	0011 1111	?
64	40	0100 0000	@
65	41	0100 0001	A
66	42	0100 0010	B
67	43	0100 0011	C
68	44	0100 0100	D
69	45	0100 0101	E
70	46	0100 0110	F
71	47	0100 0111	G
72	48	0100 1000	H
73	49	0100 1001	I

	Standard ASCII Codes		
Decimal	**Hexadecimal**	**Binary**	**Character**
74	4A	0100 1010	J
75	4B	0100 1011	K
76	4C	0100 1100	L
77	4D	0100 1101	M
78	4E	0100 1110	N
79	4F	0100 1111	O
80	50	0101 0000	P
81	51	0101 0001	Q
82	52	0101 0010	R
83	53	0101 0011	S
84	54	0101 0100	T
85	55	0101 0101	U
86	56	0101 0110	V
87	57	0101 0111	W
88	58	0101 1000	X
89	59	0101 1001	Y
90	5A	0101 1010	Z
91	5B	0101 1011	[
92	5C	0101 1100	\
93	5D	0101 1101	]
94	5E	0101 1110	^
95	5F	0101 1111	_
96	60	0110 0000	`
97	61	0110 0001	a
98	62	0110 0010	b
99	63	0110 0011	c
100	64	0110 0100	d
101	65	0110 0101	e
102	66	0110 0110	f
103	67	0110 0111	g
104	68	0110 1000	h
105	69	0110 1001	i
106	6A	0110 1010	j
107	6B	0110 1011	k
108	6C	0110 1100	l
109	6D	0110 1101	m
110	6E	0110 1110	n
111	6F	0110 1111	o
112	70	0111 0000	p
113	71	0111 0001	q
114	72	0111 0010	r
115	73	0111 0011	s

Standard ASCII Codes

Decimal	Hexadecimal	Binary	Character
116	74	0111 0100	t
117	75	0111 0101	u
118	76	0111 0110	v
119	77	0111 0111	w
120	78	0111 1000	x
121	79	0111 1001	y
122	7A	0111 1010	z
123	7B	0111 1011	{
124	7C	0111 1100	|
125	7D	0111 1101	}
126	7E	0111 1110	~
127	7F	0111 1111	DEL
128	80	1000 0000	
129	81	1000 0001	
130	82	1000 0010	
131	83	1000 0011	
132	84	1000 0100	
133	85	1000 0101	
134	86	1000 0110	
135	87	1000 0111	BEL
136	88	1000 1000	BS
137	89	1000 1001	HT
138	8A	1000 1010	LF
139	8B	1000 1011	VT
140	8C	1000 1100	FF
141	8D	1000 1101	CR
142	8E	1000 1110	SO
143	8F	1000 1111	SI
144	90	1001 0000	
145	91	1001 0001	DC1
146	92	1001 0010	DC2
147	93	1001 0011	DC3
148	94	1001 0100	DC4
149	95	1001 0101	
150	96	1001 0110	
151	97	1001 0111	
152	98	1001 1000	
153	99	1001 1001	
154	9A	1001 1010	
155	9B	1001 1011	ESC
156	9C	1001 1100	
157	9D	1001 1101	

Decimal	Standard ASCII Codes Hexadecimal	Binary	Character
158	9E	1001 1110	RS
159	9F	1001 1111	
160	A0	1010 0000	╯
161	A1	1010 0001	╮
162	A2	1010 0010	╰
163	A3	1010 0011	╭
164	A4	1010 0100	↑
165	A5	1010 0101	↓
166	A6	1010 0110	←
167	A7	1010 0111	→
168	A8	1010 1000	○
169	A9	1010 1001	▴
170	AA	1010 1010	▾
171	AB	1010 1011	▸
172	AC	1010 1100	◂
173	AD	1010 1101	◇
174	AE	1010 1110	◆
175	AF	1010 1111	□
176	B0	1011 0000	Tı
177	B1	1011 0001	Ȧ
178	B2	1011 0010	⊕
179	B3	1011 0011	θ
180	B4	1011 0100	⊏
181	B5	1011 0101	╞
182	B6	1011 0110	Ω
183	B7	1011 0111	℧
184	B8	1011 1000	Σ
185	B9	1011 1001	σ
186	BA	1011 1010	(x)
187	BB	1011 1011	Ӿ
188	BC	1011 1100	±
189	BD	1011 1101	⊐
190	BE	1011 1110	×
191	BF	1011 1111	÷
192	C0	1100 0000	Ā
193	C1	1100 0001	à
194	C2	1100 0010	ς
195	C3	1100 0011	丈
196	C4	1100 0100	ā
197	C5	1100 0101	µ
198	C6	1100 0110	▫
199	C7	1100 0111	,

	Standard ASCII Codes		
Decimal	Hexadecimal	Binary	Character
200	C8	1100 1000	†
201	C9	1100 1001	§
202	CA	1100 1010	Ē
203	CB	1100 1011	⊗
204	CC	1100 1100	Կ
205	CD	1100 1101	X̄
206	CE	1100 1110	k̄
207	CF	1100 1111	‼
208	D0	1101 0000	¥
209	D1	1101 0001	Ȧ
210	D2	1101 0010	Ö
211	D3	1101 0011	Ü
212	D4	1101 0100	¢
213	D5	1101 0101	Ñ
214	D6	1101 0110	ä
215	D7	1101 0111	ö
216	D8	1101 1000	ü
217	D9	1101 1001	₱
218	DA	1101 1010	ē
219	DB	1101 1011	ė
220	DC	1101 1100	à
221	DD	1101 1101	è
222	DE	1101 1110	ñ
223	DF	1101 1111	ƒ
224	E0	1110 0000	SP
225	E1	1110 0001	▘
226	E2	1110 0010	▖
227	E3	1110 0011	▝
228	E4	1110 0100	▗
229	E5	1110 0101	▚
230	E6	1110 0110	▞
231	E7	1110 0111	▀
232	E8	1110 1000	▄
233	E9	1110 1001	▌
234	EA	1110 1010	▐
235	EB	1110 1011	▛
236	EC	1110 1100	▜
237	ED	1110 1101	▙
238	EE	1110 1110	▟
239	EF	1110 1111	█
240	F0	1111 0000	┌
241	F1	1111 0001	─

Decimal	Standard ASCII Codes Hexadecimal	Binary	Character
242	F2	1111 0010	┐
243	F3	1111 0011	┬
244	F4	1111 0100	├
245	F5	1111 0101	│
246	F6	1111 0110	└
247	F7	1111 0111	┘
248	F8	1111 1000	┴
249	F9	1111 1001	┤
250	FA	1111 1010	┼
251	FB	1111 1011	◤
252	FC	1111 1100	◢
253	FD	1111 1101	◥
254	FE	1111 1110	◣
255	FF	1111 1111	

Appendix N

Technical Specifications

Printing

Printing method	Serial impact dot matrix
Printing speed	160 characters per second in 10 and 12 CPI 137 characters per second in 17 CPI
Paper feed	100 ms/line (at 1/6 inch/line) Sprocket or friction feed
Printing direction	Bidirectional, logic seeking Unidirectional in bit image modes
Character set	96 standard ASCII characters 88 international characters 64 special symbols 32 block graphics characters 189 user defined characters
Character size	2.4 mm x 2.0 mm standard 10 CPI characters
Character matrix	Standard characters: 9 dot x 9 dot Block graphics: 6 dot x 6 dot User defined: 7 dot x 4 to 11 dot Bit image modes: 7 or 8 dot x 60 dots/in. 7 or 8 dot x 120 dots/in. 7 or 8 dot x 240 dots/in.
Line spacing	1/6 or 1/8 inch standard n/72 inch or n/144 inch programmable

Column width

	Delta-10	Delta-15
10 CPI	80	136
12 CPI	96	163
17 CPI	136	233

Paper		Delta-10	Delta-15
Paper type	Single sheets	8-10 in. wide	8-15 in. wide
	Roll paper	8.5 in. wide	8.5 in. wide
	Sprocket-feed paper	3-10 in. wide	5-15.5 in. wide
Thickness	One-part forms	0.07-0.10 mm	0.07-0.10 mm
	Max. 3-part forms	0.28 mm max.	0.28 mm max.
Roll diameter		5 in. max.	5 in. max.

Printer		Delta-10	Delta-15
Dimensions	Height	148 mm (5.8 in.)	148 mm (5.8 in.)
	Width	392 mm (15.2 in.)	542 mm (21.3 in.)
	Depth	315 mm (12.4 in.)	315 mm (12.4 in.)
Weight		7.8 kg (17.2 lb.)	10.2 kg (22.5 lb.)
Power	120 VAC ± 10% 60Hz		
Ribbon	Star Micronics P/M SF-02B, or Underwood ½ in. x 2 in. spools		

Parallel interface

Interface	Centronics-compatible, 7 or 8 bit
Synchronization	By externally supplied strobe pulses
Handshaking	By ACK or BUSY signals
Logic level	TTL
Connector	57-30360 Amphenol

Serial interface

Interface	Asynchronous RS-232C
Bit rate	110, 300, 600, 1200, 2400, 4800, 9600 baud
Word length	1 start bit 7 or 8 data bits Odd, even or no parity 1 or 2 stop bits
Handshaking	Serial busy, 1 byte mode Serial busy, 1 block mode ACK mode XON/XOFF mode

Appendix O

The Parallel Interface

Delta has both a parallel interface and a serial interface to communicate with the computer that it is connected to. The operating specifications of the parallel interface are as follows:

Data transfer rate:	1,000 to 6,000 characters per second
Synchronization:	Via externally supplied $\overline{\text{STROBE}}$ pulses
Handshaking:	$\overline{\text{ACK}}$ and BUSY signals
Logic level:	Compatible with TTL level

Delta's parallel interface connects to the computer by a 36 pin connector on the back of the printer. This connector mates with an Amphenol 57-30360 connector. The functions of the various pins are summarized in Table O-1.

Functions of the Connector Signals

Communications between the computer and the Delta use many of the pins of the connector. To understand how the system of communications works we need to look at the functions of the various signals carried by the pins of the interface connector.

Pin 1 carries the $\overline{\text{STROBE}}$ pulse signal from the computer to the printer. This signal is normally held high by the computer. When the computer has data ready for the printer it sets this signal to a low value for at least 0.5 microseconds. When the printer sees this pulse on the strobe pin, it reads the data that the computer supplies on pins 2 through 9. Each of these lines carries one bit of information. A logical "1" is represented by a high signal level, and a logical "0" is represented by a low signal level. The computer must maintain these signals for a period beginning at least 0.5 microseconds before the strobe pulse starts and continuing for at least 0.5 microseconds after the strobe pulse ends.

When the Delta has successfully received the byte of data from the computer it sets pin 10 low for approximately 9 microseconds. This signal acknowledges the receipt of the data and so is called the $\overline{\text{ACK}}$ signal.

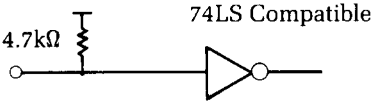
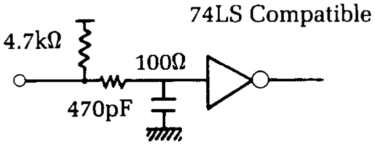
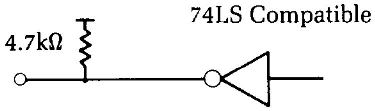
Signal Name	Circuit Example
DATA 1 - DATA 8 (To Printer)	
STROBE (To Printer)	
BUSY, ACK (From Printer)	

Figure O-1. *Delta interface timing diagram.*

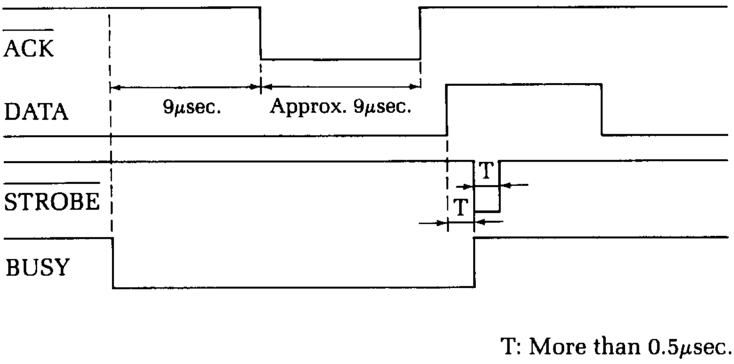


Figure O-2. *Typical interface circuit.*

Table O-1

Signal Pin No.	Signal Name	Direction	Function
1	$\overline{\text{STROBE}}$	IN	Signals when data is ready to be read. Signal goes from HIGH to LOW (for at least 0.5 microseconds) when data is available.
2	DATA1	IN	These signals provide the information of the first to eighth bits of parallel data. Each signal is at a HIGH level for a logical 1 and at a LOW level for a logical 0.
3	DATA2	IN	
4	DATA3	IN	
5	DATA4	IN	
6	DATA5	IN	
7	DATA6	IN	
8	DATA7	IN	
9	DATA8	IN	
10	$\overline{\text{ACK}}$	OUT	A 9 microsecond LOW pulse acknowledges receipt of data.
11	BUSY	OUT	When this signal goes LOW the printer is ready to accept data.
12	PAPER OUT	OUT	This signal is normally LOW. It will go HIGH if Delta runs out of paper. This signal can be held LOW permanently by turning DIP switch 2-1 off.
13	SELECTED	OUT	This signal is HIGH when the printer is on-line.
14-15	N/C		Unused.
16	SIGNAL GND		Signal ground.
17	CHASSIS GND		Printer's chassis ground, isolated from logic ground.
18	+5VDC	OUT	External supply of +5VDC.
19-30	GND		Twisted pair return signal ground level.
31	$\overline{\text{RESET}}$	IN	When this signal goes LOW the printer is reset to its power-on condition.
32	$\overline{\text{ERROR}}$	OUT	This signal is normally HIGH. This signal goes LOW to signal that the printer cannot print due to an error condition.
33	EXT GND		External ground.
34-36	N/C		Unused.

Pin 11 reports when the Delta is not able to receive data. The signal is called BUSY. When this signal is high, Delta cannot receive data. This signal will be high during data transfer, when the printer is off-line and when an error condition exists.

Appendix P

Serial Interface Specifications

Delta provides a very flexible RS232C serial interface. It can communicate at rates from 110 to 9600 baud and supports four different kinds of *handshaking*. The operating specifications of the interface are as follows:

Data transfer rate:	110-9600 baud
Word length:	1 start bit 7 or 8 data bits Odd, even or no parity 1 or 2 stop bits
Signal levels:	Mark or OFF, -3 to -15 volts Space or ON, +3 to +15 volts
Handshaking:	Serial busy, 1 byte mode Serial busy, 1 block mode ACK mode XON/XOFF mode

Delta has a DB-25 female connector on the back to connect to a computer. The functions of the pins are summarized in Table P-1

Configuring the Serial Interface

DIP switch 3 controls the configuration of the serial interface. Figure P-1 shows the location of DIP switch 3. You must remove Delta's upper case to reach this switch. See Chapter 10 for instructions on how to do this. Table P-2 describes the functions of the individual switches in DIP switch 3.

Delta's Serial Protocols

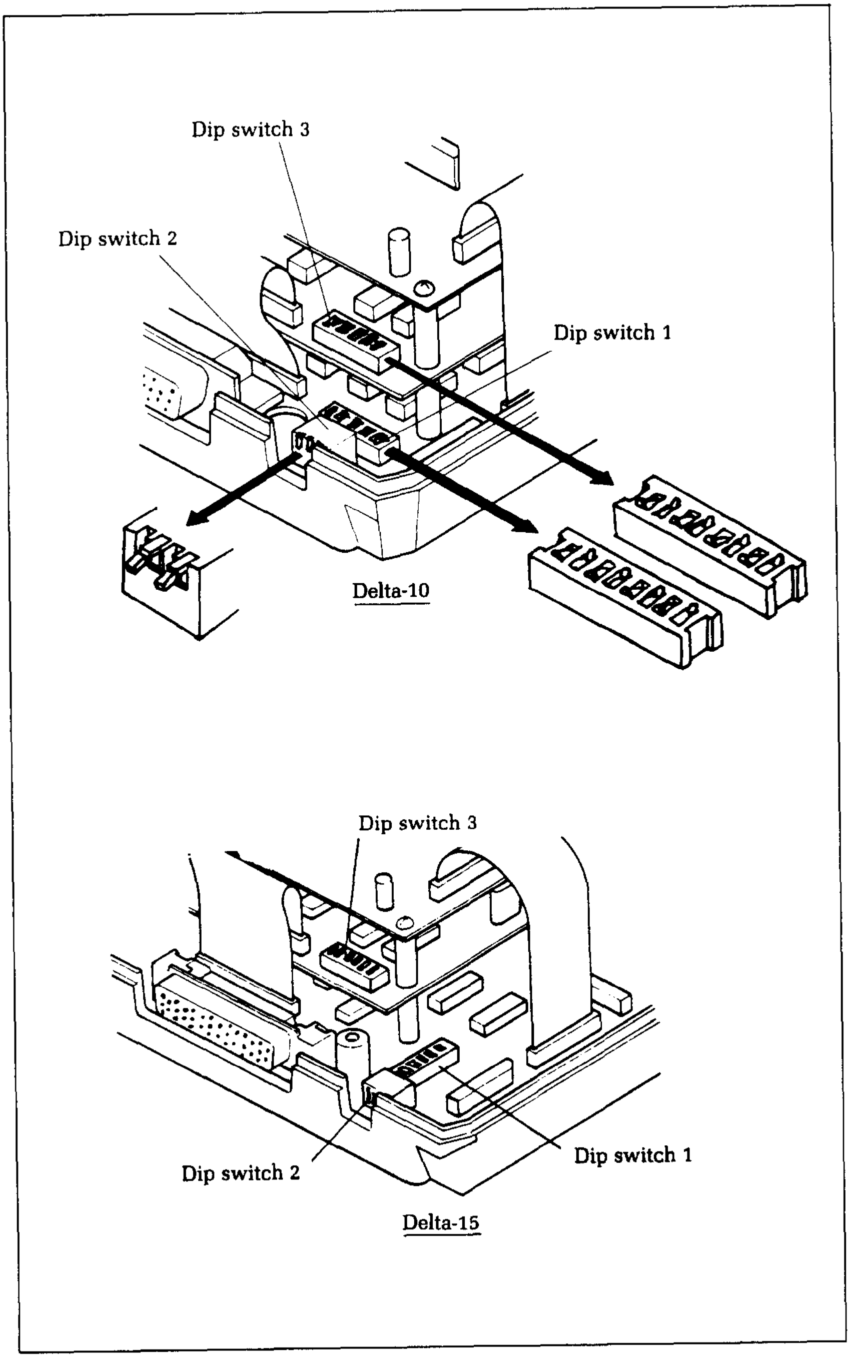
Delta has four serial protocols selected by DIP switches 3-3 and 3-4. Figure P-2 shows a typical byte of serial data and Figure P-3 shows timing charts for the 4 protocols.

Table P-1
Serial Interface Pin Functions

Signal Pin No.	Signal Name	Direction	Function
1	GND	—	Printer's chassis ground.
2	TXD	OUT	This pin carries data from the printer.
3	RXD	IN	This pin carries data to the printer.
4	RTS	OUT	This is ON when the printer is ready to receive data.
5	CTS	IN	This pin is ON when the computer is ready to send data.
6	DSR	IN	This pin is ON when the computer is ready to send data. Delta does not check this pin.
7	GND	—	Signal ground
8	DCD	IN	This pin is ON when the computer is ready to send data.
9-10	N/C		Unused
11	RCH	OUT	This is the signal line for the serial busy protocols. This pin goes OFF when Delta's buffer fills, and ON when Delta is ready to receive data. In the busy protocols this line carries the same signal as pin 20.
12	N/C		Unused
13	GND	—	Signal ground
14-19	N/C		Unused
20	DTR	OUT	Delta turns this pin ON when it is ready to receive data.
21-25	N/C		Unused.

Table P-2
DIP Switch 3

Switch	ON	OFF
3-1	7 data bits	8 data bits
3-2	Parity checked	No parity
3-3	Handshaking protocols—see below	
3-4		
3-5	Odd parity	Even parity
3-6	Data transfer rate—see below	
3-7		
3-8		



Delta-10

Delta-15

Figure P-1. DIP switch 3 is located inside the case.

Table P-3
Handshaking protocols

Protocol	Switch 3-3	Switch 3-4
Serial busy, 1 byte mode	OFF	OFF
Serial busy, 1 block mode	ON	OFF
ACK mode	OFF	ON
XON/XOFF	ON	ON

Table P-4
Data transfer rates

Baud rate	Switch 3-6	Switch 3-7	Switch 3-8
110	OFF	OFF	OFF
110	OFF	OFF	ON
300	OFF	ON	OFF
600	OFF	ON	ON
1200	ON	OFF	OFF
2400	ON	OFF	ON
4800	ON	ON	OFF
9600	ON	ON	ON

Serial busy protocols

In the serial busy protocols, Delta uses DTR (pin 20) and RCH (pin 11) to signal to the computer when it is able to accept data. These two pins go ON when Delta is ready to accept data. In the 1 byte mode they go OFF after each character is received. In the 1 block mode they only go OFF when Delta's buffer approaches capacity. In both cases they will stay OFF if the buffer is too full to accept more data.

XON/XOFF protocol

The XON/XOFF protocol uses the ASCII characters ⟨DC1⟩ and ⟨DC3⟩ (sometimes called XON and XOFF, respectively) to communicate with the computer. When Delta's buffer approaches capacity Delta will send a DC3 (ASCII 19) on TXD (pin 2) to tell the computer that it must stop sending data. When Delta is able to receive more data it sends a DC1 (ASCII 17) on TXD. The computer can then send more data until Delta sends another DC3.

ACK protocol

In the ACK protocol, Delta sends an ACK (ASCII 6) on TXD (pin 2) each time that it is prepared to receive a byte of data.

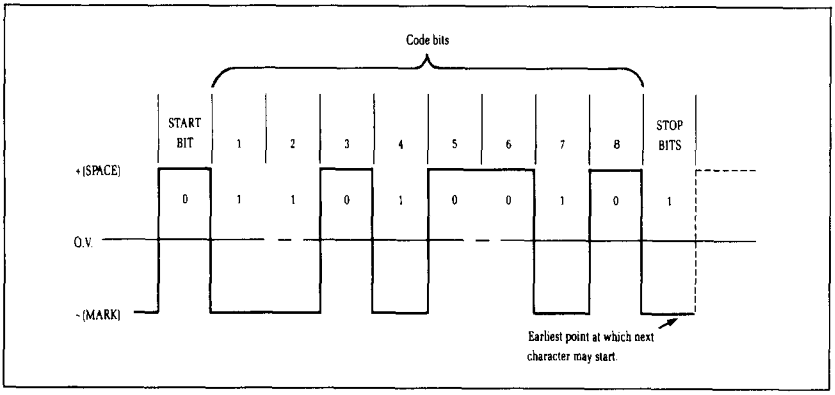


Figure P-2. *Typical data byte on the serial interface.*

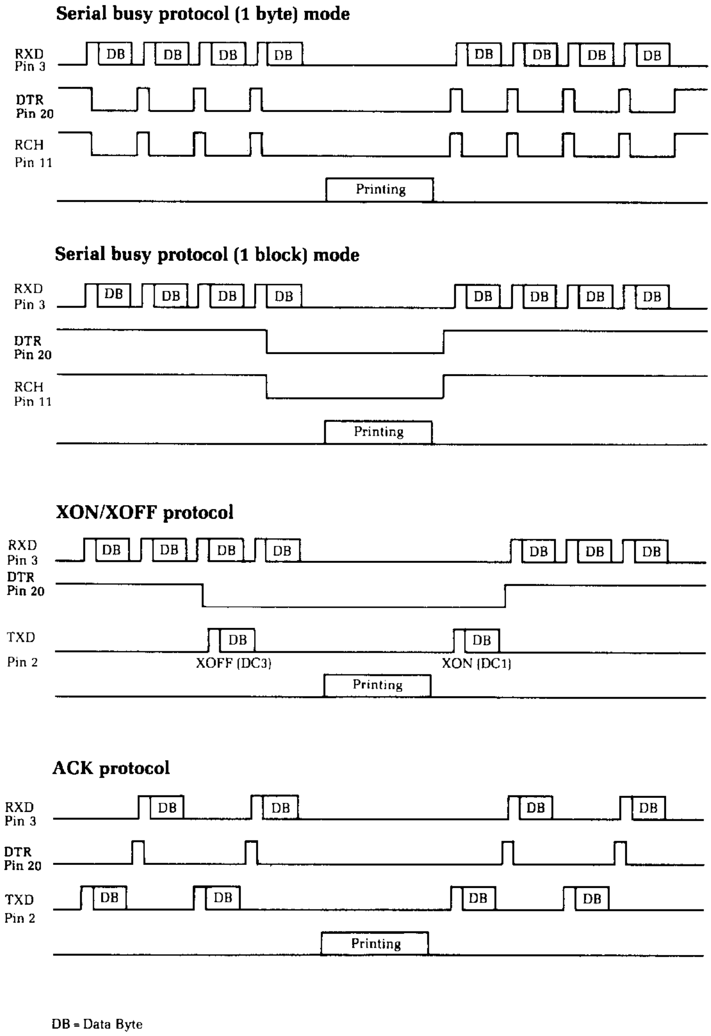


Figure P-3. *Serial protocol timing charts.*

Index

ACK protocol 301
Adjusting the print head gap 13
Advance paper 265
American Standard Code for
  Information Interchange 19
Apple II computer 23, 56, 65, 143
Applesoft BASIC 143
Array, in memory 107
ASCII 19
ASCII code conversion chart 283
ASCII codes 80, 283
Atari BASIC 193
Atari computer 193
Attribute byte 81
Auto line feed 228
Backspace 65, 268
BASIC 19, 23, 29
  Applesoft 143
  Atari 193
  Commodore 211
  Microsoft 178
  problems with 117
  TRS-80 162
  using this book with 23
⟨BEL⟩ 62, 275
Bell 61, 275
Bell code 61
Bidirectional printing 63, 277
Binary 283
Block graphic characters 66
Bold print 32
Bottom feed (Delta-15) 11
Bottom margin 50
Bridge hand program 85
  Apple II 148
  Atari 197
  Commodore 215
  Osborne 1 181
  TRS-80 165
⟨BS⟩ 268
Business graphics 111
Button, F. F. 2
Button, form feed 2
Button, L. F. 3
Button, on line 2
Byte, attribute 81
C-64 computer 24, 211
Calendar 119
Carriage return 44, 266
Character design program 95
Character graphics 66
Character matrix 291
Character sets 291
Character size 291
Character style charts 239
Character values 80
Character width 30
Character widths 88
Characters, connecting 91
Characters, defining 76
Characters, download 73
Characters, international 68
Characters, proportional 88, 270
Characters, special 66
Characters, user-defined 73
Chart program 37
  Apple II 146
  Atari 195
  Commodore 212
  Osborne 1 179
  TRS-80 163
Chart, ASCII code conversion 283
Chart, print modes 40
Charts, character style 239
CHR$ function 25
CHR$( ) 19
CHR$(10) 44, 260
CHR$(11) 59, 264
CHR$(12) 48, 262
CHR$(127) 65, 274
CHR$(13) 44, 266
CHR$(14) 31, 256
CHR$(15) 31, 255
CHR$(17) 63, 275
CHR$(18) 31, 254
CHR$(19) 63, 275
CHR$(20) 256
CHR$(27) 26
CHR$(30) 69, 273
CHR$(7) 62, 275
CHR$(8) 65, 268
CHR$(9) 55, 267
Cleaning 123
Codes 19
Column width 291
Columns of dots 78
  for graphics 101
Combining text and graphics 103

Command quick reference B/C
Command summary in numeric order 279
Commands, download characters 269
  form feed 262
  graphics 271
  horizontal position 266
  other 273
  print style 251
  user-defined characters 269
  vertical positioning 260
Commercial software 18
Commodore BASIC 211
Commodore C-64 24
Commodore computers 211
Communication, establishing 24
Compaq computer 139
Computer art 104
Computer memory 107
Condensed pitch 30, 254
Condensed print 228
Connecting characters 91
Connector, parallel interface 293
Connector, serial interface 297
Connectors, interface 4
Control character 283
Control code reference 251
Control codes 20, 25
Control key 20
Control panel 2
Cover, printer 2, 132, 138
CP/M 23
CP/M computer 177
⟨CR⟩ 266
CTRL 20
Darker print 32
Data transfer rate 293, 297
⟨DC1⟩ 275
⟨DC2⟩ 254
⟨DC3⟩ 275
⟨DC4⟩ 256
Decimal 283
Default vertical tabs 59
Defining characters 76
⟨DEL⟩ 274
Delete 65, 274
Delta Plot program 107
  Apple II 155
  Atari 203
  Commodore 221
  Osborne 1 187
  TRS-80 172
Descender 77, 81
Descender data 81
Design, printing a 104
Detector, paper-out 63, 276
Digits 91
Dimensions, external 292
DIP switch 1-5 44
DIP switch 2-3 67
DIP switch 2-4 44
DIP switch 3 298
DIP switch 3, IBM-PC 141
DIP switches 4, 227
Dot graphics 99
Dot graphics, calculating 107
Dot matrix printing 74
Dots, in columns 78
Dots, overlapping 77
Double density graphics 115, 272
Double density/double speed
  graphics 115, 272
Double space 44
Double-strike printing 32, 257
Download characters 62, 73, 269
  selecting 90
  with print modes 95
  proportional 88
Download utility program 95
  Apple II 152
  Atari 200
  Commodore 218
  Osborne 1 185
  TRS-80 169
Dust 124
EasyWriter 22
Eight bit interface 228
Eighth data bit 273
Elite pitch 30, 254
Emphasized print 32, 228, 257
⟨ESC⟩ 26
⟨ESC⟩ "!" 69, 273
⟨ESC⟩ "#" 66, 274
⟨ESC⟩ "$" 83, 270
⟨ESC⟩ "*" 80, 83, 269
⟨ESC⟩ "+" 69, 273
⟨ESC⟩ "-" 35, 258
⟨ESC⟩ "0" 46, 260
⟨ESC⟩ "1" 46, 261
⟨ESC⟩ "2" 45, 261
⟨ESC⟩ "3" 45, 262
⟨ESC⟩ "4" 34, 252
⟨ESC⟩ "5" 34, 252
⟨ESC⟩ "7" 68, 253
⟨ESC⟩ "8" 63, 276
⟨ESC⟩ "9" 63, 276
⟨ESC⟩ "=" 66, 274
⟨ESC⟩ ">" 66, 273
⟨ESC⟩ "@" 62, 277
⟨ESC⟩ "A" 45, 261
⟨ESC⟩ "a" 47, 60, 265
⟨ESC⟩ "B" 30, 253
⟨ESC⟩ "b" 57, 268
⟨ESC⟩ "C" 49, 263
⟨ESC⟩ "D" 56, 267
⟨ESC⟩ "E" 33, 257

⟨ESC⟩ "F" *33, 258*
⟨ESC⟩ "G" *33, 257*
⟨ESC⟩ "H" *33, 257*
⟨ESC⟩ "J" *46, 262*
⟨ESC⟩ "K" *100, 271*
⟨ESC⟩ "L" *115, 272*
⟨ESC⟩ "M" *57, 266*
⟨ESC⟩ "N" *50, 264*
⟨ESC⟩ "O" *50, 264*
⟨ESC⟩ "P" *59, 265*
⟨ESC⟩ "Q" *57, 267*
⟨ESC⟩ "R" *50, 263*
⟨ESC⟩ "S" *36, 259*
⟨ESC⟩ "T" *36, 259*
⟨ESC⟩ "U" *64, 277*
⟨ESC⟩ "W" *32, 255, 256*
⟨ESC⟩ "X" *90, 270*
⟨ESC⟩ "Y" *62*
⟨ESC⟩ "y" *115, 272*
⟨ESC⟩ "z" *115, 272*
Escape code *26*
Expanded print *31, 255*
F. F. Button *2*
⟨FF⟩ *262*
Font style controls *252*
Form feed *48, 262*
Form length *49*
Forms control *43*
Friction feed *5*
Function code reference *251*
Fuse replacement *125*
Fuse, primary *126*
- secondary *127*
- spare *132*

Gap, print head *13*
grafstar *143*
Graphics characters *66*
Graphics combined with text *103*
Graphics commands *271*
Graphics, business *111*
- dot *99*
- double density *115, 272*
- double density/double speed *115, 272*
- normal density *115, 271*
- plotting *107*
- quadruple density *115, 272*

Grid, character design *75*
Guide, paper *2, 132, 134*
Handshaking *293, 297*
Hex *19*
Hexadecimal *19, 283*
High resolution graphics *115*
Holder, roll paper *132*
Horizontal position commands *266*
Horizontal tab *55, 267*
⟨HT⟩ *55, 267*
IBM-PC *23, 139*
IBM-PC BASIC *141*
IBM-PC MODE command *140*
IBM-PC, listing programs *142*
Impact printer *74*
Initialize printer *277*
Initializing *62*
Ink ribbon *132*
Installation programs *18*
Installing the ink ribbon *136*
Interface connectors *4*
Interface, parallel *293*
- selection *228*
- serial *297*
- seven bit *65, 67*

International characters *68, 228*
- select *253*
- selection *231*

Italic printing *34, 228, 252*
Kaypro computer *177*
Knob, platen *2, 132, 133*
L. F. button *3*
Lamp, on line *2*
Lamp, paper out *2*
Lamp, power *2*
Lamp, ready *2*
Left margin *57*
Length of page *43*
Length, form *49*
Length, page *49*
Levers, snap *134*
LF, auto *228*
⟨LF⟩ *260*
Line feed *44, 260, 262*
Line feed commands *260*
Line spacing *43, 44, 228, 260*
Listing programs *23*
- Apple II *145*
- Atari *195*
- Commodore *212*
- IBM-PC *142*

LLIST *23*
LLIST, IBM-PC *142*
Loading paper *4*
- bottom feed (Delta-15) *11*
- roll *7*
- single sheet *5*
- sprocket feed *9*

Location for Delta *131*
Logic level *293*
Logo program *105*
Logo, printing *104*
LPRINT *23*
Macro *62, 68*
Macro instruction *273*
- defining *69*
- using *70*

Macro program *70*
    Apple II *148*
    Atari *196*
    Commodore *214*
    Osborne 1 *181*
    TRS-80 *165*
Maintenance *123*
Margin, left and right *57, 266*
    top and bottom *50, 263*
    resetting *58*
Mask *109*
Matrix, character *291*
Matrix, dot *74*
MBASIC *178*
Microsoft BASIC *23*
Mixing print modes *36*
MODE command, IBM-PC *140*
Modes, combining print *36*
Non-printing codes *22*
Normal density graphics *115, 271*
Numeral program *92*
    Apple II *150*
    Atari *199*
    Commodore *216*
    Osborne 1 *183*
    TRS-80 *167*
Off line *63, 275*
On line *275*
On line button *2*
On line lamp *2*
On-line *63*
Osborne computer *177*
Overlapping dots *77*
Page length *43, 49, 228, 263*
Paper advance *265*
Paper guide *2, 132, 134*
Paper lint *124*
Paper loading *4*
Paper out detector *63*
Paper out lamp *2*
Paper selection *4*
Paper separator *2, 132, 134*
Paper specifications *292*
    thickness *5*
    types *5*
    width *5*
Paper, single sheet *5, 63*
    sprocket feed *5*
Paper-out detector *228, 276*
Parallel cable, Apple II *144*
    Atari *194*
    IBM-PC *140*
    Kaypro *178*
    Osborne 1 *178*
    TRS-80 Model I *161*
    TRS-80 Model II *162*
Parallel interface *228, 293*
    connector *4, 293*
    timing *294*
Pica pitch *30, 253*
Pie chart *114*
Pie chart program *111*
    Apple II *157*
    Atari *205*
    Commodore *222*
    Osborne 1 *189*
    TRS-80 *173*
Pitch *30, 253*
    changing with margins *58*
    changing with tabs *57*
    condensed *30*
    elite *30*
    pica *30*
Platen *2, 13*
Platen knob *2, 132, 133*
Plotting *107*
Power lamp *2*
Power requirements *292*
Power switch *2*
PR#1 *24, 144*
Preventive maintenance *123*
Primary fuse *126*
Print density *32*
Print head *13, 74*
    gap, adjusting *13*
    replacement *127*
Print matrix *74*
Print modes *257*
    mixing *36*
    with download characters *95*
Print pitch *30, 253*
Print style commands *251*
Printer cover *2, 132, 138*
Printer driver, TRS-80 *162*
Printer initialization *277*
Printing a design *104*
    logo *104*
    download characters *83*
    text *29*
    user-defined characters *83*
Printing speed *291*
Printing, bidirectional *277*
    dot matrix *74*
    faint *127*
    unidirectional *63, 277*
Problem codes with BASIC *27, 56, 117*
Program, bridge hand *85*
    Apple II *148*
    Atari *197*
    Commodore *215*
    Osborne 1 *181*
    TRS-80 *165*
Program, character design *95*

- Program, chart 37
  - Apple II 146
  - Atari 195
  - Commodore 212
  - Osborne 1 179
  - TRS-80 163
- Program, Delta Plot 107
  - Apple 155
  - Atari 203
  - Commodore 221
  - Osborne 1 187
  - TRS-80 172
- Program, download utility, 95
  - Apple II 152
  - Atari 200
  - Commodore 218
  - Osborne 1 185
  - TRS-80 169
- Program, logo 105
- Program, macro 70
  - Apple II 148
  - Atari 196
  - Commodore 214
  - Osborne 1 181
  - TRS-80 165
- Program, numeral 92
  - Apple II 150
  - Atari 199
  - Commodore 216
  - Osborne 1 183
  - TRS-80 167
- Program, pie chart 111
  - Apple II 157
  - Atari 205
  - Commodore 222
  - Osborne 1 189
  - TRS-80 173
- Program, special character chart 66
  - Apple II 147
  - Atari 196
  - Commodore 214
  - Osborne 1 181
  - TRS-80 165
- Programs, installation 18
- Proportional characters 88, 270
- Proportional widths 81
- Protocol, ACK 301
  - serial busy 300
  - XON/XOFF 300
  - serial 297
- Quadruple density graphics 115, 272
- Quick reference B/C
- RAM 62, 83, 269
- RAM, loading standard characters 83
- Ready lamp 2
- Reference, control codes 251
- Registration card 131
- Release lever 5
- Removing the upper case 124
- Replacing a fuse 125
- Replacing the print head 127
- Reset 62
- Resolution 100, 115
- Return, carriage 44, 266
- Ribbon, ink 132
  - installation 136
  - part number 292
- Right margin 57
- Roll paper 7
  - holder 2, 132
- ROM 74, 83, 269
- Screws, shipping 131
- Secondary fuse 127
- Self-test 14
- Semicolon 35, 43
- Separator, paper 2, 132, 134
- Serial busy protocols 300
- Serial cable, IBM-PC 141
- Serial interface 228, 297
  - connector 4, 297
- Serial protocol 297
- Seven bit interface 65, 67, 228
- Shaft, roll paper 132
- Shipping screws 131
- ⟨SI⟩ 255
- Signal levels 297
- Snap levers 134
- ⟨SO⟩ 256
- Software, commercial 18
- Spacing, line 43
- Spare fuse 132
- Special character chart program 66
  - Apple II 147
  - Atari 196
  - Commodore 214
  - Osborne 1 181
  - TRS-80 165
- Special print modes 257
- Special symbols 66
- Specifications 291
- Speed, printing 291
- Sprocket feed paper 5, 9
- Subscripts 35, 259
- Summary of commands in numeric order 279
- SuperCalc 21
- Superscripts 35, 259
- Switch, DIP 4
- Switch, power 2
- Synchronization 293
- Tab 55, 57
  - horizontal 55, 267
  - horizontal, one time 57
  - vertical 55, 58, 264
  - vertical, one-time 60
- Technical specifications 291
- Temperature 131

Terminology *19*
Test, self *14*
Text, combined with graphics *103*
 printing *29*
Thickness adjustment lever *14*
Thickness, paper *5*
Timing, parallel interface *294*
Top margin *50, 263*
Tractor feed *5*
Tractor unit *2, 133*
TRS-80 BASIC *162*
TRS-80 computer *23, 49, 161*
Underlining *35, 258*
Unidirectional printing *63, 277*
Universal/Atari Parallel Interface *193*
Universal/Commodore Parallel Interface *211*
Unpacking *131*
Upper case, removal of *124*
User printer controls *22*
User-defined characters *73, 269*
 with print modes *95*
 proportional *88*
 selecting *90*
User-defined codes *68*
Utility program *95*
Vertical positioning *260*
Vertical spacing *43*
Vertical tab *55, 58, 264*
Vertical tab, one-time *60*
VIC-20 computer *211*
Voltage *131*
⟨VT⟩ *59, 264*
Weight *292*
Width, column *291*
Width, paper *5*
Widths, characters *88*
 proportional *81, 88*
Word length *297*
Word processors *21*
WordStar *22*
XON/XOFF protocol *300*

DIP Switch Settings

Switch	ON	OFF	SETTING
DIP Switch 1			
1-1	11″ page length	12″ page length	
1-2	Normal print	Emphasized print	
1-3	10 CPI (pica pitch)	17 CPI (condensed pitch)	
1-4	Normal	Italic	
1-5	1/6″ line feed	1/8″ line feed	
1-6	International character set selection-see below		
1-7			
1-8			
DIP Switch 2			
2-1	Paper-out detector on	Ignore paper-out	
2-2	Serial interface	Parallel interface	
2-3	7-bit interface	8-bit interface	
2-4	Auto LF with CR	LF must be from host	
	DIP Switch 3		
3-1	7 data bits	8 data bits	
3-2	Parity checked	No parity	
3-3	Handshaking protocols—see below		
3-4			
3-5	Odd parity	Even parity	
3-6	Data transfer rate—see below		
3-7			
3-8			

International character sets

Switch	USA	England	Germany	Denmark	France	Sweden	Italy	Spain
1-6	ON	OFF	ON	OFF	ON	OFF	ON	OFF
1-7	ON	ON	OFF	OFF	ON	ON	OFF	OFF
1-8	ON	ON	ON	ON	OFF	OFF	OFF	OFF

Handshaking protocols

Protocol	Switch 3-3	Switch 3-4
Serial busy, 1 byte mode	OFF	OFF
Serial busy, 1 block mode	ON	OFF
ACK mode	OFF	ON
XON/XOFF	ON	ON

Data transfer rates

Baud rate	Switch 3-6	Switch 3-7	Switch 3-8
110	OFF	OFF	OFF
110	OFF	OFF	ON
300	OFF	ON	OFF
600	OFF	ON	ON
1200	ON	OFF	OFF
2400	ON	OFF	ON
4800	ON	ON	OFF
9600	ON	ON	ON

Use the "setting" column to record the way the switches are set in your printer.

Command Quick Reference

Commands to control print style

⟨ESC⟩ “5”	Cancel italic print
⟨ESC⟩ “4”	Italic print
⟨ESC⟩ “7” n	Select international character set

Font pitch controls

⟨ESC⟩ “B” CHR$(1)	Pica pitch
⟨ESC⟩ “B” CHR$(2)	Elite pitch
⟨ESC⟩ “B” CHR$(3)	Condensed pitch
CHR$(18)	Pica pitch
CHR$(15)	Condensed pitch
⟨ESC⟩ CHR$(15)	Condensed pitch
⟨ESC⟩ “W” CHR$(1)	Expanded print
CHR$(14)	Expanded print
⟨ESC⟩ CHR$(14)	Expanded print
⟨ESC⟩ “W” CHR$(0)	Cancel expanded print
CHR$(20)	Cancel expanded print

Special print modes

⟨ESC⟩ “G”	Double-strike print
⟨ESC⟩ “H”	Cancel double-strike print
⟨ESC⟩ “E”	Emphasized print
⟨ESC⟩ “F”	Cancel emphasized print
⟨ESC⟩ “-” CHR$(1)	Start underlining
⟨ESC⟩ “-” CHR$(0)	Stop underlining
⟨ESC⟩ “S” CHR$(0)	Superscript on
⟨ESC⟩ “S” CHR$(1)	Subscript on
⟨ESC⟩ “T”	Cancel super and subscripts

Commands to control vertical position of the print head

CHR$(10)	Line feed
⟨ESC⟩ “0”	Set line feed to 1/8 inch
⟨ESC⟩ “1”	Set line feed to 7/72 inch
⟨ESC⟩ “2”	Set line feed to 1/6 inch
⟨ESC⟩ “A” n	Set line feed to *n*/72 inch
⟨ESC⟩ “3” n	Set line feed to *n*/144 inch
⟨ESC⟩ “J” *n*	Single line feed of *n*/144 inches

Form feed controls

CHR$(12)	Form feed
⟨ESC⟩ “C” *n*	Set page length to *n* lines
⟨ESC⟩ “C” CHR$(0) *n*	Set page length to *n* inches
⟨ESC⟩ “R” *n*	Set top margin at line *n*
⟨ESC⟩ “N” n	Set bottom margin at n lines
⟨ESC⟩ “O”	Cancel top and bottom margins

Vertical tabs

CHR$(11)	Vertical tab
⟨ESC⟩ “P” . . . CHR$(0)	Set vertical tabs
⟨ESC⟩ “a” *n*	Advance *n* line feeds

Commands to control horizontal position of the print head

CHR$(13)	Carriage return
⟨ESC⟩ “M” *n*	Set left margin at column *n*
⟨ESC⟩ “Q” *n*	Set right margin at column *n*
CHR$(9)	Horizontal tab
⟨ESC⟩ “D” . . . CHR$(0)	Set horizontal tabs
⟨ESC⟩ “b” n	Tab over *n* columns
CHR$(8)	Backspace

Download character commands

⟨ESC⟩ “*” CHR$(1) *n1 n2 m1 m2* . . . *m11*	Define download character
⟨ESC⟩ “*” CHR$(0)	Copy standard ROM characters to download RAM
⟨ESC⟩ “X” CHR$(1)	Use proportional download characters
⟨ESC⟩ “X” CHR$(0)	Cancel proportional download characters
⟨ESC⟩ “$” CHR$(1)	Use normal download characters
⟨ESC⟩ “$” CHR$(0)	Cancel normal download characters

Commands to control graphics

⟨ESC⟩ “K” *n1 n2*	Normal density graphics
⟨ESC⟩ “L” *n1 n2*	Double density graphics
⟨ESC⟩ “y” *n1 n2*	Double speed, double density graphics
⟨ESC⟩ “z” *n1 n2*	Quadruple density graphics

Macro instruction commands

⟨ESC⟩ “+” . . . CHR$(30)	Define macro
⟨ESC⟩ “!”	Use macro

Other function codes

⟨ESC⟩ “⟩”	Set eighth bit to 1
⟨ESC⟩ “=”	Set eighth bit to 0
⟨ESC⟩ “#”	Accept eighth bit as is
CHR$(127)	Delete last character
CHR$(19)	Off line
CHR$(17)	On line
CHR$(7)	Sounds bell
⟨ESC⟩ “Y” CHR$(0)	Disable bell
⟨ESC⟩ “Y” CHR$(1)	Enable bell
⟨ESC⟩ “8”	Ignore paper-out signal
⟨ESC⟩ “9”	Enable paper-out signal
⟨ESC⟩ “U” CHR$(1)	Unidirectional print
⟨ESC⟩ “U” CHR$(0)	Bidirectional print
⟨ESC⟩ “@”	Reset the printer