MX PRINTER MANUAL

with GRAFTRAX PLUS

EPSON

EPSON

EPSON AMERICA, INC.

EPSON
MX Printer Manual
GRAFTRAX^{PLUS}

by David A. Lien

COMPUSOFT® PUBLISHING
A Division of CompuSoft, Inc.
Box 19669
San Diego, California 92119, U.S.A.

International Standard Book Number: 0-932760-10-4
Library of Congress Catalog Card Number: 82-70351

10 9 8 7 6 5 4 3 2 1

Printed in the United States of America

FCC Compliance Statement For American Users

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 into a different outlet so that computer and receiver are on different branch circuits.

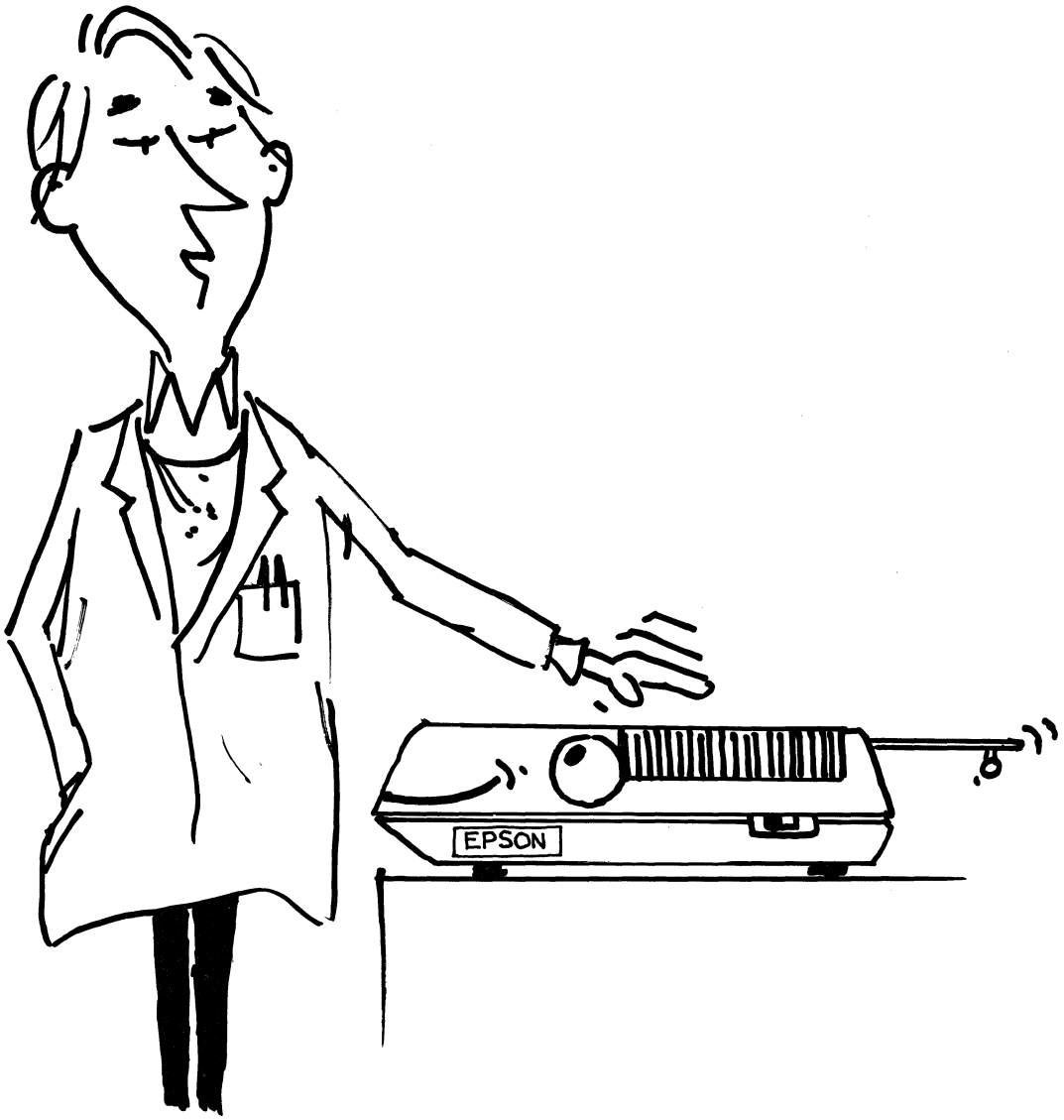
If necessary, the user should consult the dealer or an experienced radio/television technician for additional suggestions. The user may find the following booklet prepared by the Federal Communications Commission helpful:

"How to Identify and Resolve Radio-TV Interference Problems".

This booklet is available from the U.S. Government Printing Office, Washington, DC 20402. Stock No. 004-000-00345-4.

Trademark Acknowledgements

TRS-80 is a trademark of Radio Shack, a division of Tandy Corporation. Centronics is a trademark of Data Computer Corporation. Atari is a trademark of Atari Inc., a Warner Communications Company. Apple is a trademark of Apple Computer Inc. Microsoft is a trademark of Microsoft.


EPSON

A Personal Note from the Author

Congratulations on your decision to buy an EPSON MX series printer! It is a truly remarkable piece of hardware, and I believe is the best dollar value in computer printers on the market today.

"WHO NEEDS A LEARNER'S MANUAL FOR A PRINTER?" The answer — ALMOST EVERYONE EXCEPT A COMPUTER PROFESSIONAL.

Today's printers are very sophisticated compared to those of the 1970's. Most are not fully utilized because the instructions are too foggy and complicated. At the risk of insulting the alleged intelligence of a few, we're doing our best to eliminate that problem with all printers in the EPSON MX series.

I encourage you to learn all about your remarkable printer. You paid for it. Put it to work!

Dr. David A. Lien
San Diego — 1982

Table of Contents

FCC Compliance Notice iii
Trademark Acknowledgements iii
Personal Note from the Author v
Table of Contents vii
Introduction ix
Chapter 1: The Starting Line 1-1
Chapter 2: Send it a Message 2-1
Chapter 3: Decoding the Message 3-1
Chapter 4: More Print Control Commands 4-1
Chapter 5: Advance Printing and Endless Forms Control 5-1
Chapter 6: Special Features and Word Processing 6-1
Chapter 7: Seven Bits Foreign Symbols and Line Graphics Characters ... 7-1
Chapter 8: Skipping Over the Perforation 8-1
Chapter 9: Stuck with the TAB 9-1
Chapter 10: Single Sheet Friction Feed 10-1
Chapter 11: Dot Matrix Printing 11-1
Chapter 12: Introduction to Dot Graphics 12-1
Chapter 13: Advanced Graphics 13-1
Chapter 14: The Final Push 14-1
Chapter 15: Using the HI-RES Screen Dump Program 15-1
Appendix A: ASCII Chart A-1
Appendix B: Control Codes B-1
Appendix C: Control Key Table C-1
Appendix D: Sample Print Modes D-1
Appendix E: Internal Switches E-1
Appendix F: Character Fonts F-1
Appendix G: Use with TRS-80 Computers G-1
Appendix H: Use with the Atari H-1
Appendix I: Use with the IEEE Interface I-1
Appendix J: Use with the Apple J-1
Appendix K: Special Interfacing Considerations K-1
Appendix L: Printer Maintenance L-1
Appendix M: Technical Specifications M-1
Appendix N: Control Circuit N-1
Appendix O: Pinout Chart — Parallel Interface O-1
Appendix P: Data Transmission Sequence P-1
Appendix Q: Schematics Q-1
Appendix R: GRAFTRAX^{PLUS} ROM Set Installation Instructions R-1
Appendix S: Troubleshooting S-1
Appendix T: Mode Flags T-1
Index
Notice
Limited Warranty

Introduction

"But do I **really** have to take a course on 'How to operate a printer' to use this one?" No — not if you only want to use it for mundane printing. It prints "mundanes" very nicely.

If you want to **understand** and **use** the many exotic features it offers, approach your new printer in the same way as your first computer — with a healthy curiosity and an open mind. They probably far exceed what you expected from a printer inexpensive enough to call your own.

The MX-series printers work with virtually any computer properly interfaced to them. This learner's manual uses the popular MicroSoft BASIC as implemented on the TRS-80, since that version of BASIC has almost become the industry standard. However, we do switch to Applesoft in the final chapter to demonstrate the EPSON screen dump routine.

What's taught here applies to all computers, not just the Apple and TRS-80, though not all are powerful enough to take advantage of every MX-printer feature. More on that in good time.

Impatient readers should head for the appendices and usual reference materials in the back. The rest of us will believe that this is no ordinary printer and take the time to learn to use it right — the first time.

A good working knowledge of the BASIC* language is all that's required as we begin, with Chapter 1.

*Users who feel their BASIC skills are a little rusty are referred to the following books by the same author.

The BASIC Handbook (2nd Edition), Encyclopedia of the BASIC language. 480 Pages. Available through CompuSoft Publishing, Box 19669, San Diego, CA 92119.

Learning BASIC. 520 Pages. Available through CompuSoft Publishing, Box 19669, San Diego, CA 92119.

WARNING

High voltage exists inside this unit and the case should be opened only by a qualified person!

Chapter 1

The Starting Line

This important chapter shows how to unpack the printer, set it up and test it. Read it **before** you get into trouble. The time spent will be our best investment since buying the printer itself.

If you've just purchased a GRAFTRAX^{PLUS} upgrade kit for an EPSON printer purchased before March 1982, see Appendix R for installation instructions before continuing. If this manual was included with your printer, it is already equipped with GRAFTRAX^{PLUS}.

Counting the Parts

Open the box carefully and remove the contents. We should find:

1. This user's manual (obviously . . .)
2. Your new EPSON printer
3. A long box containing the ribbon cartridge
4. A wire rack to guide the paper
5. Plastic brackets and tube (MX-100 only)

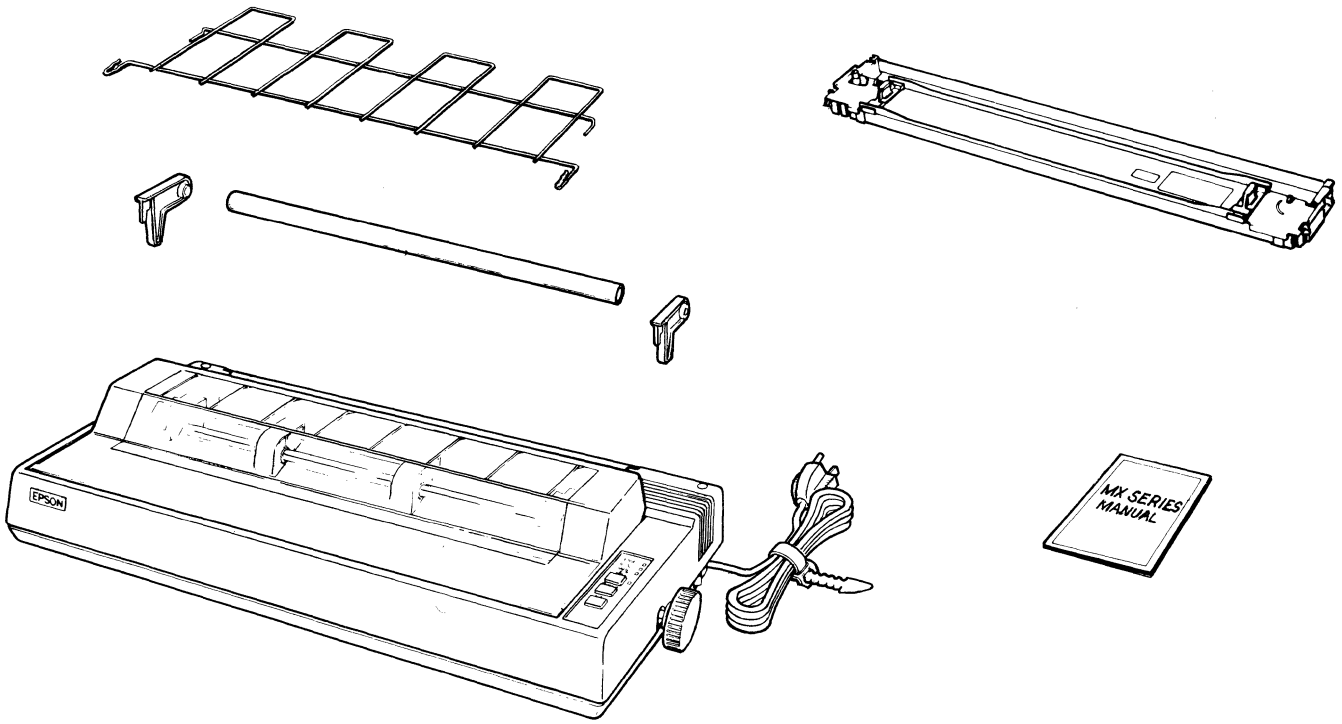


Figure 1-1A MX-100

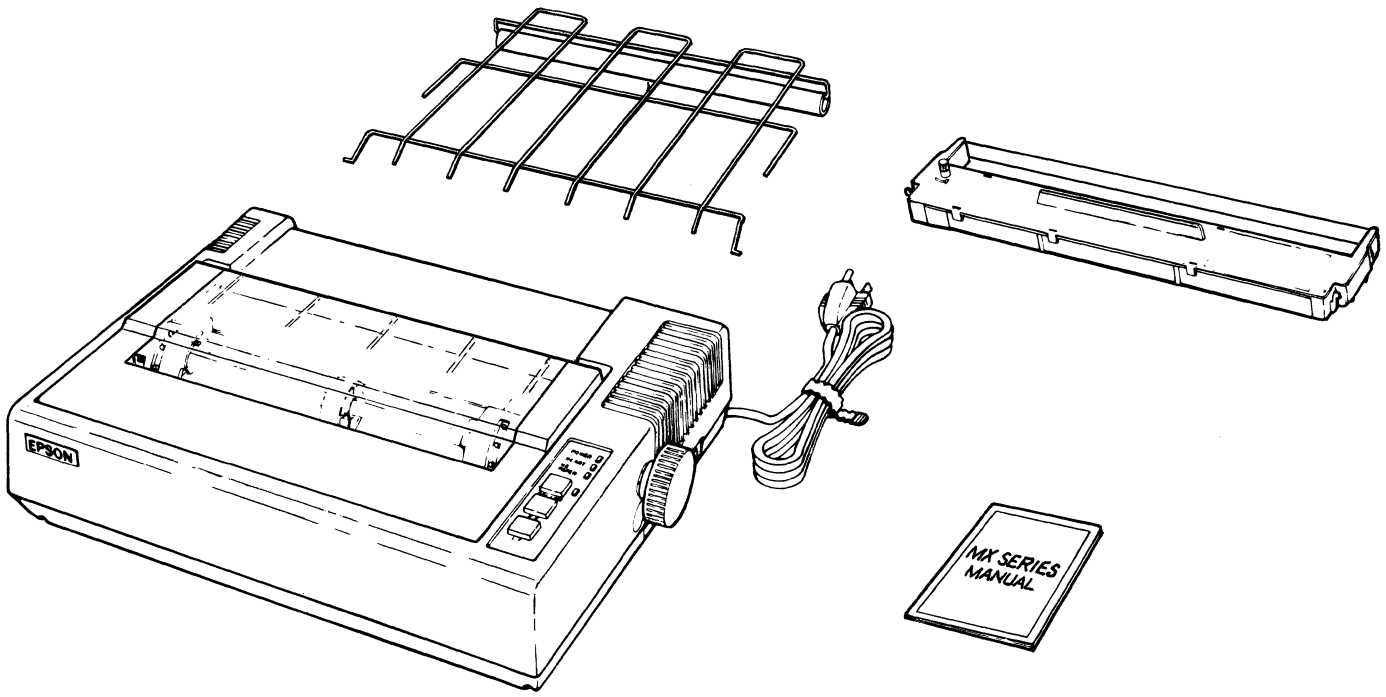


Figure 1-1B MX-80

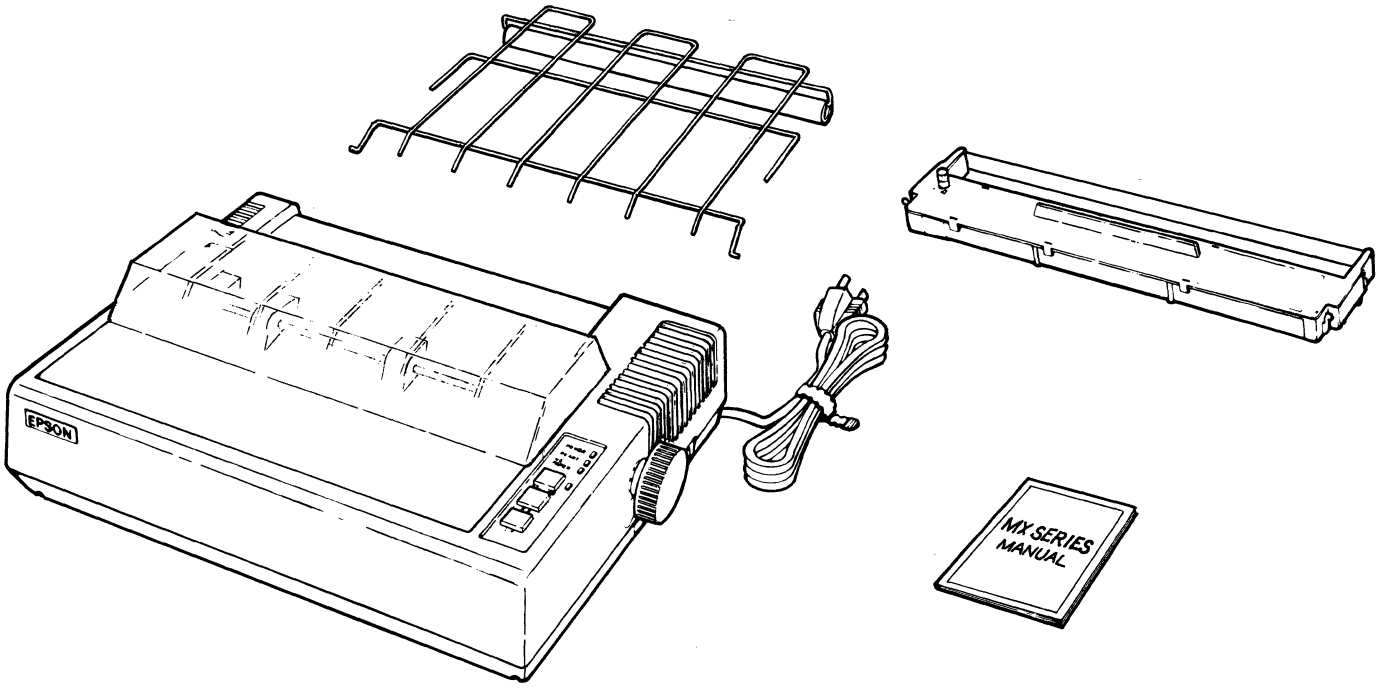


Figure 1-1C MX-80 F/T

NOTE: MX-80 F/T users who wish to use roll paper can order a special roll paper holder from their EPSON dealers.

We also need a cable to connect the printer to your particular computer. Your EPSON dealer can provide parallel cables that match the TRS-80 and Apple computers. If you use a non-EPSON cable, make sure it is wired properly.

Computers that do not use the Centronics™ standard parallel interface scheme need a special interfacing kit. EPSON dealers can supply kits for many popular computers. See Appendices I and K for optional interfaces.

Setting it up

Let's first remove the printer lid so we can work without breaking something.

Lay the printer flat on a firm surface and raise its lid to the full vertical position. These lids are rugged, but it is important that they be **fully vertical** when removed (Figure 1-2).

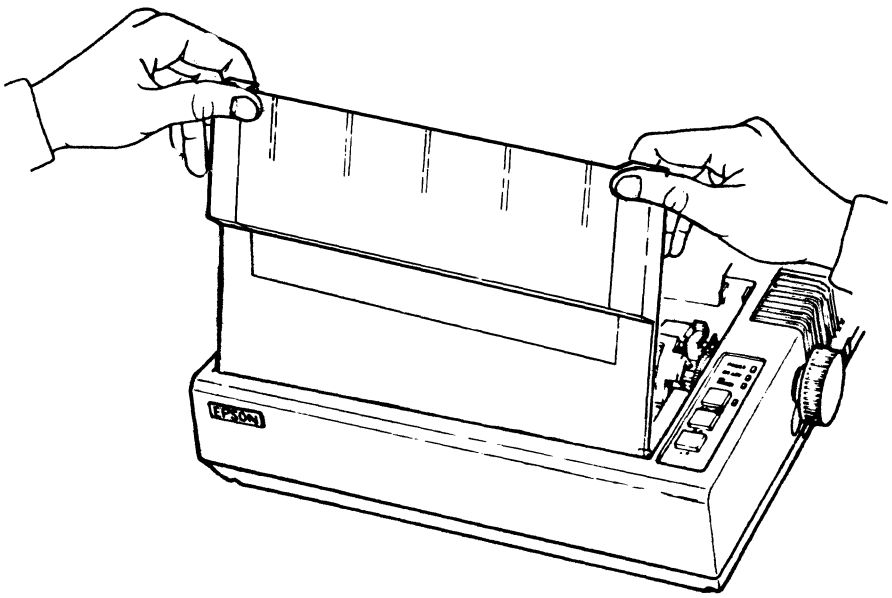


Figure 1-2

With the printer facing you, as shown in Figure 1-3, grasp the left top side of the lid and lift off the entire lid. It's simple. To replace the lid later, just reverse the procedure.

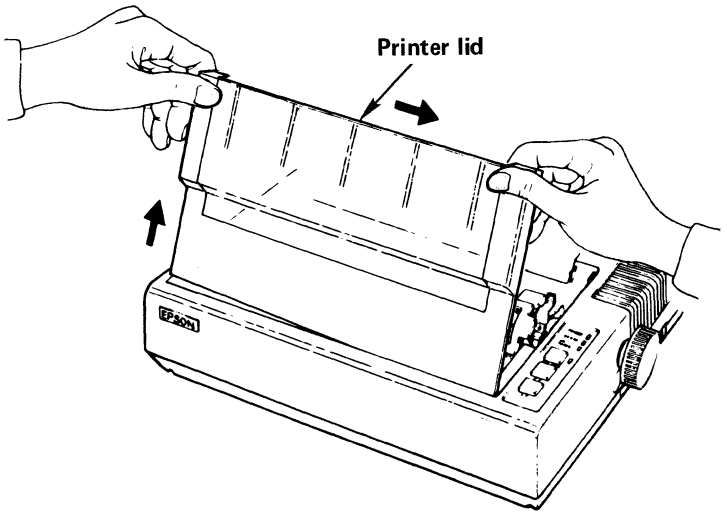


Figure 1-3

Tractor Feed Removal (MX-80 F/T and MX-100 only)

The tractor-feed paper mechanism is removable so individual sheets, like letterhead, can be easily fed through the printer. We'll learn how to use the "friction-feed" mechanism in a later chapter, but must remove the tractor mechanism now to gain easy access to some internal switches we have to set.

Locate the metal locking levers on the tractor housing (Figure 1-4). Place your thumbs on the pin-feed covers and with your forefingers pull these levers toward the **front** of the printer.

Rock the whole works backwards towards the **rear** of the printer, and lift it off.

It's easy once we get the hang of it.

Shipping Screws

Turn the printer over and lay it on a soft surface. With a Phillips-head screwdriver, remove the shipping screws and save them in case you decide later to lend the printer to Aunt Bernice in Lake Wobegon, Minnesota. These screws are in place during shipping to protect the printer from damage and are located as shown in Figure 1-5.

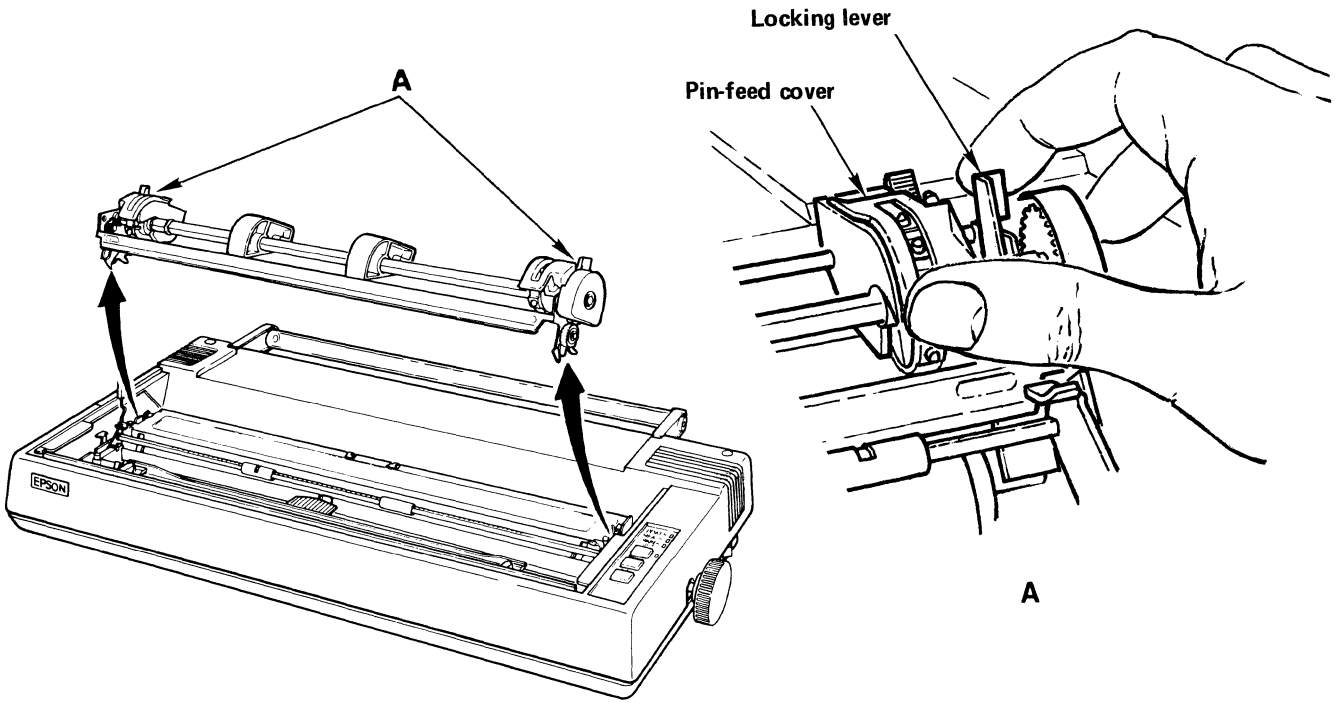


Figure 1-4

Installing the Ribbon

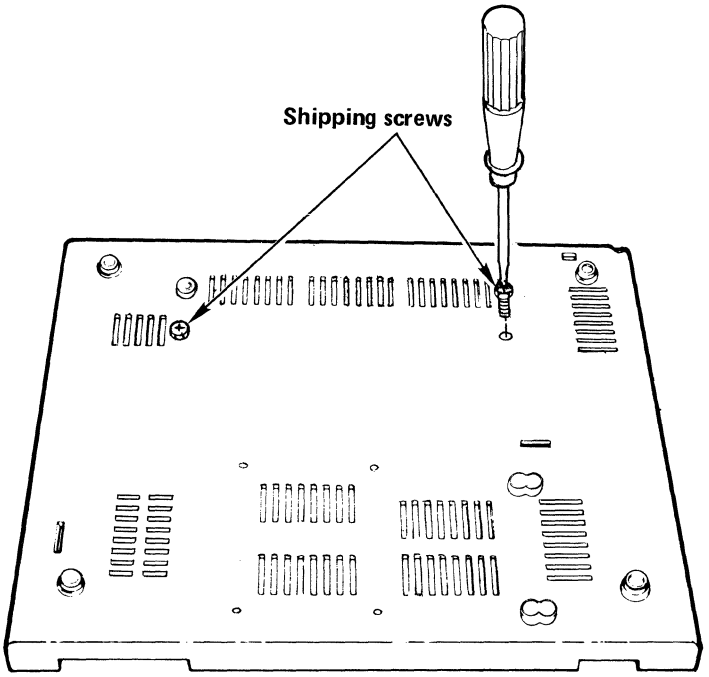
Position the printer right-side up with the EPSON label (front) facing you. Locate the paper bail (the movable metal bar with numbers 1 - 80 or 1 - 136 on it) and push it toward the back, away from you.

CAUTION: If you've not followed the directions and already tried to run the printer be careful of the print head! It may run **hot** under heavy operation.

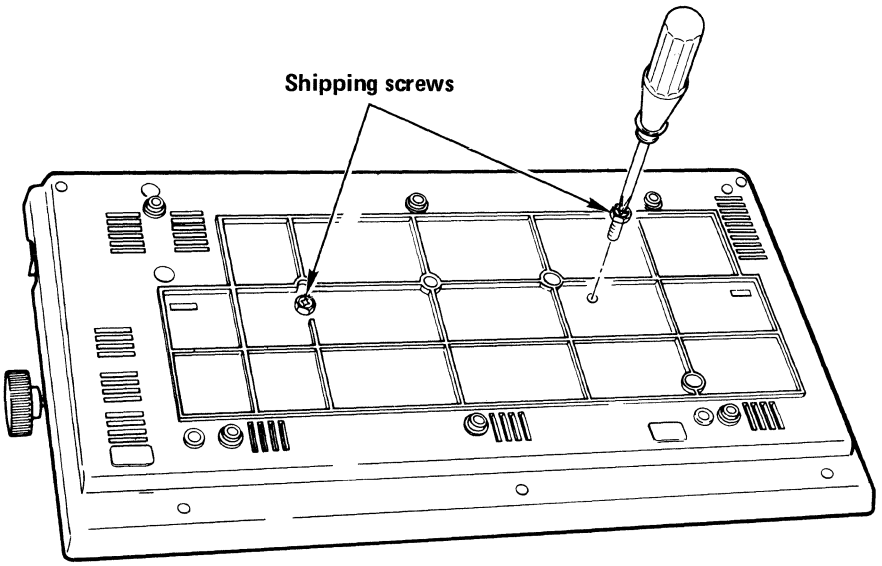
Remove the ribbon cartridge from its box and turn the plastic knob counterclockwise so the ribbon is tight (Figure 1-6A).

Holding the cartridge by its handle(s), steer the 4 tabs on its sides into the 4 slots in the printer's metal frame (Figure 1-6B). Press the cartridge firmly into place.

Using a pencil (or your fingers), lift the ribbon into the slot in front of the print head as shown in Figure 1-6C. With a little practice, we can do it in one quick operation.



MX-80, MX80 F/T



MX-100

Figure 1-5

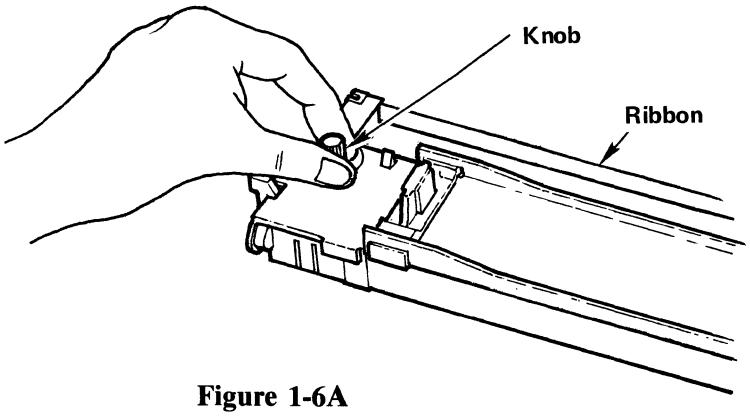


Figure 1-6A

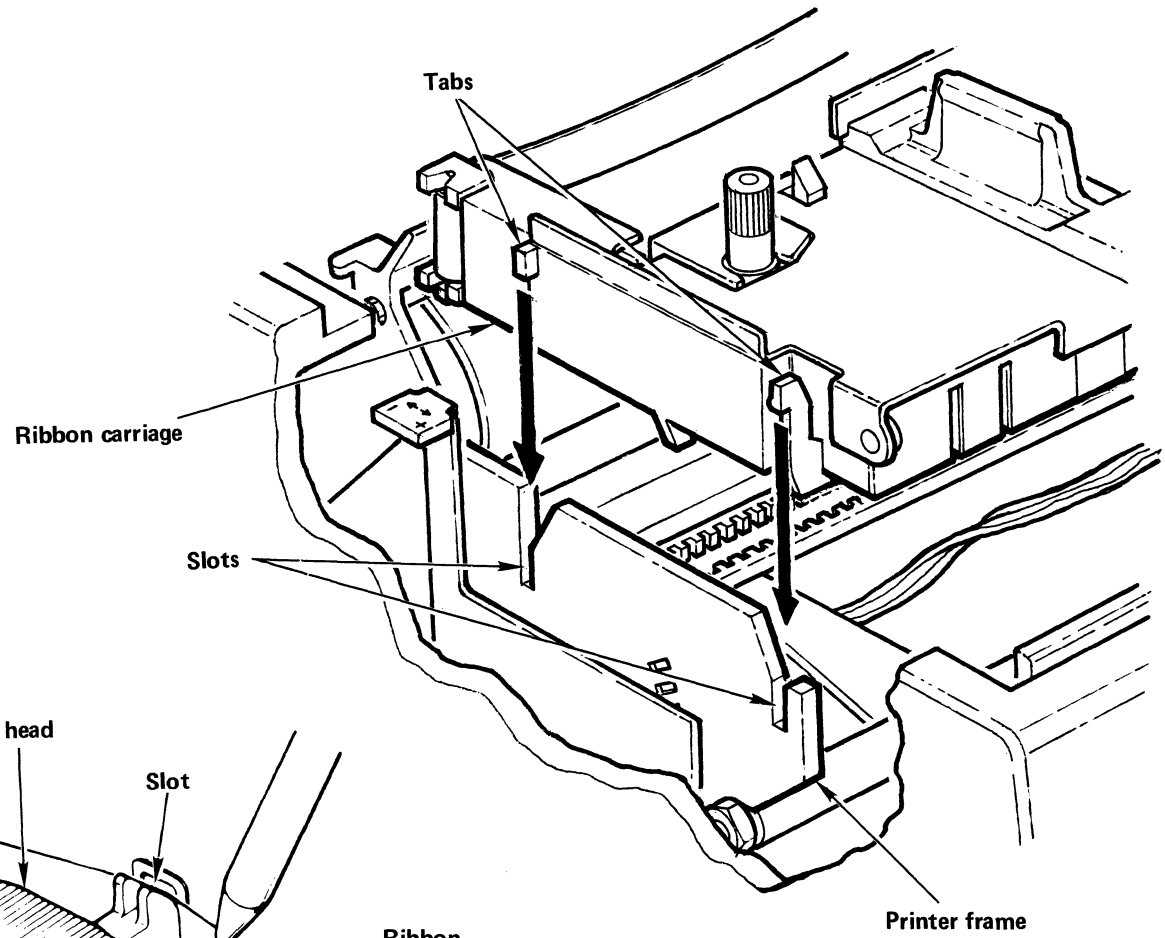


Figure 1-6B

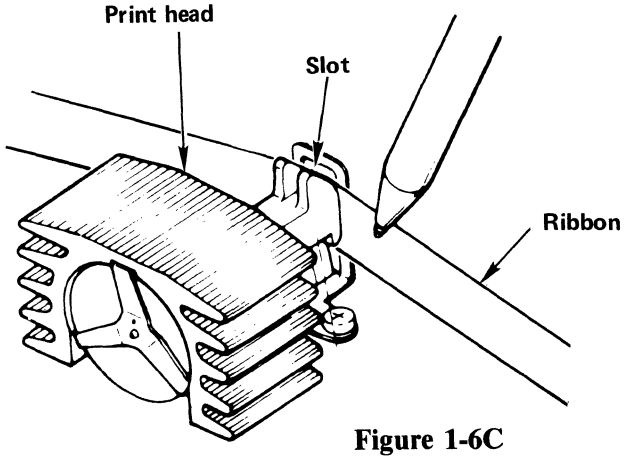


Figure 1-6C

Wind the ribbon tight again with the little plastic knob, and it's all set to go.

MX-80 owners may notice a sticker with the words EXCHANGE TIMES on the ribbon cartridge. This is a remnant of an early plan to refill the cartridges when the ribbon was worn out. EPSON **does** offer refills for the MX-100 cartridges. Some computer and office supply stores also offer refilling services at a considerable cost savings over new ones. To avoid damage to the print head, be sure to use only EPSON-approved ribbons.

Matching the Printer to the Computer

There are 12 different switches inside the printer case. They can be set to make the MX printer match a wide variety of computers and computer interfaces. We have to set them now to match our computer. In later chapters we will reset some of them to make the printer do some very unique things.

Opening the Case

To get at the switches we have to open up the case. (The translucent lid should remain off for much of our learning throughout this book.) Remove the roller knob by pulling straight, with firm but steady pressure (Figure 1-7).

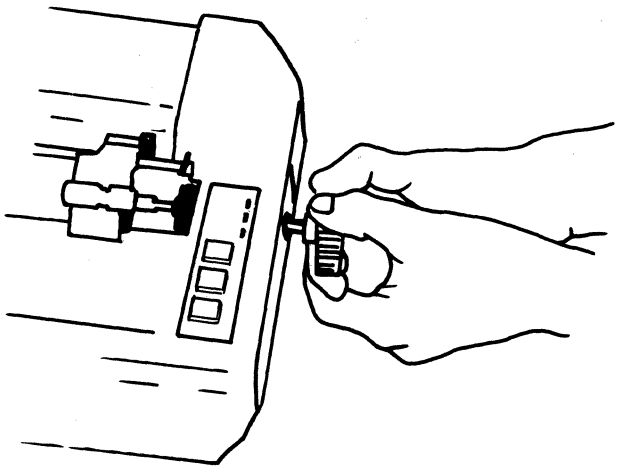


Figure 1-7

With a Phillips-head screwdriver, completely loosen (but don't remove) all screws, as shown in Figure 1-8. (MX-80 owners have to turn the printer over to remove the screws.) Place tape over the holes so the screws won't fall out when we remove the printer cover.

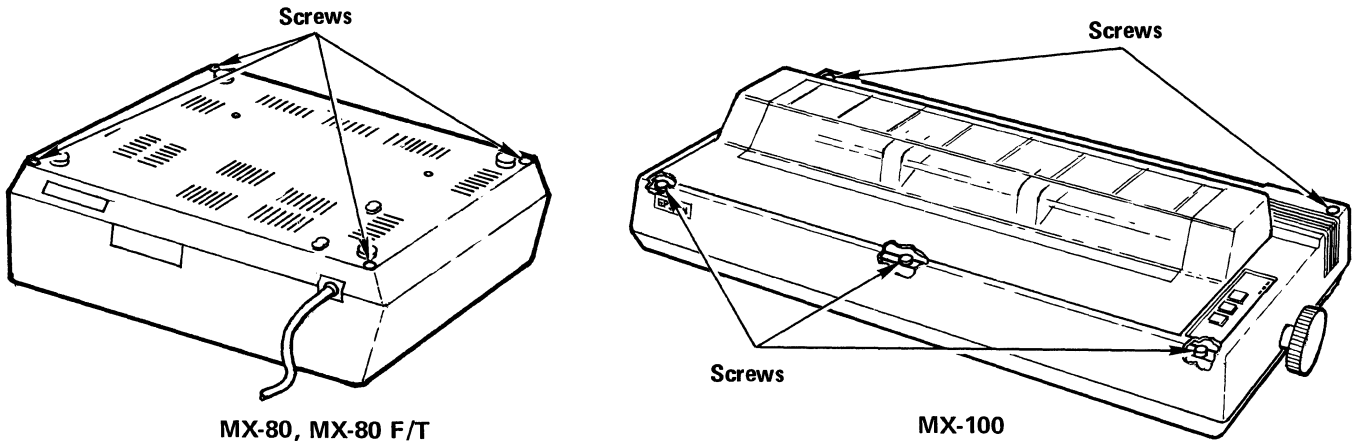


Figure 1-8

Place the printer right-side up with the EPSON label facing you. **Gently** wiggle the top cover loose. Wires are hooked to it, so be careful! We are not going to completely remove the cover — we're only going to open it enough to gain access to the switches.

Lift the cover up. Be careful not to pull the wires on the right-hand side.

Tip it **gently** forward and slightly askew, as shown in Figure 1-9. With just a bit of class we can maneuver the cover so it stands firmly in place, as a Sentinel guarding the goodies.

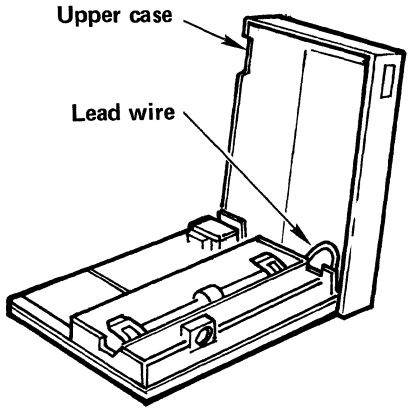


Figure 1-9

If the wires are too tight, we can disconnect them from the upper case, as shown in Figure 1-10. With care, disconnecting should not be necessary.

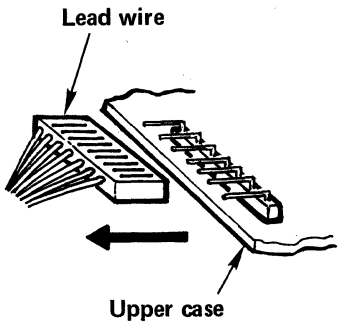


Figure 1-10

Take a minute to gawk at all the stuff in the box. Wow! As Custer said, "Look at all those . . . well, anyway."

Wonder how they sell it as cheap as they do? (Hope it prints as good as it looks.)

Setting the Switches

Position the printer as shown in Figure 1-11. Look for the 2 clear plastic caps. They are dust covers, and snap right off. **Gently** remove them.

The assembly on the right contains 8 switches, and is labeled SW1. The smaller one to its left contains 4 switches and is labeled SW2. Switches slid to the left are ON — those to the right are OFF. So far, so good.

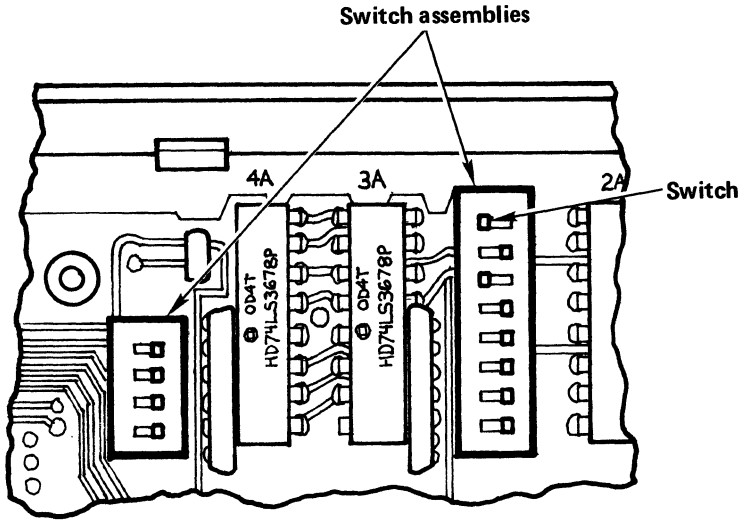


Figure 1-11

The switches probably left the factory burn-in and checkout station set something like this:

SW2	SW1
	1-8 L
	1-7 R
	1-6 L
	1-5 R
2-4 R	1-4 R
2-3 R	1-3 R
2-2 R	1-2 R
2-1 R	1-1 R

Their functions are explained in Appendix E. No need to look that up, now. Just switch them as shown above so we all start with our printers set up with the same features.

We must double-check switch 2-3. It may be factory set to the OFF (right) position. This setting is perfect for computers that automatically send a line feed with every carriage return, like the Apple. If your machine does not send automatic line feeds (e.g., TRS-80, Atari, and most others), set switch 2-3 to ON (left). If you aren't certain what your computer does, flip a coin, set switch 2-3 to ON, and leave the case loose so we can easily change it later if necessary. Double check all other switches to be sure they are set as shown above. Never move the switches when the power is ON. In fact, if the power is ON, we have no business being inside the printer!

Keep in mind that this manual explains the capabilities of the printer. It is up to you to understand your own computer and its peculiarities. We have included information on interfacing and using some of the more popular computers in Appendices G - K. A complete description of what each switch controls is found in Appendix E for those who simply can't wait to find out. The rest of us will forget about the switches and continue straight ahead.

A plastic plate is taped to the access hatch on the inside back of the case. We can either leave it in place as we reassemble the case, tape it to the outside of the hole, or add it to our growing collection of printer parts. In theory, the switches can be reset through this hatch without having to reopen the case by using a flashlight and small screwdriver or ball-point pen — like a skilled dentist.

Put the case back together and tighten the screws. The printer case assembles the same way it disassembled — very carefully. Same with the transparent lid.

Printer Cable Connection

We're getting closer to the good stuff.

Be sure both the computer and printer are turned OFF. Connect the cable to the printer **only**. EPSON can supply cables to match some computer systems. Be sure you have the right one. Do **not** hook the other end of the cable to the computer.

Take your time! Double-check to ensure that the cable is in place and locked. It may take a firm push on the connector to secure the clips. Without this tight connection, printing could be erratic. Again — leave the other end **disconnected**.

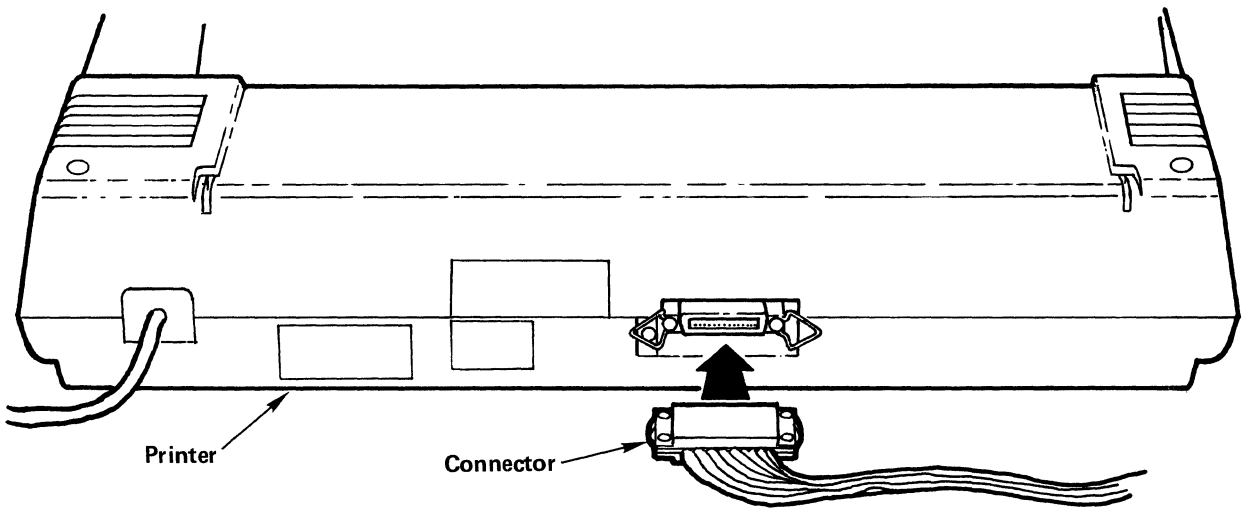
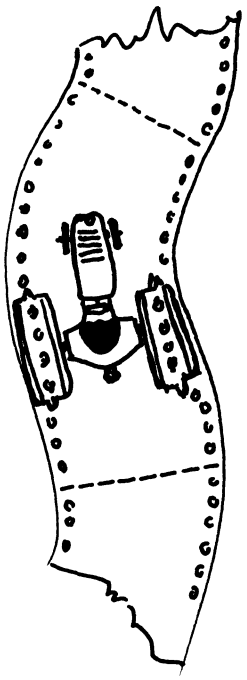


Figure 1-12



Installing the Tractor Feed

If you bought the wide-body model of the MX series, the MX-100, it's likely that a good portion of your printing will require tractor feed paper, typically about 15½ inches wide. Although most of the examples in this manual require only 9½ inch-wide paper, the self test at the end of this chapter will require MX-100 users to have wide tractor-feed printer paper.

To use tractor feed paper, we'll have to reattach the tractor mechanism. It's very easy to install, but even easier to try to install wrong! (The most common mistakes are trying to put it on backwards and misidentifying the "hooks.")

Latch the hooks on the tractor unit to the two small studs on each side of the printer (Figure 1-13).

Rock the whole unit forward until it locks into place.

Pull the RELEASE lever toward the front of the printer so the paper won't be squeezed by the friction feed rollers at the same time it's being towed along by the tractors. (More on them later.)

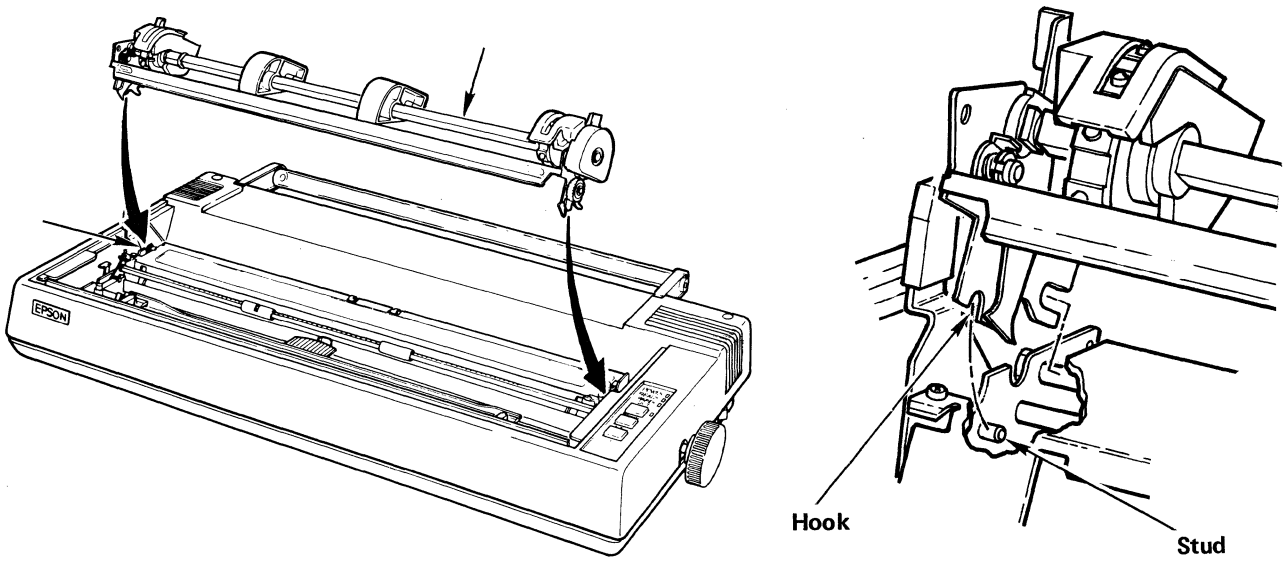


Figure 1-13

The Paper Separator

The metal rack is a paper separator. It allows the paper to feed smoothly into and out of the printer and prevents it from getting tangled up with the cables in the back.

Remove the paper separator from its packing material. MX-100 owners will also find a long plastic roller and two plastic brackets. Slide these brackets firmly into the two slots at the rear of the printer case and insert the roller between them, as shown in Figure 1-14.

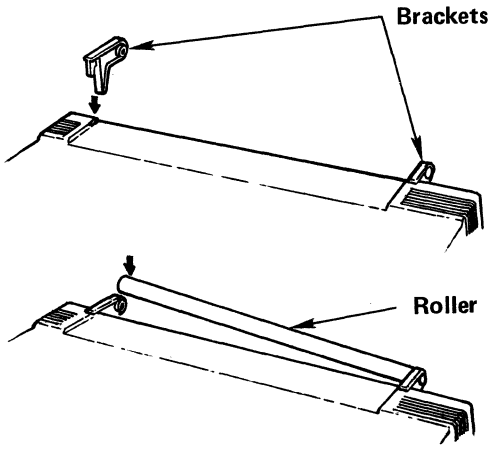


Figure 1-14

Now connect the paper separator to the bar at the top of the tractor unit, as shown in Figure 1-15. It connects to the metal framework on the MX-80. Make sure it is right-side up so there is room for the paper to feed beneath it.

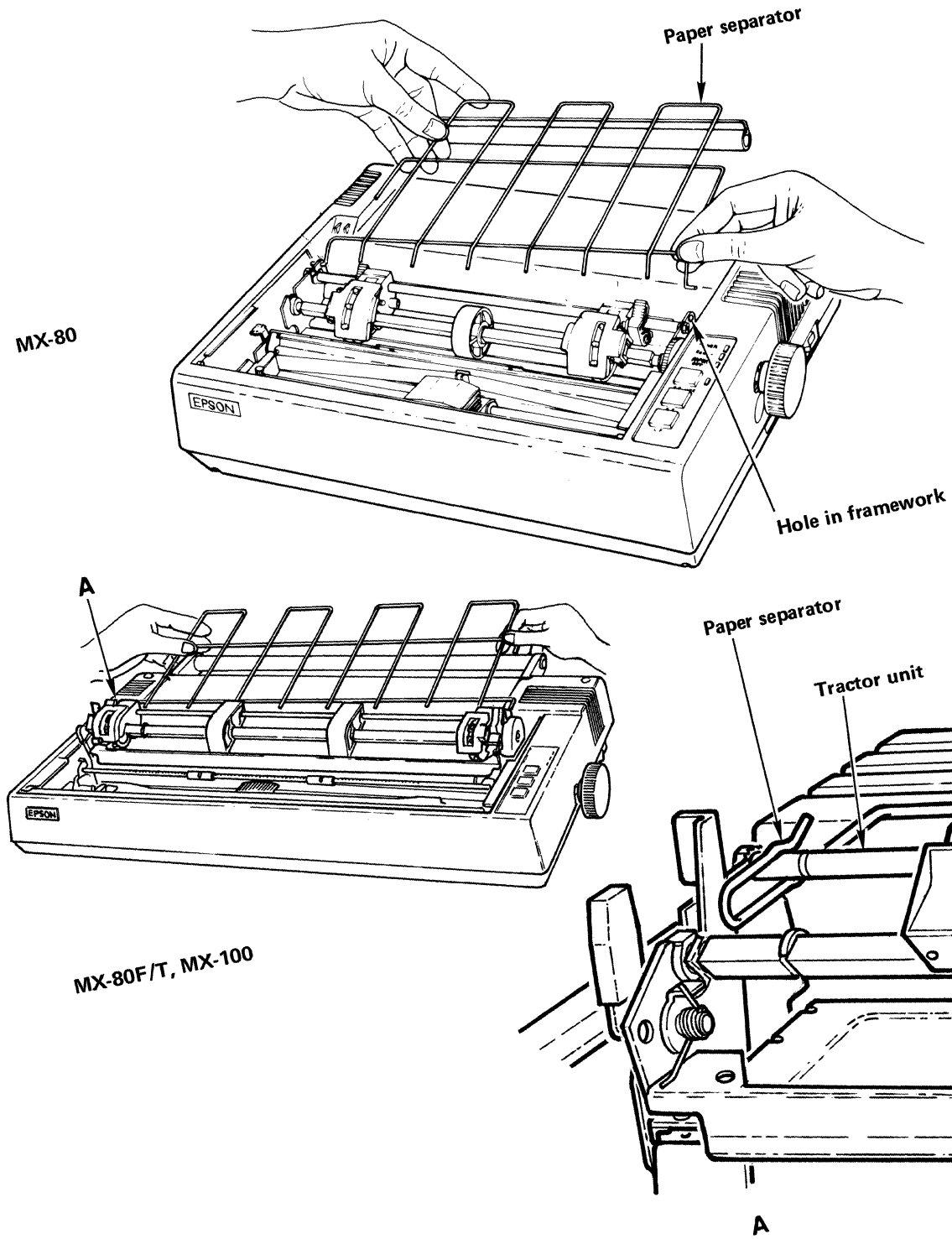

Paper separator
MX-80
Hole in framework
A
Paper separator
Tractor unit
MX-80F/T, MX-100
A

Figure 1-15

Feeding the Paper

The MX-80 tractor unit can handle pin-feed paper between 4 inches and 10 inches wide. The MX-100's range is 4 inches to 15½ inches. Both the left and right hand tractors are adjustable to match the spacing between the holes. (Tractors — we never get far from the land, do we?)

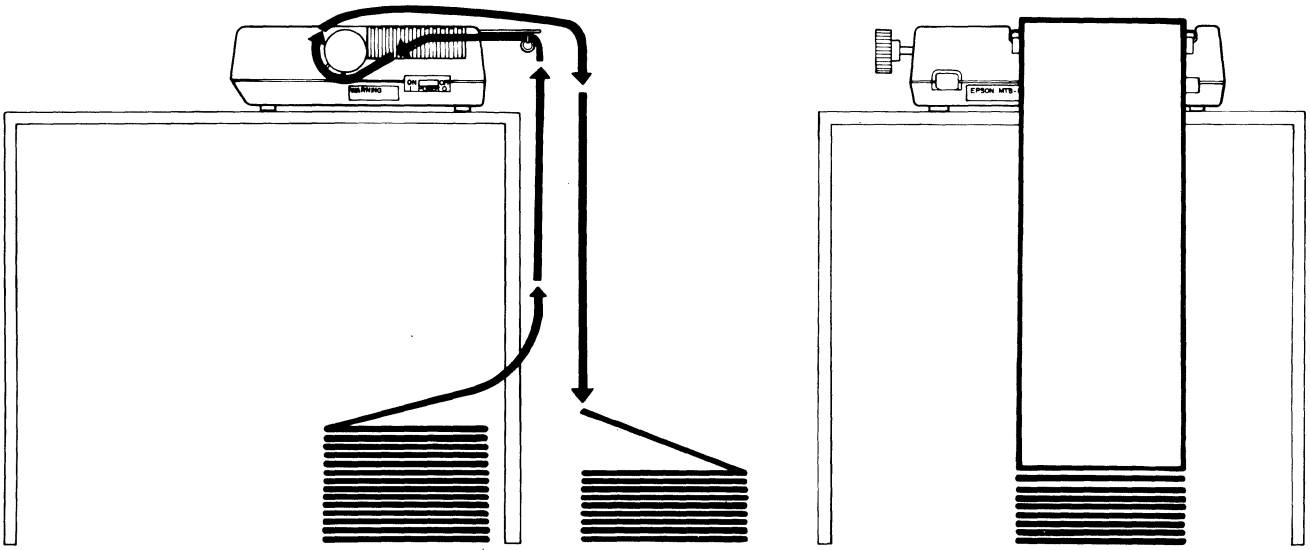


Figure 1-16

Position the paper on the floor behind and below the printer. Reliable operation depends somewhat on the weight of the paper keeping itself taut. Pull the black RELEASE lever (MX-80 F/T and MX-100) and the paper bail toward you. Open both tractor covers and position the plastic spacer(s) to your liking (Figure 1-17).

Feed the incoming paper **above** the plastic tube but **below** the wire frame, into the slot and right on around to the tractors. Move the tractors as necessary to match the paper hole spacing. The tractor position lock levers are shown in Figure 1-17.

Position the paper holes on top of the tractor teeth and close the tractor covers. Adjust one or both tractors so the paper is centered as you wish it, and is held firmly in place. Push the bail back up against the paper.

Roll the paper forward with the roller knob.

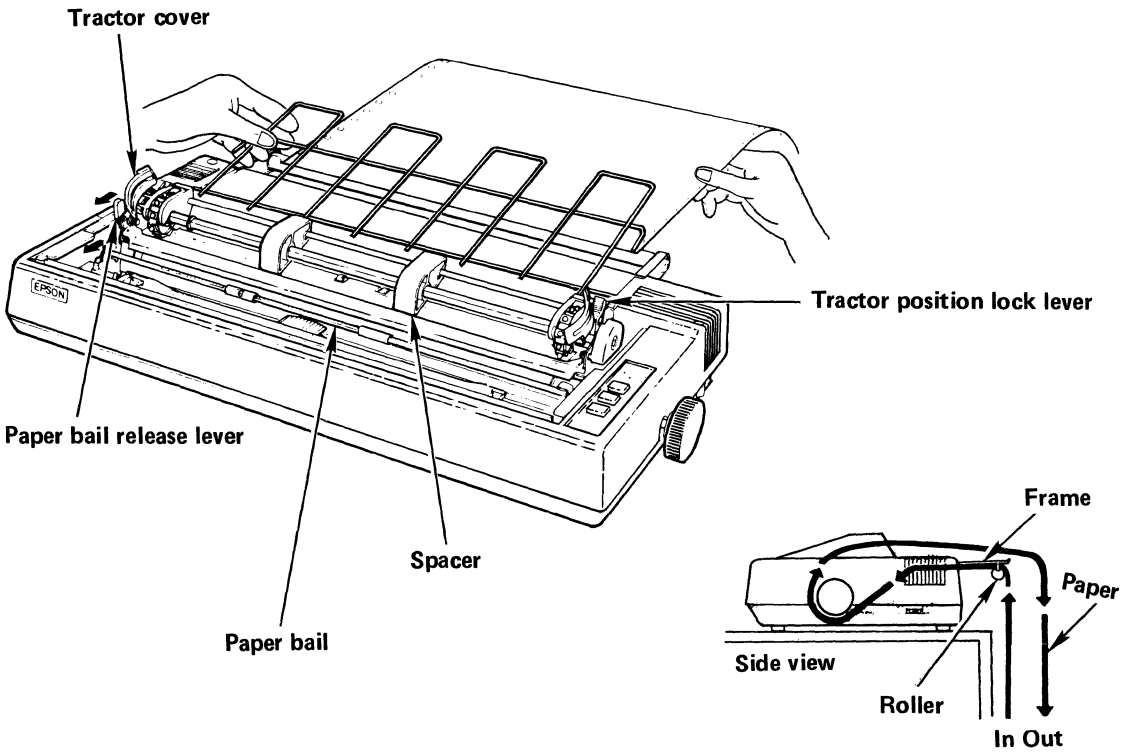


Figure 1-17

The printer moves paper forward only, and never looks back. If we must turn the roller back manually, it helps to pull lightly on the paper. It is possible to move the paper with the roller knob when the power is on, but EPSON recommends that we exercise this option with moderation.

Set the paper so a perforation between sheets is positioned just slightly below the top of the ribbon (Figure 1-18).

Adjusting for Paper Thickness

MX printers can print on all sorts of preprinted forms having multiple copies and carbons, as well as ordinary printer paper. The 7 position thickness control lever (Figure 1-18) moves the print head closer to or farther from the paper, changing the print quality somewhat. For ordinary single thickness paper, begin with the control lever at about the middle position. For multiple copy forms, pull it toward

the front of the printer. Push the lever towards the rear of the printer to produce slightly darker print.

Maximum paper thickness is 0.3 mm or 0.012 inches.

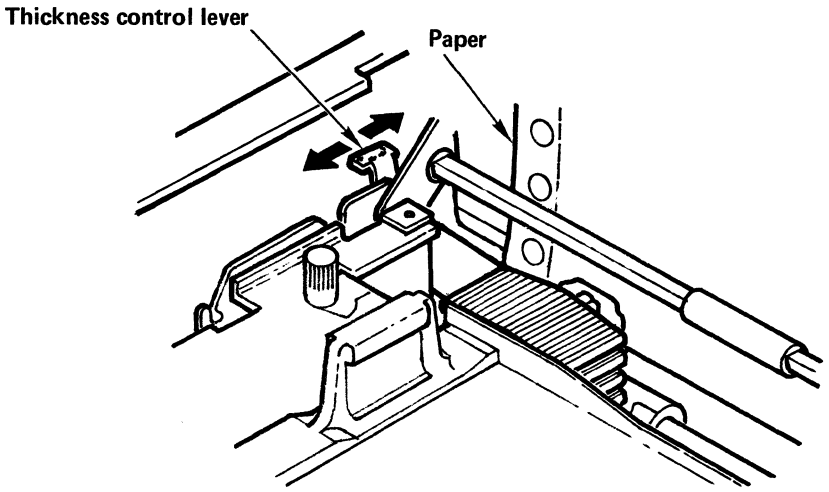


Figure 1-18

Plugging it in

One more little detail and we'll be off and running.

Printers sold in the U.S. and Canada are designed for a standard 120V 60 Hz outlet, and have a 3-wire grounding plug. **Do not attempt to defeat the grounding.** When a proper outlet has been provided, check that the POWER switch on the right-hand side (Figure 1-11) is OFF, then plug in the cord!

At Last

The big moment is here.

Turn it on!

Mmmm! Look at that! A little start-up sound, 3 green lights, and it just sits there waiting. The old days were never like this.

Where is all the motor noise? How about some big levers, lights, and alarms? Total silence, or the quiet purr of a fan. High technology in a modest little package!

If we did it right, the POWER, READY, and ON LINE lights should be lit. If we put the paper in wrong, the PAPER OUT light will also be lit, and we'd better try again.

Press the ON LINE button several times. It "toggles" the printer-to-computer cable connection, alternately hooking us up and disconnecting us from the computer (which of course we haven't hooked up, yet). Finish up your fiddling with the printer ON LINE and READY.

Now try pushing the FF (form feed) and LF (line feed) buttons.

Nothing! What's wrong? Nothing. We have to "toggle" the printer "OFF LINE" to use them. Oh.

Press the ON LINE button so the READY light goes out. Now press FF.

Good grief — look at that paper go. Now look where the paper stopped, at the exact top of the next page. Unless (until) we change it otherwise, the FF button will always advance the paper to the top of the next form (or piece of paper).

Now tap the LF button.

It advanced the paper 1/6 inch. Unless (until) we tell the printer otherwise, it prints 6 lines to an inch, and 66 lines to a page. Now hold the LF button down for a few seconds.

How about that? Both manual and automatic transmission.

Finally, press the FF button again.

Where did the paper stop this time? Yep. At the top of the next form.

And that just about covers all the obvious buttons and switches on the printer. Learning how to use it is not very formidable at all, if we take our time.

The Final Checkout

The final part of this checkout takes only seconds. It's an important thing to do, and shows us many of the print characters that are available.

Turn the printer OFF. Load the printer with wide paper, at least 15 inches on the MX-100. Hold the LF button down and turn the printer back on at the same time. GO!

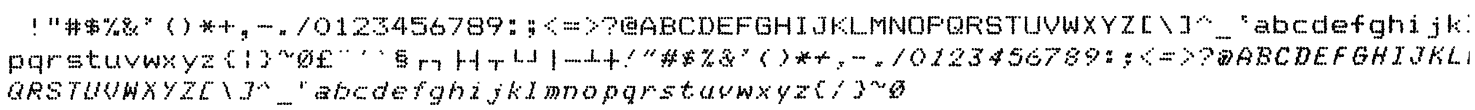


Figure 1-19

Wow! Look at that son-of-a-gun go. The test will continue as long as we hold down the LF button. Watch the head printing in both directions. Move the paper bail out of the way to see better, but keep your fingers (and hair) out of the hardware! When you've seen enough, release the LF button. The test will stop when it's ready.

We're looking at a printout of all of the characters in the printer's memory. We see the normal printer symbols as well as an italic character set and some line graphics characters. How about that!

Users who upgraded ROM chips from the early MX-80 will notice the absence of TRS-80 graphic characters. Don't fret! The new printer can go one giant step further — high resolution graphics. We'll get to it in a later chapter.

Our printer spits out characters at a pretty nice clip, 80 CPS (characters per second). Did you count them as they whizzed by?

When the self test is finished, we are automatically switched ON LINE to the computer cable.

Memory Test

How's **your** memory? Do you remember how to advance the paper to the top of the next form? Do it.

Had to go **off line**, right? Tear off the test run using the edge of the lid as a guide. Hold the lid down with one hand and pull straight up on either side of the paper. It's not easy to make a smooth tear with the wide paper, but we can do it with practice.

Hang this trophy up on the wall as a souvenir.

Whew!

That's enough for this chapter. Take a short walk to vent the exhilaration. In the next chapter we'll hook this marvel to our computer, turn IT on, and see what happens.

Chapter 2

Send it a Message

Your EPSON printer is smart. It knows how to follow instructions.

Any Code Devised by Man Can be Broken by Man

Many instructions are sent to the printer. Every letter, number, and other character travels from the computer through the printer cable in the form of a code made up of numbers. We know it as the ASCII code (American Standard Code for Information Interchange), pronounced ASK-Key.

Let's take a quick glance at Appendix A to refresh our memory. The decimal number 65 stands for the letter A, etc.

The ASCII code numbers for upper case letters, numbers, and punctuation are pretty well standardized around the world. Unfortunately, the remaining code numbers are not uniformly standardized, even among computer manufacturers within a single country.

Besides letters and numbers, we can also send special **control codes** to make the printer print **narrow** letters, **wide** letters, **bold** letters, and do many other things. To take advantage of all these features, however, the computer has to be able to **send** these special code numbers.

As we'll see, each with his own computer, not all computers can send all codes. With printer technology advancing faster than computer technology, the computer has, at least temporarily, replaced the printer as the weak link in the system.

The Code Courier

In many cases, the easiest way to send special codes is to build them into the computer program along with its regular codes for letters and numbers. We can do this easily using programs written in the BASIC (or other) computer language. Because of its simplicity and overwhelmingly popularity, we will do all our demonstrating and learning here in BASIC.

We can also send these special codes at the computer system "command level" **before** running a program. A program may even contain codes to change earlier codes, allowing us to print things the way we want them, when we want them.

The route to success in all BASIC cases is via CHR$.

The Silicon Gulch Connection?

CAUTION: Make all connections with everything turned off!

Every computer is different. If we had time, there might be a different printer manual for every computer manufactured. Unfortunately, there are just too many computers already — with more being added each month.

So, in order to provide a semi-coherent explanation of the printer's capabilities, and to demonstrate how to use it, complete with examples, we opted to write to the world's two most popular computers — the Apple and the TRS-80. Don't get mad if you have something else. Every attempt will be made to alert you to significant variations when appropriate.

If your computer is a "Discount House Digital Delight" version 84.7, head straight for its operating manual for printer interfacing instructions. If EPSON provides a special interfacing kit for your computer, read and follow those special instructions.

Sample interfacing instructions are shown below for the TRS-80 and the Apple. Others are shown in special appendices.

TRS-80 and its many "work-alikes": The computer end of the printer cable hooks to the printer port on the computer. In the case of the Model III TRS-80, this port is located on the right rear of the underside of the computer. Be sure the cable is positioned so the wires lead toward the rear of the computer, as shown in Figure 2-1.

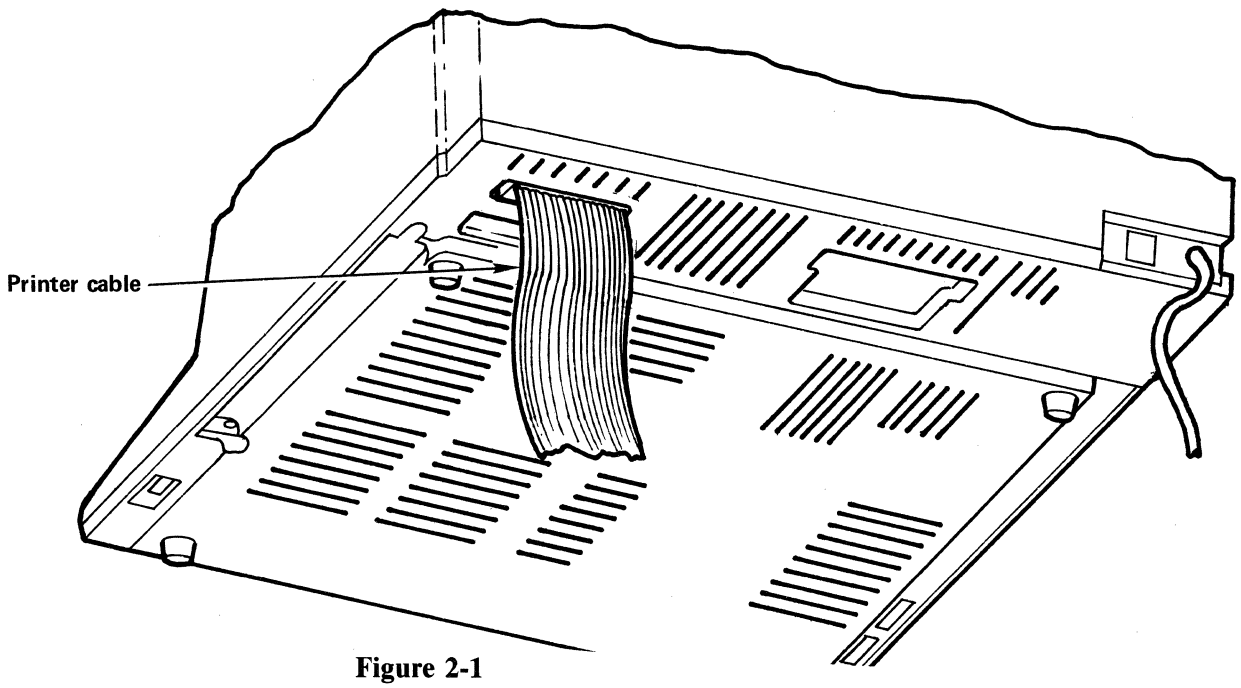


Figure 2-1

Apple: Apple users first verify that the power is OFF, then place the EPSON parallel interface card in slot 1 (not 0) of the computer's mother board. Connect the printer cable as shown in Figure 2-2.

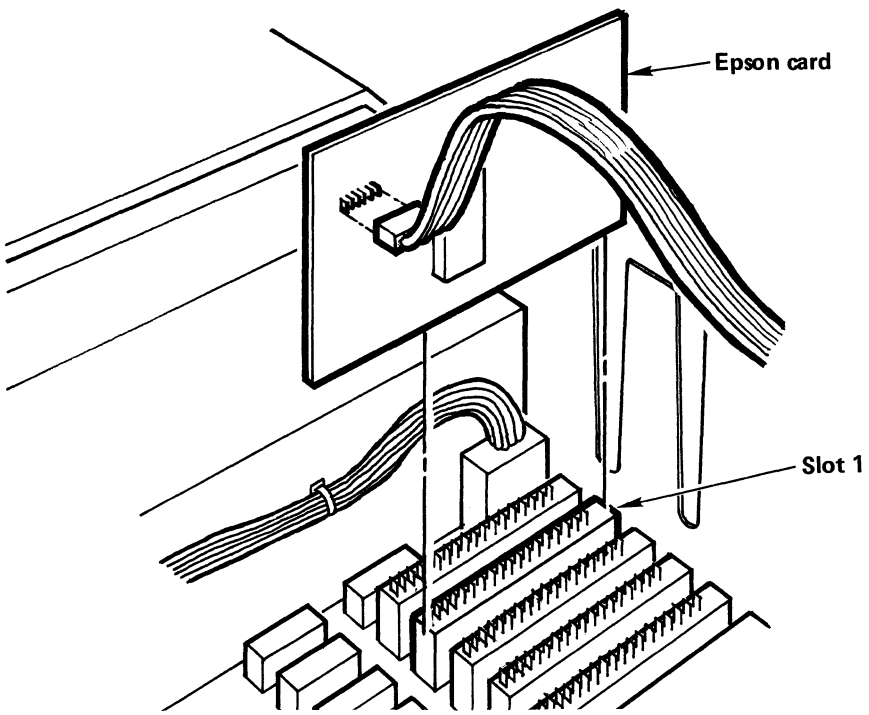


Figure 2-2

If your computer isn't one of the above, head for Appendices H, I, and K for more information.

BASIC is BASIC is BASIC . . . ?

The TRS-80, Atari, Microsoft BASIC for CP/M, System 80, PMC-80, LNW-80, ZX81, and many other computers use the words PRINT to send information to the screen, and LPRINT to send it to the printer.

Computers like the Commodore PET and Atari use a variation of the LPRINT scheme. They require that a file number, say #1, be assigned to the output device (printer). "PRINT#1", for example, directs the information to the printer, and a simple PRINT sends it to the screen. (Atari can use both LPRINT and PRINT#; see Appendix H.)

Still other BASIC dialects like Applesoft, and CBASIC for CP/M, use PRINT to send information to either the screen or both screen and printer. PR#1 and LPRINTER commands direct PRINTing to the printer, while PR#0 and CONSOLE send it only to the screen. The last output command executed determines where the information goes. It's not quite as cumbersome as it sounds.

For Example

Here are some real-live examples of how different computers send information to the screen and the printer. Study them all (to broaden the mind). Select the one that's used by your computer and give it special attention. Don't bother to type in and RUN them, unless the urge is overwhelming.

TRS-80: (Also see Appendix G)

```
10 CLS
20 INPUT "ENTER YOUR NAME "; N$
30 PRINT "START DEMO"
40 LPRINT "PRINTER DEMO FOR "; N$
50 PRINT "DEMO COMPLETED"
```

Atari: (Also see Appendix H)

```
10 GRAPHICS 0
20 DIM N$ (50)
30 PRINT "ENTER YOUR NAME ";
40 INPUT N$
50 PRINT "START DEMO"
60 OPEN #1,8,0,"P"
70 PRINT #1; "PRINTER DEMO FOR "; N$
80 CLOSE #1
90 PRINT "DEMO COMPLETED"
```

Commodore: (Also see Appendix I)

```
10 INPUT "ENTER YOUR NAME "; N$
20 PRINT "START DEMO"
30 OPEN 1,6
40 PRINT#1, "PRINTER DEMO FOR " N$
50 PRINT "DEMO COMPLETED"
```

Apple: (Also see Appendix J)

```
10 HOME
20 INPUT "ENTER YOUR NAME "; N$
30 PRINT "START DEMO"
40 PR#1
50 PRINT "PRINTER DEMO FOR "; N$
60 PR#0
70 PRINT "DEMO COMPLETED"
```

Each of these programs prints something on the screen while it INPUTs your name from the keyboard, prints on the printer, then prints it on the screen.

Since we can't write a different version of each programming example for every microcomputer on the market, we will use the TRS-80/Microsoft BASIC as the standard for most of the manual.

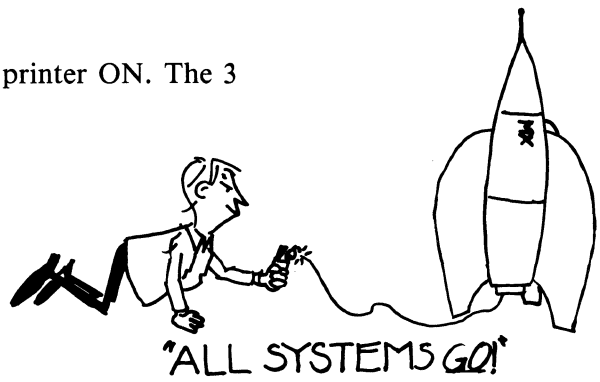
Readers with non-TRS-80 computers, remember to use PRINT or PRINT# instead of LPRINT, as above. You are responsible for opening and closing ports or files as required by your own computer. Refer to Appendices G through K now for more specific help with several popular computers before we start writing programs. If your computer doesn't happen to be specifically included in these appendices, pick one similar to your system as an example to follow. If that fails, go to the computer's instruction manual and try to decode its section on printers.

In the final chapter, we will learn how to use a high-resolution graphics screen dump, and switch to Applesoft BASIC as our Standard. Many additional comments are directed to the Apple.

This two-computer learning approach provides us with a useful comparison between the two major methods of accessing a printer. It also demonstrates how the limitations of a given computer affects our ability to utilize all the power of a printer.

All Systems are "Go"

Fire up the computer, loading in BASIC as usual. Turn the printer ON. The 3 lights, POWER, READY, and ONLINE should be lit.



Type in this WELCOME program, but do not RUN it yet:

(Check with Appendices G - K once more for any special considerations regarding your computer **before** you type it in. Best to learn the special requirements of your computer **now**!)

```
10 LPRINT TAB(12) "GREETINGS FROM THE VERSATILE"
20 LPRINT TAB(21);CHR$(14);"MX-80"          (or MX-100)
30 LPRINT TAB(21) CHR$(15) "BY" CHR$(14) " EPSON" CHR$(20)
CHR$(27) CHR$(83) CHR$(0);
40 LPRINT "(TM)" CHR$(27) CHR$(84) CHR$(18)
50 LPRINT TAB(19) CHR$(27) CHR$(75) CHR$(80) CHR$(8);
60 FOR N=1 TO 80 : LPRINT CHR$(ABS(N-40)+20); : NEXT N
70 LPRINT : LPRINT CHR$(27) CHR$(52);
80 LPRINT TAB(20) CHR$(27) CHR$(45) CHR$(1) "YOU'LL LIKE ME!"
90 LPRINT CHR$(27) CHR$(64)
```

Time Out for Emergency Training

CAUTION: If you make a typing error that causes the program to crash, or look weird, be sure to turn the printer OFF then ON again before running the corrected program. The faulty program may have sent an unwanted code "down the line." It may even have sent something unpatriotic like "don't pay any attention to the computer." The printer will reset itself to normal by simply turning it OFF and ON.

Another way to reset all the software codes to normal is:

```
LPRINT CHR$(27) "@"
```

It won't get us out of every bind, but it's a nice option to have.

It may also be necessary to shut the computer down cold, and start over from the beginning. The printer has its own internal computer, and the two computers talk to each other. If one should decide to throw a temporary snit, we have to get in be-

tween them and cool things down. Doesn't happen often, but as with any computer, a glitch on the power line or a static electricity spark can cause all sorts of heartburn. Best to know how to handle the trouble when it comes.

Operator Now Certified

Now we're prepared to make that program, so RUN.

There it is! We finally strapped these two pieces of high technology into harness and got us a real convoy.

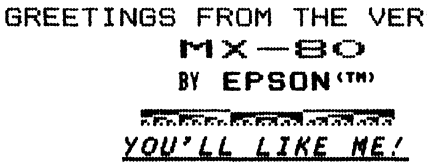


Figure 2-3

RUN it again. This time watch the print head closely.

Did you see it printing in both directions? Did you see the head print the last line twice, to make the print darker? If not, RUN it again and watch more carefully.

Breaker — Breaker!!!

Let's take a short break and a deep breath before charging on to the next chapter, where we'll dissect this program and learn why it does what it did.

Chapter 3

Decoding the Message

It took a lot of doing to print the welcome message. Let's analyze the program a line at a time so we're sure to understand it. Type LLIST to make a hard copy. Non-TRS users refer to Appendices H, I, and J for alternate listing instructions.

Use the LF (line feed) button — remember? — to roll the printout past the lid. Tear it off and keep it handy so we can study the program as we proceed.

You will probably want to SAVE the program on disk or tape to void having to type it in later.

Line by Line

Type NEW to erase the program. We're going to load it back in a line at a time, analyzing it as we go. (An alternate strategy is to load the entire program back in from disk or tape, then DELETE all the lines except line 10. Another way is to make REM LINES OF those lines not being used.)

Line 10 is straightforward. We LPRINTed starting 12 spaces from the left. Type it in and RUN, watching the head action.

```
GREETINGS FROM THE VERSATILE
```

Figure 3-1

Well, that was rather "pedestrian."

Line 20 has 3 LPRINT instructions on the same line, separated by semicolons, which are optional in most versions of BASIC. The first instruction tells the printer to TAB over 21 spaces, the second sends one of the special CHR$ (character string) codes we mentioned earlier, and the third prints "MX-80" (Type "MX-100 if that's what you have.) CHR$(14) stands for "print DOUBLE WIDTH."

Everybody back to look at Appendix B. It contains all the special codes, often called control codes. The ASCII code number we need to print (or do) what we want is shown on the same line with its explanation.

It's important to note that not all codes actually print on either the screen or the printer. Most special, or control, codes don't really PRINT anything, even though we must precede them with our computer's version of LPRINT to push them down the line to the printer. It's just the means to the end.

For example, type the following at the command level:

```
LPRINT CHR$(14);"HELLO THERE"
```

and see the words appear in double width (not double spaced).

Figure 3-2

LLIST the program again. Notice that the double width feature is no longer switched on.

The message: Each time we wish to print something in the double wide mode, we must precede it with control code 14.

When that LINE is finished printing, the double width feature is automatically turned off. In a future chapter, we'll learn another way to turn on DOUBLE WIDTH so it stays on for good, if we want it that way.

Now that we understand line 20, let's add it in and RUN our two-line program.

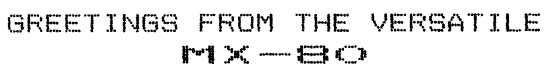


Figure 3-3

Pretty nice — eh? If you wish, do another LLIST to make sure that double width really got switched off.

The programming in lines 30 and 40 is a bit more exotic. We can quickly see that line 30 consists of a tab, two "literal strings," and 6 control characters. The control codes are of course referenced in Appendix B, but are also listed here "in order of appearance."

15 stands for **compressed character** mode

14 stands for **double width** mode

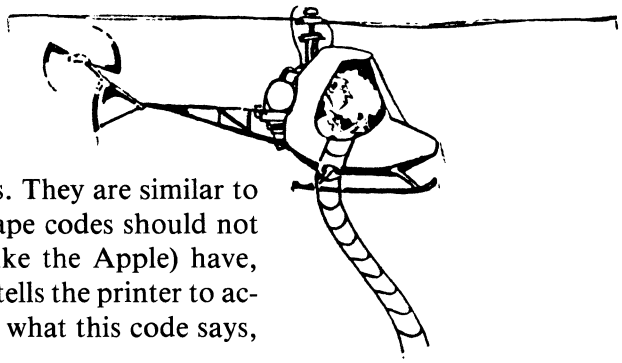
20 stands for turn off **double width** mode

27 stands for ESCAPE

83 is ASCII for the letter "S", code for either superscript or subscript mode.

0 chooses superscript mode.

which forces us to GOSUB to another topic.



The Great Escape

EPSON MX printers recognize several so called escape codes. They are similar to (and really part of) the control codes under discussion. Escape codes should not be confused with the escape key which some computers (like the Apple) have, though they are shirt-tail relatives. Receipt of an escape code tells the printer to accept what follows as a special instruction code. It should **do** what this code says, not PRINT its ASCII character equivalent.

We frequently send the special code CHR$(27) (which means ESCAPE) down the line immediately before another code that requires that ESCAPE. These special code clusters are logically called escape codes. We can either build them into a program or send them from the command level, just as we did before with the simple control codes. We'll learn how to use each escape code as it is needed.

Our First Escape Code

<ESC>"S"# specifies the superscript or subscript mode, depending on the value of #. If # is a zero, we print superscripts. All other values of # make the MX print subscripts. A superscript is "a little character above the line." A subscript is "a little character below the line."

In our case, the printer stayed in the superscript mode until it received an <ESC>"T". That's what the middle part of line 40 is all about. CHR$(27) sends the escape code, and the 84 in CHR$(84) is the ASCII code for T.

Return

RETURNing from our GOSUB, add in lines 30 and 40 and RUN.

```
GREETINGS FROM THE VERSATILE
MX-80
BY EPSON(TM)
```

Figure 3-4

Pretty impressive. Just like they do it in the big city!

The letters "BY" in line 30 are **compressed** to 17.16 characters per inch. In compressed mode, 233 characters per line (CPL) are printed on wide paper (15 inches), or 132 CPL on narrow paper (8 inches). CHR$(15) activates the mode. Regular spacing prints 10 characters per inch, which amounts to 136 CPL on wide paper or 80 CPL on narrow paper. More on that subject in a few chapters.

In the double width mode, characters are printed twice as wide as they would ordinarily, and the actual width depends on whether we are in the compressed or regular mode. Since CHR$(15) put us into compressed mode, and CHR$(14) adds double width, the word EPSON was printed in double width/compresed. In other words, CHR$(14) doubled the width of our compressed characters.

Stop now and think this idea through! It's really very easy, even if some of the words are similar. Like having a 2 speed rear axle; it doubles our options.

Finally, CHR$(20) turns off the double width feature after EPSON is printed.

Comparing our Options:

		MX-100	MX-80
Normal width	=	136 CPL	80 CPL
Normal width/compressed	=	233 CPL	132 CPL
Double width	=	68 CPL	40 CPL
Double width/compressed	=	116 CPL	66 CPL

Double strike — Unrelated to the above.

Pretty simple when laid out this way, right? These are really nice options.

Double Strike

In addition, the print head is now doing its thing twice. In fact, this program is so impressive, let's RUN it again. (C'mon, do it! Stay with the program.)

Figure 3-5

Watch the print head carefully. Notice that each line of the output is printed twice. This is called double strike. It is a left-over "benefit" of being in the superscript mode.

We normally enter double strike (not double width) mode with <ESC>"G"

Example: `LPRINT CHR$(27) CHR$(71)`

Everything on a program line following an <ESC>"G" is printed twice by the print head. Before the second time it is printed, however, the paper is rolled up just 1/216th of an inch to help fill in the spaces between the dots printed on the first pass. The result looks very bold and solid, almost like it came from a regular typewriter. Pretty clever.

Further, once the "G" switch is thrown (ASCII code number 71), everything that follows is double printed until an <ESC>"H" message is sent, returning printing to normal. (The ASCII code number for "H" is 72.)

Well, we didn't send an <ESC>"G", the superscript mode dumped double strike in our laps. Since double strike gives a nice quality print, let's leave it on for now and move on to other things. We know how to turn it off with <ESC>"H" when we want to.

Since things are going so swimmingly, let's see if we can dig up some mischief. Delete CHR$(18) from line 40:

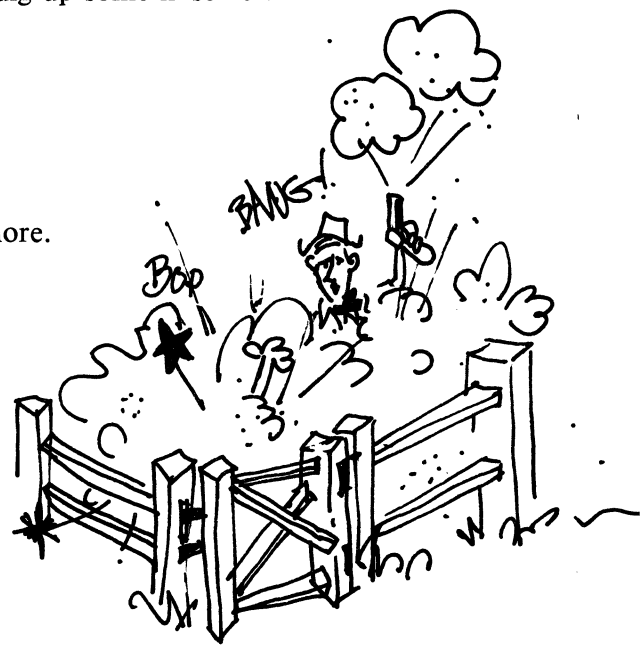
```
40 LPRINT "(TM)" CHR$(27) CHR$(84)
```

and RUN.

Same results, so what's all the fuss about? RUN it once more.

```
GREETINGS FROM THE VERSATILE
      MX-80
      BY EPSON(TM)
```

Figure 3-6



Trouble at the Old Corral

Oh-oh. Now what did we do? You might know, just when things were going so well. Looks cute though, doesn't it?

Remember, whenever a "switch" is thrown by sending a code, it stays thrown until switched back to normal. (The few exceptions to this rule will be noted as each is covered.) In this case, Code 15 was sent, and all characters were compressed (233/132 CPL). It takes a Code 18 to switch back to normal (136/80 CPL). (Everybody gallup back to Appendix B again. We'll know it well before we're done.) CHR$(18) stands for: turn off compressed characters.

Compressed mode also affects the TABs. If we are printing 233 (132) letters and numbers per line, the width of each TAB space is also compressed, compared to the normal 136 (80) CPL. That's why the lines on the paper are shifted to the left. Everything is working as it should!

Super Reset

So how do we fix it? Replace the CHR$(18) back in line 40? Good answer. But since we want to introduce the super high powered software reset switch here, let's not.

At the command level, type:

```
LPRINT CHR$(27) CHR$(64)
```

and LLIST the program.

Hot digity! No more compressed mode or double strike. Now that's a powerful escape code!

<ESC>"@" turns off all special modes activated since the printer was turned on (and resets the top of form, which we will meet in Chapter 5). The result is just like turning the power switch OFF then ON again, but it's done in software. The ASCII number for @ is 64.

We should add this super reset code to the end of every program to purify the printer when it's done. Left-over printer codes are as bad as last night's left-over beer. Add:

```
90 LPRINT CHR$(27) CHR$(64)
```

And, to restore our original program, change line 40 back to:

```
40 LPRINT "(TM)" CHR$(27) CHR$(84) CHR$(18)
```

Never Look Back — It Might be Gaining

Lines 50 and 60 might be the toughest, but they're not bad if we hang in there.

Line 50 is another escape sequence. This time <ESC>CHR$(75), better known as <ESC>"K", kicks us into graphics mode. The two codes that follow, CHR$(80) and CHR$(8), tell the printer to expect 80 graphics code numbers.

The trailing semicolon holds the print head on the same print line while it waits for the promised graphics code numbers. After the specified number of characters (80 in this case) have been received, the printer automatically kicks out of the graphics mode, back to normal printing.

(Don't worry about the CHR$(8). It's explained in detail in a later chapter dealing just with graphics.)

Line 60 uses a loop to calculate and send the 80 ASCII code numbers to the printer. The printer obviously knows what to do with them, even though we don't, at this point. Graphics is sufficiently complicated that learning it takes up entire chapters, later. Don't get hung up on these little details now. Keep charging!

Michaelangelo Should See This

Let's add lines 50 and 60, RUN it, and watch.

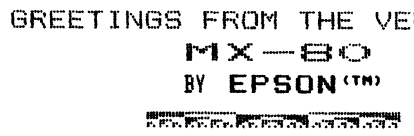


Figure 3-7

Beautiful, isn't it?

Two Easy Pieces

Only 2 more easy lines. It's downhill now.

Line 70 is a multiple statement line. The first LPRINT just inserts an LF, undoing the trailing semicolon in line 60.

```
70 LPRINT: LPRINT CHR$(20) CHR$(27) CHR$(52);
```

Figure 3-8

The rest of line 70:

```
CHR$(27) CHR$(52);
```

activates a second type font — italics. <ESC>"4" is the switch that turns it on. It's a nice type font to use for emphasis or decoration.

In addition, we underlined the italics. Underlining is activated by <ESC> "-" CHR$(1). It can be turned off at any point on a line by <ESC> "-" CHR$(0). We can start and stop underlining at will.

Add lines 70 and 80 and RUN, watching the print head.

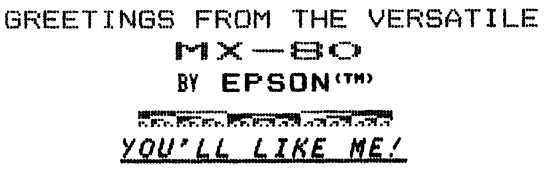


Figure 3-9

Yep! You'll like me! is normal size thanks to the CHR$(18) in line 40. It also printed in italics and was underlined, but why did it print twice?

It's that old double strike again (residue from the superscript mode). We're in double strike mode until line 90 is actually executed and brings an instant halt to **all** special modes.

Yes, there is a lot to keep track of since it comes so fast, but this is a powerful printer, and in a few hours you will be its master!

Testing, Testing

To make sure we appreciate the power of line 90, let's:

DELETE 90

and RUN the program.

No difference yet, but we just ran the program without our instant code killer. Any codes left over at the end of the program will still be active. RUN it one more time.

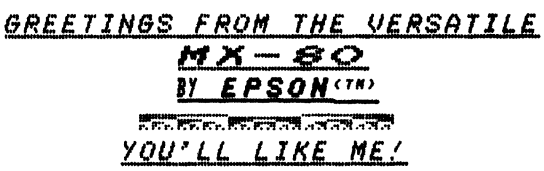


Figure 3-10

Sure makes a difference. A program listing would look the same way. A little carelessness goes a long way!

The common-sense principle to follow in all our programming with this sophisticated printer (except when teaching/learning) is to "never end a program without turning off all special codes."

Phrased differently: Clean up your own mess!

Add line 90:

```
90 LPRINT CHR$(27) CHR$(64)
```

Figure 3-11

. . . and RUN, watching the print head.

What happened? Wasn't it supposed to shut off the codes? But it did shut them off, **after** the entire program had been printed. Think of a way to prove it.

An easy way is to just LLIST the program.

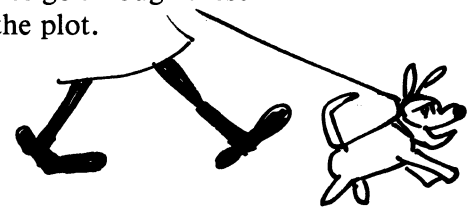
```
10 LPRINT TAB(12); "GREETINGS FROM THE VERSATILE"
20 LPRINT TAB(21); CHR$(14); "MX-80"
30 LPRINTTAB(21)CHR$(15)"BY"CHR$(14)" EPSON"CHR$(20)CHR$(27)CHR$(83)CHR$(0);
40 LPRINT "(TM)" CHR$(27) CHR$(84) CHR$(18)
50 LPRINT TAB(19) CHR$(27) CHR$(75) CHR$(80) CHR$(8);
60 FOR N=1 TO 80: LPRINT CHR$(ABS(N-40)+20);: NEXT N
70 LPRINT: LPRINT CHR$(20) CHR$(27) CHR$(52);
80 LPRINT TAB(21) CHR$(27) CHR$(45) CHR$(1) "YOU'LL LIKE ME!"
90 LPRINT CHR$(27) CHR$(64)
```

Figure 3-12

Just like the book said!

Milestone

This has been a feature-packed chapter, but it gave us a good introduction to several powerful capabilities we can put to work. Better take a good break and walk the dog before tackling Chapter 4. It wouldn't hurt at all to go through these last 2 chapters again before moving on, now that we know the plot.



Summary of Print Command Codes Learned So Far

CHR$(14) — Turns ON double width characters. (Turns OFF at the end of the line.)

CHR$(15) — Turns ON compressed character mode. (Stays ON until turned OFF.)

CHR$(18) — Turns OFF compressed character mode.

CHR$(20) — Turns OFF double width characters before the end of the line.

Special 'Escape' Codes

CHR$(27) — The ASCII symbol for "Escape." (Special code used with letter codes.)

<ESC>CHR$(45) — ASCII number for the symbol "-". (Turns OFF and ON underline mode.)

<ESC>CHR$(52) — ASCII number for the number "4". (Turns ON italics character set.)

<ESC>CHR$(64) — ASCII number for the symbol "@". (Turns OFF ALL special codes. Resets printer to start up condition.)

<ESC>CHR$(71) — ASCII number for the letter "G". (Turns ON double strike.)

<ESC>CHR$(72) — ASCII number for the letter "H". (Turns OFF double strike and Superscript/Subscript mode.)

<ESC>CHR$(75) — ASCII number for the letter "K". (Turns ON the graphics mode.)

<ESC>CHR$(83) — ASCII number for the letter "S". (Turns ON superscript or subscript mode. Leaves double strike mode ON.)

<ESC>CHR$(84) — ASCII number for the letter "T". (Turns OFF superscript or subscript mode. Leaves double strike mode ON.)

Example: LPRINT CHR$(27)CHR$(71)

Turns ON the double strike mode.

Chapter 4

More Print Control Commands

That last chapter was a heavy one, but it gave us a good overview of the printer's major features. In this chapter we'll explore a few of them in more detail, plus learn additional features related to print size and quality.

As before, in BASIC, CHR$ is the magic wand, and we are using a popular version of Microsoft BASIC as our standard. Continue to refer to Appendix B as needed to help maintain perspective on where we're going and what we're doing.

A Cheap Buzz

Type:

`LPRINT CHR$(7)`	Microsoft "Standard" BASIC-type computers
`PR#1`	Apple-type computers
`PRINT CHR$(7)`	

Sorry if that scared the dog! Just couldn't resist it. It really has nothing to do with printing, but is widely incorporated in programs as an alarm to indicate something. A built-in alarm is a standard feature on quality printers.

Sure . . . do it again. We'll wait.

If our computer has a real-time clock, we can use it as a time-is-up alarm. Just include LPRINT CHR$(7) in the program. It also makes a great audio prompter, telling the operator it's time to do something. The applications are virtually endless. Most expensive alarm clock in the place.

Technically, what we heard is called the bell, and ASCII 7 makes it ring. Buzzers have pretty well replaced bells, so we'll usually refer to it as a buzzer, alarm, or something more contemporary.

The bell also rings when there is a no paper situation or some other natural disaster occurs on the MX-80, unless internal switch 1-6 is off.

Back to Printing

In the last chapter, we saw examples of compressed, normal, and double width characters, double strike printing, italics type font, graphics, underlining and

superscripting. At this point it may seem like we're in the grip of uncontrolled confusion. Don't panic. It's controlled.

The way to make these codes our servants is to study them one at a time, exercising lots of patience and performing the many examples. No one said this was going to be easy! Few worthwhile things are. This printer's capabilities are formidable. Decide now to take the time and put in the effort needed to master it. You'll like yourself when we're done.

Varying Character Width

There are three major print features we can vary:

1 — Character width
2 — Type font
3 — Print quality

Let's take them in that order.

Normal Width

Normal width characters are those we are handed when the printer is first turned on, assuming we left switch 1-1 in the OFF position as was suggested. Normal characters are the same width as pica characters on a typewriter — 10 characters per inch. To see what they look like, type:

```
NEW
```

to clear memory of any previous program. Then enter

```
 9 PR#1      (Apple only)
20 LPRINT "NORMAL ";TAB(20);"WIDTH"
```

and RUN.

```
NORMAL              WIDTH
```

Figure 4-1

Just what we expected. Routine characters with a routine TAB out to position 20.

Compressed Width

We already saw compressed characters in the welcome program in Chapter 2. They come 17.16 to the inch. That's really packing 'em tight. We can string out 132 compressed characters on an 8 inch line (MX-80) and 233 on a 13.6 inch line (MX-100).

Compressed mode is activated by CHR$(15) and turned off by CHR$(18).

Add:

```
30 LPRINT CHR$(15);"COMPRESSED ";TAB(20);"WIDTH"
```

and RUN.

```
NORMAL                    WIDTH
COMPRESSED      WIDTH
```

Figure 4-2

Sure enough, CHR$(15) set the printer into compressed mode. It even affected the TAB. As we learned earlier, TAB's effect depends on the print width.

All characters are affected by a change in print width, even blank spaces. That's good to know. We can use it to our advantage.

LLIST the program to see what we have cooking so far. Seems that the compressed mode is still active. Now why don't they just have it shut off at the end of each line?

Well, let us ponder this point for a moment. Compressed mode is the sort of thing we might use hour after hour. A typical application is the printing of a big 132 column sheet of financial reports on 8.5-inch wide paper, instead of the wider 15.5-inch paper. This scheme provides an original copy small enough for reproducing on an ordinary duplicating machine. To have to write the computer program so each line tells the printer to **stay** in compressed mode just wouldn't make good sense.

If we really need lots of columns, we can use compressed mode on wide paper and print a phenomenal 233 columns (MX-100 only). It's perfect for printing the output of electronic "spread sheets" such as VISICALC™.

The bottom line is, if we want to get out of the compressed mode, we have to do it ourselves. We mentioned earlier that CHR$(18) is the shut off switch. Let's try it:

```
LPRINT CHR$(18)
```

and LLIST

```
20 LPRINT "NORMAL ";TAB(20);"WIDTH"
30 LPRINT CHR$(15);"COMPRESSED ";TAB(20);"WIDTH"
```

Figure 4-3

That did the trick. What is the other way to deactivate this and every other mode? That's right — <ESC>"@". The old rock-crusher command. Type

```
50 LPRINT CHR$(27) CHR$(64)
```

then RUN and LLIST

Now we are back on line.

Let's try one more line to make sure we all got the point. Add:

```
40 LPRINT "BACK TO NORMAL ";TAB(20);"WIDTH"
```

and RUN.

Arggggg! Not again! Is the printer trying to tell us something? How about . . . "Turn off each mode when you are done with it!"

Sounds like good advice. Otherwise, these stray modes hang around like a bad penny.

Now, how do we fix it? Well, since the printer is in compressed width at the end of line 30, we must shut it off before line 40 can be normal width. Change line 30 to

```
30 LPRINT CHR$(15);"COMPRESSED ";TAB(20);"WIDTH";CHR$(18)
```

and RUN.

```
NORMAL                WIDTH
COMPRESSED      WIDTH
BACK TO NORMAL        WIDTH
```

Figure 4-4

Can you feel the control we are developing over the printer? Man may yet master his toys . . . er, machines.

Mixing Print Widths on the Same Line

The various print modes we have discussed can even be mixed on the same print line. Change the program to:

```
20 LPRINT "NORMAL ";
30 LPRINT CHR$(15);"COMPRESSED ";CHR$(18);
40 LPRINT "BACK TO NORMAL WIDTH"
50 LPRINT CHR$(27)"@"
```

and RUN.

```
NORMAL COMPRESSED BACK TO NORMAL WIDTH
```

Figure 4-5

The trailing semicolons link the program lines together on one print line.

Although the ability to mix modes is very tempting, a little restraint is in order. Indiscriminate mixing of modes will make it very difficult to keep track of TAB positions. Besides, it is very easy to lose track of which mode we are in. Proceed with caution.

Double Width

Remember, each of the two basic print widths can be easily doubled. CHR$(14) turns doubling on and CHR$(20) turns it off, doubling our selection of print widths:

Double normal — 5 characters per inch

Double compressed — 8.5 characters per inch

plus the original two widths:

Normal — 10 characters per inch

Compressed — 17.16 characters per inch

Double Width Forever

In the last chapter, we promised a way to turn on double width so it stays on.

Add line 10:

```
10 LPRINT CHR$(27)"W"CHR$(1) "DOUBLE WIDTH FOR ALL"
```

Sounds like a campaign promise. RUN it.

```
DOUBLE WIDTH FOR ALL
NORMAL COMPRESSED BACK TO NORMAL WIDTH
```

Figure 4-6

Oh my! It certainly did the trick. Can you imagine the fun we could have playing with these codes? Can't stop to play now; we have to keep the sails full while the winds are favorable. Plenty of time to play later, when we know **all** the cards in the deck.

A final comment about the double width mode. If <ESC>"W"CHR$(1) turns on double width, how do we shut it off when it outlives its usefulness? Change line 10 to:

```
10 LPRINT CHR$(27)"W"CHR$(1) "DOUBLE WIDTH FOR ONE";
   CHR$(27)"W"CHR$(0)
```

TRS Model I use:

```
10 LPRINT CHR$(27)"W"CHR$(1) "DOUBLE WIDTH FOR ONE";:
   POKE 14312,0 : LPRINT
```

and RUN.

```
DOUBLE WIDTH FOR ONE
NORMAL COMPRESSED BACK TO NORMAL WIDTH
```

Figure 4-7

Looks like <ESC>"W"CHR$(0) is the OFF switch.

If too many double width codes make you feel like seeing double, rely on Appendix B.

Note that CHR$(20) does not shut off <ESC>"W"CHR$(1), and <ESC>"W"CHR$(0) does not shut off CHR$(14). Have to keep separate these

two double width modes, one "temporary" and the other "permanent." If we try to mix them, <ESC>"W" has precedence and will prevail.

More Print Width Options?

Well, some zealots may consider the superscript/subscript characters basic character types, but that's going a bit too far. We will wring them out in a later chapter.

Type Fonts

Ready for another doubling of our options? Hang on to the life jackets, 'cuz here goes.

All of the previous character width options work with the standard type font we are all used to. They also all work with the italics type font.

Change line 10 to

```
10 LPRINT CHR$(27)"4";"ITALICS TYPE FONT"
```

and RUN.

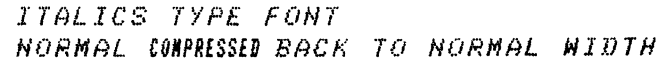


Figure 4-8

Good grief! That gives us **eight** different character sets, and we aren't even warmed up yet.

<ESC>"4" turns on the italic type. <ESC>"5" turns it OFF. Add <ESC>"5" to line 10 to see if it works:

```
10 LPRINT CHR$(27)"4";"ITALICS TYPE FONT";CHR$(27)"5"
```

and RUN.

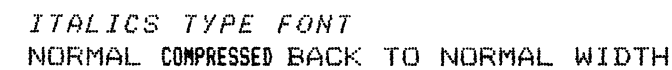


Figure 4-9

Yep, like a charm.

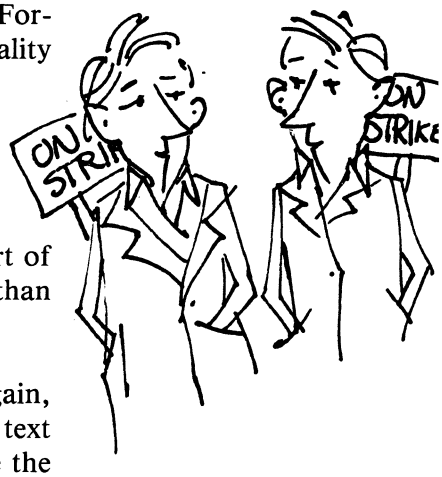
Some folks like the italics character font a lot — so much they would like to have it as their standard. Yes, even when the printer is first turned on. In an effort to satisfy all our fantasies, EPSON included internal switch 1-4 to make it so. Move it to ON if you prefer the italics character font. Don't do it now, though. We're much too busy learning about other features.

So where do all these exotic choices leave us? You guessed it — with still more to look at. Talk about getting our money's worth!

Letter Quality

So far we've looked at 4 different character widths and 2 different type fonts. Another variable we have is the **physical** way in which the characters are printed.

Normal printing is done at 80 characters per second — one pass per print line. Since letters are formed by individual dots, they leave something to be desired for some business correspondence. As the ribbon wears, the print gets lighter. Fortunately, there are two other print modes that give us near letter quality characters.



Double Striking Revisited

Recalling our complex example from the last chapter, we double printed part of the message. Double printing (or double striking) is really more sophisticated than it sounds.

Instead of simply going over what was printed the first time and hitting it again, the printer actually rolls the paper up 1/216th of an inch, then prints the text again. The effect is to fill in much of the space between the dots and make the characters look more solid. Sort of a "poor man's" high-speed typewriter.

Type NEW, then the following program:

```
30 LPRINT TAB(30)"AIN'T SCIENCE GRAND?"
```

then RUN.

```
AIN'T SCIENCE GRAND?
```

Figure 4-10

Add this line:

```
20 LPRINT CHR$(27)"G"
```

and RUN.

```
AIN'T SCIENCE GRAND?
```

Figure 4-11

Nice and bright, isn't it? Almost as good as if typed on an electric typewriter.

Dissecting Line 20

> Line 20 says 'LPRINT <ESC>"G" '. Double striking is also a byproduct of entering the superscript mode. The ASCII number for ESCAPE is 27 and 71 the ASCII number for the upper case letter G. (Everybody back to Appendices A and B.)

Most printers require escape commands to be received via ASCII numbers. With the EPSON MX series printers, the command can be sent either via CHR$(##) or as the letter itself enclosed in quotes. We used <ESC>"G" here since it's shorter to type, and perhaps easier to remember.

Try to memorize these codes as we go along. They are really not very complex once we understand the big picture, and constantly referring to Appendix B is not all that exciting.

Like most other commands, double strike stays ON until cancelled. Its opposite (for single strike) is <ESC>"H". Let's do it:

```
LPRINT CHR$(27)"H"
```

and type LLIST. Check the copy to see that we are really back in the single strike mode.

The Wallbanger Mode

There is one final print "quality" command. It is, in nice company, called the emphasized printing mode. Others have called it "the Wallbanger," "Jackhammer," "Super Whammy," etc. It is a double charged, plug-it-in-the-440V-outlet, sock-it-to-the-paper mode. Change our existing program to read:

```
20 LPRINT CHR$(27)"E"

30 LPRINT TAB(30)" PUT ON YOUR HARDHAT!"

40 LPRINT CHR$(27)"F"
```

and RUN.

```
PUT ON YOUR HARDHAT!
```

Figure 4-12

Well, that wasn't as violent as expected, but look how nice the print looks. <ESC>"E" turned it ON and <ESC>"F" turned it OFF. A quick LLIST will confirm that it's off.

You asked, "Wonder if I could print just **part** of a line in emphasized mode?" Let's try it! Change the program to:

```
10 LPRINT TAB(15)"NORMAL PRINT";

20 LPRINT CHR$(27)"E";

30 LPRINT TAB(30)" PUT ON YOUR HARDHAT!";

40 LPRINT CHR$(27)"F";

60 LPRINT " BACK TO NORMAL"
```

The trailing semicolons will make all five program lines print on one line of the printer. Now RUN to see how it works.

```
NORMAL PRINT    PUT ON YOUR HARDHAT! BACK TO NORMAL
```

Figure 4-13

Works like all the other commands so far. We can slip in and out of the emphasized mode at will. Just have to know where the ON and OFF switches are.

In addition, we can switch in this mode **permanently** by turning switch 1-5 ON. (Only do this with the printer OFF.) Don't do it now. Still more to learn first.

The Ultra Machine

How about combining the two print quality commands to give us the **ultra** in high quality dot-matrix printing? Let's put the printer in both the Wallbanger and double strike modes at the same time and see what happens. We call this combination double emphasized mode. Add lines 10 and 50 to incorporate double strike, and change the rest of the program to:

```
10 LPRINT CHR$(27)"G"

20 LPRINT CHR$(27)"E"
```

```
30 LPRINT TAB(30)"PUT ON YOUR HELMET AND GOGGLES"
40 LPRINT CHR$(27)"F"
50 LPRINT CHR$(27)"H"
60 LPRINT TAB(30)" BACK TO NORMAL"
```

and RUN.

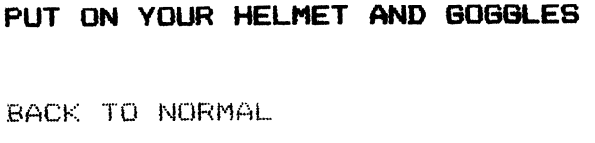


Figure 4-14

That's about as good as it's going to get, and that's not too shabby.

Ultra mode produces very good letter quality print, but takes its toll on ribbons.

Caveat Printer

There's one minor restriction on emphasized printing. It cannot be used in the compressed mode or with superscript/subscript characters. If we try, the printer automatically switches to normal width characters printed in emphasized mode.

And it's only logical, if we think about it. The compressed and script modes have the dots spaced very close together. If we turn all the engines up to Warp 7 it just might blow the print needles right through the paper.

Oh, yes. Emphasized printing is only half as fast as normal printing. Don't want to overheat the engines you know. If we then add double strike, we cut the speed in half again. Quality always has its price!

Code Summary

7 = Bell
14 = Double width. Turns off at end of line.
15 = Compressed mode on. (Can't mix with emphasized.)
18 = Cancel compressed mode.
20 = Cancel double width.
<ESC>"4" = Italics on.
<ESC>"5" = Italics off.

<ESC>"@" = Clear all special modes (the "biggie").
<ESC>"E" = Emphasized mode on. (Can't mix with compressed.)
<ESC>"F" = Emphasized mode off.
<ESC>"G" = Double strike on.
<ESC>"H" = Double strike off.
<ESC>"W"CHR$(1) — Double width on. Stays on until turned off.
<ESC>"W"CHR$(0) — Turns off double width initiated by <ESC>"W".

Chapter 5

Advanced Printing and Endless Forms Control

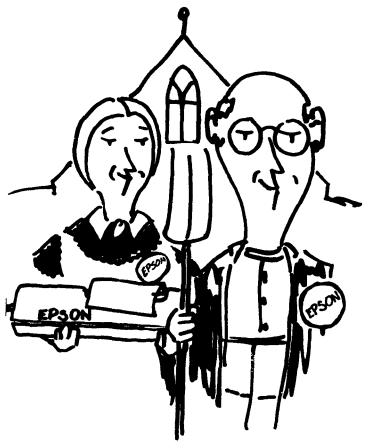
Establishing the Facts

The standard letter-size piece of paper in North America is 11 inches long. A maximum of 10 inches is usually used for printing, the remaining inch divided between the top and bottom margins. Legal-size paper is 14 inches long, of which no more than 13 inches is used.

Regular-width paper is 8½ inches wide, with no more than 8 inches used for printing. Wide printing paper, for which the MX-100 is designed, is available up to 15½ inches wide with no more than 13.6 inches used for printing.

Standard typewriter or printer spacing allows 6 lines per inch, or 66 lines per 11-inch page (78 lines for a legal-length page). A new line is started each 1/6 inch.

As we've already seen, normal character spacing is 10 per inch, to a maximum of 80 characters per line on the MX-80 (136 on the MX-100). In compressed mode (control code 15) we can print 132 characters per line on an MX-80 (233 on MX-100), or about 17 characters per inch. Double width, of course, doubles the width of the characters and halves the number of characters per line.



Having reviewed and documented these dull facts and traditions, we're pleased to learn that with simple software commands we can change many of these standards, customizing them for our needs.

So Much for Tradition

Load the printer with plenty of paper as we are going to use a pile of it. Chalk the cost up as tuition. Let's also leave the lid off since we'll be peeking at the printing frequently.

Turn off the printer. MX-80 F/T and MX-100 users adjust the paper so the perforation between two sheets is at the top of the ribbon. Then turn the roller knob back slightly so the bottom edge of the pin hole underneath the perforation is touching the metal strip that's against the roller (Figure 5-1).

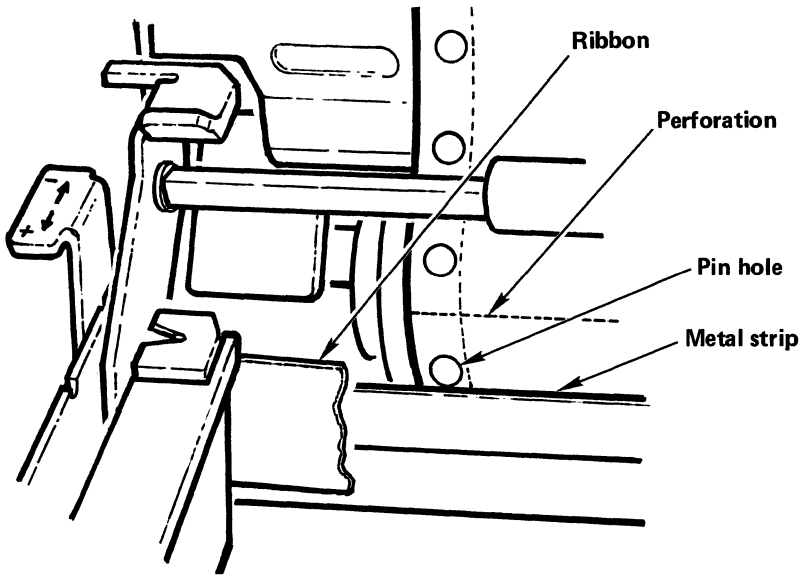


Figure 5-1

MX-80 users line up the perforation with the scribe mark on the platen (Figure 5-2).

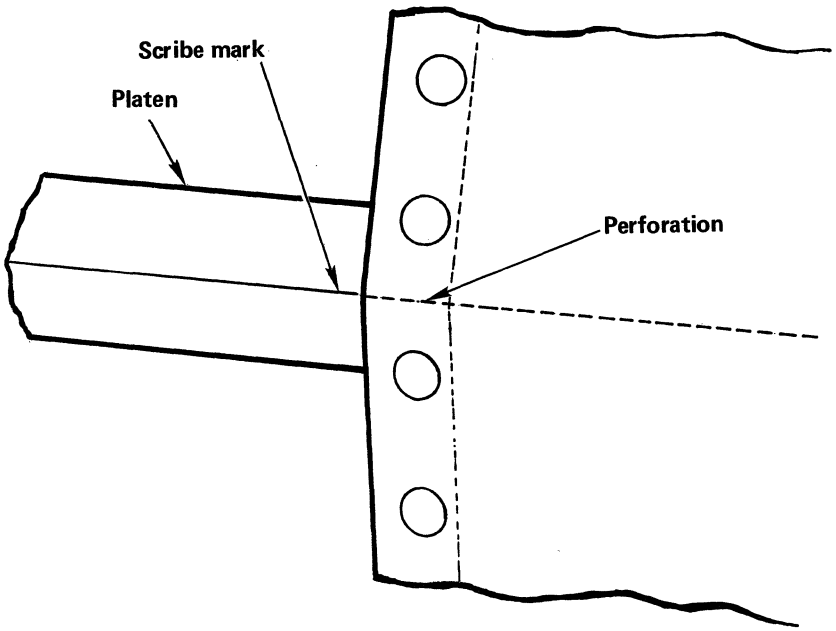


Figure 5-2

It will take a little practice to get the paper positioning perfect. We need to be able to do it right, especially for matching up with lines and boxes on preprinted forms such as invoices, bills of sale, purchase orders, and the like.

Some users find it easier to position the paper in the ball park, then finish positioning by reference to the match mark on the pin feed holder, one of the tractor pins, something else that moves with the paper, or some point over which the paper travels. Practice will make perfect.

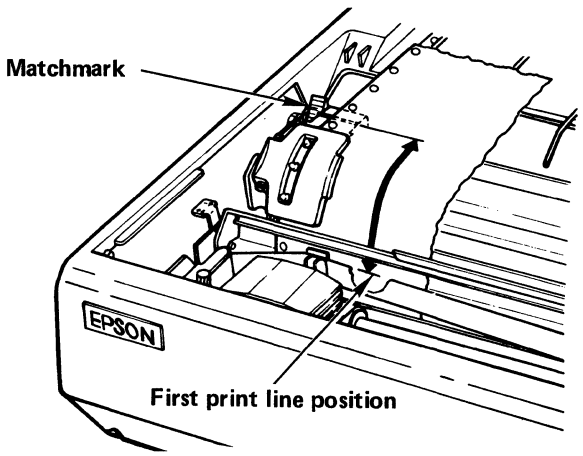


Figure 5-3

Once the paper is in position, turn the printer ON again. This electronically sets the top of form memory, known in the biz as TOF, at the current paper position. Despite what may follow, the printer always remembers where the top of the next sheet of paper is, even if we don't. It has a memory like an elephant. In fact, since the printer cannot "sense" the paper perforations, if we set the TOF incorrectly, the printer will flawlessly go to that same **wrong** position on succeeding sheets. It is up to us to position the paper correctly.

Type in this new program:

```
9 PR# 1                (Apple only)

20 FOR N = 1 TO 66

30  LPRINT N           (Apple = PRINT N)

40 NEXT N

49 PR# 0               (Apple only)
```

and RUN.

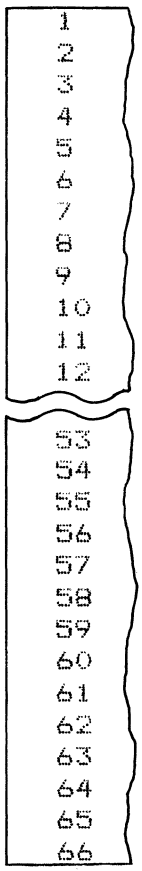


Figure 5-4

Well, that was singularly unexciting. What did it tell us? Where is the top of the next form?

1. It told us that the paper length is exactly 66 rows. No more and no less.
2. Since the numbers all line up in a nice column, and there's no vertical spacing between them, the printer does not automatically insert margins at the top or bottom of a page.

The Form Feed — From the Outside

With the paper set where we left it, let's send a form feed (FF) command, using either ASCII code 12 or 140. (Note the duplicate set of control codes in Appendix

A. Code 12 works on some computers, but it's unpredictable on others. Some computers insist on our adding + 128 to the lower ASCII number, but more about that **computer** problem later.) We'll use 140 since it seems to be the most reliable. Type:

```
LPRINT CHR$(140)
```

Good grief! The train overshot the station! Doesn't life have enough trials without this?

Apparently not. An external FF command acts the same as the FF button on the printer, with one extremely important exception. Since we have to push commands down the line from BASIC with an LPRINT, and each LPRINT automatically sends a CR and an LF (over which we have no control), each time we send an LPRINT we toss an extra LF into the soup. Terrific!

So Now What Do We Do?

Well, the problem isn't going away, so we'll confront it head on.

First, to console ourselves a bit, let's go back to the printer, push it OFF LINE, push FF, and see what happens.

Yep, the paper went right back to its proper place at the top of the next form. (Yes, yes, this is using up paper. Time is more valuable than the paper, so forget it.)

The point of this exercise is:

1. With every LPRINT comes an LF.
2. The printer counts every LF and knows where the paper is at all times (even if we don't).

The Form Feed — In BASIC Software

Let's modify our resident program to print only 10 lines, then have it automatically roll the paper up to the top of the next form. Change it to read:

```
9 PR# 1                    (Apple)
20 FOR N = 1 TO 10
30  LPRINT N
```

```
40 NEXT N
45 LPRINT CHR$(140)   (or 12)
49 PR# 0                    (Apple)
```

Switch the printer back ON LINE

and RUN.

Curses! (the villain said), there's that extra LF again. (Did you look at it?) We overshot the top of the next form.

Well, it's back to the printer buttons again. OFF LINE then FF will get us to the top of the next sheet.

What went wrong? All we added was a simple FF in line 45. Where did that extra line feed come from? Hmmmm? (Knit 1, Pearl 2.)

That's right! Every LPRINT statement automatically sends an LF to the printer. We forgot to include a trailing semicolon in line 45 to suppress it.

Sounds simple enough. Let's change line 45 to:

```
45 LPRINT CHR$(140);
```

and RUN again.

Time Out for Controlled Confusion

We are genuinely on the horns of a dilemma.

Atari users are wondering what's causing the dilemma. The semicolon at the end of line 45 padded it, expanding it to 40 characters. Therefore, we should delete the semicolon from the above program, and RUN again. Don't forget to put it back in for our next adventure!

Since the Apple PR#0 disconnects the printer before the final LF from line 45 is executed, Apple users are seeing different results than everyone else and are wondering what all the shouting is about. Apple users replace PR#0 in line 49 with END, then RUN and see what happens.

Get the idea, now? It overshot by one line again.

Let's all read the following Philosophy carefully. It is absolutely vital that we understand the concept of the endless loop to use any printer for serious business-type applications.

A Little Homespun Philosophy

Having discovered the problem, and it is a very real and vital one, let's analyze the needs of the average user of the FF command. Typically, it is a businessman sending out a long string of bills. Or, perhaps it's an organization sending out a pile of form letters, or printing address labels. In any case, what happens is repetitive, with perhaps certain places on preprinted forms filled in by the printer.

This means the printer runs continuously. The program will certainly not stop after each form is printed. The program printing all these forms will be locked in a continuous loop, READing in and processing new information from disk, tape, data lines, or even from the keyboard, then printing it in a specific format on the forms.

Understanding and appreciating that concept is vital to understanding how FF is used.

Now, if we can delay execution of the extra LFs caused by the sending of form control commands, we can effectively forget them. Sort of like taxes. Delay them long enough and they don't matter.

We learned long ago in our study of elementary BASIC that a semicolon (;) at the end of a PRINT line temporarily suppresses its LF and CR. But when the program comes to its end, the suppressed LF/CR catches up. What do you suppose would happen if we put our LPRINT CHR$(140) FF command into the continuous loop with the form printing statements? You got it!

NOTE: The suppressed LF/CRs don't just keep adding up and dump out at the END, spilling all over the floor. No more than 1 of them can hang over our head, awaiting disposition.

Charge!

We can modify our resident program so it can't end without human intervention. Everyone add:

```
49 GOTO 20
```

and RUN. Forget about the paper! If you have to fuss, worry about the Medfly, or the snowpack.

Now we are all on the same channel, and the printer hasn't forgotten where the top of the form belongs.

When the floor is strewn with paper and the point is made, you may hit <BREAK>, (or <CTRL-C>) and let a silent smile creep across your ugly puss. Success is sweet, eh Crock! (Crunch!)

Onward

Suppose we're not using full 66 line forms. When's the last time you received a statement on a sheet of paper that big? Oh really? And it started out "Greetings from the President . . ."?

Let's suppose we're printing on forms that are only half that long, 33 lines per sheet. How do we set the FF to automatically go to the top of a form that isn't 66 lines long? Tho't you'd never ask. (Tho't we'd never get here!)

Setting the Form Length — By Number of Lines

<ESC>"C" sets the scene for us to set the form length. Add line 10:

```
10 LPRINT CHR$(27)"C"CHR$(33);
```

<ESC>"C" followed by CHR$(##) specifies how many lines are to be on the form — the form length. It also resets the top of form to the current print head position and resets the form length from 1 to 127 lines long (don't use 0 or 128). The form length is automatically set to 66 lines when the printer is turned on or reset with <ESC>"@".

We set the form length to 33 in line 10. The semicolon at the end of the line delays the inevitable LF until the program ends.

Press FF to TOF

and RUN (for several sheets).

Remarkable! All this fussing and stewing really does pay off.

The current listing reads:

```
10 LPRINT CHR$(27)"C"CHR$(33);
20 FOR N = 1 TO 10
```

```
30   LPRINT N
40 NEXT N
45 LPRINT CHR$(140);
49 GOTO 10
```

Setting the Form Length — By Number of Inches

<ESC>"C"N specifies the form length in number of lines. Does that mean that a form of 66 always represents 11 inches? Not necessarily. In Chapter 11 we'll learn how to change the line spacing from the standard one sixth of an inch to anywhere from 1/216 inch to over one inch. The form length of 66 lines can really vary from less than one inch to over 66 inches. This could be one big headache, so EPSON provided an alternate means of declaring the form length. Change line 10 to:

```
10 LPRINT CHR$(27)"C"CHR$(0)CHR$(11);
```

and RUN.

NOTE: Users having problems with this program should see Appendices G through I.

The form length has been set to 11 inches. And it will stay 11 inches no matter what weird line spacing we may try. It's sort of insurance against our own clumsiness.

In the sequence <ESC>"C" 0 N, the number N represents the number of inches in the form length. It can be a whole number from 1 to 22.

There's not a lot more to say about FF and LF. It's easy to let them draw us into trouble, yet the trouble is so easy to avoid. If the very concept is giving you problems, go back and restudy the Homespun Philosophy, then start the chapter again from the beginning.

Type:

```
LPRINT CHR$(27)"@"
```

to clear the printer memory of any radical codes and reset the paper to the Top Of Form. Now we're set to proceed to the next chapter.

Code Summary

12 & 140 = Form feed

<ESC>"C" CHR$(##) = Sets form length by number of lines and resets TOF
1 < = ## < = 85

<ESC>"C"CHR$(0)CHR$(##) = Sets form length in inches and resets TOF
1 < = ## < = 22

Chapter 6

Special Features and Word Processing

Underlining

Underlining is a special mode, just like the others. To switch on underlining, send <ESC>"-"CHR$(1). To turn it OFF, use <ESC>"-"CHR$(0).

Type in this new program:

```
5 E$ = CHR$(27)
9 PR#1          (Apply only. Also, PRINT instead of LPRINT)
20 LPRINT TAB(10)"UNDERLINING IS AN "; E$"-"CHR$(1);
30 LPRINT "IMPORTANT "; E$"-"CHR$(0); "TECHNIQUE"
40 LPRINT E$"@"
99 PR#0         (Apple only)
```

and RUN.

```
UNDERLINING IS AN IMPORTANT TECHNIQUE
```

Figure 6-1

The program shows an efficient way to handle the often used <ESC> code. Storing it in string variable E$ in line 5 streamlines the program.

The <ESC>"-"CHR$(1) sequence at the end of line 20 sets up the underline mode. The number can be from 1 to 255. <ESC>"-"CHR$(0) in line 30 turns underline off.

RUN it again and watch the print head. Notice that the line is printed in one pass, although the print head goes through some funny gyrations while it prints IMPORTANT. While in underline mode, the bottom pin (pin 9) is fired while every character is printed. Even blank spaces are underlined unless they occur at the end of the print line. In fact, even spaces created by a horizontal TAB are underlined. Amazing.

Underlining can also be effectively combined with several other modes. Add:

```
10 LPRINT E$"G"
```

and RUN to try it with double strike. Notice how the first three words are double printed before IMPORTANT is printed. The printer handles underline as a very special mode indeed.

Go ahead and try several different modes in line 10. We'll wait right here. Underlining even works with superscripts and subscripts, to be covered shortly.

Overstriking

In the old days, underlining was done by printing a line, returning the carriage (print head) without advancing the paper, and printing the underline character, ASCII 95, under the appropriate characters in that line. This is called overstriking.

As we saw above, we don't need to overstrike to achieve underlining, but we can use the overstriking idea to do such things as slashing zeros, crossing sevens, etc. On the MX printers, overstriking is done one character at a time.

NOTE: Some word processor programs underline by a process of successive backspacing and overstriking with the underline character (ASCII code 95). The result appears as a broken line instead of the solid line produced by <ESC>"-".

Enter the following new program:

```
9 PR#1                (Apple only)
10 LPRINT TAB(10)"JAMES BOND 0"; CHR$(8)"/";
20 LPRINT "0";CHR$(8);"/";
30 LPRINT "7";CHR$(8);"-"
40 LPRINT CHR$(27)"@"
99 PR#0               (Apple only)
```

and RUN.

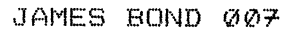


Figure 6-2

Fun! CHR$(8) is similar to the backspace key on a typewriter. When sent to the printer, CHR$(8) causes the printer buffer to print its contents, then reposition the print head over the last character printed — in perfect position for overstriking. Although overstriking one character at a time may seem a bit cumbersome, it is a small price to pay for such a useful feature.

To convince ourselves that CHR$(8) really does empty the printer buffer each time it is sent, let's add a couple of time delays:

```
15 FOR I=1 TO 2000 : NEXT I

25 FOR I=1 TO 2000 : NEXT I
```

and RUN.

The trailing semicolons in lines 20 and 30 would normally hold off printing until line 30, but CHR$(8) causes printing each time it is sent. (Some users will find that their printer will backspace without clearing the buffer.)

One quirk in using the backspace. In expanded mode, CHR$(8) causes a full double width backspace as we would expect. The fun begins when several backspaces are done in succession. All except for the first one are normal-width backspaces.

Is a Slashed Zero a Zorro?

Slashed zeros are widely used in the computer field so they can be easily distinguished from the letter 'O'. EPSON has given us a way of slashing all our zeros without resorting to overstriking. We simply move internal switch 1-7 to the ON position. Once the change is made, all zeros are printed with slashes.

Let's all shut down the computer and printer now and reset switch 1-7. Recheck the switch chart in Chapter 1 if necessary. This is no time to get the switches mixed up.

OK. Now turn everything back on, and type in:

```
9 PR #1                              (Apple only)

10 LPRINT TAB(10)"JAMES BOND 00";

30 LPRINT "7";CHR$(8);"-"

40 LPRINT CHR$(27)"@"

99 PR #0                             (Apple only)
```

and RUN.

Aha, that slashed zero might just give our firm that "computer professional" image we've been striving for. If that isn't your style and you don't really want the slashed zero, turn everything off again and reset switch 1-7 to OFF. We'll wait. If you've come in a bus or with friends, they'll wait.

Unidirectional Printing

Now who would want to suppress the powerful bidirectional printing feature of our MX printer? Someone who is trying to print tidy columns of numbers in compressed mode, that's who. Enter this new program:

```
9 PR#1                    (Apple)
20 LPRINT CHR$(15)
30 B=1001
40 FOR N=1 TO 10
50 I = B * .12345
60 B = B + I
70 LPRINT TAB(10); I, B
80 NEXT N
90 LPRINT CHR$(27)"@"
99 PR#0                   (Apple)
```

and RUN.

```
 123.573          1124.57
 138.829          1263.4
 155.967          1419.37
 175.221          1594.59
 196.852          1791.44
 221.154          2012.6
 248.455          2261.05
 279.127          2540.18
 313.585          2853.76
 352.297          3206.06
```

Figure 6-3

Just what we expected. Normal bidirectional printing and two peaceful columns of numbers in compressed character mode. They don't look too bad, but under close scrutiny the numbers do look a little out of line. For a comparison, add:

```
10 LPRINT CHR$(27)"U"CHR$(1)
```

and RUN again.

```
123.573          1124.57
138.829          1263.4
155.967          1419.37
175.221          1594.59
196.852          1791.44
221.154          2012.6
248.455          2261.05
279.127          2540.18
313.585          2853.76
352.297          3206.06
```

Figure 6-4

<ESC>"U"CHR$(N), where N is larger than zero, causes left to right printing only. Bidirectional is overridden. It really does improve the looks of the printout where precision column line up is desired. Unidirectional printing stays on until an <ESC>"U"CHR$(0), <ESC>"T", or the master reset code is sent.

We even have another version of unidirectional printing.

Try:

```
10 LPRINT CHR$(27)"<"
```

and RUN, watching the print head.

Clearly, that didn't do us much good. This version shuts off at the end of the first print line, hence bidirectional printing returns in the second row. Delete line 10 and change line 70 to:

```
70 LPRINT TAB(10); I,B; CHR$(27)"<"
```

and RUN.

Even though <ESC>"<" is placed at the end of the print line, it causes the entire line to be printed from left to right. Since it is inside a loop, all 10 lines are printed unidirectionally.

Keep your eyes peeled for more unidirectional printing in the next chapter.

Subscripting and Superscripting

Subscripts are used mostly by mathematical types to label variables.

For example: Physicists use V_0 and V_1 as initial and final velocity. Superscripts are used for footnotes and as exponents in mathematical expressions.

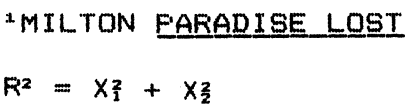


Figure 6-5

Superscripts are created on some printers by rolling the paper backwards a bit, printing the superscript, then rolling the paper forward so the print head is back on the current line. Subscripts are printed in a similar fashion. EPSON has eliminated all that fuss and bother. They've come up with a way of printing "short" characters; sort of like a can after someone stepped on it.

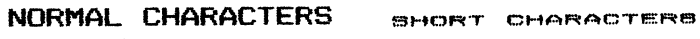


Figure 6-6

Being much shorter than the other character types, the script characters can easily be printed high or low on the line to give the desired superscript or subscript without any fancy partial line feeds.

We can do subscripting or superscripting with ease on our MX printer. <ESC>"S"CHR$(##) turns ON both super and subscript modes. If ## is 0, we get superscripts. Otherwise, we get subscripts. The off switch for both modes is <ESC>"T". Let's try it on for size:

Enter this new program:

```
9 PR#1                          (Apple)
20 LPRINT TAB(10)CHR$(27)"S"CHR$(1)"SUBSCRIPTING ";
30 LPRINT CHR$(27)"T"; "AND "; CHR$(27)"S"CHR$(0);
40 LPRINT "SUPERSCRIPTING "; CHR$(27)"T";"ARE EASY"
```

```
60 LPRINT CHR$(27)"@"
99 PR#0                          (Apple)
```

and RUN.

```
SUBSCRIPTING AND SUPERSCRIPTING ARE EASY
```

Figure 6-7

<ESC>"S"CHR$(1) in line 20 kicks in subscript mode.

<ESC>"T" in line 30 shuts it off. <ESC>"S"CHR$(0) in line 30 starts superscript mode, and

<ESC>"T" shuts it off in line 40.

Line 60 is our often used master shutoff switch, thrown in to cancel any left over codes.

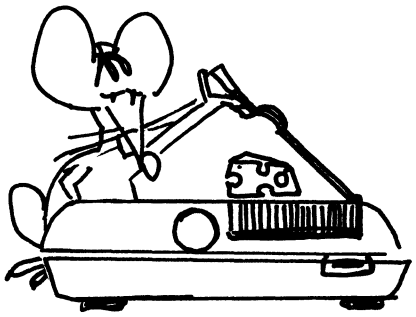
We mentioned that these script modes leave us in double strike printing. To verify it, add:

```
50 LPRINT TAB(10)"BUT THEY LEAVE US IN DOUBLE STRIKE"
```

and RUN.

```
SUBSCRIPTING AND SUPERSCRIPTING ARE EASY
BUT THEY LEAVE US IN DOUBLE STRIKE
```

Figure 6-8



Yup. It shore do. Line 60 cancels it.

Better Mousetrap

But we do have a better way to exit superscript and subscript modes. Change the "T" in lines 30 and 40 to "H" and line 50 to:

```
50 LPRINT TAB(10)"OUT OF DOUBLE STRIKE"
```

and RUN.

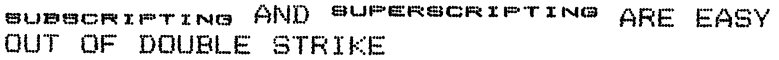


Figure 6-9

<ESC>"H" cancels both double strike and script characters, but leaves the printer in the unidirectional printing mode.

Party Mixer

Now we come to the fun part; mixing scripts with other print modes. Weird, you say? Not so.

For example, printing superscripts and subscripts in compressed mode might just add a touch of class to our output. On the other hand, people might think we're a bit strange if we underline our script characters. On the other hand, just working with computers . . .

There are two modes that don't work well with super/subscripts. The first is double width mode.

The <ESC>"S" sequence is ignored when the printer is printing double width. Try it. Add:

```
10 LPRINT CHR$(14);
```

Change line 40 to:

```
40 LPRINT "SUPERSCRIPTING "; CHR$(27)"H"
```

and RUN.

Yes, the entire line is printed double width, and subscripting is nowhere to be seen. (We changed line 40 so narrow paper users wouldn't let their print head stab their roller.) Now, EPSON must have a good reason for shutting off the script mode, but as "foot loose and fancy free" programmers, it is our duty to try it anyway. Right? Right! Delete line 10 and change line 20 to:

```
20 LPRINT TAB(10)CHR$(27)"S"CHR$(1)CHR$(14)"SUBSCRIPTING ";
```

See the CHR$(14) we snuck in **after** the <ESC>"S"? RUN it to see what happens.

SUBSCRIPTING AND SUPERSCRIPTING
OUT OF DOUBLE STRIKE

Figure 6-10

Well, now we know. Clearly, script is ignored when double width is active. But cancel double width, and script will rear its head once again.

If we are using script characters and then go into graphics mode (much more about that later), when we exit graphics mode, script characters are cancelled but we remain in double strike. (Yes, it is getting late, isn't it!)

Script characters likewise do not mix with emphasized mode; emphasized characters simply take charge. When emphasized is cancelled, it reverts right back to script mode. Script mode is only temporarily suppressed. Each code we turn on must be turned off manually.

Also, do not cancel unidirectional print while in script mode or you may be in for a surprise. We'll leave you here to play with script and other character modes as the spirit moves. Enjoy.

Creeping Script

One last point about script characters. If you stumble across an application that requires frequent switching in and out of script mode, be aware that the paper will creep a bit. This is true with any kind of double strike printing. Each time we leave script mode, the paper does a slight 1/216 inch paper feed. Not enough to cause ulcers, but noticeable if done several times on the same line. To see this, type in this new program:

```
9 PR#1                          (Apple)
10 LPRINT CHR$(15)
20 FOR X=1 TO 30
30  LPRINT "MX";
40  LPRINT CHR$(27)"S"CHR$(1);
50  LPRINT "80";
60  LPRINT CHR$(27)"H";
70 NEXT X
```

```
80 LPRINT CHR$(27)"@"
99 PR#0                          (Apple)
```

and RUN.

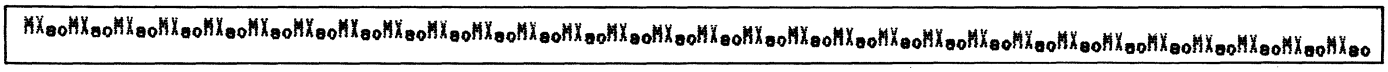


Figure 6-11

Line 10 throws us into compressed mode. The rest of the program sets up a loop where we jump back and forth between script mode and normal mode on the same line. Even though we turn off script mode with <ESC>"H", the script characters are printed in double strike, which causes the creeping down the line.

A Word About Word Processing

Now that we've seen an impressive array of printer capabilities, how do we apply them to our daily correspondence? The fact is, printer technology is a step or two ahead of most word processing programs. Each time a new printer or ROM change is announced, there is a time lag before the word processing programs can be modified to match. But this cloud does have a silver lining.

There are several widely used word processing programs for microcomputers, including SCRIPSIT™ and The Electric Pencil™. Most share a common weakness. At this writing, they do not allow us to send our choice of CHR$ commands to the printer. We can overcome this computer software deficiency by sending commands to the printer from the BASIC language prior to printing from the word processing program, but it is a nuisance.

Let's say, for example, that we want to print a high quality business letter by using double strike and emphasized to make several carbons. We have to enter BASIC and send the following instructions **before** calling up the word processor program and printing the letter:

```
LPRINT CHR$(27)"E"

LPRINT CHR$(27)"G"
```

<ESC>"E" turns on emphasized printing for the carbons, and <ESC>"G" turns on double strike.

If we'd wanted, we could have sent a CHR$(15) to set compressed mode, or even CHR$(27)"W"CHR$(1)CHR$(15) for double width compressed instead of emphasized printing. And don't forget italics and double width (with <ESC>"W", not CHR$(14)) characters. We're overwhelmed with options!

We recall that all these features stay on until specifically turned OFF by their "opposite" <ESC> codes. Loading a word processor program into the computer will not usually affect what the printer has stored in its own memory as long as we don't turn the printer off. And if for some reason, it does, we can "permanently" set some features (compressed, italics, and emphasized) via the internal printer switches.

Some More Good News

Since we wrote the first MX-series manual for the original MX-80, an increasing number of "patch" programs have become available for the more popular word-processing programs. These software patch programs allow the user to modify the original word processor software to send the codes we need to control our printer from within the word processing text.

Some software houses have written word processors specifically for the MX series. Some even carry on like they invented the MX series printers! Ah well, success has many fathers, but failure is an orphan.

The software patch program is a great idea, except some of them permit control over only a limited number of printer features — whatever the software writer happens to consider important.

A few word processing programs like Wordstar for CPM systems, Lazy Writer for the TRS-80, and Letter Perfect for the Apple have cleverly bypassed the problem of rapidly changing printer technology. In these programs, users are allowed to imbed any of the 256 ASCII codes directly in the text. With this capability, we can keep up with any printer changes and take full advantage of what we've learned so far, plus what follows.

With some word processors it is even possible to incorporate graphics. Now that's flexible! This approach allows for the creation and easy printing of custom letterheads for those who have the inclination to create them.

> Let me sneak in a quiet "THANXs!" here to the many readers of my other EPSON MX series manuals who have written with their custom letterheads printed on MX printers. I really enjoy hearing from you!

Figure 6-12 shows a printout created by the Lazy Writer (*) word processing program. Since every word processor is different and this is just one example to stimulate the creative juices, no explanation will be offered. But do notice that these "generic" word processors can take full advantage of MX printers.

Insect Extermznator

Rid your house of disgusting bugs **forever!**
New scientific breakthrough: ZQ_2TR^4 is a permanent barrier between revolting insects and your loved ones.

Protect your children from dangerous multipedal creatures.

Just paint Kontrol Debug along your baseboards. It leaves a rich black finish, tacky to the touch, which traps insects and small animals instantly on contact.

Manufacturer is not liable for any damage to property or loss of children, friends, or pets.

Please rush me one bottle of Kontrol Debug for the fantastically low price of $129.95

NAME_____

ADDRESS_____

CITY_____ ST__ ZIP_____

Figure 6-12

Code Summary

CHR$(8) — Backspace
<ESC>"-"CHR$(1) — Turns on underline
<ESC>"-"CHR$(0) — Turns off underline
<ESC>"<" — Prints single line from left to right
<ESC>"S"CHR$(1) — Turns on subscripting
<ESC>"S"CHR$(0) — Turns on superscripting
<ESC>"T" — Turns off subscripting or superscripting and unidirectional printing
<ESC>"H" — Turns off scripting and double strike
<ESC>"U"CHR$(N) — Turns ON unidirectional (left to right) printing if N is not zero
<ESC>"U"CHR$(0) — Turns off unidirectional printing

Chapter 7

Seven Bits, Foreign Symbols, and Line Graphics Characters

Time out for 7-Bit Printer Interfaces

In the olden days (i.e., several years ago) 7 lines or electrical circuits were used to pass data from a computer to a printer. These 7 lines were adequate to send the original standard ASCII set of numbers, from 0 to 127. (The math whizzos amongst us will quickly check their toes to verify that 2 to the 7th power is indeed 128.)

Many microcomputer manufacturers now use an eighth line to expand the number of ASCII options to 0-255. This makes room for additional codes for graphics and other special characters, including special printer codes.

Unfortunately, many printer interface devices still use the old 7-bit system and restrict their computers to only sending code numbers from 0-127. Some MX-series printer applications require the up-to-date 8-bit system with code numbers up to 255.

TABing in compressed mode is one example of this requirement. Users with only 7-bit capability (most Apples) can set printer TABs only up to column 127. To move further across the page, it's back to strings of blank characters and trailing semicolons.

EPSON is trying to make allowances for these computers by providing a software way to temporarily send out that eighth bit. It is often the printer interface card that sabotages the eighth bit, not the computer. In the Apple's case, Apple BASIC adds 128 to all ASCII codes. The interface card shuts off the eighth bit so that codes sent to the printer will conform to standard ASCII.

Here's how the eighth bit feature works. Simply send <ESC>">" to turn on the printer's eighth bit software, and <ESC>"=" to turn it off. This feature allows seven bit interfaces to send CHR$ codes of greater than 128. By so doing, it allows us to set very long column lengths, or very wide line spacings.

This promising innovation is somewhat limited in its application. The main "got-cha" is that if the eighth bit remains set during ESCAPE sequences, it may cause more problems than it solves. Worse yet, this limitation makes sense if you think about it.

Let's take a dry run through setting a horizontal TAB with an <ESC>"D" sequence, pretending that this eighth bit was set (ON). If we ran:

(Don't type it. This is a dry run, remember?):

```
9 PR# 1

10 LPRINT CHR$(27)">"

20 LPRINT CHR$(27)"D"CHR$(88)CHR$(0)

99 PR#0
```

and the eighth bit stayed active throughout line 20, the printer would think we were a little strange.

Setting the eighth bit would effectively add 128 to all four codes in line 20, so the printer would receive CHR$(155), CHR$(196), CHR$(216), and CHR$(128). The 155 stands for <ESC> just like 27; we got lucky on that one. But the 196 is certainly not the "D" the printer is looking for; hence, this sequence flops if the eighth bit remains on. Other escape sequences will work quite well.

For example if we needed to set the column length to 130 columns we could use this program:

```
9 PR# 1

10 PRINT CHR$(15)   (for narrow paper)

20 PRINT CHR$(27)">"

30 PRINT CHR$(27)"Q"CHR$(130)

99 PR# 0
```

What you see is what you get.

It seems that we are stuck somewhere between a rock and a hard place, but let's not give up too soon. There is some practical use for this feature after all. By setting the eighth bit, we can very reliably access the built in printer symbols represented by ASCII codes 128 - 255. Looking at the self test or Appendix A, we see that these codes represent the italics characters, some foreign symbols, and a few line graphics characters.

After setting the eighth bit, we may wish to shut it off completely with <ESC>"=". And, if we wish for the printer to pass the eighth bit as it receives it (i.e., reset it to normal), use <ESC>"#".

Users whose computers can send all 8 bits have no need to mess with these codes. Others may find in them the way to use more of their EPSON printer features.

Foreign Characters

A multi-lingual printer? Yes and no. Our MX printer does not have a complete built in foreign character set, but it has the next best thing. Included in the ASCII character set are the following symbols:

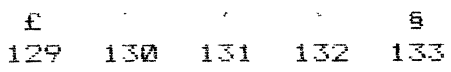


Figure 7-1

Not all the symbols in Figure 7-1 are characters themselves, but many of the European characters that we may have occasion to use can be formed with these special symbols and the overstrike code — CHR$(8). For example, type:

```
9 PR# 1                    (Apple)
10 LPRINT "ENGLAND: ";CHR$(129)
20 LPRINT "FRANCE: ";CHR$(133)
30 LPRINT "GERMANY: "; CHR$(97)CHR$(8)CHR$(130)
40 LPRINT "ITALY: "; CHR$(117)CHR$(8)CHR$(132)
50 LPRINT "SPAIN: "; CHR$(126)CHR$(8)CHR$(110)
99 LPRINT CHR$(27)"@"
```

and RUN.

Yes, 7-bit computers will have to turn on the eighth bit in order to send codes greater than 127. In fact, we can just turn the eighth bit on and leave it on if we are willing to accept italic characters.

Notice that the higher set of codes is a duplicate of the lower set, except most of them are italics. Here's a perfect program to practice on. After every lesson, there's a quiz.

```
ENGLAND: £
FRANCE: §
GERMANY: ä
ITALY: ù
SPAIN: ñ
```

Figure 7-2

Spinning the globe, we find the English pound and the French franc. German and Italian characters are formed by overstriking to combine lower case letters with codes 130 and 132 respectively. And the Spanish character uses . . . ah, er, it looks like we have a bit of a problem here. The tilde and the "n" seem to overlap a bit. Hmmmm . . .

What would happen if we printed the "n" as a subscript? It should squish down underneath the tilde. Shall we give it a try?

Change:

```
50 LPRINT "SPAIN: "; CHR$(126)CHR$(8);
```

and add:

```
60 LPRINT CHR$(27)"S"CHR$(1)CHR$(110)
```

and RUN.

```
ENGLAND: £
FRANCE: §
GERMANY: ä
ITALY: ù
SPAIN: ñ
```

Figure 7-3

Amazing! With patience and perseverance we can do wonders with this printer.

Line Graphic Characters

While we are nosing around this region of the ASCII chart, let's see what we can do with the 11 line graphic characters EPSON has bestowed on us.

134	149	150	151	152	153	154	156	157	158	159
┌	┐	├	┤	┬	└	┘	│	─	┴	┼

Figure 7-4

Enter this new program:

```
9 PR#1                                  (Apple)
10 LPRINT CHR$(27)"A"CHR$(8)
15 LPRINT CHR$(27)">"                   (7-bit computers)
30 FOR I=1 TO 4: FOR J=1 TO 8
40 READ N : LPRINT CHR$(N+100);
50 NEXT J: LPRINT: NEXT I
60 LPRINT CHR$(27)"@"
70 DATA 34,57,57,49,34,57,57,49
80 DATA 56,34,49,56,56,34,49,56
90 DATA 56,53,54,58,51,50,54,56
100 DATA 53,57,52,52,51,50,57,54
```

and RUN.

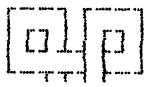


Figure 7-5

Not bad, but why are the vertical lines out of alignment? Could it be that bidirectional printing offsets the lines just slightly? How could we find out? Why isn't there a line 20 in the above program? How many hints do we need?

Add:

```
20 LPRINT CHR$(27)"U"CHR$(1)
```

and RUN.

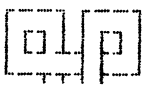


Figure 7-6

<ESC>"U" threw the printer into unidirectional printing and solved our alignment problem.

Now about the rest of the program:

Line 10 sets the line spacing at 8 dots, just enough for the vertical lines to touch end-to-end. (We'll explore <ESC>"A" fully in Chapter 11.)

Line 30 starts two loops: one for the 4 rows and the other for the 8 characters per row.

Line 40 reads the code numbers from the data lines and prints the ASCII characters. Notice that the numbers are stored without the leading "1", thus saving us from a lot of typing.

Line 50 closed both loops, adding a Line Feed at the end of each row.

And line 60 prepares the printer for its next assignment.

Now that our fingers are rested, change line 30 to:

```
30 FOR I=1 TO 8: FOR J=1 TO 8
```

and add the rest of the data:

```
110 DATA 34,57,51,50,58,58,57,49

120 DATA 56,34,51,50,52,52,49,56

130 DATA 56,53,54,56,56,53,54,56

140 DATA 53,57,57,54,53,57,57,54
```

and RUN.

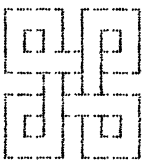


Figure 7-7

Truly aMAZEing.

Figure 7-8 shows a diagram that even graphic artists might be proud of.

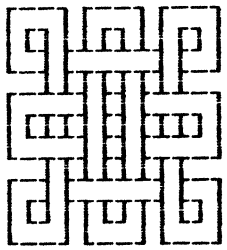


Figure 7-8

Table 7-1 shows the line graphic character codes for 7 bit machines when the eighth bit is set.

Table 7-1

DEC	CHAR
6	┌
21	┐
22	├
23	┤
24	┬
25	└
26	┘
28	│
29	─
30	┴
31	┼

Code Summary

<ESC>">" — Sets (turns ON) the eighth bit.
<ESC>"=" — Resets (turns OFF) the eighth bit.
<ESC>"#" — Printer accepts eighth bit "AS IS" from computer without bit fiddling.
<ESC>"A" N — Sets VERTICAL LINE SPACING to N/72". See Chapter 11.

Chapter 8

Skipping Over the Perforation

Here's a short and simple chapter, for a change.

Hopscotch Revisited

MX printers have a built in skip over perforation feature, known in the biz as SOP. The printer can be instructed to automatically skip over the perforations in continuous feed paper. It's a variation on the old sidewalk game of hopscotch.

The end result is a nice neat printout with no printing on the cracks. This feature is of special value to programmers who send a steady stream of information to the printer when LISTing their latest creation.

SOP may be software controlled by sending a special code down the line, or hardware controlled by switch 2-4 inside the printer. Let's exercise the SOP under software control, first.

Put the SOP to Work

To activate SOP we send

```
<ESC>"N"
```

to the printer, followed by a CHR$ code indicating the number of lines to be skipped. Any number from 1 to 127 will work, but if we don't specify how many and simply press <ENTER> it will skip 13 lines, about 2 1/6 inch. (The ASCII code for CR is 13). Of course, the number of lines skipped cannot be greater than the length of the page set by <ESC>"C". (Some readers try everything!)

Soft SOP

Type in this demonstration program:

```
NEW
9 PR#1     (Apple)
10 LPRINT CHR$(27)"N"CHR$(18)
20 X=X+1
30 LPRINT X
40 GOTO 20
```

Be sure the paper is set to top of form and RUN.

<CTRL-C> or <BREAK> the program RUN when you've had enough.

Line 10 sends an <ESC>"N" down the line turning on the SOP. Each time the paper got within 18 lines of the perforation, the printer leaped over it.

SOP works great with program listings too! If a BASIC program is longer than 54 lines, just send:

```
LPRINT CHR$(27)"N"CHR$(12)
```

from BASIC and LLIST the program. The listing will not crowd the perforation.

HINT: Since this total SOP white space is 2 inches, when LLISTing a program, roll the perforation up an extra 1 inch beyond where we usually set it. That way, 1 inch of white will be automatically placed at the top and bottom of each sheet.

SOP stays ON until the printer is turned OFF or it is cancelled by:

```
LPRINT CHR$(27)"O"
```

Everybody turn it OFF, one way or the other.

Adjustable Skip Distance

Suppose we are more concerned with simply missing the perforation than having a big bottom margin. Let's change line 10 to:

```
1Ø LPRINT CHR$(27)"N"CHR$(4);
```

and RUN.

The printer skipped over only four lines. You get the idea, so <BREAK> the program.

Permanent Skip Over Perf

If the nature of our printing is such that we want **all** of our printouts to automatically skip over the perforation, we can flip an internal switch. From then on, we won't have to worry about including a SOP command in every software program. The printer will always SOP, unless we tell it not to, in software.

Let's all turn our computer and printer OFF and unplug them from "the mains". Set internal switch 2-4 to the ON (left) position.

Plug them back in, adjust the paper to the top of form, and turn them back ON.

Reenter our prior program, without line 10, and RUN.

How about that? The SOP came on automatically when the printer was turned ON, and was set to a "skip distance" of 6 lines, or 1 inch.

We can still send

```
<ESC>"N" CHR$(##)
```

to change the skip setting, and use

```
<ESC>"O"
```

to shut SOP off. Let's try shutting SOP off. Now type:

```
LPRINT CHR$(27)"O"
```

Hit FF to bring the printer back to top of form, and RUN the program one last time. The margins are crowded again, just like normal.

Now that we know how switch 2-4 works, shut the whole works down, pull the power plugs, and slide it back to the OFF (right) position. We need to bring the thing back to normal in order not to interfere with the many other things we have yet to learn.

Code Summary

<ESC>"N" = Skip over perforation, set to 13 lines
<ESC>"N"CHR$(##) = Skip over perforation on, set to ## lines
1 <= ## <= 127
<ESC>"O" = Cancel skip over perforation

Chapter 9

Stuck With the TAB

Horizontal TABs

For any printer feature to work, it must receive orders from its computer. The computer program is worth a thousand words, or, er, something like that. Anyway, type in this simple NEW program:

```
9 PR#1                  (Apple)

20 FOR N = 5 TO 75 STEP 5     (narrow paper)

20 FOR N = 5 TO 136 STEP 5    (wide paper)

30   LPRINT TAB(N)N

40 NEXT N

99 PR#0                 (Apple)
```

and RUN.

AHa! Different folks are seeing different results. What a revoltin' development this is!

With many "old-time" computers, (2 or 3 years old), the printer will TAB over only as far as 39, 63, or 79 instead of all the way, then the printhead will return to the far left and stay there.

This is a computer software limitation. No, it isn't the printer's fault!

Everyone back to Appendices G-K for examples of BASIC TAB limitations and solutions.

The TAB function in BASIC was originally designed to work with narrow screen widths. Suddenly, we give it all this wide-carriage freedom and it doesn't know how to handle it. Shades of young adulthood.

Users who find their printer merrily printing numbers all the way out past 70 (120 on wide paper) can give their computers a little pat on the — ah, keyboard.

Lucky readers flushed with the latter success can go one step further. Let's go into the compressed character mode and find the limits of our TABbing ability.

Change the program to read:

```
9 PR#1              (Apple)
10 LPRINT CHR$(15);
20 FOR N = 5 TO 135 STEP 5    (narrow paper)
20 FOR N = 5 TO 250 STEP 5    (wide paper)
30   LPRINT TAB(N)N
40 NEXT N
50 LPRINT CHR$(18)
99 PR#0             (Apple)
```

and RUN.

Anyone's printer make it all the way to column 130 (230 on wide paper)? The rest of us will just groan, while our computer drops out of the race early at somewhere around 60 or 130. What you see is what you got.

All is not lost if the computer can't keep up with the printer. When printing on a form that needs the full 132 (or 233) column width, use program TABs as long as possible. Insert spaces the rest of the way, either one at a time or via the STRING$ statement. It's frustrating, but is about the only universal way out. And, it will work until that new computer arrives. (Ouch!)

Quiz Time

Before we scrap the above program, let's make a modest modification to see what happens. Add a semicolon to the end of line 30:

```
30   LPRINT TAB(N)N;
```

and RUN.

```
  5   10   15   20   25   30   35   40   45   50   55   60   65   70   75   80   85   90   95   100  105  110  115  120  125
130   135
```

Figure 9-1

Apple II users and possibly others are seeing strange apparitions. (Charles Dickens liked that word.) Apple BASIC TABs are designed for its 40 column screen. TABs past the 40 column limit are treated as strings of blank spaces. For example, there are 50 blank spaces between the 45 and 50.

There are several ways around this "short TAB" problem. One is to TAB over with PRINT statements containing blanks between the quotes. A nicer way, (not available on the Apple II) is to use STRING$ to insert blank spaces. The Apple computer has a similiar yet limited function called SPC(n) which prints n blank spaces. See *The BASIC Handbook,* 2nd Edition by your humble servant for details.

The point of this program was to show that the MX printers will accept horizontal TABs from BASIC, and help us determine our own computer's capability to send those TABs.

Internal Printer TABs

Yes, the printer itself has the ability to internally store horizontal TAB stops. This is not at all the same as being told by a BASIC program to TAB over to a certain print position.

Setting the internal TABs are like setting the TABs on an ordinary typewriter. Once set, we can quickly jump to each internal TAB stop, in sequence, by using control code CHR$(9). This printer capability is especially handy, considering the shortcomings of the BASIC TAB, particularly on wide paper.

Enter this new program (Apple users see Appendix J first)

```
9 PR#1                    (Apple)

10 LPRINT CHR$(27)"D" CHR$(10)CHR$(60)CHR$(110)CHR$(0)

        (TRS Model I use POKE 14312,0)

12 LPRINT CHR$(15)        (If using narrow paper)

15 PRINT CHR$(9);CHR$(3);          (Apple only)

20 LPRINT CHR$(9)"TABS";

30 LPRINT CHR$(9)"ARE";

40 LPRINT CHR$(9)"EASY"

50 LPRINT CHR$(27)"@"
```

and RUN.

TABS ARE EASY

Figure 9-2

Here's what all the shouting is about:

Line 10 used <ESC>"D" to open the door to accept H TABs

Line 10 set TABs at 10, 60, and 110 using CHR$(##)

Line 10 closed the H-TAB door with a NULL, CHR$(0) (or CHR$(128)). (TRS Model I users see Appendix G)

Line 15 reassigns the Apple's printer initialization character

Lines 20, 30 and 40 "called" the TABs, with CHR$(9) and printed something

Line 50 reset everything back to normal

Pretty slick, isn't it?

So what are the limitations of this built in horizontal TAB feature? Well, to start with, we are allowed a maximum of 28 tab settings across a line. The <ESC>"D" sequence must be terminated by a code of 0 or 128. (Clever readers will deduce that we cannot set TABs at 0 or 128. The rest of us will just keep stumbling on.)

We are not allowed to use 255 as a terminator. The maximum value of a TAB is the maximum number of characters per line allowable in the current character width.

Default TAB Setting

There are horizontal TABs already set even before we send the <ESC>"D" sequence. Change line 10 to a REMark line and RUN the program again.
RUN.

TABs set every 8 spaces are included free from the factory.

The H-TAB system makes a lot of sense on a wide carriage printer by letting us TAB beyond the limits of many computers, without resorting to cumbersome LPRINT statements. H-TAB works in both expanded and compressed character mode. However it is even more valuable when printing in compressed mode. From a practical standpoint, there's no reason to send a TAB number greater than 136 unless we are in compressed mode using wide paper.

Seven Bits Revisited

Remember our earlier discussion about 7-bit computers, and their limitations? TABing in compressed mode is another case where we need all 8 bits. Users with only 7-bit capability (most Apples) can set TABs only up to column 127. Setting the eighth bit via <ESC>">" doesn't work well with <ESC>"D", so it's back to strings of blank characters and trailing semicolons.

Page Width

The maximum carriage width can be set in software. This protects narrow width paper from an overambitious print head.

Width is set by sending an

```
<ESC>"Q"
```

down the line, followed by any number from 1 to 231. (Remember, 7-bit computers can't send numbers greater than 127).

Using the program already in the computer, restore line 10 back to its original self:

```
10 LPRINT CHR$(27)"D" CHR$(10)CHR$(60)CHR$(110)CHR$(0)
```

and type:

```
LPRINT CHR$(15)
```

```
LPRINT CHR$(27)"Q"CHR$(70)    (for 8 inch wide paper)
```

or

```
LPRINT CHR$(27)"Q"CHR$(100)    (for 11 inch wide paper)
```

and RUN.

Was the word EASY printed at the start of the next line? The program reduced the page width so it shouldn't fit on the same line. A horizontal TAB command of 110 can't TAB to column 110 if the page width is set to 70.

Note that <ESC>"Q" does not set the left margin; it only controls the right margin. Freely seasoned with common sense, page width can be a useful feature.

Type

```
LPRINT CHR$(27)"@"
```

to restore this mess back to the normal mess.

Vertical TABs

Vertical TABing was included in early versions of MX printers, but users found that for most applications, vertical positioning was easily done with line feeds. So the vertical tab code VT (ASCII 11) feeds the paper one line just like LF (ASCII 10). Therefore if you have software developed around this feature, then you will have to change it using line feeds in a loop.

Code Summary

<ESC>"D" — Sets printer horizontal tabs
<ESC>"Q"CHR$(##) — Sets column width to ## lines
1 < = ## < = 231

Chapter 10

Single Sheet Friction Feed

Hooray! No more fighting with reams of paper to print a short letter. The MX-80 F/T and MX-100 have friction feed capability. We can use them almost like a typewriter. Company letterhead and envelopes feed through like a charm.

NOTE: EPSON does not provide a retrofit friction feed kit for the standard MX-80, and use of non-EPSON add-ons may void your warranty. (Better read the warranty notice.) MX-80 users with tractor feed only, skip now to the multiple copies and forms section near the end of this chapter.

Back in Chapter 1 we learned how to remove and replace the tractor feed mechanism. We're going to learn how to feed through single sheets of paper now, so everyone remove the tractor feed mechanism.

Sensors on Full

A paper-out sensing mechanism is located near the paper entrance. With the printer OFF, alternately insert and remove a sheet of paper several times, listening for its click, to know what to listen for.

This sensor can cause us problems when we're feeding single sheets. As part of its assigned duties, if no paper is present at the sensor, it stops the printer from printing. If it didn't, the printhead might print all over the rubber roller, making a mess, and maybe even ruining it.

In order to print a full single sheet, we have to "defeat" the sensor. There are at least 3 ways to do it:

1. Feed a second sheet of paper in behind the first one after the first one has passed part way through.
2. Send a software code to disable the sensor.
3. Set switch 1-3 ON (left) to "permanently" disable the sensor.

Let's go through these options, in order.

The Backup Sheet

Feeding a second sheet of paper is an inconvenience, but it is an option open to those whose computers can't take advantage of options 2 and 3. More on that later.

Locate the black RELEASE lever on the left. For friction-feed operation, (what we're doing), this lever should be toward the rear of the printer to hold the paper securely. Let's check it out.

With the printer OFF, push the release lever toward the rear. Feed in a single sheet of typing paper using the paper advance knob, just like with a typewriter. If the paper feeds in a bit crooked, release it as necessary with the release lever, and carefully line it up. Adjust the little roller(s) on the paper bail to your satisfaction and push the bail up against the paper.

Type in this program:

```
9 PR# 1                    (Apple)

10 FOR I=1 TO 55

20   LPRINT "PAPER OUT"

30 NEXT I

99 PR# 0                   (Apple)
```

and RUN.

The sheet will feed normally until the paper-out signal is activated. Then the bell goes off and the printer stops. It's very frustrating when we are trying to print an important letter.

Well,

```
We
  2
   Never
     Say
       Die,
```

and all that nonesense, so turn off the printer and load in a new sheet of paper. This time, have a second back-up sheet waiting in the wings.

RUN the program again.

When the sheet is about half way through, slip the backup sheet behind and underneath it. Friction will carry it right along with the first sheet. This second sheet holds the paper-out switch down and allows the entire first sheet to pass through. A "tag-along" sheet is a necessity when printing envelopes using this option.

Disable Paper-Out Sensor by Software

Here's a smoother way to handle the problem. Load a new sheet of paper and type:

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

and RUN the program again. (Don't use a second sheet this time.)

Observe that all four printer lights are on at the same time. This is the only time we'll ever see it.

If the entire page printed all right without a backup sheet, skip the next two paragraphs.

Some users found that the <ESC>CHR$(8) didn't do any good. Well, clutch your diplomas 'cuz the explanation is a bit technical.

Under normal circumstances, printer cable pins 11, 12, and 32 are activated when the out of paper is sensed. <ESC>CHR$(8) protects pins 11 and 32 from the paper-out signal, but it doesn't protect pin 12. Since most computers don't monitor pin 12, <ESC>CHR$(8) works just peachy.

Computers that **do** monitor pin 12 (e.g. TRS-80) will halt printing when the paper is out. The only way <ESC>CHR$(8) can be effective with these computers is to get out the toolbox and disconnect and ground cable pin 12. This procedure is not recommended for amateur space cadets. EPSON dealers can provide a special TRS-80 cable with pin 12 "altered" (Part # 8222).

For every ON switch, there is an OFF switch.

```
<ESC>CHR$(9)
```

resets the paper-out sensor to its original condition.

Disable Paper-Out Sensor — Switch Setting

The third way to handle single sheet feed is to permanently disable the paper out sensor by sliding internal switch 1-3 to ON. This has the same effect as <ESC>CHR$(8); it allows printing to continue regardless of the paper status. (A great way for the careless to mess up the roller.) If you set switch 1-3 on, it powers up deactivated and <ESC>"9" in software activates it.

No matter which way switch 1-3 is set, <ESC>CHR$(8) and <ESC>CHR$(9) deactivate and activate the paper-out sensor, unless the computer monitors pin 12.

Multiple Copies and Carbon Forms

The MX printers are designed to print 1 original and up to 2 copies at a time. To get good second and third copies, throw the <ESC>"E" software switch (emphasized printing) and/or the <ESC>"G" software switch (double strike printing) as we discussed in A Word About Word Processing in Chapter 6.

For thick forms, we may have to adjust the head adjusting lever to make room for the paper and optomize print quality. For single-sheet operation, the lever should be in the middle position (straight up). For multiple copy forms, adjust it toward the front of the printer and watch the head move with the lever. EPSON recommends a maximum paper thickness of 0.3 mm (0.012 inch). Reach for the magnifying glass and ruler.

Code Summary

<ESC>CHR$(8) — Deactivates paper-out sensor.
<ESC>CHR$(9) — Activates paper-out sensor.

Chapter 11

Dot Matrix Printing

(Subtitled, "Does a Dull Title Make a Topic Dull?")

My Computer Can't Do High Resolution Graphics!

Correction, your computer **can** do high resolution graphics — with the MX series printers. Any computer that has the BASIC CHR$ function or equivalent and the correct computer/printer interface can punch out super graphics. It's as easy as sending CHR$(0), CHR$(1), . . . through CHR$(255).

Unfortunately, that may not be as simple as it sounds. If you are fluent in Binary math, you know that it takes 8 bits to count from 0 to 255 to specify the above 256 codes (2 to the 8th power is 256). Even if you aren't fluent in Binary math, it still does. Stick with us.

Many computers and printer interfaces support only 7 bits of code data, hence, they transmit to the printer only the ASCII code numbers between 0-127 (2 to the 7th power is 128). At this writing, EPSON's Apple parallel interface card is in this category. It's not impossible to drive dot graphics with only 7 bits, but it does make life more difficult. As mentioned earlier, we do have the set bit 8 command <ESC>">", which helps somewhat, but the printer still refuses to fire more than 7 pins at a time.

So what do 7 bit users do? Simple. So as to not exclude large numbers of friends, all examples from here on use only 7 bits. We fly in formation under the same banner.

How Dot Matrix Printing Works

To really understand dot matrix printing, we have to understand how the print head works. It contains 9 wires, or pins, positioned one above the other, as seen in Figure 11-1.

Each wire is driven by a "gun", or electromagnet that "fires" when told to do so by the electronics inside the printer. The printer electronics responds to the ASCII code instructions we send from the computer in the form of "ASCII numbers".

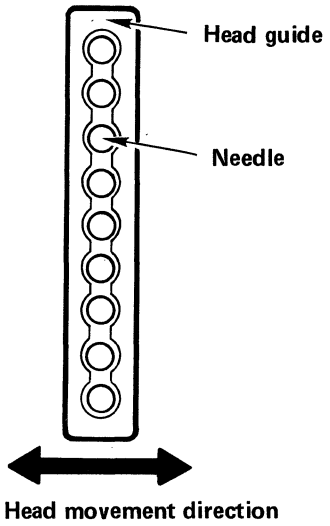


Figure 11-1

For example, to print the letter "I", wires 1 and 7 are fired first. They hit the ribbon, which makes marks on the paper where the left top and bottom of the letter "I" should be.

Then the head shifts over a bit and wires 1,2,3,4,5,6 & 7 all fire at once, printing the center of the letter "I". One more shift and one more shot of wires 1 and 7 and the letter "I" is complete (Figure 11-2).

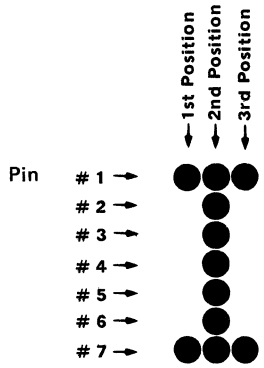


Figure 11-2

This is called dot matrix printing. Dots are printed according to a predesigned matrix or grid system, where each letter, number and punctuation mark is formed by an arrangement of dots. As we have seen, this complex printing process takes place **very** quickly.

The pins fire in groups or clusters. The firing patterns are already programmed inside the printer, matching the Alphanumeric (letter and number) characters and special symbols found in our ASCII charts. No, we cannot fire an individual pin yet, but that is what all this is eventually leading up to.

Appendix F shows all our Alphanumeric characters constructed within a 9-dot-high by 5-dot-wide matrix. Every letter, number, and punctuation character is designed to fit within that box. The uppercase letters are formed by the top seven pins. Some of the lower-case letters have descenders that extend below the line to give much more readable characters. Descenders drop 2 dots below the usual bottom line, and require the firing of pins 8 and 9.

Notice that the sixth COLUMN is always empty. In fact, it isn't even shown. It's reserved for horizontal spacing in between the characters.

No Alphanumeric character is wider than the "standard" 5 dots. If we look closely, it might appear that letters, like the "O" are wider. Actually, they contain more than 5 dots across the width, but those dots are spaced a bit closer than those in a "T". A "W" has even more horizontal dots, but its overall width does not exceed the width of the standard 5 dot pattern. The middle dots are just compressed to make more attractive characters. This technique makes your EPSON print characters easier to read.

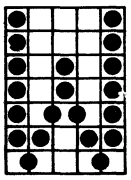


Figure 11-3

Our smallest alphanumeric character, the period, requires only 4 dots in a 2 by 2 grid. Our highest resolution in TEXT mode is therefore 4 dots.

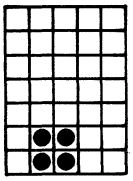


Figure 11-4

For a complete listing of all our standard printer characters, look at the printout from Chapter 1. It's right there, hanging on your wall, isn't it?

How it All Lays out on Paper

We have seen that all the characters can be contained in a 9-dot-high matrix. What Appendix F doesn't show is that the spacing between lines is a fixed 3 dots. This makes each text line a total of 12 dots high.

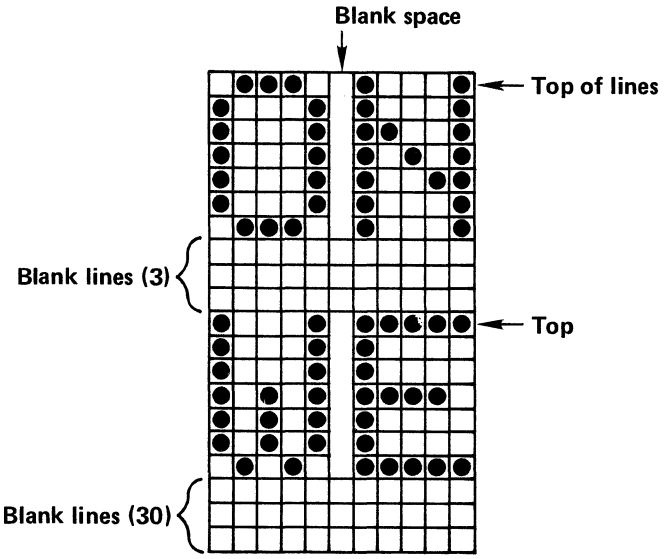


Figure 11-5

Here's another way of looking at it. There are 12 dots from the top of one row of text to the top of the next row. Since the spacing between the pins on the print head is 1/72 inch, each lines requires 9/72 + 3/72 = 12/72, or 1/6 of an inch. That's 6 lines per inch. No it isn't just a coincidence that it's the same as standard typewriter line spacing.

So what does all this higher mathematics have to do with graphics? Just this. If we are to create high-resolution graphics on our printer, we have to eliminate the dead space between lines by changing the automatic line spacing to some value less than 1/6 inch. We certainly don't want a blank space 3 dots or more high between our rows of graphics!

The Line Spacing is Variable

Now that we understand the dot matrix concept and standard 12 dot "top-to-top" vertical spacing, let's learn how to **change** it to suit our specific needs.

We have a choice of 3 "easy-to-use" line spacings:

HEIGHT (INCHES)	VERT DOTS	SOFTWARE CODES
1/6	12	Esc "2"
1/8	9	Esc "Ø"
7/72	7	Esc "1"

Let's try a simple test program and vary the spacing.

```
NEW

  20 LPRINT "LINE ONE"

  30 LPRINT "LINE TWO"

  40 LPRINT "LINE THREE"

PR#1                    (for Apple)
```

and RUN.

```
LINE ONE
LINE TWO
LINE THREE
```

Figure 11-6

Nothing new here. Spacing of 1/6 inch just like we're used to.

At command level type:

```
LPRINT CHR$(27) "Ø"
```

and RUN it again.

```
LINE ONE
LINE TWO
LINE THREE
```

Figure 11-7

That's what 1/8 inch spacing looks like. From the top of one character to the top of the one on the next print line is only 9 dots instead of 12. It's a little crowded.

We can tighten it up a bit more by typing:

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

and RUN again.

```
LINE ONE
LINE TWO
LINE THREE
```

Figure 11-8

That's too tight for comfort. Spacing of 7/72 inch with only 7 dots from top to top. Let's go back to where it looks better.

```
LPRINT CHR$(27)"2"
```

and RUN.

```
LINE ONE
LINE TWO
LINE THREE
```

Figure 11-9

That's better — 1/6 inch spacing. Any of these print spacing commands can be built into a program or used at the command level.

For the Aficianado

The 3 options above cover the needs of most users, but there are many more fine-grain options that can be used to "tweak" the vertical spacing. We will lightly explore how it's done to demonstrate the capability. Those who are really jazzed by this sort of thing can let their own imagination fly them away to where it feels good.

Life in the Fast Lane

Everybody add line 10:

```
10 LPRINT CHR$(27)"A" CHR$(12)    (or 12 + 128 = 140)
```

and RUN.

Same as before. Nothing changed. Then what is line 10 all about?

Line 10 contains the following 2 distinct commands:

1. <ESC>"A". It opens Pandora's box to let us set the vertical line spacing accurate to a single dot — 1/72nd of an inch. Mercy!
2. CHR$(##). It allows us to specify how many dots we wish to roll down between the top of one line's printing and the top of the next. The number can range from 1 to 85. We picked 12, since that is the default mode. (TRS-80 fans and some others may have to use CHR$(#+128) just to be on the safe side.)

Whatever new line spacing we specify is in effect as soon as line 10 is received by the printer.

To dramatically illustrate the possibilities this feature opens up:

DELETE lines 30 and 40

and change the following lines:

```
5 S=1                    '1 DOT

9 PR#1                              (Apple)

10 LPRINT CHR$(27)"A"CHR$(S)  (Model I TRS use S+128)

20 LPRINT "LINE ONE",S

50 S=S+1  :  GOTO 10
```

Line 5 starts us off with a dot spacing of 1/72 inch.

Line 10 brings the dot number "S" in as a variable.

Line 20 prints the dot spacing in 72nds of an inch.

Line 50 adds 1 to the dot spacing, and we do it all again.

RUN it until the dot spacing reaches at least 24, then <BREAK> OR <CTRL-C> to stop execution. Wow!

Apple users will notice a slight variation in the pattern around S=9 and S=12. Remember 9 is the Apple printer initialization code, so the computer isn't very happy about using it for line spacing. Code 12 is a form feed, and it adds an extra line feed to the printout.

But What do it Mean, Boss?

Well, Rochester, it's like this.

Where the number "1" is printed, the lines are spaced only one dot apart, instead of the usual 12 dots. That's why the letters are printed all over each other.

Check the printout and find where the spacing is 12. Doesn't that look more like what we're used to?

Now look at 24-dot spacing. Aha! It's double spacing. Be sending the high-powered command in line 10, we plug in our own value of S and make the printer give us just about any spacing we want between lines (up to 85). And, we can do it either inside a program or at the command level. That ought to kick the old mind into overdrive!

Let's LIST the program on paper to take a closer squint at it.

```
LLIST
```

The line spacing stayed where it was when we STOPped the program. That spacing is now the new standard, replacing our power-up value of 12 dots = 1/6 inch.

```
LINE ONE        1
LINE ONE        2
LINE ONE        3
LINE ONE        4
LINE ONE        5
LINE ONE        6
LINE ONE        7
LINE ONE        8
LINE ONE        9
LINE ONE        10
LINE ONE        11
LINE ONE        12
LINE ONE        13
LINE ONE        14
LINE ONE        15
LINE ONE        16
LINE ONE        17
LINE ONE        18
LINE ONE        19
LINE ONE        20
LINE ONE        21
LINE ONE        22
LINE ONE        23
LINE ONE        24
```

Figure 11-10

We can, however, easily shift back to the original (default) line spacing of 12 by typing:

```
LPRINT CHR$(27)"2"
```

Try it, then type:

```
LLIST
```

Ah! Sanity has returned. Whenever we send <ESC>"2", the spacing returns to the power-up value of 6 lines per inch — 12 vertical dots per line.

There are three other ways of returning the line spacing to its original size:

1. Using the <ESC>"A" sequence we just learned:
   LPRINT CHR$(27)"A" CHR$(12)
2. Turn the printer OFF, then ON.
3. Use the master reset code <ESC>"@"

Life Under the Microscope

Want even more control? Ask and ye shall receive. Change line 10 to:

```
10 LPRINT CHR$(27)"3"CHR$(S)
```

TRS Model I use:

```
10 LPRINT CHR$(27)"3";: POKE 14312,S : LPRINT
```

and RUN.

Press <CTRL-C> or <BREAK> to stop.

How's that for splitting hairs? We cracked the sound barrier and are closing in on the speed of light. The line spacing is now settable in increments of 1/216 inch. That's three times as fine as the <ESC>"A" line spacing.

S can range from 1 to 255. 7-bit computers can use <ESC>">" to select line spacing greater than 127. With this degree of accuracy, you can bet the boss will expect first rate art work.

<ESC>"3" stays on until it is changed to something else by one of our options. Now try:

```
10 LPRINT CHR$(27)"J"CHR$(S)
```

and RUN.

Vas is los???

<ESC>"J" turns on special line spacing for **one line only,** then defaults back to its previous setting. As soon as the <ESC>"J" is seen, the contents of the printer buffer is printed, followed by a line feed. This is true regardless of where the <ESC>"J" occurs on the program line.

Viva la difference!

Type LPRINT CHR$(27)"@" to clear out all the codes before continuing to the next topic.

Caveat

Life is never as rosy as the advertisements. If it was, where would the challenge be?

There are several code numbers already reserved for special things. If we look closely at the printouts from this chapter, we may see that the spacing when the following numbers are used may not be what we anticipated:

8 — reserved for backspace
9 — reserved for horizontal TAB
10 — reserved for line feed
12 — reserved for form feed
13 — reserved for carriage return

There are no nice clean ways around this problem — just very messy ones. (See Appendices G and J for some ways to POKE these codes to the printer.) Best we restrict our selection to the remaining numbers.

. . . and that's plenty to think about in this chapter.

Code Summary

CHR$(27)"A"CHR$(##) — Sets line spacing to ##/72 inch
1 < = ## < = 85
CHR$(27)"J"CHR$(##) — Sets line spacing to ##/216 inch
Turns off at end of line
1 < = ## < = 255
CHR$(27)"0" — Sets line spacing to 9 dots = 9/72 inch
CHR$(27)"1" — Sets line spacing to 7 dots = 7/72 inch
CHR$(27)"2" — Returns line spacing to 12 dots = 1/6 inch
CHR$(27)"3"CHR$(##) — Sets line spacing to ##/216 inch
Stays on until turned off
1 < = ## > = 255
CHR$(8) — Backspace
CHR$(9) — Horizontal tab
CHR$(10) — Line feed
CHR$(12) — Form feed
CHR$(13) — Carriage return

Chapter 12

Introduction to Dot Graphics

Caution — Entering High Resolution Space

To create dot graphics on the printer, we must enter a completely new world: graphics mode. In fact, our printer has two graphics modes: high resolution and super-high resolution.

We will investigate them one at a time starting with high resolution (known in the biz as "hi-res" graphics). In this mode, predefined characters do not exist — only dots. We create our own characters or images by arranging the dots however we wish.

In the graphics mode we have complete control over which pins fire, and when. It sounds like a lot of fun, and is, but there are an awful lot of dots just in one line, let alone an entire page.

Think of it this way. There are 80 "regular" characters in a normal line (row) on narrow width paper. Each character is 6 dots wide (5 dots plus 1 blank column). That comes to 480 columns in each row. On wider paper, there are 136 characters using 6 dots per line apiece, or 816 columns per line. Good grief!

Fortunately, we don't have to fill all the columns in every line. In fact, the first thing we do when entering graphics mode is to tell the printer how many columns of dots we will send it, a row at a time.

Warp 3

Graphics mode is entered by sending the printer a cluster or sequence of 4 codes:

Format: <ESC>"K" N1 N2

Example: CHR$(27)"K" CHR$(7) CHR$(1)

CHR$(27) is of course the ESCAPE code, and <ESC>"K" means kick into graphics mode.

CHR$(7) says: "reserve 7 columns for graphics". N1 can range from 0 to 255 (0-127 for 7-bit computers).

CHR$(1) says: "My job is to make reservations for really big crowds of dots. Multiply me by 256, then add me to N1 for the correct number of columns to

expect. If you don't need me right now, just call me 0." In our example 1 × 256 = 256.

N2 can range from 0 to 7. Numbers greater than 7 are interpreted MODULO 8. That's just math talk meaning the printer interprets them as the remainder after they are divided by 8. For example:

8 & 16 & 24 . . . each equals 0 modulo 8. (There's 0 remainder after any of them are divided by 8.)

9 & 17 & 25 . . . each equals 1 modulo 8. (The remainder is 1 after any of them are divided by 8.)

15 & 23 & 31 . . . each equals 7 modulo 8, etc.

The maximum number of columns that can be specified is 7 * 256 = 1792 + N1 (255) = 2047. Problem is, wide paper is only wide enough for 816 columns, and narrow paper is only wide enough for 480 columns.

So, in the above example with N1 = 7 and N2 = 1, the number of columns requested is:

N1 + 256 * N2 = (7) + 256 * (1) = 263 columns of graphics.

Note that if this sum exceeds 480 (or 816), only the appropriate number of columns will actually be printed. The number 480 is specified by N1 = 224 and N2 = 1. 816 is specified by N1 = 48 and N2 = 3. Also, the printer will "hang-up" if we do not send the number of graphic bytes specified in the <ESC>"K" sequence.

The saki flowed like melting snow from Mt. Fuji when they contrived that wild scheme. Let's see if we can wring it out.

Suppose we want to shoot 100 columns of graphics to the printer. What numbers will we use for N1 nd N2?

```
CHR$(100)CHR$(0)          for 100 columns
```

OK, so we got lucky.

How about 300 columns? Well, 300 - 256 = 44. So let's try:

```
CHR$(44)CHR$(1)           for 300 columns
```

Since N2 = CHR$(1), 1 * 256 is added to N1. 44 + 256 = 300

Not really so bad. Pass the saki!

Enter these lines, but don't RUN:

```
NEW

  9 PR#1                          (Apple)

  10 LPRINT CHR$(27)"K"CHR$(50)CHR$(0);
```

TRS-80 Model I use:

```
  10 LPRINT CHR$(27)"K"CHR$(50)CHR$(8);
```

Model I can't send CHR$(0) reliably, and 8 is interpreted as a zero for N2, per the "modulo" explanation above.

Don't forget the semicolon or there will be lots of problems. The printer is expecting 50 columns of graphics code in the same or very next line. The semicolon of course connects print lines together.

How many columns is line 10 reserving, 50 or 306?

(Answer = 50). Don't RUN yet.

Firing the Pins

There are 9 pins in the print head. In graphics mode, we control the top 8. (The middle 7 for 7-bit computers.)

We'll label these top 8 pins as follows:

128 - o	Top
64 - o	
32 - o	
16 - o	
8 - o	
4 - o	
2 - o	
1 - o	Bottom
o	(Ninth pin not used for graphics)

From now on we will refer to the second pin (pin 1 above) as the bottom pin when using graphics.

Why not label the pins 1,2,3,4,. . .8? Well, the numbers shown are the actual ASCII numbers that fire the respective pins. CHR$(128) fires the top pin, while CHR$(1) fires the bottom one. CHR$(7) fires the bottom three (4+2+1).

It all goes back to binary math. (Oh, great!) If we send a decimal 1 (0000 0001 binary), the bottom graphics pin fires. A decimal 2 (0000 0010 binary) fires pin 2. A decimal 3 (0000 0011 binary) fires pins 1 and 2 and so on.

Add these lines:

```
20 FOR J=1 TO 50
40   LPRINT CHR$(1);
60 NEXT J
70 LPRINT
99 PR#0          (Apple)
```

and RUN.

Figure 12-1

Sure enough, fifty little dots. CHR$(1) in line 40 caused the bottom pin to fire. The semicolon suppressed the line feeds. The for-next loop fired the bottom pin 50 times.

Change line 40 to:

```
40   LPRINT CHR$(127);
```

and RUN.

Figure 12-2

Wow! We hit the jackpot. Using the bottom of an old Coke bottle as a magnifier, we see that each column is 7 dots high. (Remember we are using only 7 pins.) There are 50 columns of 7 rows of dots. The sum of the bottom 7 pin numbers is:

1 + 2 + 4 + 8 + 16 + 32 + 64 = 127.

(or 127 = 0111 1111 binary)

So that's how they do it!

Quiz Time

Question: What single decimal number will allow us to fire only the pins labeled 1, 4, and 16?

Think it through now . . . how about 1 + 4 + 16 = 21?

Let's try

```
40   LPRINT CHR$(21);
```

and RUN.

Figure 12-3

Oh — winning is so much fun!

Don't be too concerned about what movie is playing on your video screen if you're using PRINT instead of LPRINT. The video may not know how to handle these codes. It's what's happening on the printer that counts.

The Grand Scheme

This is starting to make sense. The "logic pattern" must be:

Pin Labels	0	1	2	3	4	5	6	7	8	9	10	11	12	----	255	
128															●	128
64															●	64
32															●	32
16															●	16
8									●	●	●	●	●		●	8
4					●	●	●	●					●		●	4
2			●	●			●	●			●	●			●	2
1		●		●		●		●		●		●			●	1
	0	1	2	3	4	5	6	7	8	9	10	11	12	----	255	

Graphic Codes

If you're not a real whizzo at binary math, a few minutes studying the above chart along with what else we've learned in this chapter may turn on that big light upstairs.

Now for the Bad News

Before racing off to create a HI-RES forgery of the Mona Lisa, be aware that the above is how it is supposed to work. Reality is brutal — there are some exceptions to almost every rule.

Code numbers 9 and 13 for the Apple, and 0, 10, 11, and 12, for the TRS-80 create havoc when they are used as N1 or N2. (It's deja vu from the last chapter.) Other computers may have trouble with different codes. Even worse, the printer accepts codes from 0 to 255 to determine the dot pattern, but many computers are not able to SEND codes greater than 127. It is the old "missing eighth bit" caper. Like the "lost chord".

Sigh! Once again the printer can out perform many computers. In Apple's case, the parallel interface card deactivates the eighth bit so Apple users can control only 7 pins. Using <ESC>">", 7-bit computers can use numbers from 128 to 255 for N1 and N2, but the printer steadfastly refuses to fire more than 7 pins at a time. Rats!

Pinpoint Control

We've seen that each pin is associated with a number. The numbers 1, 2, 4, 8, 16, 32, 64, 128 all relate to the mathematical powers of 2. Here's the relationship:

$2^7 = 128$
$2^6 = 64$
$2^5 = 32$
$2^4 = 16$
$2^3 = 8$
$2^2 = 4$
$2^1 = 2$
$2^0 = 1$

Let's see if we can fire the pins, in sequence, from the bottom up. Make the program read:

```
9 PR#1                                    (Apple)

10 LPRINT CHR$(27)"K" CHR$(94)CHR$(1);
```

```
20 FOR J=1 TO 50
30 FOR I=0 TO 6
40 LPRINT CHR$(2^I);        (^ means exponentiate)
50 NEXT I
60 NEXT J
70 LPRINT
99 PR#0                              (Apple)
```

and RUN

Note how the buffer fills, then drops its codes to paper via the print head, and the process repeats . . .

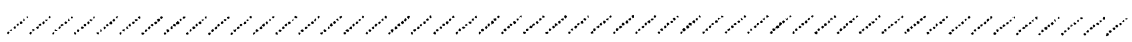


Figure 12-4

Very impressive.

Note that we are not printing the slash as a preprogrammed character (ASCII number 47); but are actually positioning the dots one at a time.

Sine Waves

This one's for the math types. Follow it if you can, for it **is** a helpful example. If you don't understand it, it's not the end of your career with computers and printers.

We know that to fire a single pin, we send the printer a number that represents its respective power of 2. Line 40 (below) varies the power of the number 2 in accordance with the trigonometric sine function. Change or add the following program lines:

```
10 LPRINT CHR$(27)"K" CHR$(85)CHR$(1);
20 FOR J=1 TO 11
30 FOR I=0 TO 6 STEP .2
```

```
40 LPRINT CHR$(2^INT(3.4*(SIN(I)+1)));  (^ = Exponent)

80 GOTO 10
```

and RUN.

Let it print a few lines while we study the next paragraph. Don't look at the printout too long or you may get seasick.

Figure 12-5

In line 40, the entire expression to the right of the exponent sign chooses an appropriate exponent for 2. The exponents determine which pin is fired, giving us a nice sinusoidal curve. Really rather clever! If you don't follow the math, just nod your head knowingly, and bluff. It doesn't really matter here.

<BREAK> or <CTRL-C> the program and cycle the printer OFF then ON again.

Who Sez we Ain't Got no Class??

Here are some "aerobic exercises" to wring out the entire system. Enter this new program:

```
9 PR#1                                (Apple)

10 LPRINT CHR$(27)"A"CHR$(7)

20 FOR R=1 TO 3

30 LPRINT TAB(10);

40 LPRINT CHR$(27)"K"CHR$(14)CHR$(0);

50 FOR N=1 TO 14

60 READ D : LPRINT CHR$(D);

70 NEXT N : LPRINT : NEXT R

80 LPRINT CHR$(27)"2"
```

```
99 PR#Ø                              (Apple)
1ØØ DATA 4,4,4,4,4,61,71,7Ø,6Ø,4,4,4,4,4
11Ø DATA 72,79,89,1Ø7,77,73,127,89,73,79,73,73,73,73
12Ø DATA 16,112,54,17,81,17,126,16,48,8Ø,16,16,16,16
```

TRS-80 Model I:

```
4Ø LPRINT CHR$(27)"K"CHR$(14)CHR$(8);
```

and run.

Figure 12-6

How about that one, culture vultures? Here's how it works:

Line 10 sets the line spacing for 7-pin graphics. The G clef is printed in three lines.

Line 20 takes care of the looping.

Line 30 moves the figure away from the left margin.

Line 40 kicks the printer into graphics mode - 14 columns per line.

Lines 50 - 70 read each code from the DATA lines and send it to the printer.

The NEXT R in line 70 sends control back to line 20 to start the next line.

Line 80 returns the line spacing back to 6 lines per inch.

And that's the name of that tune.

Code Summary

<ESC>"K" N1 N2 - Enters graphics mode. N1 and N2 determine length of graphic line.
0 < = N1 < = 255
0 < = N2 < = 255
N2 > 7 is interpreted MODULO 8.

Chapter 13

Advanced Graphics

Firing Pins at Seven Paces

The MX printer allows any computer to control the firing of its 8 active pins by simply sending it ASCII code numbers between 0 and 255. Conversely, these 256 code numbers are sufficient to control all 8 pins. Multiline graphics programs will therefore usually be done with a top-of-line to top-of-line spacing of 8 dots (8/72 inch).

As mentioned several times now, many computers, or their printer interfaces, will not pass code numbers greater than 127. The <ESC>">" feature allows these computers to send codes up to 255 when entering the graphics mode, but they can control only 7 of the 8 active pins in graphics mode. With that common computer deficiency in mind, the examples in the remaining chapters are limited to 7 dots per line.

The limitaton is not a crippling one. We just have to adjust for it. (The same field of corn can be harvested by a 2-row corn picker as by a 4-row corn picker — it just takes longer.)

Users with 'superior' computers can follow right along at our 7 dot clip, knowing that your computers can use all 8 dots to cut a wider swath, but not until we've all finished this course.

Housekeeping

Let's develop a fairly complex graphics picture a step at a time. Everybody start by typing in these lines:

```
NEW

9 PR#1                                    (Apple)

10 LPRINT CHR$(27)"A"CHR$(7)

20 N=50

30 LPRINT CHR$(27)"K"CHR$(N)CHR$(0);

(TRS MODEL I use 30 LPRINT CHR$(27)"K"CHR$(N)CHR$(8);)
```

but don't RUN yet.

Line 10 sets the top-to-top line spacing at 7 dots.

Lines 20 and 30 KICK the printer into graphics mode. N specifies the number of graphic columns. By making it a variable, we can easily change it later, right within the program, to print graphic lines of different lengths.

As a preliminary software test before things get too complicated, we'd better check the line spacing. Add these lines:

```
25 FOR X=1 TO 3

40 FOR I=1 TO 50: LPRINT CHR$(127); : NEXT I

50 LPRINT : NEXT X
```

We should always return the line spacing to its "normal" power-up figure of 12 by ending every program with:

```
900 LPRINT : LPRINT CHR$(27)"@"    (or <ESC>"2")

999 END                            (Apple use PR#0)
```

Now we can RUN.

After 3 passes of the print head, our printout should look like this:

Figure 13-1

What are we Trying to do?

We are creating a universal program that will READ and process large amounts of DATA, and is simple to use. READ and DATA statements are used since DATA need be entered in the program only once, and DATA lines are easily edited. Think about this paragraph and understand it since the going will get rougher before it gets easier.

Let's delete lines 25 and 50 and change line 20 so it will READ in N, the number of dots to be printed in a specific row:

```
20 READ N: LPRINT TAB(10);
```

First National DATA Bank

HI RESolution graphics require lots of DATA. It comes with the territory.

So, what should our DATA lines look like? How should we format them so they contain all the information we need in a reasonably concise fashion?

We know the ASCII numbers 0 through 127 are used to specify the firing combinations for the 7 pins. It's easy to READ them in one at a time and shuttle them along to the printer. But occasionally, we'll need to send a single one of these numbers to the printer several times in succession. To avoid needless typing, let's use a minus sign as a flag to tell the computer that its number represents the number of times something is to be repeated. The next number of the pair will specify the desired combination of pins to be fired.

For example:

```
100 DATA -42,127
```

means: "fire all 7 pins (#127) 42 times"

Think it through before continuing.

Got it?

We also have to tell the printer the number of columns to print on each line. Let's dedicate the first number in each DATA line to that purpose.

Enter this line of data and we'll try out the system:

```
1020 DATA 55,0,3,7,15,31,63,63,-42,127,63,63,31,15,7,3
```

Recheck the numbers carefully to be sure they are copied correctly.

The first number tells the printer to expect 55 bytes of graphic information. The rest of the numbers specify pin combinations to be fired in each column, except for the suspicious number pair in the center: -42,127. Remember what they mean?

The Program

We also need a "DO LOOP" to read in the data:

```
40 FOR G=1 TO N

50 READ X
```

```
120 NEXT G

130 LPRINT
```

The numbers read into X are the actual pin firing instructions. If X falls in the range 0–127, we print it in line 60 below. Line 60 also includes a "filter" to snag negative numbers:

```
60 IF X>=0 THEN LPRINT CHR$(X); : GOTO 120
```

TRS-80 Model I:

```
55 IF PEEK(14312)<>63 THEN 55

60 IF X>=0 POKE 14312,X : GOTO 120
```

If X is negative:

The line 60 test fails and execution "falls through" to

Line 70, below, where the "G" FOR/NEXT loop is reset and the next value is READ into Y from the DATA line.

Line 100 LPRINTs it ABS(X) times:

Add these lines:

```
70 G = G-X-1

80 READ Y

90 FOR I=1 TO ABS(X)

100 LPRINT CHR$(Y);

110 NEXT I
```

TRS Model I:

```
100 POKE 14312,Y

105 IF PEEK(14312)<>63 THEN 105
```

LIST and recheck the completed program to make sure everything is correct. Here's what it should look like:

```
9 PR#1                                    (Apple only)
10 LPRINT CHR$(27)"A"CHR$(7)
20 READ N: LPRINT TAB(10);
30 LPRINT CHR$(27)"K"CHR$(N)CHR$(0);
40 FOR G=1 TO N
50 READ X
60 IF X>=0 THEN LPRINT CHR$(X); : GOTO 120
70 G = G-X-1
80 READ Y
90 FOR I=1 TO ABS(X)
100 LPRINT CHR$(Y);
110 NEXT I
120 NEXT G
130 LPRINT
900 LPRINT : LPRINT CHR$(27)"@"
999 END                                   (Apple use PR#0)
1020 DATA 55,0,3,7,15,31,63,63,-42,127,63,63,31,15,7,3
```

Summary of TRS-80 Model I changes:

```
30 LPRINT CHR$(27)"K"CHR$(N)CHR$(8);
55 IF PEEK(14312)<>63 THEN 55
60 IF X>=0 POKE 14312,X : GOTO 120
100 POKE 14312,Y
105 IF PEEK(14312)<>63 THEN 105
```

Finally, we are ready to try it out . . . so run.

Figure 13-2

Big deal! Maybe if we water it, it will grow.

All that Work for one Line?

We shouldn't complain too loudly. It works, and no one said creating graphics was easy.

Our current "universal" program can only handle one line of data, and it's becoming increasingly apparent that HI-RES graphics requires lots of data. The following changes allow it to handle virtually unlimited amounts of data:

```
15 ON ERROR GOTO 900          (or ON ERR GOTO)

130 LPRINT : GOTO 20
```

Line 130 creates an infinite loop, sending execution back to the READ statement after printing each line.

Line 15 gives the program a smooth ENDing when the data is all read. When there is no more data, an OUT-OF-DATA error message appears, and the program is cleanly ended, even resetting the printer back to its power-up defaults.

Let's add 2 more lines of data to see how it works:

```
1000 DATA 49,-8,0,3,15,63,-34,0,63,15,3,0

1010 DATA 49,-7,0,63,-3,127,7,-32,0,7,-3,127,63
```

and RUN.

Figure 13-3

It sprouted flowers!

Rowdy Characters

Add the following lines:

```
1070 DATA 44,-12,8,-32,127
1080 DATA 39,-17,0,64,96,112,120,124,-12,127,124,120,
          112,96,64
```

and RUN.

Figure 13-4

Oops! There's trouble in River City. (Note the non-symmetry.) Fortunately, we've seen these same control code problems before. Hate to keep bringing this subject up, but it does disturb our use of the printer.

We've seen codes 9 and 13 cause trouble on the Apple, and codes 0, 10, 11, and 12 cause trouble on the TRS-80. If we send them to the printer via CHR$, it interprets them as control codes instead of pin firing instructions. Unfortunately, that makes sense.

Code 8 can also cause trouble since the computer uses it as a backspace control character. In line 1070, we tried to send a whole sequence of them, and on some computers the program blew up! Different computers will create different terrible pictures.

For Experts Only

The errors caused by these codes do not always occur, but it is best to avoid them if possible. Short of rewriting the program to circumvent use of these numbers, one way is to bypass the CHR$ function and POKE our graphic codes directly to the printer driver in the computer's memory. That memory address is 49296 (if the printer card is in slot 1) in the Apple, 14312 on the TRS-80, Model I, and Model III'ers try OUT 251, ### if necessary. Other computer experts will have to check their computer's technical manual for its printer driver memory address.

The advantage of POKE over LPRINT CHR$ in this specific situation is that it bypasses some of the nasty tricks we keep encountering with BASIC. The disadvantage is that indiscriminate POKEing can cause unbelievable software crashes. Unless you really understand POKEs, best live with the limitations inherent with your computer/printer combination, and avoid troublesome codes when possible. Model I users will need to use POKE for this particular example.

A Confession

Line 1070 was artifically injected into this conversation to illustrate some of the problems the rebel character codes can cause. We're playing for real now, so change line 1070 to:

```
1070 DATA 44,-12,0,64,112,120,124,127,127,15,7,67,99,51,3,
          63,123,71,3,3,71,123,63,3,51,99,67,7,15,127,127,
          124,120,112,64
```

and RUN.

Figure 13-5

Ah so! Turned up-side down it looks like a flying saucer.

NOTE: Since the maximum line length on an Atari is 120 characters, data lines 1040, 1060, and 1070 will have to be broken into 2 data lines. This will not affect the program run as long as the numbers remain in the same order.

To see the final picture we have to type a few more lines. (Think of it as the price of greatness.)

```
1030 DATA 56,-11,127,123,124,126,-28,127,126,124,123,
          -11,127
1040 DATA 55,0,96,120,127,126,-7,127,63,15,3,125,112,120,
          124,126,-16,127,126,124,120,112,125,3,15,63,
          -7,127,126,127,120,96
```

```
1050 DATA 52,-4,0,112,60,94,111,119,119,123,123,-32,127,
          123,123,119,119,111,94,60,112
1060 DATA 49,-7,0,64,112,120,124,-7,127,120,119,120,115,
          116,120,112,112,113,112,112,113,112,112,120,116,
          115,120,119,120,-7,127,124,120,112,64
```

Whew! Lots of data. The price of glory is high, but this is ridiculous.

Cross your fingers and RUN.

Figure 13-6

It was worth it! We now have the tools to design our own graphics. Be sure to save the finished program as we will use it in the next chapter. Do it now!

Super High Resolution Graphics

Hold the phone. We ain't seen nothing yet. The printer sports another mode of high resolution graphics that is **twice** as dense horizontally. In place of the 480 columns (816 columns per wide sheet) we can make it print 960 (or 1632) dots in super hi res mode.

How do we do it? Like most things, it's really simple. Each time the print head fires, it moves over only **half** the normal distance. The dots overlap as shown in Figure 13-7.

Figure 13-7

To enter this super graphics mode, we use

```
<ESC>"L" N1 N2.
```

N1 and N2 have the same meaning as in normal <ESC>"K" sequence. The code letter "L" simply replaces the letter "K". To reserve the entire 960 columns for graphics, use N1 = 192 and N2 = 3. For 1632 columns, use N1 = 96 and N2 = 6.

Note that the printer cannot print more than 960 columns on the MX-80 (1632 on the MX-100). If we specify more than 960 (1632) columns (e.g. using N2 = 7), the printer will just wait until that many codes are delivered. It will, of course, only print the first 960 (1632) columns.

Let's try it with the current program. Change line 30 to:

```
30 LPRINT CHR$(27)"L"CHR$(N)CHR$(0);
```

and RUN.

Figure 13-8

Wow! It sure packs 'em in tight. In fact, this change totally altered the "aspect ratio" of our figure. Can you think of a way we could quickly regain the original aspect ratio in this new graphics mode? Sure. Just double up on the printing. Change:

```
30 LPRINT CHR$(27)"L"CHR$(2*N)CHR$(0);

60 IF X>=0 THEN LPRINT CHR$(X)CHR$(X); : GOTO 120

100 LPRINT CHR$(Y)CHR$(Y);
```

and RUN.

Figure 13-9

Line 30 reserves twice as many columns for graphics, and lines 60 and 100 oblige by printing each character twice.

Shady Characters

This super hi res mode paves the way to very detailed graphics. With one-half dot horizontal spacing and one-third dot vertical spacing, we have very tight control of the print head. Our graphic creations are limited only by the size of the dots and our imaginations (. . . to coin a phrase).

Horizontal double-density graphics can be used to produce shading similar to that done in newspapers. The following program prints a sort of "gray scale" to give the effect of changing shades. Now we can print several hundred sheets with this program to wallpaper your new computer room. Enter:

```
NEW

9 PR#1                                  (Apple)

10 DIM A(16) : E$=CHR$(27) : LPRINTE$"A"CHR$(7)

20 FOR I=1 TO 16 : READ A(I) : NEXT I

30 DATA 127,88,45,37,120,3,15,78,82,98,115,18,51,14,101,1

40 FOR K=1 TO 10 : FOR J=1 TO 10

50 LPRINT E$"L"CHR$(31)CHR$(8);

60 FOR I=1 TO 16 : LPRINTCHR$(A(I));

80 NEXT I

90 FOR I=16 TO 1 STEP -1 : LPRINTCHR$(A(I));

110 NEXT I : NEXT J : LPRINT : NEXT K

120 LPRINT E$"@"
```

and RUN.

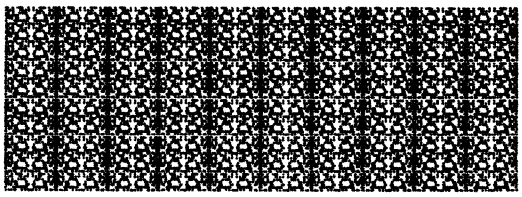


Figure 13-10

Yes, it is slow as molasses. Let's see if we can figure out the method behind the madness while it prints.

Array A(I) stores the codes in line 20 that are then used in line 60 to print the pattern shown in Figure 13-10. This pattern is repeated 100 times by the FOR NEXT loops in lines 40 and 110.

Figure 13-11 shows another example in which the dot pattern is random instead of fixed.

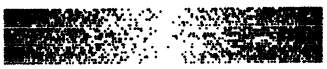


Figure 13-11

If this high density stuff really excites you, imagine what we can do with the vertical spacing set to 1/216 inch! Whew, time for a cold shower.

Code Summary

49296 — Poke location for Apple to send information to the printer.

14312 — Poke location for TRS Model I

251 — Out location for TRS-80 Model III

<ESC>"L" N1 N2 — Enters double density graphics mode. N1 and N2 determine length of graphic line.

Chapter 14

The Final Push

Only true masochists failed to save the DEMON program from the last chapter (Figure 13-6). We'll now delete its DATA lines and reconstruct it for more universal applications. Delete lines 1000-1080.

Mix and Match

The graphics capability of the MX printers is truly astounding. As we begin to fully comprehend the power and versatility at our finger tips, it's natural to wonder about mixing text and graphics on the same line. Can it be done? Sure, nothing to it. Just be sure to hang trailing semicolons on the ends of our program lines as needed to keep the printer from doing any unwanted line feeds.

A few swift chops to our existing program will demonstrate the point. We can use it as a subroutine to read and print graphics strings.

Change the following:

```
10 LPRINT CHR$(27)"A"CHR$(10)          or 10+128

15 GOTO 140

130 RETURN
```

Line 10 increases the line spacing to give our next figure a little breathing room (10 dot line height).

Lines 15 and 130 turn the first part of the program into a subroutine.

We'll start with a single line of graphics to set the stage:

```
140 GOSUB 20 : LPRINT

150 DATA 60,-27,127,126,100,64,1,11,63,-27,127
```

and RUN.

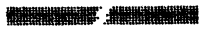


Figure 14-1

OK. The subroutine works.

No, it's not another demon. It is broken sign-board that needs a label. The sign will be printed in these three parts: 15 columns of graphics, 30 columns of text (5 characters), and 15 columns of graphics.

Now we have to mix and match graphics and text on the same line. First the graphics:

```
160 GOSUB 20

170 DATA 15,127,8,28,62,93,-8,28,0,0
```

and RUN.

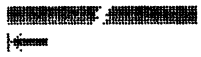


Figure 14-2

Then text:

```
180 LPRINT "82 CM";
```

and RUN.

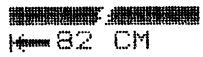


Figure 14-3

And more graphics:

```
190 READ N : GOSUB 30

200 DATA 15,0,0,-8,28,93,62,28,8,127
```

and RUN.

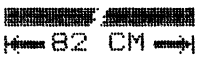


Figure 14-4

Nothing to it!

NOTE: As long as semicolons are used to delay the line feeds, graphics and text are easily mixed on the same line. New lines can be started by simply adding line feeds with LPRINT statements as shown in line 140.

And while we're at it, if the printer is in some exotic text mode when we enter graphics, the printer will return to that mode when it is done printing graphics. The exception is the superscripting/subscripting mode. When we use graphics, we are kicked out of scripting mode, but remain in double strike.

The Long Lines Division

None of our examples so far have used the full 480 (816 on the MX-100) columns available to the printer. In fact, we've always used 100 or less since the printer will be used by computers with different capabilities. As we expand the widths of our displays, these differences quickly become obvious.

TRS-80 type computers, for example, can send the full range of codes, 0-255, (8 bits) via the CHR$ function. (A few must be poked, as we've seen.) These computers can therefore choose any graphics width from 0 to 480 (816) columns without a problem.

Printer cards for Apple II type computers typically pass ASCII numbers from 0 to 127 (7 bits). This forces us into extra programming and devious means to utilize the complete graphics range of the printer.

In any one <ESC>"K" sequence, 7-bit computers are restricted to the ranges 0-127, 256-383, 512-639, and 768-816. Seeing is believing, so delete lines 150-200, and add:

```
150 DATA 150,-150,127
```

and RUN.

Figure 14-5

Here's what happened. The 150 is reduced by 128, and sent to the printer as 22. Owners of 7-bit computers see a bar 22 columns long (did you count them?). Users with 8-bit computers see a bar 150 columns long.

If we really understand what we're doing, the 7-bit limitation can be overcome. Apple users add:

```
12 PRINT CHR$(27)">"
```

and RUN.

Figure 14-6

Much better, but don't get too excited. We still can't fire more than 7 pins — even with <ESC>"<". Here is an alternate solution: delete 12, and add:

```
20 READ N

140 GOSUB 20: GOSUB 20: PRINT

150 DATA 23,-23,127,127,-127,127
```

and RUN.

Figure 14-7

Now we're getting somewhere. Instead of trying to print the entire line in one shot, we hooked the two print routines together with a semicolon. The first printed 23 columns, and the second printed 127 columns.

Really Long Lines

Suppose we want to use even longer lines, up to the maximum of 480 columns (816 on wide paper). To specify a line length greater than 255, the last number in the <ESC>"K" N1 N2 sequence must be 1 or greater. We can accommodate the entire range of line widths by changing:

```
25 N2=INT(N/256) : N1=N-256*N2

30 LPRINTCHR$(27)"K"CHR$(N1)CHR$(N2);
```

If the desired line width (N) is greater than 255, CHR$(N2) adds 256*N2 columns, and CHR$(N1) takes care of the difference. Let's test this upgrade by changing:

```
10 LPRINT CHR$(27)"A"CHR$(7)

140 GOSUB 20: LPRINT

150 DATA 300,-300,127
```

and RUN.

Figure 14-8

We don't have to count the dots to check it. There are 60 dots per inch, so the line should be 5 inches long.

Problems, Problems

I'm afraid that some users wound up with a slightly different result, even though 300 should be valid for the current program. For example:

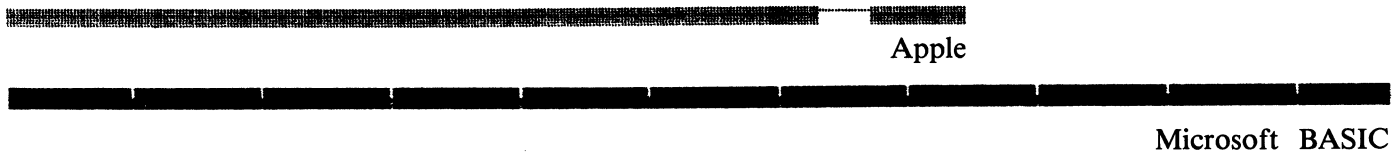


Figure 14-9

It seems that anything printed past certain column limitations gives us scrambled eggs. The limitation depends on the computer. Sample solutions are:

```
12 PRINT CHR$(9);"255H"     (Apple)

12 WIDTH LPRINT 255         (Microsoft BASIC)
```

and RUN.

Figure 14-10

Why the Apple solution works is still a mystery to those that found it, but it does work! Use it whenever your Apple programs require continuous graphic strings of more than 255 columns. (Note: this helpful trick has its drawbacks. See Appendix J for more information.)

Apple Pie

But fear not! We couldn't end this happy tale on a sour note. EPSON has provided a far superior way for Apple users to create high resolution graphics in BASIC. A screen dump program for use with Apple disk systems completely eliminates the need for wrestling with the above problems. Apple disk users can either GOTO Chapter 15 right now for this solution, or continue on with the rest of us. Screen dump routines for the Atari are rumored, and your computer dealer will probably have them for other computers as well.

Pictogram

Let's see if we can parlay strings of graphic characters into a useful format. This new program prints strings of graphics in short bursts; a technique that works on everyone's computer. (Famous last words #5,729.) Enter:

```
5 WIDTH LPRINT 255                    (Microsoft BASIC)

9 PR#1                                (Apple)

30 FOR I=1 TO 5: READ A(I): NEXT I

40 DATA 8,107,92,107,8

60 FOR R=1 TO 5

80 READ N

90 FOR I=1 TO N: GOSUB 300

100 NEXT I: LPRINT

120 NEXT R

130 DATA 30,25,17,11,8

140 LPRINT CHR$(27)"@"

198 PR#0                              (Apple)

199 END
```

```
300 LPRINT " "CHR$(27)"K"CHR$(5)CHR$(0);
310 FOR C=1 TO 5
320 LPRINT CHR$(A(C));
330 NEXT C: RETURN
```

TRS-80 Model I:

```
300 LPRINT " "CHR$(27)"K"CHR$(5)CHR$(8);
```

and RUN.

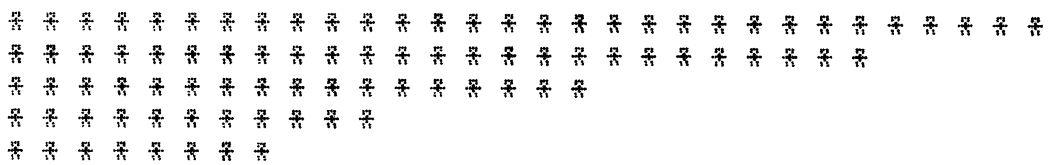


Figure 14-11

The data for the population figure is read into array A in line 30.

Each figure is printed in subroutine 300.

Data line 130 determines how many figures are printed on each row.

We have the makings of a horizontal bar graph, but it needs some trimmings. Let's add a few labels and move the whole works over a bit.

```
20 LPRINT: LPRINT CHR$(27)"A"CHR$(7)
50 GOSUB 200: LPRINT
70 LPRINT TAB(5)1990-R;
80 GOSUB 200: READ N
110 GOSUB 200: LPRINT
200 LPRINT TAB(12)CHR$(27)"K"CHR$(5)CHR$(0);
210 FOR I=1 TO 5:  LPRINT CHR$(127); : NEXT I
220 RETURN
```

TRS-80 Model I:

```
200 LPRINT TAB(12)CHR$(27)"K"CHR$(5)CHR$(8);
```

and RUN.

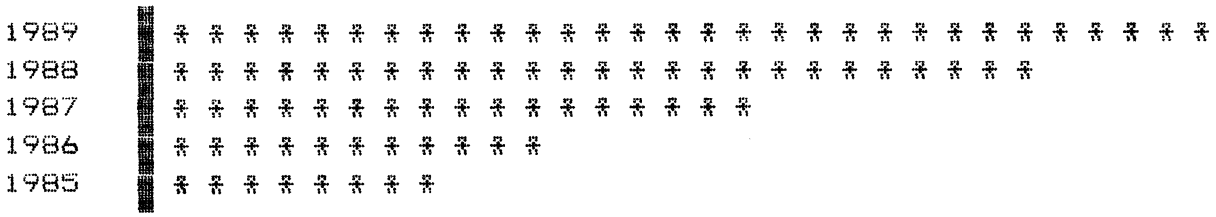


Figure 14-12

That's better. Subroutine 200 prints the vertical bar that separates the labels and the graph.

If we add a quick title and legend, we'll be done.

```
10 LPRINT TAB(23)CHR$(14)"POPULATION PROJECTION"

140 LPRINT: LPRINT TAB(37);: GOSUB 300

150 LPRINT " = 1,000"
```

and RUN.

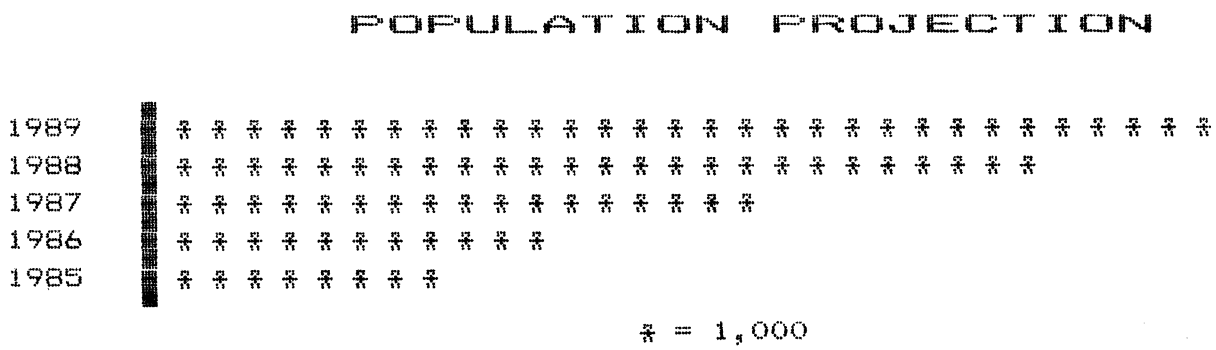


Figure 14-13

Let Those Creative Juices Flow

Now that we've seen a variety of examples of how high resolution graphics can be used, it's your turn to create solutions to your own needs. The applications are virtually unlimited.

Figure 14-14 shows a sample of several different character fonts created by one user.

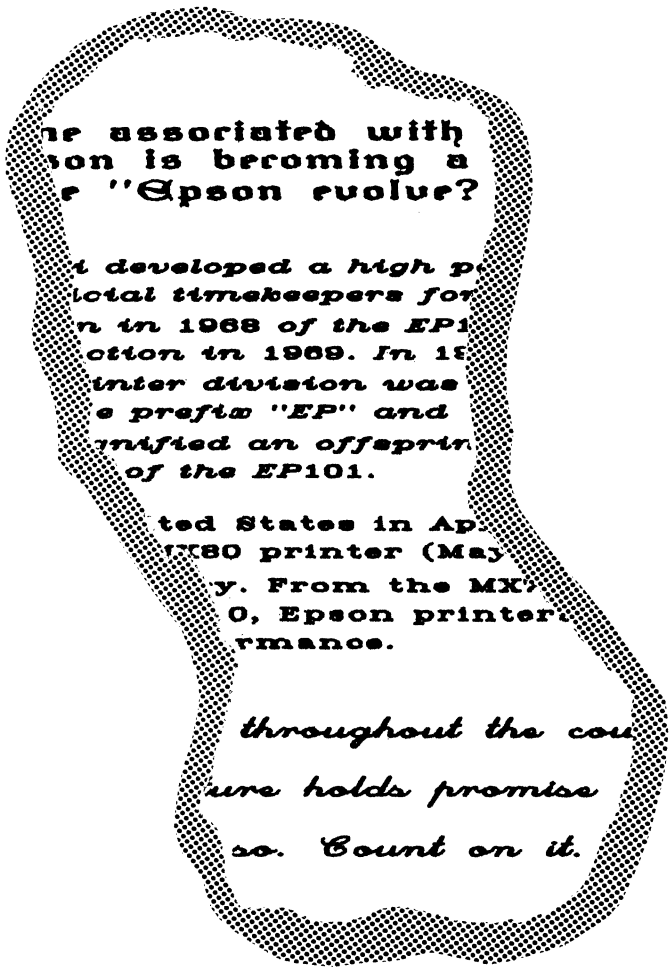


Figure 14-14

Happy pin firing!

Code Summary

PRINT CHR$(9) "255H" — Allows Apple users to print past column 255 without heartburn.

WIDTH LPRINT 255 — Allows Apple Microsoft BASIC users to print past column 255 without heartburn.

Chapter 15

Using the HI-RES Screen Dump Program

Should I Study This Chapter?

EPSON dealers can supply a machine language screen-dump diskette for Apple users with 48K disk systems. It permits creating exotic hi-res graphics on the Apple screen, then dumping them, dot for dot, to the MX printer. We show it here as another example of how computer software can take advantage of the printer's capability.

Since the hi-res printer graphics are related to the Apple's (much as the original MX-80's graphics replicate the TRS-80's), this particular software is of interest mainly to Apple owners. Knowing this crazy field however, some ingenious software writer working in his attic will probably come up with a partial hi-res program for use with other popular computers.

Meanwhile, if another brand of computer is feeding your MX, you can just as well ignore this chapter. Go in peace, and enjoy your printer.

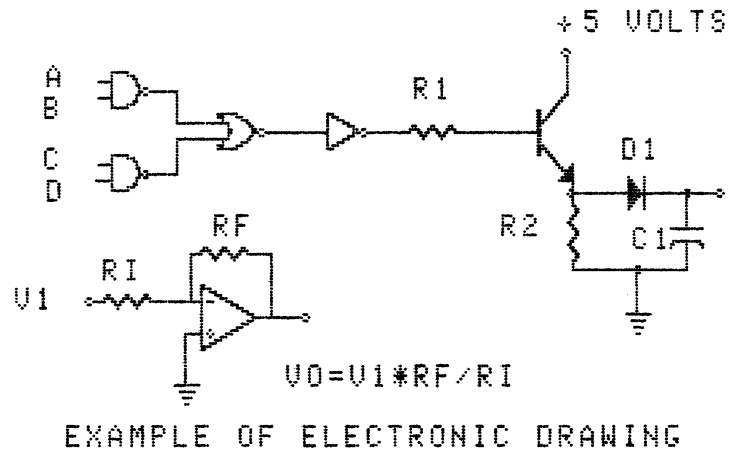


Figure 15-1

Apple Dump

The screen-dump diskette comes complete with an operating system and requires only a single disk drive.

Load the demonstration diskette in drive 1 and boot the system. The HELLO program will automatically load and execute. After a brief commercial from the copyright holder, the program asks:

```
IS THE PICTURE ALREADY IN MEMORY? (Y/N)?
```

We respond with:

```
N    <RETURN>
```

The computer retaliates with:

```
INSERT DISK CONTAINING PICTURE
THEN PRESS RETURN
?
```

Since the DOS and sample pictures are on the same diskette, press

```
<RETURN>
```

What follows is the HIRES diskette catalog of sample programs and files. The catalog includes these programs: HELLO, HIRES INSTRUCTIONS, EPSON.HIRES.OBJ. The HELLO program loads the screen dump routine and prompts us with questions. This is the program we are running now. The HIRES INSTRUCTIONS program gives hints on how to use the screen dump. The EPSON.HIRES.OBJ program is the actual screen dump routine written in machine code. It can be accessed with the USR command from BASIC.

The rest of the files are hi-res demonstration pictures, and at this writing include: BESSEL 2, BESSEL, INVADERS, EPSON DEMO PIC, CHART, DISNEY CHAR, SCHEMATIC, PORTRAIT, BURT, and CHESS.

Follow the Yellow Brick Road

The HELLO program guides us through the actual process of printing the pictures. It's completely self-explanatory (as every confusing technical book says), but here's some step-by-step guidance.

```
WHAT IS THE FILENAME OF THE PICTURE
?
```

The program is asking for the name of the picture we want to print. Let's try:

```
DISNEY CHAR     <RETURN>
```

The computer reports:

```
LOADING PICTURE
```

then

```
IS THIS THE CORRECT PICTURE (Y/N);
?
```

flashes before our very eyes. We type:

```
Y       <RETURN>
```

It wants to know more:

```
WHAT SIZE? (R=REGULAR L=LARGE)?
```

Oh, regular size is fine:

```
R       <RETURN>
```

Still more options, so the next query is:

```
NORMAL(N) OR REVERSE(R)?
```

This can be a bit confusing. NORMAL prints every white dot on the screen as a black dot on the white paper. That is what we think of as **normal** printing. If we want the picture to appear with a black background, exactly as it is on the screen, we request REVERSE. Got that?

For now, let's try REVERSE:

```
R       <RETURN>
```

The last question is:

```
WHAT IS THE PRINTER SLOT # ?(1-7)
?
```

Double check to see that the printer is ON and loaded with paper. Our printer card is in slot 1, so:

```
1       <RETURN>
```

Now we just sit back and relax while the printer does its thing. No fair peeking until it finishes. All together now — PEEK!

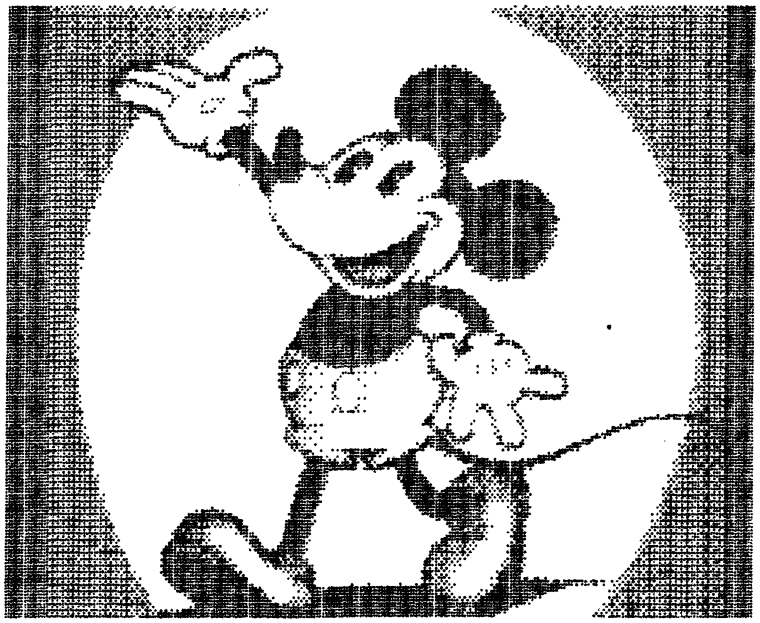


Figure 15-2

Wow, how long do you think it took the artist to create that picture? Can you imagine arranging all those little dots the way we learned in the last few chapters?

There are cameras and digitizers on the market that will convert pictures to electronic impulses and place them in the computer's memory. The point is, we can create pictures in memory by whatever means are available, then use the EPSON HIRES program to dump them to the printer.

Go ahead and try the different print options with DISNEY CHAR, large and small, reverse and regular.

Print several of the pictures; try the name of another one instead of DISNEY CHAR. Most of them load into the low memory graphics page. We'll wait for you here.

Be an Artist for Fun and Profit

Impressed, or just intimidated?

Let's see what we can do about making our own mess on the paper. Return to BASIC and type:

```
PR#1
RUN HIRES INSTRUCTIONS
```

(Don't worry if the title says EPSON TX-80 HIRES graphic routine — it's been modified for the MX series printers.) Abort the program, then enter:

```
PR#Ø
```

Tear off this list of instructions so we can refer to them as we scurry along. According to these instructions, we can write our own program to fill a memory page with hi-res graphics, then use the USR function to print it out. Let's see if they are putting us on. Type in this short program:

```
NEW

    1Ø HOME

    2Ø HGR

    3Ø HCOLOR=7

    4Ø FOR A=Ø TO 2Ø.4 STEP .Ø2

    5Ø R = 1Ø * A * COS(A) * SIN(A)

    6Ø HPLOT 138 + R * COS(A) , 79 + R * SIN(A)

    7Ø NEXT A
```

and RUN.

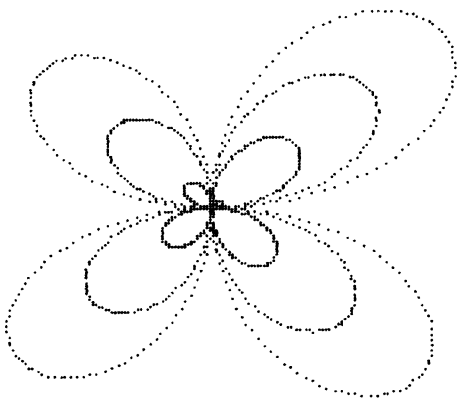


Figure 15-3

Not too shabby for rank amateurs.

Here's what happens:

Line 10 clears the screen.

Line 20 selects the low-memory graphics page. This part of memory is used for high resolution graphics, while text is stored on a different 'page' of memory. 48K users have two different chunks of memory that can be used for hi-res graphics. Refer to the Apple Programming Reference Manual for more information.

Line 30 selects a color.

Line 40 and 70 set up a loop for plotting a figure.

Line 50 calculates the radius as a function of the angle. We are using polar coordinates in case you hadn't guessed. If you don't really care, it doesn't much matter — the picture is still pretty.

Line 60 converts from polar to rectangular coordinates and plots a single point. Whew! It's a good thing computers don't require a good math background.

Why all This Math?

Using math equations to create graphics is easier than doing it the hard way, point-by-point. They do, however, restrict us to graphs or plots of natural phenomena, and who among us will claim credit for creativity in that area? (There's probably some evolutionist . . .) Type:

```
TEXT
```

to return us back to the TEXT page. Even though we can't see it, the picture remains stored in the HI-RES page of memory.

We can play with line 50 to get different figures. Try:

```
50 R=3 * A
```

and

```
50 R=30 * (1-SIN(A))
```

and

```
50 R=80 * COS(3*A/4)
```

(Change line 40 to 40 for A = 0 TO 26 STEP .01)

and RUN.

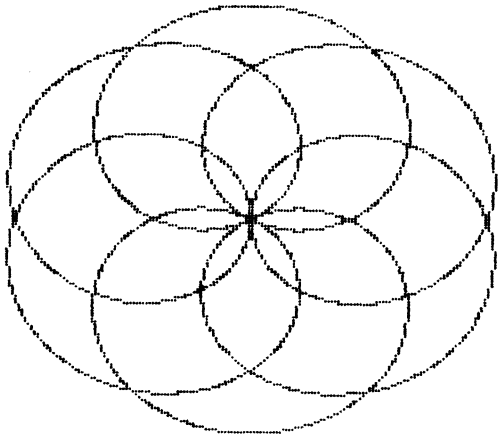


Figure 15-4

When you have a design you like, add this screen dump "linking routine" to our program:

```
80 POKE 10,76 : POKE 11,00 : POKE 12,96

90 PRINT "BLOAD EPSON.HIRES.OBJ"

100 PRINT USR(0001)
```

and RUN.

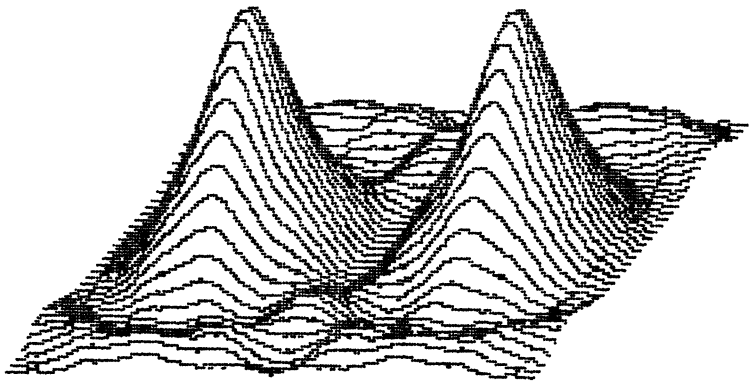


Figure 15-5

Lines 80 and 90 are straight out of the HIRES INSTRUCTIONS program listing. They load the screen dump routine without leaving the BASIC program.

As indicated in the HIRES instructions, we use the machine language routine named EPSON.HIRES.OBJ (Line 90), and call USR(ABCD) where:

```
A = PAGE (Ø=PG.1, 1=PG.2)

B = SIZE (Ø=SMALL, 1=LARGE)

C = PICTURE (Ø=REG., 1=INVERSE)

D = PRINTER SLOT (1 TO 7)
```

Therefore, our USER(0001) in line 100 calls the routine to print the picture with:

Page 1 (Low) of memory, small size, regular picture, and printer slot #1.

Change the USR statement to suit your printing desires.

Figure 15-6

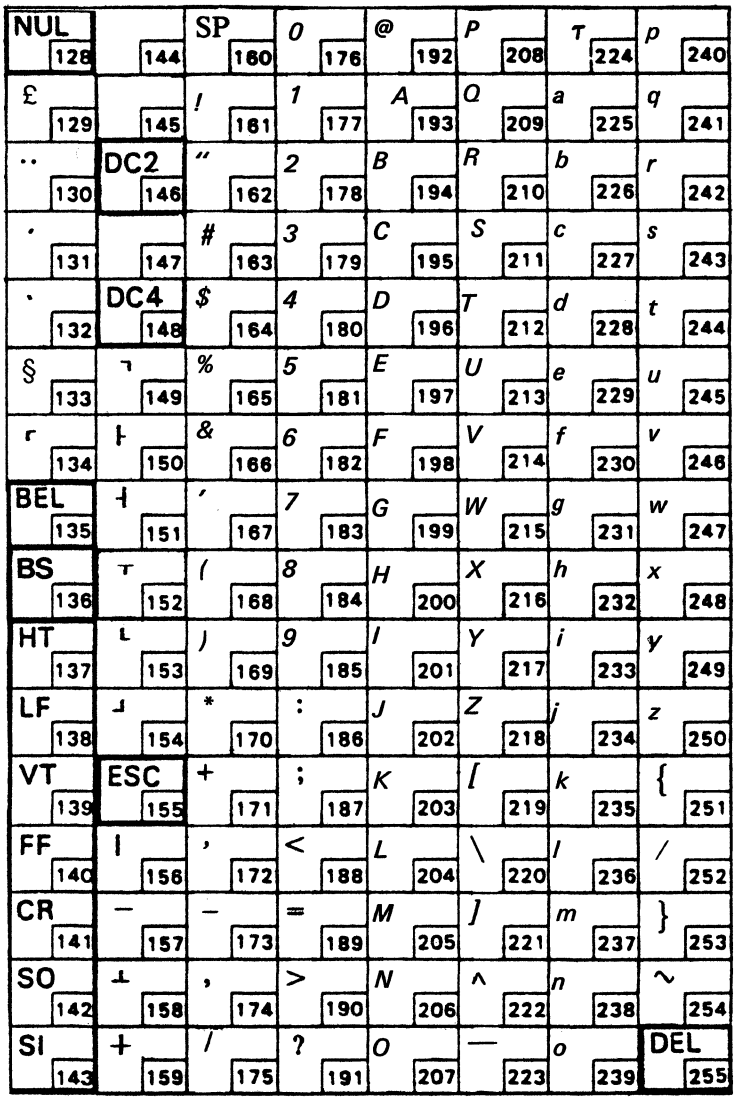
Commencement Address

Now that we've learned everything there is to know about the MX printer and HIRES graphics, our education is ready to begin. I'd enjoy hearing from you and seeing some of your artistic creations.

Appendix A

ASCII Chart

NUL 0	16	SP 32	Ø 48	@ 64	P 80	96	p 112
1	17	! 33	1 49	A 65	Q 81	a 97	q 113
2	DC2 18	" 34	2 50	B 66	R 82	b 98	r 114
3	DC3 19	# 35	3 51	C 67	S 83	c 99	s 115
4	DC4 20	$ 36	4 52	D 68	T 84	d 100	t 116
5	21	% 37	5 53	E 69	U 85	e 101	u 117
6	22	& 38	6 54	F 70	V 86	f 102	v 118
BEL 7	23	' 39	7 55	G 71	W 87	g 103	w 119
BS 8	24	(40	8 56	H 72	X 88	h 104	x 120
HT 9	25	) 41	9 57	I 73	Y 89	i 105	y 121
LF 10	26	* 42	: 58	J 74	Z 90	j 106	z 122
VT 11	ESC 27	+ 43	; 59	K 75	[91	k 107	{ 123
FF 12	28	, 44	< 60	L 76	\ 92	l 108	¦ 124
CR 13	29	- 45	= 61	M 77	] 93	m 109	} 125
SO 14	30	. 46	> 62	N 78	^ 94	n 110	~ 126
SI 15	31	/ 47	? 63	O 79	_ 95	o 111	DEL 127


NUL 128
144
SP 160
0 176
@ 192
P 208
τ 224
p 240
£ 129
145
! 161
1 177
A 193
Q 209
a 225
q 241
·· 130
DC2 146
" 162
2 178
B 194
R 210
b 226
r 242
· 131
147
163
3 179
C 195
S 211
c 227
s 243
· 132
DC4 148
$ 164
4 180
D 196
T 212
d 228
t 244
§ 133
149
% 165
5 181
E 197
U 213
e 229
u 245
r 134
150
& 166
6 182
F 198
V 214
f 230
v 246
BEL 135
151
' 167
7 183
G 199
W 215
g 231
w 247
BS 136
152
(168
8 184
H 200
X 216
h 232
x 248
HT 137
153
) 169
9 185
I 201
Y 217
i 233
y 249
LF 138
154
* 170
: 186
J 202
Z 218
j 234
z 250
VT 139
ESC 155
+ 171
; 187
K 203
[219
k 235
{ 251
FF 140
156
, 172
< 188
L 204
\ 220
l 236
/ 252
CR 141
157
- 173
= 189
M 205
] 221
m 237
} 253
SO 142
158
. 174
> 190
N 206
^ 222
n 238
~ 254
SI 143
159
/ 175
? 191
O 207
— 223
o 239
DEL 255

	DEC	0	16	32	48	64	80	96	112	128	144	160	176	192	208	224	240
	HEX	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
DEC	HEX																
0	0	NUL		SP	0	@	P	‘	p	NUL		SP	*0*	*@*	*P*	*τ*	*p*
1	1			!	1	A	Q	a	q	£		*!*	*1*	*A*	*Q*	*a*	*q*
2	2		DC2	“	2	B	R	b	r	¨	DC2	”	*2*	*B*	*R*	*b*	*r*
3	3			#	3	C	S	c	s	’		*#*	*3*	*C*	*S*	*c*	*s*
4	4		DC4	$	4	D	T	d	t	‘	DC4	*$*	*4*	*D*	*T*	*d*	*t*
5	5			%	5	E	U	e	u	§	¬	*%*	*5*	*E*	*U*	*e*	*u*
6	6			&	6	F	V	f	v	r	⊢	*&*	*6*	*F*	*V*	*f*	*v*
7	7	BEL		’	7	G	W	g	w	BEL	⊣	‘	*7*	*G*	*W*	*g*	*w*
8	8	BS		(	8	H	X	h	x	BS	⊤	*(*	*8*	*H*	*X*	*h*	*x*
9	9	HT		)	9	I	Y	i	y	HT	⌊	*)*	*9*	*I*	*Y*	*i*	*y*
10	A	LF		*	:	J	Z	j	z	LF	⌋	*	*:*	*J*	*Z*	*j*	*z*
11	B	VT	ESC	+	;	K	[	k	{	VT	ESC	+	*;*	*K*	*[*	*k*	{
12	C	FF		,	<	L	\	l	:	FF	|	,	<	*L*	\	*l*	*/*
13	D	CR		—	=	M	]	m	}	CR	—	–	=	*M*	*]*	*m*	}
14	E	SO		.	>	N	^	n	~	SO	⊥	,	>	*N*	^	*n*	~
15	F	SI		/	?	O	_	o	DEL	SI	+	*/*	*?*	*O*	_	*o*	*DEL*

Appendix B

Control Codes

	Dec	Hex	Symbol	Function	Page
%	0	00	NUL	Follows <ESC>"D" as terminator for TABS	9-4
	7	07	BEL	Sounds buzzer for 1/3 second. Paper out rings for 3 seconds	4-1
	8	08	BS	Backspace. Empties printer buffer, then backspaces print head one space	7-3,6-3
	9	09	HT	Horizontal Tabulation. Print head moves to next tab stop	9-3
	10	0A	LF	Line Feed. Printer empties its buffer and does line feed at current line spacing and Resets buffer pointer to zero	9-6
%	11	0B	VT	Vertical Tab — does single line feed	9-6
%	12	0C	FF	Advances paper to next logical TOF (top of form)	5-4
	13	0D	CR	Carriage Return. Prints buffer contents and resets buffer character count to zero	8-1
	14	0E	SO	Shift Out. Turns on double width mode to end of line unless cancelled by DC4 (20)	3-2,4-5
	15	0F	SI	Shift In. Turns on compressed character mode. Does not work with emphasized mode. Stays on until cancelled by DC2 (18)	3-3,4-3
%	18	12	DC2	Turns off compressed characters and empties buffer	3-5,4-3
%	20	14	DC4	Turns off double width mode (shift out only)	4-5
	27	1B	ESC	ASCII code for ESCAPE. Prepares printer to receive control codes	3-3
<ESC>	35	23	#	Accepts eighth bit "as is" from computer	7-3
<ESC>	45	2D	–	Underline mode. N=0 turns underline OFF. N>0 turns underline ON Format: <ESC>"-" N	3-8,6-1
<ESC>	48	30	0	Sets line spacing to 1/8"	11-6
<ESC>	49	31	1	Sets line spacing to 7/72"	11-6
<ESC>	50	32	2	Returns line spacing to default of 1/6"	11-6
<ESC>	51	33	3	Sets line spacing to N/216". Stays on until changed Format: <ESC>"3" N, 1 < = N < = 255.	11-10
<ESC>	52	34	4	Italic character set ON	3-7,4-7
<ESC>	53	35	5	Italic character set OFF	4-7
<ESC>	56	38	8	Ignores "paper out" sensor	10-3
<ESC>	57	39	9	Enables "paper out" sensor	10-3
<ESC>	60	3C	<	One line unidirectional print. Prints current line only from left to right	6-5
<ESC>	61	3D	=	Clears eighth bit. (i.e. sets to zero.)	7-1,7-3
<ESC>	62	3E	>	Sets eighth bit to 1	7-1,11-9,12-6,14-4
<ESC>	64	40	@	Resets all special modes to power up state including Top Of Form	2-6,3-6,5-8
<ESC>	65	41	A	Sets spacing of LF (line feed) to N/72" Format: <ESC>"A" N, 1 < = N < = 85.	7-6,11-7,11-9
<ESC>	67	43	C	Sets form length (FL) to N lines. Default is 66 Format: <ESC>"C" N, 1 < = N < = 127. Resets top of form.	5-8
<ESC>	67	43	C	Sets form length (FL) to N inches. Default is 11 Format: <ESC>"C" 0 N, 1 < = N < = 22. Resets top of form.	5-9
<ESC>	68	44	D	Reset current tabs and sets up to 28 HT (horiz tabs) TABs may range up to maximum width for character and printer size. E.G. Maximum TAB for normal characters on MX-80 is 80. Format: <ESC>"D" N1 N2 N3 . . . NN 0. Terminate TAB sequence with zero or 128.	9-4

	Dec	Hex	Symbol	Function	Page
<ESC>	69	45	E	Turns on emphasized mode. Can't mix with superscript, subscript, or compressed modes	4-10,6-10,10-4
<ESC>	70	46	F	Turns off emphasized mode	4-10
<ESC>	71	47	G	Turns on double strike mode.	3-5,4-9,6-10,10-4
<ESC>	72	48	H	Turns off double strike mode, superscript, and subscript modes	3-5,6-8
<ESC>	74	4A	J	Sets line spacing to N/216" for one line only and when received causes contents of buffer to print Format: <ESC>"J" N, 1 < = N < = 255.	11-10
<ESC>	75	4B	K	Sets dot graphics mode to 480 dots per 8" line (816 dots for 13.6" line) Format: <ESC>"K" N1 N2, N1 and N2 determine line length. Line length = N1 + 256*N2. 1 < = N1 < = 255. 0 < = N2 < = 255 (Modulo 8, i.e. 8 = 0)	3-6,12-1
<ESC>	76	4C	L	Sets dot graphics mode to 960 dots per 8" line (1632 dots 13.6" line) Format: <ESC>"L" N1 N2, N1 and N2 determine line length. Line length = N1 + 256*N2. 1 < = N1 < =255. 0 < = N2 < =255 (Modulo 8, i.e. 8 = 0)	13-10
<ESC>	78	4E	N	Sets skip over perforation to N lines Format: <ESC>"N" N, 1 < = N < = 127.	8-1
<ESC>	79	4F	O	Resets skip over perforation to 0 lines	8-2
<ESC>	81	5B	Q	Sets column width Format: <ESC>"Q" N, 1 < = N < = maximum number of characters/line.	9-5
<ESC>	83	5D	S	Sets superscript/subscript modes Format: <ESC>"S" N, N=0 = > superscript, N>0 = > subscript.	3-3,6-6
<ESC>	84	5E	T	Resets superscript, subscript, and unidirectional printing (does not turn off double strike from script modes)	3-3,6-5,6-6
<ESC>	85	5F	U	Unidirectional printing. Prints each line from left to right Format: <ESC>"U" N, N=0 = > OFF, N>0 = > ON.	6-5,7-6
<ESC>	87	61	W	Double width printing. Stays ON until turned OFF Format: <ESC>"W" N, N=0 = > OFF, N=1 = > ON. Has precedence over Shift Out (SO = CHR$(14)).	4-6,6-11
	127	7F	DEL	Deletes last character in printer buffer.	
	128	80	NUL	Follows <ESC>"D" as terminator for TABS	9-4
	135	87	BEL	Sounds buzzer for 1/3 second. Paper out rings for 3 seconds.	
	136	88	BS	Backspace. Empties printer buffer, then backspaces print head one space.	
	137	89	HT	Horizontal Tabulation.	
	138	8A	LF	Line Feed.	
	140	8B	VT	Vertical Tab — does single line feed.	
	141	8C	FF	Advances paper to TOF (Top Of next Form).	
	142	8D	CR	Carriage Return.	
	143	8E	SO	Shift Out. Turns on double width. Turns OFF at end of line.	
	144	8F	SI	Shift In. Turns on compressed character mode. Does not work with emphasized mode.	
	146	92	DC2	Turns off compressed characters.	
	148	94	DC4	Turns off double width mode. (Shift out only)	
	155	9B	ESC	ASCII code for ESCAPE.	
	255	FF	DEL	Deletes last character in printer buffer.	

NOTE: Numbers flagged with a % may require the addition of 128 to make them work reliably. When in doubt, add 128.

Appendix C

Control Key Table

What to Type	Dec	Hex	Char
ctrl @	0	00	NUL
ctrl A	1	01	SOH
ctrl B	2	02	STX
ctrl C	3	03	ETX
ctrl D	4	01	EOT
ctrl E	5	05	ENQ
ctrl F	6	06	ACK
ctrl G	7	07	BEL
ctrl H or ←	8	08	BS
ctrl I	9	09	HT
ctrl J	10	0A	LF
ctrl K	11	0B	VT
ctrl L	12	0C	FF
ctrl M or RETURN	13	0D	CR
ctrl N	14	0E	SO
ctrl O	15	0F	SI
ctrl P	16	10	DLE
ctrl Q	17	11	DC1
ctrl R	18	12	DC2
ctrl S	19	13	DC3
ctrl T	20	14	DC4
ctrl U or →	21	15	NAK
ctrl V	22	16	SYN
ctrl W	23	17	ETB
ctrl X	24	18	CAN
ctrl Y	25	19	EM
ctrl Z	26	1A	SUB
ESC	27	1B	ESC
n/a	28	1C	FS
ctrl shift-M	29	1D	GS
ctrl	30	1E	RS
n/a	31	1F	US

NOTE: Control characters do not print, they **control** the printer. This is the standard on **most** keyboards that have the control key.

Appendix D

Sample Print Modes

NORMAL TYPE ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghij
klmnopqrstuvwxyz!"#$%&'()*+,-./0123456789:;<=>?@
[\]^_{|}~£¨`§

DOUBLE WIDTH TYPE ABCDEF

COMPRESSED TYPE ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz!"#$%&'()*+,-.

DOUBLE WIDTH COMPRESSED ABCDEFGHIJKLMNOPQ

DOUBLE STRIKE TYPE ABCDEFGHIJKLMNOPQRSTUVWXYZ!#$

EMPHASIZED TYPE ABCDEFGHIJKLMNOPQRSTUVWXYZ!#$%&'

DOUBLE STRIKE EMPHASIZED ABCDEFGHIJKLMNOPQRSTUVW

SUPERSCRIPT CHARACTERS/SUBSCRIPT CHARACTERS

SUPERSCRIPT COMPRESSED/SUBSCRIPT COMPRESSED

UNDERLINE CHARACTERS ABCDEFGHIJKLMNOPQRSTUVWXYZ!

ITALICS

NORMAL TYPE ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghij
klmnopqrstuvwxyz!"#$%&'()+,-./0123456789:;<=>?@*
[\]^_{|}~£¨`§

DOUBLE WIDTH TYPE ABCDEF

COMPRESSED TYPE ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz!"#$%&'()+,-.*

DOUBLE WIDTH COMPRESSED ABCDEFGHIJKLMNOPQ

DOUBLE STRIKE TYPE ABCDEFGHIJKLMNOPQRSTUVWXYZ!#$

EMPHASIZED TYPE ABCDEFGHIJKLMNOPQRSTUVWXYZ!#$%&'

DOUBLE STRIKE EMPHASIZED ABCDEFGHIJKLMNOPQRSTUVW

SUPERSCRIPT CHARACTERS/SUBSCRIPT CHARACTERS

SUPERSCRIPT COMPRESSED/SUBSCRIPT COMPRESSED

UNDERLINE CHARACTERS ABCDEFGHIJKLMNOPQRSTUVWXYZ!

HI-RES GRAPHICS

GRAPHICS CHARACTER SET
┌┐├┼┬┴┘│─┼┼

Appendix E

Internal Switches

Never set the 12 internal DIP (dual in-line pin) switches when the power is ON. Turn **both** the printer and computer OFF.

SW2	SW1
	1-8 L
	1-7 R
	1-6 L
	1-5 R
2-4 R	1-4 R
2-3 R	1-3 R
2-2 R	1-2 R
2-1 R	1-1 R

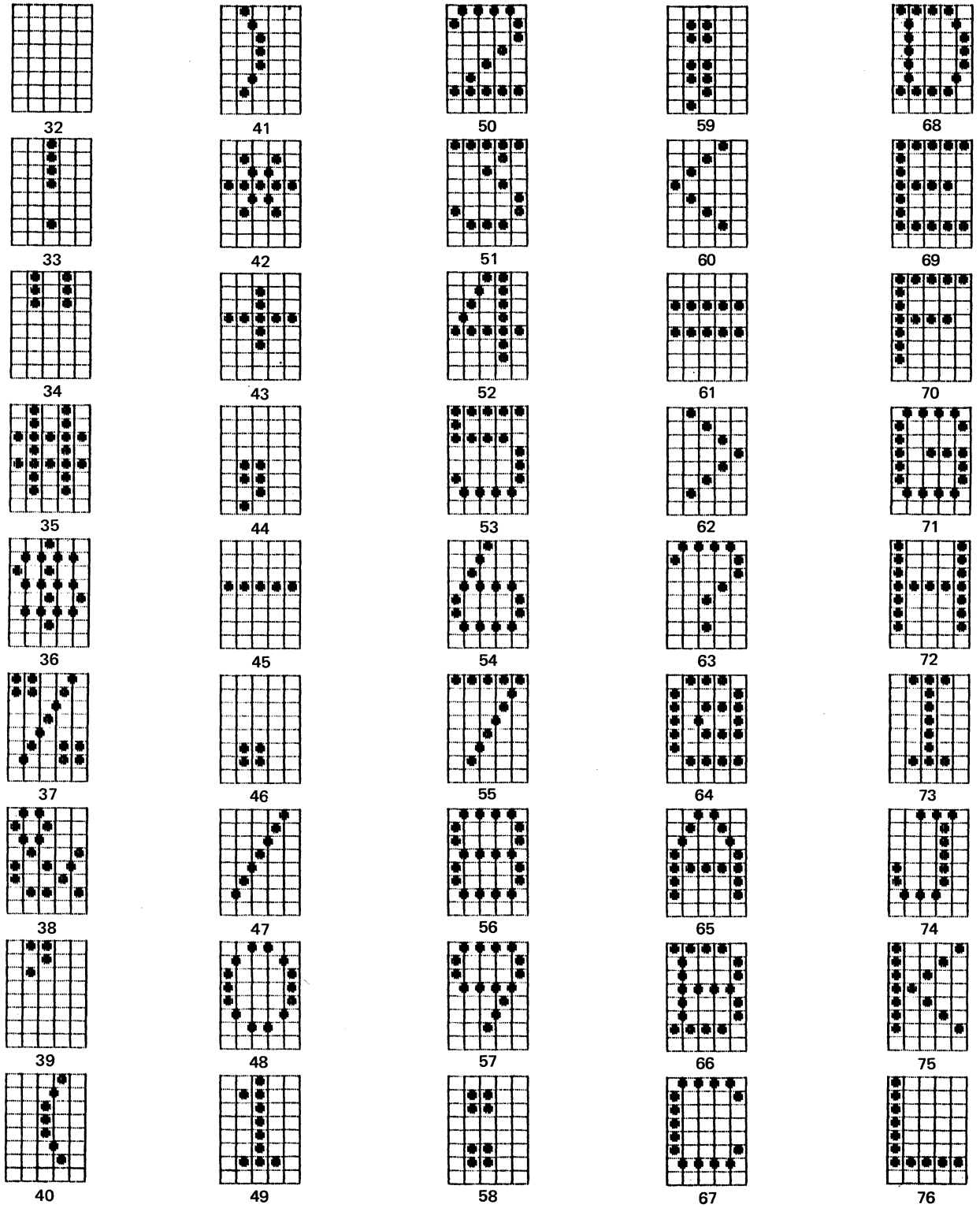
Switch summary:

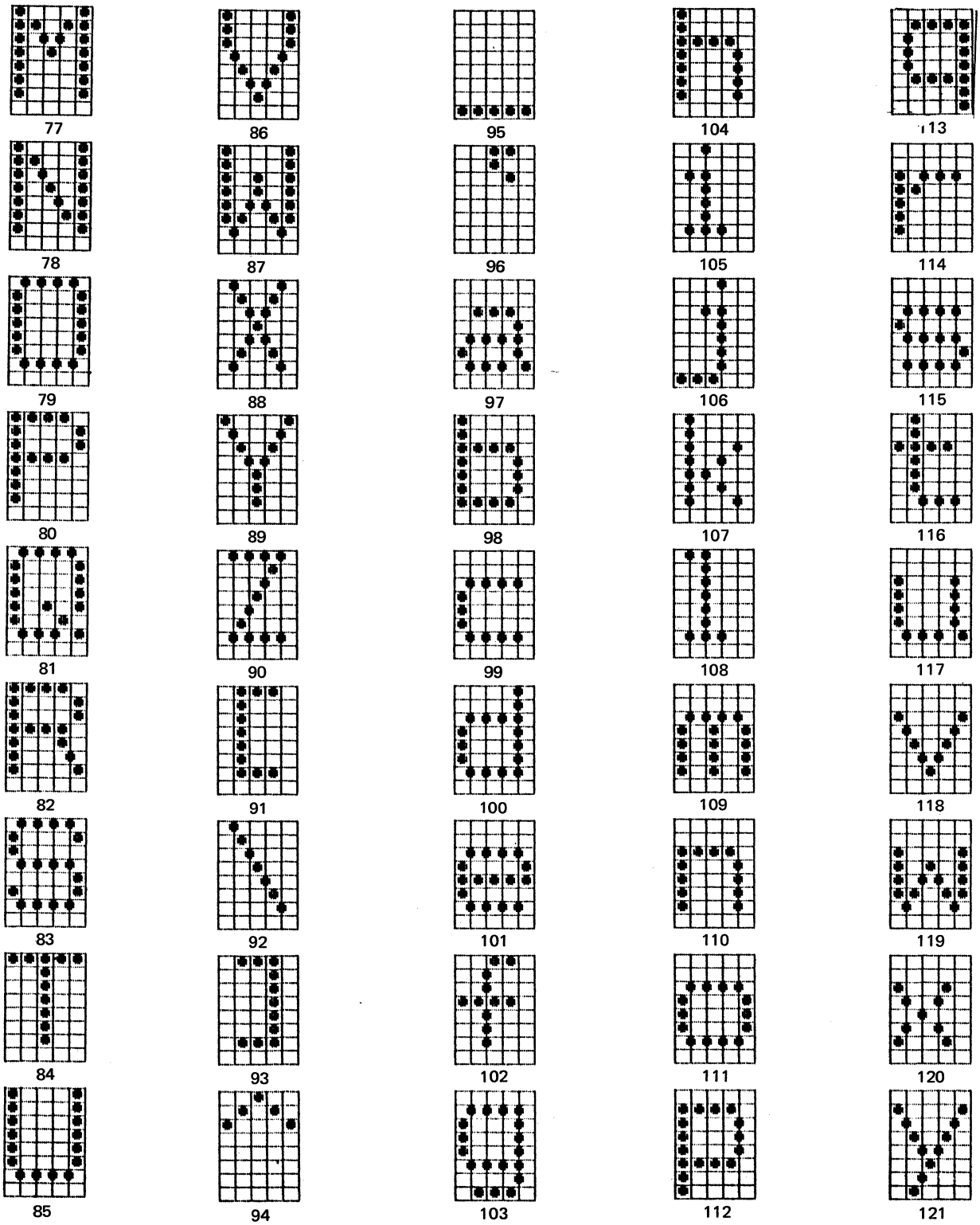
	ON	OFF
SW1-8	Select fixed	Select not fixed
SW1-7	Slashed zero	Regular zero
SW1-6	Buzzer on	Buzzer off
SW1-5	Emphasized	Normal
SW1-4	Italic	Normal
SW1-3	Paper out sensor off	Paper out sensor on
SW1-2	Not used	Not used
SW1-1	Compressed	Normal
SW2-4	1 inch skip over perf on	Normal
SW2-3	Auto LF with CR	LF must be from host
SW2-2	Not used	Not used
SW2-1	Not used	Not used

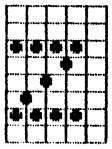
Switch Number	Normal Position	Function
1-8	ON	When the switch is ON, printer is permanently in the "selected" mode and no external command can "deselect" it. In the OFF position, it theoretically could be selected and deselected by external software codes. Since the printer does not recognize these codes, we leave it in the ON position. This refers to pin 36 on the connector. Some computers control this pin; if they do control the pin, SW should be OFF.
1-7	OFF	Allows the option of regular or slashed zero. If ON, slashed zero is printed. If OFF, regular zero is printed.
1-6	ON	If ON, buzzer will sound at ASCII 7 or paper-out signal. If OFF, buzzer will not sound at all.
1-5	OFF	Selects the default printing mode. OFF gives normal print characters. ON gives emphasized characters. Emphasized has priority over compressed mode (SW 1-1).
1-4	OFF	Determines the print font used. OFF gives normal characters. ON gives italic characters.
1-3	OFF	Affects paper out detection. With switch OFF, paper-out signal terminates printing. With switch ON, printing continues without paper (this will not work if computer monitors pin 12. PE signal will go high if SW 1-3 is ON).
1-2	OFF	Not used.
1-1	OFF	Determines default print width. OFF gives normal (10 CPI) character width. ON gives compressed (17.16 CPI) characters. Emphasized mode (SW 1-5) has priority over compressed mode.
2-4	OFF	When ON, gives automatic 1 inch skip over perforation. Great for readable program listings. When OFF, gives no skip over perforation.
2-3	OFF	Forces automatic LF with each CR. When OFF, LF must be provided via software as needed.
2-2	OFF	Not used.
2-1	OFF	Not used.

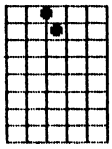
Appendix F

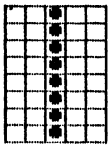
Character Fonts

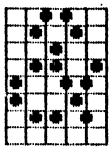


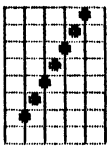

77
86
95
104
113
78
87
96
105
114
79
88
97
106
115
80
89
98
107
116
81
90
99
108
117
82
91
100
109
118
83
92
101
110
119
84
93
102
111
120
85
94
103
112
121

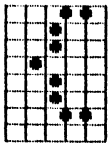

122

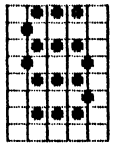

132

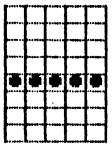

156

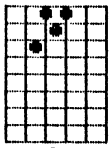

166

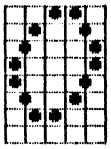

175

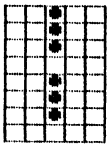

123

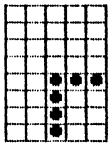

133

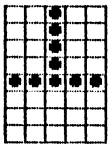

157

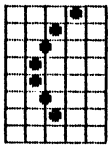

167

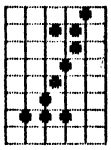

176

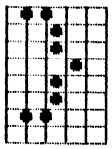

124

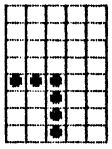

134

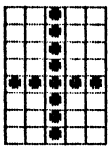

158

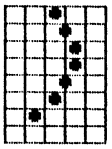

168

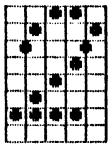

177

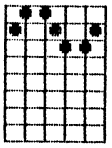

125

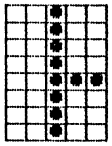

149

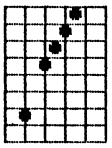

159

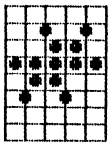

169

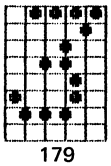

178

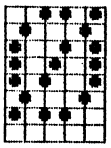

126

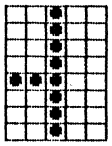

150

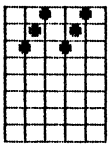

161

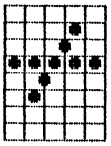

170

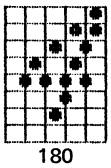

179

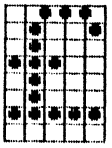

ALTERNATE CODE 48

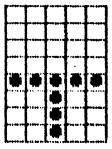

151

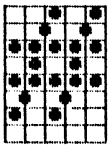

162

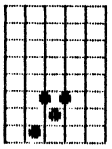

171

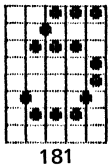

180

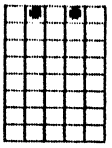

129

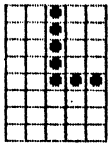

152

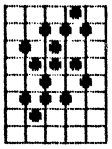

163

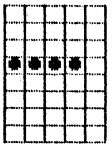

172

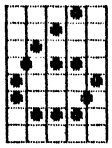

181


130


153


164


173


182

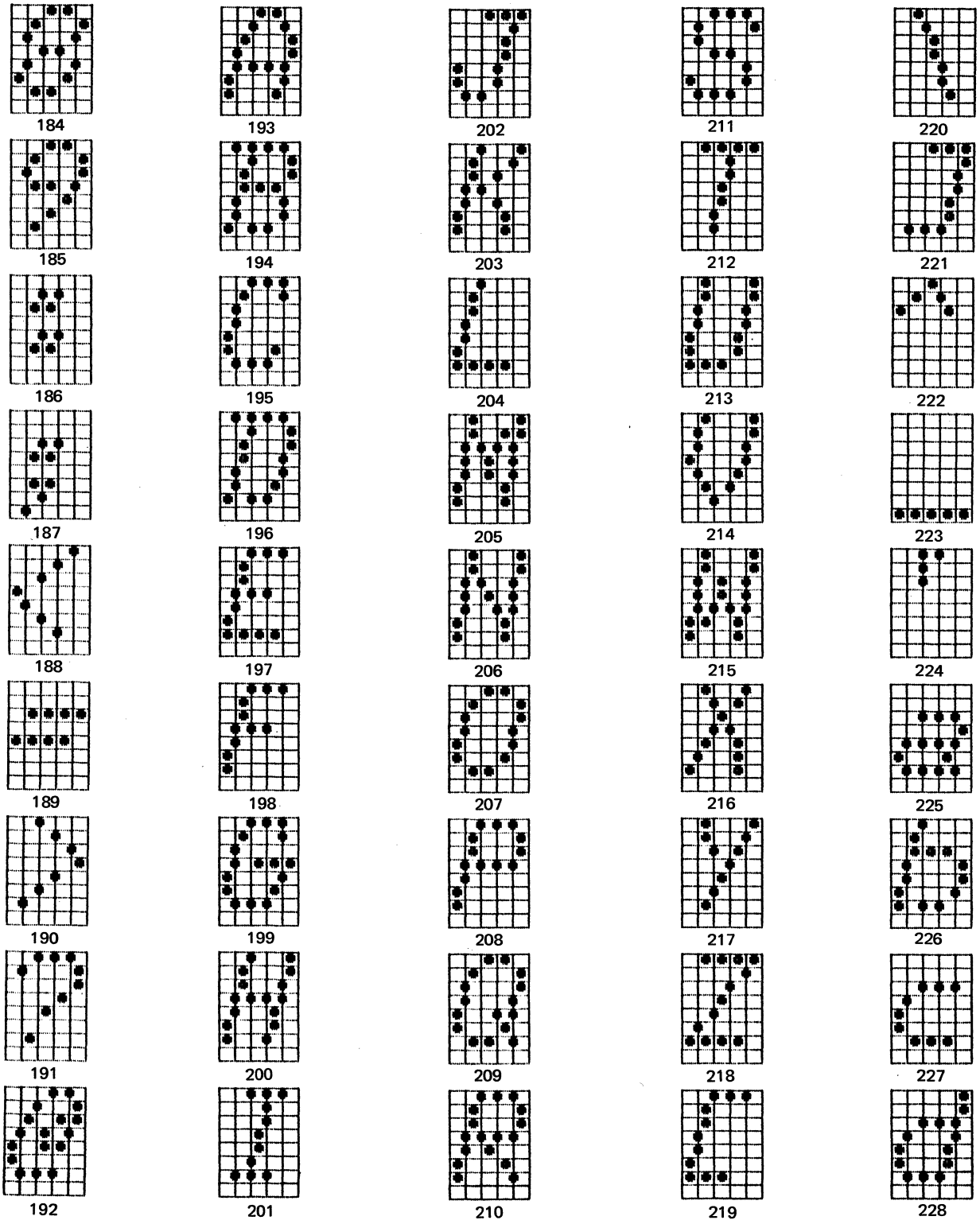
131

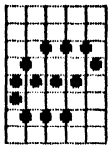
154

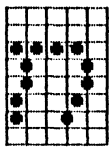
165

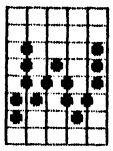
174

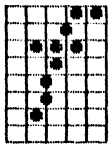
183

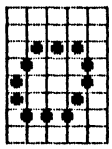

184
193
202
211
220
185
194
203
212
221
186
195
204
213
222
187
196
205
214
223
188
197
206
215
224
189
198
207
216
225
190
199
208
217
226
191
200
209
218
227
192
201
210
219
228

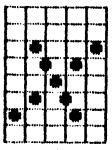

229

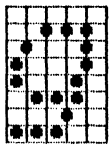

238

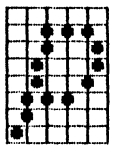

247

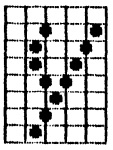

230

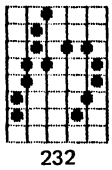

239

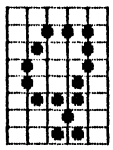

248

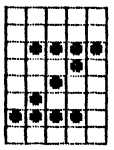

231

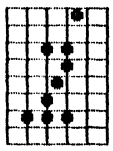

240

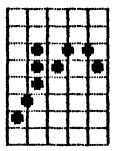

249

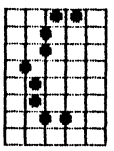

232

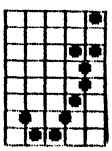

241

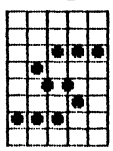

250

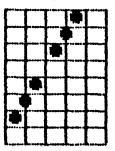

233

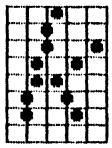

242

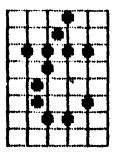

251

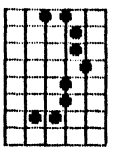

234

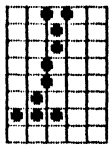

243

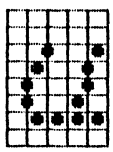

252

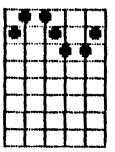

235

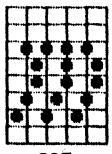

244

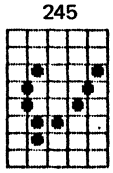

253

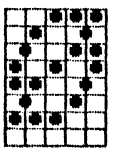

236


245


254


237


246


ALTERNATE CODE 176

Appendix G

Use with TRS-80 Computers

TRS-80 computers do not send a line feed at the end of each line. The printer must provide the line feed. So TRS-80 users should set Switch 2-3 ON. Otherwise, no line feed will occur.

Model I Without Expansion Interface

EPSON provides a TRS-80 interface kit to allow Model I users to have the best in printers without going to the expense of an expansion interface. It consists of a PC board that fits in the printer, and a short connecting cable. See your dealer.

A marginal alternative to the EPSON TRS-80 interface kit with its short cable is the Radio Shack printer interface, part number 26-1411. It requires a separate 5 volt power supply and some "homebrew" wiring; thus it is not recommended to other than true-hardware types.

Model I With Expansion Interface

The Model I has a nasty quirk that poses problems when sending a CHR$(0) to the printer. The printer driver software in the Model I interprets it as a NUL code and does not pass it on to the printer. That hurts since zero is used in several circumstances. There are basically two ways around the problem.

One is to take advantage of any alternate codes that the printer may interpret as a zero. In the WELCOME program, for example, changing the CHR$(0) in line 50 to CHR$(8) will solve the problem since N2 in the ESCAPE K N1 N2 sequence happens to work modulo 8. (The printer interprets codes 8, 16, 24, . . . as a zero.) Okay, so we got lucky on that one.

The other approach is to tackle the problem head on using the POKE statement. Line 30 in the WELCOME program should be changed to:

```
30 LPRINT TAB(22); CHR$(15) "BY" CHR$(14) " EPSON" CHR$(27)

CHR$(83); : POKE 14312,0
```

The POKE statement sends the 0 directly to the printer buffer without getting intercepted by BASIC. Unfortunately, this nifty solution causes a problem of its own. Using a POKE upsets the printer timing in such a way that we risk losing data. The way to prevent any data loss is to add:

```
35 IF PEEK(14312)<>63 THEN 35
```

Line 35 checks to make sure the printer is ready to accept more codes. If not, the program stays in a delay loop until the printer is ready. All POKEs should be followed by a similar PEEK.

The total changes required to make the welcome program function as intended on the Model I are:

```
30 LPRINT TAB(22); CHR$(15) "BY" CHR$(14) " EPSON"
   CHR$(27) CHR$(83); : POKE 14312,0
35 IF PEEK(14312)<>63 THEN 35
50 LPRINT TAB(18) CHR$(27) "K" CHR$(80) CHR$(8);
```

Zero is also used as a terminator for the <ESC>"D" sequence in Chapter 9. We could also use 128, but poking a zero seems to be the most effective way on some Model I's. We will also have to POKE codes 10, 11, and 12 when they cause trouble, especially in the graphics mode.

In Chapter 9, we run into a problem with horizontal TABs. After 63 columns, the computer sends a carriage return (on early Model I's). Using the STRING$ statement, we can get around it. In the first H-TAB program change line 30 to:

```
30 LPRINT STRING$(N-PEEK(16539),32)N
```

The address 16539 is the current position in the printer buffer. By subtracting that number from the column number we want, (N), we insert the additional number of blank spaces (ASCII code 32). Tricky, but effective.

Be sure that when you use <ESC>"@" to reset printer codes, you don't accidently hit shift-@. The printer will ignore it, and the result can cause havoc when trying to clean up a program.

Model II

The MX printers work fine with Radio Shack cable #26-4401. Be sure the connectors on both ends make good contact.

The Model II is in many ways similar to the other models. Its disk BASIC interpreter uses LPRINT in the same way, and the control functions respond generally the same, except as follows:

Initialization

The Model II may need to be initialized with its own FORMS command in order to properly feed a printer. Initialization can be performed either at the DOS or BASIC levels.

From DOS, type:

```
FORMS
```

The computer will respond with:

```
PRINTER READY  (Y/N)
```

Assuming the printer is properly connected and turned ON, type:

```
Y <en>
```

and

```
Q <en>
```

Initialization is complete and we are returned back to DOS.

From BASIC, type:

```
SYSTEM "FORMS" <en>
```

After finishing the questions and typing:

```
Q <en>
```

We are automatically returned to BASIC.

Glitches

Under certain circumstances <ESC>"G", the double-strike mode may cause some timing problems. The PAUSE command should clear this up. Example:

```
10 SYSTEM"PAUSE"
```

causes the program to wait until the printer is available, and may save the day.

TRS-80 Model III

No interface kit is necessary for the Model III. Just plug the EPSON cable into the printer port, and fire away.

The Model III **does not** share the Model I's problem with sending code 0 to the printer. Nevertheless, codes 10, 11, and 12 can still mean trouble. An alternative way to send codes to the printer is:

```
OUT 251,#
```

where # represents the code we wish to send. If this creates timing problems, you may need to test the status of the printer before you send it each code number. For example:

```
100 IF INP(251)<>61 THEN 100
```

places the program in a "holding pattern" until the printer is ready for more. See the computer reference manual for more details.

Using the Standard Radio Shack Cable

There is a difference between the 'official' EPSON printer cable and the ones supplied by Radio Shack. They are wired slightly differently.

The EPSON cable allows separation of the CR (carriage return) and the LF (line feed) commands. This, in turn, allows such things as underlining, overstrikes to slash zeros, sevens, the letter Z, to "black out" material, etc. Fortunately, the printer provides other ways for us to do most of these things.

If you do not need to separate CR and LF and already have a Radio Shack printer cable, it should work fine. If purchasing a new cable, buy an EPSON cable to keep all the options open.

EPSON's Original Model I/III cable (#8220) does not allow disabling of the paper-out sensor on F/T printers. Replace with part #8222 or disconnect and ground wire 12 of the 8220 cable, if you are handy with hardware.

Appendix H

Use With the Atari

Using Atari 800/400 Computer with Atari 850 Interface

Lines ending with a semicolon cause the Atari to automatically "pad" the rest of the line with spaces until it is 40 characters long, when using LPRINT. Therefore, it is advisable to avoid LPRINT. The good news is there IS another approach to printing that allows semicolons to be used without too many problems.

```
10 OPEN #7,8,0,"P"
```

The #7 opens file #7; 8 signifies output device; 0 is not used; and P assigns the device as printer.

```
20 PRINT #7;"ABC";

30 PRINT #7;"DEF"

40 CLOSE #7
```

prints

```
ABCDEF
```

on printer.

Port #7 is the printer port usually used by the Atari.

The bad news is that when using PRINT #7, a semicolon on the final print line of a program will also "pad" the line to 40 characters, similar to LPRINT. Therefore the buffer is not emptied until an extra PRINT #7 is received or the buffer is filled.

To use the ASCII control codes we often need to add 128. Atari uses codes 0-31 for its graphics characters.

If inverse characters are sent as data in the dot graphics mode, the eighth bit will be set, causing the top pin to fire for every character. Highly undesirable!

Since the Atari can not directly read data elements into an array, they first have to be assigned to a variable and then the variable assigned to the array. For example, in the shady characters program in Chapter 13, change line 20 from:

```
20 FOR I=1 TO 16 : READ A(I) : NEXT I
```

to

```
20 FOR I=1 TO 16 : READ M : A(I) = M : NEXT I
```

The Atari 850 interface has a 4 second time-out. If it does not receive an acknowledge signal from the printer within 4 seconds of sending it a command, the computer will send a "Timeout", Error Code #138 message. The program can catch this error using the BASIC TRAP command. Consult your Atari manual for use of TRAP.

Semicolons or commas must be used to separate all literal and variable strings.

The maximum length of a string is 99 characters, and all string variables must be DIMensioned.

The Welcome Program

Because the Atari computer does not have the TAB feature and uses a different method for accessing the printer, you should skip the first half of Chapter 9, up to Internal Printer TABs. We also use the TAB statement in several programs in the manual to space RUNs over from the left margain. You can ignore those TAB statements. The welcome program in Chapter 2 also will not work as written. Below is a listing of the program modified to work on the Atari.

```
5 OPEN #7,8,0,"P"
10 PRINT #7; "GREETINGS FROM THE VERSITILE"
15 REM         1234567890
20 PRINT #7; "          "; CHR$(14); "MX-80"
30 PRINT #7; "          "; CHR$(15); "BY"; CHR$(14); " EPSON";
CHR$(20); CHR$(27); CHR$(83); CHR$(0);
40 PRINT #7; "(TM)"; CHR$(27); CHR$(84); CHR$(18)
45 REM         12345678
50 PRINT #7; "        "; CHR$(27); CHR$(75); CHR$(80); CHR$(8);
60 FOR N=1 TO 80 : PRINT #7; CHR$(ABS(N-40)+20); : NEXT N
70 PRINT #7 : PRINT #7; CHR$(27); CHR$(52);
80 PRINT #7; "       "; CHR$(27); CHR$(45); CHR$(1); "YOU'LL LIKE
ME!"
90 PRINT #7; CHR$(27); CHR$(64)
100 CLOSE #7
```

Listings

Listing a program in the computer's memory to the printer is accomplished with LIST "P" or L. "P".

Pin Diagram

Figure H-1 shows Atari cable end that goes into the 850 interface. The edge connector on the other end, not shown, is for a Centronics 737-1. This end will have to be taken off and replaced with an Amphenol 57-30360 connector and wired as shown in Table H-1.

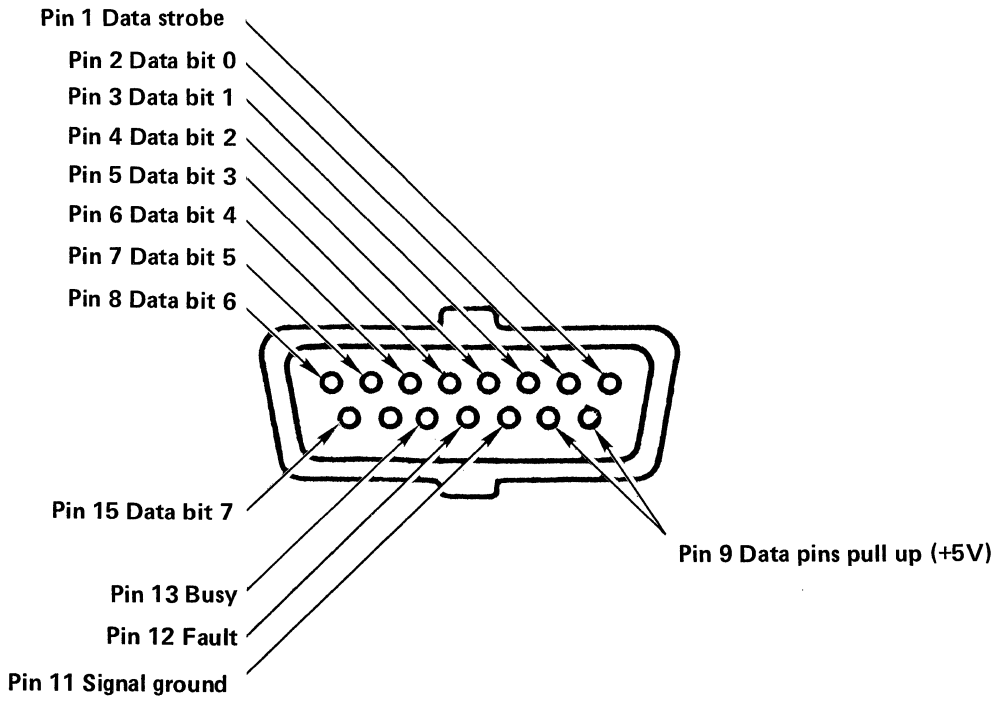


Table H-1

Atari		EPSON	
PIN 1	DS	PIN 1	DS
PIN 2	BIT 0	PIN 2	BIT 0
PIN 3	BIT 1	PIN 3	BIT 1
PIN 4	BIT 2	PIN 4	BIT 2
PIN 5	BIT 3	PIN 5	BIT 3
PIN 6	BIT 4	PIN 6	BIT 4
PIN 7	BIT 5	PIN 7	BIT 5
PIN 8	BIT 6	PIN 8	BIT 6
PIN 9	BIT NC		
PIN 10	BIT NC		
PIN 11	SIG. GROUND	PIN 16	OV
PIN 12	FAULT	PIN 32	ERROR
PIN 13	BUSY	PIN 11	BUSY
PIN 14	NC		
PIN 15	BIT 7	PIN 9	BIT 7

Appendix I

Use With the IEEE Interface

EPSON makes an I/F that conforms to IEEE 488 standard as a listen-only device. Its uses:

1. Peripheral printer for mini/micro computer systems.
2. Hard copy output device for intelligent data terminals.
3. Forms generation for scientific and data communication systems.

Computers, such as HP-85 with IEEE output ports, work fine with the EPSON 8161 IEEE 488 I/F.

Commodore Pet is another story. Commodore Pet uses "Pet ASCII", not standard ASCII. To print characters on a printer as they appear on the screen, you must use a character-conversion program, which comes on a diskette sold by Pet. If you are not sure how to use the program, refer to the "Pet CBM Personal User's Guide." EPSON dealers can supply a special EPSON IEEE cable for use specifically with Pet computers.

There are two switches on the IEEE I/F board. One 6 pin dip switch is for the printer device address. Factory set (pin 2 and 3 on) at #6. Most computers assign printers as device #4.

There is one 4 pin dip switch. Three of the four switches have to do with select IN function and have no effect. One of four marked JPet1 will ground (on) switch 2 pin 4.

Even though the switches have names like JPET1, JPET2, JPET3 they have nothing to do with Commodore.

IEEE 488 Pin Configuration

Pin No.	Signal Name	Description
1	DIO1	Data Bit 1
2	DIO2	Data Bit 2
3	DIO3	Data Bit 3
4	DIO4	Data Bit 4
5	EOI	End Or Identify
6	DAV	Data Valid
7	NRFD	Not Ready For Data
8	NDAC	Not Data Accepted
9	IFC	Interface Clear
10	SRQ	Service Request
11	ATN	Attention
12	FG	Frame Ground
13	DIO5	Data Bit 5
14	DIO6	Data Bit 6
15	DIO7	Data Bit 7
16	DIO8	Data Bit 8
17	REN	Remote Enable
18	GND	Ground
19	GND	Ground
20	GND	Ground
21	GND	Ground
22	GND	Ground
23	GND	Ground
24	GND	Ground

Appendix J

Use With the Apple

Install the interfacing card in the Apple according to the directions that accompany it. All examples in this manual assume that the card is in slot 1.

Since the Apple interface card sends a line feed at the end of every line, we don't need to force one. Accordingly, switch 2-3 in the printer should be OFF. Double spacing will result otherwise.

Apple computers do not separate the PRINT commands to the screen and to an external port, such as a printer. The use of either PRINT or LIST will result in output to only the screen, unless otherwise instructed.

Apple Integer BASIC

The MX printers are activated by ASCII code numbers from 0 - 255. The easiest way to send these codes in BASIC is via the CHR$ function, a feature not supported by Apple integer BASIC. In theory, the codes could all be poked from BASIC, but it doesn't seem to be a practical option.

Applesoft

Apple disk users having trouble with the programs may wish to heed the advice of the Apple DOS Manual. Apple recommends that all DOS commands executed from a BASIC program (i.e., with a line number) be preceded by CTRL-D (CHR$(4)). For example:

```
10 PRINT CHR$(4)"CATALOG"
```

gives a catalog of all files on the disk.

The DOS commands most frequently needed are PR#1 and PR#0. They can usually be issued from either inside a program (with a line number) or outside the program at the "command level." In most cases, a simple PR#1 or PR#0 will do fine. In Chapter 4, while we're learning printer control, it is essential that they be included inside the program. In fact, you'll find that your programs work better if you use PR#1 and PR#0 consistently within your programs. For example, if PR#1 is used at command level, PR#0 will have no effect if used within a program.

There are several ways to issue a CTRL-D from a BASIC program. One is to use CHR$(4) as in the following example

```
9 PRINT CHR$(4)"PR#1"

10 PRINT "VIDEO AND PRINTER"

20 PRINT CHR$(4)"PR#0"

30 PRINT "VIDEO ONLY"
```

Many disk users find the CHR$(4), called "Control D", in lines 9 and 20 unnecessary (unless the program accesses the disk). They simply use:

```
9 PR#1

20 PR#0
```

instead. Experiment to see what works best with your system.

The Welcome Program

The welcome program in Chapter 2 should work just fine on the Apple if you change all LRPINTs to PRINT. Type in the program, but DO NOT RUN it yet.

We need to activate the printer port. Assuming you placed the interface card in the Apple slot #1, type:

```
PR#1 <cr>
```

The computer output will now be sent to the printer port, and to the screen.

To see if everything is hooked up and properly initialized, type:

```
LIST <cr>
```

and the program should be LISTed on the screen and printer.

Now RUN to see the welcome program.

To return printing back to the screen only, enter:

```
PR#0
```

Then:

```
LIST
```

The program should LIST on the screen only.

Breaking a Program

When a program enters an endless loop or gets stuck in graphics mode awaiting more graphic codes, either the RESET key or <CTRL-C> will break the program. The latter stops the program or listing and leaves the printer on line. The RESET key not only stops the program, but returns the information

flow to the screen only, just like a PR#0 (unless, of course, the RESET key is totally deactivated by the switch under the Apple keyboard. In that case use <CTRL> <RESET>). Note that if you load Applesoft from diskette or cassette, <RESET> will kick the computer into a monitor level. For most of our examples, <CTRL-C> will be your best choice.

Screen Width Versus Printer Width

The Apple video screen display is limited to 40 characters per row. Unfortunately, it restricts the printer line width to the same 40 characters during BASIC program listings unless we take evasive action. Fortunately, the solution is simple.

If we send this code sequence down the line (don't do it yet — wait for below)

```
<CTRL I> 8Ø N  <cr>
```

the printer line width becomes the full 80 columns used by standard printer paper. The "80" can be any number from 40 to 255 as far as the computer is concerned. The printer might have different ideas.

The prior code sequence also diverts all output to the printer **only**! The video display is completely disabled after the CTRL-I # N sequence. The way to bring it back is by:

```
<CTRL I> I
```

In the process, the printer width defaults back to 40 columns.

Apple Mush

But how do we send control characters to the printer? The most obvious way is with the CTRL key on the keyboard. Let's try it.

1. Activate the printer by typing PR#1 <return>
2. Hold down the CTRL key and press I.
3. Type 80N <return>

Wonderful! Not only does nothing print on the screen, but the printer says, ?SYNTAX ERROR.

Not to worry. Try pressing a few keys on the keyboard followed by <return>. The video is disconnected, but the printer works fine. If there is a program in memory, it will LIST fine up to 80 columns wide. To regain control of the video, use the CTRL key again:

```
<CTRL I>

I <return>
```

A cleaner way to do the same thing is with CHR$(9) in place of <CTRL I>. Try:

```
PRINT CHR$(9)"80N"
```

This kicks us into the printer only mode without an annoying ?SYNTAX ERROR. To return, use:

```
PRINT CHR$(9)"I"
```

Type carefully as the latter doesn't show up on the screen.

Unfortunately, CHR$(9) is also used by the printer to activate horizontal TABs. We can eliminate any conflict by changing the printer initialization code to CHR$(1) with:

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

See Chapter 9. Once done, we can use CHR$(1) in place of CHR$(9) in the above example, and use CHR$(9) for horizontal TABulation.

Seven Bit Limitation

The Apple interface card does not pass ASCII codes greater than 127. The printer receives codes 128 - 255 as duplicates of 0 - 127. This limitation can be overcome by <ESC>" >" when accessing italics, foreign symbols, and line graphics (see ASCII chart in Appendix A), but it is not recommended during escape sequences unless you fully understand how it works.

Special Apple Graphics Trick for Long Lines

When printing long lines of graphics, the Apple can be quite finicky. To smooth out the problem, add this line at the beginning of a graphics program, after the PR#1:

```
10 PRINT CHR$(9)"255H"
```

One drawback in using this method. For some reason, it disables the line feed. You'll have to control the linefeed "manually" in the rest of the program by adding CHR$(10) as needed. It's sort of like the kid down the block who does a super job mowing your lawn, but leaves big piles of cuttings for you to pick up. The solution to one problem creates a different problem.

Tricky Codes

Most Apple users will encounter difficulty when using codes 9 and 13 in ESCAPE sequences and graphics mode. To corral these and other rowdy codes, we can poke the ASCII code numbers directly to the printer driver. Use:

```
POKE 49296,9
```

in place of

```
PRINT CHR$(9)
```

Unfortunately, using the POKE statement instead of PRINT creates timing problems between the printer and computer. To avoid such complications, the POKE should always be followed by a test of the printer to see if it is ready for more data. On the Apple, this is done with a PEEK statement. Example:

```
10 PR#1
20 POKE 49296,9
30 IF PEEK(49601)>127 GOTO 30
40 PR#0
```

If location 49601 (C1C1 Hex) is negative, the printer has not yet picked up the nine and stored it in its buffer. If location 49601 is positive, the printer is ready to receive more data. By using this test we outflank the funny code numbers like 9 and 13.

Visicalc™

Specialized utility programs are often designed with little regard to the world of printers. However, some industrious Visicalc™ users have discovered a few undocumented commands that allow us to switch our MX series printers between normal and compressed width characters without leaving Visicalc™!

Use the sequence:

```
/ P 1 <RETURN>
- " <CTRL O> <RETURN>
```

to enter compressed mode.

Returning to normal mode is just as easy. Use:

```
/ P 1 <RETURN>
- " <CTRL R> <RETURN>
```

Try it!

Appendix K

Special Interfacing Considerations

Interfacing

The MX printers are designed to interface directly with parallel printer computers like the Apple II and the TRS-80 Models I, II, and III. Other interfaces, such as RS-232 and IEEE 448 are supported by means of installing optional interface boards inside the printer case. See your EPSON dealer for availability.

Interfacing Cables

The following table identifies known cables which are wired correctly for use with the MX printer:

Computer	Parallel Interface Cable
Apple II	EPSON I/F P/N 8131 EPSON cable P/N 8232 [1] Apple Centronics parallel I/F
Apple III	Apple Computer Inc. universal parallel interface card
Atari 400	[2] Macrotronics P/N A4P-3
Atari 800	[2] Macrotronics P/N A8P-3
Atari 400/800	Macrotronics P/N A850E with 850 I/F
IBM	IBM's special interface and cable
TRS-80 Model I (without Expansion Interface)	Radio Shack P/N 26-1411 EPSON 8120 interface with EPSON 8221 cable
TRS-80 Model I (with Expansion Interface)	EPSON cable P/N 8220 EPSON cable P/N 8222 [3] Radio Shack P/N 26-1401
TRS-80 Model II	Radio Shack P/N 26-4401
TRS-80 Model III	EPSON Cable P/N 8220 EPSON Cable P/N 8222 [3] Radio Shack P/N 26-1401

[1] Requires modification to ground the most significant bit.
[2] Contact Macrotronics for additional information.
[3] Cut pin 35 at the printer end of the cable.

Notes on RS-232 Interfacing

EPSON provides support for users who prefer to use a serial interface in place of the parallel interface for which the printer was designed. EPSON developed several serial interface cards. The original cards (#8140 and #8141) cannot be used at all with GRAFTRAX^{PLUS}.

The next card (#8150), provides a 2K buffer as well as a new design to eliminate the handshaking problems in the original boards. As a result, users have full control of the printer features.

(Add 8151 - 2K buffer X-ON/X-OFF)

Pinout of 8150 (25 pin connector)

Pin 1 — Chassis Ground
Pin 2 — N/C
Pin 3 — RXD
Pin 7 — Signal Ground
Pin 20 — Busy CDTR

Pinout of 8145 2K buffer w/current LOOP

Pin 1 — Frame ground
Pin 2 — TXD (Always held at "Mark")
Pin 3 — RXD
Pin 6 — DSR (Can be strapped high by jumper)
Pin 7 — Signal ground
Pin 8 — DCD (Can be strapped high by jumper)
Pin 11 — REV Channel (busy)
Pin 20 — DTR (Busy)
Pin 17 — TTY TXD
Pin 24 — TTY TXD (Return)
Pin 25 — TTY RXD
Pin 23 — TTY RXD (Return)

Pinout of 8151

Pin 1 — Frame ground
Pin 2 — TXD (Active)
Pin 3 — RXD
Pin 7 — Signal ground
Pin 20 — Busy

All EPSON serial RS232C Interfaces act as DTE (data terminal equipment). They expect to be hooked up to and receive data from DCE (data communications equipment), which means we expect data on line 3. If the data source is a DTE, then lines 2 and 3 must be crossed. A minimum of 4 lines are needed to make an effective connection.

1. Protective ground (FG)
3. Received Data
7. Signal Ground
20. DTR (Busy handshake needed if over 300 baud with 2K buffer boards - 8141 will not work will GRAFTRAX^{PLUS})

2K Buffer - 8145 - RS232C (EIA level) and/or current loop
2K Buffer - 8250 - RS232C (EIA level) only
2K Buffer - 8151 - RS232C (EIA level) only with Xon/Xoff and ACK/NAK.

See manual that comes with I/D board for switch settings.

On I/F board: set the printer switch settings as described in this manual according to your preferences. The most important switch is SW2-3, which controls the line feed.

All three I/F's have busy on line 20, which is just the voltage level to signal when printer is ready to accept data can be inverted level by internal jumper.

Appendix L

Printer Maintenance

A clean and comfortable environment will insure the best possible service from your printer (and its operator).

Find it a Nice Home

Your MX printer is designed to take a lot of punishment, but there is no reason to become careless. In fact, it will last a lot longer if you pamper it. Treat it at least as well as your pet rock.

— Avoid operation in direct sunlight.
— Avoid use in areas where there is a high concentration of dust or grease.
— Avoid humid locations and heat sources such as a furnace.
— Never install it near large electrical machines.
— Your printer (like your computer) likes a nice comfortable range of temperatures. The safe operating range is 5°C (40°F) to 35°C (95°F).

Periodic Maintenance

Clean the printer periodically with a soft brush to remove paper and dust particles. The exterior can be cleaned with a mild detergent and water or denatured alcohol. The interior can be cleaned with denatured alcohol.

The ribbon cartridge has an expected life of 3 million characters. Replacements are available through your dealer. EPSON also offers ribbon refills on the MX-100 cartridge (Sorry, MX-100 only).

EPSON provides technical manuals for some printers to assist do-it-yourself'ers in really gumming up the works. Contact your dealer for price and availability.

Lubrication

Lubrication is an important factor in maintaining optimum performance of the MX printer mechanisms. EPSON recommends two lubricants: O-2 and G-2. O-2 is used to lubricate the carriage shaft(s) and the platen bearings. This should be done every 6 months or after printing one million lines. G-2 is used for all other lubrications. These are performed upon overhaul or after printing five million lines. Complete lubrication instructions are contained in the MX-100 and MX-80 technical reference manuals.

Changing the Print Head

The print head's expected life is 100,000,000 characters (assuming an average of 14 dots per character). Readers who are easily diverted by trivia can estimate the average number of characters printed per line and the average number of lines printed per day. Multiply them together and divide into 100,000,000. The result is the estimated print head life in days. (Then count the legs and divide by four . . .).

If and when the print head needs replacement, make sure you order the correct one for your machine. The MX-100 print head has a longer ribbon cable than the MX-80 version. Next, make sure the printer is off and the print head is cool to the touch. It may run very **hot** under heavy use.

Remove the printer lid and ribbon cartridge. Turn the head lock lever clockwise and pull the print head straight up and off.

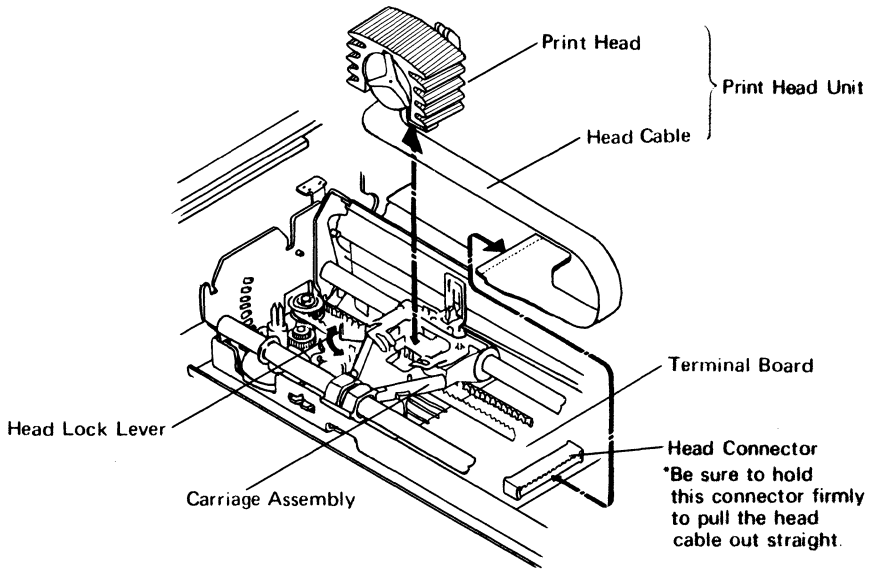


Figure L-1

Don't pull too far! It's still connected to the base of the printer by a flat ribbon cable. Grasp the cable (not the head connector block!) and pull gently but firmly to your right.

You have just removed your first EPSON print head. That was too easy!

To install a new print head assembly, snap the new cable into the head connector, lay the print head on its mount and turn the head lock lever counterclockwise.

Voila! Back in business. And it took less time than solving the Rubik's™ cube.

Appendix M

Technical Specifications

Printing

Printing method	Impact dot matrix
Printing speed	80 characters per second
Line feed time	Approximately 200 msec (at 1/6″/line.
Printing direction	Bidirectional, logic seeking. May be set to unidirectional (left to right) printing via software codes.
Character set	255 ASCII characters in a 9 × 9 dot matrix
Character size	2.1 mm (W) × 3.1 mm (H) (0.083″ × 0.11″) MX-80 2.1 mm (W) × 3.1 mm (H) (0.08″ × 0.12″) MX-100
Dot graphics density MX-80	480 dots/8″ line horizontal (normal mode). 960 dots/8″ line horizontal (super hi res mode).
Dot graphics density MX-100	816 dots/13.6″ line horizontal (normal). 1632 dots/13.6″ line horizontal (super hi res mode).
Line spacing	1/6 inch normal. Programmable in increments of 1/72 inch and 1/216 inch.
Columns (MX-80)	80 columns (normal size) 40 columns (double width) 132 columns (compressed) 66 columns (compressed/double width)
Columns (MX-100)	136 columns (normal size) 68 columns (double width) 233 columns (compressed) 116 columns (compressed/double width)

Paper

Paper feed	Adjustable sprocket feed. Friction feed on MX-100 and MX-80 F/T. Roll paper holder available for MX-80 F/T.
Paper type	Fanfold. Single sheet on MX-100 and MX-80 F/T. Roll paper on MX-80 F/T with optional holder.
Paper width	101.6 mm to 254 mm (4″ to 10″) on MX-80 101.6 mm to 393.7 mm (4″ to 15.5″) on MX-100.
Paper thickness	0.3 mm (0.012″) maximum.
Number of copies	One plus two carbon copies.

Printer

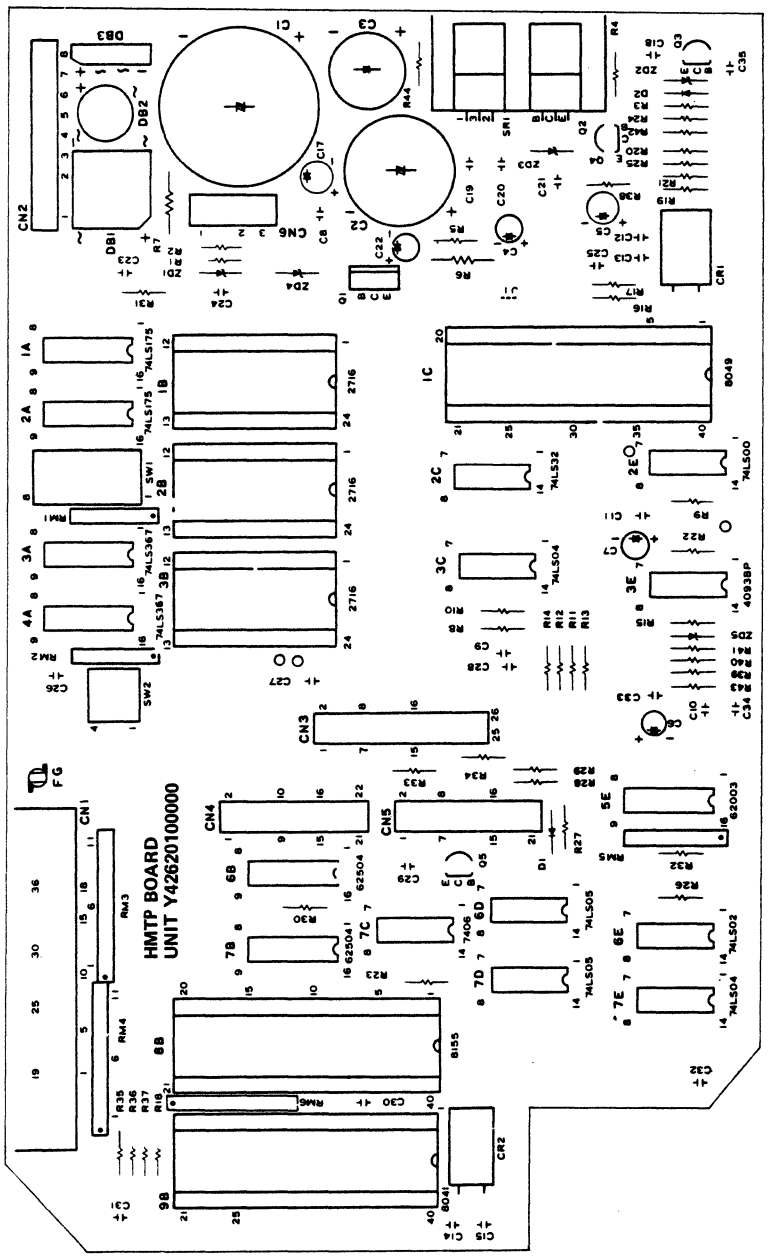
Ribbon	Cartridge ribbon (exclusive use), black.
MTBF	5 million lines (excluding print head life).
Print head life	100 × 10 (6) characters.
Dimensions	374 mm (W) × 305 mm (D) × 107 mm (H) (14.7″ × 12.0″ × 4.2″) 592 mm (W) × 393 mm (D) × 133 mm (H) (23.3″ × 15.5″ × 5.2″) on MX-100
Weight (approximate)	5.5 kg (12.1 lbs) - MX-80. 10 kg (22 lbs) - MX-100.
Power	115V plus or minus 10% 49.5–60.5 Hz.
Power requirement	100 VA max.
Temperature	Operating 5°C to 35°C (41°F to 95°F) Storage −30°C to 70°C (−22°F to 158°F).
Humidity	Operating 5% to 90% (no condensation). Storage 0% to 95% (no condensation).
Shock	Operating 1 G (less than 1 msec.). Storage 2 G (less than 1 msec.).
Vibration	Operating 0.25 G, 55 Hz (max.). Storage 0.50 G, 55 Hz (max.).
Insulation resistance	10M (ohm symbol) between AC power line and chassis
Dielectric strength	Between AC power line and chassis, AC 1 KV (RMS) 50 Hz or 60 Hz during 1 minute and no abnormal condition shall be observed.

Parallel Interface

Interface	Standard Centronics parallel. Optional RS-232 and IEEE.
Data transfer rate	4,000 CPS (max.).
Synchronization	By externally supplied STROBE pulses.
Handshaking	By ACKNLG or BUSY signals.
Logic level	Input data and all interface control signals are compatible with the TTL level.

Appendix N

Control Circuit



Appendix O

Pinout Chart - Parallel Interface

Connector use — Data exchange between the MX-printer and an external computer (parallel).
Number of pins — 36
Part number — 57-30360 (AMPHENOL)
Pin assignment — Refer to Table O-1 below.

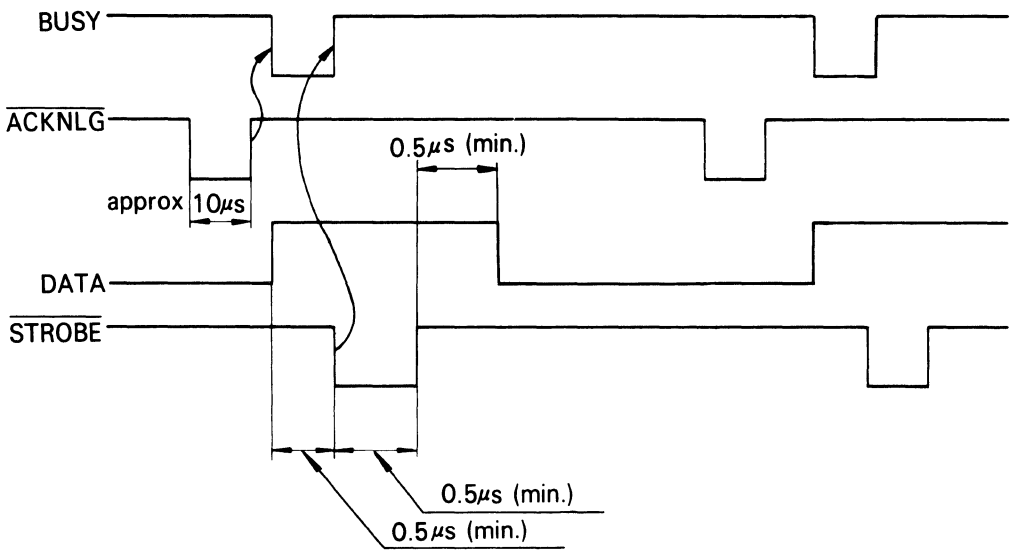
Signal Pin No.	Return Pin No.	Signal	Direction	Description
1	19	$\overline{\text{STROBE}}$	In	STROBE pulse ot read data in. Pulse width must be more than 0.5 μs at receiving terminal. The signal level is normally "HIGH"; read-in of data is performed at the "LOW" level of this signal.
2	20	DATA 1	In	These signals represent information of the 1st to 8th bits of parallel data respectively. Each signal is at "HIGH" level when data is logical "1" and "LOW" when logical "0".
3	21	DATA 2	In	
4	22	DATA 3	In	
5	23	DATA 4	In	
6	24	DATA 5	In	
7	25	DATA 6	In	
8	26	DATA 7	In	
9	27	DATA 8	In	
10	28	$\overline{\text{ACKNLG}}$	Out	Approx. 5 μs pulse. "LOW" indicates that data has been received and that the printer is ready to accept other data.
11	29	BUSY	Out	A "HIGH" signal indicates that the printer cannot receive data. The signal becomes "HIGH" in the following cases: 1. During data entry 2. During printing operation 3. In OFF-LINE state 4. During printer error status.
12	30	PE	Out	A "HIGH" signal indicates that the printer is out of paper.
13	—	SLCT	Out	This signal indicates that the printer is in the selected state.
14	—	$\overline{\text{AUTO FEED XT}}$	In	With this signal being at "LOW" level, the paper is automatically fed one line after printing. (The signal level can be fixed to "LOW" with DIP SW pin 2-3 provided on the control circuit board.)
15	—	NC		Not used.
16	—	0V		Logic GND level.
17	—	CHASSIS-GND	—	Printer chassis GND. In the printer, the chassis GND and the logic GND are isolated from each other.
18	—	NC	—	Not used.
19 to 30	—	GND	—	TWISTED-PAIR RETURN signal GND level.

Signal Pin No.	Return Pin No.	Signal	Direction	Description
31	—	$\overline{\text{INIT}}$	In	When the level of this signal becomes "LOW", the printer controller is reset to its initial state and the print buffer is cleared. This signal is normally at "HIGH" level, and its pulse width must be more than 50 μs at the receiving terminal.
32		$\overline{\text{ERROR}}$	Out	The level of this signal becomes "LOW" when the printer is in — 1. PAPER END state 2. OFF-LINE state 3. Error state
33	—	GND	—	Same as with Pin Nos. 19 to 30.
34	—	NC	—	Not used.
35				Pulled up to +5V through 3.3 kΩ resistance.
36	—	$\overline{\text{SLCT IN}}$	In	Data entry to the printer is possible only when the level of this signal is "LOW". (Internal fixing can be carried out with DIP SW pin 1-8. The condition at the time of shipment is set "LOW" for this signal.)

NOTES: 1. "Direction" refers to the direction of signal as viewed from the printer.
2. "Return" denotes "TWISTED PAIR RETURN" and is to be connected at signal ground level.
As to the wiring for the interface, be sure to use a twisted-pair cable for each signal and never fail to complete connection on the Return side. To prevent noise effectively, these cables should be shielded and connected to the chassis of the host computer and the printer, respectively.
3. All interface conditions are based on TTL level. Both the rise and fall times of each signal must be less than 0.2 μs.
4. Data transfer must not be carried out by ignoring the $\overline{\text{ACKNLG}}$ or BUSY signal.
(Data transfer to this printer can be carried out only after confirming the $\overline{\text{ACKNLG}}$ signal or when the level of the BUSY signal is "LOW".)

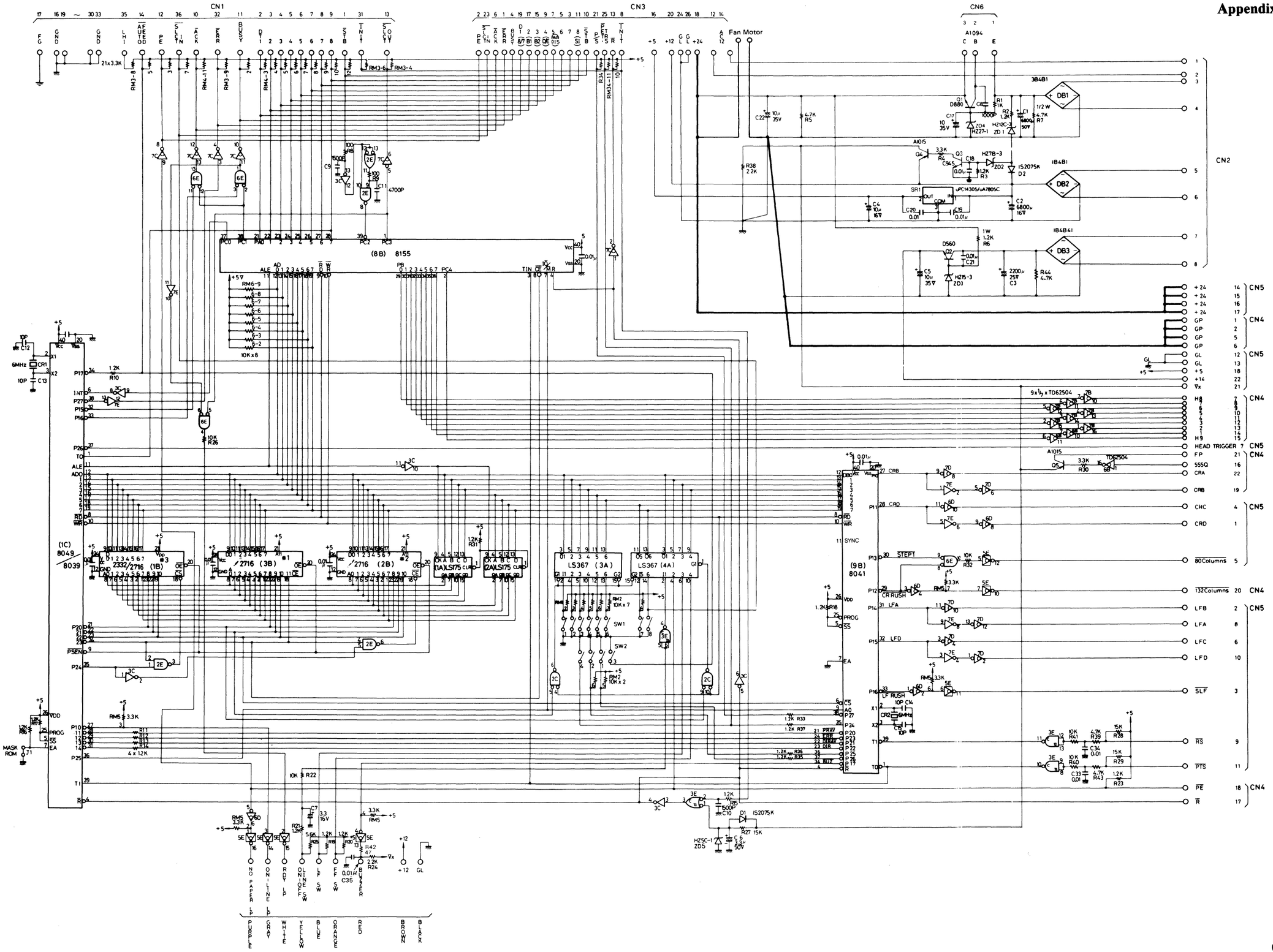
Appendix P

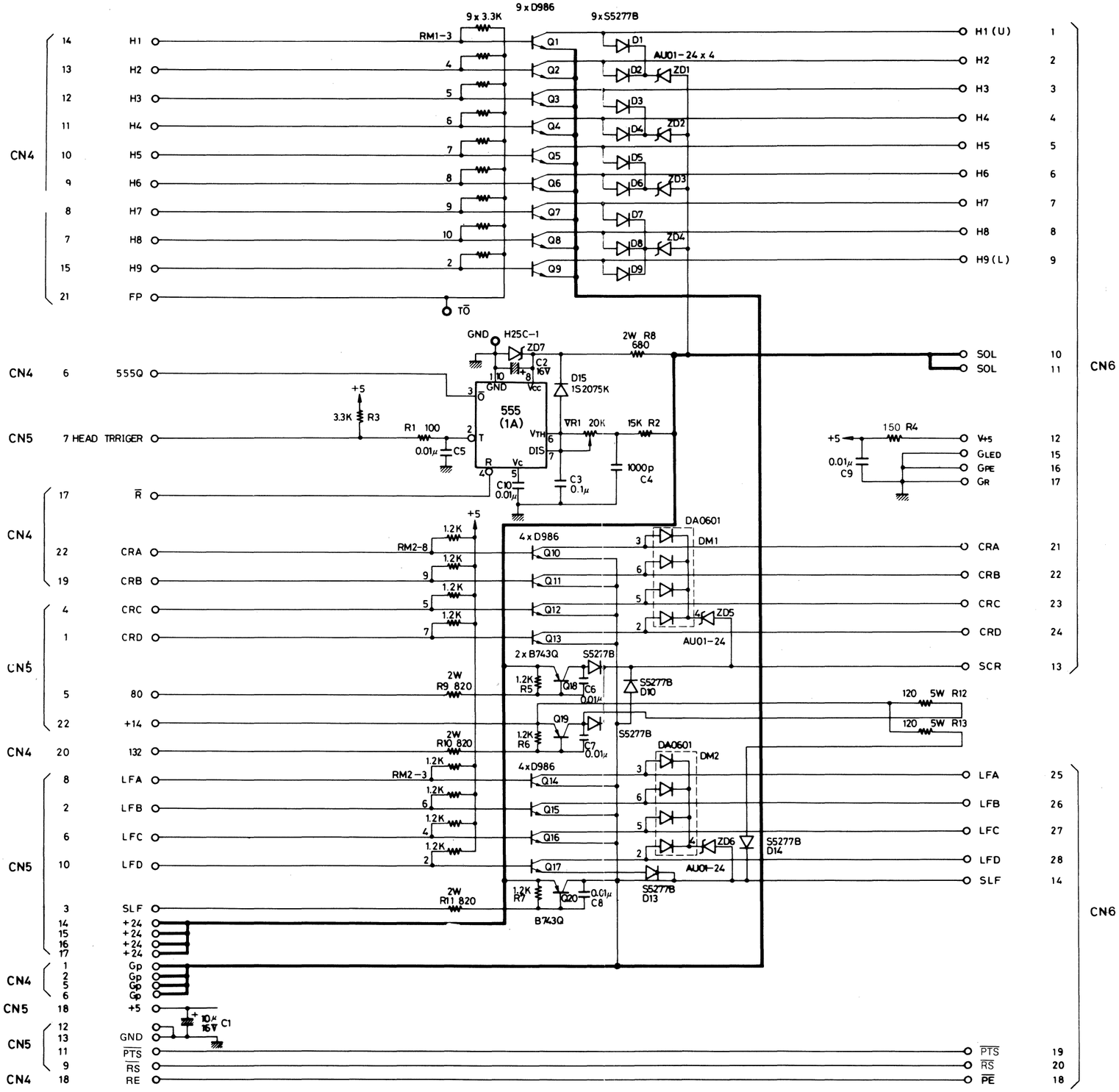
Data Transmission Sequence



Appendix Q

Schematics


CN1
CN3
CN6
Fan Motor
CN2
(8B) 8155
(1C) 8049 8039
(9B) 8041
2332/2716 (1B)
2716 (3B)
2716 (2B)
LS367 (3A)
LS367 (4A)
6MHz
MASK ROM
10K x 8
4 x 1.2K
HEAD TRIGGER
CN4
CN5


9 x 3.3K
9 x D986
9 x S5277B
RM1-3
Q1
Q2
Q3
Q4
Q5
Q6
Q7
Q8
Q9
D1
D2
D3
D4
D5
D6
D7
D8
D9
ZD1
ZD2
ZD3
ZD4
AU01-24 x 4
CN4
14 H1
13 H2
12 H3
11 H4
10 H5
9 H6
8 H7
7 H8
15 H9
21 FP
TŌ
H1 (U) 1
H2 2
H3 3
H4 4
H5 5
H6 6
H7 7
H8 8
H9 (L) 9
GND
H25C-1
ZD7
C2 16V
2W R8 680
SOL 10
SOL 11
CN6
CN4 6 555Q
CN5 7 HEAD TRRIGER
+5
3.3K R3
R1 100
0.01μ C5
555 (1A)
GND
Vcc
VTH
DIS
R
Vc
D15 1S2075K
VR1 20K
15K R2
1000p C4
C3 0.1μ
C10 0.01μ
150 R4
0.01μ C9
V+5 12
GLED 15
GPE 16
GR 17
CN4 17 R̄
22 CRA
19 CRB
4 CRC
1 CRD
1.2K
RM2-8
4 x D986
Q10
Q11
Q12
Q13
DA0601
DM1
AU01-24
ZD5
CRA 21
CRB 22
CRC 23
CRD 24
CN5
2 x B743Q
S5277B
Q18
Q19
R5
R6
C6 0.01μ
C7 0.01μ
D10
5 80
22 +14
CN4 20 132
2W R9 820
2W R10 820
2W R11 820
120 5W R12
120 5W R13
SCR 13
DA0601
DM2
RM2-3
4xD986
Q14
Q15
Q16
Q17
ZD6
D14
D13
AU01-24
CN5 8 LFA
2 LFB
6 LFC
10 LFD
3 SLF
LFA 25
LFB 26
LFC 27
LFD 28
SLF 14
CN6
14 +24
15 +24
16 +24
17 +24
CN4 1 Gp
2 Gp
5 Gp
6 Gp
CN5 18 +5
10μ 16V C1
CN5 12 GND
13 GND
CN5 11 PTS
9 RS
CN4 18 RE
PTS 19
RS 20
PE 18

Appendix R

GRAFTRAX^{PLUS} ROM Set Installation Instructions

1. Remove the upper case from the printer, by removing the four (five on MX-100) head screws from the bottom of the case (top of case for MX-100).

2. Set the printer upright, holding the top and the bottom of the case together. Use care not to lose the screws that you have just removed.

3. Carefully pull the black knob off the shaft, on the right side of the printer.

4. Slowly lift the left side of the printer case and slide it off the shaft, sticking out of the right hand side. Beware of the wires attached to the switches and indicator lamps on the top half. Carefully stand the top half of the case on its right hand side next to the protruding shaft.

5. Remove the black 2332 ROM or 2716 EPROM(s) from their socket(s), near the back of the printer. From one to three ICs are plugged into sockets 1B, 2B, and 3B (see Appendix N). Removal is performed using an IC removal tool or flat blade screwdriver. Gently pry straight up on each IC being careful not to damage the IC, IC socket, or DIP switches.

6. Using a pair of cutters cut the small wire loop (jumper J1) located near the center of the printer, next to a very large IC labeled 8049 (see Appendix N). If you found more than one IC in sockets 1B, 2B, and 3B, this jumper has probably been cut already.

7. Carefully insert the three ROMs marked 1B,2B,3B, into the three respective sockets marked 1B,2B,3B.

   A. Make sure that Pin 1 (the edge of the ROM with the moon shaped notch in it) is facing the front of the printer. Very important!

   B. Carefully push the 2716's into the sockets taking care not to bend over any of the pins.

   C. Visually inspect the chips to insure that none of the pins have been bent during installation.

8. The switch setting will need to be modified to be used with the GRAFTRAX^{PLUS} ROMs. See Appendix E for the function of each of the dip switches.

9. After you are sure that you have set the dip switches correctly, reassemble the printer in the reverse order of the disassembly.

10. Run a self test by depressing the line feed button and turning the printer's power on. If the self test does not work, inspect the 2716's for bent pins, and make sure that the jumper J1 is not connected.

Appendix S

Troubleshooting

Here are a few errors that are easy to make, but not so easy to find.

Problem	Solution
Printer won't print compressed or superscript/ subscript characters.	Emphasized mode has priority over compressed or script mode. Cancel emphasized mode first.
<ESC> 8 or switch 1-2 does not disable 'paper out' sensor.	Modify or replace cable so that computer can't monitor line 12.
Superscript/subscript leaves the printer in double strike mode.	Cancel the script modes with <ESC>"H" instead of <ESC>"T".
Can't get double width to turn off in the middle of a line.	Code 14 (142) can only be cancelled by 20 (148). <ESC>"W" 0 can be cancelled only by <ESC>"W"N where 1 < = N < = 255.
The eighth bit commands don't work right.	Don't use these commands if your computer can send 8 bits without them.
Having trouble with <ESC>"C"	Don't try setting a form length of 0 or 128. It interfaces with <ESC>"C" 0.
Get nothing but garbage on printer.	Check the printer cable connections. They must be tight to work properly.
Columns of compressed characters or line graphics don't line up properly.	Use Unidirectional printing: <ESC>"<" or <ESC>"U".
Trouble with codes 0 or 8-13.	Either avoid these codes or POKE them directly to your printer driver.
Ready light will not go on after new chips are installed.	Make sure Wire J1 is cut.
Skip-over-perforator doesn't work.	Can't set skip over perf greater than number of lines set by <ESC>"C".
Printer won't print superscript/subscript characters.	Double width overrides script characters.
Printer only prints italics and switch 1-4 is off.	Computer is holding MSB high. Example: Apple computers parallel interface.

Problem	Solution
Horizontal tabs will not work correctly.	Did not send null/128 correctly or computer cannot send correctly.
Bit image won't work correctly.	Computer cannot send 0 or other codes correctly.
Embedded printer codes do not operate properly with word processing software.	Great care must be used when sending ASCII codes with the <ESC> key. For example, to send <ESC>"S"CHR$(0), we can't use the "0" key to generate a zero code. "0" is 48 decimal. See Appendix A, and the word processing manual.

Appendix T

Mode Flags

Mixture of print mode: 00 = mixture available and XX ignores one mode when mixtured. P = pica; E = enlarged; C = condensed; S = superscript/subscript; D = double; Em = emphasized; I = italic; and U = underline

	P	E	C	S	D	Em	I	U
P		00	00	00	00	00	00	00
E	00		00	XX(1)	00	00	00	00
C	00	00		00	00	XX(3)	00	00
S	00	XX(1)	00		XX(2)	XX(4)	00	00
D	00	00	00	XX(2)		00	00	00
Em	00	00	XX(3)	XX(4)	00		00	00
I	00	00	00	00	00	00		00
U	00	00	00	00	00	00	00	

1. Printing is not performed in the superscript/subscript and enlarged superscript/subscript mixture modes. However, since <ESC>"S"; n code sets the unidirectional and double print flag, enlarged characters will be printed from left to right unidirectionally in double print mode.
   For example:

```
10 PRINT CHR$(27)"S"CHR$(1)

20 PRINT CHR$(27)"W"CHR$(1)
```

   will print unidirectional double width and double strike. Then if double width is cancelled (using <ESC>"W" 0) we will return to regular subscript mode.

2. As one character is composed in the superscript/subscript mode, the paper advances about 1/216 inch between the first and second printing as in the double print mode. By this time, the printer will enter into double print mode. Therefore, even if another double print mode were specified, it would be ignored.

3. In the condensed and emphasized mixture mode, emphasized printing is performed ignoring the condensed mode. If emphasized is set, then condensed, printing will be emphasized. When emphasized is canceled, we will be in condensed.

4. As described in (1), emphasized double print is performed unidirectionally ignoring superscript/subscript mode.

5. Italic characters and underline are available in all the above print modes.

6. Superscript and subscript are not available in both enlarged and emphasized modes. Inputting an <ESC>"S" code in these modes causes a double print flag setting. Therefore, double printing is available. When superscript or subscript printing modes are continued just after these modes, enlarged and emphasized modes must be canceled by inputting DC4 and <ESC>"F" codes respectively.

NOTICE

Please send your comments, suggestions, and any errors you might find to:

The EDITOR
Box 19669
San Diego
California 92119
U.S.A.

Thank You.

The Index

Subject **Page**

Automatic line feed, proper setting 1-12,E-2,G-1,J-1
Bell 4-1,B-1,B-2,E-1
Break a program 5-8,J-2
Cable, printer 1-3,1-12,2-2,G-2,G-4,K-1
Character width 4-2,4-5,5-1
Character set A-1,F-1
CHR$ 2-1,3-1,4-1,10-1,J-1
Codes
  ASCII 2-1,3-1,5-4,7-1,11-1,A-1
  Control 2-1,3-1,B-1,C-1,J-1,J-3,H-1
  Escape 3-3,3-10,6-1,A-1
  Problem 3-7,11-8,11-10,12-6,13-7,G-1,J-4
  Reset 3-6,B-1
Compressed characters 3-3,4-3,B-1,B-2,E-2
Dot matrix printing 11-1,11-3
Double strike 3-4,4-8,6-7,B-1
Double width printing 3-1,3-4,4-5
Emphasized printing 4-9,B-1,E-2
Exchange times 1-8
FF button 1-19,5-5
Foreign character 7-3
Form feed command 5-4,5-5
Form length — see page length
Friction feed 1-4,10-1
Graftrax^{PLUS} ROM 1-1,1-20,R-1
Graphics
  High resolution 11-1,12-1,13-3,13-9
  Line characters 7-4
  Mode 3-6,12-1
  Pin labels 12-3,12-4,12-5
  Program examples 2-6,12-4,12-7,12-8,13-11,14-2,14-6
Italics type font 3-7,4-7,B-1,E-2
LF button 1-19,1-20,3-1
Line feed command B-1,B-2
Line spacing 4-5,7-6,11-5
LLIST 3-1
LPRINT 2-3,3-1,H-1,J-2
Maintenance L-1
ON LINE button 1-19
OUT 13-7,G-3
Overstriking 6-2,7-3

Subject **Page**

Page length 5-1,5-4,5-8,8-1
Page width 9-5,12-1
Paper feeding 1-16,1-19
Paper out sensor 10-1,10-3,B-1,E-2
Paper release lever 1-13,10-2
Paper separator 1-14
Paper standards 1-13,5-1
Paper thickness adjustment 1-17,10-4
PEEK 13-4,G-1,G-2,J-5
Pinout signals H-1,I-2,K-2,O-1
POKE 11-10,13-4,13-7,15-7,G-1,J-4
Power
  Requirements 1-18,G-1
  Specifications M-1
  Switch 1-18
PR# (Apple) 2-4,J-1
PRINT 2-3,J-1
PRINT# 2-3,H-1
Print head L-1
Printer
  Interfacing 1-3,1-12,2-2,G-1,K-1
  Reset 2-6,3-6
  Speed 4-8,4-11,M-1
Ribbon installation 1-5
Roll paper holder 1-3
Schematics ..... Appendix Q
Screen dump (Apple) 15-1
Self test 1-13,1-20
Semicolons 3-1,4-10,5-6,5-7,12-3,14-1,H-1
Seven bit limitation 7-1,7-3,8-3,11-1,12-6,13-1,13-3,J-4
Shipping screws 1-4
Skip over perforation 8-1,B-2,E-2
Slashed zero 6-3,E-2
Superscript/subscript 3-3,6-6,6-9,B-1,B-2,T-1
Switch settings 1-10,E-1,E-2
TABs 3-1,4-3,9-1,9-5,B-1,B-2,G-2
Tractor feed 1-4,1-13
Top of form 1-19,1-20,5-3,5-9
Underlining 3-8,6-1,B-1,B-2
Unidirectional printing 6-4,7-6,B-1
USR 15-7
Welcome program 2-6,G-1,H-2,J-1
Word processing 6-10