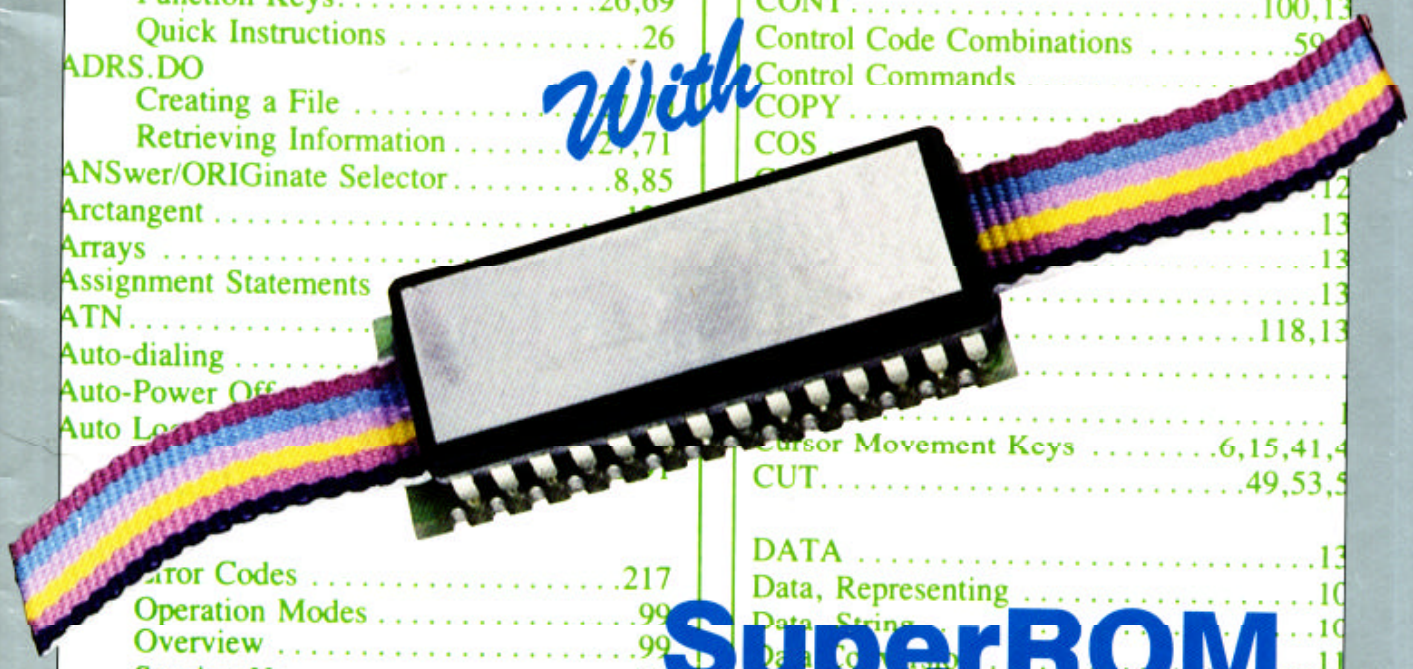
An IDG Communications Publication $3.95/CAN $4.95

portable 100

TANDY LAPTOP COMPUTING VOLUME 4 NUMBER 4 NOVEMBER 1987

INDEXING With SuperROM



ALSO THIS ISSUE:

USING TANDY 600 FILE
BUILD A VIDEO INTERFACE
FUNCTION PLOTTING
BUSINESS QUOTE FORMATS

PLUS: New products, more programming tips, and software utilities!

SUPER ROM

Lucid Spreadsheet Write ROM Database Outliner

NOW YOU CAN REALLY HAVE IT ALL!

All on one ROM. Truly the finest four programs available for the Model 100 — guaranteed. Try it for 30 days. If you aren't blown away by the excellence return it for a full refund.

$199.95

The four best programs for the Model 100 all on one ROM. 32K of power without using any RAM for program storage. This is the PCSG Snap-In ROM that just presses easily into the little ROM socket in the compartment on the back. You access the four right from the main menu like built-ins.

Write ROM — the definitive word processor for the Model 100. Function key formatting or dot commands. Search and replace. Library feature — inserts words, phrases or whole documents into text from just a code. MAP lets you see a picture of your document. In all there are 60 features and functions. No one can claim faster operation. FORM lets you create interactive forms with on-screen prompts that you can answer from the keyboard. Nothing else for the Model 100 compares with the features of Write ROM. Exactly the same as the Write ROM sold as a single program. Infoworld says it "makes the Model 100 a viable writing unit ... surpassed our highest expectations for quality and clarity."

Lucid Spreadsheet: This is the one PICO magazine says "blows Multiplan right out of the socket" and Infoworld performance rated as "excellent" and said "makes the Model 100 compute." Gives you features you cannot get with Lotus 123. Lets you build spreadsheets in your Model 100 that would consume 140-150K on a desktop. Program generating capability with no programming knowledge required. Variable column widths. Includes find and sort with function key control. It's fast, recalculates like lightning. No feature has been taken from the original, only new ones added.

Database: This is a relational data base like no other. You can do everything from mailing lists to invoices. No complicated pseudo-coding, you create input screens as simply as typing into TEXT. You are not limited by size; you can have as large an input screen as you wish. Prints out reports or forms, getting information from as many files as you like. Complete math between fields. Total interface with Lucid worksheets.

Outliner: Does everything that Thinktank does on a PC but a whole lot better. Includes a Sort for your headlines. Lets you have headlines of up to 240 characters. Has cloning, hoisting and sideways scroll up to 250 characters. Like Lucid, this one sets a new standard for outliners. This is the way to plan and organize your projects.

Present Lucid and Write ROM owners can upgrade for $150. If you have both it's $125.

As usual PCSG sells the Super ROM on a thirty day guarantee. If for any reason you are not satisfied, simply return it for a full refund.

We are excited about this product. Super ROM gives the Model 100 the true power of a desktop. No other multi-program ROM has software that compares. But don't take our word for it. We invite you to make that comparison yourself. Priced at $199.95 on Snap-In ROM.

Got stuck with somebody else's multi-ROM? We'll upgrade it for $150.

(214) 351-0564

PORTABLE COMPUTER SUPPORT GROUP

11035 Harry Hines Blvd., #206, Dallas, TX 75229

MC, Visa, American Express, Check, or C.O.D.

Circle 40 on Reader Service Card

SOUNDSIGHT ANNOUNCES

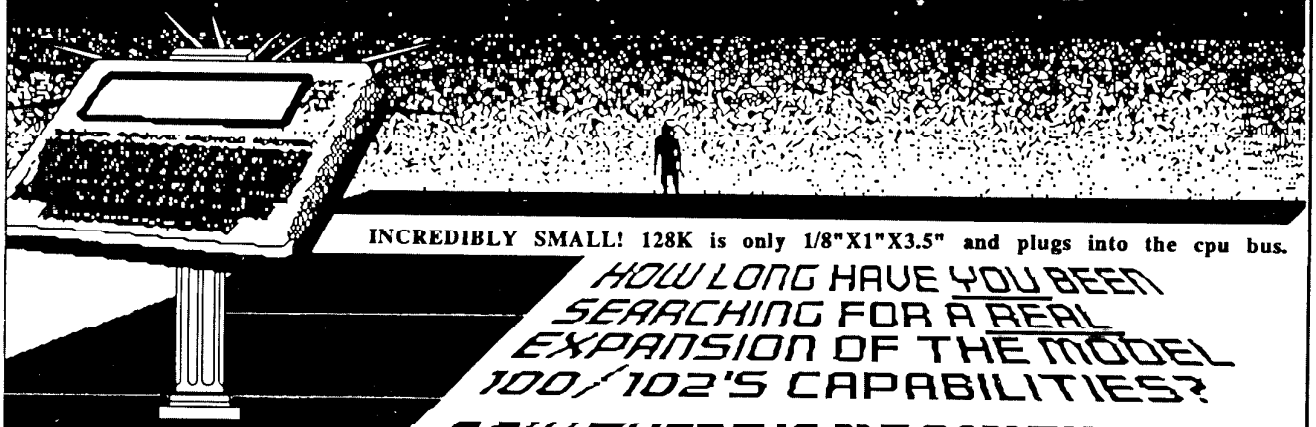
MEGAMEMORY

A NEW ERA IN RAM.

128K TO 10 MEGABYTES

RAM EXPANSION FOR THE TANDY 100 AND 102 COMPUTERS

CREATES AND EDITS* FILES AS LARGE AS THE AMOUNT OF RAM PURCHASED. FULL I O FUNCTIONS IN BASIC. FASTER, IN SOME APPS, THAN THE MODEL 100/102. OPTIONAL 4 PLACE ROM BANK. PARALLEL CPU BUS. TOTAL SIZE OF 1 MEGABYTE MODULE IS 1"X1.3"X3.5". DOES NOT COVER COMPUTER'S I O PORTS OR BATTERY COMPARTMENTS.



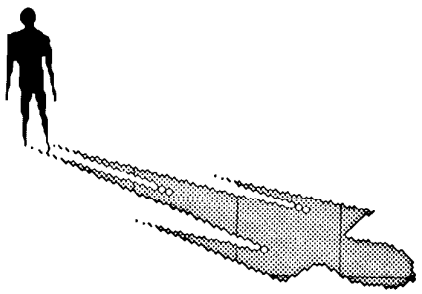
INCREDIBLY SMALL! 128K is only 1/8"X1"X3.5" and plugs into the cpu bus.

HOW LONG HAVE YOU BEEN SEARCHING FOR A REAL EXPANSION OF THE MODEL 100/102'S CAPABILITIES?

NOW THERE IS MEGAMEMORY.

Utilizing the same proven operating system software SoundSight placed in Standard Oil Service Stations from Alaska to Florida, MEGAMEMORY now offers FULL TEXT EDITING of files of any size existing in the RAM bank. *Text Editor Priced at $75.00

Currently undergoing tests for use by the LA Co. Sheriff's Dept, Megamemory also offers 3rd Party Development,(C language), POS systems and Vertical Market Applications.



The advantage of our widely used operating system lies not only in providing the first Text Editor accomodating model 100/102 files greater than 32k, but in our coming support software which includes the Text Editor, Spreadsheet, and Database.

PRICES: 128K-$329.00, 256K-$529.00, 384K-$625.00, 512K-$699.00, 640K-$769.00, 768K-$839.00, 896K-$899.00, 1 MEGABYTE-$965.00.

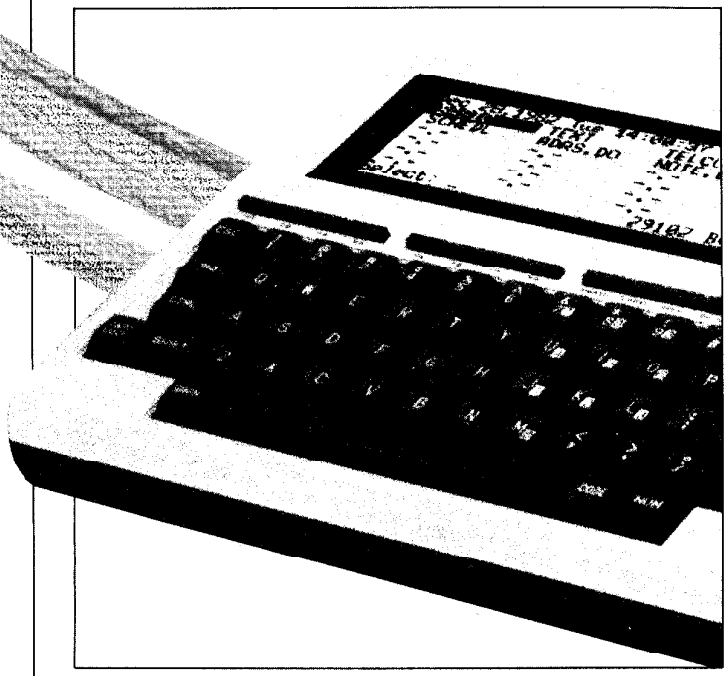
Additional memory prices available up to 10 megabytes in bottom mounted RAM pan. All RAM bank modules are actual byte sizes listed and do not include the Model 100 or 102's RAM.

CONTACT:

SOUNDSIGHT MBM INC. 225 W. Broadway, Suite 509, Glendale, CA., 91204 Ph (818) 240-8400

Circle 76 on Reader Service Card

portable 100



VOL. 4, NO. 4 NOVEMBER 1987

ON THE COVER

FEATURES

DEPARTMENTS

Cover Photograph by Peter Lanza

Editor
Roger Strukhoff
(603) 924-9471
Managing Editor
Steven F. Smith
Technical Editor
Alan L. Zeichick
Assistant Editor
Janet M. Tiampo
Senior Contributing Editor
Carl Oppedahl
Contributing Editors
Greg Bouffard
Robert S. Dupree
G.N. Gilbert
Jim Gove
Stephen Schenkel
Art Director
Lou Ann Morin

Publisher
Jeff DeTray
Associate Publisher
Jim McBrian
(603) 924-9471
Advertising Coordinator
Gayle Patria

Corporate Production Director
Dennis Christensen
Manufacturing Manager
Susan Gross
Assistant Manufacturing Manager
Leslie Walden
President/CEO
Michael S. Perlis
Vice President/Finance
Roger Murphy
Director of Circulation
Frank S. Smith
Circulation Manager
Bonnie Welsh
Single Copy Sales Manager
Linda Ruth
Direct Marketing Manager
Paul Reuss
Telemarketing Manager
Elizabeth R. Kehn
Director of Credit Sales
William M. Boyer
ISSN 0893-942X

Portable 100 is an independent journal not connected with any hardware, software, or peripheral equipment manufacturer. Portable 100 is published monthly by CW Communications/Peterborough, Inc., 80 Elm St., Peterborough, NH 03458. Second Class postage paid at Peterborough, NH and additional mailing offices. Subscription rates in the U.S. are $24.97 for 12 issues. In Canada $45.97 (Canadian). In Mexico, $29.97, U.S. funds drawn on a U.S. bank. Other foreign subscriptions (surface mail) are $44.97 for 12 issues; air mail is $79.97, U.S. funds drawn on a U.S. bank. Address all subscription correspondence to Portable 100, *80 Elm St., Peterborough, NH 03458.* Portable 100 *makes every effort to assure the accuracy of articles published in the magazine.* Portable 100 *assumes no responsibility for damages due to errors or omissions. Volume 4, Number 4, November, 1987.*

The Choice is Yours

This is the fourth issue of the revitalized *Portable 100*. Letters keep pouring in from readers who are glad to have their magazine back. One of the keys to *Portable 100*'s popularity is its technical content. Maintaining vigilance in all things technical for the past couple of years has been Alan Zeichick. Alan is now working full-time here in Peterborough, and we thought you'd like to hear a few words from him. Alan. . .

"I talked with a young man yesterday, a third-year computer-science student at a midwestern university. He called looking for advice—which computer should he buy? He wanted a portable and he wanted to buy from Radio Shack (despite the fact that he could get huge discounts on IBM and Zenith through his university). The Tandy 102 is affordable, but limited; the 200's not much better. The 600 is on sale, but obviously a dead-end. And, the new Tandy 1400 LT is appealing because of its power and IBM compatibility, but the price is steep for a college student.

"It was a difficult question. A year ago, six months ago, I would have recommended the Tandy 102 as a good, general-purpose notebook computer; but with the MS-DOS standard firmly entrenched, the 1400 LT offered the most promise, especially for a bright computer-science major.

"When the Model 100 was introduced, it gave the most bang for the buck. At *Portable 100*, we bragged about its built-in screen, handy modem, communication ports, full-sized keyboard. Compared to the Compaq Portable, Kaypro IV, and PCjr, it was quite a deal.

"But the world changed. The Traveling Writer spawned the Ultimate ROM II, but Visicalc and WordStar inspired such stars as Lotus 1-2-3 Release 2, dBase III+, MultiMate Advantage and Aldus Pagemaker. And, a 1,200-bps modem card for an IBM PC is available for less than $100.

"In a world of personal laser printers, 3Com networks, hard disks and CD-ROM (compact disc, read-only memory), how can the Model 100-family laptop survive?

"The way it always has—as a specialty, niche product. No, the Tandy 102 is probably not the best computer for the technically-minded college student, or for home education, or even for the average hobbyist. You'll find, and *Portable 100* will cover, those applications that the Model 100 shines at: journalism, field instrumentation, device controlling. Wherever the 1400 LT is overkill, or IBM compatibility is a handicap, you'll find a Model 100.

"The computer industry is constantly changing. That's not a surprise, but what is surprising is how responsive Tandy is to those needs. Unlike Epson America, for example, which abandoned its proprietary Valdocs computers in favor of its IBM clone, Tandy still supports its older machines while concentrating on its newer models. And like Compaq, Tandy's machines aren't simple copies of the competition, but new models in their own right. Tandy continues to set its own trends."

That said, let's get on with the magazine.

Roger Strukhoff, Editor

Rebirth of a Classic

Glad to have you back. The new *Portable 100* came in just as my Model 100 was reborn by Traveling Software's Ultimate ROM II with the 60-character display. Wow! I abandoned the little guy years ago when serious work required more than 40 characters. Many thousands of dollars of laptops followed. If only Tandy would offer an 80-character option, they would consume the market.

Other wonderful things: ortho rubber bands plus Merrit Computer's SafeSkin keyboard protector. Totally quiet and sealed against dirt, too. PG Design's RAM bank: three banks of 32K is heaven, and battery life is fine. Finally nicad conversion. Works great.

My Model 100 is again my constant companion. Now if only I could tuck in a cold beer dispenser. . . .

Albert R. Karel
Los Angeles, CA

DON'T FORGET THE TECHIES

Just a line to let you know how pleased I was to receive my copy of *Portable 100*. Frankly, I never expected to see this magazine again! I considered *Portable 100* to be one of the best magazines delivered to my door each month, and there are many to which I subscribe.

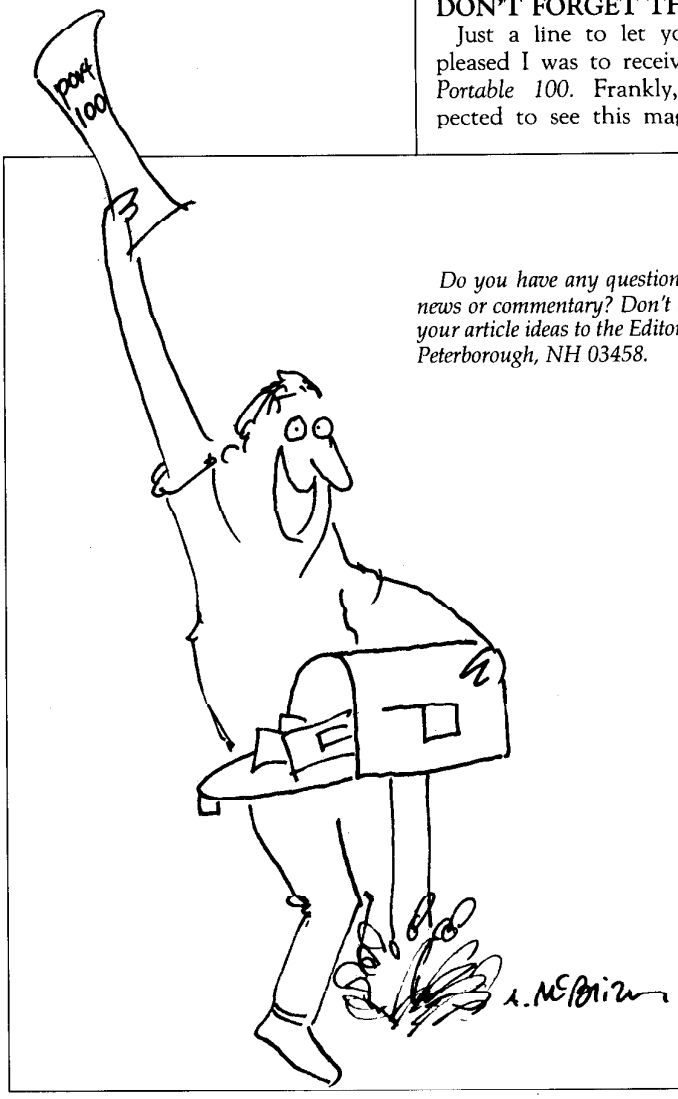
In the September issue's editorial, you discussed some future articles, and one of those mentioned is the capability of using a monitor with the Model 100. Many of us are looking forward to that article. Of course, that capability is expensive and requires a disk drive that I do not want or need. The Model 100 owners here are anxiously awaiting that article.

I think there has been some hesitation to include articles that are very technical in nature, perhaps feeling that the average reader will not understand. This is especially true of electronic articles that require knowledge of schematic diagrams and construction techniques. Please don't hold these articles back because there are many of us that do have technical backgrounds. And if the subject is useful to those who have not, we can help with the conversions and modifications. But if we don't have those articles, many useful changes to the Model 100 might never be known.

I have converted my Model 100 to rechargeable batteries such as suggested in your August 1987 issue. The only difference is that I used a diode in conjunction with the resistor to prevent unwanted discharge of the cells. It works fine and sure does save on battery purchases.

Many of the ham, or amateur radio operators in this area are using the Model 100 for digital modes of communication, including radio teletype, Morse and the new packet communications mode. Ham communications requires that we both listen and then transmit, but we can't do both at the same time as we can with a telephone modem—so conventional communications software isn't very good for our purposes.

We have been trying to find a split-screen terminal program to solve our problem. A split-screen communica-



Do you have any questions, suggestions, news or commentary? Don't be shy — send your article ideas to the Editor at 80 Elm St., Peterborough, NH 03458.

USE YOUR DESKTOP COMPUTER AS A DISK DRIVE FOR YOUR M-100.

Try *Disk*+ for 30 days. If you aren't as excited as we are, return it for a full refund.

When we designed *Disk*+ we did it out of necessity. We wanted a way that we could just connect a Model 100 to our desktop computer with a cable and save files onto the desktop's disk drive. We wanted it to be so simple to use it would be self-explanatory.

Picture this. *Disk*+ comes to you on a Snap-in ROM and a diskette for your desktop. You take a quarter and open the little compartment on the back of your Model 100. Then you just press the ROM into the socket. *Disk*+ appears on your main menu just like a built-in.

You connect your Model 100 to your other computer using an RS232 cable (available from PCSG for $40).

You just place the *Disk*+ diskette into the desktop's drive and turn on the computer. It powers up automatically and says "awaiting command" on your desktop's screen. Then you just put the widebar cursor on the Model 100 main menu on *Disk*+ and press ENTER. You are shown your RAM files arranged just like the main menu.

To save a file to your other system's disk drive, you just move the widebar cursor to the file you want to save and press ENTER. It is saved instantly with no further action.

To look at the disk directory, you just press a function key on your Model 100. You see immediately the disk directory on your Model 100 screen, and it is arranged just like your Model 100's main menu.

To load a file from the diskette to your Model 100, you just move the widebar cursor to the file and press ENTER. The file is transferred to your Model 100's RAM instantly. You can press F8 and go back to the main menu, and the file you loaded from diskette is there, ready to use.

It is so nice to be able to keep your documents, programs (both BASIC and machine code) and *Lucid* spreadsheet files on the diskette, and bring them back when you need them. All files are ready to run or use with no changes or protocol by you.

If you have access to a desktop computer and don't have *Disk*+, then evidently we have done a poor job telling you about it.

All files and programs that you load or save, go over and come back exactly as they are supposed to be because of full error checking. This guaranteed integrity is really a comfort. *Disk*+ is wonderful in so many other ways. For example, you can do a "save all" of all your RAM files with just a touch of a function key. That group of files is saved on the diskette under a single filename with a .SD (for subdirectory) extension. Any time you want, you can bring back all those files at once, or just one or two if you like, again with one-button ease.

Disk+ takes up no RAM. That's zero bytes either for storing the program or for operating overhead.

What really excites most *Disk*+ users is text file cross compatibility. Your Model 100's text files are usable on your desktop computer, and your desktop's text files become Model 100 text files.

This means you can write something on your Model 100, and with *Disk*+ transfer it instantly to your desktop and start using it right away on your bigger computer. Or the way we like to work is to type in a document on the desktop computer and then transfer it to our Model 100 with *Disk*+. Then we print out the document, beautifully formatted, using WRITE ROM.

Disk+ works with just about every micro sold, from IBM PC and its clones, to all Radio Shack computers (yes, all), to Apple II, Kaypro, Epson and most CPM. Just ask us. More than likely, your computer is supported.

Incidentally, hundreds of Model 100 owners have gone to their Radio Shack stores and bought a color computer because it is so low priced, and with *Disk*+ they have an inexpensive disk drive.

And if that weren't enough, how about this: *Disk*+ also provides cross-compatibility between different computers like IBM, Apple or the Model 4 using the Model 100 as the intermediary device. Quite a feature!

The snap-in ROM is really great because you can use other ROMs like *Lucid* or WRITE ROM. They snap in and out as easily as an Atari game cartridge and you never lose your files in RAM.

Anyone who ever uses *Disk*+ simply can't do without it. But so many times we have had new users call us and say, "Wow! I had no idea when I ordered it that *Disk*+ would be so fantastic. I just couldn't believe that I could use my desktop computer's disk drive with my Model 100 just like it is another main menu."

That's why we sell *Disk*+ on a thirty-day trial. If you aren't completely satisfied, return it within thirty days for a full refund. Priced at $149.95 on Snap-in ROM. MasterCard, Visa or COD.

1-214-351-0564

PORTABLE COMPUTER SUPPORT GROUP

11035 Harry Hines Blvd., #207, Dallas, Texas 75229 • 1(214) 351-0564

PCSG provides hotline software support for the Model 100. Call us at 1-214-351-0564

Circle 71 on Reader Service card.

tions program would allow us to type answers to incoming data into a buffer. When it is our turn to transmit, the information can be sent. The split screen normally consists of two sections, separated by a horizontal line. The bottom section contains the outgoing information, perhaps only one or two lines, which would scroll to make room for new data. We've not been able to find such a program for the Model 100, though such software for other computers does exist. Can you help?

James Parsons
San Angelo, TX

We've never seen a Model 100 communications program such as you described. If any reader or amateur radio enthusiast can offer assistance, please drop a line to the Editor of Portable 100. *-Ed.*

JERKY INTERPRETER REVISITED

Regarding the question in August's I/O column, regarding BASIC programs sometimes locking up for a short time, I have a suggestion.

This has been a problem with Microsoft's interpreted BASICs for years, since, because of dynamic string memory allocation, BASIC will sometimes pause execution of a program in order to clean up the string space, a process known as "garbage collection." This activity will occur at various times within the running of a BASIC program, depending on such factors as the number and total length of string variables, activity of those variables, and the total rainfall in Washington state.

The only real way I've found to handle the garbage collection problem is to force a clean-up when convenient for me. This can be accomplished by executing a FRE(A$) at a point in the program where you want to force garbage collection. If the string clean-up is forced often enough, it takes less time and won't occur at undesirable moments, such as during an input routine.

John J. Genzano, III
West Chester, PA

A SHIFTY GARBAGE COLLECTOR

In the August 1987 I/O column, a reader asked why BASIC programs with many variables or large arrays sometimes appear to freeze when he is entering data. You responded with some good suggestions, but I suspect his problem is caused by one of two aspects of the way Microsoft BASIC works: variable shifting and garbage collection.

Numeric variables are stored in the free memory just above the text of the BASIC program. Simple variables are stored first, followed by array variables. When you dimension an array you allocate space for it in the lowest available memory. If you then introduce a new variable, it is added to the end of the current simple variables and any existing arrays are moved higher in memory to make room for it. This takes time—particularly for large arrays. But it can be avoided entirely by initializing all simple variables before dimensioning arrays. You could assign values to each of them, but the best way is to dimension them (yes, you can dimension simple variables). For example,

DIM AL,B,C,EX,EY,I,J,K,IN$,D$

creates these variables, assigning zero to the numeric ones and nulls to string variables. As a side benefit, all the variables are listed near the top, so you know what's already in use when you want to add more. You should perform any variable type definitions first (such as DEFINT I-K) so that the variables will be created with the proper sizes. Listing the most-used variables first yields a minor speed adjustment, since BASIC searches for them in order.

When you dimension a string, you really create a pointer to the actual string. The string itself may be within the program (in a data statement or as a string literal), or it may be in the "string space" between the end of the highest array and the top of available memory. The size of string space is set with the CLEAR command; the default is 256 bytes on the Model 100. If you assign a new value to a string already in string space, the new value is placed in the unused area, and the old one is abandoned. The latter is still there—but there's no pointer to it. When string space becomes full, BASIC does garbage collection to consolidate the strings and to get rid of any strings in string space that no longer have a pointer addressing them. The time taken by the garbage collection depends on the length of the active strings, not on the dead ones.

There is no complete cure for garbage collection. Programs were developed for the old TRS-80 Model I and other computers to speed it up with a faster collection algorithm, but I don't know if any are available for the Model 100. By increasing the size of string space you can delay the inevitable. The FRE function with a string argument forces garbage collection, so you can make it happen at a more convenient time.

If you do a lot of string swapping (common in sorting routines), you can swap the pointers instead. Or if the strings are in an array, you can set up an integer array as pointers to the string array and use an indexed sort method; you can have as many sorted versions of the same string array as you want.

Everett Ogden
Delmar, NY

THE PAUSE THAT ANNOYS

Welcome back! I'm looking forward to the new *Portable 100*. The content has always been good; hopefully, delivery will now be equally good.

I noted a letter in the August issue regarding the annoying pauses which often occur when typing in data when running BASIC programs. In addition to the techniques mentioned in the reply, another possible solution is to increase the amount of string space. This is because the pauses are often caused by the computer doing housework, rearranging strings. The more string space there is, the less housekeeping there is to do. For example, if the beginning of the program has a CLEAR 1000 statement, try CLEAR 2000.

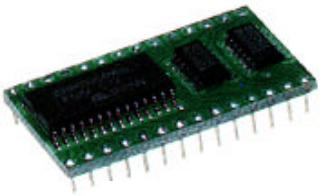
Since that increases the amount of RAM the program requires to run, it might not always be practical.

Neil Smith
Seattle, WA

CORRECTION

Portable 100 ran two items in the August issue concerning products from Portable Computer Support Group. One item concerned carrying cases, the other a product called the Business Analyst. These products are, in fact, no longer available from PCSG and were not available when the August issue was published through no fault of PCSG. *Portable 100* regrets the error.

EXPANSIONS!!!

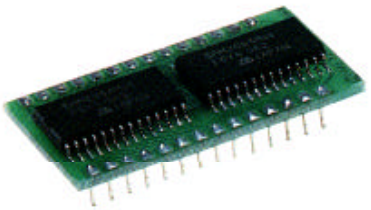


MODEL 100/NEC PC-8201A
8K Memory Module
$23—three or more $19 ea.

Easy to install. Open the case and plug-them-in. Each module expands your memory by 8K bytes. Extra low power components mean long battery life. Very low profile means a proper fit in the NEC PC-8201A. Detailed instructions make installation quick and easy. You can expand your Model 100 to 32K and your PC-8201 to 64K (2 banks of 32K each).

NEW!!!
MODEL 102
8K Memory Module
only $9.95

Easy to install. Just open the hatch and plug it in.



TANDY 200
24K Memory Module
$65—two for $59 ea.

Simply pop open the hatch with a coin and plug in one or two of these modules. Each module adds a 24K bank of memory to the TANDY 200. It's like getting two more machines. The built in COPY function key copies files between banks instantly. Like our 8K, we build these with the lowest power and most reliable memory chips available.

THE PURPLE POLICY

Try any of our products for 30 days, satisfy yourself that our service, quality, and prices add up to the best value anywhere—if not completely satisfied, you can return the product for a full refund. Prices include UPS surface shipping (in Continental USA)—even the phone call is free.

30-day money-back guarantee.

IT'S EASY TO ORDER

Send your order with payment to the address below. Or, if you prefer, credit card orders can be handled by phone—VISA, MasterCard, and American Express are welcome. California residents add 6% sales tax. Checks allow 3 weeks to clear.

1-800-732-5012 TOLL FREE
Orders only (8am—5pm PST)

(805) 987-4788 In California
For orders or customer service

PURPLE COMPUTING

420 Constitution Ave.
Camarillo, CA 93010

Telex: 888661 (PURPLE)

Canada: Canada Portable Computer, (604) 534-6441
Australia: Softech Computer Services, (2) 419-8899

PORTABLE PRINTING

30-DAY MONEY-BACK GUARANTEE

MADE FOR TRAVEL

SIZE and WEIGHT are the most important considerations when going portable. This printer is half the size of the Model-102 and weighs only 2.2 lbs.

RUGGED

Because it's thermal, there are no ribbons, no ink, no pins to wear out or replace, all you need to take with you is paper.

FEATURES

- 2.2 lbs., size 11" x 4¼" x 1¾".
- Epson MX/80 bit-map. graphics.
- Built-in auto self test function.
- 8½" wide thermal paper.
- Excellent User's Manual.
- 40/80/160 chars./line, 40 chars./sec.
- 512 byte text buffer.

(ACTUAL print sample)
80 Chars/Line, 40 CPS!
Enlarged!
underlined Emphasized!
CONDENSED gives you 160 chars. per line!

$99 GRAND TOTAL

Includes a 20' roll of paper, four "C" batteries, and a custom-made 18" cable for your computer's parallel printer port. (Please specify computer make and model.) You get everything you need to start printing immediately!

IT'S EASY TO ORDER

Send your order with payment to the address below. Or, if you prefer, credit card orders can be handled by phone—VISA, MasterCard, and American Express are welcome. California residents add 6% sales tax. Checks allow 3 weeks to clear.

1-800-732-5012 TOLL FREE
Orders only (8am—5pm PST)

(805) 987-4788 In California
For orders or customer service

PURPLE COMPUTING

420 Constitution Ave.
Camarillo, CA 93010

Telex: 888661 (PURPLE)

Canada: Canada Portable Computer, (604) 534-6441
Australia: Softech Computer Services, (2) 419-8899

Indexing with Super ROM and the Model 100

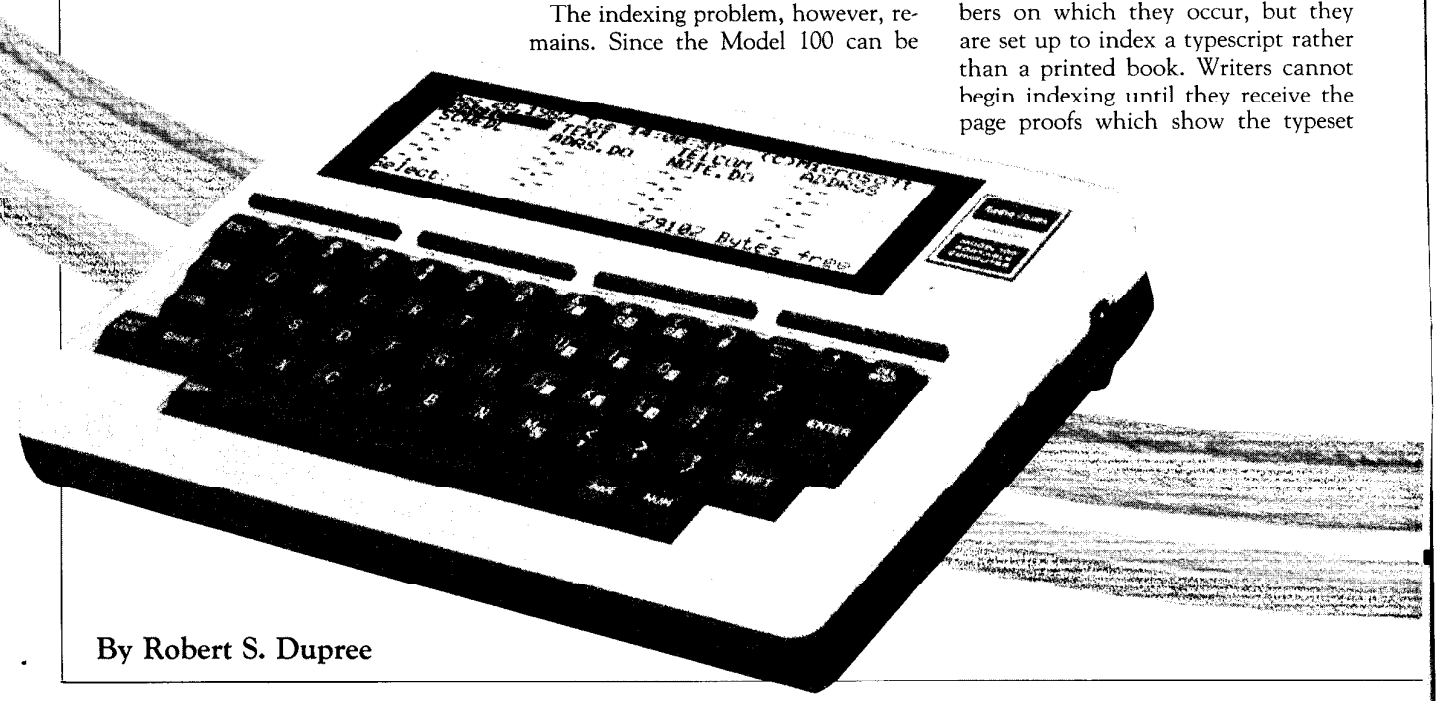
It's a writer's delight: a completely edited word-processed version ready for typesetting.

Few tasks are more tedious and prone to error than book indexing. I remember looking foward to the personal computer's word processing capabilities and its ability to sort lists; it seemed reasonable to expect the computer would also ease indexing pains, but as I soon discovered, sorting programs and even the indexing capabilities of some advanced word processing programs leave much to be desired. (Even so, they are better than a typewriter and index cards!)

My life as a writer changed drastically, however, when I acquired a Model 100. I conducted library research and took notes with unprecedented efficiency. I could cut and paste notes from one file to another, replicate bibliographical entries from my library references and copy them to my final printed text, and avoid the frustration of retyping and the inevitable errors that come with it.

The indexing problem, however, remains. Since the Model 100 can be used almost anywhere for short spurts of typing (for instance, while waiting in the doctor's office or even in the carpool), it relieves some of the tedium of entering indexed items. But I knew if a program could be devised that would facilitate index editing after the list of items and page numbers had been typed in, the Model 100 would be an indispensable companion.

Word processing programs with indexing features can create a list of designated words and the page numbers on which they occur, but they are set up to index a typescript rather than a printed book. Writers cannot begin indexing until they receive the page proofs which show the typeset



By Robert S. Dupree

text divided exactly as it will appear from the first to the last page. The word processor, can't help with this task because the original typescript pages do not correspond with the typeset ones.

An alternative, however, is available thanks to PCSG. I have worked out the following procedure, using their Super ROM, to produce a ready-to-typeset index from page proofs. Since I send my ASCII files directly to a typesetting machine through an interface, typographical errors, which are so difficult to spot and correct, are reduced because basic entries are typed in only once.

THE RECIPE

Faced with an impending deadline (most indexing must be squeezed in at a critical point in book production), I made a copy of my page proofs and, using a felt-tip highlighting pen, marked the words I wanted indexed. The list was ready to be typed into the computer.

Here's how you can enter your own index by using the Lucid Database and Write programs:

First, make up an entry form like the following and call it INDEX.DO: item:[*a*#] page:[*b*#] Create a file using Lucid Database (four on the Super menu) and the INDEX.DO view

```
beauty                59
prosody               59
metaphysics           59
energy                59
muscle                59
destiny               59
Raymond, Marcel       59
cause, formal         59
cause, poetic         59
nervous system        63
muscle                63
adolescent            63
physiognomy           63
action                63
neck                  63
school                63
psychology            63
aggression            63
```

Figure 1. The author's sample program shows the index without leading zeroes.

```
action   62
action   63
action   66
adolescen63
aesthetic60
aesthetic66
aggressio63
anacoluth66
animal   66
muscle   59
muscle   61
muscle   62
muscle   63
myopsyche61
neck     63
nervous s63
nostril  61
organ    61
```

Figure 2. An example of a .DO file in two column format.

```
action   62, action   63, action
66, adolescen63, aesthetic60,
aesthetic66, aggressio63,
anacoluth66, animal   66,
animality61, animality62, beauty
60, beauty   59, cause, fo59,
cause, po59, Cheselden65, child
64, child    63, child    61,
correspon62, cry      65, cry
```

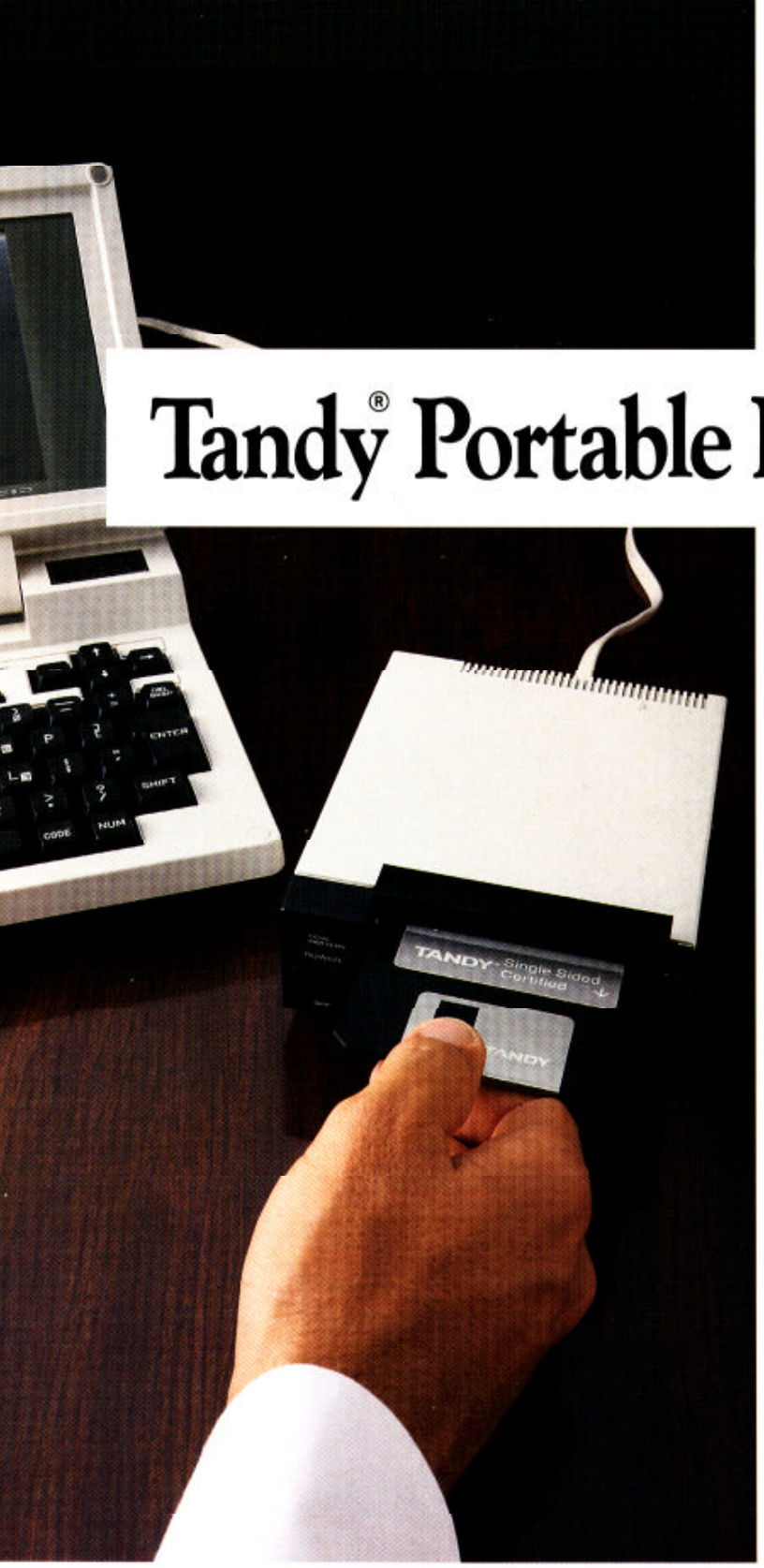
Figure 3. Use Lucid Write and the F1 key to replace all carriage returns with a comma and single space.

```
action   62, 63, 66, adolescen63,
aesthetic60, 66, aggressio63,
anacoluth66, animal   66,
animality61, 62, beauty 59, 60,
cause, fo59, cause, po59,
Cheselden65, child    61, 63, 64,
correspon62, cry      64,  65,
deafness 65, destiny  59, dream
62,  64, dynamogen62, energy   59,
```

Figure 4. Before the final edit, you must put in carriage returns, delete commas after the last numbers of each entry, and clean up the text where needed.

```
action 62, 63, 66
adolescencee 63
aesthetics 60, 66
aggression 63
anacoluthon 66
animal 66
animality 61, 62
beauty 59, 60
cause, formal 59
smell 62
space 65
Storch 61
teacher 63
transfiguration 60
transformation 60, 62
Valery, Paul 66
vengeance 64
violence 62, 63
```

Figure 5. Removing spaces and commas after the last page number.



Tandy Computer Accessories: Because there is no better value.™

Tandy® Portable Disk Drive

The ultimate storage solution for your Tandy portable.

If you're like most Tandy 100, 102 or 200 owners, you've probably found more uses than you ever imagined. But where do you keep all those great programs? Up until now, it's probably been on cassettes—a lot of cassettes. Now there's a better way.

Our battery-powered 200K Portable Disk Drive gives you fast access to 200,000 characters of data on a 3½" micro floppy diskette. With your Tandy portable's built-in software, you can create and update large documents on easy-to-store microdisks. Now you can build your own software and data library in a cost-effective manner. The microdiskettes are only $13.95 for a package of 3, and cost even less when you buy in quantity.

Come to your local Radio Shack and see the 200K Portable Disk Drive—only $199.95. (26-3814).

Send me a new 1988 computer Catalog.

Mail To: Radio Shack
Dept. 88-A-1126
300 One Tandy Center
Fort Worth, TX 76102

Name _____
Address _____
City _____
State _____
ZIP _____
Phone _____

Radio Shack®
The Technology Store™
A DIVISION OF TANDY CORPORATION

Prices apply at Radio Shack Computer Centers and participating stores and dealers. Batteries not included.

Circle 44 on Reader Service Card

screen. Type in each item at the "item" prompt and push the *Enter* key twice. The first time you index a page, type in the page number and push *Enter*. Push *F4* ("next"). For each successive item on that page, push *F2* ("dupe") and then *F4*.

After you've recorded all the items on the page, go on to the first item on the next page and push *Enter* twice. Remember to type in the new page number instead of pushing *F2*. For the remaining items, push *F2* to enter the same page number.

To avoid having to type in leading zeroes (001,023 and so on) so that the numbers are sorted as, for example, 1, 8, 33, 57 instead of 1, 33, 57, 8, use the *B* column for integer or whole number entries. Simply use the "display" function to eliminate the decimal point and two trailing zeroes (this can be done after you've entered all the items).

Next, push *F7* ("sel"), then the period ("."), and then *Ctrl-Z* to set the range. Push *Enter* and then *F1* ("DISP"). The bottom line will change and you can then select *F2* ("*.**"), delete the number using the backspace, and type in a zero ("0"). The decimal point and two zeroes following it will disappear, leaving you with integers (see pre-sorted example in *Fig. 1*).

SORRY, OUT-OF-ORDER

The early version of Super ROM could not sort on more than one key. Later versions can sort on two keys. If you have not yet returned your chip to PCSG for an update, you will need to make two passes. Since you do not have a secondary key sort, you must scan the *B* column for out-of-order numbers.

To sort page numbers belonging to the same entry, move the cursor to the entry's first page number, press the period key ("."), and use the down arrow to highlight the entire list of numbers for that entry. Press *Enter* and then *F7* ("sort"). The bottom line will change again. Press *F5* ("key") and make sure a number in the *B* column is entered. (*B1* is OK and can remain for the entire secondary sorting session because you will be sorting exclusively in column *B*.)

For subsequent secondary page number sorting, move down the column to the next item with multiple page numbers and repeat the process. You no longer need to change the key—just press *F7* ("sort"), then *F6* ("go"). When the secondary sorting is done, you are ready to paste your index into a .DO file for editing.

For future reference, print out the completed index. Set the *A* column at a width large enough to print out the longest entries, and set the *B* column to a width of three. This procedure will save you time later. When the entire index is printed out (in two columns, if you wish, by using the designated feature in the Lucid Data program), change column *A* back to nine spaces wide. You're now ready to edit.

Typographical errors, which are so difficult to spot and correct, are reduced because basic entries are typed in only once.

First, paste a copy of the index into a text file with the *A* column width set at nine and the *B* column at three. Letters at the end of longer words will drop off the screen, but the printed copy will show them, and they can be added later. It is easier to add a few letters than to backspace and delete many blank spaces in the text.

TEXT-CLEANING

You now have a .DO file in two columns (*Fig. 2*). Using Write and the *Rplc* function (*F1*), search and replace all carriage returns with a comma followed by a single space (*Fig. 3*). Repetitions of the key word must be removed, but this text-cleaning is easy.

To delete the word, type it in by using the *Rplc* function. Include enough spaces for a total of nine characters; this will delete any trailing spaces after the word. Press *F8* and then replace with a null string by pressing *F8* again after the prompt. When the prompt asks if you wish to confirm each replacement, type "y" and then "n" for the first entry (since that will be the indentifying word at the left of the column). Type "y" for all other entries until the filename returns (*Fig. 4*). Go through the text and put in carriage returns, delete commas after the last numbers in each entry, and clean up the text where needed.

For the final edit, I find it convenient to use the *Ctrl-D* and *Ctrl-F* combinations (moving the cursor one space or one word to the right). My left hand manages these keys and my right hand operates the *Del/Bksp* and *Enter* keys. I can go through the text (*Fig. 4*) in short order.

Ctrl-f places the cursor at the first character of the page number or next item. I can then backspace to erase the intervening blanks or to remove the spaces and comma after the last page number. A carriage return sends the next item to another line. In other words, when you're faced with a line like:

deafness 65, destiny 59, dream

use *Ctrl f* to skip the cursor over to the "d" in "destiny," backspace to erase the space and the comma, and then hit *Enter*. You can do it without lifting either hand off its respective keys (see *Fig. 5*, the edited index).

Only 250 items will fit in a file. If your index requires more, stop entering, mark the place in the text, sort the file, and divide it up into independent files (A-G, H-N, O-Z, for instance). Then, start a new file and begin entering where you left off. When you've typed in items from the second section of the text, sort your new file and cut and paste A-G, H-N, O-Z into the three sections you created earlier. Sort each of them again, and continue.

You should end up with a series of files, each completely sorted and divided into alphabetically ordered segments. When you've completed the editing, you can join the segments as text files. □

The Portable File Cabinet

The Tandy 600's File: A guide for the perplexed.

By Stephen Schenkel

Microsoft did a good job on the Tandy 600's ROM-based applications. The programs are comprehensive and are well-integrated. One of these, File, is Microsoft's answer to the need for a portable database. It's easy to use, but like many laptop software products, File has quirks, limitations, and inconsistencies. Some of them can be beaten, others can be worked around, and many must simply be accepted. Let's take a look at some of them.

SELECTION KEYS

Like the older Tandy laptops, the Tandy 600 uses what its manual calls "selection keys," used to cut, paste and copy information. In a word processor, these keys manipulate letters, words and sentences; in File, they mark records and fields.

A primer on database terminology: think of a database as a collection of Rolodex-style addresses. Fields are pieces of information, such as name and telephone numbers, that are filled in for every person in the Ro-

lodex. Records are a collection of fields specific to one individual, such as Jane Doe's name, address and phone number. A database is a collection of records, such as a list of all company employees. The Tandy 600 uses a spreadsheet analogy calling fields columns, and records rows.

By default, File works on one record at a time—the record that is always in reverse video on the screen. Thus, there is always at least one record selected, called the cursor record. To select additional records, to print perhaps, press *Shift-F8* to select a column, *Shift-F9* to select a row, or *Shift-F10* to select the entire database. A row may also be selected by pressing the left arrow with the cursor in the first column.

The selected area can be extended with F6. Pressing this key once turns the extension on; pressing it again removes the extension, and the previously selected area reverts to normal video.

CONTROL RECORDS

File's control records are contained in the four labeled rows at the top of the database. Their function is fairly well-explained in the manual, but their use can be expanded.

The first of these control rows, the ID row, contains data identifying information stored in each column. This is the one row of the database that is always on the screen, no matter where in the database the cursor is. This data is used by the Lookup and Jump commands to determine where to lookup or jump; therefore, the ID for each field should be descriptive and not contain duplications.

This ID row can also store notes after the formal label for each column. Each field can hold as many as 230 characters, but the value given in the FORM row determines the number of characters actually displayed on the screen. If, for example, the FORM row specifies 13 characters, the remaining 217 can be used for hidden notes.

The FORM row is more complex, determining the width of each column and the format of its information. There are column formats: day/date, number and text. The use of boilerplates determine column format. Note that each record can store as many as 230 characters, while 74 characters is File's maximum screen width.

Boilerplates are useful for setting up data entry as they allow various ways of entering data: quick data entry of repetitive data, consistent use of decimal points, dates and times formats, and currency formats. A good example of boilerplate use is a telephone number column. A boilerplate of *(AAA) AAA-AAAA* allows the number to be entered in various forms, with or without the hyphen and parentheses. The capital "A" indicates a text character.

When defining the FIND records, keep in mind the difference between absolute and non-exact references.

Boilerplates can also be used to set up printer control features. To underline fields when printing, set the first character in the appropriate FORM column to the underline-on command for your printer—ASCII 15 on my Tandy DMP-105—and the character to the underline-off code, in my case ASCII 14. The same principle applies for other special characters or print styles such as double-width, italic or bold.

One warning about boilerplates: the pound sign (#) indicates a digit, and should not be used in conjunction with text characters. Any text characters entered will be treated as additional boilerplates with no special meanings. Also, a FORM of *-####* does not treat a number entered as a negative; rather, File assumes the hyphen in a telephone number. Negative numbers can only be indicated by entering them when entering data.

The next control row, SORT, can be very useful. You should realize, though, that not all databases are meant for sorting. A database for storing research notes probably cannot be sensibly ordered, while an address database should be sorted by last name. Be careful: File sorts by using the ASCII character sequence. This means, for example, that a capital M (ASCII 77) is sorted before the lower-case *a* (ASCII 97). And to sort numbers properly, they must be defined in their column's FORM record, otherwise 121 (ASCII 49, 50, 49) will be considered less than 35 (ASCII 51, 53). Be patient with SORT; it's not the world's fastest, most notably with a database containing more than 50 rows.

The fifth, and final, control record is FIND, which allows the File program to search the database for rows containing specified information. When defining the FIND records, keep in mind the difference between absolute and non-exact references. Absolute references are contained in quotation marks and mean just that: FIND everything that's exactly like this example. Non-exact references, on the other hand, are not so demanding, particularly in that they don't notice any difference between upper and lower-case.

FIND is by no means limited to one item. Multiple search criteria can be entered in the same row, and File will come up with only those records matching all the criteria entered, as in all left-handed baseball players in California. This is an *and* search, looking for records that match all three criteria, *left-handed*, *baseball* and *California*. You can search using *or* criteria by producing additional FIND rows using the Find command. To find all those who play baseball, and who are either left-handed or in California, use two FIND rows: one that specifies *baseball* and *left-handed*, the other that says *baseball* and *California*. Producing the correct combination of FIND records can be quite a challenge.

Searches in text and numeric columns can be limited by using the greater than, less than and equal to signs. The comparisons with text are in relation to the ASCII sequence, so once again, M is less than *a*. Similarly, wildcard searches using the

question mark (find a single matching character) and the asterisk (find all matching characters) are useful.

As with SORT, be patient with the FIND control record. Searches take time in long databases.

COMMANDS

The commands, as displayed at the bottom of the File screen, provide several manipulative functions on all selected columns or rows. After pressing *Esc*, type either the first letter of the command name or use the *Tab* key to move the cursor to the appropriate name. Pressing *Enter* executes the command.

Many commands are straightforward. Copy inserts items into the system manager's scrap file without erasing the information on the screen. It is essentially the *cut* part of a cut-and-paste command.

Delete does what it says. Be careful: delete does not ask questions first.

Insert is the *paste* part of the cut-and-paste type command. It inserts items from the scrap into the selected regions and places extra data before the selected regions. Insert puts data only into those selected areas; if you had previously cut an entire row, but only wish to paste one column, simply select that one column before executing the Insert command. If you insert data into a record with information already contained in it, the old data is lost.

Move, simply enough, moves rows and columns. To insert a new column (such as zip code) information into an existing database, simply create the ID and FORM records for this column as the last column to the right in the database, and Move this new column to the appropriate spot.

Edit is the default, and most often used, command. While in edit mode, all the data in a given record, up to the maximum of 230 characters, is displayed at the bottom of the screen. Information in the ID, SORT, FORM and FIND rows are entered while in edit mode. Using the arrow keys moves the cursor from box to box; pressing *Enter* causes File to resort the database and leave the edit mode. In edit mode, the grey function keys *F7* through *F10* are used as cursor-control keys within records.

One of Edit's quirks is a tendency to add extra columns and rows at

the database's far right and bottom. Simply ignore them until you're finished editing, and then delete them as a group.

Find inserts an additional FIND record for various *or* criteria in searches. New FIND rows appear above those already existing, so look carefully to be sure you don't enter too many—a blank FIND row causes File to display *all* data.

The Options command in file affects the usage of FIND rows. Its effect is the same as a blank FIND row; when Options indicates that the FIND is turned off, all the data in the database appears on the screen.

Jump moves the cursor around the database. Since rows are referred to by number, though, Jump's use is limited by your knowledge of what each row contains.

Lookup is useful for building new databases containing information stored in old ones. One drawback is that while the Lookup is occurring, both databases must be resident in memory — a difficult feat on a 32K system.

The Print command lets you make a printed copy of selected data. Database information can be printed to the screen (CON:), the printer (PRN:), disk (A:*filename*) or resident memory (*filename*).

Print also has two forms of printing, text and interchange. Text causes the database to print in the form of a spreadsheet, that is, with columns split between pages. Interchange places quotation marks around records, commas between records, and new line markers (down arrows) between rows. If you print to disk or memory, you can edit this file with Word to your heart's content, using Replace to delete or change all the commas and quotation marks.

TIPS AND TRICKS

One of the tricks of File is to print only desired rows or columns. To print desired rows, first use FIND to search them out, then select them all using *Shift-F10*. Printing non-contiguous columns is slightly more difficult. First, make sure the database is saved on disk. Then, using the Move command, rearrange the columns so the ones you want are adjacent. Finally, select those columns and print.

File has no mailmerge capability. If

The sky's the limit...
...with Portable 100 magazine.

On the road . . . at home . . . in the office. . . in the air. . . the variety of tasks you can perform with your Tandy portable is expanding constantly.

To keep fully informed and up-to-date about current trends, new products, and new uses for your laptop, you need **Portable 100** magazine.

From sophisticated input/output (I/O) calls and their applications, to simple disk drives, **Portable 100** magazine covers it all.

Portable 100 gives you features, news, columns, and reviews that are thorough and timely. And they are written in a fast-paced, easy-to-read style, by leading experts in the computer field.

In upcoming issues you will discover. . .

- how new peripherals can make you more productive.
- more efficient ways to communicate with your desktop.
- where to buy low-cost public-domain software.
- how your business can be more profitable with your portable at your fingertips.
- where to find the best bulletin boards and information services.
- and much more!

Don't miss even a single valuable monthly issue. Fill out the coupon below, or to charge it to your credit card, call toll-free

1-800-258-5473
In N.H. call 1-924-9471.

Send no money now! We will gladly bill you.

☐ **YES!** I want to explore ways to be more productive with my Tandy portable. . . and save 47% off the newsstand price. Send me a year's subscription (12 issues) of **Portable 100** for $24.97.

☐ Payment enclosed ☐ Bill me
Make checks payable to **Portable 100**

Name _____

Address _____

City _____ State ____ Zip _____

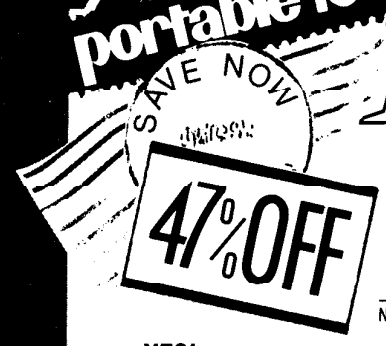
Canada $45.97 (Canadian funds), Mexico, $29.97. Foreign Surface $44.97. One year only. U.S. Funds drawn on U.S. banks. Foreign Airmail, please inquire. Please allow 6-8 weeks for delivery.

37NB4

MONEY BACK GUARANTEE:
If you are not completely satisfied with **Portable 100**, you may cancel your subscription and receive a full refund. Please allow 6-8 weeks for delivery of your first issue.

Mail to: **Portable 100,** CW Communications/Peterborough, P.O. Box 959, Farmingdale, N.Y. 11737

Save 47% on 12 Issues of Portable 100



PORTABLE 100 GOES EVERYWHERE

Delivered at a 47% Savings!

☐ **YES! Send me 12 issues of Portable 100 for the low subscription rate of $24.97. I'll save 47% off the newsstand price!**

☐ **Payment Enclosed**

☐ **Bill Me**

Please make checks payable to Portable 100

Name

Address

City

State/Zip

portable 100

Canada $45.97 (Canadian Funds), Mexican $29.97, Foreign Surface $44.97 (US Funds drawn on US Bank). Foreign Airmail please inquire. Please allow 6-8 weeks for delivery.

37NB4B

PORTABLE 100 GOES EVERYWHERE

Delivered at a 47% Savings!

☐ YES! Send me 12 issues of Portable 100 for the low subscription rate of $24.97. I'll save 47% off the newsstand price!

☐ Payment Enclosed

☐ Bill Me

Please make checks payable to Portable 100

Name

Address

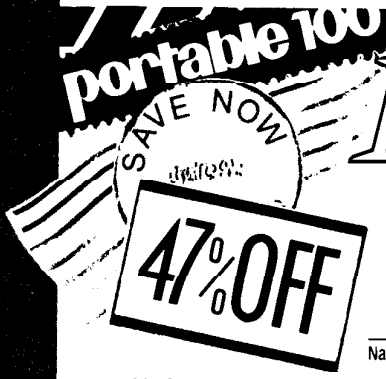
City

State/Zip

portable 100

Canada $45.97 (Canadian Funds), Mexican $29.97, Foreign Surface $44.97 (US Funds drawn on US Bank). Foreign Airmail please inquire. Please allow 6-8 weeks for delivery.

37NB4B



PORTABLE 100 GOES EVERYWHERE

Delivered at a 47% Savings!

☐ YES! Send me 12 issues of Portable 100 for the low subscription rate of $24.97. I'll save 47% off the newsstand price!

☐ Payment Enclosed

☐ Bill Me

Please make checks payable to Portable 100

Name

Address

City

State/Zip

portable 100

Canada $45.97 (Canadian Funds), Mexican $29.97, Foreign Surface $44.97 (US Funds drawn on US Bank). Foreign Airmail please inquire. Please allow 6-8 weeks for delivery.

37NB4B

you need it, you can mark the desired address in an address database, copy it into the scrap area and then insert it in the proper place in the letter. Repeat the process for each letter. It's tedious, but it works.

Databases inevitably get longer, but the Tandy 600 limits file length to 64K. When a database approaches the 64K limit, first SORT it by what you think is the most important item. Save the sorted database to disk. Now, delete the bottom half of the sorted database and save it under one name, such as BASE1; reload the full database, and save the bottom half under another name, such as BASE2.

Whenever you must split the databases, maintain the numbering order. Then, to load a bank of two or three databases—as many as three 64K databases can be resident in memory in a 224K Tandy 600—enter the Copy command in the system manager. For the filename, enter *A:BASE?.**. The *?* after the file name is a wildcard taking the place of the number; the * insures that both the .FIL and the .DAT files will be loaded.

If you're interested, you can examine File's data files directly. Enter the Word application and specify either the .DAT or the .FIL filename.

Lookup is useful for building new databases containing information stored in old ones.

They will both appear the same. The "special" characters littering the screen are File's form marks. Note that only the characters in each record appear without the trailing spaces, thereby saving memory.

To create databases from a Word document, divide the data into sections using tabs to indicate column divisions, and line ends (*Shift-Enter*) to indicate row ends. Once you have the information formatted, move it into the scrap area using Word's *copy* command. Set up the skeleton of the database by making a FORM record. Next, select the first row of the new database, and insert all the data from the scrap area.

When you make a mistake, File, and generally all Tandy 600 applications, will beep and move the cursor to where it thinks the error is located. If you can't locate the error at the cursor, look in the area immediately before it. Also, the *Esc* key can act as an "oops" key in edit. If you make a mistake when editing a single record, press *Esc* to exit the Edit mode. The record you were working on will return to its original data.

Experiment with File. The Tandy 600's portability and File's power make a fantastic combination. □

Quote Format for the Model 100

A Lucid concept that calculates everything automatically.

By G. N. Gilbert

Of all the spreadsheet programs I have on my Model 100, the quote format running under Lucid is my favorite. It utilizes a Lucid concept called tables, which is similar to lookup and choose but much more versatile.

We build custom-built products at my company. There is nothing on the shelf. Assume for a second that we build an electronic switching device that automatically controls other devices. We make two sizes: the small one controls a single device, the large one controls ten devices. The frequency used to control devices varies, and the part that generates the frequency is what is custom-designed for each buyer. Because of our unique design, only a few parts in the unit need be changed, but labor and materials costs vary with these changes.

The quote format now enters the discussion. You get the entire production cost by entering the type or model number, the variable, and quantities to quote. This includes overhead, selling price, margin (by percent, unit and for the total quantity), total sales for each quantity and the entire overhead absorption. Everything is calculated automatically.

In this article, we'll develop the spreadsheet first with labels only, and then the tables. As each table is completed, we will return to the body of the spreadsheet and develop the formulas needed to use that particular table.

THE BODY

First, all columns are nine characters wide. The majority will have two decimal places. The column designation is across the top and the row number down the left side.

Place the cursor at cell A1. As usual, we will type in the name of the file under which this will be saved, and the initials of both the designer and the prime user. In lower case, type "mstquo" for Master Quote, and the initials "gg/rr." Remember to save the original form as a master. When you load it back to the Model 100, change the name, input the data and save it under the new name.

You get the entire production cost by entering the type or model number, the variable, and quantities to quote.

In cell A3, we'll place the current date by pressing *Ctrl-D* while in Lucid's Edit mode. To track our quotations, we sequentially number each one we print. Move the cursor to A4 and type "QUOTE.NO." This number is entered manually and tracked in a logbook.

Our title must be centered. The width of our spreadsheet is eight columns, each nine characters, or a total of 72 characters wide. Half of that is 36. The title of our form is "QUOTATION FORM—WIDGETS", 24 characters long. Half of that is 12, and 12 subtracted from 36 is 24. This means the title begins in column C4, the sixth character over from the beginning. Type in the title at this point.

In A6, press the space bar four times and type "TYPE:". This will be for our model number. The remaining labels are described in *Fig. 1*. The lines used to separate the different areas in A12, A32, A50 and A53 should be replicated across to column H. Also, the dots (.) in front of some of the labels designate spaces and should not be typed.

THE TABLES

To describe a table, refer to the Lucid manual. It reads "TBL lets you have two input cells. It will look up those inputs on a chart. It finds one input on the X-axis or top row part of the chart, and the other input on the Y-axis or left column part of the chart. Where the two points cross on the table is the result that is returned." You can use either text or values for input and Lucid will return either text or values.

The first example is a rather simple

```
A7  COMPANY:                A32 '*********
B7  PREP BY:                A34 SALES
A8  PART NO.                A35 .....(W/O COMMISSION)
B8  PROJ NO.                A36 SALES
A9  VARIABLE:               A37 .....(W/ COMMISSION)
A11 ^QTY>                   A38 VAR. MAR.
A12 '=========              A39 .....(%)
A13 MAT COSTS:              A40 TOT. VAR.
A14 ...MAT                  A41 .....(%)
A15 ...SCRAP                A42 UNIT TOT.
A17 ..TOTAL                 A43 .....(MAR.)
A19 .....LABOR HOURS>>      A44 TOT MAR.
A21 LABOR $                 A46 TOT SALES
A23 M/L COSTS               A47 .....(W/O COMMISSION)
A25 OVERHEAD                A48 TOT SALES
A26 ...VAR                  A49 .....(W/ COMMISSION)
A27 ...FIXED                A50 '=========
A29 TOTAL O/H               A51 O/H
A31 TOTAL CST               A52 .....(ABSORPTION)
                            A53 '=========
```

Figure 1.

	D	E	F	G	H
61		MINI	SMALL	MEDIUM	LARGE
62	PART 1	+(B9+2)*.037	+(B9+2)*.040	+(B9+3)*.045	+(B9+4)*.045
63	PART 2	0.78	0.80	1.20	1.56
64	PART 3	0.10	0.10	0.15	0.20
65	PART 4	0.06	0.06	0.09	0.12
66	PART 5	0.29	0.30	0.45	0.58
67	PART 6	+(B9+1)*.28	+(B9+2)*.30	+(B9+4)*.32	+(B9+4)*.35
68	PART 7	0.44	0.45	0.68	0.88
69	PART 8	0.16	0.16	0.25	0.32
70	PART 9	0.17	0.17	0.26	0.34
71	PART 10	+(B9*.0265)+.11	+(B9*.0265)+.11	+(B9*.0372)+.11	+(B9*.0425)+.11
72	PART 11	0.10	0.10	0.28	0.36
73	TOTAL	+SUM(E62:E72)	+SUM(F62:F72)	+SUM(G62:G72)	+SUM(H62:H72)

Figure 2.

	A	B	C
86	0.00		0.00
87		0.00	0.00
88		1.00	+E73
89		2.00	+F73
90		3.00	+G73
91		4.00	+H73
92			

Figure 3.

table. Move the cursor to D87 and type "0". Go to F87 and type another "0". In column E88, type a "1"; in E89, a "2"; in E90, "3"; and E91, type "4".

In F88, "MINI"; F89, "SMALL"; F90, "MEDIUM"; F91, "LARGE".

Return to B6 and enter the number "3". Go to C6 and type in the formula " + TBL(B6,D87,E87:F91)" and press *Enter*. The word "MEDIUM" should appear. In this formula, we are saying "look at what is in B6 and D87, go over to the table located between E87 and F91, and return whatever is in the cell where they cross."

TBL requires two numbers or items to look up. In this case, we only needed one, so we fooled it by looking at a number in B6 and a dummy number in D87. Cute, but why not just type in the size, and forget the table bit? Later, we will use the model or type number to help us calculate our labor and material costs.

Let's continue with the pricing of each of the 11 parts in our four products. Move the cursor to E61 and type in the data and formula from *Fig. 2*.

The formulas for Part 1, Part 6 and Part 10 refer to the variable in B9. This is not a true table, but rather an area set aside for calculations.

Now, place the cursor in A86 and type in the data in *Fig. 3*. This table establishes the 1,000-unit price for each of four products. Based on the number typed in at B6, the appropriate cost will be returned. This formula appears in the next table. Move the cursor to C74 and type in the *Fig. 4* information.

The formula in E76 looks for the product type in B6 in the table located between B87 and C92. When it finds that particular type, it returns the costs to E76. Experience tells us that we can expect only six price breaks from vendors between the quantities of 1 and 100,000. We also know what these discounts will average out to be and are used in the formulas in E77 through E81. Remember, this is merely for a quotation for some time in the future. All we are doing is providing our customer with a very good estimate based on current conditions.

Now, go to B14 and type in the formula:

+TBL(B11,$C74,$D74:$E81)

Replicate this across to column H. The "$" tells Lucid that the cell following should not change when we replicate it across the rows. We want B11 to change to C11 in column C, but we don't want the cells that reference the table to change. Type a number in B6, input quantities along row 11 and press *Calc (F2)*.

Change the number in B6, press *Calc* again, and notice that the materials costs change. You can use any quantities you wish; Lucid will return only six different material costs for the quantities 0 to 100,000 based upon our table. If you type in 250, you will get the

	C	D	E
74	0.00		0.00
75		0	0.00
76		1000	+TBL(B6,A87,B87:C92)
77		5000	+E76-(E76*.025)
78		10000	+E76-(E76*.0425)
79		15000	+E76-(E76*.0475)
80		25000	+E76-(E76*.1075)
81		100000	+E76-(E76*.1225)

Figure 4.

	A	B
73	0.00	
74		0.00
75	1	+(B9*.05)+.20
76	2	+(B9*.05)+.35
77	3	+(B9*.05)+.50
78	4	+(B9*.05)+1.00

Figure 5.

1,000-piece price. If you use 1,001, Lucid will return the 5,000-piece price. The table should be structured to handle this.

Move the cursor to B15 and type the formula "+B14*.05". We plan for a five-percent scrap factor when costing products. In B17, total the two material costs with "+B14+B15". Replicate both formulas to column H.

Figure 5 calculates the labor hours by multiplying the variable by a constant assembly time. It then adds the standard hours associated with that particular product. *Figure 6* is a table to suppress zeros. Because the labor hours are associated with a product and not to volume, Lucid would print the labor hours even if a zero appeared in the quantity field. The other formulas would then try to use the labor hours in calculations that would be meaningless, and they would only clutter the spreadsheet. Go ahead and type in the data from *Figs. 5* and *6*.

In B83, for example, press the quotation key ("), the space bar, then *Enter*. The TBL formula will return a blank if the number is a zero. If that value is greater than 0, it will return whatever the value is. In cell A84, the "+1050" just means the number 10 to the 50th power. If you know what your maximum number will be, you can use it, but "+1050" is simple enough to use. The formulas in B84 calculate the labor costs by multiplying the labor hours from *Fig. 5* times our average rate of $6.50.

Next, move the cursor to B19 and type in the formula "+TBL(B6,A73, A74:B78)". This will return the labor hours. In B21 input "+TBL(B11, $A81,$A82:$H84)". This formula looks at the quantity in row 11, then goes to the zero suppression table and returns either a value if the quantity is greater than zero, or a blank if it is zero. Replicate this formula across to column H.

	A	B	C	D	E	F	G	H
81	0.00	+B11	+C11	+D11	+E11	+F11	+G11	+H11
82								
83	0.00	"(SPACE)	"(SPACE)	"(SPACE)	"(SPACE)	"(SPACE)	"(SPACE)	"(SPACE)
84	+10^50	+$C19*6.5	+$C19*6.5	+$C19*6.5	+$C19*6.5	+$C19*6.5	+$C19*6.5	+$C19*6.5

Figure 6.

The next order of business is to calculate the overhead, both variable and fixed. Again, these calculations are not truly a table, but an area off the printed part of our page. Refer to *Fig. 7* for the formulas for column B. Once you have typed in column B, replicate each across to column H.

We figure 13 percent of our labor dollars covers the vacation, overtime and holidays. Also, another 13 percent is for labor that has been diverted to other than direct labor jobs. Fringes cover insurance and pension contributions, and we use 26 percent of the vacation and overtime plus the diverted labor plus the labor costs in B21.

Our in-bound freight averages six percent of material costs, and our operating supplies about 11 percent. The fixed portion of the overhead is calculated at 56 percent of labor costs. Return the cursor to B26 and type "+B59" to bring up the sum of the variable overhead, and in B27 type "+B60" for the fixed portion. We could calculate the fixed overhead in B27, but I want to keep the calculation areas together. We'll total our overhead with the formula "+B26+B27" in B29. In B31, let's get our total cost—"+B23+B29". Naturally, each of the above formulas should be replicated to column H.

Before we get to the rest of the form, we will need to develop a table to calculate our margins. Refer to *Fig. 8*. We give a discount for volume—the higher the volume, the lower the price. Once

	A	B
54	VAC/OT	+B21*.13
55	DIVERTED	+B21*.13
56	FRINGE	+(B54+B55+B21)*.26
57	FREIGHT	+B17*.06
58	OP SUPP	+B17*.11
59	TOT VAR	+SUM(B54:B58)
60	FIXED	+B21*.56

Figure 7.

	A	B	C
61	0.00		0.00
62		100	0.51
63		500	0.50
64		1000	0.49
65		5000	0.37
66		10000	0.20
67		15000	0.15
68		25000	0.10
69		50000	0.07
70		75000	0.05
71		100000	0.03

Figure 8.

```
      A              B                    C
93 VAR MAR                    +B11                 +C11
94     0.00                   *(SPACE)             *(SPACE)
95    +10^50 +(B34-(B23+B26))/B34 +(C34-(C23+C26))/C34
96 TOT MAR                    +B11                 +C11
97     0.00                   *(SPACE)             *(SPACE)
98    +10^50       +(B34-B31)/B34       +(C34-C31)/C34
```

Figure 9.

you have entered the Margin table, go to B34 and enter the following formula:
+B31/(1-TBL(B11,$A61,$B61:$C71)

We are dividing the total cost by the reciprocal of our margin percent. In B36, type "+B34*1.05". We have sales representatives in certain parts of the country, and if they make a sale, we pay them a commission of five percent of the sales price. This is reflected in the sales price we give to our reps. These formulas should be replicated across to column H.

Figure 9 contains zero suppression tables for both the variable margin and the total margin. With the exception of the formulas, these are identical to the labor example above. We subtract our direct costs (material, labor and variable margin) from our sales price to give us a percentage. The total margin is the sales price minus the total cost, divided by the sales price for a percent. Be sure to copy these formulas to column H.

Place the cursor in B38, and type the following formula:
+TBL(B11,$A81,$A93,$A93:$H95)

Go to B40 and input the following:
+TBL(B11,$A81,$A96:$H98)

Note we used A81 in both formulas. We can do that since the number is a dummy, and in this case, a zero.

The remaining formulas need no explanation:

B42+B34-B31
B44+B11*B42
B46+B11*B34
B48+B11*B36
B51+B11*B29

In B42, we are finding the difference between our selling price and our cost. In the remaining formulas, we are multiplying the quantity by each margin, sales (with and without commission) and overhead. Make sure you replicate these formulas to column H.

FINISHING UP

Now all you need to do is to type in the data required in the first 11 lines and press *Calc*. A quotation will take less than five minutes. Our format provides all the necessary information to make a decision. If you feel you must cut your price to get your foot in the door, you have the information to make that decision. I think this is a useful tool, and it uses very little memory.

Lucid will print all eight (columns) three at a time.

I suggest you print the entire spreadsheet to get an idea of just how we put it together. Also, note how the tables are jammed in the lower part of the spreadsheet. We made effective use of the available space. (By the way, the printed portion is cells A1 to H53.)

Any time you develop a spreadsheet, I suggest you print a copy of the completed format and a copy of the spreadsheet's formulas. To do the latter, widen the columns to fit the largest formula plus one, then press the *Print* key. Press *List*, which will print the formulas and Lucid will adjust to the new size. In our spreadsheet, we used eight columns; our largest formula is 20 spaces plus one, or 21.

On an 80-column printer, with margins, you can only print three columns. Lucid will print all eight, three at a time. Retain all copies in an appropriately labeled folder. This will be your backup should something go awry later.□

Reading Fine Print

Until now, the best you could expect of a dot matrix printer were characters almost—but not quite—as good as the characters of a low-priced, manual typewriter. And that was after four, slow passes! Purge yourself of these misconceptions. In the double strike mode, Epson's LQ-800 dot-matrix printer performs sublime feats. Its text is almost offensively formal. Welcome features are a 7K buffer, traction feed, character definition capability and more than 100 typestyles, of which about ten are useful. Surprisingly, some routine items, such as tractor feed and graphics capability, are optional.

The printer is compatible with the Model 100's built-in TEXT word processing program. Commercial software is superfluous except to permit pagination and the use of text embedded commands.

When I used the LQ-800 with the Model 100, I changed DIP switch 2-8 on the printer's back panel to the "on" position. This advances the carriage at the end of each line. If user-defined characters are not needed, DIP switch 1-4 should also be turned on to enable the 7K buffer. Keep all other switches at the factory setting.

My program, PRINTR.BA, is used to configure the printer to the desired typestyle and page format. Although by no means exhaustive, it offers a wide range of options, which you can see in the program listing that follows the Input statements. By consulting the printer's manual, additional options may be appended. Pressing "-" followed by the *Enter* key at the prompt allows you to make selections in the program. Omitting "-" rejects options. In several cases—pitch and form length, for example—numbers must be entered in response to prompts. Essential commands have default settings indicated by "^" which will be sent to the printer if no selection is made.

Your left margin default is set at one-inch when the paper's left perforation lies exactly three-sixteenths

Figure 1. PRINTR.BA formatting program for Epson LQ-800 printer.

```
Ø CLEAR 1Ø4Ø:MAXFILES=2:ON ERROR GOTO
  46:X$=CHR$(27)
1 BEEP:PRINT @133,"Turn printer on":LPRINT X$
  "@";:CLS:CALL 17ØØ1:FOR Z=1 TO 35:READ X:PRINT
  CHR$(X);:NEXT Z:CALL 17ØØ6:PRINT:INPUT
  "LQ=1^  DQ=2  ":P:IF P=2 THEN LPRINT X$ "x"
  CHR$(128);
2 INPUT "R.Margin for LLIST (-) ";F$:INPUT
  "Double Strike (-) ";A$:IF A$="-" THEN LPRINT
  X$ "G";
3 INPUT "Pitch  1Ø=1 12=2^ 15=3 17=4 2Ø=5  ";A1
  :IF A1=Ø THEN A1=2
4 ON A1 GOTO 5,6,7,8,9
5 B=5:C=7Ø:A=8Ø:N=Ø.927:GOTO 1Ø
6 B=5:C=84:A=77:N=Ø.927:GOTO 1Ø
7 B=8:C=1Ø5:A=1Ø3:N=1.1:GOTO 1Ø
8 B=8:C=12Ø:A=8Ø:M=1:N=1.1:GOTO 1Ø
9 B=1Ø:C=139:A=77:M=1:N=1.1
1Ø LPRINT X$ CHR$(A);:IF M=1 THEN LPRINT X$
   CHR$(15);
11 INPUT "LQ Proportional (-) ";B$:IF B$="-" THEN
   LPRINT X$ "p1";:N=1.2Ø6
12 INPUT "L.Margin in Char.(1 in.^) ";D:IF D<>Ø
   THEN B=D
13 LPRINT X$ "l" CHR$(B);:IF F$="-" THEN LPRINT
   X$ "Q" CHR$(C);
14 INPUT "Italic (-) ";C$:IF C$="-" THEN LPRINT
   X$ "4";
15 INPUT "Emphasized (-) ";D$:IF D$="-" THEN
   LPRINT X$ "E";
16 INPUT "Subscript (-) ";E$:IF E$="-" THEN
   LPRINT X$ "S" CHR$(Ø);:IF B$="-" THEN N=1.725
   ELSE N=Ø.927
17 INPUT "Lines/in. 4=1 6=2^ 8=3 1Ø=4  ";E:IF E=Ø
   THEN E=2
18 INPUT "Form length  (66^) ";F:IF F=Ø THEN F=66
19 INPUT "Perf. Skip  (14^) ";G:IF G=Ø THEN G=14
2Ø ON E GOTO 21,22,23,24
21 E=45:GOTO 25
22 E=3Ø:GOTO 25
23 LPRINT X$ "Ø";:GOTO 26
24 E=18
 25 LPRINT X$ "3" CHR$(E);
26 LPRINT X$ "C" CHR$(F);:LPRINT X$ "N"
   CHR$(G);:INPUT "L-R print (-)";G$:IF G$="-"
   THEN LPRINT X$ "U" CHR$(1);
27 IF A1>2 AND (B$="-" OR P<>2 OR E$="-") THEN 34
   ELSE INPUT "R.M.J. (-) ";H$:IF H$<>"-" THEN 34
```

```
28 CLS:CALL 17ØØ1:PRINT STRING$(4Ø,32):PRINT @12,
   "Epson RMJ Print":PRINT @292," (F8) aborts "
   ;:CALL 17ØØ6:PRINT @8Ø,:LPRINT X$ "a" CHR$(3);
   :INPUT "Filename ";J$:IF J$="Menu" THEN 33
29 INPUT "R.Margin ";M$:IF M$="Menu" THEN 33 ELSE
   F=VAL(M$):LPRINT X$ "Q" CHR$(F);:OPEN J$ FOR
   INPUT AS 1:OPEN "LPT:" FOR OUTPUT AS 2
3Ø J=Ø:GOSUB 35:IF J=1 THEN 31 ELSE IF K1$=""
   THEN PRINT #2,K$:K=1:GOTO 32 ELSE PRINT
   #2,K1$;:PRINT #2,K$:K1$="":K=1:GOTO 32
31 IF K1$="" THEN K1$=K$:K=Ø:GOTO 32 ELSE PRINT
   #2,K1$;:K1$=K$:K=Ø
32 IF EOF(1) THEN CLOSE:GOTO 28 ELSE 3Ø
33 CLS:PRINT @173,"Reset Printer":LPRINT X$ "@";
34 MAXFILES=Ø:BEEP:MENU
35 LINE INPUT #1,K$:L=Ø:IF LEN(K$)=255 THEN J=1
   ELSE IF LEN(K$)=Ø THEN 45
36 IF K=Ø OR ASC(K$)>32 THEN 45
37 IF ASC(K$)>32 THEN 39 ELSE IF ASC(K$)=9 THEN
   L=L+5 ELSE L=L+1
38 K$=RIGHT$(K$,LEN(K$)-1):GOTO 37
39 IF B+L=9 THEN L=L+1
4Ø D=(F-B-L)*N:H=Ø:I=Ø:IF LEN(K$)<=D THEN
   L$=K$:K$="":GOTO 44
41 IF MID$(K$,D+H,1)=" " THEN I=1:GOTO 42 ELSE IF
   MID$(K$,D-H,1)=" " THEN 42 ELSE H=H+1:GOTO 41
42 IF I=1 THEN D=D+H ELSE D=D-H
43 L$=LEFT$(K$,D-1):K$=RIGHT$(K$,LEN(K$)-D+1)
44 LPRINT X$ "1" CHR$(B+L);:LPRINT L$:K=Ø:LPRINT
   X$ "1" CHR$(B);:IF K$="" THEN K=1:GOTO 35
45 RETURN
46 IF ERR=54 THEN 28 ELSE CLS:BEEP:PRINT "Error "
   ;ERR;"Line ";ERL:END
47 DATA 32,8Ø,82,73,78,84,82,46,66,65,9,32,32,32,
   32,32,32,82,46,32,75,111,1Ø7,1Ø5,99,1Ø4,32,32,
   53,45,49,45,56,54,32
```

of an inch to the right of the printer plate's left edge. This corresponds to the left-most position of the optional tractor feed. To change the left margin default, alter the values of variable *B* in lines five through nine; but note: *B* may not be equal to nine.

Right margin justification (RMJ) is one of my program's options. If you don't select this option, the program terminates by returning to the main menu. If you enter the desired document file and use the Shift-Print command, the text will be printed. The printer retains the specified configuration until changed or until its power switch is turned off. If you select RMJ, my program requests the name of the document file to be printed and line length in one-tenth-inch units. It then prints the text.

When finished, you're offered the choice of a new filename. Pressing the *F8* key terminates the program, resets the printer to the power-on default configuration and returns to the main menu. Select the unidirectional "L-R print" option for optimal results when in double strike mode and RMJ.

RMJ printing of a text with indented paragraphs can be done only through the PRINTR program because the printer's software won't justify the right margin on a paragraph's first line, if that line is indented. Lines 28-45 will correct the problem.

RMJ is compatible with all letter (LQ) and draft (DQ) quality type styles, except condensed LQ pica and elite. If you select these, the program does not offer RMJ as an option.

—Rudy E. Kokich, MD

The Last Lister?

Oh no, not another BASIC listing program! Yes, but this should be the last one because it does just about everything.

This program reads directly from the .BA files, so it's not necessary to save programs to an otherwise useless text file. (It also means it won't run on anything but a Model 100 without some different PEEKs.)

A special feature of the program is the way it handles for-next loops and conditional statements. The statements between for and next are indented and can be nested to show all of the loops active at any point. IF-THEN-ELSE lines have all the statements that refer to a particular THEN or ELSE listed under one another and each ELSE listed under its corresponding THEN. Line 21 is an excellent example.

The program's best feature is that it adds spaces where they are needed in lines such as "IFOOORATHEN-NEXTELSETC = 0." These might be fine in a program, to save RAM, but not in a listing.

Using the program is easy. To change the output device, load the lister and edit line 100. This is a DATA statement; its first line gives the total lines per page. To output to a pageless device, such as the screen or a RAM file, use zero here. The second number is the characters per line. The third entry is the file name. Set these parameters for the output device. For example, to output to the screen use:

100 DATA 0,39,"LCD:"

The program must be saved as a BASIC (.BA) file. When run, it will clear the screen, display the names of all visible BASIC programs in memory, and prompt "Program to list." Type in the name of the program to be listed. This can be in upper case letters, lower case, or mixed, with or without an extension. If the program is in memory, it will be listed. If not, you will get a "Try again" prompt. Invisible program names are accepted but not displayed.

```
1     CLEAR 6ØØ
      :DEFINT A-T
      :DEFSTR U-Z
      :P=Ø
      :A=Ø
      :DIM T(127)
      :READ L,C,V
      :CLS
      :IF V="LPT:" AND(INP(187) AND 6)<>2
      THEN PRINT "Printer not ready!"
          :RUN
      ELSE 6Ø
1Ø    Y=Y+CHR$(PEEK(G) AND 127)
      :G=G+1
      :IF PEEK(G)<128
      THEN 1Ø
      ELSE RETURN
11    X=X+CHR$(P)
12    A=A+1
      :P=PEEK(A)
      :IF P=Ø OR P=G
      THEN RETURN
      ELSE 11
13    IF O
      THEN I=INSTR(O,X," ")
          :IF I
          THEN X=LEFT$(X,I-1)+MID$(X,I+1)
              :GOTO 13
          ELSE I-INSTR(O,X,"AS")
              :X=LEFT$(X,I-1)+" AS "+MID$(X,I+2)
14    IF D AND MID$(X,D+1,1)<>" "
      THEN X=LEFT$(X,D)+" "+MID$(X,D+1)
15    U=SPACE$(T)
      :MID$(U,1)=Z
      :PRINT #1,U LEFT$(X,C-T)
      :LC=LC+1
      :IF T+LEN(X)>C
      THEN X=MID$(X,C+1-T)
          :Z=SPACE$(5)+"~"
          :GOTO 15
      ELSE Z=""
16    T=M+F
      :X=""
      :S=1
      :D=Ø
      :N=Ø
      :O=Ø
      :IF L AND LC+6>L
      THEN PRINT #1, STRING$(L-LC,13)
      ELSE RETURN
```

```
17    PC=PC+1
      :PRINT #1," Listing of "W"  " DATE$ "  "ZT;
      :IF L
      THEN PRINT #1,"   PAGE "PC;
18    PRINT #1,
      :PRINT #1,
      :LC=3
      :RETURN
2Ø    Y=""
      :P=PEEK(A)
      :IF P>127
      THEN 4Ø
      ELSE IF P
          THEN 3Ø
          ELSE GOSUB 13
21    FF PEEK(A+2)=Ø
      THEN PRINT #1,
          :PRINT #1," Length"A-AØ"bytes."
          :IF L
          THEN PRINT #1, STRING$(L-LC-2,13)
              :END
          ELSE END
      ELSE A=AN
22    AN=PEEK(A)+PEEK(A+1)*256-2^16
      :M=6
      :T=M+F
      :S=1
      :Z=MID$(STR$(PEEK(A+2)+PEEK(A+3)*256),2)
      :A=A+4
      :GOTO 2Ø
3Ø    Y=CHR$(P)
      :IF Y=":"
      THEN GOSUB 13
          :IF PEEK(A+1)=142 AND PEEK(A+2)=255
          THEN A=A+2
              :GOTO 2Ø
          ELSE IF PEEK(A+1)=145
              THEN A=A+1
                  :GOTO 2Ø
31    IF Y="," AND N=1 AND F
      THEN F=F 2
          :T=T-2
32    IF P=34
      THEN G=P
          :GOSUB 11
33    IF INSTR(" :(",Y)
      THEN S=1
      ELSE S=Ø
34    X=X+Y
      :A=A+1
      :GOTO 2Ø
4Ø    Y=""
      :G=T(P-128)
      :GOSUB 1Ø
41    IF LEN(Y)>1 AND S=Ø
      THEN X=X+" "
42    IF Y="REM" OR Y="'"
      THEN X=X+Y
          :G=Ø
          :GOSUB 12
          :GOTO 2Ø
```

GO WITH THE FLOW

Program flow begins at line 1, then jumps to line 60 for the input routine. This is to save execution time by placing the lines most often called near the start of the program. The subroutines are located in line 10 through 18 for the same reason. Line 20 examines each byte of the program being listed and dispatches tokens to line 40 and characters to line 30. Line 21 handles the end of the program; line 22 is for new line numbers.

A line-by-line analysis shows the following:

Line 1 sets aside 600 bytes for string variables. It defines "A" through "T" as integers, and "U" through "Z" as string variables. "P" and "A" are the most frequently used variables and are initialized here so that they will be on top of the variable stack. Again, this is for execution speed. "DIM T(127)" reserves space to store in ROM the location of the keywords used in BASIC.

The data "L," "C" and "V" are the page length, width and output file name located in line 100, where they can be easily changed. They need not be input every time the program is run. If the output file is the line printer (LPT:), then its status is checked. If it's not ready, then the message is "Printer not ready!" When corrected, program execution continues automatically.

Line 10 is a subroutine that accepts a Token as "G" and returns the corresponding keyword. They are stored in ROM in this manner: the first character of each keyword has its most significant digit bit (128) set, so when a character after the first one is greater than 127, the keyword is complete.

Lines 11 and 12 are the subroutine for handling quotations and "REM" statements, which are not modified by the program. If P = 0 then the end of the line has been reached. In the case of quotations, G = 34, so if P = G then it represents the closing quote mark. In either case, control is returned to the main program.

Lines 13 through 18 are the printing subroutine, which is called at the end of each statement.

Line 13 deals with "OPEN" statements, which actually don't fit the usual structure very well at all. The variable "0" points to the space after

```
43    IF Y="THEN"
      THEN GOSUB 13
           :M=M+4
           :E=Ø
44    IF Y="ELSE"
      THEN T=T-4
           :IF E
           THEN M=M-4
                :T=T-4
           ELSE E=1
45    IF Y="OPEN"
      THEN O=1
46    IF Y="FOR"
      THEN IF O
           THEN O=LEN(X)+5
           ELSE F=F+2
47    IF Y="NEXT"
      THEN N=1
           :IF F
           THEN F=F-2
                :T=T-2
48    IF Y="DEF"
      THEN D=LEN(X)+6
49    IF LEN(Y)=1 OR Y="MAX" OR Y="DEF" OR Y="TAB("
      THEN S=1
      ELSE S=Ø
5Ø    X=X+Y
      :A=A+1
      :P=PEEK(A)
      :IF S OR P-Ø OR INSTR(" :($", CHR$(P)) OR P>127
      THEN 2Ø
      ELSE X=X+" "
           :GOTO 2Ø
6Ø    FOR I=-1617 TO-14ØØ STEP 11
        :IF PEEK(I)=128
        THEN FOR J=I+3 TO I+8
                :PRINT CHR$(PEEK(J));
             :NEXT
             :PRINT SPACE$(4);
61    NEXT
      :OPEN V FOR OUTPUT AS 1
      :PRINT
      :PRINT "Program to list";
      :I=127
      :FOR J=Ø TO I
        :FOR I=I+1 TO I+9
          :IF PEEK(I)<128
        THEN NEXT I
        ELSE T(J)=I
          :NEXT J
62    INPUT W
      :IF LEN(W)=Ø
      THEN 62
      ELSE J=INSTR(W,".")
           :IF J
           THEN W=LEFT$(W,J-1)
63    FOR I=1 TO LEN(W)
        :J=ASC(MID$(W,I))
        :IF J>96
        THEN MID$(W,I)=CHR$(J-32)
```

```
64    NEXT
      :W=LEFT$(W+SPACE$(5),6)
      :ZT=TIME$
      :FOR I=-1617 TO-14ØØ STEP 11
        :IF(PEEK(I) AND NOT 8)=128
        THEN FOR J=1 TO 6
              :IF PEEK(I+2+J)=ASC(MID$(W,J,1))
            THEN NEXT J
                 :A=PEEK(I+1)+PEEK(I+2)*256-2^16
                 :AØ=A
                 :GOSUB 17
                 :GOTO 22
65    NEXT I
      :PRINT "Try again";
      :GOTO 62
1ØØ   DATA Ø,8Ø,"CAS:LIST"
      :REM  Page length, width and output file.
```

"FOR." First, all spaces after this are removed, such as the space before the keyword "END" in "APPEND" or after "OUT" in "OUTPUT". Then spaces are added on each end of "AS," which is not a keyword, to make it read right.

Line 14 is used in "DEF" statements to add a space after "SNG," "DBL" or "STR", which also are not keywords.

Line 15 sets up the margin string "U" and inserts the line number "Z," if any. It prints as much of the statement as will fit on one line and increments the line count "LC". If the whole statement won't fit on one line, then the tilde (~) is used to indicate a line overflow. The next part of the statement is printed using the same margin.

Line 15 resets the margin, clears "X" (the statement string), sets "S" so that the next line won't start with a space, and zeroes the "DEF," "NEXT" and "OPEN" flags. Then, if the end of the page has not been reached, control is returned to the main program. If the end of the page has been reached, then a form feed is generated.

Lines 17 and 18 print the page header. Note that if "L," the page length, is zero, no page number is printed.

Line 20 is the start of the main loop of the program and the dispatcher. "A" is the address of the byte of the program being considered, and "P" is its contents. If the byte represents a BASIC keyword, then "P" will have a value of 128 to 255, depending on which keyword it represents. The program will branch to line 40. Otherwise, if "P" is not zero, then it is the ASC value of a character, and the program branches to line 30. If "P" is zero, then it is an end of line flag, and the last statement is printed (GOSUB 13).

Line 21 checks for the end of the program. If it is found, it prints the amount of memory the program occupies at the end of the listing, generates a form feed and ends. If it isn't the end, then "A" is set equal to "AN," the next line's start address.

Line 22 deals with a new line number, stored in RAM in this manner: after the end-of-line flag (zero), the next two bytes point to the start of the next line, and the two bytes following represent the current line number. Both of these numbers are coded in a way that is easy for the machine to understand, in base 256 with the least significant digit first. So, the second byte times 256 plus the first byte is the number. "AN," the next line address, is PEEK(A) + PEEK(A + 1)*256.

Subracting two to the sixteenth power (64K) brings the value back into the integer range without chang-

ing it (as far as the machine is concerned). Then "M" is reset to 6 because all "IF-THEN-ELSE" conditions end with a new line, and "T" is the total margin is set to "M" plus "F," the "FOR" margin. "S" is set to prevent leading spaces, "Z" becomes the new line number, and "A" is advanced to the next program bit.

Line 30 is the start of the character handling routine. First, "Y" is set equal to the character represented by "P." The remainder of the line deals with the cases where "Y" is a colon. In every case, it indicates the end of a statement, so the present statement is printed. When an apostrophe is used for a remark, it is stored in RAM as a colon, the token for "REM" (142) and the token for an apostrophe (255), but listed as '. So "A" is advanced two. Next, "ELSE" has an implied colon before it in RAM, as it is the start of a new statement. So, "A" is advanced one.

In line 31, N = 1 indicates a "NEXT" statement. Each comma in such a statement indicates the end of another for-next loop, so if the "FOR" margin "F" is non-zero, then it is reduced by two.

Line 32 directs quotations to subroutine 11 with a return on a closed quote or end-of-line flag.

Line 33 specifies that a space will not be inserted after a space, a colon or an opening parenthesis.

Line 34 appends the character to the statement string "X," increments the address counter "A," and returns to the dispatcher.

Line 40 is the start of the token-handling routine and calls for the keyword represented by the token.

Line 41 inserts a space before keywords with a length greater than one unless it has already been specified otherwise.

Line 42 directs remark statements to subroutine 12, where they will be listed without modification until the end-of-line flag is reached.

Changing "T" changes the margin of the present statement, and changing "M" changes the margin of following statements.

Lines 43 and 44 direct the statement before "THEN" to be listed and manage the margin in conditional lines. Changing "T" changes the margin of the present statement, and changing "M" changes the margin of following statements.

Line 45 sets the "OPEN" flag "O," which changes the meaning of "FOR" and activates line 13.

Line 46 recognizes "FOR" and, if it occurs in an "OPEN" statement, changes the open flag to a pointer, or else increments the "FOR" margin "F" by two.

Line 47 detects the end of "FOR-NEXT" loops, decreases "F," the "FOR" margin, if it is greater than zero, and sets "N," the "NEXT" flag, in the event a multiple next occurs in the statement.

Line 48 sets the "DEF" pointer "D" for line 14.

Line 49 specifies that no space will be inserted after the single character keywords or "MAX," "DEF" or "TAB(".

Line 50 adds the keyword to the statement string, then looks at the next byte. Unless no space after has been specified by line 49 or the next byte is the end-of-line flag, space, colon, open parentheses, dollar sign or token, then a space is appended to the statement before returning to the dispatcher.

Line 60 is the start-of-input routine and lists all the visible BASIC programs in memory.

Line 61 opens the output file and prints the input prompt, then proceeds to find all the token pointers. Input is accepted but not displayed while this is taking place. This is an unconventional but effective time-saver.

Line 62 starts with the actual input statement. Here, whatever has been typed is displayed and, after the input is complete, any extension is dropped.

Line 63 converts any lower case letters to upper case.

Line 64 pads the program name with spaces to a length of six characters and searches for a BASIC program in memory with the same name. "AND NOT B" allows invisible program names to be recognized. If a match is found, then "A" is set equal to the start address, as is "A0." "GOSUB 17" prints the page header; then line 22 is the entry point.

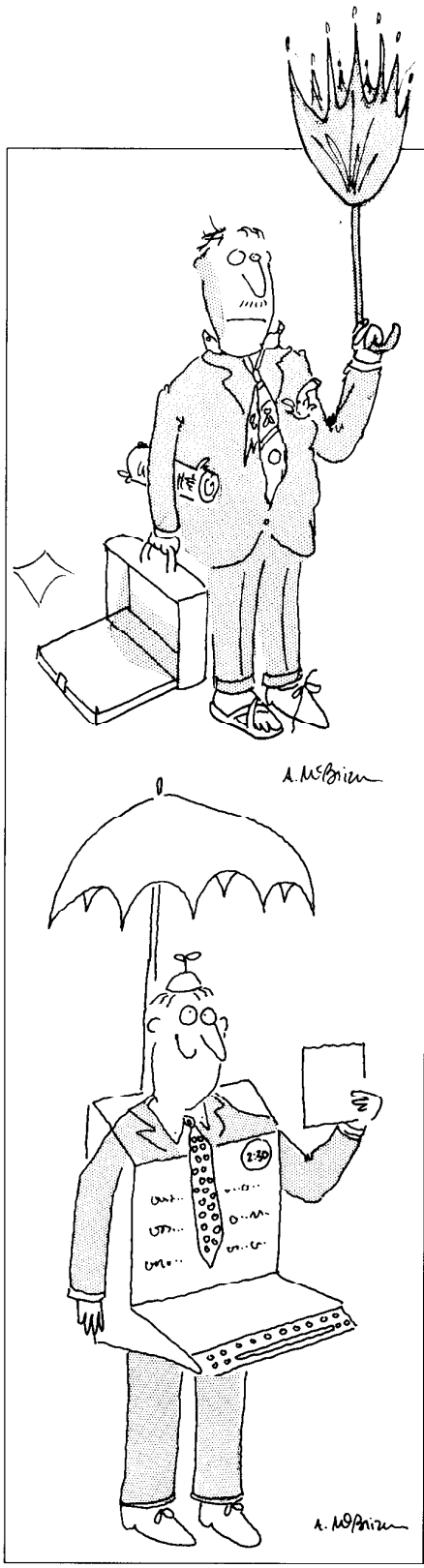
If no match is found, then line 65 prints "Try again" and line 62 accepts another file name.

That's all there is to it. As written, this program provides the necessary minimum of spaces. If more are desired, modify lines 33, 49 and 50 to taste.

Simpler variations on this program can be used to create a text file with spaces added for easy editing or with unnecessary spaces removed for compact storage. These will be left to the creative programmer.

—Larry McInteer

The Organizer



I've found the Model 100 to be the ideal tool to help with organizing my day-to-day activities. After seeing Ray Mendenhall's program ("Datebook for the Disorganized," July 1985), I decided to submit my version of a calendar program. My objective, like Mr. Mendenhall's, is to use the Model 100 to keep track of scheduled events and to remind me when those events are about to take place. I have, however, taken a somewhat different approach.

After asking you for the beginning and ending dates for the period in which you are interested, program AGENDA searches the NOTE.DO file to find entries that fall within the defined period. Selected entries are displayed one day at a time, and you signal the program to move on to the next day.

Records in the NOTE.DO file are formatted so that the first five characters represent the month and day of the event (in the format mm.dd). If character six of the entry contains the "+" symbol, it means the event starts on the date specified and continues for a given number of days. For example, an entry such as "02.10 + 04 Trip to New York" means the trip starts on February 10 and lasts for four days. Using such coding enables the program, with only one record in the NOTE.DO file, to handle events spanning several consecutive days.

My program starts by asking for the beginning and ending dates of your event. The dates are converted to Julian date format to set up a loop that determines how many times the NOTE.DO file will be searched (once for each day in the specified period). While going through each loop cycle, records of the NOTE.DO file are read, and the date in positions one through five is converted to Julian format.

If this date matches the current loop index, the entry's narrative portion is displayed. Records not matching the current loop index are ignored. Each loop cycle is terminated when either the last record of the NOTE.DO

```
1Ø REM Agenda by Hugh Lochrane
2Ø NV$-CHR$(27)+"q":RV$=CHR$(27)+"p"
3Ø CD$=LEFT$(DATE$,2)+MID$(DATE$,4,2)
4Ø M$="JanFebMarAprMayJunJulAugSepOctNovDec"
5Ø DY$(Ø)="Sun":DY$(1)="Mon":DY$(2)="Tue":
   DY$(3)="Wed":DY$(4)="Thu":DY$(5)="Fri":
   DY$(6)="Sat"
6Ø GOTO 19Ø
7Ø REM Gregorian to Julian Conversion
8Ø IF M1<3 THEN M1=M1+12:Y1=Y1-1
9Ø M1=M1+1:JD=INT(365.25*Y1)+INT(3Ø.6ØØ1*M1)
   +D1+1721Ø982:RETURN
1ØØ REM Julian to Gregorian Conversion
11Ø U=J+68569:V=INT(4*U/146Ø97)
    :U=-INT((146Ø97*V+3)/4)+U
    :Y=INT(4ØØØ*(U+1)/1461ØØ1)
    :U=-INT(1461*Y/4)+31+U)
    :M=INT(8Ø*U/2447):D=-INT(2447*M/8Ø)+U
    :U=INT(M/11):M=-12*U+2+M:Y=1ØØ*(V-49)+Y+U
    :MO$=MID$(M$,(M*3-2),3)
```

```
12Ø D2=J+1-7*INT((J+1)/7):DW$=DY$(D2)
13Ø RETURN
135 REM Strip off day, month and year
14Ø DD$=RIGHT$(A$,2):D1=VAL(DD$)
15Ø MM$=LEFT$(A$,2):M1=VAL(MM$)
16Ø YY$=RIGHT$(DATE$,2):Y1=VAL(YY$)+19ØØ
17Ø GOSUB 7Ø
18Ø RETURN
19Ø CLS:INPUT"Beg Date (MMDD)";A$:IF A$="" THEN
    A$=CD$:PRINT A$
2ØØ GOSUB 14Ø:B=JD
21Ø INPUT"End Date (MMDD)";A$:IF A$="" THEN
    A$=CD$:PRINT A$
22Ø GOSUB 14Ø:E=JD
225 REM Begin main loop
23Ø FOR I=B TO E
24Ø CLS:J=I:CT=Ø:GOSUB 1ØØ
25Ø PD$-DW$+" "+STR$(D)+" "+MO$+STR$(Y)
26Ø PRINT RV$;PD$;NV$
27Ø OPEN "NOTE.DO" FOR INPUT AS 1
28Ø IF EOF(1) THEN 39Ø
29Ø IC=Ø:LINE INPUT #1, A$
3ØØ IF VAL(LEFT$(A$,2))=Ø THEN 28Ø
31Ø D1=VAL(MID$(A$,4,2))
32Ø M1=VAL(MID$(A$,1,2))
33Ø Y1=AL(RIGHT$(DATE$,2))+19ØØ
34Ø GOSUB 7Ø
35Ø IF MID$(A$,6,1)="+" THEN IC=VAL(MID$(A$,7,2))
36Ø IF JD>I THEN 39Ø
37Ø IF I=JD OR I<JD+IC THEN CT=CT+1:GOSUB 43Ø
38Ø GOTO 28Ø
39Ø CLOSE
4ØØ IF CT>Ø THEN BEEP:INPUT "Press Enter for new
    day";A$
41Ø NEXT I
42Ø CLS:INPUT "Press Enter for Menu";A$
    :CLOSE:MENU
425 REM Display record
43Ø IF MID$(A$,6,1)="+" THEN PL$=MID$(A$,1Ø,
    LEN(A$)-9) ELSE PL$=MID$(A$,7,LEN(A$)-6)
44Ø PRINT PL$:IF CT>5 THEN INPUT "Press Enter for
    more items";A$:CLS:PRINT RV$;M;"-";D:PRINT
    NV$;:CT=Ø
45Ø RETURN
```

file is read, or when a record is read and its date is greater than the loop index. The last condition saves time by not continuing to read records from the NOTE.DO file when a date is greater than what is being searched. Of course, this requires that all records in the NOTE.DO file be in date sequence.

If entries are found for the date being searched, they are displayed. At the end of the loop for each day, you have a display of all events scheduled. Press *Enter* to continue to the next day in the specified period. The program automatically goes on to the next day, and you don't need to intervene for those days when no event is scheduled.

Program AGENDA accomplishes the fundamental tasks for helping you with a busy schedule. You can add some enhancements for additional capabilities. For example, you could add a routine to deal with events that happen every given day—say, a bowling league that meets every Tuesday. By putting in a special code at the first character of each record, and a code to signify the day of the week (0 for Sunday, 1 for Monday, and so on), you could have one record in the NOTE.DO file for each such event and have AGENDA automatically pick them up.

Another possibility is to add a printout routine so you can make a printed agenda for a specified period. This can be done by asking whether or not a printout is desired when the program begins, then adding the necessary LPRINT statements in the display subroutine.

The major routines in my program are:

(1) Lines 20-50 are merely housekeeping functions. Line 60 transfers control to the beginning of the program.

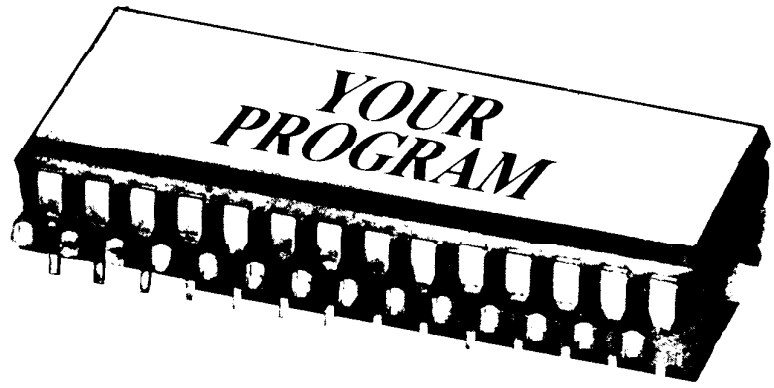
(2) Lines 70-90 are the subroutine that converts Gregorian dates to a Julian date format. For speed, this subroutine and the next one are located at the beginning of the program.

(3) Lines 100-130 are the subroutine that converts Julian dates to a Gregorian date format.

(4) Lines 190-220 are the beginning of the program that prompt for the beginning and ending dates of the period involved. Each date is converted to Julian format so it can be used as the loop's beginning and ending values.

(5) Lines 230-410 are the loop which reads through the NOTE.DO file, once for each day in the specified period. Line 300 may appear a bit strange, but I have other than calendar entries in NOTE.DO, and this statement serves to ignore them. You can delete it if you only have records in the format described above. Lines 310-330 pull out the date information from the record, and line 340 causes the date information to be converted to Julian format for comparison purposes. Line 370 compares the Julian date of the record being read with the loop index. If the two are equal, then subroutine 430 is called to display the record. —*Hugh Lochrane*

Your own Program on a ROM



8K or 32K
Your BASIC or machine code program on a custom ROM ready to snap into your Model 100, Tandy 102 or Tandy 200.
You can't tell your program from a built-in. Starting at $50.00 each. Minimum order of 10. Quantity pricing as low as $15^00

PORTABLE COMPUTER SUPPORT GROUP

11035 Harry Hines Blvd. No. 207, Dallas, Tx. 75229

1-214-351-0564

VISA / MASTER CARD / C.O.D.

Circle 52 on Reader Service Card

YOU BE THE EDITOR

Every reader has unique needs. Some seek guidance from reviews of software and peripherals. Others look to a magazine to provide tips to increase their programming proficiency. Accounts of innovative applications attract a big audience, as do articles about other users who find creative ways to use their systems.

You can help tailor **Portable 100** to your unique needs by rating the articles that appeared in this issue. Just circle the numbers indicated below on the Reader Service Card at left. Rate all the articles, some of them or just one. Then detach the card and drop it into the mail — we pay the postage.

The editors will analyze your feedback and respond by commissioning more articles of the most popular types — and fewer of the losers.

Indexing with SuperROM
Easy back-of-book indexes.
(page 8)
Excellent 200
Good 201
Poor 202

Portable File Cabinet
Understanding Tandy 600 FILE.
(page 12)
Excellent 203
Good 204
Poor 205

Quote Format for the Model 100
Making business decisions.
(page 18)
Excellent 206
Good 207
Poor 208

Utility Corner
Reading, Lister and Organizer.
(page 22)
Excellent 209
Good 210
Poor 211

The Plot Thickens
LCD function plotting.
(page 22)
Excellent 212
Good 213
Poor 214

Real Cheap Video
Build a video interface.
(page 39)
Excellent 215
Good 216
Poor 217

Checking It Twice
Avoiding disk errors.
Excellent 218
Good 219
Poor 220

OVERALL QUALITY OF THIS ISSUE

Excellent 221
Good 222
Poor 223

ADVERTISERS INDEX

Free Information

For free information on products advertised in this issue of *Portable 100,* locate the Reader Service number corresponding to the advertisement that interests you. Circle the number on the card at left and drop it into the mail. The literature you've requested will be forwarded to you without obligation. Please allow 3-5 weeks for delivery.

Reader Service No.	ADVERTISER	Page Number
48	American Cryptronics	15
47	Axonix	27
	80 Micro	42
55	Jones Service & Design	28
54	Kangaroo	43
65	Merritt Computer Products	14
40	PCSG	CII
52	PCSG	31
71	PCSG	5
75	PCSG	41
43	PG Design	47
	Portable 100	16
74	Purple Computing	7
61	Sias Engineering	17
70	Sigea Systems	48
60	Simons Products	17
76	Soundsight	1
73	Soundsight	46
67	Sunbelt Software	14
42	Tandy	CIII
44	Tandy	10
72	Traveling Software	33
41	Traveling Software	CIV
49	Ultrasoft	21
45	Ultrasoft	23
46	Ultrasoft	25

Free Product Information!

portable 100

To find out more about the products and services in *Portable 100*:

1. Fill in your name and address in the space provided below.
2. Circle the numbers on this card that correspond to the reader service numbers of the advertisers in which you are interested.
3. Fill out the reader survey and detach and mail. No postage necessary.

portable 100

November 1987
This card valid until December 31, 1987

Name _____

Company _____

Address _____

City/State/Zip _____ Phone _____

1	26	51	76	101	126	151	176	201	226	251	276				
2	27	52	77	102	127	152	177	202	227	252	277				
3	28	53	78	103	128	153	178	203	228	253	278				
4	29	54	79	104	129	154	179	204	229	254	279				
5	30	55	80	105	130	155	180	205	230	255	280				
6	31	56	81	106	131	156	181	206	231	256	281				
7	32	57	82	107	132	157	182	207	232	257	282				
8	33	58	83	108	133	158	183	208	233	258	283				
9	34	59	84	109	134	159	184	209	234	259	284				
10	35	60	85	110	135	160	185	210	235	260	285				
11	36	61	86	111	136	161	186	211	236	261	286				
12	37	62	87	112	137	162	187	212	237	262	287				
13	38	63	88	113	138	163	188	213	238	263	288				
14	39	64	89	114	139	164	189	214	239	264	289				
15	40	65	90	115	140	165	190	215	240	265	290				
16	41	66	91	116	141	166	191	216	241	266	291				
17	42	67	92	117	142	167	192	217	242	267	292				
18	43	68	93	118	143	168	193	218	243	268	293				
19	44	69	94	119	144	169	194	219	244	269	294				
20	45	70	95	120	145	170	195	220	245	270	295				
21	46	71	96	121	146	171	196	221	246	271	296				
22	47	72	97	122	147	172	197	222	247	272	297				
23	48	73	98	123	148	173	198	223	248	273	298				
24	49	74	99	124	149	174	199	224	249	274	299				
25	50	75	100	125	150	175	200	225	250	275	300				

1. Which portable(s) do you own?
- A. ☐ Model 100
- B. ☐ Tandy 102
- C. ☐ Tandy 200
- D. ☐ Tandy 600
- E. ☐ NEC 8201
- F. ☐ Olivetti M10

2. What are your most important applications for your portable?
- A. ☐ Word Processing
- B. ☐ Database Management
- C. ☐ Telecommunications
- D. ☐ Programming
- E. ☐ Scheduling/Time Management
- F. ☐ Spreadsheets
- G. ☐ Other _____

3. Do you use your portable for...
- A. ☐ Business Use
- B. ☐ Personal Use
- C. ☐ Both of the above

4. What peripherals and accessories do you currently own for your portable?
- A. ☐ Disk Drive
- B. ☐ Memory Upgrade (RAM)
- C. ☐ Accessory ROM Pac
- D. ☐ Modem
- E. ☐ Printer
- F. ☐ Tape Drive

5. What peripherals and accessories do you plan to purchase for your portable during the next 6 months?
- A. ☐ Disk Drive
- B. ☐ Memory Upgrade (RAM)
- C. ☐ Add-On ROM
- D. ☐ Modem
- E. ☐ Printer
- F. ☐ Tape Drive
- G. ☐ Carrying Case

6. How much do you plan to spend on portable computer hardware, software, and peripherals during the next 6 months?
- A. ☐ less than $100
- B. ☐ $100-$250
- C. ☐ $250-$400
- D. ☐ $400-$600
- E. ☐ $600-$750
- F. ☐ more than $750

7. How often do you use on-line services?
- A. ☐ Daily
- B. ☐ Weekly
- C. ☐ Monthly
- D. ☐ Other _____

8. In what type of business do you use your portable?
- A. ☐ Journalism
- B. ☐ Insurance
- C. ☐ Retail
- D. ☐ Education
- E. ☐ Science or Engineering
- F. ☐ Business
- G. ☐ Other _____

9. If you are not a subscriber, please circle 299 on this card.

10. If you would like a one year subscription to *Portable 100*, please circle 300 on this card. Each subscription costs $24.97 (Canada $45.97 in Canadian Funds, Mexico $29.97, US Funds drawn on US Bank, one year only). Please allow 10-12 weeks for delivery.

portable 100

November 1987
This card valid until December 31, 1987

Name _____

Company _____

Address _____

City/State/Zip _____ Phone _____

1	26	51	76	101	126	151	176	201	226	251	276				
2	27	52	77	102	127	152	177	202	227	252	277				
3	28	53	78	103	128	153	178	203	228	253	278				
4	29	54	79	104	129	154	179	204	229	254	279				
5	30	55	80	105	130	155	180	205	230	255	280				
6	31	56	81	106	131	156	181	206	231	256	281				
7	32	57	82	107	132	157	182	207	232	257	282				
8	33	58	83	108	133	158	183	208	233	258	283				
9	34	59	84	109	134	159	184	209	234	259	284				
10	35	60	85	110	135	160	185	210	235	260	285				
11	36	61	86	111	136	161	186	211	236	261	286				
12	37	62	87	112	137	162	187	212	237	262	287				
13	38	63	88	113	138	163	188	213	238	263	288				
14	39	64	89	114	139	164	189	214	239	264	289				
15	40	65	90	115	140	165	190	215	240	265	290				
16	41	66	91	116	141	166	191	216	241	266	291				
17	42	67	92	117	142	167	192	217	242	267	292				
18	43	68	93	118	143	168	193	218	243	268	293				
19	44	69	94	119	144	169	194	219	244	269	294				
20	45	70	95	120	145	170	195	220	245	270	295				
21	46	71	96	121	146	171	196	221	246	271	296				
22	47	72	97	122	147	172	197	222	247	272	297				
23	48	73	98	123	148	173	198	223	248	273	298				
24	49	74	99	124	149	174	199	224	249	274	299				
25	50	75	100	125	150	175	200	225	250	275	300				

1. Which portable(s) do you own?
- A. ☐ Model 100
- B. ☐ Tandy 102
- C. ☐ Tandy 200
- D. ☐ Tandy 600
- E. ☐ NEC 8201
- F. ☐ Olivetti M10

2. What are your most important applications for your portable?
- A. ☐ Word Processing
- B. ☐ Database Management
- C. ☐ Telecommunications
- D. ☐ Programming
- E. ☐ Scheduling/Time Management
- F. ☐ Spreadsheets
- G. ☐ Other _____

3. Do you use your portable for...
- A. ☐ Business Use
- B. ☐ Personal Use
- C. ☐ Both of the above

4. What peripherals and accessories do you currently own for your portable?
- A. ☐ Disk Drive
- B. ☐ Memory Upgrade (RAM)
- C. ☐ Accessory ROM Pac
- D. ☐ Modem
- E. ☐ Printer
- F. ☐ Tape Drive

5. What peripherals and accessories do you plan to purchase for your portable during the next 6 months?
- A. ☐ Disk Drive
- B. ☐ Memory Upgrade (RAM)
- C. ☐ Add-On ROM
- D. ☐ Modem
- E. ☐ Printer
- F. ☐ Tape Drive
- G. ☐ Carrying Case

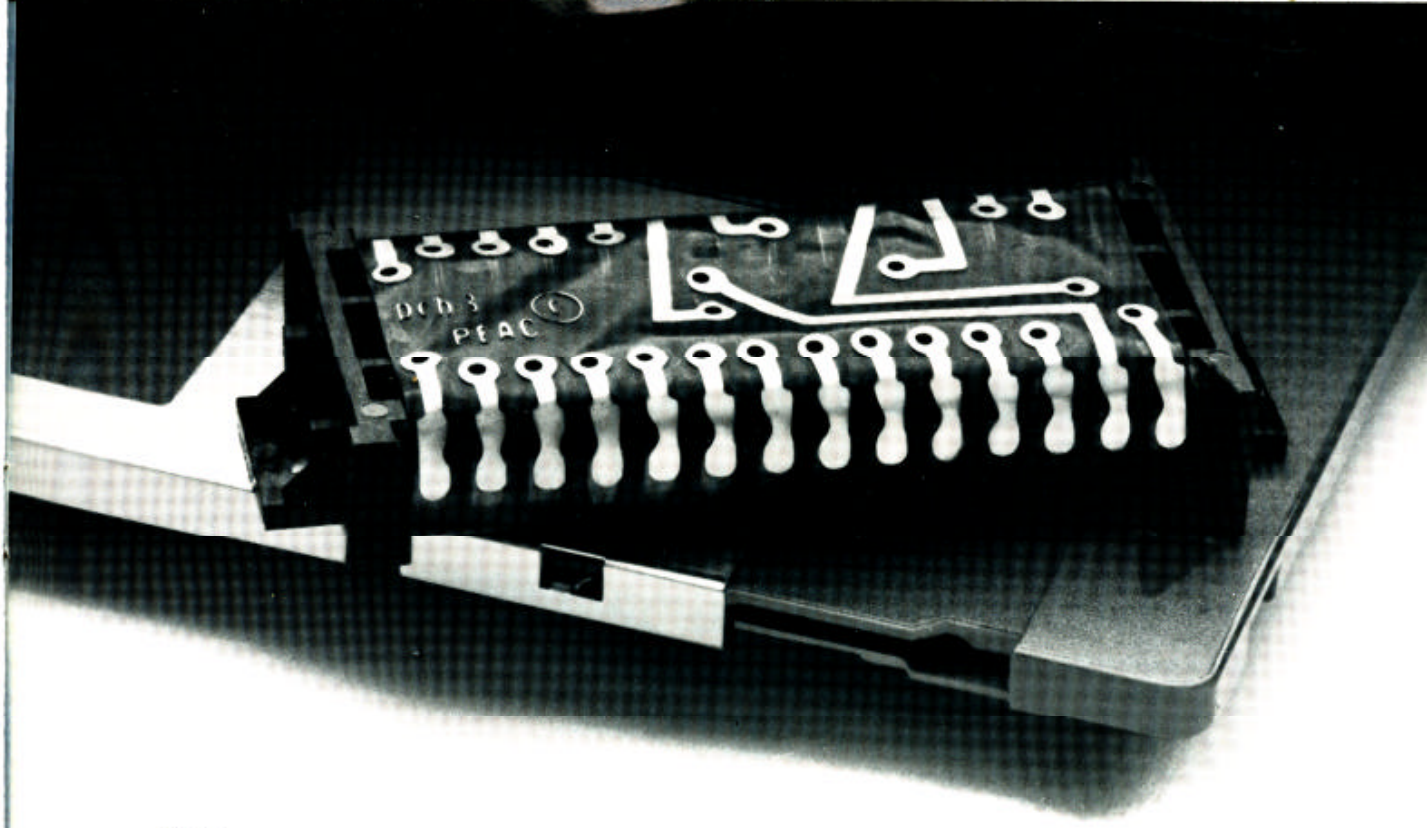
6. How much do you plan to spend on portable computer hardware, software, and peripherals during the next 6 months?
- A. ☐ less than $100
- B. ☐ $100-$250
- C. ☐ $250-$400
- D. ☐ $400-$600
- E. ☐ $600-$750
- F. ☐ more than $750

7. How often do you use on-line services?
- A. ☐ Daily
- B. ☐ Weekly
- C. ☐ Monthly
- D. ☐ Other _____

8. In what type of business do you use your portable?
- A. ☐ Journalism
- B. ☐ Insurance
- C. ☐ Retail
- D. ☐ Education
- E. ☐ Science or Engineering
- F. ☐ Business
- G. ☐ Other _____

9. If you are not a subscriber, please circle 299 on this card.

10. If you would like a one year subscription to *Portable 100*, please circle 300 on this card. Each subscription costs $24.97 (Canada $45.97 in Canadian Funds, Mexico $29.97, US Funds drawn on US Bank, one year only). Please allow 10-12 weeks for delivery.



Now you can put your own programs on a plug-in ROM chip.

A custom ROM service designed especially for corporate and government clients.

Now, Traveling Software can transform your own BASIC or machine code programs into executable applications in a built-in ROM chip—by using a process called "Guardian" that took two years and thousands of man hours to develop.

Fast, simple, economical and guaranteed.

It's a very simple, speedy process to have your own applications "burned" into a plug-in ROM chip. All you do is send Traveling Software your programs on a disk with a check for $500 for your first ROM. Then, the Traveling Software engineers will convert your program(s) into a plug-in ROM chip.

Quantity purchases can achieve a per ROM cost of only $49, with full credit for the initial $500 cost. Normal service turnaround is 7-10 working days. Optional rush service of 24-48 hours is possible.

Combine Traveling Software Programs with your own.

For orders of 50 or more custom ROM's, Traveling Software is offering the option to combine popular products like TS-DOS, T-Word, IDEA!, or T-base with your programs on a single chip.

Build your own "ultimate ROM" using any Traveling Software product with your own custom applications.

The possibilities are endless! For instance, you can even join a bar code reader data logging application with the disk access powers of TS-DOS and put it on a single plug-in chip!

Tandy and NEC Computer Sales.

Looking for a Tandy Model 100/102, 200 or NEC PC-8201 laptop computer to go along with your custom ROM's? Traveling Software is now an authorized distributor of these computers. Let Traveling Software take care of all your laptop computing needs with just one purchase order!

FEATURES

- Proven technology, thousands of Guardian ROM's have been installed since early 1985.
- Traveling Software has years of experience with laptop computers and ROM technologies.
- Economical solutions to many applications, as low as $49 per custom ROM.
- Built-in future protection! Program changes can be "reburned" into existing chips without replacing costly CMOS ROM chips.

A few current Guardian ROM customers:

- Associated Press
- McDonald's Corporation
- Swiss News Agency
- Matrix Instruments, Inc.
- Professional Growth Associates
- Biolabs Inc.
- Shell Pipeline Company
- Del Mar Avionics
- Business Systems Design, Inc.
- Ocean State Scale
- Pruett Industries
- Specialty Instruments

"Absolutely fantastic! Since we began using Guardian custom ROM's, we have installed over 500 systems in less than nine months."
Tom Villars
Project Manager
Enterprise Systems, Inc.

CUSTOM SERVICES

Circle 72 on Reader Service Card

*For your FREE **ROM** information kit call toll free:*

1 800 343-8080

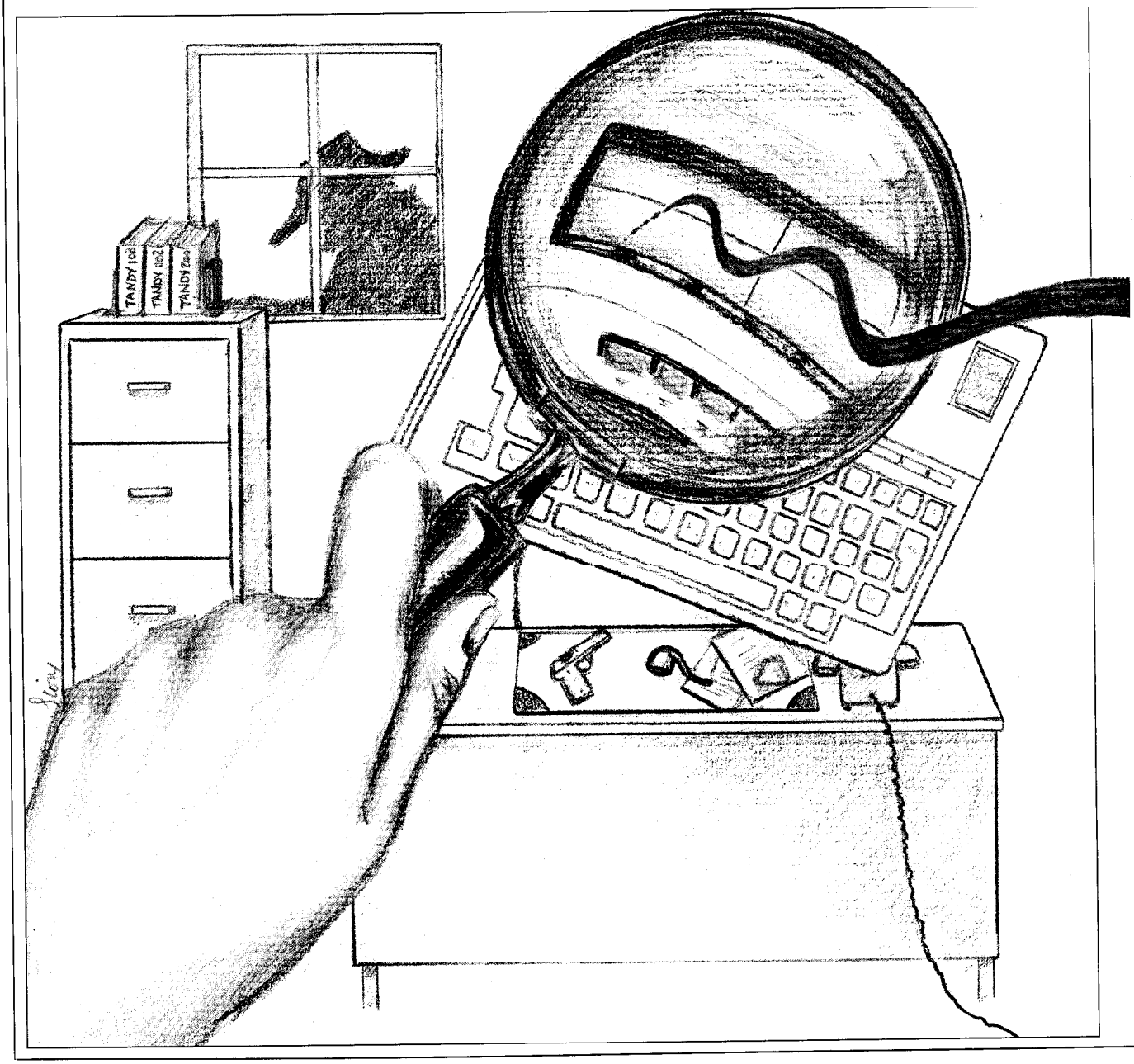
Or write: **Traveling Software**
North Creek Corporate Center
19310 North Creek Parkway
Bothell, Washington 98011
206-483-8088

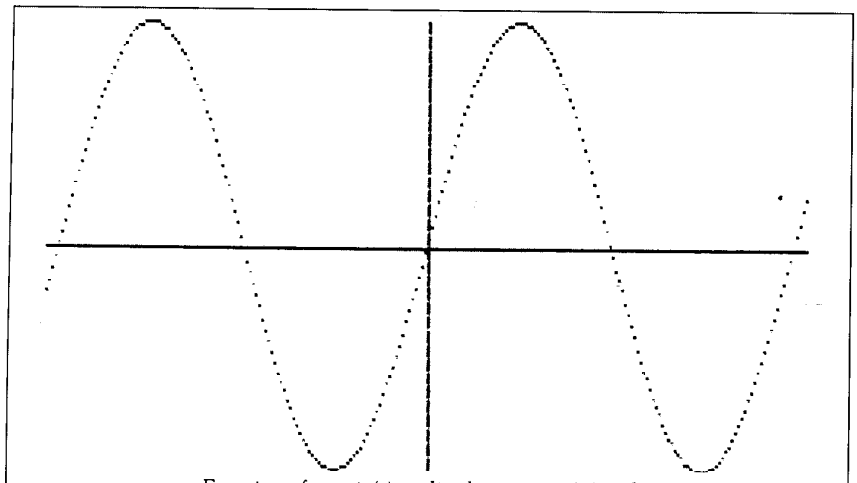
TRAVELING SOFTWARE®

The Plot Thickens

Using the Model 100 for quick-and-dirty function plotting.

By Alan L. Zeichick





Equation of y = sin(x); x lies between −6.5 and 6.5.

My father-in-law is a statistics professor. He loves gadgets, especially computers and fancy scientific calculators. Last time I saw him, he was engrossed in his latest toy: a pocket calculator with a large, two-inch-square LCD display. But this was more than a mere calculator; it was also a miniature on-screen plotter. By simply entering an equation and *x*-coordinate limits, any algebraic function could be displayed on the bit-mapped LCD. "Can your Model 100 do this?" he asked.

It took some time, but the two of us designed and programmed GRAF.BA, a simple function graphing program for the Model 100 and Tandy 102. Later, I added the flexibility to enable this program to work on the Tandy 200.

OF MICE AND MEN

Modern computer graphics programs come in many shapes and forms, from the high-powered MS-DOS drafting package AutoCAD to the Macintosh's recreational MacPaint. But Graf didn't require that level of sophistication. Indeed, its only purpose was to quickly plot the graph of a mathematical equation, allowing the mathematician, statistician or engineer to look at the *shape* of the curve. Frequently, this information is meaningful for checking the accuracy of a problem's solution.

Despite the fact that Graf was intended to create simple graphs on the laptop's liquid-crystal display, the program's design took some time. As with any non-trivial product, there were questions to be asked and answers to be found: Where will the equation come from? How will the *x*-*y* coordinate pairs be generated, and between what limits? Will the program automatically scale the finished plot? Will the program also display the graph's axes? What happens to horizontal lines, or plots of 1/0, or the square root of -1? As we answered each question, the finished product slowly took shape.

A truly sophisticated and user-friendly mathematics package should prompt the user to solve an equation and evaluate it internally. In other words, it's pretty tacky to require the user to actually modify the program by inserting the desired equation:

```
10000 Y = SIN(X)
```

That's exactly what we did, though. Why? Some versions of the BASIC language allow you to enter the right side of an equation as a string variable, say EQ$, and evaluate it as such:

```
100 EQ$ = "X + 4"
110 X = 17
120 Y = EVAL(EQ$)
```

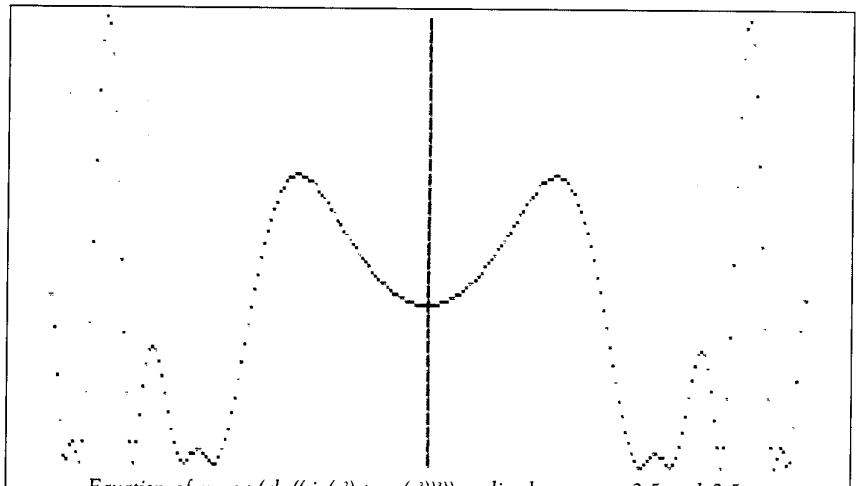
yielding a value of *y* equal to 21. But not the faithful and trusty Model 100 laptop BASIC.

There's another way, I thought, to evaluate an equation entered from the keyboard in a program: save the equation to a RAM file, and merge it into the program file, using BASIC's MERGE function. The program fragment would look something like this:

```
10 PRINT "What's the equation?"
20 LINE INPUT EQ$
30 OPEN "EQ.DO" FOR OUTPUT
   AS 1
40 PRINT #1, "80 " + EQ$
50 CLOSE #1
60 MERGE "EQ.DO"
70 X = 17
90 PRINT "Y = ";Y
```

If the user entered "Y = X + 4" at the prompt, I thought, the program would return the result 17. But no such luck; unlike esoteric functions such as PRINT and OPEN, the MERGE statement terminates program execution after the merge. Trying to use CONT to resume execution results in a *?CN can't continue* error message; the merging operation seems to change the BASIC workspace sufficiently to forbid further program interpretation.

After much experimentation, my



Equation of $y = sqr(abs((sin(x^2) + cos(x^3))^3))$; x lies between −2.5 and 2.5.

father-and-law and I became resigned to the tacky, but functional, modification of the program for each desired graph. We decided to use the simple variable names *x* and *y*, and to place the equation at the easy-to-remember line 10000; by placing a RETURN at line 10010, we can call the equation via a subroutine call. Whew! One hurdle safely passed.

At least we didn't have too much difficulty with the next problem: the case of the upper and lower function limits. A simple subroutine at lines 1400 and 1480 did the trick, along with a little error-checking. Many programmers write error-checking routines after the program is completed; I've found that if error checking is written into the product from the very start, it's less likely that potential errors will be forgotten—and besides, the user-designed error messages can help during the program building and debugging stages. I like using a common error-handling function, which I've placed at lines 2010-2060; it prints "Error" in inverse video, displays my error message and waits for the user to press a key. By assigning text to the string variable ER$ and branching to that subroutine, error messages can be displayed with minimal code duplication.

Next logical step: generating the *x*-*y* coordinate pairs. First of all, how many? On all three laptops, the screen has a horizontal resolution of 240 dots (or pixels) numbered 0 to 239. So, we'll need 240 *x* values, generated by a FOR-NEXT loop, and 240 *y* values, stored in an array appropriately named Y (line 1060).

If we're generating *x* values in a loop, what's the increment? If the initial *x* value, say, is 0, and the highest *x* value is 239—one for each pixel—what should we add to *x* each time through our loop? The answer is one, and in the general case, that's found by subtracting the lower limit from the upper, and dividing by 239. That's determined in line 1520; the variable name *XI* is an abbreviation for *x increment*.

When it's time to similarly scale the *y* coordinates, we'll need to know the lowest and highest *y* values. Let's use two variables, *YL* and *YH*, to keep track of those in our subroutine between lines 1520 and 1690. Initially, the *x* value is our lowest *x*, entered

Listing for the formatted function plotter.

```
1000 '
1010 ' Function Plotter
1020 ' Alan L. Zeichick, Portable 100
1030 '
1040 ' Dimension data array, select machine
1050 '
1060 DIM Y(240)
1070 MD=200
1080 IF MD=200 THEN VH=128 ELSE VH=64
1090 '
1100 ' Input user information
1110 '
1120 GOSUB 1400
1130 '
1140 ' Generate points
1150 '
1160 GOSUB 1520
1170 IF ER$ <> "" THEN 1310
1180 '
1190 ' Perform scale calculations
1200 '
1210 GOSUB 1760
1220 IF IN$=CHR$(27) THEN 1120 ELSE IF IN$="M" OR
     IN$="m" THEN MENU
1230 '
1240 ' Plot points on screen
1250 '
1260 GOSUB 1890
1270 '
1280 ' All done
1290 '
1300 CLS
1310 PRINT "Press Enter for BASIC"
1320 PRINT "Esc for new values or"
1330 PRINT "M for Main Menu:"
1340 IN$=INPUT$(1)
1350 IF IN$=CHR$(27) THEN 1120 ELSE IF IN$="M" OR
     IN$="m" THEN MENU ELSE END
1360 END
1370 '
1380 ' Routine to find user information
1390 '
1400 CLS
1410 PRINT "Portable 100 Function Plotter"
1420 PRINT
1430 INPUT "Enter lower x-value"; IN$
1440 IF LEFT$(IN$,1)="M" THEN MENU ELSE
     XL=VAL(IN$)
1450 INPUT "Enter upper x-value"; IN$
1460 IF LEFT$(IN$,1)="M" THEN MENU ELSE
     XH=VAL(IN$)
1470 IF XL >= XH THEN ER$="Higher value isn't
     higher.":GOSUB 2010:GOTO 1400
```

```
148Ø RETURN
149Ø '
15ØØ ' Routine to calculate array of points
151Ø '
152Ø XI=(XH-XL)/239
153Ø X=XL
154Ø ON ERROR GOTO 21ØØ
155Ø ER$=""
156Ø GOSUB 1ØØØØ:IF ER$ <> "" THEN RETURN
157Ø YL=Y
158Ø YH=Y
159Ø Y(Ø)=Y
16ØØ PRINT
161Ø PRINT "... working ...";
162Ø '
163Ø FOR L=1 TO 239
164Ø   IF L MOD 12=Ø THEN PRINT ".";
165Ø   X=X+XI
166Ø   GOSUB 1ØØØØ:IF ER$ <> "" THEN RETURN
167Ø   IF Y < YL THEN YL=Y ELSE IF Y > YH THEN
YH=Y
168Ø   Y(L)=Y
169Ø   NEXT L
17ØØ '
171Ø ON ERROR GOTO Ø
172Ø RETURN
173Ø '
174Ø ' Routine to calculate scale
175Ø '
176Ø YI=(YH-YL)/(VH-1)
177Ø '
178Ø CLS
179Ø PRINT "X-minimum ="; XL; TAB(2Ø); "maximum
    ="; XH
18ØØ PRINT "Y-minimum ="; YL, TAB(2Ø); "maximum
    -"; YH
181Ø PRINT
182Ø PRINT "Press Enter to Plot, Esc for New
    Values"
183Ø PRINT "B for BASIC or M for Main Menu."
184Ø IN$=INPUT$(1)
185Ø RETURN
186Ø '
187Ø ' Routine to plot points and axes
188Ø '
189Ø CLS
1891 IF YL=YH THEN PRINT "The plot is a
    horizontal line.";INPUT$(1):RETURN
19ØØ IF XL*XH<=Ø THEN X=239*XL/(XL-XH)+.5:LINE
    (X,Ø)-(X,VH-1)
191Ø IF YL*YH<=Ø THEN Y=(VH-1)*YH/(YH-YL)+.5:LINE
    (Ø,Y)-(239,Y)
```

earlier and *YL* and *YH* are equal to the first *y* value calculated in line 1560. The loop at lines 1630-1690 increments the *x* value by our *XI* factor, calculates a new value of *y* through a subroutine call, updates the *YL* and *YH* variables, and stores the new *y* in the *Y* array. This short seven-line loop takes the most execution time of any part of Graf; depending on the complexity of the equation at line 10000, this could take many minutes. The equation *y = sqr(abs((sin(xΛ2) + cos(xΛ3))Λ3))* took about 2.5 minutes, for instance. (By the way, plot that equation between -2 and 2—it's interesting!)

Line 1640, which prints a little period (".") on the display every few seconds, slows the loop slightly—but I feel that it's worth it. No user enjoys that feeling of helplessness while a program grinds slowly on, minute by minute: "Is it working? Is it dead? Hello?" If you want to optimize the program anyway, simply delete line 1640. Incidently, since the program contains no branches to comments, you can eliminate all of the comment lines when typing, to save effort and even conserve RAM.

During our calculation process, error-checking is in full force. System-generated errors caused by the line 10000 equation cause an ON ERROR GOTO branch to line 2100; the Model 100 error code is tested against three common math problems: illegal function calls (such as square root of -1), numeric overflow and division by zero. I recommend that you not enter line 1540, the ON ERROR statement, until you've verified that the program is working correctly—since it assumes that all errors are mathematical in origin, it might be difficult to find typographical errors.

THE POINTING FINGER

Remember how we scaled the horizontal *x* values by dividing the difference in extremes by 239, which was one less than the number of pixels? Well, we can similarly scale the vertical *y* values, by dividing by 63 on the Model 100 and Tandy 102, and 127 on the Tandy 200. How do we know which is which? Line 1070 contains a statement, "MD = 200" saying that the computer model used is the Tandy 200. If your machine is a 100 or 102, enter that value. The

```
1920 '
1930 FOR L=0 TO 239
1940   PSET (L, (VH-1)-INT((Y(L)-YL)/YI+.5))
1950   NEXT L
1960 '
1970 IF INKEY$="" THEN 1970 ELSE RETURN
1980 '
1990 ' Routine to handle errors
2000 '
2010 CLS
2020 PRINT CHR$(27); "pError"; CHR$(27); "q"
2030 PRINT ER$
2040 PRINT
2050 PRINT "Press a key..."; INPUT$(1)
2060 RETURN
2070 '
2080 ' Routine to handle math errors
2090 '
2100 IF ERR=5 THEN ER$="Illegal function call"
     ELSE IF ERR=6 THEN ER$="Numeric overflow"
     ELSE IF ERR=11 THEN ER$="Division by zero"
     ELSE ER$="Unknown error: "+STR$(ERR)
2110 RESUME 2120
2120 ER$=ER$+", x="+STR$(X)
2130 GOSUB 2010
2140 RETURN
2150 '
2160 ' Here's the equation, line 10000
2170 '
10000 Y=SQR(ABS((SIN(X^2)+COS(X^3))^3))
10010 RETURN
```

very next line uses a simple IF statement to set a vertical height (*VH*) to either 64 or 128 pixels, as appropriate. (Yes, it would have been even friendlier to use exotic PEEKs to find out which machine we have—but not quite so obvious to someone trying to understand the program, or adapt it for another computer.)

Graf's line 1760 calculates the y values' scaling increment, or *YI*. Next, the program displays the *x* and y extremes: *XL*, *XH*, *YL* and *YH*. If they look reasonable, pressing *Enter* starts the plot. Knowing the extremes gives you a frame of reference for understanding the plot, especially since the vertical and horizontal scales will probably be different. A 45-degree line, plotted by the equation $y = x$, looks like about 15 degrees when graphed on a Model 100; on a Tandy 200, it's about 28 degrees. Pay attention to the extreme values; any straight line with a slope greater than zero (in other words, going up) will be graphed from the lower left to upper right points on the LCD. Graf only shows you the *shape* of the graph, not its actual mathematical values.

Upon entering the plotting subroutines, lines 1890 through 1980, Graf checks to see if the graph is a simple horizontal line by comparing *YL* and *YH*. If the two values are identical, there's no way the graph can be meaningful, so the plot is aborted with an appropriate message.

Next, the horizontal and vertical axes are plotted if the *x* or y values span zero. How can you tell if the lower value of y, say, is less than or equal to zero, and the upper value is greater than or equal to zero? The obvious way is with a convoluted IF-THEN statement with lots of OR clauses. An easier way is to multiply the two values; if the result is non-positive, our graph can show the $y = 0$ axis. Lines 1910 and 1920 determine if an axis is appropriate, and if so, calculate its position and display it using the Model 100 BASIC LINE graphics statement.

Finally, the display of the graph itself, taken care of by a FOR-NEXT loop at lines 1940–1960. For every *x*, the corresponding y, appropriately scaled, is retrieved from the Y array and plotted using the Model 100 PSET statement. Adding one-half here and earlier compensates for the fact that PSET and LINE statements truncate their arguments: the vertical position 11.99, for example, is plotted as position 11, not 12. Adding .5 changes the truncation to an up-and-down rounding, which is more appropriate in this situation.

Notice that we subtract 63 (actually, *VH-1*) from our y coordinate. Why? Unlike the standard geometrical Cartesian plane, the Model 100's (0,0) origin is at the top left corner of the display, and the vertical pixels are numbered from the top down. Since all of our calculations assume the standard plane, numbered bottom up, subtracting from 63 effectively reverses the values.

READY TO ROLL

My father-in-law enjoyed the prototype of this program; since then, I've added some features, including the user menus (such as at lines 1310-1350), and the variable vertical height to accomodate the Tandy 200. The Graf program is essentially complete, and ready to be used for any number of applications: checking homework, examining experimental results, testing hypothesis. Since the actual mathematical function isn't limited to line 10000, you can use several lines with IF-THEN reasoning to construct your plot data points. It's also not impractical to replace line 10000 with a READ or INPUT statement, to scale plot any other *x*-y coordinate pairs. Master the possibilities!□

Real Cheap Video for the Model 100

An economical project with minimal sweat and programming.

By Greg Bouffard

With few modifications and little money, you can display the image on your Model 100's LCD screen simultaneously on a CRT.

For less than $200, I constructed a compact box featuring an existing external port, no hardware modifications on the 100, and minimal software programming (saving the 100's limited memory capacity for other uses).

I used the Model 100's printer port because of its useful functions. The *Print* key lets you print any information from the LCD screen to the printer port. In TELCOM, *F5* turns on the echo mode, and any information sent or received is also relayed to the printer port.

All the normal printer functions used in BASIC, and most application software programs, are also available. Therefore, any information displayed on the LCD screen can be sent via the printer port.

To convert the information into displayable video, I designed a printer-interface-to-video-terminal keyboard interface converter (*Fig. 1*), hereafter called a keyboard interface.

I used a Linger Enterprises Model 65/9028 VT video terminal board, purchased from Digital Research Computers (P.O. Box 381450, Duncanville, Tex. 75138). It is available

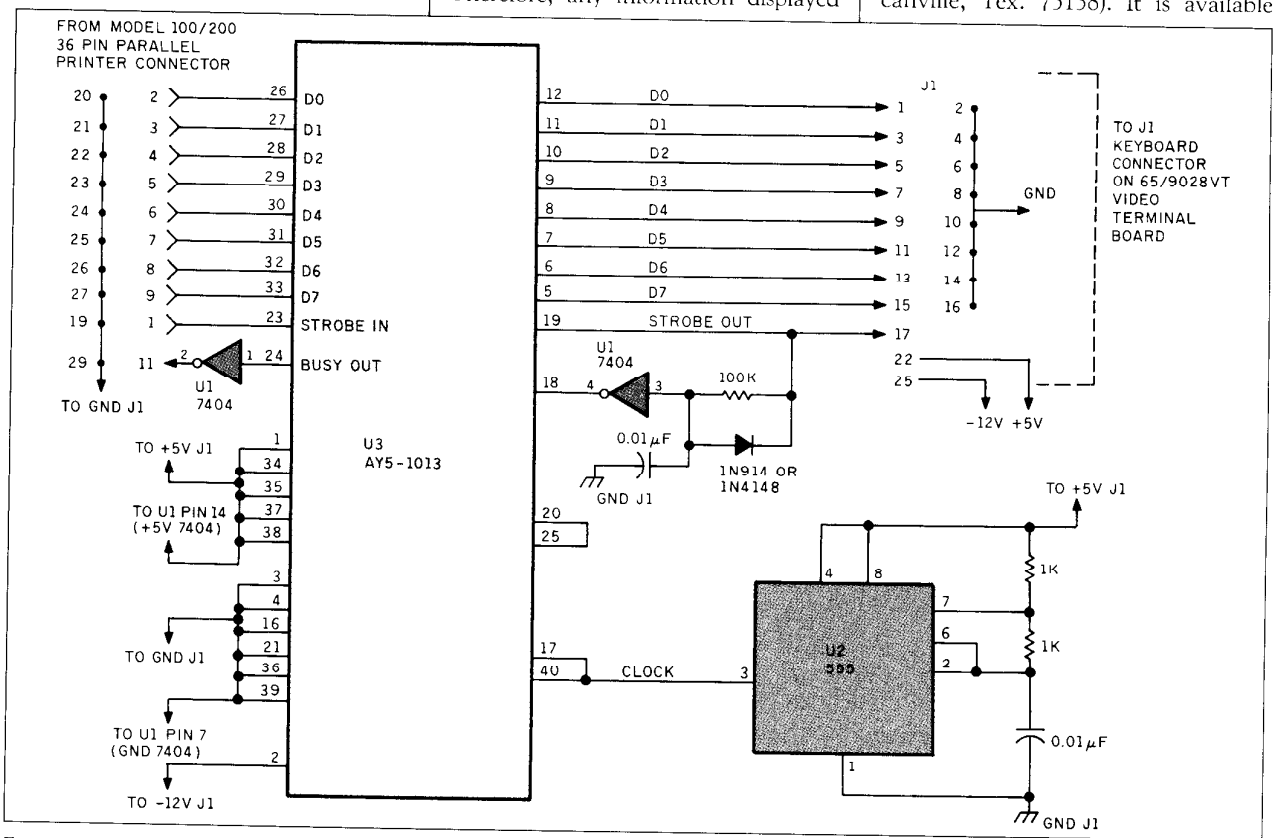


Figure 1. The Model 100/200 parallel printer to 65/9028 VT keyboard interface.

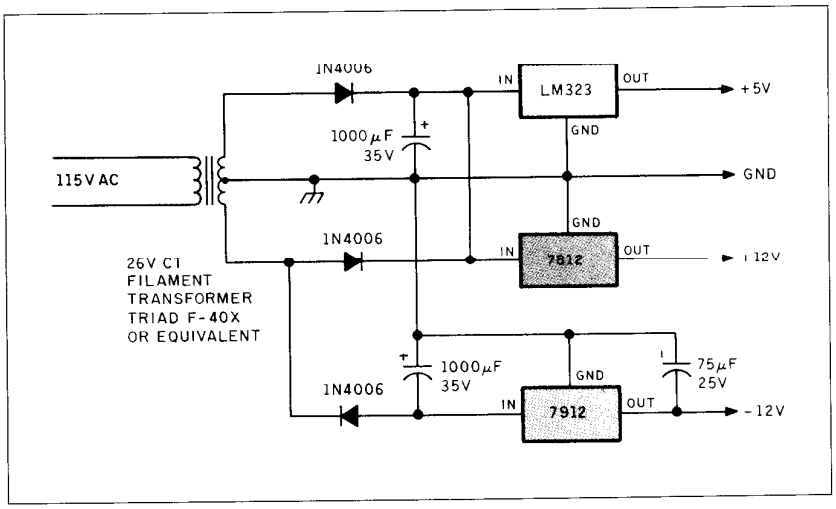


Figure 2. The power supply schematic.

unassembled for $99.95 or assembled and tested for $139.95. (If you are not an experienced kit-builder, I suggest you buy the ready-made version.)

KEYBOARD MODIFICATION

The 65/9028 is normally used with a CRT and ASCII keyboard to make a complete, ASCII, asynchronous terminal. However, I replaced the ASCII keyboard with the Model 100 and its keyboard interface circuit. This allows characters sent from the Model 100's printer port to be displayed on a CRT attached to the 65/9028.

The keyboard interface circuit consists of an AY5-1013 UART, a 555 timer, and 7404 hex inverter ICs plus support components. All parts are available from most local electronic suppliers or mail order supply catalogs. I selected components based on availability and price (approximately $15-$20).

A data monitor is another useful feature of this "real cheap" video.

Mount the components on a 3" by 4" perforated circuit board using solid hook-up wire for all connections. Next, solder the wires directly to the legs of all ICs. If you're not handy with a soldering iron, I recommend using IC sockets and soldering the connection to them.

Use a 26-pin ribbon cable to connect the keyboard interface to the 65/9028; power for the keyboard interface is supplied from the 65/9028 through this cable. A 36-pin parallel printer connector is needed so you can use the standard Model 100 printer interface cable to join the 100 to the keyboard interface. I mounted the connector interface and all interface connectors in a 8.5" by 6" by 2" metal box.

My keyboard interface and 65/9028 are supported by a power supply I constructed inside my CRT (Fig. 2). If the CRT does not have space available, you can construct the power supply in a separate box, or buy one ready-made. Make sure it has a rating of +5V at 2A and +12V and −12V.

The 65/9028 board comes with an excellent reference manual that gives detailed information about operating, assembling, and interfacing the board to a CRT and power supply. The 65/9028 can be interfaced to many types of black-and-white monitors, using either composite video or separate video and synchronous inputs.

TERMINAL BLISS

I also connected a 25-pin ribbon cable to a 25-pin DB25 female RS-232 connector to J2 (the 65/9028's RS-232 I/O connector). This step is unnecessary if you want to display

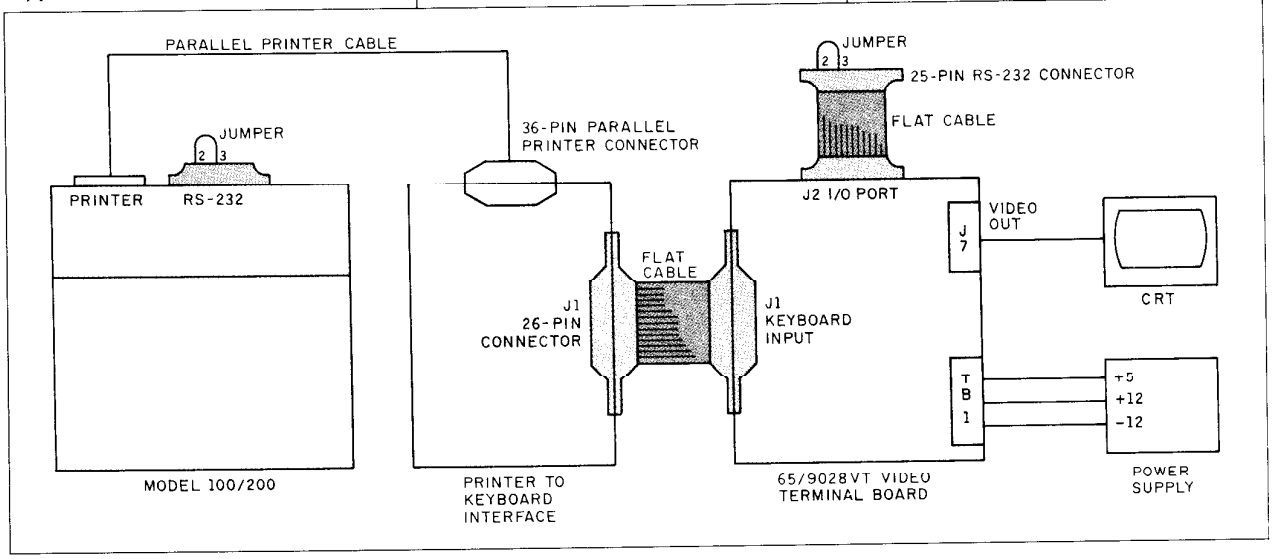


Figure 3. Connecting the keyboard interface and VT board to a monitor and power supply.

6 ROM BANK

plus powerful Nicad battery pack that gives 30 hours of power to your Model 100

A joint PCSG/CRYPTRONICS project.

Now you can access LUCID, DISK+, WRITE ROM, RAM+ and others instantly.

The 6 ROM BANK plus battery pack lets you have the ultimate Model 100 system. When you get your unit just press in up to 6 ROMS into the sockets then push in a simple plug into your Model 100's ROM socket. The ROM bank has two posts on either end that insert into the two little holes on the underside of your Model 100.

The ROM bank props up the Model 100 at the same angle and height as those little legs you've seen. The ROM BANK itself is only about 1½" deep and it runs the width of your Model 100. It only weighs one pound. It not only installs instantly, but it pops free in a second if you need everything to lie flat in a briefcase.

Change from ROM to ROM with the touch of a thumb switch.

You can go from LUCID to WRITE to DISK+ to any other ROMS just by turning the thumb switch at the side of the ROM bank. The 6 ROM BANK is a sturdy well built construction that looks like it is a part of your Model 100.

What is also fantastic is that the ROM bank has a powerful NICAD battery and recharger built right in. This power source supplies up to 30 hours of life to your Model 100 with just 6 hours of recharge. What's nice is that it recharges right from your Model 100's power adapter. This is a quick charge system and if you need power in a hurry, you can get 6 hours of life for your Model 100 by just charging for an hour and half.

Suddenly, the Model 100 is a very powerful computer.

If you have the HOLMES portable disk drive with its powerful bundled database software package or the CRYPTRONICS 128K RAM expansion along with LUCID, WRITE ROM and DISK+ in the 6 ROM bank with its rapid rechargable NICAD power source, you have the ultimate portable system. It's all available only from PCSG.

As usual we don't want you to take our word for it. The 6 ROM BANK is sold on a 30 day trial. If you are not satisfied simply return within thirty days for a full refund. MC, VISA, AM. EX. or C.O.D.

1-214-351-0564

PORTABLE COMPUTER SUPPORT GROUP

11035 Harry Hines Blvd. No. 207, Dallas, Tx. 75229

VISA / MASTER CARD / AM. EXP. / C.O.D.

Circle 75 on Reader Service Card

Give A Gift That Will Be Opened Every Month

This Season, Give 80 Micro

Why give a present that will be opened just once when you can give one that will be opened year-round? You'll be remembered this season, and every season, when you send the perfect gift for Tandy users—an **80 Micro** subscription.

Month after month, **80 Micro** will be a valuable, practical, and sometimes surprising guidebook for the dedicated Tandy user. And for an incredibly low price, it offers more assistance and inspires more efficiency than most peripherals and software packages. Just look at the benefits:

- Shop-at-home convenience with concise new product information.
- Answers to both common and unusual computing problems in "Feedback Loop".
- Thorough evaluations and quality ratings of the latest Tandy-compatible products.

Plus, a variety of type-and-run program listings, up-to-the-minute industry trends, and much more.

All in the only system-specific source on the market for Tandy users.

All for just $24.97—12 issues at **48% off** the cover price. And you won't be billed until **after** the holidays.

Send an All Seasons Greetings to someone special this season by returning the coupon, or by calling 1-800-258-5473. (In NH, dial 1-924-9471.)

YES! **I Want To Give A Year-Round Gift.**

Please send an **80 Micro** gift subscription to the person listed below. I'll pay $24.97 for 12 issues—a 48% savings.

☐ Payment Enclosed ☐ Bill me after the holidays!

Please make checks payable to **80 Micro**.

Name of Gift Recipient _____

Address _____

City _____ State _____ Zip _____

My Name _____

Address _____

City _____ State _____ Zip _____

Canada $45.97 (Canadian Funds), Mexico $29.97, Foreign Surface $44.97, 1 year only, US Funds drawn on US Bank. Foreign Airmail please inquire. All gift subscriptions will begin with the first available issue in 1988.

80 Micro • PO Box 981 • Farmingdale, NY 11737

37NB4

```
10 REM "Initialization Program"
90 CLS
100 PRINT @50, "Initialize Terminal"
110 PRINT @90, "Sets <LEADIN> to Ctrl C"
120 PRINT @130, "Sets Local Mode ON"
130 LPRINT CHR$(27);"]";CHR$(27)
131 FOR I=1 TO 500:NEXT
135 LPRINT CHR$(3);CHR$(13)
136 FOR I=1 TO 500:NEXT
140 LPRINT CHR$(3);"9"
150 LPRINT "Terminal Initialized"
160 LPRINT "<LEADIN> character is Ctrl C"
190 MENU
```

Figure 4. The initialization program sets up the lead-in characters used to configure the 65/9028 VT board and sets the VT to local mode.

video only from a Model 100, however, the connector allows the Model 100 to emulate an ANSI X3.64-1979 (VT-100), a Heath H19 or an ADM-3A terminal. If you do not want this option, jumper pin 2 to 3 on J2.

After you've finished, connect the keyboard interface and 65/9028 to a video monitor and power supply. When you turn on the unit, a cursor should be visible in the CRT's upper-lefthand corner. If not, turn off the unit, check all connections and jumpers, and use the 65/9028's reference manual to determine the correct jumper configuration.

You can now use all Model 100 printer functions to display LCD information.

Once the cursor appears on the CRT, use a standard Model 100 parallel printer cable to connect the computer to the keyboard interface (*Fig. 3*). Jumper pins 2 to 3 on the Model 100's RS-232 port connector, and connect pins 2 and 3 of the 65/9028's RS-232 I/O port. Go to TELCOM and set the port status to any baud rate and parity. As a test, I set stats to 57I1E.

Go to TERM and set up for full duplex and echo on the 100's keyboard. Now, run the initialization program (*Fig. 4*), which sets up the lead-in characters used to configure the 65/9028 board and sets it to local mode. The lead-in character will now be *Ctrl-C*.

Use *Ctrl-C* 1 to get to the 65/9028's setup status mode and set the auto linefeed to "on." If necessary, refer to the manual for information on setting the terminal's setup status.

You can now use all Model 100 printer functions to display LCD information on the CRT.

A data monitor is another useful feature of this "real cheap" video project. While running the Model 100 and the video in terminal mode, as described previously, the 65/9028 can be set to display characters received at its RS-232 port in hex. This function can be useful when troubleshooting communications problems between various PCs. Refer to the 65/9028's reference manual for information on enabling the display hex mode.

If you are a frequent Model 100 online user, just dial into the service using the Model 100's internal modem with echo (*F5*) on. The CRT will display the data received and your keyboard responses. Since the CRT can display 80 characters by 24 lines of data, using these online services with the Model 100 and video unit will be much easier and more enjoyable than using the LCD display.

The "real cheap" video display unit should also work with a Model 200 and any other computer having a standard parallel printer interface. □

U.S. POSTAL SERVICE
STATEMENT OF OWNERSHIP, MANAGEMENT AND CIRCULATION
(Required by 39 U.S.C. 3685)

1. TITLE OF PUBLICATION: portable 100
1A. PUBLICATION NO.: [illegible]
2. DATE OF FILING: Sept. 1, 198[illegible]
3. FREQUENCY OF ISSUE: Monthly
3A. NO. OF ISSUES PUBLISHED ANNUALLY: 12
3B. ANNUAL SUBSCRIPTION PRICE: $[illegible]
4. COMPLETE MAILING ADDRESS OF KNOWN OFFICE OF PUBLICATION *(Street, City, County, State and ZIP Code) (Not printers)*: 80 Elm Street, Peterborough, Hillsborough County, N.H. [illegible]
5. COMPLETE MAILING ADDRESS OF THE HEADQUARTERS OR GENERAL BUSINESS OFFICES OF THE PUBLISHERS *(Not printer)*: 80 Elm Street, Peterborough, Hillsborough County, N.H. [illegible]
6. FULL NAMES AND COMPLETE MAILING ADDRESS OF PUBLISHER, EDITOR, AND MANAGING EDITOR *(This item MUST NOT be blank)*
PUBLISHER *(Name and Complete Mailing Address)*: Jeff DeTray, 80 Elm Street, Peterborough, N.H. [illegible]
EDITOR *(Name and Complete Mailing Address)*: Roger Strukhoff, 80 Elm Street, Peterborough, N.H. [illegible]
MANAGING EDITOR *(Name and Complete Mailing Address)*: Steve Smith, 80 Elm Street, Peterborough, N.H. [illegible]
7. OWNER *(If owned by a corporation, its name and address must be stated and also immediately thereunder the names and addresses of stockholders owning or holding 1 percent or more of total amount of stock. If not owned by a corporation, the names and addresses of the individual owners must be given. If owned by a partnership or other unincorporated firm, its name and address, as well as that of each individual must be given. If the publication is published by a nonprofit organization, its name and address must be stated.) (Item must be completed)*

FULL NAME	COMPLETE MAILING ADDRESS
International Data Group	[illegible] MA [illegible]

8. KNOWN BONDHOLDERS, MORTGAGEES, AND OTHER SECURITY HOLDERS OWNING OR HOLDING 1 PERCENT OR MORE OF TOTAL AMOUNT OF BONDS, MORTGAGES OR OTHER SECURITIES *(If there are none, so state)*

FULL NAME	COMPLETE MAILING ADDRESS
[illegible]	[illegible] MA [illegible]

9. FOR COMPLETION BY NONPROFIT ORGANIZATIONS AUTHORIZED TO MAIL AT SPECIAL RATES *(Section 411.3, DMM only)*
The purpose, function, and nonprofit status of this organization and the exempt status for Federal income tax purposes *(Check one)*
☐ (1) HAS NOT CHANGED DURING PRECEDING 12 MONTHS ☐ (2) HAS CHANGED DURING PRECEDING 12 MONTHS *(If changed, publisher must submit explanation of change with this statement.)*

10. EXTENT AND NATURE OF CIRCULATION	AVERAGE NO. COPIES EACH ISSUE DURING PRECEDING 12 MONTHS	ACTUAL NO. COPIES OF SINGLE ISSUE PUBLISHED NEAREST TO FILING DATE
A. TOTAL NO. COPIES *(Net Press Run)*	[illegible]	[illegible]
B. PAID CIRCULATION 1. SALES THROUGH DEALERS AND CARRIERS, STREET VENDORS AND COUNTER SALES	[illegible]	[illegible]
2. MAIL SUBSCRIPTION	11,[illegible]	[illegible]
C. TOTAL PAID CIRCULATION *(Sum of 10B1 and 10B2)*	[illegible]	[illegible]
D. FREE DISTRIBUTION BY MAIL, CARRIER OR OTHER MEANS SAMPLES, COMPLIMENTARY, AND OTHER FREE COPIES	[illegible]	[illegible]
E. TOTAL DISTRIBUTION *(Sum of C and D)*	[illegible]	[illegible]
F. COPIES NOT DISTRIBUTED 1. OFFICE USE, LEFT OVER, UNACCOUNTED, SPOILED AFTER PRINTING	2,806	3,252
2. RETURN FROM NEWS AGENTS	7,032	7,177
G. TOTAL *(Sum of E, F1 and 2 - should equal net press run shown in A)*	26,819	27,189

11. I certify that the statements made by me above are correct and complete — SIGNATURE AND TITLE OF EDITOR, PUBLISHER, BUSINESS MANAGER, OR OWNER

PS Form 3526, July 1981 *(See instruction on reverse)* *(Page 1)*

Checking It Twice

There's nothing worse than a disk error.

By Jim Gove

Click, click, clack. Whirr. Whirr-Whirr-Sputter. [Error reading drive.] Sigh.

Although the Portable Disk Drive (PDD), thanks to its 3.5-inch technology, is a very reliable Model 100-family accessory, problems with physically or logically-damaged disks can occur. And when they do, nobody's happy.

I spend a fair amount of time on the road and always travel with my Model 100 and PDD. When I return to the office, I unload my document files onto my MS-DOS desktop using Traveling Software's LapDOS. I find the combination pretty near perfect.

One potential problem has nagged at me since my very first PDD-equipped field trip: what if my disk is defective?

Due to circumstances, it's difficult to double-check the file on disk, since one of my short stories or articles will occupy more than half of my available RAM (32K). There simply isn't enough memory remaining in RAM to pull the saved file back into the Model 100 before killing the original file. Backing up the file to a second disk (which I routinely do) is not foolproof, as mistakes or equipment malfunction can easily produce two bad disk files.

Although my laptop system has always performed flawlessly, the thought of losing a week's work gives me the jitters each time I have a work session in a hotel room, without a printer to make a printed backup. Hence my solution: a program that allows me to take a look at a file on a disk without fully loading it into RAM. The program itself only occupies 747 bytes within the Model 100 or Tandy 200; it opens a file and loads it into memory a string at a time, using only a few bytes of memory. I can read the disk-based article on the screen, and catch any major errors. If I'm in a hurry, I can let the program do all the work and simply watch for hardware error-messages.

The program is structured into four sections, a header section that contains the prompts, two functional sections where the reading and screen printing takes place, and a final section where the file is closed and the program is exited.

Two reading and screen printing sections are provided. One prints all of the text in the disk file. The second prints only the first significant string and the final string. The second section, (first and last) is fastest, and therefore is the one I usually use. A 15,645-byte test file takes about 26 seconds to display using the first-and-last method; it takes nearly four minutes using the full-text testing.

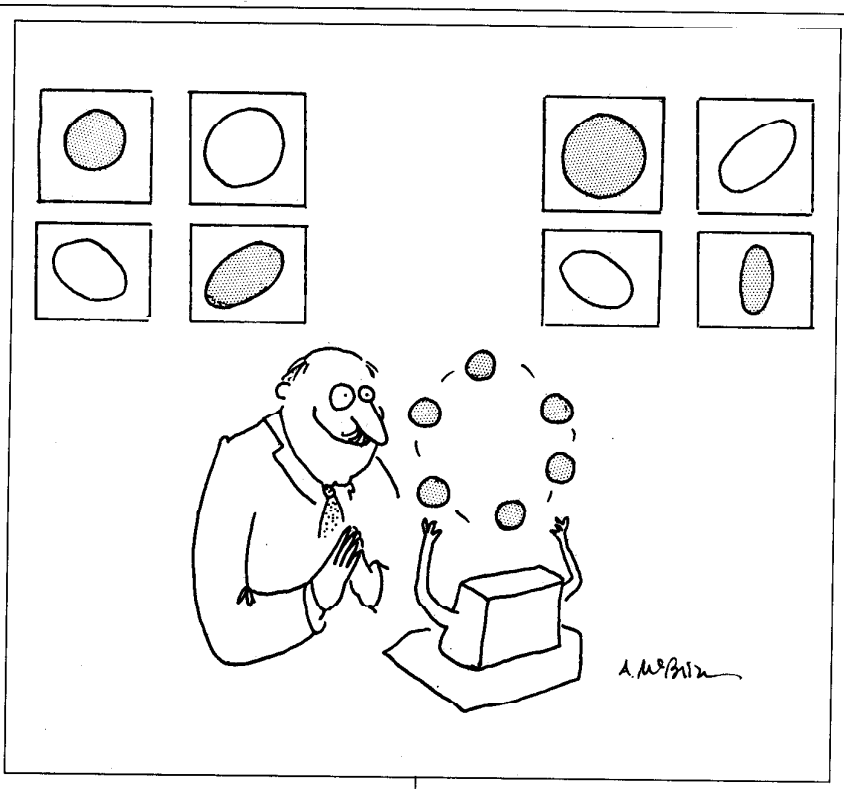
The program is a simple one, but for those who routinely produce large text files and then wonder if the load-to-disk has been successful, it's valuable.□

Listing for disk checker program.

```
10 ' FCHECK
13 '------HEADER Program Section------
15 CLEAR FRE(A)/4
20 PRINT "Please input the file name.":INPUT A$
30 PRINT:INPUT "Do you wish to print (A)ll or
   (F)irst and last"; B$
50 IF B$="A" OR B$="a" THEN 1001
60 IF B$="F" OR B$="f" THEN 2001
70 PRINT:PRINT "Incorrect input, please
   repeat":PRINT:GOTO 30
1000 ------ALL Program Section-------
1001 CLS
1002 OPEN "0:"+A$ FOR INPUT AS 1
1010 LINE INPUT #1, AA$
1011 PRINT AA$
1020 IF EOF(1) THEN 3001
1030 GOTO 1010
2000 '--FIRST & LAST Program Section--
2001 CLS
2002 OPEN "0:"+A$ FOR INPUT AS 1
2020 LINE INPUT #1, AA$
2030 PRINT AA$
2031 IF LEN(AA$)<3 THEN 2020
2040 LINE INPUT #1,AA$
2050 IF EOF(1) THEN 2070
2060 GOTO 2040
2070 PRINT AA$
3000 ' ------END Program Section------
3001 CLOSE #1
3010 PRINT:INPUT "(A)gain or (Q)uit";A$
3020 IF A$="A" OR A$="a" THEN CLS:GOTO 20
3030 PRINT:PRINT "End of Program.":END
```

Going Round In Circles

As we explored last month, the Model 100 offers reasonable bit-mapped graphics, with a 64-by-200 pixel (picture element) display. And the Tandy 200's offers twice the resolution—128-by-200 pixels. We also introduced the happy blockhead, a simple graphics drawing created with the Model 100 and Tandy 200 built-in graphics commands (PSET, PRESET and LINE). This month, we'll use these same graphics statements, but show how computation can significantly extend the capacity of Model 100-series graphics.



AROUND WE GO

A graphics function found on many other microcomputers is *circle*, which draws circles on the computer display when given the center coordinates and a radius. But the Model 100 doesn't have a circle command. What to do?

We can calculate a circle. A circle can be defined as all points in a plane a certain distance, *r* or *radius*, from another point, or *center*. Every point on the line is at a unique angle, relative to the normal *x*–*y* coordinate system, from the center. Working backwards, we can reason that if we generate all possible angles, and plot a point one radius away from the center at those angles, we'll have a circle.

A STEP AT A TIME

Let's test our reasoning. First, we'll need a loop to generate angles. Since the Model 100 family of trigonometric functions (which we'll need) use radian measure, a circle comprises

$$2 \times \pi$$

or approximately 6.3 radians. For this test, we'll pick angles one-tenth radian apart:

```
10 FOR A = 0 TO 6.3 STEP .1
```

To find a point that's at distance *radius* from the center of a circle and angle *a*, the *x*-coordinate is *sin(a)*radius* and the *y*-coordinate is *cos(a)*radius*. If the center's not at location (0,0), add the appropriate value to the *x* and *y* coordinate. Let's assume our radius is 30 pixels, and our circle's center is at (100,32). We'll add 0.5 to our center to account for the fact that the graphics commands only use the whole-number portions of arguments:

```
20 PSET (30*SIN(A) + 100.5,30*COS
(A) + 32.5)
```

Finally, let's close off the loop by selecting the next angle:

```
30 NEXT A
```

Try the program. A circle will be drawn slowly on your laptop's liquid crystal display. Well, almost a circle: It's full of gaps. Obviously, our angles should be closer together than 0.1 radians, but how close?

The circumference, or distance around, a circle is 2 × π × the radius. In our example above, the circle with a radius of 30 pixels has a circumference of about 188 pixels. But our FOR-NEXT loop only iterated 64 times. That explains the problem; for a circle this size, the STEP value should have been 0.033, which is 6.3 divided by 188.

Reducing the fractions leaves the STEP value as simply the reciprocal of the radius:

```
10 FOR A = 0 TO 6.3 STEP 1 / 30
```

Try that version of the program; you'll find that the circle is now solidly complete. Change the radius value *30* in lines 10 and 20—you'll still have a solid circle, which takes about 50 seconds to draw.

SPEED AND GRACE

Our little Model 100 is working very hard. For our radius-30 circle, the FOR-NEXT loop is iterating 190 times, and we're performing 380 trigonometric calculations. Would you believe that we can do the same task in a fraction of the time, with 24

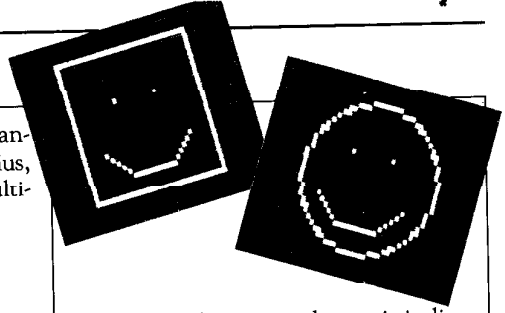
iterations and only 48 trigonometric calculations?

A circle is mathematically symmetrical. If we compute only the right side of the circle, we can easily—with simple subtraction—calculate the left side. Similarly, if we compute the upper-right quadrant, we can very easily calculate the other three quadrants. Going one step farther, if we use trigonometry to find the pixels for a mere one-eighth of the circle, we can find the remainder by using subtraction, and by swapping the sine and cosine values.

That means we only need one-eighth the number of loop iterations, and if we store the intermediate results, only one-eighth the number of sine and cosine function calls. This yields a drawing time of only eight seconds—instead of 50—for a radius-30 circle!

Let's use this approach to write a subroutine to produce high-speed circles. The variables *cx* and *cy* represent the coordinates of the circle's center, *cr* is the circle's radius, *ca* is the iterated angle, *cs* is the temporary angular sine multiplied by the radius, and *cc* is similarly the cosine multiplied by the radius:

```
10 CLS
20 CX = 100
30 CY = 32
40 CR = 30
50 GOSUB 1000
60 IF INKEY$ = "" THEN 60 ELSE
   MENU
999 ' Circle-drawing subroutine
1000 FOR CA = 0 TO 0.79 + 1/
     CR STEP 1/CR
1010 CS = SIN(CA)*CR + .5
1020 CC = COS(CA)*CR + .5
1030 PSET (CX + CS,CY + CC)
1040 PSET (CX-CS,CY + CC)
1050 PSET (CX + CS,CY-CC)
1060 PSET (CX-CS,CY-CC)
1070 PSET (CX + CC,CY + CS)
1080 PSET (CX-CC,CY + CS)
1090 PSET (CX + CC,CY-CS)
1100 PSET (CX-CC,CY-CS)
1110 NEXT CA
1120 RETURN
```



The only substantive change is in line 1000, the FOR statement. The TO value is 2 × π ÷ 8, or 0.79—but the reciprocal of the radius is added in, to ensure that the eight arc sections overlap slightly.

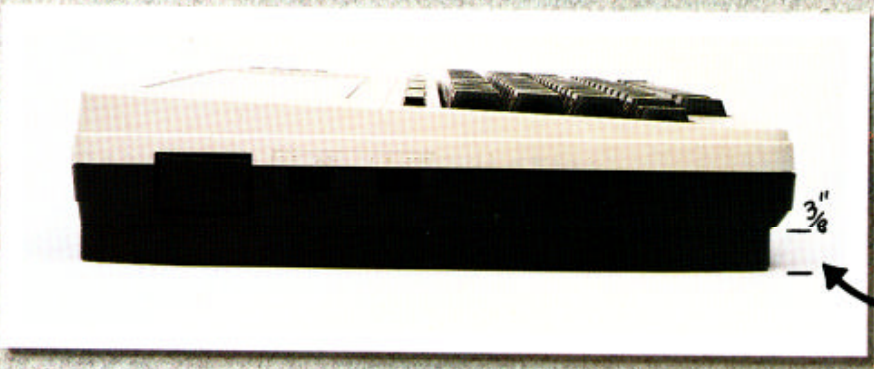
HAPPY ROUNDHEADS

To demonstrate the use of this circle subroutine in conjunction with other graphics statements, let's turn last month's happy blockhead into a happy roundhead. The only change I've made is to change the PSET statements in the circle subroutine to PRESETs to create a white circle. That, by the way, is one of my few criticisms of Microsoft's BASIC: why couldn't PSET work like LINE, and have a color argument?□

—Alan L. Zeichick

Find the memory in these pictures.

Stock Tandy 102 with 24K of memory.



Tandy 102 with 1 megabyte of RAM Vault memory by P.G. Design.

IF YOU NEED MORE MEMORY, YOU NEED THE VAULT!

The Vault adds as much as 1 megabyte of continuous RAM memory to the Tandy 102. With the Vault, you can maintain as many as 256 separate menu files (not just 19). It comes with four ROM sockets: 2 standard, 2 Radio Shack compatible. Adding only 3/8 inch of thickness, the Vault attaches firmly and easily with four screws.

For pricing information and technical questions, call us today!

313/727-2744

Circle 43 on Reader Service Card

PG DESIGN ELECTRONICS, INC.

37560 Thirty-one Mile Road, Richmond, MI 48062

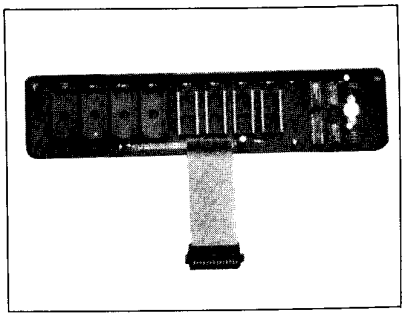
SAFE ROM HOLDER

If your laptop application calls for more than one option ROM, check out SAFE. The latest offering from PG Design Electronics, SAFE is an electronically switchable, multiple-ROM bank for the Model 100, Tandy 102 and 200, and NEC PC-8201 and PC-8300.

SAFE, priced at $129, will hold as many as eight 32K ROMs, four Tandy-compatible and four industry-standard 27C256 ROMs in the Tandy version, and eight 27C256 in the NEC version.

According to company President Peter Van Heusden, the electronic switching feature allows programming of custom 256K programs. PG Design Electronics assists users with the special programming of ROMs to take full advantage of SAFE's switching capabilities.

The six-ounce, 11.5" by 2.75" by 1" SAFE attaches to the bottom of the laptop and connects to the computer's external system bus port.



Contact PG Design Electronics, 37560 31 Mile Road, Richmond, MI 48062, (313) 727-2744.

Circle 105 on Reader Service Card.

A NEW DOS

When Tandy introduced its Portable Disk Drive (PDD), several vendors responded with improved disk operating systems (DOS) for the 3.5-inch drive. With the release of the improved PDD-2, with 200K storage per disk, some vendors again responded. Ultrasoft Innovation's latest product is Disk Power II.

Disk Power II ($59.95) offers several improvements over the original Disk Power software. The new version can handle the PDD-2's larger 80-file capacity by providing two 40-file disk subdirectories. Also, text being received via the laptop's built-in 300-bps modem can be saved portion-by-portion to disk without dropping the telephone connection.

Original Disk Power features are maintained with Disk Power II. There are rapid saves, loads or deletions of as many as 20 files in one operation, display of file size and available storage on-screen, and direct save and load of documents from TEXT.

Enhancements to BASIC include functions to read and write sequential data files as large as 64K on disk. TELCOM is also enhanced with the added ability to view files while uploading, to enter BASIC or TEXT and to return to TELCOM without dropping the telephone connection.

Contact Ultrasoft Innovations, 76 Main St., Champlain, NY 12919, (514) 457-9293.

Circle 106 on Reader Service Card.

Tandy Computers:
Because there is
no better value.™

The Choice Is Yours

The slimline Tandy 102 or the Tandy 200 with large flip-top screen.

The Tandy® 102—only $499. This slim, light portable gives you complete computing power at a portable price. Its small size makes it ideal for people on the go. With five built-in programs, the Tandy 102 is ready to go to work right away. Use the 102 as a personal word processor, address/phone directory, appointment calendar, telephone auto-dialer and modem. You can even write your own programs in BASIC.

The Tandy 200—more power for only $799. It's ideal for accountants, financial planners, or anyone who works with figures. You can perform sales forecasts, budgeting, pricing, engineering calculations and more. The Tandy 200 features BASIC programming language, a built-in direct-connect modem and a larger 40 × 16 display with double-height characters for maximum legibility. Built-in Multiplan™ makes spreadsheet analysis a snap. You also get an enhanced version of the Tandy 102's word-processing program, as well as an appointment calendar, address-and-phone directory, and telephone auto dialer.

Need more storage? Add our battery-powered Portable Disk Drive 2 for only $199.95. You'll be able to store up to 200K of data on a standard 3½" disk.

Send me a new 1988 computer catalog.

Mail To: Radio Shack
Dept. 88-A-1125
300 One Tandy Center
Fort Worth, TX 76102

Name _____
Company _____
Address _____
City _____ State _____
ZIP _____
Phone _____

Radio Shack®
The Technology Store™
A DIVISION OF TANDY CORPORATION

Multiplan/TM Microsoft Corp.

Circle 42 on Reader Service card.

Ever Dream of a Faster Modem, Longer Battery Life, and All the RAM Memory and ROM Software you want?

Wake Up! BOOSTER PAK is Here!

Traveling Software introduces the BOOSTER PAK! An incredible breakthrough for Tandy Model 100/102 laptop computers. The BOOSTER PAK gives you an unprecedented 2 megabytes of RAM memory and ROM software. At a very affordable price.

Flexible Design

The BOOSTER PAK is designed to meet your needs now and in the future. You can customize it as you go.

Easy Installation

Just snap the BOOSTER PAK onto the bottom of your Tandy 100/102. Plug in two cables and you're in business.

Built-in Software

Equally as incredible as the hardware is the built-in BOOSTER PAK software. Our unique virtual RAM disk software eliminates any need to do "bank switching." File subdirectories are supported for easy organization of your data. BASIC programs can access files as large as your RAM memory—up to two megabytes! We have also included our popular TS-DOS disk operating system for use with the Tandy disk drive and our new Desk-Link software, as well as an XMODEM program to upload/download files up to 2 megabytes directly to the virtual RAM disk. And to top it off, we have even included a fast action Asteroids game with amazing graphics in our standard BOOSTER PAK.

The BOOSTER PAK Everything You Asked For and More!

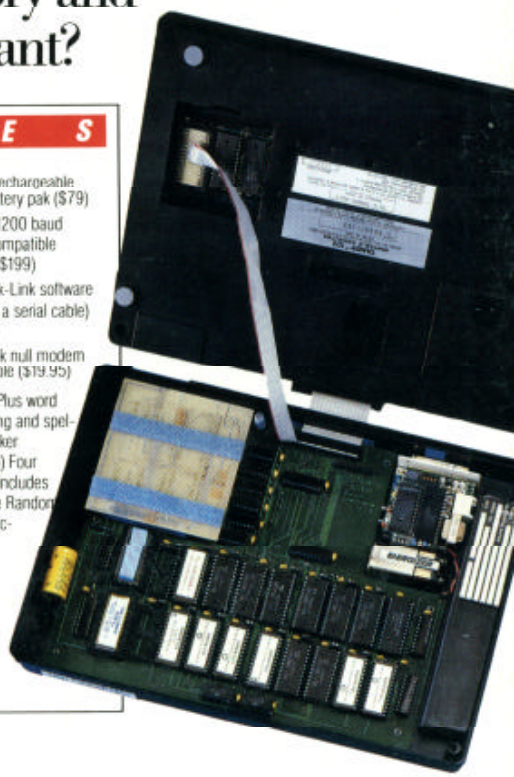
F E A T U R E S

Here's what you get in a standard BOOSTER PAK for only $429:

- 128K of RAM memory (96K available for files)
- 64K of ROM software
- 16 socketed slots for 32K RAM and/or ROM software chips
- 2 Tandy Molex ROM sockets
- Virtual RAM software (no bank switching or copying)
- File folders/subdirectories supported
- TS-DOS disk software built-in
- XMODEM file upload/downloading to virtual RAM
- Long-life lithium battery for RAM backup (rated at 7 years)
- BASIC programming access to virtual RAM
- ROM environment software (mix and match any ROM software)
- Ability to develop ROM software from 32-48UK in size
- Fast action Asteroids game (with great graphics) built-in

BOOSTER PAK Options

- 32K RAM (certified low power) chips ($20 ea.)
- 6 slot RAM expansion board ($68)
- 256K RAM expansion modules (requires expansion board) ($159 ea.)
- Internal rechargeable nicad battery pak ($79)
- Internal 1200 baud Hayes-compatible modem ($199)
- IBM Desk-Link software (requires a serial cable) ($39.95)
- Desk-Link null modem serial cable ($19.95)
- Sardine Plus word processing and spelling checker ($199.95) Four chip set includes complete Random House dictionary



Booster Pak™

To order ***BOOSTER PAK*** *call toll free*

1 800 343-808

Or write: **Traveling Softw**

TRAVELING SOFTWARE

19310 North Creek Parkw
Bothell, Washington 9801
206-483-8088

Circle 41 on Reader Service Card