

leased drive covers keeps them from being opened until the motor is off, minimizing the risk of disk damage.

While the drives themselves are quiet, the noise of the switching mechanisms used when moving from drive to drive might be disconcerting.

The amount of time a drive remains selected (and the jewel is illuminated) after a disk-oriented command has been completed might seem annoying—it's only six seconds. Again, the reason for having it that way is valid—on disk-to-disk operations or repetitive disk operations, speed may be gained by avoiding the start/stop/start cycle at the disk drive.

Eight-inch drives may be connected to the system, allowing the transferring of programs and data to the smaller media. By the time this is published, even more horizontal compatibility will have been added to the product.

Telcon may well be limiting its market by its emphasis on the Zorba as a programmer's machine. To this point Telcon has been successful enough to build a multimillion-dollar backlog of orders. But those orders are probably from individuals with assembly-language orientation, rather than the nonprogrammer whose interest is more in using it for accounting or other office duties.

In anticipation that you may wish to alter some of the CP/M software, Telcon has provided the source code for the CP/M input/output control system (which is referred to as the "Source of the Bios").

For the programmer familiar with Z-80 assembly language, this is a nice feature. Many users, however, couldn't care less. They would rather have the MBasic. MBasic is available for the machine but is not, at this writing, part of the marketed bundle.

Telcon is being nonspecific about its marketing plans for MBasic. For the majority of prospective purchasers, I would recommend MBasic over the "Source of the Bios," although in so doing I must declare a bias.

## Technical

System materials included with the Zorba include elements from Digital Research's CP/M 2.2, Microsoft's utilities and Basic, and Telcon-supplied programs and source code files.

### CP/M 2.2:

- ASM.COM—8080 Assembler, .HEX output.
- DDT.COM—.COM file debugger.
- ED.COM—Source File Editor.

- LOAD.COM—Generates .COM file from the .HEX file.
- MOVCPM.COM—Makes CP/M system image in memory.
- PIP.COM—File transfer utility.
- STAT.COM—File attributes and status ability.
- SUBMIT.COM—Batched command processor.
- XSUB.COM—Extension.
- DUMP.COM—ASCII file "hex dump" utility.

### Microsoft "Utilities" and Basic:

- M80.COM—8080 or Z-80 macro assembler, .REL output.
- L80.COM—Linker for .REL files, generates .COM files.
- CREF80.COM—Cross-reference utility for M80 assembler.
- LIB80.COM—.REL file librarian.
- CBAS2.COM—CBasic compiler language.
- CRVN2.COM—CBasic compiler language.
- XREF.COM—CBasic compiler language.

### Telcon-supplied Programs and Sources:

- SGEN.COM—SYSGEN.COM replacement; copies disk "system."
- PATCH.COM—Run to install system patches.
- PATCH.Z80—Source code for PATCH.COM.
- SETUP.COM—System configurator program.
- SETUP.Z80—Source code for SETUP.COM.
- FORMAT.COM—Universal disk formatter.
- FORMAT.Z80—Source code for FORMAT.COM.
- TRACKRD.COM—Special disk-reading program for engineers.
- TRACKRD.Z80—Source code for TRACKRD.COM.
- RELOAD.COM—Utility for Bios generation.
- SYSEQU.Z80—Part of Format, Setup and Patch source files.
- CPMBIOS.Z80—CP/M Bios source file.
- CPMEQU.Z80—Part of CP/M Bios source file.
- GEN.SUB—Submit procedure for Bios generation.
- GEN2.SUB—Part of GEN.SUB.
- KEY.MAC—Keyboard PROM contents; can be assembled.
- CPMBOOT.Z80—Monitor PROM contents; can be assembled.
- DISK.DOC—Explains files on disk.

The decision to provide so much source code is a reflection of Telcon's

assessment that its market is among engineers and assembly-language specialists. Again, if you ever need the capability, you at least have it—assuming that you know what to do with it.

## User Interface

The Zorba may be a technician's dream—and a user's nightmare. The machine is fast, but how fast does it need to be? The WordStar operator cannot type as fast as the machine can accept the text. The CalcStar operator has no great interest in speed. This machine's primary orientation is technical, unless Telcon recognizes that it has severely limited its market in this manner.

To the end they have chosen, Telcon has implemented CP/M 2.2 and has accompanied the machine with the usual ten pounds of technical literature. If the prospective nontechnical purchaser judges the machine on the strength of the documentation, Zorba will not make the sale.

The documentation, while appearing to be accurate and complete, is dry and dull. When the user is told that the application program is something that can be developed quickly or that it can take many years, bye-bye user. Take the novice user and place under his nose the intense documentation of the "innards" of CP/M, complete with Assembler instructions, memory maps, macro calls, Macro-80 coding and CBasic instruction, and he'll go buy something that he can simply turn on and use. Telcon must market its entire offering, not just part of it.

With the documentation presented for review, there was absolutely no instruction on WordStar, CalcStar or anything other than those things of interest to the technical person. Telcon must work on tutorial materials for all application packages. Despite its protestations to the contrary, CBasic is *not* that easily assimilated by a novice user. CBasic may mean "compiled" Basic, but we all know that it really means "Consultant's Basic."

Zorba and its other microcomputer-based products (Nomis-9, Nomis-12, GC-100, VCS-780 and a variety of other communications-based interface gear) will make an impact on the burgeoning portable-computer market. How large an impact it will make will depend on the focus applied by Telcon in the upcoming months.

Revamping the materials for the user, rather than the programmer, may make the difference. ■

# BASIC **VS.** JRT PASCAL:

A NO-HOLDS-BARRED COMPARISON.

**EASE-OF-USE** By dividing programs into modules, JRT Pascal makes even very complex programs—of nearly any size—a breeze to manage. Pascal code is *self-documenting*; program sections are identified by meaningful names, not line numbers. Error messages are verbal, not number codes. JRT offers 12 data types (to Basic's 2 or 3), and it has both regular and hex numbers.

**POWER** For power—the ability to write better, clearer programs, faster—Pascal is the run-away winner. Example: JRT simplifies programming by accomplishing complicated operations (for Basic) with one command:

<b>Basic</b>	<b>JRT Pascal</b>
IF A\$ = "V" OR A\$ = "W" OR A\$ = "X" OR A\$ = "Y" OR A\$ = "Z" THEN...	IF A IN ['V':Z] THEN...

**FLEXIBILITY** JRT's wide variety of data types reduces programming restrictions. And the data types are not all fixed in size. There are 3 looping statements (Basic has 1). With JRT, very large programs can be created and run, because program modules can be spread over many diskettes. Common modules can be used for several programs. Basic generally limits strings to 255 bytes; JRT strings go up to 64K.

**EFFICIENCY** Whereas Basic relies on a static, inefficient memory map to allocate storage, JRT's *dynamic storage* fills every available main storage area; there's no waste. With Basic, sub-routine modules must be linked together; with JRT, they can be linked—but don't have to be. JRT's more powerful commands run faster; typically, you'll write Pascal programs 3 to 10 times faster than in Basic. *Exclusive:* JRT lets you directly access the CP/M\* operating system for better total system control.

**NOW...** Consider our copy policy. (If you want to make copies, it's OK with us—so long as they're not for re-sale.) Check our astounding price: **\$29.95!**—and *satisfaction is guaranteed—or your money back.* Basic versus JRT Pascal: which comes out on top? Right! The coupon below is for your convenience. Or call. Today.

Here's the real shocker!		
Features	Basic	JRT Pascal
Structured programs	No	Yes
Separate compiled modules	"Chaining"	Structured procedures with auto-loading & purging
Arithmetic precision	Usually 6 or 7 digits	14 digits
Indexed files	No	Yes
Maximum string size	255 characters	64,000 characters
Loop statements	1	3
Data types	Usually 2 or 3	12
CASE statement	No	Yes
Introduced	1965	1980
Price	???	\$29.95!

Full support for indexed files

CRT screen formatting & full cursor control

Facilities for formatting printed reports

File variables & GET/PUT

Dynamic arrays

SEARCH procedures for fast table look-up

Extended CASE statements

Random files to 8 megabytes with variable length records

64K dynamic strings

Activity analyzer prints program use histogram

14 digit BCD FLOATING POINT arithmetic

True dynamic storage

Advanced assembly interface

Fast one-step compiler; no link needed

Efficient compiler needs only 85K diskette space

Maximum program size: more than 200,000 lines

More than 200 verbal error messages

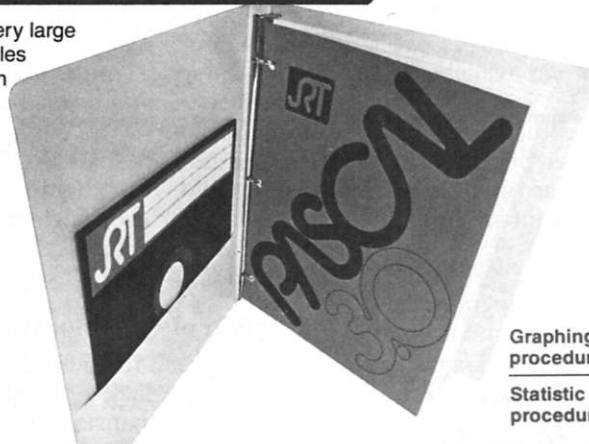
Separate compilation of auto-loading external procedures

No limits on procedure size, nesting or recursion

175-page user manual with 3-ring protective plastic binder & 5 1/4" or 8" diskettes

Handy JRT Pascal reference card

Graphing procedures  
Statistic procedures



THE COMPLETE PASCAL FOR CP/M.

# \$29.95!

JRT/PASCAL 3.0

Send **JRT SYSTEMS** or  
to **45 Camino Alto/G2** phone **415/388-0530**  
**Mill Valley, CA 94941**

Here's my \$29.95; please send me JRT Pascal. I understand that if I'm not completely satisfied, I can return it within 30 days—with the sealed diskettes unopened—for a full refund.

I need the 5-1/4" diskettes for  Apple CP/M;  Heath, Hard Sector;  Heath, Soft Sector;  Northstar;  Osborne;  Superbrain;  Televideo;  Xerox 820. I need  8" SSSD diskettes.

Name \_\_\_\_\_

Address \_\_\_\_\_

City \_\_\_\_\_ State \_\_\_\_\_ Zip \_\_\_\_\_

Check  C.O.D.  MasterCard  VISA  
(CA residents add sales tax. Add \$6 for shipping outside North America.)

Card # \_\_\_\_\_ Exp. \_\_\_\_\_

Signature \_\_\_\_\_

\*CP/M is a Digital Research TM. A 56K CP/M system is required.

---

# A Real-Time Compiler System

*The sometimes difficult relationship between microcomputers and compilers may soon improve. This hardware/software approach should work with nearly all languages and computer systems.*

By Martin Reiffin

This is a new real-time compiler system for microcomputers.

By "real-time," I mean that the compiler analyzes and translates source code into machine code concurrently as the programmer types in the program at the keyboard. The idea is to eliminate much of the drudgery of programming in a compiled language, such as Pascal, C, Fortran or PL/I.

The following description explains the theory of the system and how it can be implemented on any micro using a microprocessor with interrupt capability. (The system status is patent-pending; it is not yet available to the end-user market.)

I have implemented the system on two S-100 computers: A Godbout system and a CCS/Cromemco system, using the keyboard interrupt scheme.

## Background

Since humans write programs in programming language and computers execute only machine language, it's often necessary to translate from one language to the other.

When the programming language is high-level (that is, abstract in the sense that it does not explicitly manipulate the computer registers and other hardware), the translation of the original program is performed by other programs, called compilers or interpreters. The original program is called the source code, and the resulting program translation is called the object code.

In addition to translation, the compiler also must perform lexical, syntactic and semantic analyses of the source code.

Lexical analysis is performed by a scanner and is the process by which a sequence of source code bytes are formed into meaningful symbols or tokens—in the same way that a sequence of characters is formed into

English words. These symbols are then analyzed by a parser to determine if they are arranged in a relation that conforms to the rigid grammatical rules, or syntax, of the programming language.

The semantic analysis determines if the symbols conform to still other rules not conveniently expressed by the language grammar.

Performing these analyses is comparable to parsing the words of an English sentence. If the sequence of symbols violates a syntactic or semantic rule, an error has been committed and the compiler must inform the programmer with an error message.

After translation, the compiled object code usually is linked and loaded—joined with other object code modules to form a complete machine-code program that can be executed by the computer.

## Description of the Prior Art

In recent years, the increase in software costs, the lack of skilled programmers, the expansion of the computer market and the under-utilization of much available hardware has hastened the adoption of high-level languages and concentrated efforts to make their use more efficient.

However, programming in a high-level language is still slow, tedious and inefficient. For example, a compiled language requires a repeated sequence of steps: loading the editor, writing or editing the source code, loading the compiler, executing the compiler, loading the linker, executing the linker, running the program and repeating the sequence when errors appear.

During much of this time, the programmer waits for completion of the loading or execution steps, and this waiting is both wasteful and boring. As a result, the programming process

is slow and expensive.

It is generally accepted that the output of the average professional programmer is only about five to ten lines of debugged source code per day.

The recent widespread use of microcomputers has compounded the programming problem. Most users presently writing programs for micros are not professionally trained as programmers. They are unwilling to spend time and effort programming in a compiled language with present microcomputer systems.

Instead, the majority of personal computer programmers use Basic interpreters.

A Basic interpreter is syntactically sparse, cryptic in revealing errors, incapable of using local variables, incapable of passing parameters to subroutines, unable to invoke subroutines by name, incapable of linking library modules and lacking in both data structures and flow-control structures. This makes it difficult to write error-free programs for any but the simplest applications. Basic, therefore, is regarded as a poor vehicle for learning good programming technique.

These Basic interpreters are also too slow for many applications.

Nevertheless, the mechanics of compilation with present microcomputer systems are so inconvenient that Basic interpreters dominate the field.

## Prior Schemes

Numerous attempts have been made to minimize the disadvantages of conventional compiler usage. One such scheme is the so-called "incremental compiler." As each line of source code is entered at the console, it is analyzed for syntax without consideration of

---

*Address correspondence to Martin Reiffin, 27 E Gate Road, Danbury CT 06810.*

---

the context of the entire program.

If the line is error-free, the programmer can enter the next line of code. Otherwise, an error message is displayed and the error must be corrected before further lines are entered. After the entire program is entered, it undergoes further syntactic and semantic analyses with respect to the context-dependent rules, after which code generation and execution may take place.

The incremental compiler scheme has some merit when used with those languages that have few context-dependent restraints, such as Basic. For modern structured languages, such as Algol, Pascal, PL/I, C and Ada, the limited local analysis that can be performed after entry of each line is a relatively small portion of the total analysis required and is not worth the overhead.

Another well-known scheme is "repeated recompilation." In this approach, the syntactic and semantic analyses after entry of each line include all context-dependent rules and consider the entirety of the partial program entered to that point.

Therefore, upon editing even a single byte of the source code, the entire source file must be recompiled from the beginning. Since this recompilation must be completed before new lines may be entered, the large overhead of this scheme prohibits its use with any but the smallest programs.

In an effort to avoid the overhead of complete recompilation, a number of schemes of partial recompilation have been devised. These schemes generally involve a complex data structure for the source code and/or a complex editor.

Programmers having sufficient expertise in the particular scheme being used may make a change in the source code requiring only a recompilation of the changed portion. These approaches require an excessive amount of computer memory and/or highly specialized skill of the programmer.

These schemes have had no use except as experimental curiosities in academic research. Commercially available compilers still require the repeated sequences of the load, edit, load, compile, load, link and run steps as described above.

## Object of the Invention

The ideal computer architecture for programming should have the following characteristics:

- The compilation should be performed in real-time as the source code is entered or edited by the programmer.

- The system should be capable of implementing a modern block-structured language having a powerful syntax, such as Algol, C, Pascal or PL/I.

- There should be no waiting for disk accesses during program writing, editing or compilation, except to the extent necessary for initial entry or saving of source files.

- An error message should be displayed instantly after a syntax error is entered at the console.

- Correction of source code errors should be fast and easy, without reloading the editor, source file or compiler.

- Compilation should be finished almost instantly after the last line of an error-free source program is entered.

- The compilation should be complete in that no further syntactic or semantic analysis is required, and the absence of an error message should assure that the program has no errors.

- The programmer should be able to stop execution of the object code at any time, examine the values of all

variables and continue execution, all without requiring the prior insertion of write statements, breakpoints or other debugging code into the source program.

- The architecture should not impose any additional requirements of unique language syntax, complex editor mechanics or scheme-specialized programmer expertise.

The primary object of my invention is to provide a novel computer system that closely approaches this ideal architecture.

## Summary of the Invention

The following is a summary of a preferred embodiment of my invention.

The programmer invokes the real-time compiler-editor system by typing its command file name at the keyboard console. The command file containing the software portion of the system is then read into memory from a disk. The source buffer—a memory region that will contain the source code—is initialized so that its first stored byte is a predetermined code, which will be called a Pause Mark.

Execution of the compiler then

**GET IBM-PC Capacity at a Fraction of IBM'S Price!**

**Now NETRONICS 16 Bit EXPLORER 88-PC Kit**  
Starts at Just ~~\$399.95~~ - Accepts All IBM Peripherals.

**It's true! Now you can enjoy the power of the Intel 8088—the same microprocessor which powers the IBM-PC—and run any program compiled for the IBM-PC...starting at only \$399.95!**

Take this easy, low cost way to learn 16-bit technology! Two-board system features:

1. 8088 mother board with 5-slot expansion bus; accepts any hardware designed for IBM-PC; and
2. 64K memory board, expandable to 256K; with IBM compatible RS232 communications port.

Any disk-operating system which works on the IBM will work directly on the EXPLORER 88-PC, and all programs compiled for the IBM will run on it.

The system monitor ROM included in the Starter system features a user-friendly operating system that allows easy program generation and debugging. The commands include display/modify memory...display/modify registers...input/output data to 1/0 ports...block moves...single-step trace mode...go/run with optional breakpoint and register reports...cassette load/save with file labels...plus a complete system test program that tests and reports condition of ROM, RAM, cassette interface, timer, DMA controller, interrupt controller, and the communications port. These test programs not only allow easy debugging of software but they serve as hardware and software learning tools.

The EXPLORER 88-PC STARTER KIT includes a mother board, memory/I/O board, all components needed, sockets for IC's used, one 62-pin bus connector and complete assembly/test instructions. All you need is a soldering iron, solder, a power supply, and a standard RS232 terminal (Netronics has 2 low-cost ones to choose from).

Explorer 88-PC Starter Kit...\$399.95  
+ 10.00 p&i  
 (wired & tested, add \$100.00)  
 Extra 62-pin connectors at \$4.25 ea.  
+ 1.00 p&i.

Use your own terminal with the EXPLORER 88-PC or, if you plan to expand it to be fully IBM compatible, we offer our IBM compatible keyboard and an IBM compatible color graphics board (only available wired and tested).

- IBM compatible keyboard...\$299.95 + 10.00 p&i.
- IBM compatible color board...\$299.95 + 10.00 p&i.
- Additional ROM required...\$35.00.

Set your own pace! Invest and learn, at the rate YOU want! Add to your EXPLORER 88-PC:

- Deluxe heavy-duty steel cabinet that houses either two 5 1/4" floppies or a 5 1/4" hard disk with one floppy. This cabinet features a brushed finish front panel and a wood-grained sleeve.
- EXPLORER 88-PC Cabinet...\$199.95 + 18.00 p&i.

A heavy-duty open frame power supply with fan that can be used in your own cabinet or installed into the Netronics cabinet is available as follows:

- 10 amp power supply for system + 2 floppies...\$149.95 + 8.00 p&i.
- As above + extra power for 1 hard disk...\$169.95 + 8.00 p&i.
- IBM compatible disk controller board. Controls four 5 1/4" floppy drives (w/2 drive cable). Available wired and tested only...\$250.00 + 8.00 p&i.
- Monitors and BIOS source listings: available on either disk or hard copy at \$35.00. Please specify format and system required.
- INTEL 8086/8088 user manual...\$15.00 + 1.50 p&i.
- THE 8086 BOOK by RECTOR & ALEX...\$16.00 + 1.50 p&i.

Over 100 EXCLUSIVE Products and Kits—including the "Speak Easy" universal voice synthesizer, a Diagnostic card with built-in logic probe for the IBM-PC, terminals, monitors, the ELF and EXPLORER 85 computers, and much more, are described in our upcoming catalog. It's yours FREE if you check here

For Canadian orders please double the amount of p&i shown. IBM-PC is a registered trademark of IBM Corporation.

"p&i" stands for "postage and insurance".

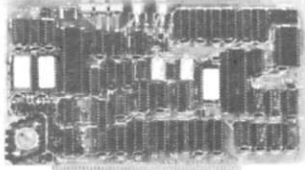
**CALL TOLL FREE 1-800-243-7428 for Charge Card Orders.**  
In Conn., call 203-354-9375. Conn. res. add sales tax.

TO ORDER BY MAIL, CHECK BOXES FOR PRODUCTS DESIRED AND MAIL ENTIRE AD TO:

**NETRONICS R & D LTD.**  
333 Litchfield Rd., New Milford, CT 06776

Amount enclosed OR  Charge my  VISA  MASTERCARD  
Acct. No. \_\_\_\_\_ Exp. Date \_\_\_\_\_  
Signature \_\_\_\_\_  
PRINT NAME \_\_\_\_\_  
Address \_\_\_\_\_ City \_\_\_\_\_ State \_\_\_\_\_ Zip \_\_\_\_\_

**80 CHARACTER VIDEO BOARD - S-100**



- All This on ONE BOARD:
- Keyboard port with TYPE-AHEAD buffer
  - 8275 CRT controller with light pen port
  - Two 2716's - program & character rom's
  - Optional 2716 for CHARACTER GRAPHICS
  - All screen & keyboard ram
  - SIMULTANEOUS I/O or Memory mapped
  - Z-80 MPU - 2 or 4 Mhz system clock
  - Easy to adapt Software
  - Uses only EASY-TO-GET parts
  - Use in any S-100 system
  - 696 Bus Compliance: D8 M16 I8 T200
  - Build for less than \$200

Now includes crystal & heat sink.

**Introducing The VDB-A**

Bare board with Documentation **\$49.50**  
 + \$2.50 S&H (Ill. res add 6% tax)  
 Add 3% for Mastercard and Visa

*Simpliway* PRODUCTS CO.

P.O. Box 601, Hoffman Estates, IL 60195  
 312/359-7337



With every box of 3M or Nashua diskettes you will receive **FREE** a durable plastic library case for convenient storage of your diskettes.

Order **NOW**. Offer good for a limited time.

Specify 3M or Nashua.

Both quality disks... Both competitively priced.

**5 1/4"** Specify Soft 10 or 16 Sector price/box of 10\*  
 1 side sngl/dbl density ..... \$26.00/box  
 2 sides dbl density ..... \$34.00/box

**8"** Specify Soft or 32 Sector  
 1 side sngl density ..... \$28.00  
 1 side dbl density ..... \$32.00

Call or write for complete list of other products avail.

\*NO HIDDEN SHIPPING CHARGES  
 All orders shipped freight prepaid

**KEYSTONE INFORMATION PRODUCTS**  
 10 Bloody Brook Road, Amherst, NH 03031  
 (603) 673-2287

Checks VISA MC VISA



**EXPLORING CP/M**

Explore the mazes of the CP/M operating system with this menu-driven 8-inch disk which allows you to (1) look into every hidden cranny of a CP/M disk; (2) change any ASCII or hex byte; (3) recover erased files; (4) create "autoload" disks; (5) find and lock out bad sectors; (6) reconstruct files from crashed disks. And more! \$39.95.

Hard-copy instructions included. For recommended parallel reading: CP/M PRIMER by Murtha and Waite, \$14.95. Send check for prompt shipment. Add \$2.25 per order for shipping. Return for refund if not satisfied. CP/M is a registered trademark of Digital Research, Inc. Requires two drives, 32K.

**TELEPRINT, INC.**

P.O. Box 10A, Sylvania GA 30467

begins by reading the Pause Mark as the character in the first location of the buffer. When the compiler reads the Pause Mark, it will enter an infinite loop, repeatedly reading the same location until the content of this location is changed by the editor to a blank space.

When the programmer strikes a key on the console keyboard, the central processor unit (CPU) executes the following interrupt sequence (described for an 8080 or Z-80 microprocessor): upon completion of the instruction currently being executed, the processor enters the interrupt mode and communicates its new status to the system by emitting an interrupt acknowledge signal.

Upon receipt of this signal, the interrupt hardware gates an RST instruction onto the data bus. The processor then executes the RST instruction, which is a one-byte call to a selected location in low memory where a jump ("vector") to the interrupt service routine comprising the editor is stored.

The interrupt service routine first saves the stack pointer and other CPU registers. If the struck key corresponds to an alphanumeric or other noncontrol character, it is placed into the second location of the source code buffer immediately after the Pause Mark. The buffer pointer is then advanced to the next location, the CPU registers are restored, the CPU interrupt is enabled and the Ret instruction is executed to return control to the compiler.

The compiler continues to execute its infinite loop in which it repeatedly reads the Pause Mark character in the first location of the source code buffer. This sequence is repeated as the programmer strikes additional keys at the keyboard; the successive characters are entered into successive locations in the source code buffer as the buffer pointer advances. This sequence continues until a key corresponding to a control character is struck.

If this control character is a carriage return, the corresponding ASCII code (13) is inserted into the buffer, the buffer pointer is advanced, the Pause Mark code is then inserted into the buffer location adjacent to the carriage return code, and the original Pause Mark code in the first location is replaced by the ASCII code (32) for a blank space.

The Pause Mark location thus has been advanced from its original point to the end of the first line of the

source code.

Upon return to the compiler from the interrupt service routine, the compiler pointer accesses the first buffer location and reads the code for a space instead of the Pause Mark. The compiler then will advance its pointer repeatedly to the next buffer location and perform its lexical, syntactic and semantic analyses on the first line of source code stored in the buffer.

The compiler may either display an error message or emit compiled object code, as is appropriate, until the compiler pointer reaches the new Pause Mark inserted at the end of the first line of source code.

When it reaches the new Pause Mark, the compiler again enters an infinite loop, without advancing the pointer, until the editor eventually moves the Pause Mark to the end of the next line, whereupon the compiler is free to compile this next line of source code.

Control characters other than a carriage return may be entered by striking appropriate keys to perform the conventional editing functions of a screen editor. For example, errors in the present line of source code may be corrected by moving the cursor backward. This does not affect the compiler, which cannot advance beyond the Pause Mark at the end of the previous line.

However, if by hitting the appropriate control key the cursor is moved up one or more lines to a position before the Pause Mark, a Recompile Flag is set so as to enter a recompile mode. In this event, upon return to the compiler, the latter is reinitialized so that it may recompile the source code from the very beginning of the source buffer. Subsequent editing or text insertions cause the editor to move the Pause Mark to an updated location, adjacent to the end of the line preceding the most recently edited line.

When the compiler finds a syntax error in the source code, it displays an error message. The programmer may then edit the source so as to correct the error. Upon return from the editor, the compiler is reinitialized to recompile the source code.

The entered source code and emitted object code preferably are stored in memory so that disk accesses will not unduly interfere with the editing and compilation processes. If the source or object code buffer gets filled, its contents may be stored in a disk file in the conventional manner employed by editors and word processors such

# D&N MICRO PRODUCTS, INC.

3702 N. Wells St.  
Fort Wayne, Ind. 46808  
(219) 484-6414

TERMS \$3.00 shipping, Foreign orders add 15%, Indiana residents add 5% sales tax.

## COMPUTER

### MICRO-80 COMPUTER

Z-80A CPU with 4Mhz clock and CP/M 2.2 operating system. 64K low power static memory. Centronics parallel printer port. 3 serial ports. 4" cooling fan. Two 8" single or double sided floppy disk drives. IBM single density 3740 format for 243K or storage, double density format for 604K of storage. Double sided drives allow 1.2 meg on each drive. Satin finish extruded aluminum with vinyl woodgrain decorative finish. 8 slot backplane, 48 pin buss compatible with OSI boards.

**MODEL 80-1200** \$2995

2 8" Single sided drives

**MODEL 80-2400** \$3495

2 8" Double sided drives

### MICRO-65 COMPUTER

6502 CPU with 2Mhz clock and DOS-65 operating system. 48K of low power static memory. 2 serial ports and 1 Centronics parallel port. 2 8" single or double sided drives. Satin finish extruded aluminum with vinyl woodgrain finish. 8 slot backplane, 48 pin buss compatible with OSI. Will run OSI 65D and 65U software. Includes Basic E/65 a compiled BASIC for 6502 CPU.

**MODEL 65-1** \$2995

2 8" Single sided drives

**MODEL 65-2** \$3495

2 8" Double sided drives

**BP-580 8 Slot Backplane** . . . . \$ 47

OSI 48 pin Buss compatible

### MEM-CM9 MEMORY/ FLOPPY CONTROLLER

24K memory/floppy controller card uses 2114 memory chips, 1 8K and 1 16K partition. Supports OSI type disk interface

**24MEM-CM9** . . . . . \$325

**16MEM-CM9** . . . . . \$260

**8MEM-CM9** . . . . . \$180

**BARE MEM-CM9** . . . . . \$ 50

Controller on assembled unit

add. . . . . \$ 90

**BIO-1600 Bare IO card** . . . . . \$ 50

Supports 8K of memory, 2 16 bit parallel ports, 5 serial ports, with manual and Molex connectors.

## PRINTERS

### Okidata

**ML82A**, 120 cps, 10" . \$409

**ML83A**, 120 cps, 15" . \$895

**ML84 Parallel**, 200 caps, 15" . \$1150

### C. Ioth

**8510AP** Prowriter, parallel . . . \$419

120 cps, correspondence quality

**8510APD** Prowriter, serial . . . \$585

**F10-40PU** Starwriter, parallel \$1319

Letter quality daisy wheel

**F10-40RU** Starwriter, serial . \$1319

**F10-55PU** Printmaster . . . . \$1610

parallel, Letter quality daisy

wheel

**F10-55RU** Printmaster, serial \$1610

### DISK DRIVES AND CABLES

**8" Shugart SA801** . . . . . \$385

single sided

**8" Shugart SA851** . . . . . \$585

double sided

**FLC-66** ft cable from D&N . . . \$69

or OSI disk controller to 8" drive

**5 1/4" MPI B51** disk drive with . \$450

cable, power supply and

cabinet. Specify computer type.

**FLC-5 1/4** cable for connection . \$75

to 5 1/4 drive and D&N or OSI

controller, with data separator

and disk switch. Specify

computer type

## HARDWARE

### OSI COMPATIBLE

**IO-CA10X Serial Printer Port** . \$125

Specify Device #3 or #8

**IO-CA9 Parallel Printer Port** . \$150

### CMOS-MEM

64K CMOS static memory board, uses 6116 chips, 3 16K, 1 8K and 2 4K blocks, Partitionable for multi-user, OSI type disk controller, 2 IO mapped serial ports for use with D&N-80 CPU. Ideal way to upgrade from cassette to disk.

**64K CMOS-MEM** . . . . . \$500

**48K CMOS-MEM** . . . . . \$405

**24K CMOS-MEM** . . . . . \$260

**16K CMOS-MEM** . . . . . \$210

**BARE CMOS-MEM** . . . . . \$ 50

Controller . . . . . add. \$ 90

2 IO mapped serial ports . . . . \$125

on assembled memory board

**Z80-IO** 2 IO mapped serial . . . \$160

ports for use with D&N-80 CPU

card

**FL470 Disk Controller** . . . . . \$155

Specify 5 1/4 or 8" drive



## STANDARD CP/M FOR OSI

### D&N-80 CPU CARD

The D&N-80 CPU allows the owner of an OSI static memory computer to convert to Industrial Standard IBM 3740 single density disk format and CP/M operating system. Double density disk operation is also supported for 608K of storage on an 8" diskette. When used with a 5 1/4" disk system 200K of storage is provided. Optional parallel printer and real time clock. Also available for polled keyboard and video systems. Compatible with C2, C3, C4 and 200 series OSI computers.

### INCLUDES CP/M 2.2

**D&N-80-1** Serial 8" disk \$595

**D&N-80-2** Video 5 1/4" disk \$595

**D&N-80-3** Video 8" disk \$595

**Option 001** \$ 60

Parallel printer and  
real time clock.

**HARD DISK DRIVER** \$140

Allows D&N-80 CPU board to control OSI 40 or 80 meg hard disk unit. Will not destroy OSI files. Will also allow for a true 56K CP/M system. Specify 40 or 80 meg drive.

**BUS TRANSFER** \$135

Allows for D&N-80 and OSI CPU to be in the computer at the same time. Toggle switch provides for alternate CPU operation.

**DISK TRANSFER** \$100

Utility program to transfer OSI CP/M format disk to IBM 3740 single density format. Will also transfer IBM to OSI format.

### SYSTEM HARDWARE

#### REQUIREMENTS

D&N-80 CPU, D&N FL470 or OSI 470 controller, 48K memory at D000-DFFF, 4K memory at D000-DFFF, two disk drive cables.

**FORMAT TRANSFER** \$15

You supply software on 8" diskette D&N will transfer OSI CP/M format to IBM 3740 CP/M format. Can also transfer IBM 3740 CP/M format to OSI CP/M format. Original diskette returned.

as CP/M ED and WordStar. The use of bank-select memory schemes or the advent of 16-bit microprocessors—with their larger addressable memory space—will obviate the need for disk storage until the compilation is finished.

### Detailed Description

The details of this invention are illustrative of one of the many forms it may take in practice. The invention and novelty reside in neither the hardware nor the software taken separately, but rather in combination of both.

The major hardware components of the overall system are shown in Fig. 1.

The CRT Console refers to any suitable terminal having a keyboard for entry of the source code to be compiled and for entry of editing commands to change the code. The terminal also comprises a video display for implementation of a screen editor. The keyboard is preferably integral with the video display, forming a unitary console that has an RS-232C serial link to the remainder of the system.

This serial link is connected to the Input Port, which is preferably embodied as a UART (such as the 1602, AY-5-1013 or TMS 5501). Each keystroke on the keyboard of the CRT Console results in the serial transmission to the UART of a train of bits constituting the ASCII byte corresponding to the struck key.

The UART reforms the bits into that byte, which is transmitted in parallel on the data bus to the accumulator of

the CPU. The UART also provides an output port.

Execution of an Out command by the CPU results in the transmission on the data bus of a byte from the accumulator to the UART, which then may serially transmit the byte to the CRT Console for display on the video screen.

In the usual operating mode of a conventional microcomputer system, the status of the input port is repeatedly tested by the CPU in a polling loop until the input port status indicates that a byte of data has been received and is available in the UART-received data register.

My invention instead employs an interrupt mode of operation, whereby the CPU normally executes the compiler until the UART receives a byte from the CRT Console.

The "data available" line of the UART is then activated; this in turn activates the Interrupt Controller to cause the CPU to execute the editor. Upon entry of the received character into the Source Buffer in main memory, or upon completion of an editing command, a Ret instruction is executed by the CPU to cause it to resume its execution of the Compiler from the point where it was interrupted.

As the Compiler is executed, it should perform lexical, syntactic and semantic analyses of the program source code stored in the Source Buffer. The Compiler also emits object

code and stores it in the Object Buffer. Upon completion of entry and compilation of the source code program, control of the CPU may be passed to the Interpreter for execution of the object code if the latter is in the form of intermediate code.

The programmer may be given the option of saving the source code and/or object code in secondary storage, such as disk or tape media. Instead of generating intermediate code (p-code) for interpretation, the Compiler may be of the type that emits executable machine code.

The Compiler may require only a single pass through the source code, in the manner of the usual recursive-descent Pascal compiler. If the Compiler requires more than one pass, the first pass should perform the syntax analysis so as to reveal all syntax errors.

The interrupt facility enables the programmer to stop execution of the machine-code program at any time, examine the values of the variables and then continue execution. No additional hardware is required for this extra function, and the extra software is minimal.

### Circuitry and Hardware

Fig. 2 refers to the circuitry and hardware components directly involved in the interrupt operation. Upon striking a key, a train of pulses constituting the byte corresponding to the struck key is emitted from the RS-232C serial port O.

A converter gate, C1, converts the pulse train from RS-232C levels to TTL (transistor-transistor-logic) levels to match the requirements of input port IN of the UART. The latter forms the serial pulse train into an eight-bit byte, which is stored in the received data register of the UART. The latter then outputs a data-available signal at pin DAV, whose signal is transmitted by gate G1 to a vectored interrupt input VI of the Priority Encoder.

Although only one input pin VI of the latter is shown, this chip has vectored interrupt input pins to which other interrupting devices may be connected.

The Priority Encoder arbitrates competing interrupt requests at its inputs and determines the request having the highest priority. Its enable input EI is grounded as shown (Fig. 2).

Assuming that the interrupt request from the Console and the UART win the priority contest, the Encoder then transmits a three-bit code, A0,A1,A2,

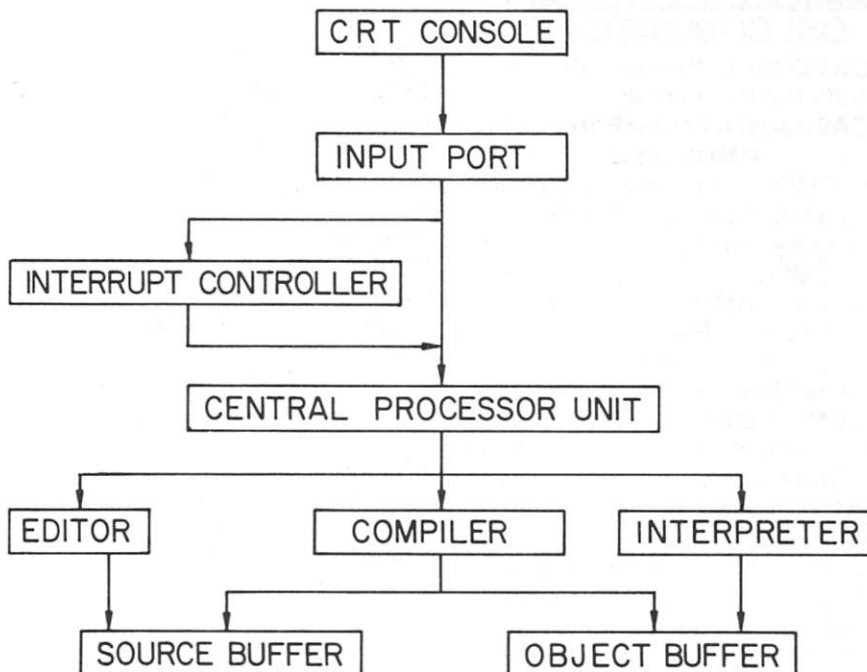


Fig. 1. Schematic diagram showing relation of the major hardware components constituting a preferred embodiment of the computer system in accordance with this invention.

to the respective inputs of the Interrupt Vector Register.

The other five inputs of the latter are held positive by potential source +V, so that the resulting byte input to this register chip constitutes an RST call instruction. The signal at output pin GS of the Priority Encoder is transmitted by gate G2 to the latch enable input LE of the Interrupt Vector Register. This causes the latter to latch the RST call instruction into its internal flip-flops.

Activation of output pin GS of the Priority Encoder also transmits an interrupt signal through AND gate A1 to the interrupt request pin INT\* of the Z-80 CPU. Upon completion of the present instruction (and assuming that the interrupt of the processor is enabled), the CPU's status pins IORQ\* and M1\* are activated and their signals are transmitted by gates G3, G4 to AND gate A2 to form the INTA (interrupt acknowledge) signal. INTA is inverted by gate G5 and fed to the output enable pin OE of the Interrupt Vector Register, whereupon the RST call instruction at the inputs of the latter is jammed onto the data bus.

The RST instruction is then input to and executed by the Z-80, causing the CPU to push the contents of the program counter onto the stack, and further causing the CPU to jump to a predetermined location in low memory. This location stores a vector, or three-byte jump instruction, to an interrupt service routine.

The interrupt service routine in-

cludes the editor, as well as a subroutine, to store the contents of the CPU registers. Control of the CPU is then retained by the editor until either a character has been entered into the source code buffer or an editing operation has been completed.

The editor includes an input instruction that, when executed, causes the CPU to place the address of the UART's port on the Address Bus. This address is tested by the Comparator, and if it matches that of the port, the output pin EOut is activated to signal the Decoder.

The Decoder is controlled by other control and status signals (not shown) in the conventional manner so as to transmit a signal RDE\* to the corresponding input RDE of the UART. The byte in the received data register (not shown) of the UART is then gated onto the data bus and transmitted to the accumulator within the CPU.

In the embodiment of the invention shown in Fig. 2, I chose to use the following integrated circuits:

UART:	1602
Priority Encoder:	74LS148
Interrupt Vector Register:	74LS373
Decoder:	74LS155
Comparator:	25LS2521
C1:	1489
C2:	1488

### System Sequence

Fig. 3 shows the sequence of operation of the overall system. The Compiler normally has control of the CPU

and either is in an infinite loop upon reaching a Pause Mark in the source code buffer or is in the process of analyzing the source code in the buffer.

The occurrence of a Keystroke at the terminal causes an Interrupt, whereupon the CPU is vectored to the interrupt service routine. The latter includes a subroutine to perform the Save Registers procedure shown in the drawing.

The Editor is then executed by the CPU. If the Keystroke corresponds to a control character, then an editing procedure, such as a cursor movement, screen scroll, character deletion or line deletion, is performed. If the Keystroke corresponds to an alphanumeric character or other valid source-code character, the latter is entered into the source code buffer and displayed on the video screen and the screen cursor is advanced to the next character position.

The interrupt service routine then jumps to its subroutine to perform the Restore Registers procedure, whereby the registers of the CPU are restored to their original values at the instant of the interrupt.

The Enable Interrupt instruction

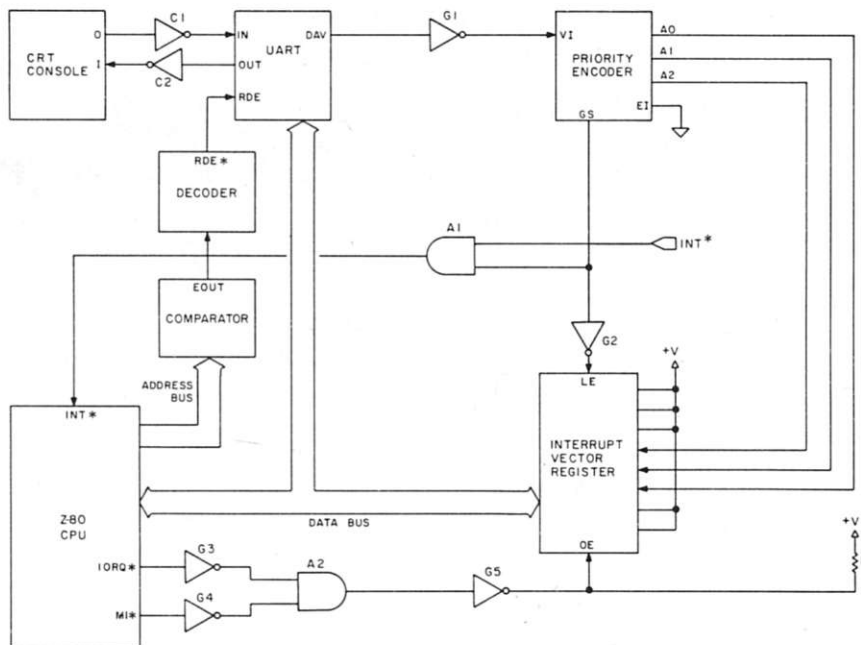


Fig. 2. Diagram showing interrupt logic and circuitry of the system hardware.

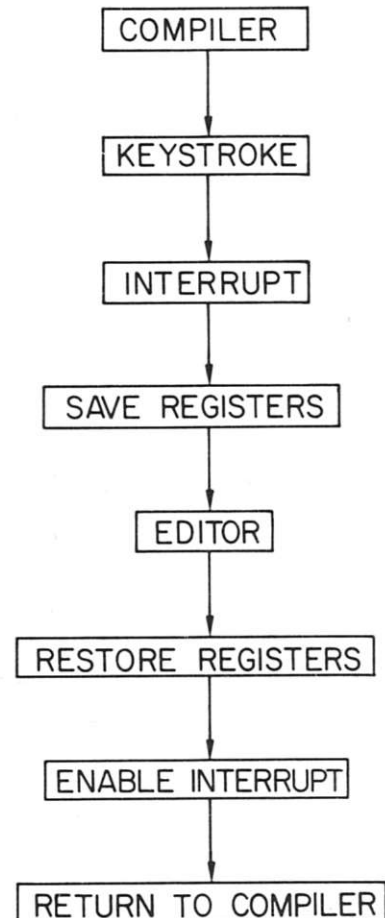


Fig. 3. Flowchart showing sequence of operations.



(EI) is then executed by the CPU so that the latter may respond to the next interrupt. Finally, the RET instruction is executed so that the CPU may return to the Compiler. The Compiler then resumes execution from the point where it was interrupted.

Fig. 4 shows the sequence of operations of the Compiler. After initialization, the Compiler performs its Read Character function whereby the byte in the first location of the source code buffer is read.

If this byte is the predetermined code designated as the Pause Mark, the Compiler pointer does not advance and the Compiler enters an infinite loop. It continues to read the same location until the content of this location is changed by the editor to a blank. When this change occurs, the Compiler memory pointer is incremented to the next location of the buffer so that the Compiler exits from its Pause loop, as indicated by the legend Advance Memory Pointer.

As indicated by Symbol?, the lexical analyzer of the Compiler then determines if the source character read-in constitutes the last character of a symbol, such as an identifier, operator or punctuation mark. If not, the Read Character function is executed again until a symbol is recognized.

The syntax analyzer of the Compiler then determines whether this symbol conforms to Correct Syntax in accordance with the grammar of the programming language. If not, an Error Message is displayed.

If the syntax is correct, the Read Character function is repeated until an error is found or until End Of Program is reached. In this event, the Code Generator may be invoked if this function is performed as a separate pass.

Alternatively, code generation may be performed concurrently with the lexical and syntactic analyses. The generated code then may be saved on disk and/or executed at the option of the programmer, as indicated by the legend Save/Execute Object Code.

For clarity in illustration, I'll show how the simple and widely published compiler PL/0 of Prof. N. Wirth ("Algorithms + Data Structures = Programs," Prentice-Hall, Inc., 1976, pp. 280-347) may be modified for implementation in the present invention. In the description below, the following identifiers have been added and do not appear in the original PL/0 compiler as published: CONT, PEEK, RECOMPILE, PTR, PM and SP.

The first statement in the modified compiler is:

IF NOT CONT THEN

The Boolean variable CONT is False upon initial entry into the compiler, signifying that this is not a continuation of a previous execution. That is, the compiler has just been entered for the first time during the present session.

The subsequent assignment statements therefore are executed to initialize the contents of the arrays WORD, WSYM, SSYM, MNEMONIC, DECLBEGSYS, STATBEGSYS and FACBEGSYS shown on pp. 346 and 347 of the Wirth treatise.

The values of these arrays remain fixed throughout execution of the compiler and the above conditional

IF statement obviates the need to reexecute all of these assignment statements upon subsequent reinitializations of the compiler for recompilations. That is, after the first test of the variable CONT, it is set equal to True so as to bypass the assignment statements thereafter when recompilation is required.

After the conditional block of array assignments, a Peek assembly-language function is invoked to read the content of the memory location immediately preceding the start of the source code buffer (the location of the recompile flag). If this location contains the ASCII code for the letter R, then a compiler procedure Recompile is invoked to reinitialize the variables ERR, CC, CX and LL, and to assign the value of constant AL (10) to the variable KK.

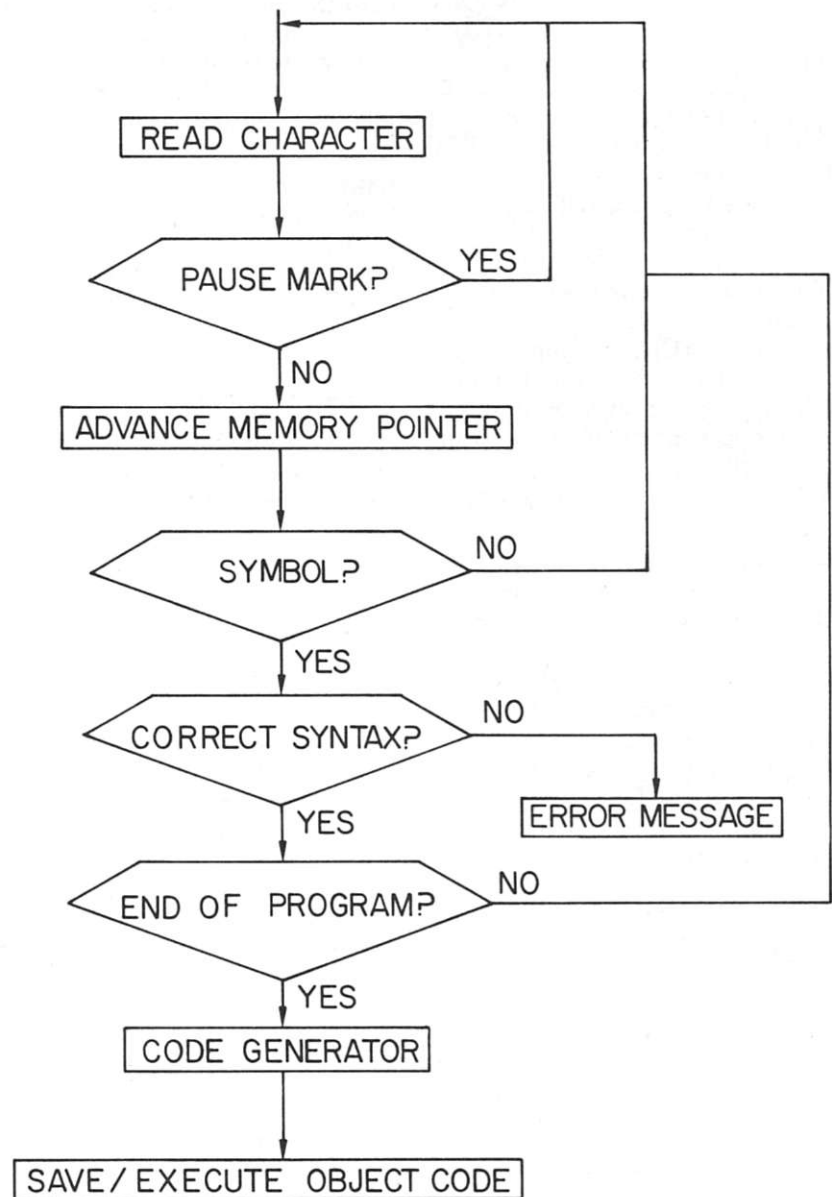
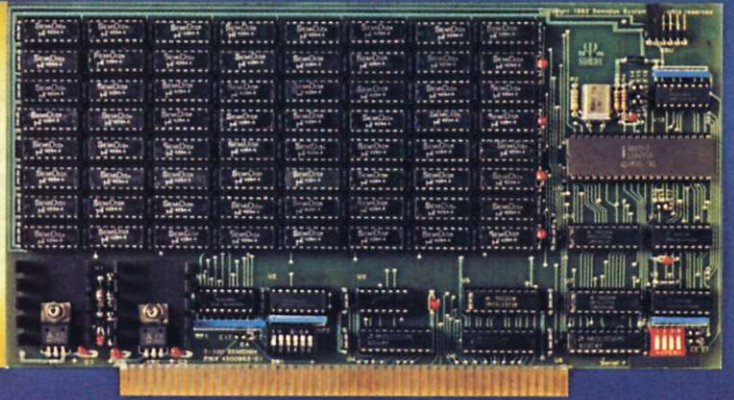


Fig. 4. Flowchart showing sequence of operations of the compiler.

# THE PRICE OF **FAST** WAS JUST SHATTERED!



## 256Kbyte SemiDisk™ **\$995**

For more than a year, we've been making the most advanced disk emulator available for micro-computers. The one that's taken the "waiting" out of computing. Now, we have some more news that'll set the world on fire: A price cut! The NEW 256Kbyte board is only \$995. And the 512Kbyte SemiDisks for the S-100 and TRS-80 Model II are \$1495. (1Mbyte unit is \$2350.) So, what are you waiting for?

The SemiDisk is the **ORIGINAL** single-board microcomputer disk emulator. It has a greater storage density than any other: 1 Mbyte per board! And we've been shipping them for over a year! We didn't do this with 'me too' engineering. Our products are true innovations, based on reliable technology and proven designs, without the need for custom components.

Floppies are ok for data transfer or long-term storage. But they fall far short as online storage. If you are using high level languages, spelling checkers, word processors, databases and other disk-intensive software, you know the price you are paying: time. Your productivity is going down the drain. The SemiDisk disk emulator will save time and increase your productivity.

Even better, Release 5.0 of the SemiDisk CP/M-80 installation software contains SemiSpool, an automatic printer buffer. No extra hardware is required; it's all in the software. Up to 8 Mbytes of buffer space! It's a better solution than a \$350 64Kbyte printer buffer that wastes space on your desk. Send documents of almost any length to the printer at a very high speed, then continue using the computer immediately. No Waiting!

### SemiDisk

*still*  
It's the disk the others are trying to copy.

## SemiDisk Systems, Inc.

P.O. Box GG Beaverton, OR 97075 (503) 642-3100

Call 503-646-5510 for CBBS™/NW, a SemiDisk-equipped computer bulletin board.

SemiDisk trademark of SemiDisk Systems, Inc. Copyright © 1983 SemiDisk Systems, Inc.

Circle 375 on Reader Service card.



The RECOMPILE procedure also sets the value of a pointer variable PTR equal to the address of the beginning of the source code buffer. The pointer PTR is the memory pointer of the compiler's lexical analyzer, and it's successively advanced from byte to byte of the source code to read the latter. The lexical analyzer reads in the byte in the memory location pointed to by the pointer PTR.

The lexical analyzer embodies another major change in the PL/0 compiler. It is embodied in the procedure GETSYM, which also has nested therein the procedure GETCH.

GETSYM's first statement is:

```
WHILE CH = ' ' DO GETCH;
```

This constitutes part of an infinite loop that repeats for as long as GETCH returns the ASCII code (32) for a space. As explained below, the procedure GETCH will return the space code 32 whenever it reads the Pause Mark.

GETCH's first statement is:

```
CH := PTR^;
```

so as to read into the variable CH the contents of the source memory location pointed to by the pointer variable PTR. The next statement of GETCH is:

```
IF CH = CHR(PM) THEN
```

This statement tests the byte read for the Pause Mark (PM) value, a constant equal to 35. This value was chosen because it is a visible character and was otherwise unused.

Following the IF clause, the assignment is:

```
CH := CHR(SP)
```

The SP is equal to the ASCII code (32) for a space. Control then returns to GETSYM, where the condition of the WHILE clause is satisfied so that it again invokes GETCH. This sequence is repeated and results in an infinite loop for as long as the byte in the memory location being read is the Pause Mark.

After the editor changes that byte from the Pause Mark to the ASCII code for a space, the loop will be broken, because the Boolean condi-

tion of the IF clause no longer will be satisfied (variable CH won't equal PM). Instead, the following ELSE clause will be executed so as to advance the source memory pointer PTR to the next memory location by the statement:

```
PTR := PTR + 1 ;
```

The next invocation of GETCH will read the next source memory location to enable the compiler to continue its advance through the source code. The pointer PTR is repeatedly advanced, with each successive call of GETCH, until it reaches the new Pause Mark inserted by the editor, as described below.

### Editor's Sequence

Fig. 5 shows the sequence of operations of the editor. The Input Character function is performed in response to the Keystroke (Fig. 3). The editor then determines if the input byte is a Control Character. If not, the character is entered into the source code buffer as indicated at Char Into Mem Buffer. The input character is also displayed on the screen as indicated at Char to Video Console.

If the input character is a control character, the editor then determines if it is a Carriage Return?. If not, the appropriate one of the editor's routines for handling control characters is called, as indicated by the legend To Control Char Routine, and as described below with reference to Fig. 6.

Still referring to Fig. 5, if the input character is a carriage return, then a new Pause Mark is written into the source code buffer adjacent to the end of the current line, as indicated at Insert New Pause Mark. The old Pause Mark is changed to a blank space, as indicated by the legend Remove Old Pause Mark.

For convenience in finding the location of the Pause Mark, a memory word location is reserved as a Pause Register for storage of the location address of the Pause Mark. The Pause Mark's new address thus is stored in this memory register, as indicated by the legend Update Pause Register.

The ASCII code for a carriage return (13) is then entered into the source code buffer adjacent to the Pause Mark, as indicated by Insert CR Into Mem Buffer. The ASCII code for a line-feed (10) may be entered after the carriage return if this convention is desired.

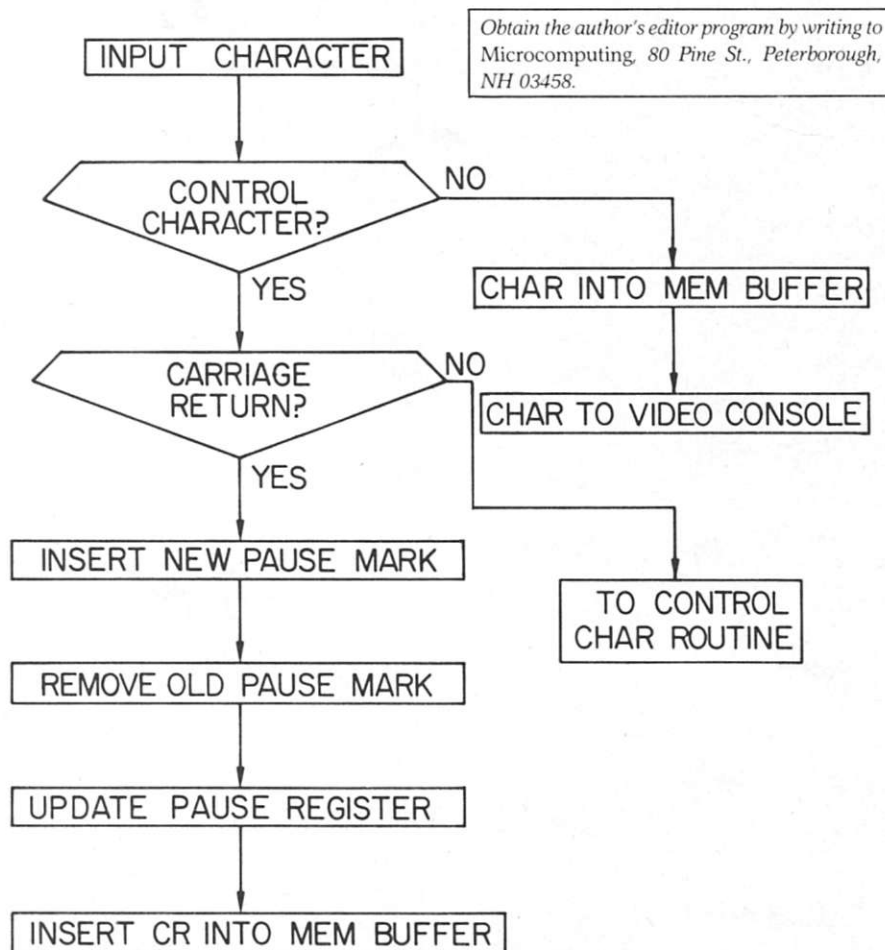


Fig. 5. Flowchart showing sequence of operations of the editor.

Fig. 6 shows the sequence of operations of the editor routines for handling control characters input at the console. The input character first is tested to determine if it is the code for the Cursor Up operation. If not, it is tested to determine if it is the code for the Screen Up operation. If not, the input control character is handled in a conventional manner that will not be described further, as indicated by Process Other Control Char.

If the input control character is the code for the Cursor Up or Screen Up, then the respective operation Move Cursor Up or Scroll Screen is performed. In the former case, the cursor is moved up one line on the video screen. In the latter case, the screen is erased and is rewritten to display those lines of the source code buffer immediately preceding the erased lines.

As indicated at Insert New Pause Mark, a pause mark is put in adjacent to the end of the source code buffer line immediately preceding the line now bearing the new cursor position. The operations Remove Old Pause Mark and Update Pause Register are then performed in the same manner as described above, with respect to Fig. 5.

The operation Set Recompil Flag causes reinitialization of the compiler when the latter resumes control of the CPU after return from the interrupt service routine. This flag is preferably a memory location wherein a predetermined code may be stored to inform the compiler that recompilation of the source code is required.

In my version, this recompile flag is set to require recompilation whenever the cursor is moved up or the screen frame is scrolled up. That is, it is assumed that whenever the cursor is moved to point to source code that may have been compiled already, this code will be changed so as to require recompilation.

### Another Way

An alternative method would be to set the recompile flag only if the previously compiled code is actually changed, since it is possible that the programmer may scroll up the screen and then scroll down again, without making any change in the source code.

Another alternative would be to maintain a memory register holding the address of the latest position of the compiler pointer. The editor then might compare this address with that

of the source location pointed to by the cursor to determine if the editing changes are being made to already-compiled source code.

Although these alternative schemes result in fewer recompilations, my invention has the advantage of simpler implementation. Furthermore, the compilation process is so much faster than the manual typing of source code at the console that the compiler will recompile all but the largest programs and catch up with the programmer before he can type more than a few new lines of code. Therefore, the reduction of the number of recompilations to the absolute minimum is not essential.

The editor is written in Pascal with calls to 16 external assembly-language procedures and functions. The following is a description of those routines unique to this invention.

Upon entry to the editor, the Boolean variable ECont is tested to determine if this invocation of the editor is the first entry of the present session or if it's a continuation. If ECont is False, then it is set equal to True and the following procedures are called: INIT, NEWFILE, VECTOR and TOPL0.

INIT clears the source code buffer, sets the memory pointer to the start of the buffer, inserts Pause Mark at the first location of the buffer, sets the contents of Pause Register to the address of this first location, initializes the cursor to the first row and first column of the screen and sets the recompile flag pointer to the memory location preceding the first byte of the buffer.

NEWFILE prompts the programmer to select either a new file for entry of source code or an old file for editing. If the editing option is chosen, the file is read into the source code buffer from a disk and the first screen of source code is displayed.

The procedure Vector calls the external assembly procedure Poke three times to store in low memory (20H) the jump vector to subroutine SAVREGS, which stores the contents of the CPU registers. In response to an interrupt activated by a keystroke at the console, the CPU executes the RST4 call instruction and executes this jump vector and then the SAVREGS subroutine. After the registers are saved, a jump instruction in the subroutine sends the CPU to the editor.

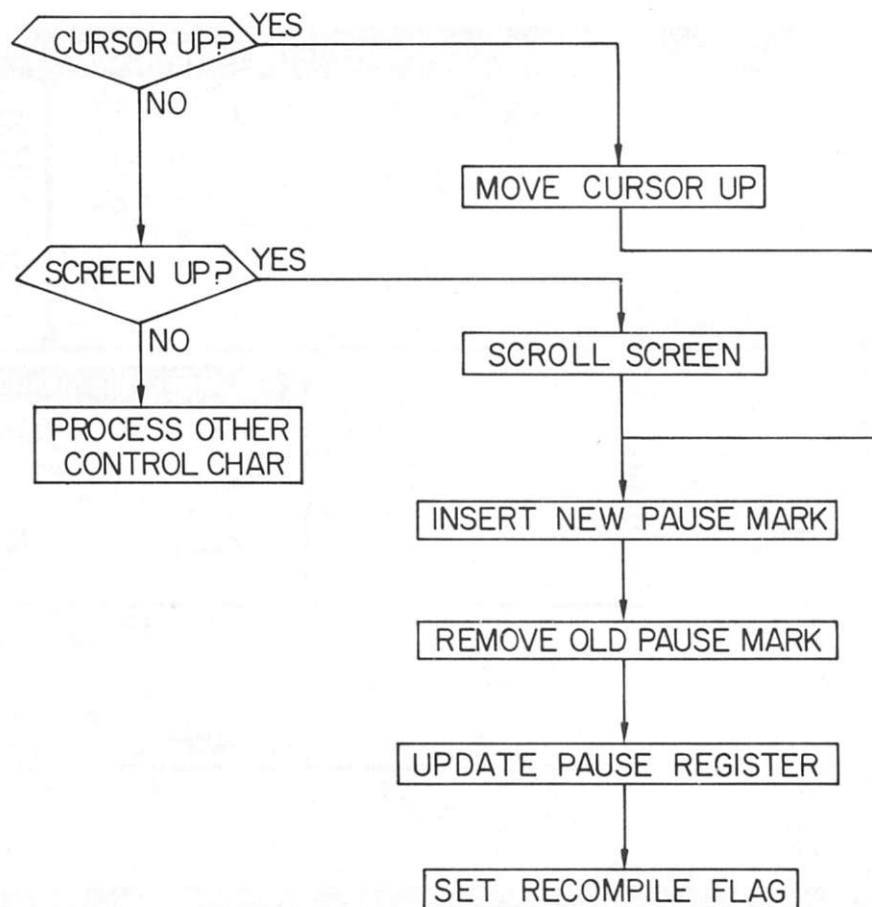


Fig. 6. Flowchart showing sequence of operations of control-character routines of the editor.

The TOPL0 procedure passes control to the PL/0 compiler. It usually is called after the editor has completed the character entry or editing function corresponding to the key struck at the terminal. In this case, the procedure is called after initialization of the editor.

### Reading Into a Variable

The next statement of the editor is not reached until an interrupt occurs in response to a keystroke. This statement reads into a variable the ASCII code input from the UART's received data register. This input byte is tested to determine if it is a control character or alphanumeric character (greater than 31). An alphanumeric character would be entered into the source code buffer and displayed on the video screen in the conventional manner.

If the input byte is the control code for either moving the cursor down or scrolling the screen frame down, the appropriate procedure is called and concludes with an invocation of the procedure Update.

UPDATE enters into the source buffer a new Pause Mark, adjacent to the end of the old line in the case of a cursor down operation, and adjacent to

the end of the invisible line preceding the first displayed line in the case of a scroll down operation. UPDATE also removes the old Pause Mark by substituting the ASCII code for a space (32) in place of the Pause Mark in its previous location. The Pause Register also is updated to the address of the new Pause Mark location.

If the input byte is the control code for either moving the cursor up or scrolling the screen frame up, the corresponding procedure performs the respective operation. This procedure concludes with invocations of UPDATE and RECOMPILE.

RECOMPILE stores the ASCII code for the letter R in the memory location immediately preceding the start of the source code buffer so as to constitute the recompile flag noted above. Upon completion of the cursor up operation or the screen scroll up operation, the CPU will return to the compiler, which will test the recompile flag, determine that the flag is set and then call its reinitialization procedure to force the compiler to recompile the source code from the beginning of the source buffer.

If the input byte is the ASCII code

for a carriage return (13), the procedure CRET is called. This routine enters into the source buffer a new Pause Mark adjacent to the carriage return code, removes the old Pause Mark and updates the Pause Register, among other more conventional functions (such as adding a line-feed code to the buffer, updating the cursor and scrolling the video display if the present line is the last line of the screen).

### One of Many Forms


It should be understood that the preferred embodiment described above and shown in the drawings is illustrative of merely one of the many forms that the invention may take in practice. Numerous modifications thereof can be made without departing from the scope of this invention.

For example, instead of the Pause Mark implemented as a predetermined code entered into a memory location within the source code buffer, the pause location may be defined for the compiler by a memory address stored in a register. The compiler then may be prevented from analyzing code stored in memory locations beyond

Circle 14 on Reader Service card.

**FIRST QUALITY COMPONENTS - NOT MAIL ORDER "SECONDS"**

#### ARIES ZERO INSERTION FORCE SOCKETS



cam actuated, true zero insertion - tin plated solder tail pins - capable of being plugged into dip sockets, including wire wrap.

Stock No.	No. Pins	1-8	10-49	50
11055	24	4.98	\$4.35	\$3.90
11056	28	5.15	4.50	4.05
11057	40	6.81	5.95	5.35
11058	64	12.02	10.50	9.45

#### IC-KOOLERS

UNITRACK<sup>®</sup> dissipate heat from IC's, producing longer life and better performance. Just bolt IC-Kooler on - heat is collected from top and bottom of IC and dissipated. Won't shake loose!

Stock No.	No. Pins in IC	Price
22225	14	\$ .29
22226	18	\$ .29
22227	18	\$ .29
22228	20	\$ .29

#### DIGITAL MULTIMETER

Single rotary switch operation. Large easy to read 5 1/2 digit display. 800 hours operating life with single 9v battery. Seven functions - (DC Volts, DC Amps, Ohms, AC Volts, AC Amps, Diode and Resistor Junction, Audible Continuity Check).

Stock No. 61303 **\$79.95**  
Full 1 year warranty

Stock No. 62304 Carrying case with belt loop **\$9.95**

#### TI WIRE WRAP SOCKETS

Tin plated phosphor bronze contact - 3 wrap

Stock No.	No. Pins	1-99	100-499	500
11301	8	\$ .40	\$ .36	\$ .30
11302	14	.59	.54	.45
11303	18	.64	.58	.48
11304	18	.73	.66	.55
11305	20	.99	.90	.75
11306	22	1.12	1.02	.85
11307	24	1.25	1.14	.95
11308	28	1.52	1.38	1.15
11309	40	2.05	1.86	1.55

#### TI LOW PROFILE SOCKETS

Tin plated copper alloy 688 contact pins with gas tight seal.

Stock No.	No. Pins	1-24	25-99	100
11201	8	\$ .10	\$ .09	\$ .08
11202	14	.14	.13	.12
11203	16	.16	.15	.14
11204	18	.18	.17	.15
11205	20	.20	.18	.16
11206	22	.22	.20	.18
11207	24	.24	.22	.20
11208	28	.28	.26	.25
11209	40	.40	.37	.33

#### WILD ROVER

Touch switch capsule. Operating motion is .005" without the use of a levered arm. Extremely fast on and off with low noise. Normally open - rated 115 VAC, 1.6 amp-30 milliohm resistance - .615 radius by .160 thick.

Stock No.	1-5	10 & Up
12098	\$1.42	\$1.28

#### 60/40 ROSIN CORE SOLDER

Stock No.	Dia. (inches)	Length (feet)	Weight (oz.)	Price
50075	.062	9	1.5	\$1.16
50076	.062	25	4	2.39
50077	.062	50	8	4.25
50078	.032	33	1.5	1.31
50079	.032	88	4	2.47
50080	.032	175	8	4.57

#### ELPAC POWER SUPPLIES - DC/DC CONVERTERS

SINTEC Stock No.	ELPAC Part No.	Input Voltage (VDC)	Output Voltage (VDC)	Current (mA)	Dimensions (HxWxD) in inches	Price
13825	CB3801	3.0-7.0	12±0.6	0-25	48x51x3.05	\$ 7.95
13826	CB3811	3.0-7.0	-12±0.6	0-25	48x51x3.05	7.95
13827	CB3802	3.0-7.0	15±0.7	0-20	48x51x3.05	7.95
13828	CB3812	3.0-7.0	-15±0.7	0-20	48x51x3.05	7.95
13829	CB3804	3.0-7.0	28±1.4	0-10	48x51x3.05	7.95
13830	CB3814	3.0-7.0	-28±1.4	0-10	48x51x3.05	7.95

Stock No. 13801 "Floppy Disc" Power Supply For Winchester Drives **\$109.00**  
13801-1 Data Sheet for 13801 ... 25

#### Special of the Month!

**HIGH PERFORMER II UHF-TV PREAMPLIFIER**  
Adds gain to UHF Reception

59002 Kit **\$37.50**  
59003 Assembled Unit **\$59.50**

For more detailed information call for catalog

#### MODUTEC

Miniclamp AC Volt-Ammeter allows singling one conductor out of many without disarrangement.

Stock No.	AC Amperes	Price
13730	0-25A	\$39.50
13731	0-50A	39.50
13732	0-100A	39.50

Accessory Line Splitter allows fast readings of AC power consumption of plug in equipment without separation of leads.  
Stock No. 13727 **\$9.95**

Pocket Sized Battery Tester for all types of small batteries from 1.35v to 4.5v.  
Stock No. 13733 **\$13.95**

Volt-I-CATOR automotive diagnostic meter plugs into lighter socket and indicates battery condition and charging rates.  
Stock No. 13736 **\$15.95**

AC Voltage Tester plugs into any 110v service receptacle to check line voltage over 50-150 VAC.  
Stock No. 13735 **\$14.95**

VOM-MULTITESTER versatile Volt-Ohm-Milliammeter in small package.  
Stock No. 13729 **\$13.95**

#### ELPAC POWER SUPPLIES - SOLV SERIES FULLY REGULATED

SINTEC Stock No.	ELPAC Part No.	Output Voltage	Output Current Rating	Dimensions (HxWxD) in inches	OVP	Price
13802	SOLV15-5	5	3.0A	4-7/16x4-2	Fixed included	\$39.95
13803	SOLV15-12	12	1.5A	4-7/16x4-2	Fixed included	39.95
13804	SOLV15-15	15	1.2A	4-7/16x4-2	Fixed included	39.95
13806	SOLV15-24	24	0.75A	4-7/16x4-2	Fixed included	39.95
13808	SOLV30-5	5	8.0A	5-5/8x4-7/8x3-1/8	OVP-4	59.95
13809	SOLV30-12	12	4.0A	5-5/8x4-7/8x3-1/8	OVP-4	59.95
13810	SOLV30-15	15	3.3A	5-5/8x4-7/8x3-1/8	OVP-4	59.95
13812	SOLV30-24	24	2.0A	5-5/8x4-7/8x3-1/8	OVP-4	59.95

13802-1 Data Sheet for SOLV Series ... 25

#### PIN FORMING TOOL

puts IC's on their true row to row spacing. One side for 300 centers, Flip tool over for devices on 600 centers. Put device in tool and squeeze.

NEW! Stock No. 10200 **\$14.95** (ANTI-STATIC MODEL)

ONE TOOL DOES 8 thru 40 PINS! Stock No. 11059 **\$12.95**

#### OK MACHINE AND TOOL

##### IC INSERTION/EXTRACTION KIT

Includes DIP IC extractors and inserters to accommodate all IC's from 14 to 40 pins. Tools that engage conductive surfaces are CMOS safe and include grounding lug light.

Stock No. **\$37.74**  
13309

##### SOCKET WRAP

DIP socket saved plastic panels with numbered holes in pin locations. Slip onto socket before wire wrapping to identify pins. Also write on them for location, IC part number, function, etc. Simplifies initial wire wrapping, troubleshooting and repair.

13295 14 pin \$1.99  
13296 16 pin \$1.99  
13297 20 pin \$1.99  
13298 24 pin \$1.99  
13299 28 pin \$1.99  
13300 24 pin \$1.99  
13301 28 pin \$1.99  
13302 40 pin \$1.99  
13303 96 pin \$1.99

**\$1.82 per pack**

##### IC EXTRACTOR

One-piece, spring steel construction. Will extract all LSI, MSI and SSL devices with 8 to 24 pins.

Stock No. 13313 **\$2.10**

**SINTEC** Drawer Q Milford CO. NJ 08848-9990

VISA MasterCard

**TOLL 800-526-5960**  
**FREE in NJ (201) 996-4093**

We accept VISA, MC, C.O.D., CHECK, or M.O. \*INCLUDE SHIPPING CHARGES. \$100 to \$250 - \$3.00 over \$250 - \$5.00

this address, which may be incremented and decremented by the editor.

The interrupt that causes control of the CPU to pass from the compiler to the editor may be activated by a timer or clock instead of by the keyboard. That is, the compiler may be periodically interrupted and the input port polled to test if a key has been struck. If not, the interrupt is terminated and control returns to the compiler.

If polling the port reveals that a key has been struck, then the interrupt service routine editor takes control and is executed in the manner described above for the disclosed preferred embodiment. For most applications, clock interrupts at intervals of about every ten to 30 milliseconds should be frequent enough to keep up with keys stroked at the keyboard.

The recompile flag may be set whenever the compiler determines that the source code contains an error. That is, it may be assumed that whenever an error is revealed, the source code will be changed so as to require recompilation.

Another possible modification is to eliminate recompilation from the beginning of the source code in instances where the error occurs in the last completed line.

During the compilation of that line, the resulting register values, table entries, stack manipulations, variable assignments and code buffer entries are temporarily stored and are not entered until the syntax analysis of the source line is completed and determines that the line conforms to the grammar.

If the line contains an error, these

temporary entries are discarded and the compiler pointer is moved back to the end of the previous line, thereby obviating recompilation. However, this scheme still will require recompilation if source lines previous to the last line are modified.

Still another possibility would be to have the editor advance Pause Mark after entry of each character or after entry of each delimited symbol.

This would provide the advantage of revealing an error almost instantly upon its entry to the keyboard, instead of waiting until completion of the current line. The disadvantage would be that recompilation would be required for every minor typing error without giving the programmer a chance to correct it before it is scanned and parsed. ■

Program Listing. N. Wirth's PL/O compiler as modified by author.

```
(* LAST CHANGE AUG. 15, 1982 *)
PROGRAM PL0;
(*SS*)
(*$Z $D000*)

(* PL/O COMPILER WITH CODE GENERATION *)

CONST
CR = 13;      (* CARRIAGE RETURN *)
LF = 10;      (* LINE FEED *)
PM = 35;      (* PAUSE MARK *)
SP = 32;      (* SPACE *)
BUP = $9000;  (* START (STX) SOURCE BUFFER *)
ENF = 26;     (* END FILE *)
NORW = 11;    (* NO. OF RESERVED WORDS *)
TXMAX = 100;  (* LENGTH OF IDENTIFIER TABLE *)
NMAX = 14;    (* MAX. NO. OF DIGITS IN NUMBERS *)
AL = 10;      (* LENGTH OF IDENTIFIERS *)
AMAX = 2047;  (* MAXIMUM ADDRESS *)
LEVMAX = 3;   (* MAXIMUM DEPTH OF BLOCK NESTING *)
CXMAX = 200;  (* SIZE OF CODE ARRAY *)

TYPE
SYMBOL =
(NUL, IDENT, NUMBER, PLUS, MINUS, TIMES, SLASH, ODDSYM,
EQL, NEQ, LSS, LEQ, GTR, GEO, LPAREN, RPAREN, COMMA, SEMICOLON,
PERIOD, BECOMES, BEGINSYM, ENDSYM, IFSYM, THENSYM,
WHILESYM, DOSYM, CALLSYM, CONSTSYM, VARSYM, PROCSYM);

ALFA = PACKED ARRAY[1..AL] OF CHAR;
OBJECT = (CONSTANT, VARIABLE, PROCEDURE);
SYMSET = SET OF SYMBOL;
FCT = (LIT, OPR, LOD, STO, CAL, INT, JMP, JPC); (* FUNCTIONS
INSTRUCTION = PACKED RECORD
      F: FCT;      (* FUNCTION CODE *)
      L: 0..LEVMAX; (* LEVEL *)
      A: 0..AMAX;  (* DISPLACEMENT ADDRESS *)

END;

(*
LIT 0,A : LOAD CONSTANT A
OPR 0,A : EXECUTE OPERATION A
LOD L,A : LOAD VARIABLE L,A
STO L,A : STORE VARIABLE L,A
CAL L,A : CALL PROCEDURE A AT LEVEL L
INT 0,A : INCREMENT T-REGISTER BY A
JMP 0,A : JUMP TO A
JPC 0,A : JUMP CONDITIONAL TO A *)

VAR
C: CHAR;
PTR: ^CHAR; (* POINTER TO SOURCE BUFFER *)
RFLAG: ^CHAR; (* RECOMPILE FLAG *)
CONT: BOOLEAN; (* FLAG FOR INITIALIZATION *)
CH: CHAR; (* LAST CHARACTER READ *)
SYM: SYMBOL; (* LAST SYMBOL READ *)
ID: ALFA; (* LAST IDENTIFIER READ *)
NUM: INTEGER; (* LAST NUMBER READ *)
CC: INTEGER; (* CHARACTER COUNT *)
LL: INTEGER; (* LINE LENGTH *)
EK: INTEGER;
ERR: INTEGER;
CX: INTEGER; (* CODE ALLOCATION INDEX *)
LINE: ARRAY[1..81] OF CHAR;
A: ALFA;
CODE: ARRAY[0..CXMAX] OF INSTRUCTION;
WSYM: ARRAY[1..NORW] OF SYMBOL;
WORD: ARRAY[1..NORW] OF ALFA;
SSYM: ARRAY[CHAR] OF SYMBOL;
MNEMONIC: ARRAY[FCT] OF
      PACKED ARRAY[1..5] OF CHAR;
DECLBEGSYS, STATBEGSYS, FACBEGSYS: SYMSET;
TABLE: ARRAY[0..TXMAX] OF
      RECORD
        NAME: ALFA;
        CASE KIND: OBJECT OF
          CONSTANT: (VAL: INTEGER);
          VARIABLE, PROCEDURE: (LEVEL,ADR: INTEGER)
        END;
      END;

EXTERNAL FUNCTION PEEK(LOC: INTEGER): CHAR;
EXTERNAL PROCEDURE ABORT;

PROCEDURE ERROR(N: INTEGER);
BEGIN
  Writeln(' *****', ' : CC-1, ', N:2);
  ERR := ERR + 1
END (* ERROR *);

PROCEDURE GETSYM;
VAR
  I, J, K: INTEGER;

PROCEDURE GETCH;
BEGIN
  CH := PTR^;
  IF CH = CHR(PM) THEN
    CH := CHR(SP)
  ELSE
    BEGIN
      PTR := PTR + 1;
      IF CH = CHR(ENF) THEN
        BEGIN
          WRITE('PROGRAM INCOMPLETE');
          CC := 0;
          WRITE(CX:5, ' ');
          END;
        IF (CH = CHR(CR)) OR (CH = CHR(LF)) THEN
          CH := CHR(SP)
        END (* ELSE *)
      END (* GETCH *)
    BEGIN (* GETSYM *)
      WHILE CH = ' ' DO GETCH;
      IF CH IN ['A'..'Z'] THEN
        BEGIN (* IDENTIFIER OR RESERVED WORD *)
          K := 0;
          REPEAT
            IF K < AL THEN
              BEGIN
                K := K+1;
                A[K] := CH
              END;
            GETCH
          UNTIL NOT (CH IN ['A'..'Z','0'..'9']);
          IF K >= KK THEN
            KK := K
          ELSE
            REPEAT
              A[KK] := ' ';
              UNTIL KK = K;
            ID := A;
            I := 1;
            J := NORW;
            REPEAT
              K := (I+J) DIV 2;
              IF ID <= WORD[K] THEN J := K-1;
              IF ID >= WORD[K] THEN I := K+1
            UNTIL I > J;
            IF I > J+1 THEN
              SYM := WSYM[K]
            ELSE
              SYM := IDENT
            END
          END
        ELSE
          IF CH IN ['0'..'9'] THEN
            BEGIN (* NUMBER *)
              K := 0;
              NUM := 0;
              SYM := NUMBER;
              REPEAT
                NUM := 10*NUM + (ORD(CH)-ORD('0'));
                K := K+1;
              UNTIL NOT (CH IN ['0'..'9']);
              IF K > NMAX THEN ERROR (30)
            END
          ELSE
            IF CH = '.' THEN
              BEGIN
                GETCH;
                IF CH = '=' THEN
                  BEGIN
                    SYM := BECOMES;
                    GETCH
                  END
                ELSE
                  SYM := NUL;
                END
            ELSE
              BEGIN
                BEGIN
                  GETCH;
                  IF CH = '=' THEN
                    BEGIN
                      SYM := BECOMES;
                      GETCH
                    END
                  ELSE
                    SYM := NUL;
                END
              ELSE
                BEGIN

```

More →

Listing continued.

```

SYM := SSYM[CH];
GETCH
END
END (* GETSYM *) ;
PROCEDURE GEN(X: FCT; Y,Z: INTEGER);
BEGIN
IF CX > CXMAX THEN
BEGIN
WRITE('PROGRAM TOO LONG');
ABORT
END ;
WITH CODE[CX] DO
BEGIN
F := X;
L := Y;
A := Z
END
END (* GEN *)
PROCEDURE TEST(S1,S2: SYMSET; N: INTEGER);
BEGIN
IF NOT (SYM IN S1) THEN
BEGIN
ERROR(N);
S1 := S1 + S2;
WHILE NOT(SYM IN S1) DO GETSYM
END
END (* TEST *) ;
PROCEDURE BLOCK(LEV, TX: INTEGER; FSYS: SYMSET);
VAR DX: INTEGER; (* DATA ALLOCATION INDEX *)
TX0: INTEGER; (* INITIAL TABLE INDEX *)
CX0: INTEGER; (* INITIAL CODE INDEX *)
PROCEDURE ENTER(X: OBJECT);
BEGIN(* ENTER OBJECT INTO TABLE *)
TX := TX + 1;
WITH TABLE[TX]DO
BEGIN
NAME := ID;
KIND := F;
CASE K OF
CONSTANT: BEGIN
IF NUM > AMAX THEN
BEGIN
ERROR (30);
NUM := 0
END;
VAL := NUM
END ;
VARIABLE: BEGIN
LEVEL := LEV;
ADR := DX;
DX := DX+1;
END ;
"PROCEDUR: LEVEL := LEV
END
END (* ENTER *) ;
FUNCTION POSITION(ID: ALFA): INTEGER;
VAR I: INTEGER;
BEGIN (* FIND IDENTIFIER ID IN TABLE *)
TABLE[0].NAME := ID;
I := TX;
WHILE TABLE[I].NAME <> ID DO I := I+1;
POSITION := I
END (* POSITION *) ;
PROCEDURE CONSTDECLARATION;
BEGIN
IF SYM = IDENT THEN
BEGIN
GETSYM;
IF SYM IN [EQL, BECOMES] THEN
BEGIN
IF SYM = BECOMES THEN ERROR (1);
GETSYM;
IF SYM = NUMBER THEN
BEGIN
ENTER(CONSTANT);
GETSYM
END
ELSE
ERROR (2)
END
ELSE
ERROR (3)
END
ELSE
ERROR (4)
END (* CONSTDECLARATION *) ;
PROCEDURE VARDECLARATION;
BEGIN
IF SYM = IDENT THEN
BEGIN
ENTER(VARIABLE);
GETSYM
END
ELSE
ERROR (4)
END (* VARDECLARATION *) ;
PROCEDURE LIS'CODE;
VAR I: INTEGER;
BEGIN (* LIST CODE GENERATED FOR THIS BLOCK *)
FOR I := CX0 TO CX-1 DO
WITH CODE[I] DO
WRITELN(L, MNEMONIC[F]:5, L:3, A:5)
END (* LISTCODE *) ;
PROCEDURE STATEMENT(FSYS: SYMSET);
VAR I, CX1, CX2: INTEGER;
PROCEDURE EXPRESSION(FSYS: SYMSET);
VAR ADDOP: SYMBOL;
PROCEDURE TERM(FSYS: SYMSET);
VAR MULOP: SYMBOL;
PROCEDURE FACTOR(FSYS: SYMSET);
VAR I: INTEGER;
BEGIN
TEST(FACBEGSYS,FSYS, 24);
WHILE SYM IN FACBEGSYS DO
BEGIN
IF SYM = IDENT THEN
BEGIN
I := POSITION(ID);
IF I = 0 THEN
ERROR(11)
ELSE
WITH TABLE[I] DO
CASE KIND OF
CONSTANT: GEN(LIT, 0, VAL)
VARIABLE: GEN(LOD,

```

```

LEV-LEVEL, ADR);
PROCEDUR: ERROR(21)
END ;
GETSYM
END
ELSE
IF SYM = NUMBER THEN
BEGIN
IF NUM > AMAX THEN
BEGIN
ERROR(30);
NUM := 0
END ;
GEN(LIT, 0, NUM); GETSYM
END
ELSE
IF SYM = LPAREN THEN
BEGIN
GETSYM;
EXPRESSION([RPAREN]+FSYS);
IF SYM = RPAREN THEN GETSYM
ELSE ERROR(22)
END ;
TEST(FSYS, [LPAREN], 23)
END
END (* FACTOR *) ;
BEGIN (* TERM *)
FACTOR(FSYS+(TIMES, SLASH));
WHILE SYM IN [TIMES, SLASH] DO
BEGIN
MULOP := SYM;
GETSYM;
FACTOR(FSYS+(TIMES, SLASH));
IF MULOP = TIMES THEN
GEN (OPR,0,4)
ELSE
GEN(OPR,0,5)
END
END (* TERM *) ;
BEGIN (* EXPRESSION *)
IF SYM IN [PLUS, MINUS] THEN
BEGIN
ADDOP := SYM;
GETSYM;
TERM(FSYS+[PLUS, MINUS]);
IF ADDOP = MINUS THEN GEN(OPR,0,1)
END
ELSE
TERM(FSYS+[PLUS, MINUS]);
WHILE SYM IN [PLUS, MINUS] DO
BEGIN
ADDOP := SYM;
GETSYM;
TERM(FSYS+[PLUS, MINUS]);
IF ADDOP = PLUS THEN GEN(OPR,0,2) ELSE GEN(OPR,0,3)
END
END (* EXPRESSION *) ;
PROCEDURE CONDITION(FSYS: SYMSET);
VAR RELOP: SYMBOL;
BEGIN
IF SYM = ODDSYM THEN
BEGIN
GETSYM;
EXPRESSION(FSYS);
GEN(OPR,0,6)
END
ELSE
EXPRESSION([EQL, NEQ, LSS, GTR, LEQ, GEQ]+FSYS);
IF NOT(SYM IN [FOL, NEO, ISS, LEO, GTR, GEQ]) THEN
ERROR (2)
ELSE
BEGIN
RELOP := SYM;
GETSYM;
EXPRESSION(FSYS);
CASE RELOP OF
EQL: GEN(OPR,0, 8);
NEQ: GEN(OPR,0, 9);
LSS: GEN(OPR,0,10);
GEQ: GEN(OPR,0,11);
GTR: GEN(OPR,0,12);
LEQ: GEN(OPR,0,13);
END (* CASE *)
END (* ELSE *)
END (* ELSE *)
END (* CONDITION *) ;
BEGIN (* STATEMENT *)
IF SYM = IDENT THEN
BEGIN
I := POSITION(ID);
IF I = 0 THEN
ERROR (11)
ELSE
IF TABLE[I].KIND <> VARIABLE THEN
BEGIN (* ASSIGNMENT TO NON-VARIABLE *)
ERROR (12);
I := 0
END ;
GETSYM;
IF SYM = BECOMES THEN
GETSYM
ELSE
ERROR (13);
EXPRESSION(FSYS);
IF I <> 0 THEN
WITH TABLE[I] DO
GEN(STO, LEV-LEVEL, ADR)
END
ELSE IF SYM = CALLSYM THEN
BEGIN
GETSYM;
IF SYM <> IDENT THEN
ERROR (14)
ELSE
BEGIN
I := POSITION(ID);
IF I = 0 THEN
ERROR (11)
ELSE
WITH TABLE[I] DO
IF KIND = PROCEDUR THEN
GEN (CAL, LEV-LEVEL, ADR)
ELSE
ERROR (15);
GETSYM
END (* ELSE *)
END (* ELSE IF *)
ELSE IF SYM = IFSYM THEN
BEGIN
GETSYM;
CONDITION([THENSYM, DOSYM]+FSYS);
IF SYM = THENSYM THEN

```

More →

Listing continued.

```
GETSYM
ELSE
  ERROR(16);
CX1 := CX;
GEN(JPC,0,0);
STATEMENT(FSYS);
CODE[CX1].A := CX
END (* ELSE IF *)

ELSE IF SYM = BEGINSYM THEN
  BEGIN
    GETSYM;
    STATEMENT([SEMICOLON, ENDSYM]+FSYS);
    WHILE SYM IN [SEMICOLON]+STATBEGSYS DO
      BEGIN
        IF SYM = SEMICOLON THEN
          GETSYM
        ELSE
          ERROR (10);
          STATEMENT([SEMICOLON, ENDSYM]+FSYS)
        END ;
        IF SYM = ENDSYM THEN
          GETSYM
        ELSE
          ERROR (17)
        END (* ELSE IF *)
      END
    END
  END (* ELSE IF *)

ELSE IF SYM = WHILESVM THEN
  BEGIN
    CX1 := CX;
    GETSYM; CONDITION([DOSYM]+FSYS);
    CX2 := CX;
    GEN(JPC,0,0);
    IF SYM = DOSYM THEN
      GETSYM
    ELSE
      ERROR (18);
      STATEMENT(FSYS);
      GEN(JMP,0,CX1);
      CODE[CX2].A := CX
    END ; (* ELSE IF *)

    TEST(FSYS, [ ], 19)
  END (* STATEMENT *) ;

BEGIN (* BLOCK *)
  DX := 3;
  TX := TX;
  TABLE[TX].ADR := CX;
  GEN(JMP,0,0);
  IF LEV > LEVMAX THEN ERROR (32);
  REPEAT
    IF SYM = CONSTSYM THEN
      BEGIN
        GETSYM;
        REPEAT
          CONSTDECLARATION;
          WHILE SYM = COMMA DO
            BEGIN
              GETSYM;
              CONSTDECLARATION
            END ;
            IF SYM = SEMICOLON THEN GETSYM ELSE ERROR (5)
          UNTIL SYM <> IDENT;
        END ;
      END
    IF SYM = VARSYM THEN
      BEGIN
        GETSYM;
        REPEAT
          VARDECLARATION;
          WHILE SYM = COMMA DO
            BEGIN
              GETSYM;
              VARDECLARATION
            END;
            IF SYM = SEMICOLON THEN
              GETSYM
            ELSE
              ERROR(5)
            UNTIL SYM <> IDENT;
          END; (* IF VARSYM *)
        WHILE SYM = PROCSYM DO
          BEGIN
            GETSYM;
            IF SYM = IDENT THEN
              BEGIN
                ENTER(PROCEDUR);
                GETSYM
              END
            ELSE ERROR (4);
            IF SYM = SEMICOLON THEN GETSYM ELSE ERROR (5);
            BLOCK(LEV+1,TX,[SEMICOLON]+FSYS);
            IF SYM = SEMICOLON THEN
              BEGIN
                GETSYM;
                TEST(STATBEGSYS+[IDENT, PROCSYM],FSYS, 6)
              END
            ELSE ERROR (5)
            END ;
          TEST(STATBEGSYS+[IDENT],DECLBEGSYS,7)
          UNTIL NOT(SYM IN DECLBEGSYS);
          CODE[TABLE[TX0].ADR].A :=CX;
          WITH TABLE[TX0] DO
            BEGIN
              ADR := CX; (* START ADR OF CODE *)
            END ;
          CX0 := CX; GEN(INT,0,DX);
          STATEMENT([SEMICOLON,ENDSYM]+FSYS);
          GEN(OPR,0,0); (* RETURN *)
          TEST(FSYS, [ ], 8);
          LISTCODE;
        END (* BLOCK *) ;

PROCEDURE INTERPRET;
CONST STACKSIZE = 500;
VAR P,B,T: INTEGER; (* PROGRAM-,BASE-,TOPSTACK-REGISTERS *)
I: INSTRUCTION; (* INSTRUCTION REGISTER *)
S: ARRAY [1..STACKSIZE] OF INTEGER; (* DATASTORE *)

FUNCTION BASE(L: INTEGER): INTEGER;
VAR B1: INTEGER;
BEGIN
  B1 := B; (* FIND BASE L LEVELS DOWN *)
  WHILE L > 0 DO
    BEGIN
      B1 := S[B1];
      L := L-1
    END ;
  BASE := B1
END (* BASE *) ;
```

```
BEGIN
  WRITELN('START PL/O');
  T := 0;
  B := 1;
  P := 0;
  S[1] := 0; S[2] := 0; S[3] := 0;
  REPEAT
    I := CODE[P];
    T := P+1;
    WITH I DO
      CASE F OF
        LIT: BEGIN T := T+1; S[T] := A
              END ;
        OPR: CASE A OF (* OPERATOR *)
              0: BEGIN (* RETURN *) T := B-1; P := S[T+3]; B := S[T]
                  END ;
              1: S[T] := -S[T];
              2: BEGIN T := T-1; S[T] := S[T] + S[T+1]
                  END ;
              3: BEGIN T := T-1; S[T] := S[T] - S[T+1] END ;
              4: BEGIN T := T-1; S[T] := S[T] * S[T+1]
                  END ;
              5: BEGIN T := T-1; S[T] := S[T] DIV S[T+1]
                  END ;
              6: S[T] := ORD(ODD(S[T]));
              8: BEGIN T := T-1; S[T] := ORD(S[T]=S[T+1])
                  END ;
              9: BEGIN T := T-1; S[T] := ORD(S[T]<S[T+1])
                  END ;
              10: BEGIN T := T-1; S[T] := ORD(S[T]<S[T+1])
                  END ;
              11: BEGIN T := T-1; S[T] := ORD(S[T]>S[T+1])
                  END ;
              12: BEGIN T := T-1; S[T] := ORD(S[T]>S[T+1])
                  END ;
              13: BEGIN T := T-1; S[T] := ORD(S[T]<=S[T+1])
                  END ;
            END ;
        LOD: BEGIN T := T+1; S[T] := S[BASE(L)+A]
              END ;
        STO: BEGIN S[BASE(L)+A] := S[T]; WRITELN(S[T]); T := T-1
              END ;
        CAL: BEGIN (* GENERATE NEW BLOCK MARK *)
              S[T+1] := BASE(L); S[T+2] := B; S[T+3] := P;
              B := T+1; P := A
            END ;
        INT: T := T+A;
        JMP: P := A;
        JPC: BEGIN IF S[T] = 0 THEN P := A; T := T-1
              END ;
        END (* WITH, CASE *)
      UNTIL P = 0;
      WRITE('END PL/O');
    END (* INTERPRET *) ;

PROCEDURE ZERO;
BEGIN
  ERR := 0;
  CC := 0;
  CX := 0;
  LL := 0;
  CH := ' ';
  KK := AL;
END;

PROCEDURE RECOMPILE;
BEGIN
  RFLAG := BUF - 1;
  RFLAG := ' ';
  PTR := BUF + 1;
  ZERO;
END; (* RECOMPILE *)

BEGIN (* MAIN PROGRAM *)
  IF NOT CONT THEN
    BEGIN
      CONT := TRUE;
      RFLAG := BUF - 1;
      RECOMPILE;
      FOR CH := 'A' TO 'Z' DO SSYM[CH] := NUL;
      WORD[1] := 'BEGIN'; WORD[2] := 'CALL';
      WORD[3] := 'CONST'; WORD[4] := 'DO';
      WORD[5] := 'END'; WORD[6] := 'IF';
      WORD[7] := 'ODD'; WORD[8] := 'PROCEDURE';
      WORD[9] := 'THEN'; WORD[10] := 'VAR';
      WORD[11] := 'WHILE';
      WSYM[1] := BEGINSYM; WSYM[2] := CALLSYM;
      WSYM[3] := CONSTSYM; WSYM[4] := DCSYM;
      WSYM[5] := ENDSYM; WSYM[6] := IFSYM;
      WSYM[7] := ODDSYM; WSYM[8] := PROCSYM;
      WSYM[9] := THESYM; WSYM[10] := VARSYM;
      WSYM[11] := WHILESVM;
      SSYM['+'] := PLUS; SSYM['-'] := MINUS;
      SSYM['*'] := TIMES; SSYM['/'] := SLASH;
      SSYM['('] := LPAREN; SSYM[']'] := RPAREN;
      SSYM['='] := EQL; SSYM[','] := COMMA;
      SSYM['.'] := PERIOD; SSYM[':'] := NEQ;
      SSYM['<'] := LSS; SSYM['>'] := GTR;
      SSYM[';'] := SEMICOLON;
      MNEMONIC[LIT] := 'LIT'; MNEMONIC[OPR] := 'OPR';
      MNEMONIC[LOD] := 'LOD'; MNEMONIC[STO] := 'STO';
      MNEMONIC[CAL] := 'CAL'; MNEMONIC[INT] := 'INT';
      MNEMONIC[JMP] := 'JMP'; MNEMONIC[JPC] := 'JPC';
      DECLBEGSYS := [CONSTSYM, VARSYM, PROCSYM];
      STATBEGSYS := [BEGINSYM, CALLSYM, IFSYM, WHILESVM];
      FACBEGSYS := [IDENT, NUMBER, LPAREN];
    END; (* IF NOT CONT *)

    C := PEEK(BUF - 1);
    IF C = 'R' THEN
      RECOMPILE;
    CH := ' ';
    GETSYM;
    BLOCK(0,0,[PERIOD]+DECLBEGSYS+STATBEGSYS);
    IF SYM <> PERIOD THEN ERROR (9);
    IF ERR = 0 THEN INTERPRET ELSE WRITE('ERRORS IN PL/O PROGRAM');
    WRITELN
  END ;
```



---

# Sony's Marketable Micro

*With its many unique features, including micro floppies and an assortment of available add-ons, the SMC-70 is the "one and only" micro from Sony.*

By Bruce Kline



*The Sony SMC-70 microcomputer, featuring 3½-inch micro floppy disk drives.*

The Japanese microcomputer industry appears to be waiting on technology's sideline. A few companies, though, are actively pushing products: Sharp, Panasonic, NEC, Epson and Toshiba. And now Sony is joining that list—with a system that could only be the product of long-term development: the Sony SMC-70 (Sony Microcomputer Products, Sony Drive, Park Ridge, NJ 07656).

Sony is selling a "system," not just an eight-bit microcomputer. The minimum system could be as small as the SMC-70 computer/keyboard with a monochrome monitor and a cassette recorder; however, Sony is promoting the SMC-70 as a business machine that includes the SMC-70 with dual disks, a color monitor and a printer. Available expansions include a numeric keypad, up to four micro floppy disk drives, an eight-inch floppy controller and a video signal converter for connection to a TV. Other peripherals that are or will be available from Sony include a light pen, a cache disk unit, a second RS-232C interface, an IEEE-488 interface, a battery back-up unit, a bus expansion unit, an 8086 adapter, a 192K bank RAM for the Z-80 and up to 768K of RAM for the 8086.

The packaging is compact and well-

---

*Address correspondence to Bruce R. Kline, Pantown Road, Vergennes, VT 05491.*

---

# LOWEST SOFTWARE PRICES GUARANTEED

We hereby certify that your purchase from Discount Software represents the lowest price sold anywhere. If you find a lower price on what you purchased within 30 days, send the ad and we'll refund the difference.

## CP/M

**ARTIFICIAL INTELLIGENCE**  
 Medical (PAS-3) .....\$849  
 Dental (PAS-3) .....\$849  
**ASHTON-TATE**  
**\$4??** dBASE II...  
 call for price

Financial Planner .....\$595  
 Bottom Line Strategist...\$349

**ASYST DESIGN/FRONTIER**  
 Prof Time Accounting...\$549  
 General Subroutine...\$269  
 Application Utilities...\$439

**DIGITAL RESEARCH**  
 CP/M 2.2  
 Intel MDS.....\$135

**\$149** Northstar

**\$159** TRS-80 Model II  
 (P&T)  
 Micropolis.....\$175

**\$98** CBasic-2

Display Manager.....\$319  
 Access Manager.....\$23E  
 Multiplan.....\$219

**\$449** PL/1-80

BT-80.....\$179  
 MAC.....\$85  
 RMAC.....\$179  
 Sid.....\$65

**\$90** Z-Sid

DeSpool.....\$49  
 CB-80.....\$459  
 Link-80.....\$90

**FOX & GELLER**  
 Quickscreen.....\$135  
 Quickcode.....\$265

**\$65** DUtl

**MICRO-AP**  
 Selector IV.....\$295  
 Selector V.....\$495  
**MICRO DATA BASE SYSTEMS**  
 HDBS.....\$269

Discount Price MDBS.....\$1099  
 DRS or QRS or RTL.....\$319  
 MDBS PKG.....\$1999  
**MICROPRO**

**\$289** WordStar

**\$199** Mail Merge

WordStar/Mailmerge...\$399  
 WS/MM/SpellStar.....\$549  
 Customization Notes...\$44

**\$199** SpellStar

DataStar.....\$249  
 InfoStar.....\$349  
 ReportStar.....\$254  
 Wordmaster.....\$119  
 Supersort I.....\$199  
 Calc Star.....\$129

**MICROSOFT**

**\$229** Basic-80

**\$329** Basic Compiler

**\$349** Fortran-80

**\$549** Cobol-80

M-Sort.....\$175

**\$159** Macro-80

MuSimp/MuMath.....\$224  
 MuLisp-80.....\$174

**ORGANIC SOFTWARE**  
 Textwriter III.....\$111  
 Datebook II.....\$269  
 Milestone.....\$269

**OSBORNE (McGraw/Hill)**  
 G/L, or AR & AP, or PAY...\$59  
 All 3.....\$129  
 All 3 + CBASIC-2.....\$199  
 Enhanced Osborne.....\$299

**PEACHTREE**  
 G/1, A/R, A/P, PAY, INV(each)\$399  
 PB Version Add \$234  
 Peachcalc.....\$249

Other.....Less 10%  
**STAR COMPUTER SYSTEMS**  
 G/L, A/R, A/P, Pay(each)...\$349  
 All 4.....\$1129  
 Legal or Property Mgt.....\$849

**STRUCTURED SYSTEMS**  
 Business Packages (call)

**SORCIM**  
**\$249** SuperCalc

Act.....\$157

**SUPERSOFT**

Ada.....\$270  
 Diagnostic II.....\$89  
 Disk Doctor.....\$89  
 Forth (8080 or z80).....\$149  
 Fortran.....\$319  
 Ratfor.....\$79  
 C Compiler.....\$225  
 Star Edit.....\$189  
 Scratch Pad.....\$266  
 StatsGraph.....\$174  
 Analyza II.....\$45  
 Disk Edit.....\$89  
 Encode/Decode II.....\$84  
 Optimizer.....\$174  
 Term II.....\$179  
 Utilities I or II.....\$54

**SOFTWARE DIMENSIONS/ACCOUNTING PLUS**  
 Per Module.....\$399

**UNICORN**  
 Mince or Scribble (each)...\$149  
 Both.....\$249  
 The Final Word.....\$270

**WHITESMITHS**  
 "C" Compiler.....\$600  
 Pascal (incl "C").....\$850

**"PASCAL"**  
 Pascal/MT+ Pkg.....\$429  
 Compiler.....\$315  
 SP Prog.....\$175  
 Pascal Z.....\$349  
 Pascal/UCSD 4.0.....\$670

**DATA BASE**  
 dBASE II.....Call 4??  
 FMS-80.....\$799  
 FMS-81.....\$399  
 Condor I & III.....Call  
 Superfile.....\$159

**"WORD PROCESSING"**  
 Perfect Writer.....\$189  
 WordSearch.....\$114  
 SpellGuard.....\$199  
 Peachtex.....\$289  
 Spell Binder.....\$349  
 Select.....\$495  
 The Word.....\$65

**\$145** The Word Plus

Palantier-1 (WP).....\$385  
**"COMMUNICATIONS"**  
 Ascrom.....\$149  
 BSTAM or BSTMS.....\$149

**\$139** Crosstalk

**\$89** Move-it

**"OTHER GOODIES"**

Micro Plan.....\$419  
 Plan 80.....\$495  
 Target PlannerCalc.....\$79  
 Target Financial Modeling...\$299  
 Target Task.....\$299  
 Tiny "C".....\$89  
 Tiny "C" Compiler.....\$229  
 MicroStat.....\$224  
 Vedit.....\$130  
 MiniModel.....\$449  
 StatPak.....\$449  
 Micro B+.....\$229  
 String/80.....\$84  
 String/80 (source).....\$279  
 ISIS CP/M Utility.....\$199  
 Lynx.....\$199  
 Supervyz.....\$95  
 ATI Power (tutorial).....\$75  
 Mathe Magic.....\$95  
 CIS Cobol.....\$765  
 Forms II  
 Basic.....\$249  
 Zip MBasic, CBasic.....\$129

**APPLE II**

**ASHTON-TATE**  
 (See CP/M Ashton-Tate)

**BRODERBUND**  
 G/L (with A/P).....\$444  
 Payroll.....\$355

**INFO UNLIMITED**  
 EasyWriter (Prof).....\$155  
 EasyMailer (Prof).....\$134  
 Datadex.....\$129

**MICROSOFT**  
 Softcard (Z-80 CP/M).....\$239  
 Fortran.....\$179  
 Cobol.....\$499

**MICROPRO**  
 (See CP/M Micropro)  
**VISICORP**

Visicalc 3.3.....\$189  
 Desktop/Plan II.....\$219  
 Visiterm.....\$90  
 Visidex.....\$219  
 Visitrend/Visiplot.....\$259

Visifile.....\$219  
 Visischedule.....\$259  
**PEACHTREE**  
 G/L, A/R, A/P, PAY, (each)...\$224  
 PeachPack P40.....\$395

**ACCOUNTING PLUS**  
 G/L, AR, AP, INV, (each)...\$385

**"OTHER GOODIES"**  
 Super-Text II.....\$127  
 Data Factory.....\$269  
 Mini Factory.....\$139  
 DB Master.....\$184  
 Versaform VS1.....\$350

**IBM PC, 16 BIT 8, DISPLAYWRITER**

**"WORD PROCESSING"**

Wordstar.....\$289  
 Spellstar.....\$199  
 Mailmerge.....\$199  
 Easywriter.....\$314  
 EasySpeller.....\$159  
 Select/Superspell.....\$535  
 Write On.....\$116  
 Spellguard.....\$189  
 Textwriter III.....\$111  
 Spellbinder.....\$349  
 Final Word.....\$270

**"LANGUAGES & UTILITIES"**

Crosstalk.....\$174  
 Move-it.....\$129  
 BSTAM or BSTMS.....\$149  
 Pascal MT+ /86, SPP.....\$679  
 CBasic 86.....\$294  
 Act 86.....\$157  
 Trans 86.....\$115  
 XLT 86.....\$135  
 MBasic (MSDOS).....\$329  
 MBasic Compiler (MSDOS)....\$329  
 Both.....\$629  
 CBasic Compiler (MSDOS)....\$495  
 Cobol (MSDOS).....\$649  
 Pascal (MSDOS).....\$429  
 Fortran (MSDOS).....\$429  
 "C" (MSDOS).....\$429  
 CP/M 86.....\$239

**"OTHER GOODIES"**  
 Lotus 1-2-3.....\$329  
 SuperCalc.....\$269  
 VisiCalc.....\$219  
 Visiplot/trend.....\$259  
 Visidex.....\$219  
 Easyfiler.....\$359  
 Mathemagic.....\$95

dBase II.....Call 4??  
 Condor Q & R, Others.....Call  
 Statpak.....\$449  
 Optimizer.....\$174  
 Desktop Plan.....\$259

## FREE WITH PURCHASE:

**Complete Software Buyer's Guide (\$5.00 value)**  
 Filled with facts and usable advice about scores and scores of software programs from accounting and business systems to word processing and utilities.

**Exclusive "Hotline"**  
 Our reputation for courteous and knowledgeable service has resulted in calls from people who never purchased our products. Now a separate "hotline" is available to customers only.

**Confidential Software BargainGrams**  
 Regular notices of insider's bargains not available to the general public.

### DISCOUNT SOFTWARE

Outside Continental U.S.--add \$10 plus Air Parcel Post. Add \$3.50 postage and handling per each item. California residents add 6 1/2% sales tax. Allow 2 weeks on checks. C.O.D. \$3.00 extra. Prices subject to change without notice. All items subject to availability. \*Mfr. trademark. Blue Label \$3.00 additional per item. CP/M is a registered trademark of DIGITAL RESEARCH, INC.

ORDER TOLL-FREE  
 VIA VISA OR  
 MASTERCARD:

**1 800 421-4003**

Calif: 1 800 252-4092

6520 Selma Avenue, Los Angeles, CA 90028

# DISCOUNT SOFTWARE

MC 683

organized. The first two micro floppies mount within the main unit and take up an insignificant amount of space. The main unit has room for two expansion modules, but the modules must not require any external connections. Memory or cache disks are ideal candidates for this position.

The next three expansion modules mount in a unique way—the main unit literally grows to accommodate them. The power supply is on a sliding tray and forms the rear of the main unit. When you slide the power supply away from the main unit, a gap is formed where the expansion modules, which are in their own small cases, can be inserted.

Except for the monitor, a powerful system can be assembled as a single unit. However, there are some inherent disadvantages to all of this neat packaging: the keyboard is permanently attached to the main unit, a fan is required for air movement and independent hardware houses will have the added problem of supplying cases for some of their expansion modules.

### The Main Unit

The micro floppies have a tendency to steal the show, but the main unit,

the SMC-70, is the heart of this system. The SMC-70's vital statistics are listed in Table 1, but a few more observations are in order.

1. The video RAM is mapped into the I/O space, thus freeing the main memory for programs and data. Although accessing the video RAM with I/O instructions may be a little inconvenient, it's still preferable to using up a sizable hunk of main memory.

2. The character font is programmable. The default font uses one-dot descenders. Each character can have blink and inverse attributes set individually.

3. The video display consists of three overlaid planes: characters, graphics and background, with characters having priority.

4. The SMC-70 interfaces directly to two kinds of monitors: monochrome and high-resolution RGB (red/green/blue). A television interface is available, but at \$200 it's a dubious option. Caution: High-resolution RGB monitors are expensive, yet once you see the SMC-70 working with one, it'll be hard to settle for monochrome.

5. Interlaced scanning is used in the

super high-resolution mode (640×400 B/W pixels). This means that the monochrome monitor should have a long decay phosphor to avoid flickering.

6. The keyboard is serviced by an 8041A, a microcomputer that is configured as an intelligent peripheral interface. The 8041A takes care of scanning the keyboard, forming the key code (including the programmable function keys), repeating the key if it is held down (timing is programmable) and interrupting the Z-80A when a key is depressed (optional).

7. A help key is provided, but it is not used in any of the built-in software.

8. The keyboard does not extend far enough for a palm rest.

9. There is an excellent set of edit keys, including four logically arranged cursor keys.

10. The F and J keys have dimples to aid in touch-typing. This is especially important on a computer keyboard where the hands are often leaving the home positions.

11. Cables do not come with the unit, and hard-to-find eight- and 13-pin DIN style plugs are required. (I hope Sony realizes its error in doing this.)

12. The Basic reference manual, the system monitor manual and the hardware reference manual do not come with the unit. Only the operating instructions and two reference charts are included. This is carrying unbundling much too far. It's like buying a computer and finding that the CPU chip is not included. Documentation is a necessary component of any hardware or software product; the product is incomplete and unusable without it.

To add insult to injury, Sony charges \$28 to \$43 for each of the manuals, and they weren't even available until about a month after the system was first marketed. (By the way, the ROM Basic and System Monitor reference charts that do come with the unit are handy.)

13. The ROM Basic supplied with the unit is complete with powerful graphics commands. Sony Basic runs close to the speed of Applesoft.

Since floating point numbers in Sony Basic have 14 significant digits, Sony Basic is a little slower than Applesoft during number crunching. On the other hand, it is a little faster during other processing.

14. The system monitor is surprisingly powerful for a business system. It has a mini-assembler, disassembler and a memory and peripheral test, as well as access given to the monitor's

Circle 209 on Reader Service card.



**MODEL SK7100**

## ABSOLUTELY FREE!

SEE BELOW FOR DETAILS

The most advanced Clamp-On Tester with many performance features

- Range selection knob automatically advances the specific scale
- Easy reading — no confusion
- Accurate AC current measurements assured because of round clamp core with built-in balanced coils
- Most reliable taut band, internal core-magnet meter
- Trigger lock device, retains reading for use in hard to get places
- Automatic "Zero-Adjust" ohms scale
- Overload protection up to 150% for one minute ON ALL RANGES
- Test leads lock-in for added safety
- Accuracy ±3% full scale on all ranges
- Accessories included: carry case, test leads and instructions
- Size: 8.25x3.25x1.4"
- Reasonably priced
- Full satisfaction — money back guarantee.

10 USEFUL RANGES	
AC Current	6, 15, 60, 150, 300, 600A
AC Voltage	150, 300, 600V
Resistance	0-20kΩ(1kΩ center scale)



**Personal Switcher**

Model P01

## POWER SUPPLY

For Lab or Original Equipment

**FEATURES:** Efficient 30 kHz switching frequency • Four Models satisfy most applications • Years of trouble-free service • Each side AC line fuse protected • Tele-Tale LED "Pwr-On" Panel Indicator • Three separate voltage outputs • Metal enclosure provides physical and EMI protection • For experimental use or permanent power source • Soft start feature protects critical circuits • Parallel operation acceptable for higher current needs • Push-in terminals, accept wire or test lead • Light-weight, easy to use • AC line cord permanently attached • Most reliable power source for a variety of uses and applications • 48 hour burn-in assures MTBF of 3½ years, reasonably priced at \$1.90/watt • Full one year guarantee • 2-tone anodized case • Custom volt/current outputs on special order • Input surge protection • Automatic short circuit protection and restoration • UL recognized components • Handy Service Aid

**SPECIFICATIONS:** Input: 90-132VAC, 47-440Hz • Dual AC Input Fuses • Line Regulation: ±0.1% Max. for 10% input change • Load Regulation: ±0.2% Max. on #1 Output • Ripple Noise: Typ. 1% PP Max. • Over Voltage Protection • Reverse Polarity Protection • Compact, only 7½" x 4" x 2¼" • Fast load transient response • 5 volt adj. • ±10% DC Output: 42 Watts continuous • 70% Efficiency

Qty.	Model	Output # 1	Output # 2	Output # 3	Total
PS-1	5V-6A	+12V-0.5A	-12V-0.5A		
PS-2	5V-6A	+15V-0.4A	-15V-0.4A		
PS-3	5V-6A	+12V-0.5A	-5V-1A		
PS-4	5V-3A	+24V-0.6A	-24V-0.6A		

SK-7100 Clamp-On Tester

FREE Clamp-On Tester, with any 4 Units Purchased. NC

Price of any Personal Switcher or SK Tester is \$99.50 each.

Sub-Total \_\_\_\_\_

Mass. res. add 5% Tax \_\_\_\_\_

Shipping & Handling \$4.50 \_\_\_\_\_

**TOTAL \_\_\_\_\_**

CALL TOLL FREE 1-800-343-1455

Within MASSACHUSETTS 1-617-682-6936

**Com inc** 1545 Osgood St. Unit 11AV, No. Andover, MA 01845

Charge to:  MasterCard  Visa  American Express  Check/Money Order

Card # \_\_\_\_\_ Exp. Date \_\_\_\_\_

Name \_\_\_\_\_

Address \_\_\_\_\_

City \_\_\_\_\_ State \_\_\_\_\_ Zip \_\_\_\_\_

Signature \_\_\_\_\_

SCHOOLS—LABS: QUANTITY PRICING (10 or more) ON REQUEST

utility subroutines.

15. The SMC-70 has a built-in clock/calendar with battery back-up.

16. The fan is a bit noisy, but the rest of the unit has a solid feel.

### Micro Floppy Disks

I intended to put an eight-inch drive on my system, but after seeing the micro floppies, I quickly changed my mind. In a word, they're cute! But that's not all. Each disk holds 280K, and the drives are fast and quiet. Imagine carrying a megabyte of data in your shirt pocket.

The micro floppies are 3½ inches in diameter and are in a rigid plastic case. You can slide a shutter over the head access slot when the floppy is not in use. The rigid case and the shutter ensure that the magnetic medium is well-protected.

Whether or not Sony's micro floppy format will become an industry standard is still up in the air.

In response to Sony's introduction of the micro floppy, a number of disk drive manufacturers quickly got to-

gether and formed a standards committee—a well-used technique to take the wind out of the sails of a manufacturer who is early to market with a product. It is interesting to note that Hewlett-Packard has just released its Series 100 and Series 200 computers with the Sony drives. Also, Tandon is planning to produce a compatible drive early next year.

One sour note is that the micro floppy drive does not come with an operating system. CP/M is another \$150. A disk version of Sony Basic is also sold separately for \$150. Although Disk Basic is powerful, it's a bit expensive for a second language. I say "second" because I installed an AMD 9511A math chip and used Forth with it to obtain an impressive performance.

### Software

Sony is after the small-business market, judging from the first available software. Table II lists application software that is available. Apparently all of this software was produced in the U.S. The database management

system is by Condor Computer and the accounting system is by SSG.

As far as languages go, there is a disk version of Sony Basic, and Pilot Plus with CB-80 is planned.

### Conclusion

By getting one of the first machines marketed, I knew I was inviting trouble. So far, that trouble has manifested itself as lack of information and support, but the start-up problems are mostly solved now and there is support. I'm happy to say that I have not flushed out any hardware bugs yet.

The dealers have an "800" number to call for help on technical questions, but Sony's most knowledgeable people tend to be tied up with computer shows and the like. Users with a business configuration of all Sony components running Sony-supplied software will probably have things run fairly smoothly. The technical types, though, will have a few trails to blaze. ■

<b>CPU</b> Processor: Z-80A. Clock: 4.028 MHz. Interrupts: Nonmaskable and mode 1.	control keys; five programmable function keys; one help key (programmable); numeric keypad interface.
<b>Memory</b> Main memory: 64K bytes (64K bit dynamic RAMs). Video RAM: Graphic—32K bytes (16K bit dynamic RAMs). Character—2K bytes (static). Attributes—2K bytes (static). Font table—2K bytes (static). ROM: 32K bytes total. System monitor—9K bytes. Sony Basic—22K bytes. Default font—1K bytes.	Audio cassette: Eight-pin DIN jack; 1200 b/s baud rate; motor on/off switch. Printer interface: 25-pin D connector; TTL level; standard eight-bit parallel transfer. RS-232C interface: 25-pin D connector; TTL level; 75-19,200 b/s baud rate; implemented with 8251A. Speaker: Eight ohms, three levels. Earphone: Minijack. RGB video output: 25-pin D connector; 0V-0.7V, 75 ohms. Monochrome video: Eight-pin DIN jack; 1 Vp-p, 75 ohms, sync negative. Light pen interface: Five-pin DIN jack. Slots for expansion (50-pin connectors): Inside—Two slots. Outside—Three slots. Maximum current—2.4A at +5V. 1.2A at +12V. 70mA at -12V.
<b>Display</b> Character display: 8×8 dot matrix/character; 80 or 40 characters by 25 lines; eight colors. Border area: 16-color. Output interface: Color—RGB analog signal; TTL level separate sync signal; composite sync signal. Monochrome—Composite video signal, TTL level separate sync signal.	Expansion unit: 50-pin connector.
<b>I/O Interface</b> Keyboard: Encoded with 8041A micro-processor; 72 keys; eight display	<b>General</b> Dimensions: 14.5×3.625×17.5 inches. Weight: Ten pounds, nine ounces. Price: \$1475—main unit; \$650—single micro floppy drive; \$1100—dual micro floppy drives; \$5—micro floppy disk; \$90—numeric keypad.

Table 1. Complete list of the Sony SMC-70 specifications.

<b>Operating System</b> CP/M.
<b>Languages</b> Sony Basic. Sony Disk Basic. CB-80. Pilot Plus.
<b>Word Processing</b> Letterwriter. Word Processor. Spelling Checker. Mail List. W/P Math.
<b>Spreadsheet</b> VisiCalc. SuperCalc.
<b>Accounting System</b> General Ledger. Accounts Payable and Receivable. Order Entry System. Payroll System. Inventory Control System.
<b>Database Management</b> Record Management System. Report Generator. Database Management System.
<b>Communications</b> On-line Communications (IBM 3275). Batch Communications (IBM 2780/3780). TWX and Database Access System. Multiterminal Emulation.

Table 2. Software for Sony's SMC-70.

# TS-1000 Printing Power For Under \$100

*Timex-Sinclair's lowest-priced printer for the lowest-priced micro is no toy. It can capably do the job—and at a fraction of the cost.*

By Jim Stephens

**T**imex-Sinclair has done it again! Its new, redesigned model 2040 printer far exceeds its first model. But, more importantly, the printer is available on a local basis. That is, you can buy it at your local Timex-Sinclair distributor. Will wonders never cease!

Sinclair Research promised a printer when the early ZX-80 units were introduced in 1980. They even were able to market a small printer overseas and in Canada. The first one had several handicaps, such as four-inch, metallic-coated paper that had a tendency to jam if it was allowed to stay in one spot on the roller for a long time.

The few units that were "smuggled" into the U.S. were not impressive, but they were the only ones available that responded directly to the ROM's printer commands. However, they did do a reasonable job of printing. I waited a year for my printer, mainly because I couldn't get one locally. Now that I have my new model, the wait was worth it.

I had seen a few advance photos of the new 2040 printer, but based on past experience I felt that production probably was still about six months away. To my surprise, a full-page newspaper advertisement featured

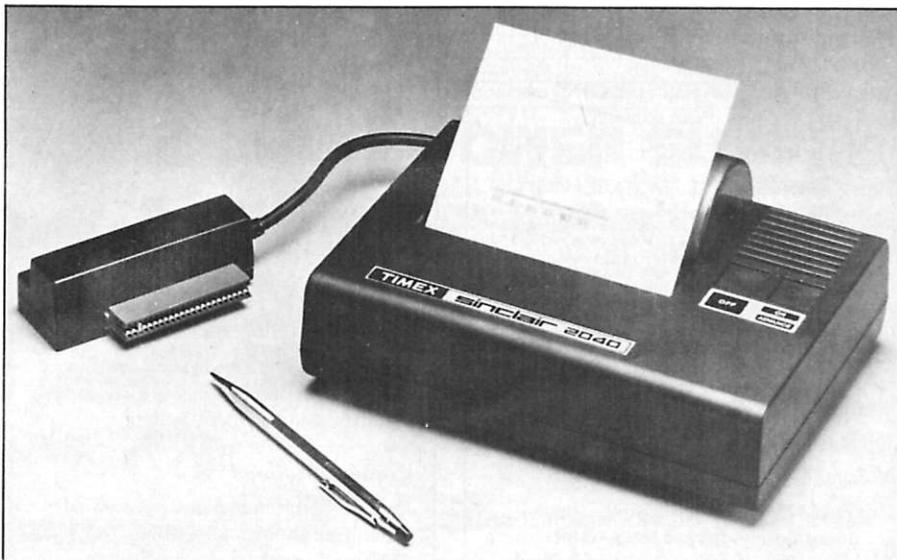
the printer at a discount price a week after I had seen the advance photos!

Still unbelieving, I called the store. Sure enough, they had more than 20 units in stock at this one outlet alone. I was there when the doors opened. The clerk said she had sold four printers the night before the advertisement came out.

## Sizing Up the 2040

The 2040 measures 2×6×8 inches of black, high-impact plastic, and is larger and heavier than the original Sinclair printer. It weighs in at about three pounds, which is almost three times the size and weight of the original model. The printer comes with its own heavy-duty power supply. This supply is a wall-plug-type unit similar to the original supply, but is more than twice as large. Output of the supply is 24 volts at 1.2 amps!

The best news is that the paper is now white, 4¼ inches wide, and of a high-quality thermal specification. The earlier model had a width of only four inches and looked like burnished aluminum. The new paper is supplied in lengths of 85 feet, and fits into a paper tray that is covered with a clear plastic dust cover. The new model is similar to the design of the CAI P40 printer, which is also made for the ZX-81 and TS-1000 computers.



*The Timex-Sinclair 2040 thermal printer is compatible with both the TS-1000 and TS-2000 computers. The printer incorporates a dot matrix print mechanism with full graphics and text capability. It prints two lines per second in two modes.*

*Address correspondence to Jim Stephens, 2324 Denwood Drive, Nashville, TN 37214.*

The print is 5×7 dot matrix of 32 columns. The print speed is a quiet two lines per second, which means a full screen can be copied in less than 12 seconds. Unlike the CAI P40 printer, the 2040 responds to the LPrint, LList, and copy commands directly rather than having to peek and poke memory locations for operation. The new unit has its own power on/off switch and, by holding the on switch down, the paper can be advanced manually. This is a necessary function, since the last line is below the tear-off blade when the last print line is completed.

## Operation

The new printer comes supplied with an eight-inch, permanently attached interface cable terminated with a backplane connector/adaptor. The connector is attached to the backplane of the TS-1000, and the 16K RAM Pack is connected to the adaptor by a feed-through extension board. The RAM Pack now will be located about two inches away from the computer, making it appear rather unstable. However, no problems have been experienced with wobble.

For operation, the printer cable is attached to the TS-1000, and the power supply is plugged into the wall outlet. The printer is switched on manually and the TS-1000 is activated. It doesn't appear to matter if one of the units is off. I accidentally left my printer on for 24 hours and it still operated perfectly the next day. The two regulators under the vents on the right side were giving off plenty of heat. I suppose the extra power supply is necessary, but I detest all that extra cable cluttering up my limited work space.

The documentation is brief, but adequate. A built-in test procedure is described to determine if the printer is operational. This test consists of holding down both the on and the off switch simultaneously. This automatically prints alternate rows of "eights" and "ones." If this test is passed, you are ready to use the printer commands.

LPrint is a direct command that prints all of the characters in the quotation marks following the LPrint. LList lists out your entire program unless the break key is pressed. The copy command copies the entire screen to the printer. All of the screen will be printed even though there may be just one line on the display. This can waste a lot of paper. When the printer is operating, the display acts as though it is in the Fast mode. This can

be irritating, but I'm usually watching the printer do its thing.

The instruction manual gives most of its attention to explaining what to do to avoid damage to the printhead. The use of the supplied paper is highly recommended, since low-quality paper can "ruin the print head in a few minutes of operation," according to the instructions. Apparently, the printer is fragile and subject to serious damage if the printhead is jammed.

I had one moment of panic when I first turned on the printer and my TS-1000. The display was not operating! All I could get were skewed

---

The 2040 is by far  
the best printer for the price.  
It is small and portable.  
It operates with  
the ROM Basic  
and never misses a character.

---

black lines. After I quickly turned everything off and reconnected all of the plugs, no further problems were experienced.

Watching this little printer quietly spew out characters at two lines per second is fascinating. I had expected a horrible churning noise like that of my little printing calculator, which sounds like it's grinding coffee. The 2040 printer is hardly audible over the sound of the television, the kids, the dogs and the radio.

The instructions end abruptly with a short paragraph describing how to use the printer with machine code. They describe how the use of the Out instruction activates and deactivates the printing mechanism. This information will be useful for machine-code programs that require the use of the printer.

## A Look Under the Hood

Even though it probably voided the warranty, I could not resist a peek at the mechanics of the most inexpensive printer on the market.

After carefully removing the four retaining screws, I lifted off the bottom, which exposed an excellent-quality printed circuit board made by

Alphacom. The fact that it was produced in Taiwan was not discouraging, since almost everything seems to be made in Taiwan these days.

Turning the board over reveals a sparsely populated component side that's dominated by one 40-pin 8224IC and four others of unknown makes and models. I was most interested in the printhead mechanics, and explored this to the greatest extent.

The printing is done by a pair of two-inch stationary thermal wafers connected to the board by ribbon cable. The wafers, which are flat against the platen, do move from side to side about 1/8-inch to produce the characters' horizontal dots. The movement is driven by a plastic cam, and return is done by a spring. This will be a large source of wear. The roller and wafer cam are driven by nylon gears that have sensing switches for information on starting and stopping.

There is a lot of metal within the case, but it is mainly die-cut and stamped, with no machined or precision parts. This would explain the low cost. I see no reason why this mechanism would not last for thousands of cycles, other than the one nylon cam used to shift the printing wafers.

With little movement of the heads, no impact printing and the use of quality paper, this little printer should easily last the life of the system.

## The Best for the Bucks

The 2040 is by far the best printer available for the price. Actually, at \$99.95, it's the only printer for its price! It is not full-size, will not print on plain paper, and can't compare to the more sophisticated units, but it does the job at a fraction of the cost.

With the printer, your computer can now do many things like the big ones. It can generate hard copy that can be used for such things as records management and program debugging, and, with the proper formatting, it can even be photocopied using two 11-inch lengths for a complete 8½×11-inch printed report. The thermal print photocopies beautifully.

The extra power supply is a nuisance, and it lacks the frills of a software-controlled power on and off. These shortcomings, however, can be overlooked when you consider the advantages. It is small and portable. It operates with the ROM Basic and never misses a character. It is quiet and the cost can't be beat. Now where is that minidisk?■

# Increase the Power of CP/M

*If you find CP/M lacking in some areas, this utility program will provide more power to you and your micro.*

By Mitchell K. Hobish

The way that CP/M has virtually taken over the marketplace for eight-bit microcomputer operating systems testifies to its utility. However, it's been described as less than "user friendly"—a capacity to which all interfaces between the human operator and the computer should aspire.

For this reason, several "front ends" (programs that provide some of the capabilities CP/M *should* provide) have appeared.

Microshell and Unica supply several CP/M enhancements, many of which emulate Bell Labs' Unix, the sometime-contender in best-operat-

ing-system debates. While these programs allow greater flexibility in terms of program redirection, they do not allow access to all the data extant on a disk. And, while Microshell, for example, supports multiple statements on a command line, it's still rather cumbersome if you want to copy several files from one disk to another.

## More Power to You

Enter another utility program: Power, from Computing! (2519 Greenwich, San Francisco, CA 94123). Actually, this is a package of about 50 programs. Since the cost of the package is \$169, it averages out to a little over \$3 per utility program—quite a bargain when you consider what you get. (See Table 1 for a listing of the available commands.) The program requires about 12K of memory, beginning at 100H.

Most of the programs supplied are menu-driven. That is, entry of one of the utility names at the keyboard (for example, Copy) results in a listing of all the files, each with an associated number, on a disk to the CRT. Merely enter the number of a file (or series of files, or all the files) and the destination drive, and the files will be transferred appropriately. Try that with PIP!

The same format holds for utilities that rename files, reclaim previously erased files, load, generate the size in sectors and bytes, and on and on. The

Command	Function
CHECK	Calculate checksum for file.
COPY	Transfer files.
DIR	Directory listing.
DISK	List disk parameters.
DS	Display and substitute hex code.
DUMP	Dump ASCII code.
DUMPA	Dump formatted ASCII code.
DUMPH	Dump formatted HEX code.
DUMPX	Dump formatted ASCII and HEX code.
ERA	Erase files.
EX	Execute program at given address.
EXIT	Leave Power, go to designated address.
FILL	Fill memory with specific HEX byte.
GO	Load a program and execute.
GROUP	List group numbers of files.
JP	Jump to entered address.
LOAD	Load a file to given memory address.
LOG	List POWER default settings.
MOVE	Move block of memory.
READ	Read track and sector.
READGR	Read group.
RECLAIM	Recover erased files.
REN	Rename files.
RUN	Run a .COM program from Power.
SAVE	Save file from memory.
SEARCH	Search memory for string.
SETDIR	Set file to \$DIR.
SETRO	Set file to \$R/O.
SETSYS	Set file to \$SYS.
SETWR	Set file to \$R/W.
SIZE	List size of file in sectors and bytes.
SPEED	Set display speed.
STAT	List free and used disk space.

Table 1. Listing of commands available with Computing!'s Power utility program.

Address correspondence to Mitchell K. Hobish, Laboratory of Chemical Evolution, University of Maryland, College Park, MD 20742.

use of numbered files helps alleviate the problem of typing erroneous file names; it also saves time.

Power also allows you access to specific sectors and groups on the disk. This data may be loaded into memory at a location you designate, and may then be operated on appropriately, before being written back out to disk to the sectors you designate. Never before has the CP/M operator had such control over information stored on disk.

### Useful Commands

One of the commands I find particularly useful is TEST. This nondestructive test of the designated disk results in generation of a unique CRC checksum. It also identifies bad sectors and collects them into a special file, designated SYSTEM. Hence, it's invisible to the user; CP/M will never attempt to write data to those sectors. This is an excellent way to assess the integrity of any disk—especially those that may appear to be "bargains."

Another aspect of the user-friendliness of the package is the way it allows use of the semicolon as a delimit-

ter between disk drive designation and file name (e.g., B;filnam.ext).

Furthermore, I have always found onerous the necessity of shifting to get an asterisk, unshifting to get a period, and then shifting again to get another asterisk for complete wild-card commands. Well, Power takes care of that,

---

Never before has  
the CP/M operator  
had such control over  
information stored on disk.

---

too; a simple "\*\*\*" will be handled appropriately.

There are times when I would prefer differently formatted directory listings and other customizations. Through the Log command, the user can set any number of columns for listings, set Page on or off (number of lines read out to CRT before stopping and waiting for a <CR> for the next page) and show system files in direc-

tory listings. Even the speed at which text is read out to the CRT may be varied.

And there's more... Through the DS command, several bytes of Power's code may be modified to provide more customization: prompt-changing, sign-on message, characters per line and so on. The modified code may be saved to disk for a completely personalized package.

### Pick Up Power

The only area I found even slightly lacking was the documentation; the manual in the package I received had blurred and missing letters. I would have preferred that the manual be oriented vertically along the long axis of the page instead of the horizontal format currently supplied. In addition, I found that several of the commands were incompletely documented.

These are mere quibbles, however. I would strongly recommend this powerful addition to CP/M for anyone who wants more access to files on the disk. Power is an excellent buy—made even better by a money-back guarantee. ■

Circle 72 on Reader Service card.

## ATTENTION EPSON OWNERS

The ultimate solution to your paper storage problem at an unbelievable low cost of only **\$9.95 SET**



**MX-70/80**                      **MX-100**

### PRINTER-STILTS™ PRINTER SUPPORTS

FEATURES:

- Precision machined from solid aluminum.
- Tilted to facilitate viewing paper.
- Soft rubber feet to absorb vibration.
- Easy to install — no tools required.
- O-rings insure tight fit into printer recesses.
- Natural finish aluminum.

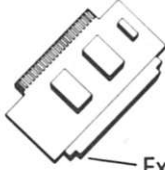
Available in two styles (MX-70/80 or MX-100) for your EPSON Printer. The MX-70/80 style also fits the IBM & TI PC dot matrix printers and the HP82905B printer. To order **PRINTER-STILTS™** printer supports give style and send \$9.95 plus \$2.00 postage/handling or call today: Louisiana residents add 5% tax.

**DATATEK INC.**  
Dept. 200  
P.O. Box 5956  
Shreveport, La. 71135  
(318) 868-0068 or 868-2241 til 10 PM CST



Circle 60 on Reader Service card.

## NEW VIC RABBIT CARTRIDGE AND CBM 64 RABBIT CARTRIDGE



**\$39.95**  
(includes Cartridge and Manual)

"High-Speed Cassette Load and Save!"

Expansion Connector on the VIC Cartridge

"Don't waste your Life away waiting to LOAD and SAVE programs on Cassete Deck."  
Load or Save 8K in approximately 30 seconds! Try it — your Un-Rabbitized VIC takes almost 3 minutes. It's not only Fast but VERY RELIABLE.  
Almost as fast as VIC Disk Drive! Don't be foolish — Why buy the disk when you can get the VIC Rabbit for much, much less!  
Easy to install — it just plugs in.  
Expansion Connector on rear.  
Works with or without Expansion Memory.  
Works with VIC Cassete Deck.  
12 Commands provide other neat features.  
Also Available for 2001, 4001, and 8032

## Eastern House

3239 Linda Dr.  
Winston-Salem, N.C. 27106  
(919) 924-2889 (919) 748-8446



# How to Succeed in Business (with a little help from Apple)

*Learn the secret of success in business with this Apple II program that analyzes your company's sales and cost figures.*

By Gregory Glau

One of the best things businessmen can do is to analyze profit projection figures. It's really helpful to get a line on what may happen to your sales/costs/profits.

Unfortunately, many of us look in only one direction: forward. It seems that most businessmen are optimists, so we naturally expect sales to grow, assume our profits will rise right along with them and think we'll cut overhead to the bone at the same time.

## Look in Both Directions

The Break-even program (Listing 1), written for the Apple II, allows the businessman to look at his sales and cost figures from *both* directions. In other words, it allows us to simulate the bottom line (profit or loss) for our business—based on conditions that we set. And more than that, it prints out a range of amounts developed from this simulation criteria.

We control not only the starting figures for sales and costs, but also the way in which these figures will vary for any particular simulation we'd like to see. Our simulation also can cover any time period we wish. We can use year-end figures, amounts for one month or for one quarter or whatever.

To keep things simple, we'll work with three amounts: Sales, Direct Costs and Overhead.

The Sales category is just that—your total sales for whatever period you want to use.

Direct Costs deals with the labor and materials involved in producing whatever you sell. In a manufacturing business, Direct Costs would include

the raw materials, plus the labor required to create the product, plus any other costs directly associated with the production of the items. In a retail business, Direct Costs consists of material purchases, freight and any labor directly connected with selling your product.

Overhead involves expenditures such as administrative salaries, book-keeping costs, insurance expenses, utilities and office supplies. Overhead, simply, is what it costs just to open your doors every day, whether or not you sell anything.

A financial statement normally will total and list Sales, Direct Costs and Overhead. When you deduct your costs from your sales, what's left is

your net profit.

We'll keep things simple and work with only these totals because we want to create a clear simulation of what will happen to them if your business gets better or worse—if sales and costs increase or decrease.

All of this is based on your own data, of course. The program's print-out lists your sales along the top and your costs up and down the left side. Remember, though, that we want to see a range of values, so your starting figures will be in the middle of the others.

For example, say you enter \$400,000 for your sales figure. Let's also assume you want to see a four percent range for your sales. Here's what the top line of your printout will show:

Sales Volume →  
353894 368640 384000 400000 416000  
432640 449945

Notice your \$400,000 starting sales figure in the middle of the other amounts.

For your sales, the printout shows your starting figure, plus three dollar values for increased sales and three figures for decreased sales, all according to the percentage you select.

To the right of your starting figure, sales are increased by four percent each year. These amounts are cumulative, percentage-wise; each year they are four percent higher than the previous year.

To the left of your starting figure,

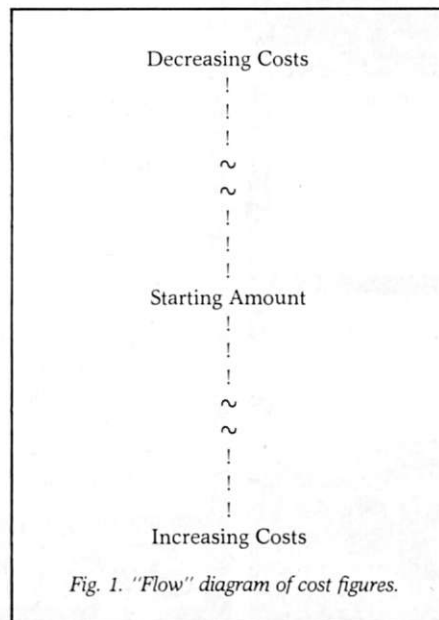


Fig. 1. "Flow" diagram of cost figures.

Address correspondence to Gregory Glau, PO Box 1627, Prescott, AZ 86302.

# FRANKLIN'S BAKER'S DOZEN!



## 13 Good Reasons to Buy the **ACE1200**

1. **Apple® II-compatible**
2. **CP/M®-compatible**
3. **128K of RAM**
4. **Built-in floppy disk drive**
5. **Disk controller**
6. **80 column card**
7. **Serial interface**
8. **Parallel interface**
9. **Upper and lower case**
10. **VisiCalc® keys**
11. **Cursor control pad**
12. **Numeric pad**
13. **Auto repeat keys**

Extras can more than double the price of your personal computer. Not so with the Franklin ACE 1200. It's the professional computer system that includes the extras—and a long list of exclusive Franklin features that make it the most extraordinary value on the market today.

The ACE 1200 has everything you'll need to add a color or black and white monitor, modem, printer, back-up disk drive and other accessories. You can choose from the enormous selection of Apple programs and peripherals because the ACE 1200 is hardware- and software-compatible with

the Apple II. And, with the built-in CP/M card, you can run both Apple II and CP/M programs. Franklin's CP/M operates three times as fast as many competing systems, drastically reducing processing time for most business applications.

The Franklin ACE 1200—the most extraordinary value on the market today. Call or write today for the name of your local authorized Franklin dealer.

Franklin ACE is a trademark of Franklin Computer Corporation. Apple is a registered trademark of Apple Computer Inc. CP/M is a registered trademark of Digital Research Inc. VisiCalc is a registered trademark of Visi Corp.



**FRANKLIN**  
COMPUTER CORPORATION

2138 Route 38, Cherry Hill, NJ 08002 609-482-5900 Telex: 837-385

Circle 59 on Reader Service card.

Listing 1. Break-even program for the Apple II.

```

100 DIM C(21): REM THIS DIMENSIONS OUR COST FIGURE TABLE
110 HOME : PRINT : GOSUB 11000: PRINT
120 PRINT
130 PRINT "ANSWER 1 TO SKIP THE INSTRUCTIONS"
140 PRINT "AND GO RIGHT TO THE DATA-ENTRY AREA."
150 PRINT
160 PRINT "ANSWER 2 TO SEE THE INSTRUCTIONS"
170 PRINT "AND THEN ENTER DATA"
180 PRINT : PRINT "ANSWER 3 TO STOP NOW.": PRINT
190 INPUT Q
192 IF Q = 1 THEN 1000
194 IF Q = 2 THEN 200
196 IF Q = 3 THEN PRINT "END OF PROGRAM": END
198 GOTO 110
200 HOME : PRINT : GOSUB 11000: PRINT
210 PRINT "THIS PROGRAM WILL ALLOW YOU TO ENTER"
220 PRINT "YOUR SALES VOLUME AND COST DATA, AND"
230 PRINT "THEN WILL PRINT A CHART FOR YOU, BASED"
240 PRINT "ON YOUR OWN INFORMATION."
250 PRINT
260 PRINT "TO KEEP THINGS SIMPLE, WE"
270 PRINT "ONLY ENTER YOUR TOTAL SALES VOLUME, AND"
280 PRINT "THE TOTAL OF YOUR 'COST OF SALES.'"
285 PRINT
290 PRINT "THIS INCLUDES YOUR DIRECT COSTS, SUCH"
300 PRINT "AS LABOR AND MATERIALS. YOU ARE ALSO"
310 PRINT "ASKED TO ENTER YOUR OVERHEAD COSTS."
320 PRINT
330 PRINT "THE TOTAL OF THESE TWO - DIRECT COSTS"
340 PRINT "AND OVERHEAD COSTS - WILL BE"
350 PRINT "YOUR TOTAL 'COST OF SALES.'"
355 PRINT : GOSUB 11000: PRINT
360 PRINT "HIT ANY KEY TO CONTINUE..."
370 GET A$
380 HOME : PRINT : GOSUB 11000: PRINT
390 PRINT "YOUR SALES DATA WILL BE PRINTED IN"
400 PRINT "SEVEN COLUMNS, WITH THE INITIAL "
410 PRINT "'STARTING SALES' IN THE CENTER. THEN,"
420 PRINT "SINCE YOU WERE ASKED FOR A PERCENTAGE"
430 PRINT "INCREASE/DECREASE FIGURE, THE COLUMNS"
440 PRINT "TO THE RIGHT AND LEFT OF THE 'STARTING "
450 PRINT "SALES COLUMN WILL SHOW THIS AMOUNT"
460 PRINT "INCREASED (RIGHT SIDE) AND DECREASED"
470 PRINT "(LEFT SIDE) BY THE PERCENTAGE YOU"
480 PRINT "SPECIFIED."
490 PRINT
500 PRINT "THIS IS A CUMULATIVE PERCENT, BY THE"
510 PRINT "WAY, SO YOU CAN EASILY SEE WHAT "
520 PRINT "HAPPENS TO YOUR SALES IF THEY INCREASE"
530 PRINT "OR DECREASE BY THE % YOU ENTERED."
540 PRINT : GOSUB 11000: PRINT
545 PRINT "HIT ANY KEY TO CONTINUE...": GET A$
550 HOME : PRINT : GOSUB 11000: PRINT
560 PRINT "YOUR COST INFORMATION IS TOTALED."
570 PRINT "YOU ARE ALSO ASKED TO ENTER A PERCENT"
580 PRINT "THAT YOU'D LIKE TO SEE YOUR COSTS"
590 PRINT "INCREASE AND DECREASE."
600 PRINT
610 PRINT "THIS INFORMATION IS PRINTED AS THE"
620 PRINT "LEFT-HAND COLUMN OF YOUR PRINTOUT,"
630 PRINT "AND - LIKE THE SALES FIGURES - HAS "
640 PRINT "YOUR STARTING 'COST' AMOUNT IN THE"
650 PRINT "CENTER OF THE NUMBERS, WITH THE TOTALS"
660 PRINT "FOR YOUR COSTS INCREASING (GOING DOWN"
670 PRINT "THE CHART) AND DECREASING (GOING UP"
680 PRINT "THE CHART) ACCORDING TO THE PERCENT"
690 PRINT "YOU SPECIFIED TO START WITH."
700 PRINT : GOSUB 11000: PRINT
705 PRINT "HIT ANY KEY TO CONTINUE...": GET A$
710 HOME : PRINT : GOSUB 11000: PRINT
710 PRINT "BY READING ACROSS FOR SALES AND THEN"
720 PRINT "DOWN THE LEFT-HAND COLUMN FOR COSTS,"
730 PRINT "YOU CAN DETERMINE YOUR PROFIT OR LOSS"
740 PRINT "BASED ON WHAT MAY HAPPEN TO YOUR "
750 PRINT "BUSINESS."
765 PRINT : GOSUB 11000: PRINT
770 PRINT "HIT ANY KEY TO START...": GET A$
1000 HOME : PRINT : GOSUB 11000: PRINT : PRINT

```

More →

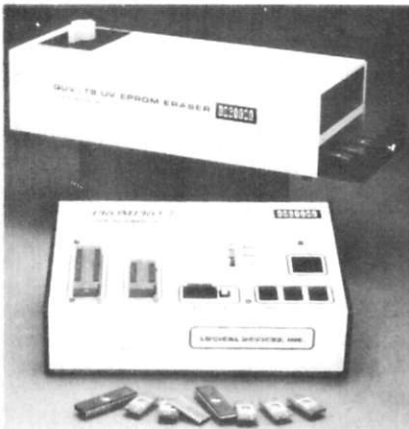
Circle 373 on Reader Service card.

# UV EPROM ERASER

- \* Erases over 15 EPROMS - 15 minutes erase time
- \* Element life 7700 hours
- \* Intensity: 12Ws 1/2cm<sup>2</sup> at 1"
- \* Erases all UV EPROMS (2716, 2732, 2516, 2532, etc.)

**\$49.95\***

\*HOBBY MODEL



**INDUSTRIAL MODEL**  
 QUV-T8 / 2N  
**\$68.95**  
 WITH TIMER AND  
 SAFETY SWITCH  
 QUV-T8 / 2T  
**\$97.50**

## INTELLIGENT PROGRAMMER STAND ALONE RS-232

- \* RELIABLE
- \* EASY COPY (No external equipment needed)
- \* USER FRIENDLY

COMPATIBLE:  
 IBM PC, TRS-80, APPLE, CPM,  
 FLEX, TEKTRONICS, MDS

**(MCS-48)**  
 PROGRAMMING  
 PRICE INCLUDES  
 PERSONALITY MODULE  
**\$489.00**

PROGRAMS 2508, 2516, 2532, 2716, 27C16, 27C32,  
 2732, 2732A, 2758, 8748, 8749H, 8748H  
 OPTIONAL MODULES: 2564, 2764, 8755A, 8741  
 \* STAND ALONE, CRT, OR COMPUTER CONTROL  
 \* UPLOAD/DOWNLOAD IN MOTOROLA OR INTEL HEX FORMAT  
 \* MICROPROCESSOR BASED \* 4 K INTERNAL RAM  
 \* 90 DAY PARTS & LABOR WARRANTY ON ALL PRODUCTS

SOON TO BE RELEASED:  
**PROMPRO-8 128K Version \$689.**  
**MONEY BACK GUARANTEE**

### LOGICAL DEVICES INC.

781 W. OAKLAND PARK BLVD. • FT. LAUDERDALE, FL 33311  
 Phone Orders (305) 974-0967 • TWX: 510-955-9496

Circle 140 on Reader Service card.

## OMNITEK COMPUTERS INTERNATIONAL, INC.

1300 MAIN STREET TEWKSBURY, MASS 01876  
 617-851-4580

- \*CBM64.....295.00
- Verbatim 5.25" D.L.....25.00
- 5 1/2" Head Cleaning Kits.....5.00 each or 3 for \$12.00
- Okidata Microline 80.....299.00
- Okidata Microline 82A.....399.00
- Okidata Microline 83A.....629.00
- Okidata Microline 92 (160 C.P.S.) corresponds mode.....499.00
- Okidata Microline 93.....799.00
- Smith Corona Daisy Wheel Printer.....559.00
- 13" Green Monitor.....99.00
- B.M.C. 13" Color Monitor.....299.00
- Epson FX80 FT.....539.00
- Epson MX-100.....629.00
- Radio Shack MIV 64K w/2 drives and RS232.....1787.00
- 40 track economy drive Power Supply with case.....179.00
- Tandon drives with Power Supply and case
- 40 track singlehead.....249.00
- dualhead.....339.00
- 80 track singlehead.....299.00
- dualhead.....399.00
- 5.25" Power Supply and case.....39.00 or 10 for \$340.00
- \*BASF 40 track D.D. 5 1/4" new disk drive, as is,  
 no return.....89.00
- 8" Power Supply and case.....99.00 or 10 for 935.00
- Brothers HR-1 D.W. Printer.....795.00
- Full Commodore Line .....CALL.....VIC 20.....99.00

## OMNITEK COMPUTERS INTERNATIONAL, INC.

TRS-80 is a reg. trademark of Tandy Corp. Prices are for mail order only TERMS:  
 Check, money order, Mastercard and Visa accepted. F.O.B. Tewksbury-freight extra. Minimum \$5.00 S & H. Mass residents add 5% sales tax. Write for FREE CATALOG.

Listing continued.

```
1002 PRINT "YOUR STARTING FIGURE MUST BE A ": PRINT "POSITIVE AMOUNT.": PRINT
: PRINT
1003 INPUT "STARTING SALES VOLUME ";S(4)
1004 IF S(4) < = 0 THEN 1000
1006 HOME : PRINT : GOSUB 11000: PRINT : PRINT
1007 PRINT "REMEMBER TO ENTER THE PERCENT"
1008 PRINT "INCREASE/DECREASE AS A DECIMAL."
1009 PRINT : PRINT "FOR INSTANCE, 5% SHOULD BE"
1010 PRINT "ENTERED AS .05, WHILE 10%"
1011 PRINT "SHOULD BE ENTERED AS .10."
1012 PRINT : PRINT "1, THEN, IS THE SAME AS 100%."
1013 PRINT "THIS IS THE LARGEST YOU CAN GO.": PRINT
1015 PRINT "WHAT'S THE PERCENT"
1020 INPUT "INCREASE/DECREASE FOR EACH PERIOD ? ";PS
1030 IF PS > 1 THEN 1006
1035 IF PS < 0 THEN 1006
1040 HOME : PRINT : GOSUB 11000: PRINT : PRINT
1100 INPUT " DIRECT COSTS ";D
1150 HOME : PRINT : GOSUB 11000: PRINT : PRINT
1200 INPUT "OVERHEAD COSTS ";O
1210 C(11) = O + D: REM C(11) IS OUR TOTAL COST TO START WITH
1215 IF C(11) > = 1 THEN 1220
1216 HOME : VTAB 5: PRINT "YOUR DIRECT COSTS + OVERHEAD"
1217 PRINT "COSTS TOTAL LESS THAN ZERO."
1218 PRINT : PRINT "THEY MUST BE A POSITIVE FIGURE.": PRINT
1219 PRINT "ENTER ANY KEY TO START OVER "; GET A#: GOTO 1040
1220 HOME : PRINT : GOSUB 11000: PRINT : PRINT
1230 PRINT "TO GET YOUR COST FIGURES TO GO"
1240 PRINT "UP AND DOWN THE SIDE OF YOUR PRINTOUT,"
1250 PRINT "WE NEED TO ENTER THE PERCENT YOU'D"
1260 PRINT "LIKE TO SEE THESE FIGURES "
1265 PRINT "INCREASE/DECREASE."
1270 PRINT : PRINT
1280 PRINT "REMEMBER, PLEASE ENTER THIS FIGURE AS"
1282 PRINT "A DECIMAL (10% = .10, 4% = .04, ETC.)"
1284 PRINT : INPUT "PERCENT INCREASE/DECREASE ";PC
1290 IF PC < 0 THEN 1220
1295 IF PC > 1 THEN 1220
1300 REM : NOW WE HAVE TO FIGURE SALES
1310 REM : AMOUNTS USING THE VARIABLES
1320 REM : S(1) THRU S(7)
1330 REM : S(4) WILL BE THE INPUTTED SALES AMOUNT
1340 S(5) = (1 + PS) * S(4)
1350 S(6) = (1 + PS) * S(5)
1370 S(7) = (1 + PS) * S(6)
1380 REM : NOW WE HAVE TO DECREASE THE AMOUNTS
1390 S(3) = (1 - PS) * S(4)
1400 S(2) = (1 - PS) * S(3)
1410 S(1) = (1 - PS) * S(2)
1500 REM : NOW WE HAVE TO FIGURE THE AMOUNTS
1510 REM : FOR THE COST FIGURES.
1520 REM : C(11) IS OUR STARTING COST TOTAL
1530 REM : LET'S FIGURE THE INCREASED COSTS FIRST
1600 FOR Y = 12 TO 21
1610 C(Y) = (1 + PC) * C(Y - 1)
1620 NEXT Y
1630 REM : NOW LET'S FIGURE THE DECREASED COSTS
1640 FOR Y = 10 TO 1 STEP - 1
1650 C(Y) = (1 - PC) * C(Y + 1)
1660 NEXT Y
1700 REM : CONVERT TO INTEGERS
1710 FOR X = 1 TO 7
1720 S(X) = INT (S(X))
1730 NEXT X
1740 FOR Y = 1 TO 21
1750 C(Y) = INT (C(Y))
1760 NEXT Y
2000 REM : NOW WE WILL PRINT ALL THE DATA
2010 HOME : PRINT : GOSUB 11000: PRINT : PRINT
2015 PRINT "ENTER TODAY'S DATE, PLEASE:"
2016 PRINT : INPUT E#
2017 PRINT
2020 PRINT "ANSWER 1 TO PRINT..."
2030 PRINT " (TURN ON YOUR PRINTER)"
2040 PRINT : PRINT "OR 2 TO STOP NOW."
2045 PRINT
2050 INPUT Q
2060 IF Q = 2 THEN PRINT "END": END
2070 IF Q > 2 THEN 2000
2080 IF Q < 1 THEN 2000
2100 REM : PRINTING SECTION
2110 D$ = CHR$ (4)
2120 PRINT D$;"PR#1"
2130 PRINT ""
2140 PRINT " >>>>> BREAKEVEN CHART <<<<<<"
2150 PRINT " ": PRINT " "
2152 PRINT "THIS REPORT WAS PRINTED ON ";E#;"."
2153 PRINT " ": PRINT "TO READ THIS CHART, FIND THE SALES FIGURE YOU WANT
ON THE"
2155 PRINT "TOP LINE, AND THEN LOOK DOWN THE LEFT SIDE, THE COST LINE, UN
TIL YOU"
2157 PRINT "FIND THE COST YOU WANT. THE FIGURE WHERE THE LINES"
2159 PRINT "INTERSECT IS YOUR PROFIT OR LOSS BASED ON YOUR SALES AND COST
S."
2160 PRINT " ": PRINT " ": PRINT "THE INITIAL SALES VOLUME IS $ "; INT (S
(4));" ."
2170 PRINT "THE RATE OF INCREASE/DECREASE IS "; INT (100 * PS);" %."
2175 PRINT " "
2177 PRINT "THE TOTAL STARTING COSTS ARE $ "; INT (C(11));" ."
2180 PRINT "AND THE COSTS ARE INCREASING/DECREASING"
2190 PRINT "AT THE RATE OF "; INT (100 * PC);" %."
```

Microcomputing welcomes conversions of this program for the Commodore, Heath, IBM and other popular microcomputing systems.

the amounts decrease by four percent each year. The figure immediately to the left of your starting amount is four percent lower than the \$400,000 figure, the next is four percent less and so on.

So what you see is a range of values, based on your starting sales figure and the percent amount you enter to increase and decrease this amount.

If you want to see your sales increase/decrease at a 50 percent rate, or a three percent rate, the program will do it for you.

So for sales, you're asked for the starting sales amount, then the percentage of increase/decrease in decimal form (you enter ten percent as .10, 14 percent as .14, and so on).

### Totaling Costs

Cost amounts are handled the same way. Since there are basically two things that make up your total cost of sales—direct costs and overhead expenses—the program asks for each one individually. It then totals them for you.

Then, the program will ask what percent you want to see your costs increase/decrease, as they run up and down the left side of the printout. Again, enter this as a decimal. This percent does not have to be the same as you use for sales. It can be whatever you want, depending on the simulation ranges you want to see.

The maximum you can enter is 100 percent (expressed as a decimal, this is 1). The program also won't allow you to enter a negative percentage figure, as it goes up and down anyway with your final figures. You can eliminate these restrictions, if you wish, by deleting lines 1030, 1035, 1290 and 1295.

Again, your starting value will be in the center of the other amounts. The figures going up from your starting amount show your total costs decreasing. The amounts going down from your starting figure show your total costs increasing (see "flow" diagram in Fig. 1).

The program also asks you for the current date; then it's ready to print.

If you make an error as you enter something, just continue on to the printing part; you can stop there and start over.

Your printer should be in slot 1; you can change this at line 2120 if your printer resides elsewhere. (See Table 1 for a list of the variables the program uses. See Fig. 2 for a complete printout of a simulation; the starting amounts and percentage increase/decrease fig-

More →

Listing continued.

```
2195 PRINT : FOR Q = 1 TO 79: PRINT "-"; NEXT : PRINT " "
2196 PRINT " "
2197 PRINT "DIRECT COSTS"
2198 PRINT "PLUS OVERHEAD"
2199 PRINT " !"
2200 PRINT " ! SALES VOLUME ---->"
2202 PRINT " V"
2205 PRINT " ";
2210 FOR X = 1 TO 7
2220 Z9 = S(X): GOSUB 15000
2230 Q9 = LEN (Z9%)
2240 PRINT TAB( 11 - Q9)Z9%;
2250 NEXT
2252 PRINT " "
2260 PRINT " ";: FOR Q = 1 TO 70: PRINT "-";: NEXT
2290 PRINT " "
2300 FOR Y = 1 TO 21
2310 Z9 = C(Y): GOSUB 15000
2320 Z9% = Z9% + " !"
2325 Q9 = LEN (Z9%)
2330 PRINT TAB( 10 - Q9)Z9%;
2340 FOR X = 1 TO 7
2350 Z9 = S(X) - C(Y): GOSUB 15000
2360 Q9 = LEN (Z9%)
2370 PRINT TAB( 11 - Q9)Z9%;
2380 NEXT X
2385 PRINT " "
2390 NEXT Y
2400 PRINT " ";: FOR Q = 1 TO 70: PRINT "-";: NEXT
2500 PRINT " "
3000 D$ = CHR$( 4)
3010 PRINT D$;"PR#0"
3015 HOME
3020 PRINT "END OF PROGRAM": END
11000 INVERSE : PRINT " >>>> BREAKEVEN <<<< "
11010 NORMAL
11020 RETURN
15000 Z9 = INT (Z9)
15010 Z9% = STR$( Z9)
15020 RETURN
```

**S(4).....starting sales volume**

**S(1) - S(3) decreasing sales volume**

**S(5) - S(7) increasing sales volume**

**D.....direct costs**

**O.....overhead expenses**

**C(11).....total cost of sales**

**C(1) - C(10) decreasing total costs**

**C(12) - C(21) increasing total costs**

**PS.....percent of increase/decrease for sales**

**PC.....percent of increase/decrease for costs**

**X.....counter for sales volume figures**

**Y.....counter for total cost figures**

**Q.....Input entry to continue the program**

**E\$.....the date of the printout**

**Z9%.....the amount as an integer**

**Q9.....the length of the printed amounts, so  
the chart will all line up properly**

**A\$.....used in GET statements to read through the  
instructions**

Table 1. List of variables used in Break-even program.

ures are shown for both sales and costs.)

We don't use commas in our printout, by the way, so we can print more amounts on the hard copy. And we don't try to put the same information on your screen; there are simply too many numbers to allow much comprehension.

The sales are in array S. This single-dimension array has only seven elements, so it's not dimensioned.

Costs are in array C, which is also a single-dimension array. However, it contains 21 items—your starting costs plus ten amounts higher and ten amounts lower. So, it is dimensioned in line 100.

### Line Specs

Lines 110-190 ask if you wish to see the instructions for the program, and let you skip them if you wish. Lines 200-700 are the actual instructions; they summarize what we've covered here.

Lines 1000-1295 are the entry lines for the program, where you enter your amounts and percentages. The program also sees that the Sales and the total of Cost plus Overhead are positive figures. Since the amounts will both increase and decrease according to the percentages you select, you need to start with positive amounts.

Lines 1300-1660 do all the math to get your sales and cost totals and put them in their proper array niches. Lines 1700-1760 change your values to integers, so the math will be correct.

Lines 2000-3020 do the printing of the chart for you, and let you exit so you can start over if you've made an error in entering. Lines 11000-11020 make up a subroutine that prints the break-even line across the screen.

Lines 15000-15020 convert the amounts to strings (Z9%) so we can measure their lengths when we print them.

### Out of the Red

Use this program to help determine your break-even point. Learn that if your sales drop X percent, your net profit will drop by Y amount. On the sample chart [Fig. 2], a five percent drop in sales causes a 100 percent decrease in profits.

You can use this program to determine exactly what figure you need to hold your total costs down to if your sales drop X percent.

Do you keep track of your sales and costs on a monthly basis? Then run a few simulations using various percen-

tages, grouped around your current monthly figures.

You'll soon see the relationships that exist between them. And, of course, you'll know your exact break-even points. You'll see how an X percent change in costs, coupled with a Y percent change in sales, will affect your bottom line.

You should do the same for your quarterly and yearly data. Run range printouts for any scenarios you can imagine. You'll have the exact bottom-dollar amounts on your printouts.

### Situation Simulator

In effect, you can make this printout show whatever you want it to; you can use it to simulate any business situation you can imagine. And best of all, since its figures go both up and down, you can really learn from the information.

Considering the present economy, it's more important than ever to know what our bottom line will be if business isn't quite as good as we expect, and what it will be if costs increase faster than we anticipate.

This program will let you simulate almost any comparison focusing on sales vs cost totals. ■

THIS REPORT WAS PRINTED ON 6/5/83.

TO READ THIS CHART, FIND THE SALES FIGURE YOU WANT ON THE TOP LINE, AND THEN LOOK DOWN THE LEFT SIDE, THE COST LINE, UNTIL YOU FIND THE COST YOU WANT. THE FIGURE WHERE THE LINES INTERSECT IS YOUR PROFIT OR LOSS BASED ON YOUR SALES AND COSTS.

THE INITIAL SALES VOLUME IS \$ 100000.  
THE RATE OF INCREASE/DECREASE IS 10 %.

THE TOTAL STARTING COSTS ARE \$ 95000.  
AND THE COSTS ARE INCREASING/DECREASING AT THE RATE OF 5 %.

DIRECT COSTS PLUS OVERHEAD ! ! ! ! V	SALES VOLUME ---->						
	72900	81000	90000	100000	110000	121000	133100
56880 !	16020	24120	33120	43120	53120	64120	76220
59873 !	13027	21127	30127	40127	50127	61127	73227
63024 !	9876	17976	26976	36976	46976	57976	70076
66342 !	6558	14658	23658	33658	43658	54658	66758
69833 !	3067	11167	20167	30167	40167	51167	63267
73509 !	-609	7491	16491	26491	36491	47491	59591
77378 !	-4478	3622	12622	22622	32622	43622	55722
81450 !	-8550	-450	8550	18550	28550	39550	51650
85737 !	-12837	-4737	4263	14263	24263	35263	47363
90250 !	-17350	-9250	-250	9750	19750	30750	42850
95000 !	-22100	-14000	-5000	5000	15000	26000	38100
99749 !	-26849	-18749	-9749	251	10251	21251	33351
104737 !	-31837	-23737	-14737	-4737	5263	16263	28363
109974 !	-37074	-28974	-19974	-9974	26	11026	23126
115473 !	-42573	-34473	-25473	-15473	-5473	5527	17627
121246 !	-48346	-40246	-31246	-21246	-11246	-246	11854
127309 !	-54409	-46309	-37309	-27309	-17309	-6309	5791
133674 !	-60774	-52674	-43674	-33674	-23674	-12674	-574
140358 !	-67458	-59358	-50358	-40358	-30358	-19358	-7258
147376 !	-74476	-66376	-57376	-47376	-37376	-26376	-14276
154744 !	-81844	-73744	-64744	-54744	-44744	-33744	-21644

Fig. 2. Printout of simulation of a break-even chart.



Hello thayuh. This is Eben Flow, proprietor of the Fish or Cut Bait Company, buyer and seller of lobstah bait for 49 years. My hobbies are collecting linoleum samples, squashing flies and playing pac-person on my home computer.

But here on Martinicus Rock, off the coast of Maine, the power can be a tad erratic. So, to cure the brownout and blackout problems, and to keep them spikes and surges off my picture tube, I got me a **MAYDAY** Uninterruptible Power Supply from SUN RESEARCH. Them fellas fixed me up real good and real light on my pocketbook, too. Got me a **MAYDAY** for my mini-calcaputer with a voltage regulator and everything for only 325 clams. They even included the battery in a nice waterproof box. Handy out here, you know. Now, if **MAYDAY** would only keep them sea dogs out of my barrel. . .

**MAYDAY** - Protection even you can afford!



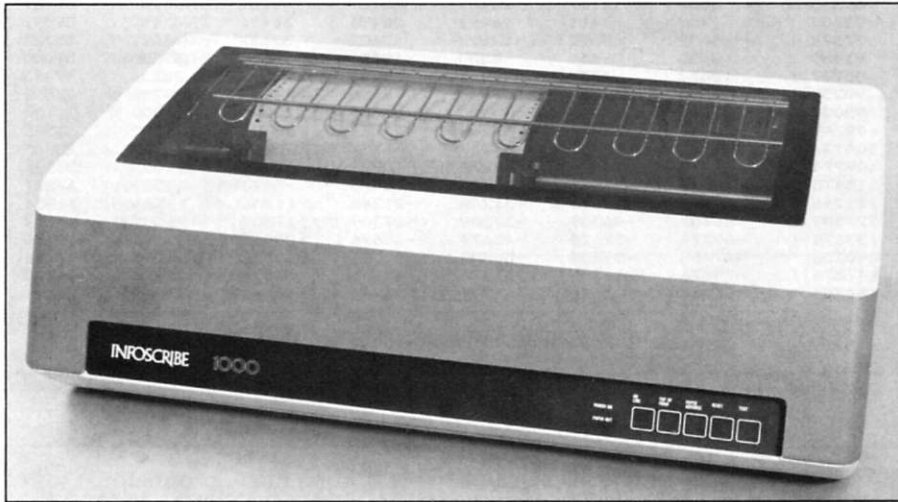
SUN RESEARCH, INC.  
Box 210  
New Durham, NH 03855  
603/859-7110  
TWX 5102974444

---

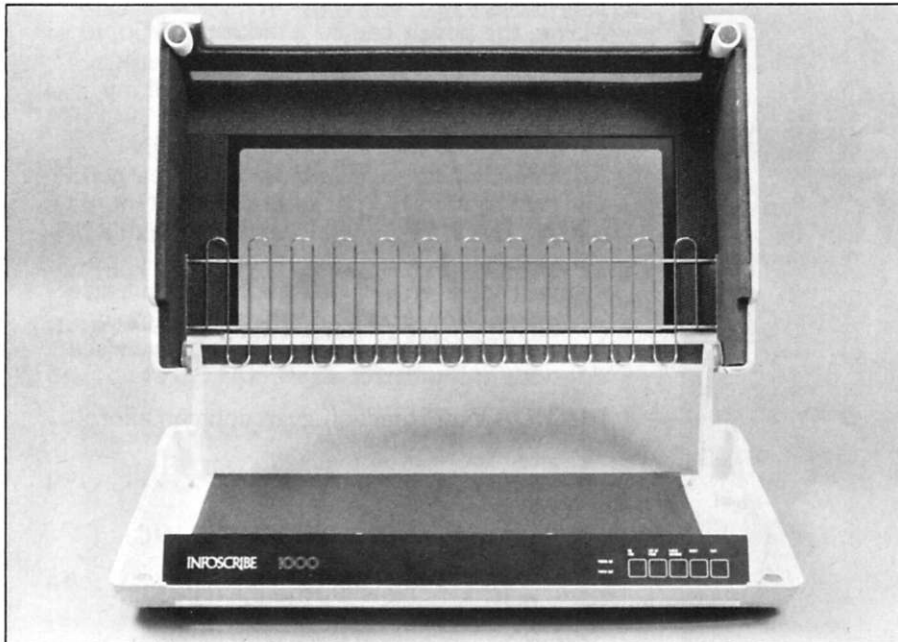
# Printer Survival Kit

*Shhh—the Infoscribe 1000 is printing. It's a heavy-duty, quality printer with the emphasis on quiet!*

By Jim Hansen



*The Infoscribe 1000 printer. This 200 character per second printer offers bidirectional, logic-seeking printing at 10, 12 and 16.5 characters per inch.*



*The printer case is of heavy construction, with plenty of sound-deadening foam rubber on top and bottom. The round dots at the top corners are magnets which mate with those on the bottom section of the case, used to hold the cover shut.*

In this month's printer survival kit, we'll look at the Infoscribe 1000 dot matrix printer. This heavy-duty, 200 character per second printer, equipped with a sound-deadening case, is intended for continuous service.

## Outside the Infoscribe

Externally, the Infoscribe 1000 is one of the more colorful printers on the market. The cream-colored injection-molded plastic case has a large, stylish red stripe across the front and down each side. The printer control panel is mounted on a partially recessed lip extending from the bottom section of the case.

The printer window is a 21×9¾-inch flat plate of plexiglass bonded to the top of the printer case. Visibility into the printing area is good.

The main ac line fuse, a classic toggle switch for power and a receptacle for the detachable line cord are located on the right side of the back panel. The other side of the back panel has a conventional Centronics-type interface connector and a dip switch used to set the baud rate and to stop bit selection for the serial interfaces.

The printer has two paper-feed paths—one underneath the unit (to be fed through a hole in the top of the printer stand) and one through a slot under the front lip of the case, for front-loaded paper. There is no provision for paper-feed from the rear of the printer. Paper is ejected through a slot near the top of the rear panel.

The paper path can handle form widths of 1.5 to 16 inches. Although I did not test this unit on multipart forms, the manual states that it will

---

*Address correspondence to Jim Hansen, PO Box 234, New Boston, NH 03070.*

---

print an original plus five copies. The forms-thickness adjustment knob is located on the left side of the printer mechanism, inside the case.

The Infoscribe 1000, manufactured by Infoscribe, Inc., 2720 S. Croddy Way, Santa Ana, CA 92704, offers a number of features that make it a versatile printer for line-printing applications on larger computer installations. These include character pitches of 10, 12 and 16.5 characters per inch (as well as double-width printing at half these densities), 80 or 132 columns per line, preprogrammed forms lengths of 11 and 12 inches, a programmable perforation skip, selectable auto-line-feed, printing at either six or eight lines per inch vertical spacing, two internally stored character fonts, downloadable fonts (those not stored in the printer, but sent from the host computer), a programmable VFU tape (this amounts to vertical tabs), super/subscript capability, and (with operator intervention and programming) underscore.

Infoscribe also offers bit-mapped graphics at 72 dots per inch, both vertically and horizontally. Although not tested, the implementation is conventional. Since both the horizontal and vertical resolutions are the same, no printer-induced distortion of graphics images will take place.

This printer can be ordered with several memory options to provide more print buffer (460 characters is standard, but up to 3K of additional memory may be installed). If downloaded fonts are used, some memory otherwise assigned to the print buffer is used to hold the fonts.

Three interfaces are standard with the Infoscribe. The standard Centronics-type parallel interface is most often used for local hookup. RS-232C serial and 20 mA current loop serial interfaces are also available. Both data terminal ready and X-On/X-Off signaling (to indicate when the printer is ready for more data) can be used with the RS-232C interface.

### Inside

The internal mechanics layout of the Infoscribe is similar to others built under the same license. (The Datasouth DS180 printer, reviewed in the March 1983 *Microcomputing*, is built under the same license as this one.) The sideplates and most of the rest of the mechanism are made of aluminum.

Paper motion is controlled by a stepper motor driving the tractors through

---

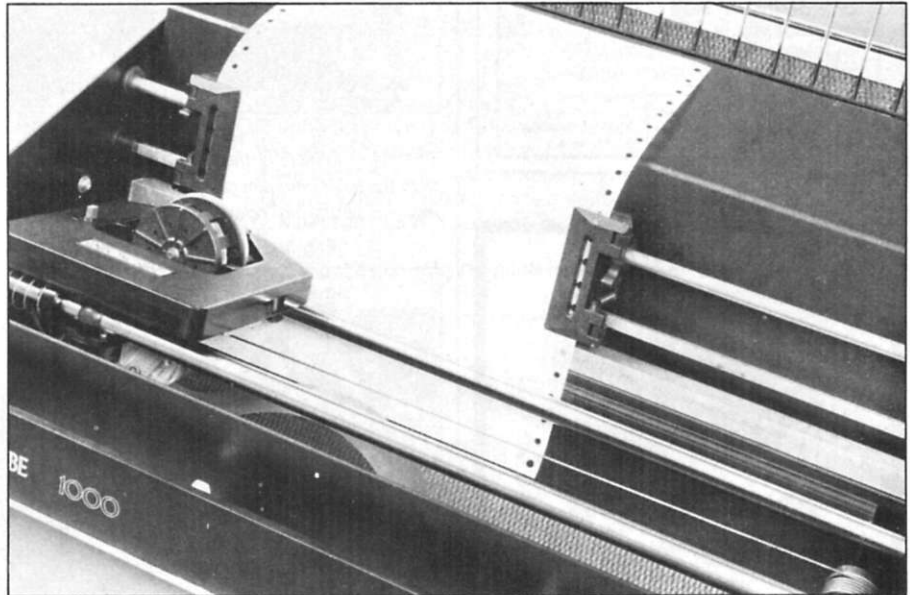
The Infoscribe 1000  
is a versatile printer  
for line-printing applications  
on larger computer installations.

---

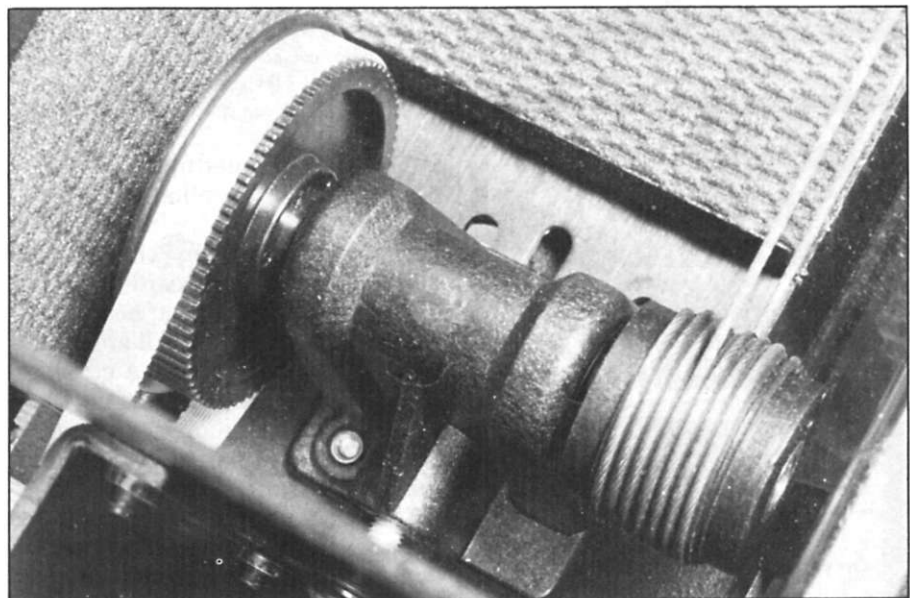
a pulley-timing belt combination. Head motion is controlled with a dc servo motor driving a timing belt, which in turn drives a cable stretched

across the printing bay. The cable is tensioned by repositioning a casting mounted to the printer frame. The ribbon is driven by a small dc motor mounted on the head carriage.

The electronics for the printer are contained in a well-shrouded enclosure in the back of the print area. The controller is Z-80-based, and was designed to use either 24- or 28-pin read-only memories, allowing for enormous expansion possibilities. Eight 2114 static random access memories provide 4K of memory. A CTC (a Z-80



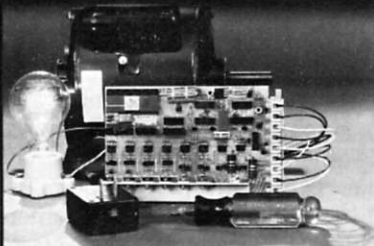
The printing bay of the Infoscribe 1000 is completely open. Any form up to 16 inches wide can be accommodated by the adjustable tractors. Printed text can be easily read as soon as it is on the paper. The cartridge-loaded ribbon and head is at left. Changing the ribbon is a one-handed job; it's easily snapped into place on top of the carriage. The ribbon motor, under the carriage, is a small dc motor. The head-to-platen adjustment is easily accomplished by turning a knob on the left sideplate.



The head cable tension, and thereby print quality, is maintained by sliding this casting left or right along the printer base. The timing belt shown around the large pulley passes through the right sideplate to a dc servo motor.



## Analog and Power Control I/O.....in a Single Board Computer

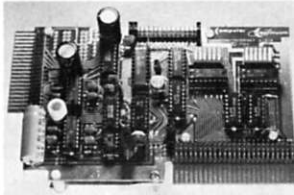


6801 or 68701 MPU with 2K ROM or EROM, 128 RAM, timer, 8 12-bit analog inputs, 8-bit analog output, 8 AC or DC inputs or outputs, serial I/O, digital I/O, watchdog timer, power supply.



Wintek Corp.  
1801 South Street  
Lafayette, IN 47904  
317-742-8428

## A/D 8 channels + D/A 8 channels



### APPLY YOUR MICRO TO ANALOG SIGNAL PROCESSING & CONTROL

Connects to  $\mu$  processor buss of most Z80 & 6502 computers including Timex, TRS-80, Apple, CBM 64. Our pin rearrangement feature eases interfacing.

**FAST:** 200,000 samples per sec with Z80A.

**EASY TO PROGRAM:** One PEEK chooses the channel, starts conversion & gets data. Routines provided.

**SOFTWARE AVAILABLE:** FFT in Z80, 256 points in one second; storage scope for Timex.

**LOW COST:** Assembled & tested \$195. Cable \$15. Please call or write for free catalog.

**Computer Continuum**

301 Sixteenth Avenue  
San Francisco, CA 94118 (415) 752-6294

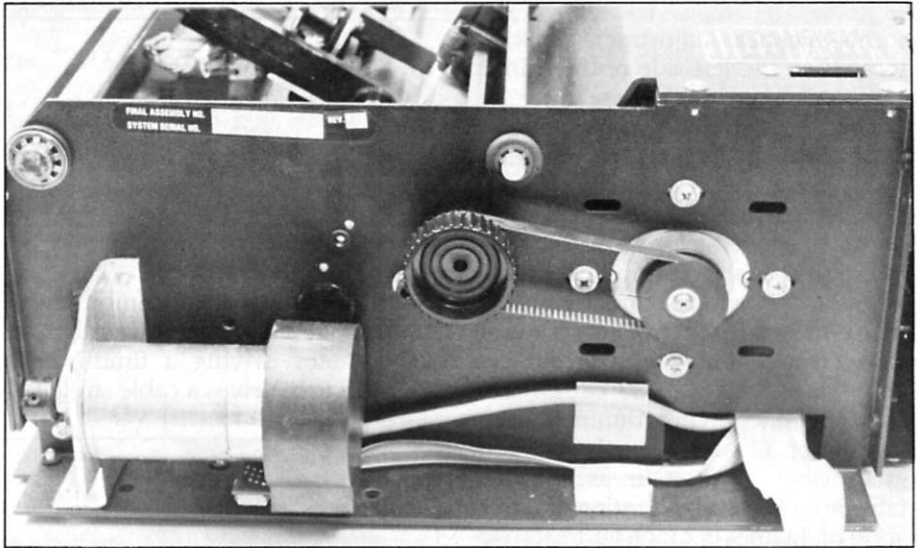
## this publication is available in microform



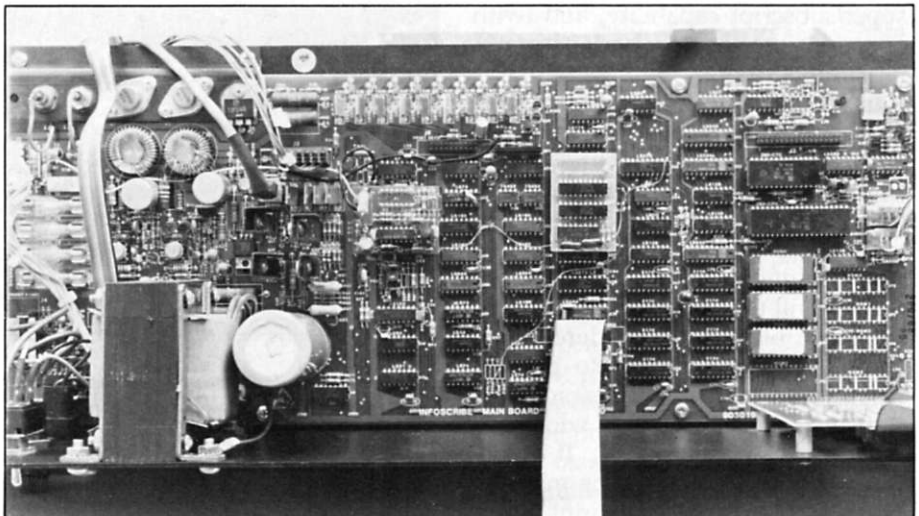
### University Microfilms International

300 North Zeeb Road  
Dept. P.R.  
Ann Arbor, MI 48106  
U.S.A.

18 Bedford Row  
Dept. P.R.  
London, WC1R 4EJ  
England



The right sideplate of the Infoscrite 1000, showing the head carriage motor (which is servo-driven, rather than the more common stepper motor drive system) and the paper-feed motor and timing belt. The line-feed motor is of unusual power, not requiring reduction pulleys to increase torque as in many printers.



The controller board and power supply. The printer is operated from a rather small transformer. Power supplies and load circuits are fused internally with the four fuses shown here at the left end of the circuit board. The controller is Z-80-based, and has provision for 4K of RAM and sockets for four ROM or EPROM chips. Two character sets can be resident in the printer at a given time.

family chip) counter-timer is used to time controller operations.

The power supply, with the exception of the transformer, is mounted on the main controller board with separate fuses for the 5-volt, head, servo and paper-feed power. All are clearly marked for easy service. A considerable amount of rework was noted on the controller, including two "piggy-back" boards, apparently added after the main board was designed.

The controller and power supply, as designed in this printer, are located in a nearly airtight box. No forced air or convection cooling was used to control the internal temperature.

This printer responds to a limited number of commands over the inter-

face. Only lines per inch, character pitch density, character set selection, graphics or alphanumerics and super/subscript vertical paper-positioning may be set by controls from your computer. (You also can load fonts and set the VFU tape from your host computer.) There are no horizontal tabs or fancy paper-handling commands available.

The manual is straightforward and completely covers the printer in technical terms. It is printed on 8½ × 11 paper and punched for standard ring binders. Hand-drawn illustrations are used throughout the document and serve their purpose well. I found no omissions or obvious errors, but would guess that experienced users

# Powerful CP/M<sup>®</sup> Software.

For Apple, Osborne, Xerox, Kaypro, North Star, SuperBrain, Heath/Zenith, and others.

Now only **\$29.95** each!

NEVADA

## COBOL

was \$199.95 now only **\$29.95.**

When we introduced Nevada COBOL in 1979, it was loaded with innovations. Today's Edition 2 is even better!

- Extremely Compact. You can compile and execute up to 2500 statements in 32K RAM, 4000 statements in 48K, etc.
- It's based upon the ANSI-74 standards with level 2 features such as compound conditionals and full CALL CANCEL.
- You can distribute your object programs royalty FREE!
- You get a diskette, 153-page manual with lots of examples and 16 complete COBOL source code programs.

Also available: COBOL Application Packages, Book 1 \$9.95

NEVADA

## PILOT

was \$149.95 now only **\$29.95.**

- Perfect for industrial training, office training, drill and testing, virtually all programmed instruction, word puzzle games, and data entry facilitated by prompts.
- John Starkweather, Ph.D., the inventor of the PILOT language, has added many new features to Nevada PILOT. There are commands to drive optional equipment such as Video Tape Recorders. There's a built-in full-screen text editor, and much more.
- Meets all PILOT-73 standards for full compatibility with older versions.
- You get a diskette, 114-page manual and ten useful sample programs.
- See Review in *Microcomputing*, January 1983, page 158.

NEVADA

## FORTRAN

was \$199.95 now only **\$29.95.**

- Based on ANSI-66 standards with some 1977 level features.
- IF . . THEN . . ELSE constructs.
- A very nice TRACE style debugging.
- 150 English language error messages.
- You get a diskette, 174 pages of Documentation and five sample programs. Requires 48K RAM.

NEVADA

## EDIT

was \$119.95 now only **\$29.95.**

- High quality text editing for micros!
- A character-oriented full-screen video display text editor designed specifically to create COBOL, BASIC and FORTRAN programs.
- Completely customizable tab stops, default file type, keyboard control key layout and CRT by menu selection.
- The diskette comes with an easy to read manual.

To make our software available to even more micro users, we've slashed our prices. What's more, we're offering a money back guarantee. If for any reason you're not completely satisfied, just return the package—in good condition with the sealed diskette unopened—within 30 days and we'll refund your money completely.

This is a limited time offer, so order yours today!

**Shipping/handling** fees. Add \$4.00 for first package and \$2.00 each additional package. OVERSEAS Add \$15.00 for first package and \$5.00 each additional package. Checks must be in U.S. funds and drawn on a U.S. bank!

Trademarks: CP/M, Digital Research; TRS-80, Tandy Corp.; TeleVideo, TeleVideo Systems, Inc.; Apple II, Apple Computer Inc.; Osborne 1, Osborne Computer Corp.; Xerox 820, Xerox Corp.; Kaypro, Non-Linear Sys.; Heath/Zenith, Heath Co.; IBM, International Business Machine, Corp. © 1983 Ellis Computing.

MAIL TODAY! To: Ellis Computing  
3917 Noriega St.  
San Francisco, CA 94122  
(415) 753-0186



**ELLIS COMPUTING**

The CP/M-80 Operating Systems and 32K RAM are required.

Indicate diskette format:

- 8"  SSSD (Standard IBM 3740 format)
- 5 1/4"  Apple CP/M  Osborne
- North Star DD  North Star SD
- TRS-80 Mod I (4200 hex)  TRS-80 Mod I/Mapper
- Heath, Hard Sector  Heath, Soft Sector
- Micropolis Mod II  Superbrain DD DOS 3.X (512 byte sectors)
- Xerox 820 (Kaypro)  TeleVideo

Indicate software packages:  COBOL  PILOT  
 FORTRAN  EDIT

Send my order for \_\_\_\_\_ packages @ \$29.95 each Total  
COBOL Applications Package @ \$9.95 each Total

- Check enclosed  In CA add sales tax
- MasterCard  VISA  Shipping/handling
- TOTAL

# \_\_\_\_\_ Exp. Date \_\_\_\_\_

Signature \_\_\_\_\_

Ship to: Name \_\_\_\_\_

Street \_\_\_\_\_

City/St/Zip \_\_\_\_\_

Country \_\_\_\_\_

# Enjoy the SEXPLOSION



**Subscribe Today**  
to The Adult Book and  
enjoy the latest guide  
to bedroom programs and  
games geared to creative  
and joyful living and loving.

## The Adult Book

Bourbon Street Press  
3225 Danny Pk., New Orleans LA 70002  
Telephone (504) 455-5330

Circle 132 on Reader Service card.

### '68' MICRO JOURNAL™

6800-6809-68000

★ The only ALL 68XX Computer Magazine

USA  
1 YR. — \$24.50 2 Yr. — \$42.50 3 Yr. — \$64.50  
\*Foreign Surface Add \$12 Yr. to USA Price  
Foreign Air Mail Add \$35 Yr. to USA Price  
\*Canada & Mexico Add \$5.50 Yr. to USA Price

OK, PLEASE ENTER MY  
SUBSCRIPTION

Bill my: M/C  — VISA

Card # \_\_\_\_\_  
Expiration Date \_\_\_\_\_  
For  1-Yr.  2 Yrs.  3 Yrs.  
Enclosed \$ \_\_\_\_\_  
Name \_\_\_\_\_  
Street \_\_\_\_\_  
City \_\_\_\_\_  
State \_\_\_\_\_ Zip \_\_\_\_\_

68 Micro Journal  
5900 Cassandra Smith Rd.  
Hixson, TN 37343

### PRESERVE MICROCOMPUTING WITH BINDERS & FILE CASES.



Keep your issues of *Microcomputing* handy and protected in handsome and durable library file boxes or binders. Both styles are bound in dark blue leatherette with the magazine logo stamped in gold.

File boxes: each file box holds 12 issues, with spines visible for easy reference.

\$5.95 each, 3 for \$17.00, 6 for \$30.00

Binders: each binder holds 12 issues and opens flat for easy reading.

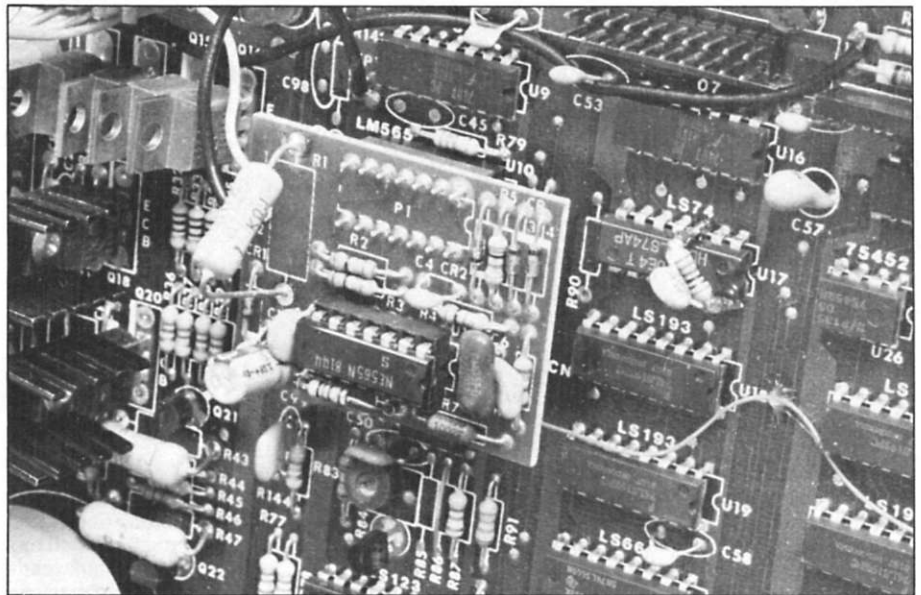
\$7.50 each, 3 for \$21.75, 6 for \$42.00

(USA postage paid. Foreign orders must include \$2.50 per item.)

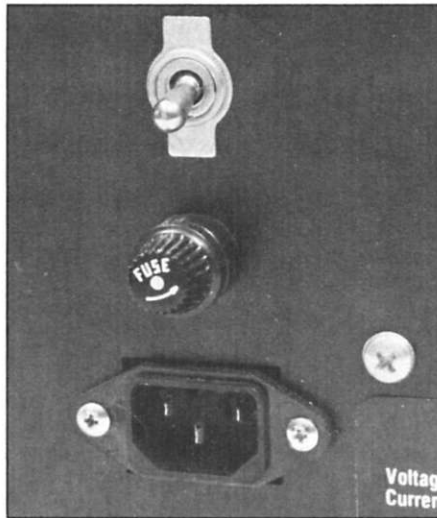
Please state years desired (1977 to 1984).

Send check or money order to:

Jesse Jones Box Corp., P.O. Box 5120, Philadelphia, PA 19141; please allow 6 to 8 weeks for delivery. Sorry, no C.O.D. or phone orders.



Probably one of the biggest pains in the neck for an engineer is having to add or to modify a controller board after some purchasing agent went on a rampage and bought 10,000 boards before the need for the change was found. The Infoscrite 1000 evidently had this problem. Shown here is one of two piggy-back boards added. Notice the resistor/capacitor combination added across IC U17. This sort of problem happens to everyone (even the Japanese) and is no reflection on the manufacturer or design staff. Such changes usually do not degrade the reliability of the product; they just look bad and embarrass the engineer responsible.



The ac power switch, primary fuse and power cord connector. I included this photo in appreciation of the engineer at Infoscrite who stood his ground and put in a "real" switch—a toggle right out of the 1940s. I have always hated rocker switches, and at last I know I'm not alone. I wonder who my friend at Infoscrite is.

will find this manual easier to read than newcomers to the field.

### Subjective Commentary

The Infoscrite's large, open printing bay that allows easy access for changing forms—as well as clear visibility of what is being printed—is an appealing feature. So is the heavy case, which was designed to keep printer noise under control.

I judged the Infoscrite very quiet

for a tabletop printer of any speed, but I didn't measure it personally. The manual claims about 54 dB, compared to around 72 dB for a Selectric typewriter. The Infoscrite case's seals and acoustic foam have paid off, and I have no reason to doubt the noise-level figure given in the manual.

I missed not having a rear paper-feed path. Most tabletop printers (or most printers in general, for that matter) allow paper to be fed from the back of the unit; this one doesn't.

Also, no provision was made for a tear bar. Users must be careful when removing copy from the printer, or paper may be pulled out of the tractor and have to be reloaded.

The interface connector at the back of the printer is of high quality, but was not equipped with retainers to hold the interface cable in place. I had to reconnect the interface cable several times during my evaluation because it had vibrated loose during operation.

My biggest concern is what happens to the internal controller temperature on a hot day in the computer room. Being sealed as tightly as it is must make it hotter than necessary, which will ultimately cause semiconductor failure. (Semiconductor lifetime in terms of thousands of hours is keyed directly to operating temperature.) Evidently, Infoscrite was satisfied that the black structure around the controller radiated enough heat for

Line Length (characters)	Time to Complete 20 lines (seconds)	Throughput (characters per second)
20	4.5	88.8
40	7.0	106.6
80	11.5	139.1
7633 chars. of text*	75.0	101.7

\*A preliminary version of this review was printed using an Epson QX-10 computer. The line length was set to 75 characters, and the text double-spaced. This is a typical type of printing job and gives a realistic idea of the actual real-life throughput of the printer.

Table 1. Throughput measurements for the Infoscrite 1000 printer. Except as noted, these tests were run from an Apple computer using a parallel interface card. The printer was set to ten cpi print density, six lines per inch.

sufficient cooling to take place.

The printer supplied for evaluation did not include a UL sticker. Whether this means that the printer has not been UL-tested, or that someone in production left it off is not clear. My personal inspection satisfied me that this is a quality printer; I would find it hard to believe that it doesn't meet or exceed UL standards.

While its print quality is not out of the ordinary, the Infoscrite is excellent in terms of overall execution. It's clearly put together by a manufacturing and engineering team that cares. ■

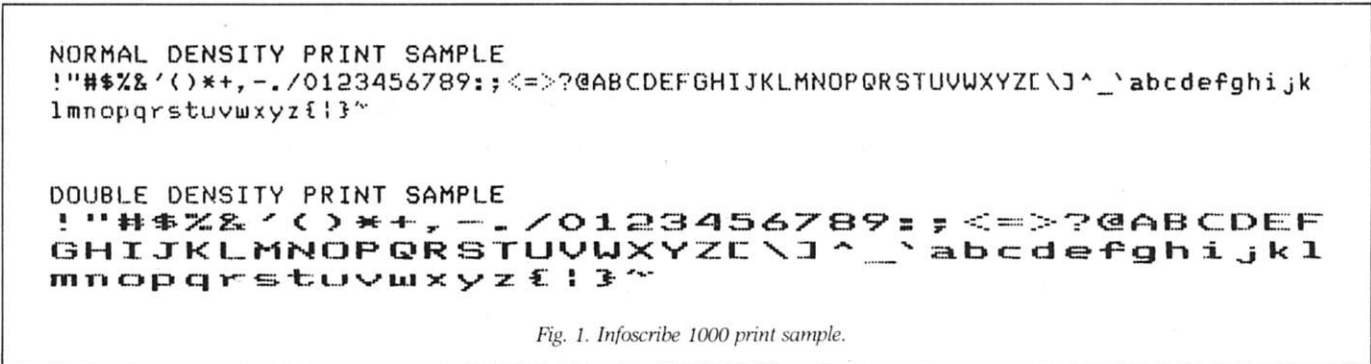


Fig. 1. Infoscrite 1000 print sample.

Circle 92 on Reader Service card.

## 68 <sup>XX</sup> Products \$ALE

JPC is closing out some of its SS-50/30 Product Line . . . and having a Sale on the rest! Close out when stock is gone, Sale ends Sept. 30, 1983.

### CLOSE OUT

MX-6 SS-50 Extender . . . . .	\$15.95
CK-7 Real Time Clock . . . . .	\$45.95
DAC-5 Dual Channel A/D . . . . .	\$59.95
PA-15 Parallel Interface . . . . .	Sold Out
TS-11 Motor Control . . . . .	Sold Out

### SALE (\*)

TC-3 High Speed Cassette Interface . . . . .	\$49.95
AD-16 16 Channel A/D . . . . .	\$69.95
CFM/3 Cassette File Manager on Cassette	\$19.95
CFM/3 Cassette File Manager on EPROM	\$24.95
BASIC/3 High Speed Cassette Basic . . . . .	\$39.95

(\*)Specify 6800 or 6809

Terms: Cash, Master Card or Visa  
Shipping & Handling \$3.50 (US)  
\$5.50 (Canada) \$15.00 (Foreign)

  
JPC PRODUCTS CO.  
Phone (505) 294-4623  
12021 Paisano Ct.  
Albuquerque, N.M. 87112

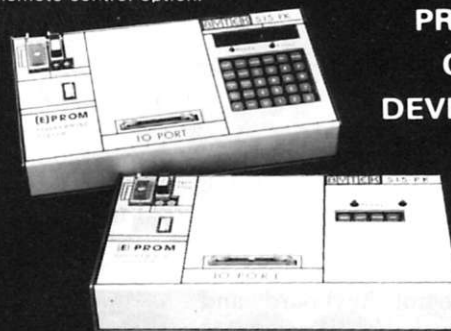
Circle 396 on Reader Service card.

## BYTEK's SECOND GENERATION UNIVERSAL (E)PROM PROGRAMMER—SYSTEM 15

### Features:

- Bipolar PROMS.
- Micros - (8748 & 8749)
- I/O - 6 baud rates, 13 formats including Intellec, Textronix and Motorola.
- EPROMs, (2708-27256)
- Gang option - programs eight at once.
- Remote control option.

\$445  
\$15-R



PROGRAMS  
OVER 250  
DEVICE TYPES

### FUNCTIONS:

DISPLAY DEVICE DATA  
EDIT RAM DATA  
DEVICE PROGRAM  
TYPE SELECTION

CRC-RAM  
LOAD DATA  
COMPARE FIELDS  
FILL MEMORY FIELD  
BLOCK MOVE  
DIAGNOSTICS  
and more.



**BYTEK**

COMPUTER SYSTEMS CORP

2283 S. Federal Hwy.

Delray Beach, FL 33444

(305) 272-2052

# A VIC Printer For the Do-It-Yourselfer

*Dust off your ASR33 Teletype and put it to work with your VIC-20.  
All you need is this simple interface circuit.*

By George R. Steber

It might seem strange that a peripheral device for a computer actually can cost more than the computer itself. Yet that's often the case with the VIC-20, which can be bought for less than \$100.

A printer is a useful peripheral, but it's not absolutely necessary. It would be nice to have one for producing listings of Basic programs and other relatively simple data lists. In most other cases, the video display handles the input/output situation adequately.

Is there a cheap printer that can serve limited purposes?

## A Low-Cost Printer

For years the workhorse of the minicomputer industry was the venerable ASR33 Teletype.

The ASR33 teletypewriter is a serial device, with integral keyboard and printer, that uses the ASCII code. It is designed to operate at 110 baud and may be configured for RS-232 voltage levels or 20 mA current loop.

The best part of the ASR33 is that it can be found as a surplus item for a price of around \$75 at hamfests or computerfests. That's enough to warm the heart of old Scrooge himself!

The next question, of course, is:

How does the ASR33 work with the VIC-20? With a simple interface (it really is simple) and knowledge of some of the inner workings of the VIC, you can find out.

## Printer Requirements

The first task is to locate an ASR33 for the job. The ASR33 teletypewriter can be found in various types of configurations; unfortunately, this may cause some confusion for the neophyte.

The basic requirement for the ASR33 in this application can be stated simply. It *must* be configured for 20 mA operation.

A large number of ASR33s retired from service are already set up for the 20 mA operation. In these cases, no

modification will be necessary. However, you may encounter one that is set up for RS-232 or something else. Your best bet in that case is to have someone familiar with internal wiring make the necessary change to 20 mA.

The ASR33 should be set up for 20 mA operation with two wires (which form the 20 mA loop) brought out of the machine for interfacing purposes.

Note that when the ends of the two wires are touched together, the machine will revert from the space mode to the mark mode. In other words, it stops "clunking" and starts "humming." This is a good preliminary check to see if your machine is working okay. The next step is connection to the VIC interface.

## VIC Interface

The interface circuit from the VIC-20 to the ASR33 is shown in Fig. 1. Power is tapped directly from the VIC; little current is needed. Only a few parts are required for the circuit. They should be readily available at your local parts store; total cost should be a dollar or so. All resistors are

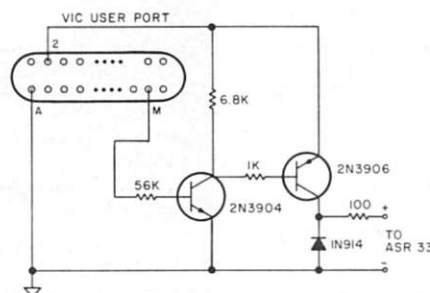


Fig. 1. VIC-to-ASR33 interface circuit.

Address correspondence to George R. Steber, 9957 N. River Road, Mequon, WI 53092.

¼-watt, ten-percent carbon types. The transistors and diode are common general-purpose items.

The interface is connected between the VIC user port and the ASR33 as shown. It will be powered up automatically when the VIC is turned on. I suggest that a 24-pin edge connector be connected to the user port. Do not solder directly to the VIC board. (This is a cheap design, but we have to draw the line somewhere!)

Mount the circuitry of the interface on a small PC board and install it in a convenient place. Use care in laying out the parts and check your work closely. If everything looks okay, plug it into the user port and the ASR33, and proceed.

When power is applied, the ASR33 should begin humming. If not, reverse the leads to the ASR33. If you cannot get the ASR33 to hum, your circuit

probably needs work.

#### Printer Software

One fact you may not know is that the VIC has serial I/O routines already implemented in ROM. It's just a matter of calling these routines to output data to the ASR33. The best way to accomplish this is to open a channel (via RS-232 software in ROM).

A simple program to generate a listing of a Basic program stored in your VIC is shown in Fig. 2. This program may be entered any time, provided one caution is observed. The RS-232 software in ROM sets up buffers in the top 512 bytes of RAM. If you have a Basic program in RAM, it may not work properly if it uses this part of RAM.

There are ways around this problem. One solution is to reserve 512 bytes of RAM, hidden from your Basic

program, for the RS-232 buffers. (The avid computer buff will, of course, find other solutions.) I saved 512 bytes of RAM by following the instructions at the start of the Basic program—POKE 52, 28 : POKE 56, 28.

When you've finished your listing or other printing task, be sure to print and close the channel. The necessary instructions to do this are shown in Fig. 2. This will divert all output back to the video screen.

#### Voila!

So there you have it—an inexpensive VIC-20 printer. I've had mine in operation for several months, and I've found it to be helpful in listing and debugging Basic programs.

An additional benefit is that the ASR33 prints a full 80 characters per line and makes beautiful listings. (It should be noted that only uppercase letters are printed, and special symbols do not print because they lack valid ASCII codes.)

The ASR33 is not the most elegant printer available, but it's inexpensive and fairly reliable. Used sparingly, it should last quite a while in this application. ■

```
To open the channel and list program on ASR33, type:
OPEN 128,2,3,CHR$(163)+CHR$(160):CMD128:LIST
To close the channel and return to video screen:
PRINT#128:CLOSE128
```

Fig. 2. VIC Basic commands to print on ASR33.

Circle 189 on Reader Service card.

# TRS-80\*

100% Radio Shack Equipment

## SAVE A BUNDLE

Order Toll Free 1-800-874-1551

FLA Residents 904-438-6507 collect

EPSON, OKIDATA, CITOH, TABCO Printer Switches



## SALES CO.

704 W Michigan Ave; P.O. Box 8098  
Pensacola, FLA 32505

\*TRS-80 is a trademark of Tandy Corporation.

# Do-It-Yourself Recorder For the VIC-20

*Build this simple cassette interface circuit that lets you use an ordinary recorder with the VIC—and save yourself \$70.*

By Jim Brousseau

I bought my VIC-20 computer, frankly, because it was inexpensive. Consequently, I thought \$70 was a bit too much to pay for a cassette recorder to store programs on. So to operate the recorder I already own with the VIC-20, I designed the circuit shown in Fig. 1.

To write a program onto cassette, the VIC-20 outputs a digital signal to pin E-5 of the cassette port. The circuit

shown converts the digital signal to an analog-signal-level suitable for the microphone input of a cassette recorder. The CD4049 inverter and the LED provide a visual indication that the VIC-20 is outputting data to the cassette.

To read programs from cassette, the VIC-20 requires an active low-digital pulse on pin D-4 of the cassette port. The earphone output of the cassette re-

order is converted back into a digital signal by the LM3900. Because the VIC-20 requires an active low pulse, inverter CD4049 is required.

The 5-volt relay and the 18-ohm resistor control the remote switch of the cassette recorder. When Load or Save is commanded, the VIC-20 turns on the 6.7 V cassette motor drive on pin C-3 of the cassette port. When this happens, the relay switches on, starting the cassette recorder.

Since I don't have an extra playback switch on my recorder, I chose to disable the cassette switch option by tying pin F-6 of the cassette port to ground. If you use this option with your VIC-20, you can tie pin F-6 to a switch.

This circuit is relatively immune to volume-level setting. The LED tied to the tape input to the circuit can be used to give an indication of input level.

I've used this circuit on several other computers and have found it to be effective. And it could save you \$70. ■

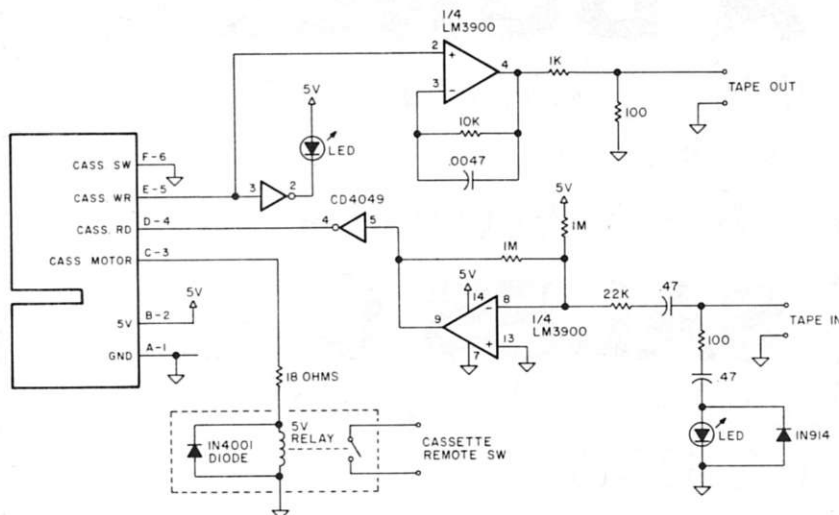


Fig. 1. VIC-20 cassette interface. Note: CD4049 pin 1 is 5 V, and pin 8 is ground.

Address correspondence to Jim Brousseau, 6900 Firebird Dr., Orlando, FL 32810.

# Son of "The Inflation Fighter" The **Orange+Two** \$1095 Suggested List Price



New Math from Orange+ Computer Systems

Z80A CPU + 6502 CPU = "**Orange+Two**"

We give you two computers for the price of one, because we've put them both in the same machine. The "Orange+Two" is the latest "state of the art" personal business/home computer with two microprocessors, each operating independently. Consider: CP/M 3.0, Digital Research's new enhanced CP/M version, AND 6502. That means the "Orange+Two" will run over 20,000 currently available programs, including some of the most popular business and game software on the market today. And, it uses the new easy-to-learn, easy-to-program language: FORTH-79.



#### STANDARD FEATURES:

- Z80A plus 6502 microprocessing units
- + 64K RAM Bank Switchable, fully expandable
- + Built-in disk drive controller for two drives
- + Programmable ASCII keyboard with numeric keypad and auto repeat
- + Built-in control-reset function
- + Audio volume control switch
- + High resolution video display with graphics
- + Cassette interface for tape backup unit
- + Game input and output connector for joystick
- + Operating languages: FORTH-79, BASIC, CP/M 3.0
- + Strong, solid, high-impact case with removable lid, allowing full access to interior
- + Metal base plate with motherboard connectors for fast board removal and replacement (no screws)
- + 110/220 switch selectable power supply. Operating range 47-65 Hz
- + Programmable EPROMS (2764). Allows user to do custom ROM programming

The "Orange+Two" is the pacesetter of the future. Why buy 1970's technology at 1983 prices, when you can purchase two computers for the cost of one. Ask your favorite dealer for the "Orange+Two".

The "Orange+Two" carries a full 90 day warranty. Nine-month extended warranty is available for an additional \$99.00.

Prices & Specifications are subject to change without notice. Limited quantities are subject to availability.

If "Orange+Two" is not available locally, you can order factory direct. To order factory direct, call **(213) 999-5210** or send bank check or money order for \$1095\* to Collins International Trading Corporation, 23801 Calabasas Road, Suite 2050, Calabasas, California 91302

Allow 30 days for delivery from receipt of order.  
\*California residents add 6½% sales tax.

DEALER INQUIRIES INVITED.

Circle 159 on Reader Service card.



# The Intelligent Toaster

*We continue our journey into the world of single-chip intelligence by exploring the programming and use of the Intel 8748 en route to mastering computer-controlled devices.*

By Mark J. Robillard

You will find, in the pursuit of intelligent control, that the "computer-on-a-chip" will provide a cost-effective way of implementing a design. It offers an experimenter the flexibility of designing a complex control system that uses a small number of parts and isn't so tedious to construct.

In the March and May issues of *Microcomputing*, we began our venture into the world of single-chip intelligence with an in-depth study of the inner workings of one of these products: the Intel 8748. We covered the basic pin functions and dabbled a bit with instructions that had to do with accessing external memory. With this under our belt, let's move on.

## Accumulator Instructions

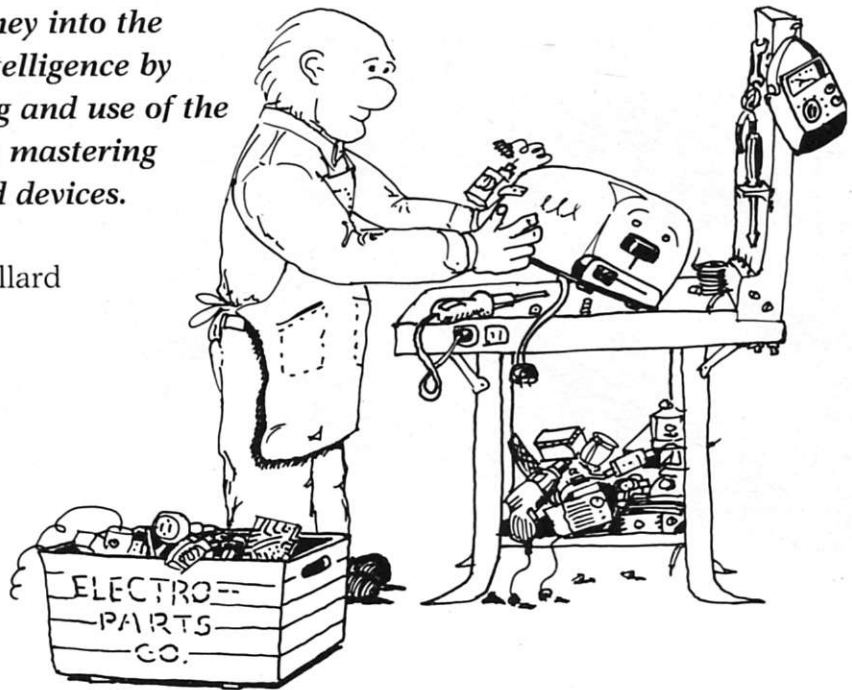
Remember I said that all data moves go through the accumulator? Well, let's see just what operation can be done with this register. The following is a list of accumulator instructions:

INC A	CPL A
DEC A	SWAP A
CLR A	DA A

These are the accumulator instructions that utilize only the accumulator. (Register and logical operations involving the accumulator will be covered separately.)

Basically, the list includes some of the more simple instructions in the 8748 repertoire. INC A will add +1 to the value already there. If the previous value was FF (all 1s), an INC will reset the accumulator to zero. Upon doing this, however, the carry flag (not yet described) will assume a value of 1.

DEC will do the opposite. Every



time it is involved, the value in the accumulator will decrease by one. CLR will reset the accumulator to all zeros. CPL will complement each bit.

If the value FF was stored into A before CPL was exercised, the value 00 would result. This instruction comes in handy if you have to invert data that is coming from a port, or if you have to output data to a port that is connected through open collector drivers. The CPL instruction would assure the correct signal polarity output.

SWAP is an interesting instruction. Have you ever started working with the lower four bits of a register, and then wanted to move them up to the high-order four bits? This type of operation is done frequently when building an address from hex keypad entry.

The SWAP instruction will perform an actual transfer of high-order nibble (D7, D6, D5, D4) for low order. The result you'll see in the accumulator looks like this:

Before	D7	D6	D5	D4
	1	0	1	1
	D3	D2	D1	D0
	1	1	1	0
After	D7	D6	D5	D4
	1	1	1	0
	D3	D2	D1	D0
	1	0	1	1

## Adjusting the BCD Value

Working with four-bit values brings to mind a lot of BCD applications.

Doing arithmetic in BCD can be cumbersome without some help from the microprocessor. The DA instruction provides this help with its ability to readjust the BCD value in the accumulator after performing addition between two BCD numbers.

The instruction (DA) will adjust the result of the addition (now left in the accumulator) into two BCD digits. This is done in the following sequence:

1. Check bits 0 through 3 for a value greater than 9. If the value is not that high, go to step 2. If the contents is greater than 9 or if the auxiliary carry bit (to be covered later) is 1, then increment the accumulator by 6.

2. Check bits 4 through 7 for a value greater than nine. If the value is not that high, then proceed to the next instruction. If the value is greater than 9 or if the carry bit is a 1, then the values in the four upper bits are incremented by 6. If an overflow occurs, the carry bit is set.

Aren't you glad you don't have to do

---

Address correspondence to Mark J. Robillard, c/o MJR Digital, Inc., PO Box 630, Townsend, MA 01469.

---

all that in separate programming statements? The DA single-byte instruction does it all. In fact, all the accumulator instructions we just covered are one-byte-only types.

### Program Status Word

During our discussions, the carry bit and auxiliary carry flag have come up. I guess it's time to cover the processor's way of checking status when doing arithmetic or logical operations.

The Program Status Word (PSW) should be a relatively familiar entity to those of you with backgrounds in other microprocessors. It is here that the Carry Flag and Auxiliary Carry reside.

Let's look at the structure of the PSW location in the 8748. (Fig. 1 describes the PSW.)

As you can see in Fig. 1, the first three bits act as a sort of pointer to the stack memory area. I mentioned that there are eight levels of subroutine nestings that may be handled by the stack; these locations reside just above the first register back in Data Memory. These three bits indicate which two-byte location is next in line for storing a return address.

To see exactly how this is performed, let's digress a moment into the two areas that use the stack: Call

instructions and Interrupts.

### Interruption of Instruction Sequence

Commonly used routines or groups of instructions may be written once and used many times through the use of the Call instruction. In larger microprocessors this would equate to a Jump to subroutine.

There are eight versions of the Call instruction:

Call (page 0)	Call (page 4)
Call (page 1)	Call (page 5)
Call (page 2)	Call (page 6)
Call (page 3)	Call (page 7)

Each Call instruction has a unique hex op-code, which allows you to jump to a subroutine located anywhere in program memory as long as you specify the page where the routine

begins.

Remember that the 8748 works with 256-byte chunks of memory at a time. The first 1024 locations (pages 0-3) are located on-chip. The other thousand may be accessed as external program memory.

If you select memory bank 1 (SEL MB1 instruction) you get another 2K of external program space. It should be cautioned, however, that the Call instruction and any other page-oriented instruction will function in that relative page regardless of which bank is selected. Page 0 of bank 1 includes locations 2048-2303 of external program memory.

The Call instruction consists of two bytes. The first, of course, is the op-code. One of the eight page-related

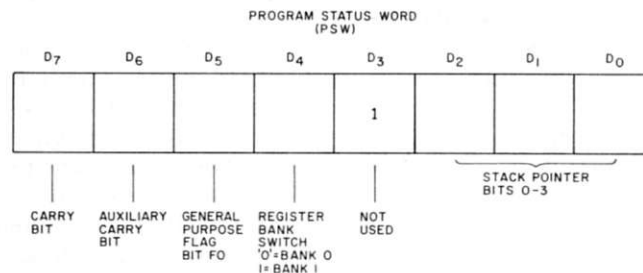


Fig. 1. Bit assignments of the Program Status Word.

# WHAT REALLY IS INSIDE YOUR COMPUTER?

Find out in **INSIDE YOUR COMPUTER** from Wayne Green Books. I.R. Sinclair takes the cover off your computer and shows you what's inside and what it does. Novices will find information on:

- Microprocessors
- Interpreters
- Registers
- Input/output
- Machine language
- Logic operations

A look at programming ties it all together—how hardware and software make a microcomputer work. The information applies to any microcomputer system. A glossary of computer terms and an appendix on binary, decimal, and hexadecimal conversion make the book all the more valuable.

\$12.97, softcover, 109 pp., 5½ × 8½.

ISBN #0-88006-058-1

Call **TOLL FREE 1-800-258-5473** for credit card orders. Or mail your order with payment or complete credit card information. Include \$1.50 for shipping and handling.

Photocopy of coupon is acceptable for ordering.

Send to:  
Wayne Green Inc.  
Attn: Book Sales  
Peterborough, NH 03458  
Dealer Inquiries Invited



## Yes, I want to know what's inside my computer!

Send me \_\_\_ copies of **INSIDE YOUR COMPUTER**. (BK7390) Enclosed is \$12.97 per copy plus \$1.50 shipping and handling.

MASTERCARD bank # \_\_\_\_\_  VISA  AMEX

Card # \_\_\_\_\_ Expires \_\_\_\_\_

Signature \_\_\_\_\_

Name \_\_\_\_\_

Address \_\_\_\_\_

City \_\_\_\_\_ State and Zip \_\_\_\_\_

Send To: WAYNE GREEN BOOKS Attn: Book Sales Peterborough, N.H. 03458  
UPS Delivery if complete street address is given. **337B7Y**

codes is used here.

The second byte is the location of the subroutine (address) within that page. When the processor encounters a Call, the first thing it does is store the present value of the program counter and the upper four bits (carry, aux carry, F0 and bank) on to the stack.

Because it is possible to address up to 4K (internal and external banks 0 and 1) of program memory, there is a need for 12 address bits. (The actual placement of this information on the stack is outlined in Fig. 2.)

After the stack is loaded, the low-order eight bits of the program counter are reloaded from the second byte of the Call instruction. The upper bits (A8, A9, A10) are adjusted according to the bank of the Call, and execution is resumed at that new location. Normal instruction processing continues until a return from subroutine (RET) instruction is encountered.

Either of the two return instructions, RET and RETR, will enable you to retreat to the original calling point, but they don't return you exactly the same way. The RET instruction will decrement the stack pointer (PSW bits 0-3) to point to where you stored the directions for home.

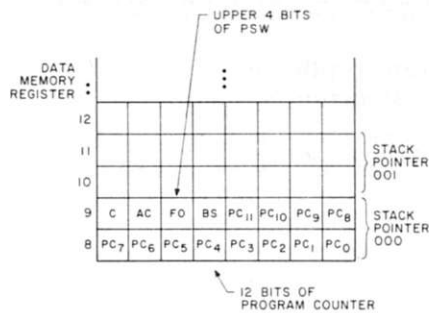


Fig. 2. Details of internal stack operation.

The program counter will then do its previous set of clothes (a-dress?) and resume processing at home. (Notice it left behind the upper four bits of the PSW.) The RET instruction may be used in cases where the status of the PSW bits either are not used or are unimportant to that particular point in the program.

The next Call instruction will clobber those bits that are left, so be sure you don't need to know what their state was when you left.

For those of you who tend to be extra cautious, there is a return that will restore the PSW to its initial state. The RETR instruction resumes processing at home and restores the PSW. Be a lit-

tle brave and use RET occasionally; the Surgeon General has not determined it to be unhealthy! (On that note, why don't you take a break here? Get up, walk around... this stuff can be pretty dry. I know; I've fallen asleep three times already!)

Well, so far we've seen how the accumulator may be accessed directly. We've investigated the stack and the use of subroutines.

There is another case where 8748 program execution may be interrupted. The INT input pin will cause an immediate interrupt. This is a hardware function that is involved whenever a low level is detected on the INT pin. When it occurs, both the program counter and the upper four PSW bits are saved on the stack.

Hardware inside the processor then loads the address value 003 into the program counter and execution is picked up there. Normally, you would put a jump instruction (JMP) in location 003, which would catapult you into a general-purpose interrupt routine. (We'll cover jumps a little later on.)

After executing the interrupt service routine, you can reenter your program right where you left off by performing the RETR instruction. So keep in mind

Circle 107 on Reader Service card.

## FREE ESTIMATES COST MONEY

If you give estimates in order to secure business for your company, you know they take time and cost you MONEY!

**GROUT & ASSOCIATES**

**EXBIDITE**

**COST ESTIMATES  
CONTRACT BIDDING**

**TO CUT THE COST OF ESTIMATING AND PROMOTE YOUR BUSINESS, YOU NEED EXBIDITE**

Compiling cost estimates or contract bids becomes easy with **EXBIDITE**. The **EXBIDITE SOFTWARE PACKAGE** enables salesmen to quickly and efficiently create itemized estimates based upon inventory items and services requested by a customer. The estimate lists items or services, quantities, price, totals, as well as information identifying the company, customer, annotations, terms and type of project. Once created, the estimate can be printed for the customer and stored on disk for easy reference or updating later, alleviating the need for long recalculations should the details of the client's request change.

**EXBIDITE** is ideal for businesses like the retail lumber industry which routinely create bids from long lists of materials requested by customers. Prices can be easily updated, and the margin of profit for each estimate can be adjusted according to individual items or across the entire estimate. **EXBIDITE** can handle an unlimited number of inventory items. Included is a program which creates inventory tables, simplifying the initial creation of an inventory data base.

**EXBIDITE** is a basic software package available for the TRS 80 MODEL I and III. One disk drive, a printer, and 48k of memory are required. Documentation is clearly written and includes an easy to follow tutorial enabling the user to quickly learn the programs.

AND THE IBM PC!

PRICE: \$39.95

**EXBIDITE** is available from

**GROUT & ASSOCIATES**

26324 Edgewater Blvd. N.W.

Poulsbo, Washington 98370

415-472-7183

- Professional Appearance
- Minimize Errors
- Save Time
- Pinpoint Profits
- Maximize Business

Circle 345 on Reader Service card.

# LYNC

## COMMUNICATIONS SOFTWARE

**WHY BUY 3 PROGRAMS?  
LYNC HAS ALL 3 IN ONE!**

1) FILE TRANSFER
2) REMOTE OPERATION
3) COMMUNICATE WITH TIME-SHARE

Easy set up on most systems.  
Menu driven install program.  
Command Mode of operation  
Help Screen for Lync & Term

Z80 based CP/M \$155

PC DOS \$155 TRSDOS \$75

**We're Not the Biggest, Just the Best.**

**International Software Alliance (805) 966-3077**

**1835 Mission Ridge Santa Barbara, CA 93103**

\* DEALER AND OEM INQUIRIES INVITED \*

FORMATS: 8" Standard; IBM PC DOS; TRS-80 Mod II with CP/M; VectorGraphic; Cromemco; North Star; Osborne; Xerox 820 (8" or 5"); SuperBrain; NLS KayPro; TeleVideo; TRS-80 Model I or III; Zenith Heathkit; Victor 9000; Sanyo; Altos; Fujitsu; Otrona.

CP/M is a trademark of DIGITAL RESEARCH. LYNC is a trademark of MIDNIGHT SOFTWARE.

when you're programming to carve out some space around location 003.

In fact, take a look right now at Fig. 3; it shows the layout of the program memory. Indicated in the figure are the reset and interrupt locations. The timer interrupt shown there will be covered later.

There may be times when you might not want to be interrupted. In fact, it's a good idea to avoid interruption while you're processing an interrupt request. The 8748 provides the ability to disable or enable the operation of the INT line through the use of two instructions: EN I and DIS I.

### PSW (Continued)

Getting back to where we left off, let's investigate the remainder of the program status word. Bit 4 (D3) is not used, and when it is read, a logic 1 will appear in this place.

The next bit location to the left indicates which register bank is being utilized. When you do either a SEL RB0 or a SEL RB1 instruction, it is this bit that is affected.

D5 is a general-purpose readable and writable flag; it's called F0. In fact, elsewhere within the microcomputer is another flag bit called F1.

These flag bits can be used whenever you need someplace to mark the fact that something has occurred.

If your program determines, for instance, that you are in a certain mode (like "transmit" in a radio controller), this bit may be used to mark and store that fact. Any time you need to know what mode you're in, just read the flag bit.

A number of instructions will manipulate or utilize both flags:

```
CLR F0    JF0
CLR F1    JF1
CPL F0
CPL F1
```

The first set (CLR) does just what it implies. The contents of flags 0 and 1 will be zero after each are executed. The next set assumes that the contents will be the exact opposite of what they were previous to the instruction (CPL).

The column on the right is our first encounter with conditional branching. JF0 means that if the contents of F0 is 1, then a jump to a certain address will occur. The value of the place to go is contained in the following byte. You may traverse 256 places, which is one page. As you can see,

both flags may be set, cleared and tested. What more could you want?

Bit 6 of the PSW and bit 7 are carry flags. The first one (AC), the auxiliary carry, is active following the DA (decimal adjust) instruction we talked about earlier. When an overflow occurs between BCD digits, this flag goes positive.

The other carry bit is used to indicate that the accumulator has overflowed (going from FF to 00) during the previous instruction. This is helpful when doing addition; it indicates when a carry-over operation must be

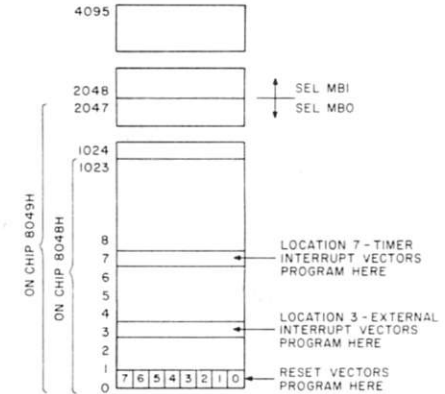


Fig. 3. Map of internal program memory.

Circle 166 on Reader Service card.

# FREE SHIPPING

## IBM® Personal Computer Products

Davong 5 MB Hard Disk System - \$1495.00 12 MB - \$1995.00

IBM PC-2 Drive System **\$ CALL**  
3" DUAL DRIVE SUBSYSTEM **\$725.00**

**Quadram** - Quadboard with Parallel

Port, Serial Port, Clock/Calendar, Expandable to 256K

64K on brd. - \$340.00 128K on brd. - \$395.00  
192K on brd. - \$439.00 256K on brd. - \$499.00

### Quadram Memory Expansion

192K Maximum  
64K on brd. - \$230.00 128K on brd. - \$290.00  
192K on brd. - \$350.00

AST & PERYSYST MEMORY EXPANSION PRODUCTS **\$ CALL**

### Amdek Monitors

Mod 300 Phosphor - \$150.00 Composite Color III - \$345.00  
IBM RGB Compatible Color II - \$599.00 Color I - \$300.00

### IBM/TRS 80 Disk Drives/Cabinets

TM 100 Single 40 Track Drive - \$189.00  
TM 100-2 Double 40 Track Drive - \$280.00

TEAC 5 1/4" SLIM SINGLE & DOUBLE DRIVE SUBSYSTEMS **\$ CALL**

## Apple II® Computer Products

SYSCOM APPLE COMPATIBLE SYSTEM **\$725.00**

Apple Compatible Controller Card... **\$ 79.95**

Apple Compatible Disk Drive w/Cabinet & Cable... **249.00**  
w/Controller... **315.00**

TEAC SLIM LINE 5 1/4" DRIVE **\$265.00** DUAL SLIM LINE **\$525.00**

Printer/Graphics Interface... **99.95**

Davong 5 MB Hard Disk System - \$1495.00 - 12 MB - \$1995.00

Apple Compatible Joysticks... **29.95**

## Epson/Smith-Corona Printers

MX80 **\$425.00** FX80 **\$550.00** MX100 **\$675.00**

TRS 80 / Parallel Printer Cable **\$20.00**

IBM Parallel Printer Cable **\$35.00**

STAR MICRONICS GEMINI 10 **\$ CALL**

GEMINI 15 **\$ CALL**

Smith Corona TP-1 Letter Quality Daisy Wheel... **\$575.00**



TRS-80 MOD III Disk Controller Incl: Disk Controller, Power-Supply, Mounting Hardware, Cables & Instruction Manuals **\$239.00**

POWER SUPPLIES AND CABINETS Dual 8" Slim Line - \$180.00 Dual 5 1/4" - \$ 99.00  
Single 5 1/4" - \$ 69.00

3" DUAL DRIVE SUBSYSTEM FOR IBM... **\$725.00**

VISA, MASTERCARD (\$100 Min., Add 2%)

Or Certified Check

90 Day Warranty (Parts & Labor)

TRS 80 is a Registered Trademark, Tandy Corp.

Prices Subject to Change Without Notice

**DATA-MAIL**  
P.O. Box 818, Reseda, CA 91335  
1-800-635-5555

FREE SHIPPING IN CONTINENTAL U.S.

**(213) 993-4804**  
(IN CALIF.)

performed.

It's possible at any time to change the entire contents of the PSW simply by moving the data resident in the accumulator directly into it. The MOV PSW,A allows this.

Conversely, the contents of the PSW may be moved into the accumulator by executing a MOV A,PSW instruction.

### Arithmetic and Logic Functions

That about does it for the PSW. Those carry flags bring up another group of instructions. The list below describes the range of arithmetic and logic operations that come with the controller.

ADD A,Rr	ORL A,Rr
ADD A,#data	ORL A,#data
ADDC A,Rr	XRL A,Rr
ADDC A,#data	XRL A,#data
ANL A,Rr	
ANL A,#data	

Notice that each operation has two methods of getting the immediate data to operate on the accumulator.

The first, ADD A,Rr, adds the contents of A to the contents of the register specified (0-7); the result is left in the accumulator. The next does the same, except it tests the carry flag to see if the previous operation overflowed.

The 8748 has the capability of performing logical ANDs, ORs and Exclusive ORs on its data. The next three sets illustrate the operations as they exist in the chip. Either the data immediately following or the specified register provides the other operand.

Arithmetic with computers can be performed in a number of ways. It's possible to multiply the contents of a register or accumulator by shifting the bits one position to the left. If you were to write an algorithm to perform this shift using MOVE, AND and OR

instructions, it would be relatively complex. The 8748 allows you to perform shifts, either left or right, with the use of one-byte op-codes.

RL A	RR A
RLC A	RRC A

The op-codes above pertain to the shift instructions. The left column will shift the contents of the accumulator one bit to the left. At that time, a zero will be placed into bit 0. The next one will shift left into the carry flag.

Bit 7 will replace the previous information in the flag, which will be placed in bit 0 (a vicious circle). The column on the right will shift everything to the right.

### Register Operations

Throughout these discussions, both register banks that are resident on-chip have been briefly mentioned. You have seen bank selection instructions, registers used as immediate data and registers used as address pointers.

One section of instructions is dedicated specifically to these registers. Let's look at them:

MOV A,Rr	INC Rr
MOV Rr,A	DEC Rr
MOV A,#data	INC @R0
MOV Rr,#data	INC @R1
XCH A,Rr	DJNZ Rr, addr

The first two on the left allow you to traffic data between any register and the accumulator. Both of these and the last one in the column move between the two places.

Of course, XCH will change both locations where the MOVs affect only the destination, be it register or accumulator. The MOV A,#data and MOV Rr,#data are two-byte instruc-

tions that allow you to set a value into either. The second byte contains the data to be transferred.

The column on the right allows you to add one or subtract one from any register.

By the way, these instructions that specify an Rr are only one byte in length. Each register has its associated op-code. So the INC Rr instruction is really eight separate ones! The same applies for any register-based instruction.

---

There are many opportunities  
within the instruction set  
to catapult yourself  
to another location.

---

The INC @R0 and INC @R1 instructions really pertain to manipulating internal data memory and should have been covered earlier. I guess they were inadvertently swept under the rug, even though they're so useful. Any location in data memory pointed to by either R0 or R1 may have its contents incremented.

Last, but certainly not least, is the DJNZ instruction. This operation is worth its weight in gold. With it, you can decrement the contents of a specified register and test to see if it has become zero. If the test proves negative, it will jump you to the address specified in the following byte. (It does all this in one instruction.)

It's useful in counting delays or counting anything, and you'll find that the register array included on-chip also can be helpful.

In many instances, you may not even need to use the data-memory storage area. These registers are important because you can use them not only for data storage but for pointing to other locations during jump operations.

### Branches, Jumps And General Leap-Frogging

You have seen, by now, that there are many opportunities within the instruction set to catapult yourself to another location. Typically, this kind of leap-frogging would occur whenever a decision block is encountered.

Within the 8748, a number of specific conditions may be tested, and a jump can be performed as a result. The general JMP instruction has been

Circle 334 on Reader Service card

IT'S HERE

## COMPOSE YOURSELF

### MUSIC FOR THE TIMEX-SINCLAIR COMPUTER

You've balanced your checkbook, calculated your biorhythms and chased the Klingons out of the galaxy—now isn't it about time to do something creative?

Introducing  
**MOZART**

- plugs into all ZX81 and TS1000 without modification.
- allows addition of RAM pack and other modules.
- three pitched voices for musical tones and noise generator for sound effects.
- programmable in BASIC or machine code.

- menu-driven screen oriented music editor allows user to enter, save, or modify a musical composition.
- three voices may be individually arranged to be played, transposed, or repeated.
- complete instruction manual with programming examples included.

yours for  
a song **\$79.91**

572 AVENIDA DE LA PLATA  
NEWBURY PARK, CALIF. 91320 (805) 498-1735

• \$3.00 shipping and handling  
• 15 day unconditional money-back guarantee

covered, as have the JF0, JF1 and DJNZ commands. The list below outlines the rest of them.

JMPP,@A	JT0,addr	JBN,addr
JC,addr	JNT0,addr	JTF,addr
JNC,addr	JT1,addr	
JZ,addr	JNT1,addr	
JNZ,addr	JNI,addr	

Quite a list, isn't it? Let's start with the left column.

The first is a complex table-oriented jump, much like the MOVP,@A instruction. Here, the contents of the program memory location pointed to by the number in the accumulator will replace the low-order eight bits of the program counter. It will jump you anywhere within the current page.

This instruction may come in handy

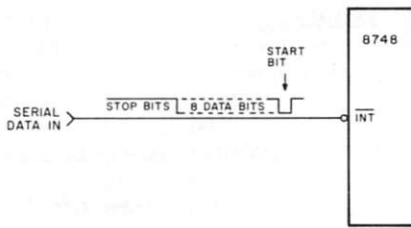


Fig. 4. Method of receiving serial data.

if you are running a machine that is receiving codes from an external terminal. Each incoming code catapults you into specific command handlers at varying places within program memory.

The next four will test the state of the carry flag or test whether the contents of the accumulator is zero and take appropriate action.

Moving into the middle column, we see the opportunity to test the state of the T0 and T1 input pins of the chip. Of course, if you were to use T0 as an input, you couldn't perform the ENT0 CLK instruction. You will find that these input pins may come in handy for receiving serial data or for counting applications.

Another instruction that can be used for serial information is the JNI command. This one will jump to the specified address whenever the interrupt line is low. (Now that may be confusing.)

Let me explain. Remember I said that you can disable interrupts? Well, when they're disabled, you can use this line as a general-purpose input line by employing this test instruction. Where does the serial data come in?

Let's look at an example of receiving data (Fig. 4).

The initial low-going pulses of the start bit will trigger an interrupt. At this time, your service routine should provide the ability to disable further interrupts and start looking at (polling) the INT line with the JNI instruction.

By using the internal timer, you should be able to count bit times and successfully receive the serial data word. Of course, it would be necessary to know in advance what bit rate the data will be clocking in at.

The value that you set in the timer would be one-bit time. If the INT line remains low for one complete bit time, then the data bit is a zero. Moving through the word, after the eighth bit has been received, you start looking for high-stop bits. When the complete word is finished being processed, simply enable the interrupts to catch another transmission.

The column on the right shows only two more jump instructions. In actuality, the first, JBN, is a general name for eight separate jumps. Each bit of the accumulator may be tested for a logic 1 level using this instruction.

Circle 230 on Reader Service card.

# UOLISP

## An Optimizing Compiler and Assembler

### An Excellent System for A.I. Robotics & Intelligent Systems



*Documentation*  
Comprehensive manual covering all aspects of the system. Numerous examples of each facility are included.


*Interpreter*  
Over 125 functions implemented in base interpreter  
Additional Development Software Available

*Fast Load Libraries*  
Compiled code can be stored in relocatable files

*Requirements*  
CP/M System -- Also available for TRS-80 Model I or Model III. 48K. dual disks.

*Ordering*  
System Manual -- \$20  
Complete System -- \$150

VISA and Mastercard  
Please include expiration date and Card No.



**FAR WEST SYSTEMS & SOFTWARE, INC.**  
P.O. BOX 3301 Eugene Oregon Ph (503) 485 5155

Circle 262 on Reader Service card.

## Epson, OKI, IDS, NEC, Diablo, Qume



### ACOUSTIC ENCLOSURES

- Reduces Noise Up to 90%
- Bottom Feed Capability
- Heavy Duty Acrylic Cover
- Woodgrain Finish

**SAVE!**  
**SAVE!**  
FROM  
**\$99<sup>00</sup>**

**Micro Printercenter™**

### Dealer & Ordering Info

**800-343-4311**  
Master Charge and Visa Accepted  
Shipping & Handling Charges Additional

## CAB-TEK, Inc.

Riverside St. Nashua, NH 03062  
CIVILIZING COMPUTERS

MPC I \$99 (MX 80) MPC II \$129 (OKI82)  
MPC III \$179 (83A, MX100) MCP IV \$199 (Daisy Printer)  
Power Control & Ventilation \$80  
Paper Rack \$30 Bottom Feed Brackets \$30  
MPC I SHOWN

```

1 7 8748 DEVELOPMENT SYSTEM
2
3
5 CLEAR 5000
10 CLS:PRINT"ONE MOMENT PLEASE":DIM CODE(1024)
15 FOR X = 0 TO 1023:CODE(X) = 0:NEXT X
20 CLS:PRINT "MCS-48 DEVELOPMENT SYSTEM"
30 OUT=128:IN=144:BUS=49152:VOLTS=247
40 P2=49153:P3=49154:CTL=49155:STATUS=176
50 GOSUB 200
60 PRINT:PRINT "SELECT FROM THE FOLLOWING"
70 PRINT:PRINT "1) PROGRAM 8748"
80 PRINT "2) READ 8748"
90 PRINT "3) EMULATE 8748"
100 INPUT A
110 ON A GOTO 1000, 2000, 3000
120 GOTO 10
200 POKE(CTL),IN
210 POKE(P3),VOLTS
220 POKE(P2),STATUS
230 RETURN
500 HI=0:LO=0:POKE(CTL),OUT
505 L=X
510 IF L=> 512 THEN HI=HI+ 2: L = L - 512
520 IF L=> 256 THEN HI=HI OR 1: L = L - 256
530 LO=L:HI=HI OR VOLTS
540 POKE(BUS),LO
550 POKE(P3),HI
560 RETURN
1000 CLS:PRINT "8748 PROGRAMMER"
1005 GOSUB 1010
1007 GOTO 1240
1010 PRINT:PRINT "PROGRAM IS IN:"
1020 PRINT "1) SYSTEM RAM"
1030 PRINT "2) CASSETTE FILE"
1040 PRINT "3) TO BE ENTERED BY KEYBOARD"
1050 INPUT A:IF A > 3 THEN GOTO 1000
1060 IF A = 3 THEN GOTO 1150
1070 IF A = 1 THEN GOTO 1235
1080 IF A = 2 THEN INPUT "FILE NAME =":A$
1090 OPEN "I",-1,A$
1100 FOR X = 0 TO 1023
1110 IF EOF(-1) THEN GOTO 1240
1120 INPUT #=-1, CODE(1024)
1130 NEXT X
1140 GOTO 1240
1150 CLS
1160 INPUT "ENTER START ADDRESS":A
1170 INPUT "ENTER END ADDRESS":B
1180 CLS:FOR X = A TO B
1190 PRINT X:=""
1195 INPUT C$
1200 IF C$ = " " THEN GOTO 1235
1210 IF C$ = "R" THEN X = X - 1:GOTO 1190
1220 CODE(X) = VAL(C$)
1230 NEXT X
1235 RETURN
1240 PRINT "INSERT 8748 INTO SOCKET"
1250 PRINT"PRESS <ENTER> TO PROGRAM"
1260 INPUT A
1270 CLS:PRINT"PROGRAMMING LOCATION"
1280 FOR X = 0 TO 1023
1300 VOLTS = VOLTS AND 251
1310 POKE(P3), VOLTS
1320 VOLTS = VOLTS AND 223
1330 POKE(P3), VOLTS
1340 GOSUB 500
1350 VOLTS = VOLTS OR 8
1360 POKE(P3), VOLTS
1370 POKE(BUS), CODE(X)
1380 VOLTS = VOLTS AND 239
1390 POKE(P3), VOLTS
1400 VOLTS = VOLTS AND 191
1410 POKE(P3), VOLTS
1420 VOLTS = VOLTS OR 64
1430 VOLTS = VOLTS AND 127
1440 POKE(PE), VOLTS
1450 FOR T = 1 TO 5:NEXT T
1460 VOLTS = VOLTS AND 191
1470 VOLTS = VOLTS OR 128
1480 POKE(P3), VOLTS
1490 VOLTS = VOLTS OR 64
1500 POKE(P3), VOLTS
1510 VOLTS = VOLTS OR 16
1520 POKE(P3), VOLTS
1530 POKE(CTL), IN
1540 VOLTS = VOLTS OR 4
1550 POKE(P3), VOLTS
1560 VFY= PEEK(BUS)
1570 IF VFY <> CODE(X) THEN PRINT "PROBLEM AT LOCATION":X:GOTO 1240
1580 PRINT@21,X:NEXT X
1590 GOSUB 200
1600 PRINT "PROGRAMMING COMPLETE"
1610 PRINT "YOU MAY REMOVE DEVICE"
1620 PRINT "PRESS <ENTER> FO MENU"
1630 INPUT A:GOTO 20
2000 CLS
2010 PRINT "INSERT 8748 INTO SOCKET"
2020 PRINT "PRESS <ENTER> TO READ"
2030 INPUT A
2040 CLS
2045 PRINT"READING LOCATION"
2050 FOR X = 0 TO 1023
2060 VOLTS = VOLTS AND 223
2070 POKE(P3), VOLTS
2080 GOSUB 500
2090 VOLTS = VOLTS OR 8
2100 POKE(P3), VOLTS
2110 POKE(CTL), IN
2120 CODE(X) = PEEK(BUS)
2130 VOLTS = VOLTS AND 247
2140 POKE(P3), VOLTS
2150 PRINT@17,X:NEXT X
2155 GOSUB 200
2157 FOR X = 0 TO 1023
2160 IF C <> 0 THEN GOTO 2170
2162 NEXT X
2165 PRINT "PART IS ERASED"
2168 GOTO 2350
2170 PRINT "READING COMPLETE"
2180 PRINT:PRINT "PRESS <1> FOR MENU"
2190 PRINT "PRESS <2> TO EXAMINE MEMORY"
2200 PRINT "PRESS <3> TO ENTER PROGRAM MODE"
2210 INPUT A
2220 ON A GOTO 20, 2230, 1240
2230 CLS:PRINT "EXAMINE MEMORY"
2240 PRINT:PRINT "ENTER START ADDRESS"
2250 INPUT A:CLS
2260 FOR X = A TO 1023
2270 PRINT X "=" CODE(X)
2280 A$=INKEY$:IF A$="" THEN 2280
2290 IF A$ = " " THEN GOTO 2330
2300 IF A$ = "R" THEN X = X - 1:GOTO 2270
2310 IF A$ = "0" THEN GOTO 20
2320 CODE(X) = VAL(A$)
2330 NEXT X
2340 PRINT "END OF MEMORY"
2350 PRINT "PRESS <ENTER> FOR MENU"
2360 INPUT A
2370 GOTO 20
3000 CLS
3010 PRINT "8748 EMULATOR"
3020 PRINT "INSERT 8035 INTO SOCKET"
3030 PRINT "PRESS <ENTER> TO BEGIN"
3040 INPUT A
3050 CLS
3055 GOSUB 1010
3059 CLS:PRINT"LOADING EMULATION RAM"
3060 FOR X = 0 TO 1023
3070 GOSUB 500
3080 STATUS = STATUS AND 223
3090 POKE(P2), STATUS
3100 STATUS = STATUS OR 64
3110 POKE(P2), STATUS
3120 STATUS = STATUS AND 191
3130 POKE(P2), STATUS
3140 POKE(BUS), CODE(X)
3150 STATUS = STATUS AND 127
3160 POKE(P2), STATUS
3170 STATUS = STATUS OR 128
3180 POKE(P2), STATUS
3190 STATUS = STATUS OR 32
3200 POKE(P2), STATUS
3210 PRINT@22,X:NEXT X
3220 CLS:PRINT "EMULATOR"
3230 PRINT:PRINT "1) TO BEGIN RUNNING"
3240 PRINT "2) TO STOP"
3250 PRINT "3) TO RETURN TO MENU"
3260 INPUT A
3270 IF A = 1 THEN VOLTS = VOLTS OR 8:POKE(P3), VOLTS:PRINT:PRINT "PROGRAM RU
NNING":GOTO 3230
3280 IF A = 2 THEN VOLTS = VOLTS AND 247:POKE(P3), VOLTS:PRINT:PRINT "PROGRAM
STOPPED":GOTO 3230
3290 IF A = 3 THEN GOTO 20
3300 GOTO 3220

```

Listing 1. Basic program that operates 8748 development system.

Substituting the numbers 0 through 7 into the n spot will call out the bit you wish to test. This instruction is particularly useful in I/O processing, where you are looking for the state of a certain bit out of a field of eight to change.

The last jump has to do with the internal timer. Up until now, this piece of hardware has not been covered.

### Timer

Included within the 8748 is a general-purpose eight-bit counter that

may be operated under program control. Counters come in handy when long time delays must be utilized. The processor need only set up the counter value, connect some constant frequency clock to it and go off doing other things while it is busy counting

# HOW TO CONTROL YOUR CP/M™ MICRO—AT ONLY \$3. PER COMMAND.

## A Remarkable Program For CP/M Users.

Of course, CP/M is a wonderful operating system. That's why so much serious business software has been created for it.

BUT, CP/M is not easy to work with. That's why you need to take the **POWER!** trip.

**POWER!** is a super-power-packed, user-friendly program that lets you take immediate and complete control of CP/M. And at a cost of only \$3. per command, it's the software buy of the year.

### Over 55 Housekeeping Utilities.

**POWER!** is over 55 prompted, user-friendly CP/M utility programs all rolled into one 15k package. It takes care of all of these frustrations and more:

—**BDOS errors? POWER!** ends BDOS errors and gives you a way out.

—**Accidentally erased a file?** If you accidentally erase a program or disk file, **POWER!** restores the erased files.

### —Can't remember file names?

**POWER!** assigns a number to each file on your disk. So, to copy files from disk to disk, you don't have to fiddle with PIP anymore. You just pick the file from a numbered menu and **POWER!** copies it for you. No more typing errors! **POWER!** also marks original files and their copies for you; and you can compare files to find identical copies regardless of name.

—**Lose data on a glitched disk?** If a glitched disk makes it impossible to call up a long word processing text, **POWER!** can fix the glitch. This means you may have to retype only a couple of sentences instead of losing 20 pages of text.

### —Trouble with "bargain" disks?

**POWER!'s** disk testing function gathers any bad sectors of the disk into a special file so that CP/M thinks those parts of the disk are already used and never attempts to write to them. The rest of the disk is then safe to use.

### —CP/M scrolls too fast through text files?

**POWER!** spools through files for you,

page by page, file by file, or line by line with instant halt by touching the space bar.

### —Need to reorganize files?

**POWER!** sorts and formats the directory in 4 different ways. And you can easily copy or move files from user area to user area. **POWER!** creates 32 user areas instead of CP/M's 16.

### —Need to change memory?

**POWER!** searches, displays and lets you change memory wherever you want. You can even automatically run software anywhere in memory. And you can inter-mix your search with as many wild card jokers as you need to find, for instance, all occurrences of "Sam Jones" and "Sid James" just by typing "S??J??". And **POWER!** also lets you read or write to any sector or track very simply.

### —Changing disks? You can forget the ubiquitous Control C to change disks.

**POWER!** can do it for you automatically. And **POWER!** doesn't require a system disk in any drive, so Drive A is open for use, when **POWER!** is in control of CP/M.

### —Afraid of HEX numbers? POWER!

automatically converts Hex to Decimal, Binary or ASCII.

### Special Password Protection, Too.

**POWER!** now includes a special program that lets you lock sensitive files, so that only you can access them. Without the secret PASSWORD which you can create and change at will, no prying eyes will ever know your secret file even exists. A great way to protect financial or scientific data from unauthorized eyes. Just this single program alone would be worth the price of **POWER!**, but there are over 55 more just as valuable programs in this power-packed-package.

### At \$169., It's A Bargain.

Space doesn't permit describing all

the wonderful ways

### POWER!

can put you in complete control of your CP/M micro. But see for yourself. There's a Money Back Guarantee. At the low price of \$169., each powerful command costs you less than \$3. A true bargain!

### POWER! Is Better Than Ever!

Eventhough "InfoWorld", "Microsystems" and "Interface Age" call **POWER!** great, we have improved **POWER!**—including a completely rewritten 120-page easy-to-read documentation. (Previous purchasers of **POWER!** may exchange their original disk for an updated version with the new commands and a brand new manual—for only \$35.)

### Take The POWER! Trip Today!

**POWER!** will operate in any standard CP/M or MP/M system, including CP/M-86, IBM PC, Apple (Z80 card), Osborne, Kaypro, HP, TeleVideo, TRS-80 conversions, S100's including NorthStar, Vector, Morrow, CompuPro, etc. Up and running immediately, no configuration necessary—for hard disks and floppies.

At only \$3. per command, you can afford to **Take the POWER! Trip.** Call or send in your order today.

### NOW AVAILABLE FOR MS-DOS, TOO.

**ONLY \$169. Money Back Guarantee.** Charge & COD Orders Welcome.

**TOLL FREE (800) 428-7825 Ext. 962**  
**IN CA: (800) 428-7824 Ext. 962**  
**DEALERS AND OEM'S (415) 567-1634 Ext. 962**

**InfoWorld**  
Software Report Card

**Power**  
Version 2.55

	Poor	Fair	Good	Excellent
Performance	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Documentation	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Ease of Use	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Error Handling	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

© 1982 by Popular Computing, Inc. A subsidiary of CW Communications Inc., Framingham MA. Reprinted from InfoWorld.



# Take The POWER!™ Trip.

## COMPUTING!

2519Z Greenwich San Francisco, CA 94123

**TOLL FREE (800) 428-7825 Ext. 962**  
**IN CA: (800) 428-7824 Ext. 962**  
**DEALERS AND OEM'S (415) 567-1634 Ext. 962**

**ONLY \$169.** Calif. add 6½% sales tax  
 CP/M \$169.  CP/M-86 \$169.  MP/M \$249.

Card No. \_\_\_\_\_  
 Exp. Date \_\_\_\_\_  
 Computer \_\_\_\_\_  
 Your Name \_\_\_\_\_  
 Company Name \_\_\_\_\_  
 Address \_\_\_\_\_  
 City/State/Zip \_\_\_\_\_



the time away.

Think of it as the timer built into some ovens. If you had to sit in front of the oven all day waiting for a roast to cook, you'd get nothing else done.

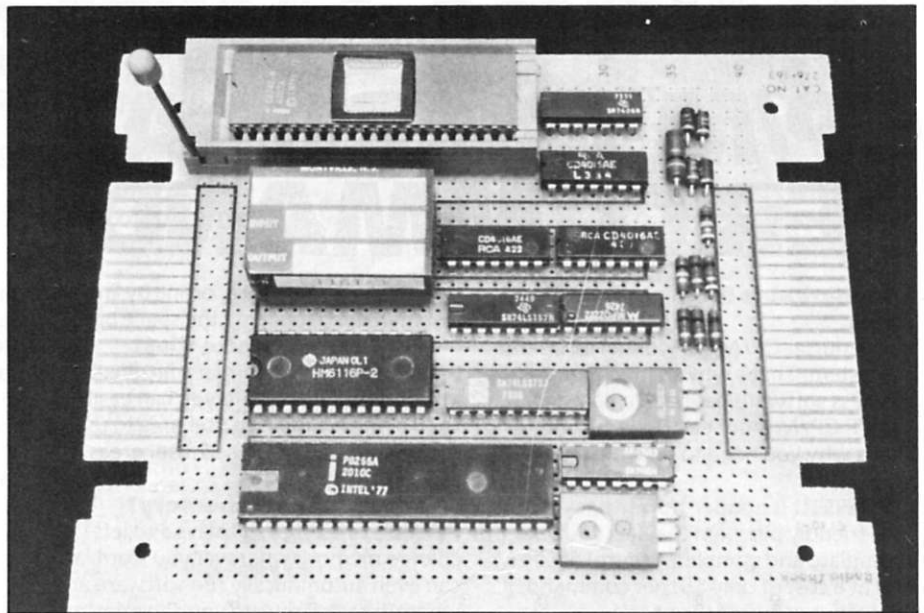
This particular counter only counts up. That is, every clock pulse will increment the value contained in it. When it overflows (FF to 00), there is a flag associated within the hardware that gets set. It is this flag that the JTF instruction tests.

You also have the option of allowing an interrupt to occur upon overflow. When this is enabled, through the use of the EN TCNTI instruction, the program counter will vector you to location 007 in program memory.

The same rules apply to this interrupt as in the general purpose external one. Disabling it is done by performing a DIS TCNTI instruction.

In reality, there are two modes of operation for the counter. The first, and most used, is the timer mode. This is where a known time base clock is applied to increment the value. The time base is derived from the master clock oscillator. (Fig. 5 shows the internal hook-ups of the two modes.)

As you can see, in the timer mode a STRT T instruction will connect the



View of the programmer/emulator board.

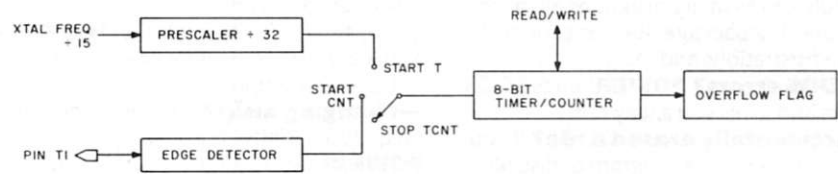


Fig. 5. Internal timer operation.

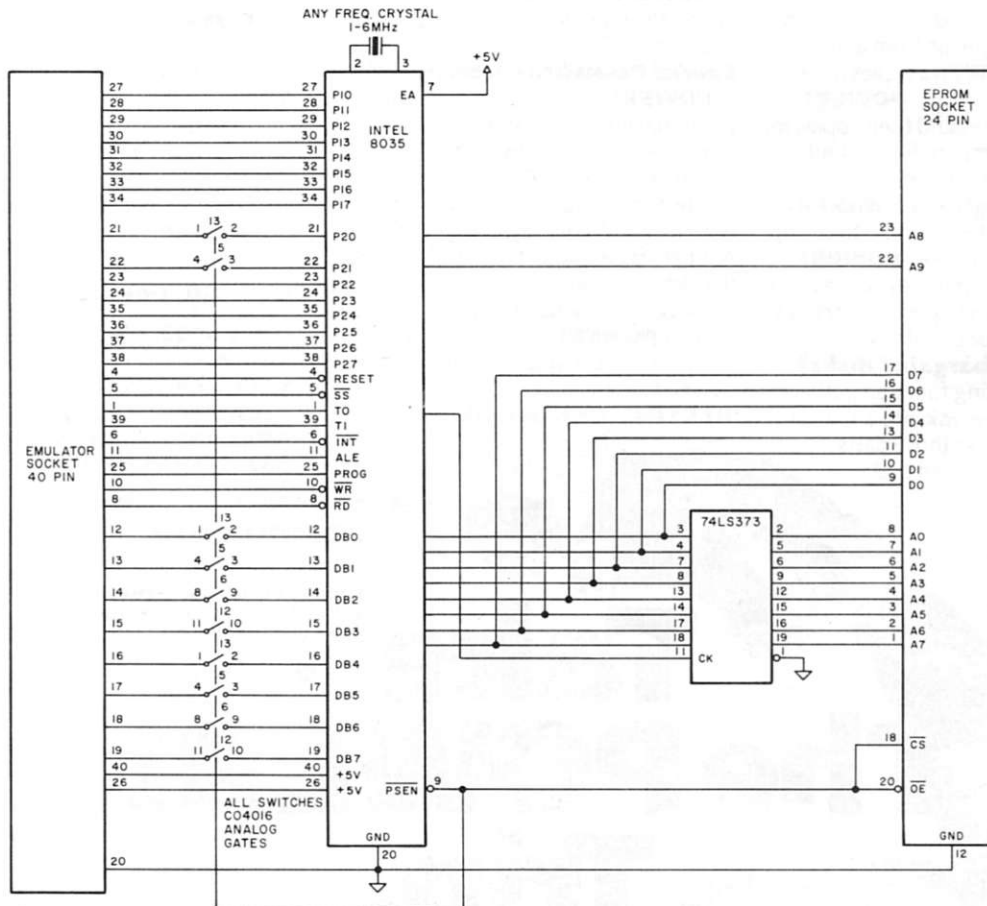


Fig. 6. Schematic of stand-alone 8748 emulator.

counter to the master clock (actually, the crystal frequency divided by 15) through a divide-by-32 prescaler.

Initial values may be entered into the counter through the use of the MOV T,A instruction. Conversely, the contents of the timer may be read at any time through the use of MOV A,T.

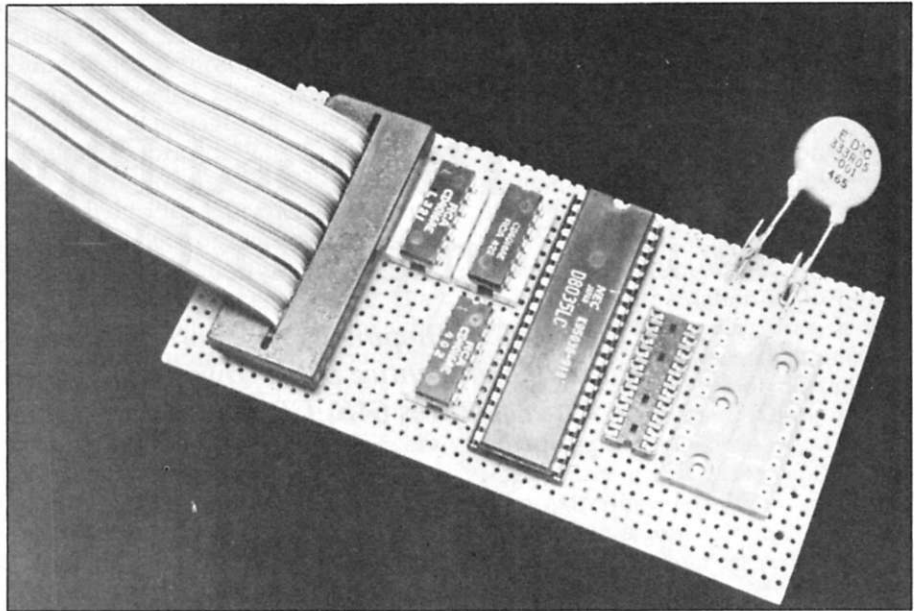
When using the counter mode, the multipurpose T1 input becomes the clock source. This allows you to count events that may happen and interrupt when enough have occurred.

The only other instructions not covered so far deal with I/O. These will be covered quite thoroughly next month, when we'll design and construct the Command Communicator of the Unimem system.

### Program Emulation

Emulation is performed through the use of an 8035 part. This basically is an 8748 without the internal ROM. External program memory is connected as it was described previously. (See photos for a picture of the complete programmer/emulator and for a picture of a stand-alone emulator.)

You can insert either an EPROM or another 1K memory part into the program memory socket.



The emulator board.

I have built a 1K ROM emulator that plugs into the socket. With it I can change, via keyboard commands, the data. The programmer/emulator, however, has this programmable feature built in.

At this time, look at Listing 1; this is the Basic program that operates the

board. I've tried to add enough remarks so that it's self-documenting.

Throughout the next few months, you'll find yourself getting deeper into the programming and use of the 8748. I hope you've found this series, so far, a useful introduction to small-size intelligence. ■

Circle 24 on Reader Service card.

# RIBBONS

Low Price • FREE Shipping  
SATISFACTION GUARANTEED

## RIBBON CARTRIDGES top quality factory fresh

Cartridges for use on these printers:	price each in quantity of			
	1-5	6-23	24-99	100 +
MX-70, MX-80, IBM PC	7.41	6.45	5.61	4.88
MX-100	19.96	17.36	15.09	13.13
Prowriter, PC 8023A-01	7.98	6.94	6.04	5.25
RS LP2, LP3, LP5	7.98	6.94	6.04	5.25

## RIBBON LOOPS

top quality nylon refills for your old cartridge

Loops for use on these printers:	price each in quantity of			
	1-5	6-23	24-99	100 +
MX-70, MX-80, IBM PC	3.56	3.09	2.69	2.34
MX-100	5.41	4.71	4.09	3.56
Prowriter, PC 8023A-01	2.46	2.14	1.86	1.62
RS DMP 400, LP6, LP8	2.04	1.77	1.54	1.34
RS DMP 200, DMP 500	3.56	3.09	2.69	2.34
RS LP2, LP3, LP5	2.46	2.14	1.86	1.62
Spinwriter (nylon)	2.46	2.14	1.86	1.62

Cartridges and loops may be mixed for quantity prices. Our FREE CATALOG includes loading instructions for loops. Discounts available for schools. Florida res. add 5% tax.

VISA **DATA SYSTEMS** MasterCard  
(305) 788-2145 • Box 99 • Fern Park, FL 32730

Circle 197 on Reader Service card.

## First came MICROPROOF™:

"There is simply no finer program available..."  
(Creative Computing, March 1982)

Now:

# Electric Webster SPELLING CHECKER

The Ultimate:

- FAST**—Can proof ten pages in a minute
- EASY**—Operates at the stroke of a key
- COMPLETE**—50,000 word literal dictionary
- COMPACT**—Fits on 5¼" double density disk
- VERSATILE**—Use with all W P programs
- CORRECTS**—(Optional, add \$60.00)
- AFFORDABLE**—\$89.50 (TRS-80®), \$149.50 (CP M®)

**CORNUCOPIA  
SOFTWARE, INC.**

1625 Beverly Place  
Berkeley, CA 94707

Contact your local dealer, or order direct — (415) 524-8098