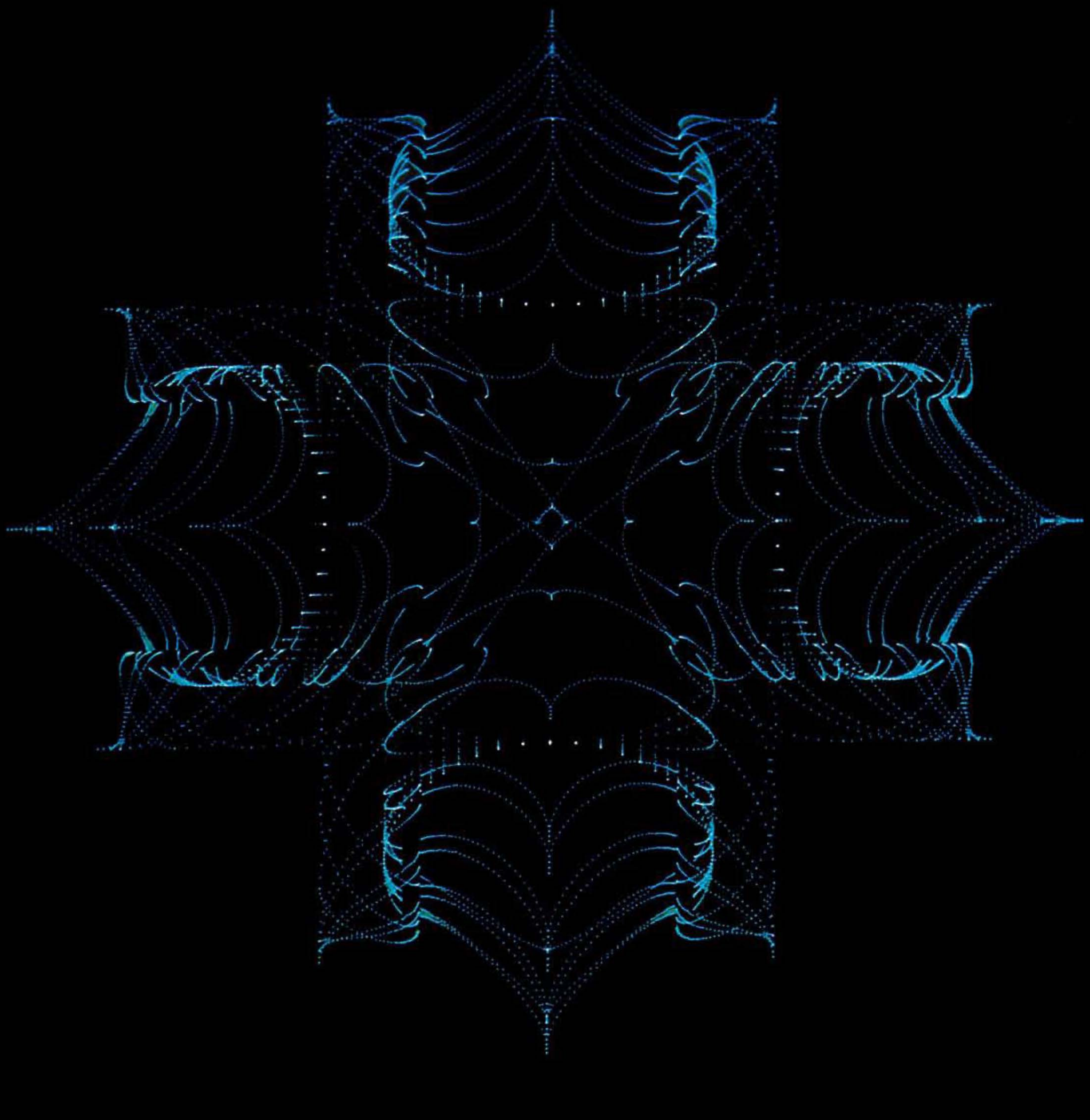
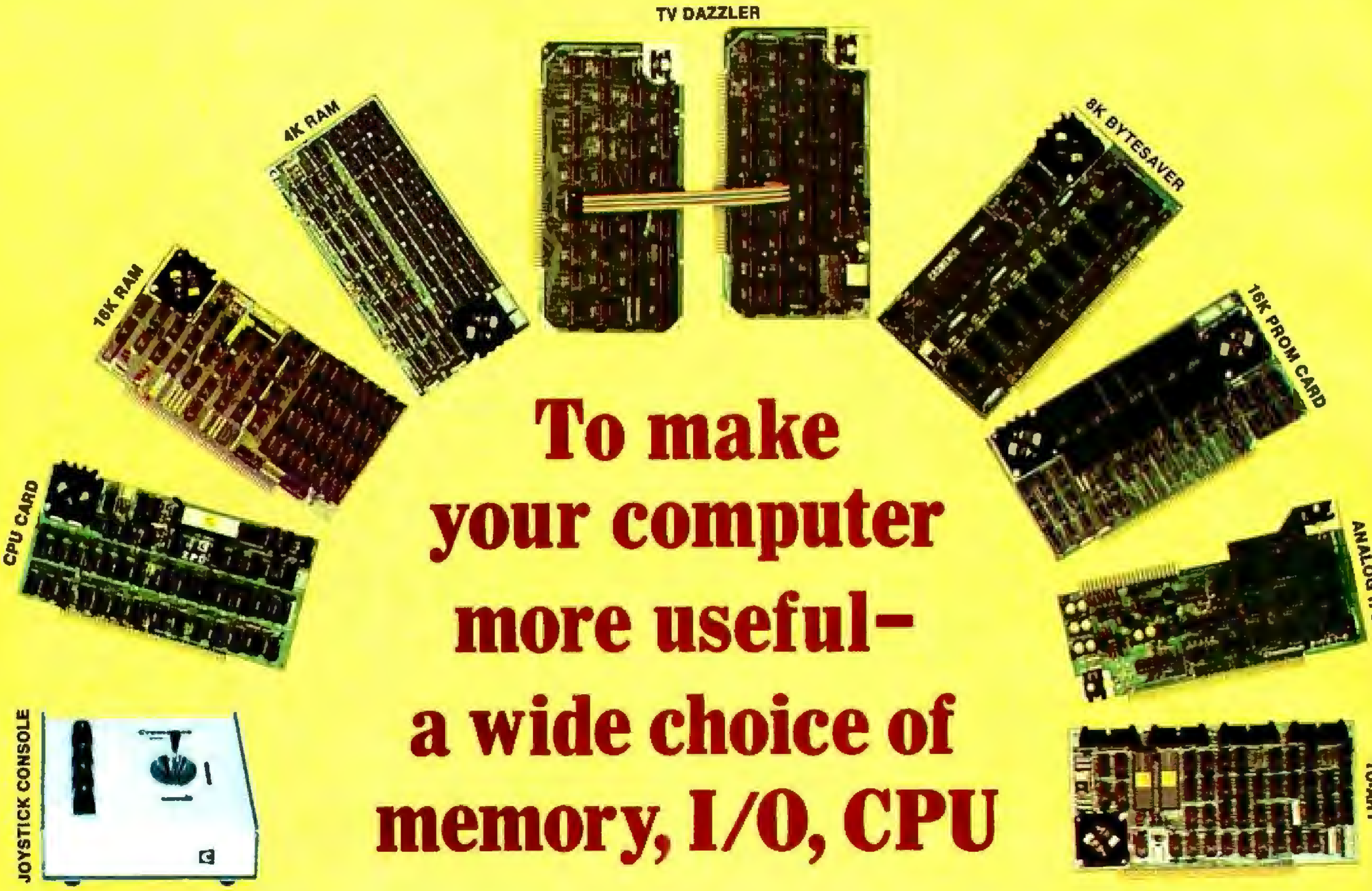
OCTOBER 1977 VOLUME 2, Number 10 $2.00 in USA

BYTE

the small systems journal





To make your computer more useful– a wide choice of memory, I/O, CPU

Your computer's usefulness depends on the capability of its CPU, memories, and I/O interfaces, right?

So here's a broad line of truly useful computer products that lets you do interesting things with your Cromemco Z-1 and Z-2 computers. And with your S-100-compatible Altairs and IMSAIs, too.

CPU

- Z-80 MICROPROCESSOR CARD. The most advanced μP card available. Forms the heart of our Z-1 and Z-2 systems. Also a direct replacement for Altair/IMSAI CPUs. Has 4-MHz clock rate and the power of the Z-80 μP chip. Kit (Model ZPU-K): $295. Assembled (Model ZPU-W): $395.

MEMORIES

- 16K RAM. The fastest available. Also has bank-select feature. Kit (Model 16KZ-K): $495. Assembled (Model 16KZ-W): $795.
- 4K RAM. Bank-select allows expansion to 8 banks of 64K bytes each. Kit (Model 4KZ-K): $195. Assembled (Model 4KZ-W): $295.
- THE BYTESAVER — an 8K capacity PROM card *with integral programmer*. Uses high-speed 2708 erasable PROMs. A must for all computers. Will load 8K BASIC into RAM in less than a second. Kit (Model BSK-0): $145. Assembled (Model BSW-0): $245.
- 16K CAPACITY PROM CARD. Capacity for up to 16K of high-speed 2708 erasable PROM. Kit (Model 16KPR-K): $145. Assembled (Model 16KPR-W): $245.

I/O INTERFACES

- FAST 7-CHANNEL DIGITAL-ANALOG I/O. Extremely useful board with 7 A/D channels and 7 D/A channels. Also one 8-bit parallel I/O channel. Kit (Model D + 7A-K): $145. Assembled (Model D + 7A-W): $245.
- TV DAZZLER. Color graphics interface. Lets you use color TV as full-color graphics terminal. Kit (Model CGI-K): $215. Assembled (Model CGI-W): $350.
- DIGITAL INTERFACE (OUR NEW TU-ART). Interfaces with teletype, CRT terminals, line printers, etc. Has not one but *two serial I/O ports* and *two 8-bit parallel I/O ports* as well as 10 on-board interval timers. Kit (Model TRT-K): $195. Assembled (Model TRT-W): $295.
- JOYSTICK. A console that lets you input physical position data with above Model D + 7 A/D card. For games, process control, etc. Contains speaker for sound effects. Kit (Model JS-1-K): $65. Assembled (Model JS-1-W): $95.

PROFESSIONAL QUALITY

You get first-class quality with Cromemco.

Here are actual quotes from articles by independent experts: "The Cromemco boards are absolutely beautiful" . . . "The BYTESAVER is tremendous" . . . "Construction of Cromemco I/O and joystick are outstanding" . . . "Cromemco peripherals ran with no trouble whatsoever."

Everyone agrees. Cromemco is tops.

STORES/MAIL

So count on Cromemco. Look into these Cromemco products at your store. Or order by mail from the factory.

We wish you pleasure and success with your computer.

Cromemco
incorporated
Specialists in computers and peripherals
2432 CHARLESTON RD., MOUNTAIN VIEW, CA 94043 • (415) 964-7400

Circle 353 on inquiry card.

In the Queue

BYTE OCTOBER 1977

Volume 2 Number 10

Foreground

Background

Nucleus

BYTE is published monthly by BYTE Publications Inc, 70 Main St, Peterborough NH 03458. Address all mail except subscriptions to above address; phone (603) 924-7217. Address all editorial correspondence to the editor at the above address. Unacceptable manuscripts will be returned if accompanied by sufficient first class postage. Not responsible for lost manuscripts or photos. Opinions expressed by the authors are not necessarily those of BYTE. Address all subscriptions, change of address, Form 3579, and fulfillment complaints to BYTE Subscriptions, PO Box 361, Arlington MA 02174; phone (617) 646-4329. Second class postage paid at Peterborough NH 03458 and at additional mailing offices—USPS Publication No. 102410. Subscriptions are $12 for one year, $22 for two years, and $32 for three years in the USA and its possessions. Add $5.50 per year for subscriptions to Canada and Mexico. $25 for a one year subscription by surface mail worldwide. Air delivery to selected areas at additional rates available upon request. $25 for a one year subscription by air delivery to Europe, or DM 60,— for a one year subscription through our European distributor in Darmstadt, West Germany. Single copy price is $2.00 in the USA and its possessions, $2.40 in Canada and Mexico, and $3.50 in Europe or DM 8,— through our European distributor in Darmstadt, West Germany, and $4.00 elsewhere. Foreign subcriptions and sales should be remitted in United States funds. Printed in United States of America.

PUBLISHERS
Virginia Peschke
Manfred Peschke
EDITOR IN CHIEF
Carl T Helmers Jr
PRODUCTION MANAGER
Judith Havey
CIRCULATION MANAGER
Gregory Spitzfaden
ASSISTANT PUBLISHER
Debra Boudrieau
EDITOR
Christopher P Morgan
CO-OP EDITOR
Scott Morrow
PRODUCTION EDITORS
Karen Gregory
Nancy Salmon
ADVERTISING
Debra Boudrieau, Director
Noreen Bardsley
Virginia Peschke
EDITORIAL ASSISTANT
Ingrid Nyland
PRODUCTION ASSISTANT
Cheryl Hurd
CONVENTION SUPERVISOR
Elizabeth Alpaugh
DEALER SALES
Ginnie F Boudrieau
CLUBS
David Wozmak
TRAFFIC MANAGER
Edmond C Kelly Jr
ART
Wai Chiu Li
Ellen Shamonsky
SPECIAL PRODUCTS
Floyd W Rehling
Medellin Stephens
David Wozmak
EXECUTIVE SECRETARY
Patricia Clark
RECEPTIONIST
Jacqueline Earnshaw
DRAFTING
Douglas Glen
Stephen Kruse
Lynn Malo
Bill Morello
Dorothy Shamonsky
TYPOGRAPHY
Custom Marketing Resources Inc
Goodway Graphics
PHOTOGRAPHY
Ed Crabtree
PRINTING
Rumford Press
EDITORIAL CONSULTANT
Daniel Fylstra
ASSOCIATES
Walter Banks
Steve Ciarcia
David Fylstra
Portia Isaacson
FOREIGN DISTRIBUTOR
Pan Atlantic Computer Systems GmbH
Frankfurter Str 78
D61 Darmstadt West Germany
(0 61 51) 29 29 23

ADVERTISING SALES REPRESENTATIVES
National:
Parker & Short
1 Lakeside Office Park
Wakefield MA 01880
(617) 246-2293
Southern California:
Buckley/Boris Associates Inc
912 South Barrington
Suite 202
Los Angeles CA 90049
(213) 826-4621
Northern California:
Jules E Thompson Inc
Hearst Building
Suite 1111
San Francisco CA 94103
(415) 362-8547

In This BYTE

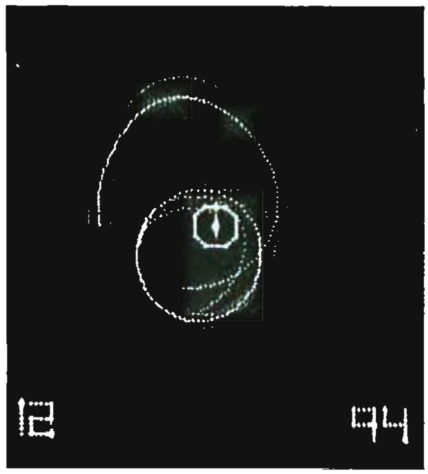
One whole subset of the personal computing world is provided by the users and manufacturers of programmable calculators. All the problems of creating applications software which users must solve on bigger machines are present, and often intensified by lack of scale, in these smallest of personal computers. William B Jenkins gives some useful information on the general process of creating an application program, and the specific problems of doing it on an SR-52 programmable calculator, in his article entitled **How to Write an Application Program.**

One of the conveniences of the 6800, 6502 and similar microprocessors is a relative branch method which allows one to construct position independent code which can be relocated by simply moving the programs involved. But these forms are typically limited to a 1 byte displacement, a limitation which Robert Borrmann shows how to overcome in the 6800 case by using appropriate stack manipulations and "long branch" subroutines. Read his article **Relocatability and the Long Branch** in this issue.

Looking for a different type of board game to play on your computer? How about the current game fad Othello (known as Reversi in England)? In **Othello, a New Ancient Game** Richard O Duda provides a short article with details for this game of skill and tactics.

This month, Mike Wimble concludes his 3 part series about an APL interpreter with **An APL Interpreter for Microcomputers, Part 3: Mathematical Processing.** With this segment, the functional design of interpreter is completed. Watch future issues for results of the Great APL Interpreter Contest inspired by Mike's article.

At first glance a simulator designed to run on the computer it is simulating may not seem very useful. Kin-man Chung feels differently for he wrote one. His article, **An 8080 Simulator,** describes one such program and gives ideas on how it can be put to good use.



page 86

For those who tire of the many versions of the Star Trek game, there are many much more interesting and interactive graphics games to consider. In his article, **How to Implement Space War,** Dave Kruglinski provides readers with a version of the classic graphics game, Space War, which was originated in the early 1960s by students at MIT, and has taken an amazingly long time to be documented in versions for personal computers. Dave's 8080 version is complete with orbiting space ships, spiraling torpedoes and dynamic effects implemented with limited resolution point plotting graphic display.

Is your computer cold? Add some vitamin C for a new high in resistance to frustration and rude language. Turn to J Gregory Madden's **C: A Language for Microprocessors?**, a description of an excellent structured programming language which could be adapted to microprocessor use from its origins on large PDP-11s with the Unix operating system.

Do you use cassettes as your principal mass storage medium? Then you will benefit from Wayne D Smith's discussion of **Fundamentals of Sequential File Processing** when it comes time to write software using such media.

Want to get involved in pitch generation for computer music synthesis? Thomas Schneider explains several approaches you might consider in his article, **Simple Approaches to Computer Music Synthesis.**

Using flowcharts to gather the logic for a program does not mesh with the current trend of structured programming. One technique that is directed towards the structured program approach is the use of Warnier-Orr diagrams. Use of these diagrams, as described by David Higgins in his article **Structured Program Design,** will result in accurate, well structured programs that will work correctly the first time they are executed.

The home computer has many uses besides number crunching and game playing. One of these uses, discussed by David Holladay in **Computer Information Arrangement,** is an information retrieval system. This type of system could be used to make your own dictionary type reference, help keep track of your files with cross reference, or simply make a personal version of the *Schwann Catalog* for your record collection.

Sensible automobile owners have long had the habit of recording mileage and gasoline filling figures at each visit to the service station. In this issue John P Bauernschub explains how to **Analyze Your Car's Gas Economy with Your Computer** in a short article presenting a complete BASIC program for this application.

Are you looking for a stimulating thought game to play with your computer? The game of **Mastermind** as described by W Lloyd Milligan in his article of that name will force you to think in a very logical manner if you want to have a chance at winning.

Announcing the West Coast's largest Personal Computing Show. April 28, 29, and 30, 1978 at California's brand new Long Beach Convention Center. This is a selling show with 180 booths (each draped, carpeted and with 500 watts of electricity). Three full days of conference sessions. There will be home brew exhibits, exhibitors lounge, inquiry badge system, computerized registration, a newsroom, and a full blown advertising and promotional campaign to bring you thousands of qualified buyers.

PERCOMP '78®

I wouldn't miss it for anything.

Send me the show kit:

Name:________________ Title:________________

Company:________________________________

Address:________________ State:________________

Zip Code:________________ Phone:________________

Call: (714) 973-0880 Or mail this coupon: PERCOMP 78, 1833 E. 17 St., Suite 108, Santa Ana, CA 92701.
Sign early, corner booths are limited.

Circle 354 on inquiry card.

Editorial

The Colorful Future of Personal Computing

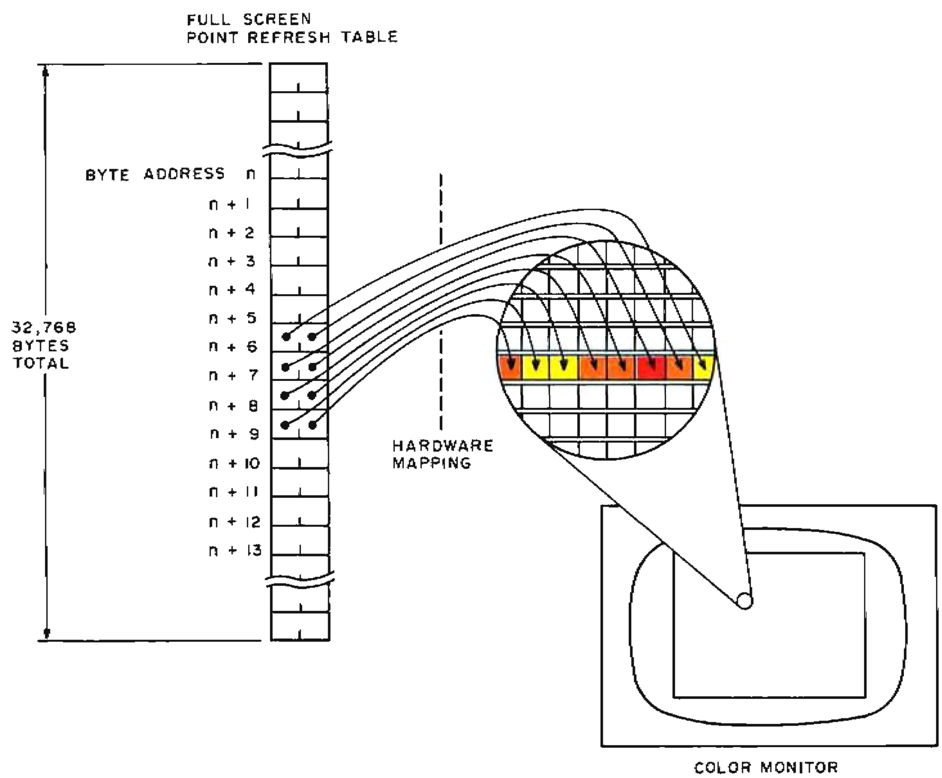


Figure 1: The color display technique which is conceptually simplest and most versatile is the directly refreshed brute force technique of assigning "n" bits to each picture element. For a personal computing context, where conventional TV monitors are used, and byte addressable memory is involved, use of four bits for each point gives 16 color levels per picture element. For a full 256 by 256 element matrix of color, a memory requirement of 32,768 bytes must be satisfied.

(or What the World Needs Is a Good Mass Produced High Resolution Color Display . . .)

By Carl Helmers

This commentary on the possibilities of color imagery and display was inspired by a fantastic image processing system which was surely witnessed by many of the 36,000 people who thronged to the National Computer Conference in Dallas TX this past June. The system in question is produced by a company called Comtal, located in Pasadena CA. It is referenced variously as the Model 8000-S and "Vision One" in the literature I picked up at the site of the demonstration in the main exhibit area of the conference.

This Vision One system is not exactly a personal computing product. Its price tag in the $70,000 range makes it a candidate for laboratory or institutional use, but hardly a peripheral for the individual of ordinary means. Its characteristics include a built in LSI-11 computer with extensive software, direct refresh raster graphics hardware with 512 by 512 8 bit picture elements (ie: 262,144 bytes in its serial CCD refresh memory), and a hard surface disk drive. One of the prime practical applications of this system is its use by the Jet Propulsion Laboratory of Pasadena as the analysis and enhancement processor for the color photos returned from the Viking landers on the planet Mars. But the artistic and personal use attributes of such a color display are immense, as was demonstrated by the

Continued on page 42

The Computer for the Professional

Whether you are a manager, scientist, educator, lawyer, accountant or medical professional, the System 8813 will make you more productive in your profession. It can keep track of your receivables, project future sales, evaluate investment opportunities, or collect data in the laboratory.

Use the System 8813 to develop reports, analyze and store lists and schedules, or to teach others about computers. It is easily used by novices and experts alike.

Reliable hardware and sophisticated software make this system a useful tool. Several software packages are included with the machine: an advanced disk operating system supporting a powerful BASIC language interpreter, easy to use text editor, assembler and other system utilities. Prices for complete systems start at $3250.

See it at your local computer store or contact us at 460 Ward Dr., Santa Barbara, CA 93111, (805) 967-0468.

PolyMorphic Systems

One Sol-20 equals three computers.

To do real work with any computer, big or small, it takes a complete system. That's one of the nice things about the Sol-20. It was built from the ground-up as the heart of three fixed price computer systems with all the peripheral gear and software included to get you up and on the air.

Sol System I costs just $1649 in kit form or $2129 fully burned in and tested. Here's what you get: a Sol-20 with the SOLOS personality module for stand alone computer power, an 8192 word memory, a 12" TV/video monitor, a cassette recorder with BASIC software tape and all necessary cables.

Sol System II has the same equipment plus a larger

capacity 16,384 word memory. It sells for $1883 in kit form; $2283 fully assembled.

For even more demanding tasks, Sol System III features Sol-20/SOLOS, a 32,768 word memory, the video monitor, Helios II Disk Memory System and DISK BASIC Diskette. Price, $4750 in kit form, $5450 fully assembled and tested.

And remember, though we call these small or personal computer systems, they have more power per dollar than anything ever offered. They provide performance comparable with mini-computer systems priced thousands of dollars more.

The functional beauty of Sol Computer Systems is more than skin deep. A look inside reveals a simple elegance of design and sturdy construction.

The Small Computer Catalog for the rest of the real computer system story.

Visit your local computer store for a copy of our fully illustrated 22 page catalog. Or you may write or call us if more convenient. Please address Processor Technology, Box B, 6200 Hollis Street, Emeryville, CA 94608. (415) 652-8080.

Processor Technology Corporation

See Sol here...

ALABAMA

ICP, Computerland
1550 Montgomery Hwy.
Birmingham, AL 35226
(205) 979-0707

ARIZONA

Byte Shop Tempe
813 N. Scottsdale Rd.
Tempe, AZ 85281
(602) 894-1129

Byte Shop Phoenix
12654 N. 28th Dr.
Phoenix, AZ 85029
(602) 942-7300

Byte Shop Tucson
2612 E. Broadway
Tucson, AZ 85716
(602) 327-4579

CALIFORNIA

The Byte Shop
1514 University Ave.
Berkeley, CA 94703
(415) 845-6366

Computer Center
1913 Harbor Blvd.
Costa Mesa, CA 92627
(714) 646-0221

DCI Computer Systems
4670 N. El Capitan
Fresno, CA 93711
(209) 266-9566

Bits 'N Bytes
679 S. State College Blvd.
Fullerton, CA 92631
(714) 879-8386

The Byte Shop
16508 Hawthorne Blvd.
Lawndale, CA 90260
(213) 371-2421

Opamp/Computer
1033 N. Sycamore Ave.
Los Angeles, CA 90038
(213) 934-3566

The Computer Mart
624 West Katella #10
Orange, CA 92667
(714) 633-1222

Byte Shop
496 South Lake Ave.
Pasadena, CA 91101
(213) 684-3311

Micro-Computer
Application Systems
2322 Capitol Avenue
Sacramento, CA 95816
(916) 443-4944

The Computer Store
of San Francisco
1093 Mission Street
San Francisco, CA 94103
(415) 431-0640

Byte Shop
321 Pacific Ave.
San Francisco, CA 94111
(415) 421-8686

The Byte Shop
2626 Union Avenue
San Jose, CA 95124
(408) 377-4685

The Computer Room
124H Blossom Hill Rd.
San Jose, CA 95123
(408) 226-8383

The Byte Shop
509 Francisco Blvd.
San Rafael, CA 94901
(415) 457-9311

The Byte Shop
3400 El Camino Real
Santa Clara, CA 95051
(408) 249-4221

Recreational Computer
Centers
1324 South Mary Ave.
Sunnyvale, CA 94087
(408) 735-7480

Computer Components
5848 Sepulveda Blvd.
Van Nuys, CA 91411
(213) 786-7411

The Byte Shop
2989 North Main St.
Walnut Creek, CA 94596
(415) 933-6252

Byte Shop
14300 Beach Blvd.
Westminster, CA 92683
(714) 894-9131

COLORADO

Byte Shop
2040 30th St.
Boulder, CO 80301
(303) 449-6233

Byte Shop
3464 S. Acoma St.
Englewood, CO 80110
(303) 761-6232

FLORIDA

Byte Shop of Miami
7825 Bird Road
Miami, FL 33155
(305) 264-2983

Microcomputer
Systems Inc.
144 So. Dale Mabry Hwy.
Tampa, FL 33609
(813) 879-4301

GEORGIA

Atlanta Computer Mart
5091-B Buford Hwy.
Atlanta, GA 30340
(404) 455-0647

ILLINOIS

Champaign Computer
Company
318 N. Neil Street
Champaign, IL 61820
(217) 359-5883

itty bitty machine co.
1316 Chicago Ave.
Evanston, IL 60201
(312) 328-6800

itty bitty machine co.
42 West Roosevelt
Lombard, IL 60148
(312) 620-5808

INDIANA

The Data Domain
406 So. College Ave.
Bloomington, IN 47401
(812) 334-3607

The Byte Shop
5947 East 82nd St.
Indianapolis, IN 46250
(317) 842-2983

Computers Unlimited
7724 East 89th Street
Indianapolis, IN 46256
(317) 849-6505

The Data Domain
7027 N. Michigan Rd.
Indianapolis, IN 46268
(317) 251-3139

IOWA

The Computer Store
of Davenport
616 West 35th Street
Davenport, IA 52806
(319) 386-3334

KENTUCKY

The Data Domain
3028 Hunsinger Lane
Louisville, KY 40220
(502) 456-5242

MICHIGAN

The Computer Store
of Ann Arbor
310 East Washington
Ann Arbor, MI 48104
(313) 995-7616

Computer Mart
of Royal Oak
1800 W. 14 Mile Rd.
Royal Oak, MI 48073
(313) 576-0900

General Computer Store
2011 Livernois
Troy, MI 48084
(313) 362-0022

MINNESOTA

Computer Depot, Inc.
3515 W. 70th St.
Minneapolis, MN 55435
(612) 927-5601

NEW JERSEY

Hoboken Computer Works
No. 20 Hudson Place
Hoboken, NJ 07030
(201) 420-1644

The Computer Mart
of New Jersey
501 Route 27
Iselin, NJ 08830
(201) 283-0600

NEW YORK

The Computer Mart
of Long Island
2072 Front Street
East Meadow, L.I. NY 11554
(516) 794-0510

The Computer Shoppe
444 Middle Country Rd.
Middle Island, NY 11953
(516) 732-4446

The Computer Mart
of New York
118 Madison Ave.
New York, NY 10001
(212) 686-7923

The Computer Corner
200 Hamilton Ave.
White Plains, NY 10601
(914) 949-3282

OHIO

Computer Mart of Dayton
2665 S. Dixie Ave.
Dayton, OH 45409
(513) 296-1248

OREGON

Byte Shop Computer Store
3482 SW Cedar Hills Blvd.
Beaverton, OR 97005
(503) 644-2686

The Real Oregon
Computer Co.
205 West 10th Ave.
Eugene, OR 97401
(503) 484-1040

Byte Shop Computer Store
2033 SW 4th Ave.
Portland, OR 97201
(503) 223-3496

PENNSYLVANIA

Byte Shop of
Delaware Valley
1045 Lancaster Pike
Bryn Mawr, PA 19010
(215) 525-7712

RHODE ISLAND

Computer Power, Inc.
M24 Airport Mall
1800 Post Rd.
Warwick, RI 02886
(401) 738-4477

TEXAS

Computer World
926 N. Collins
Arlington, TX 76011
(817) 469-1502

Byte Shop
3211 Fondren
Houston, TX 77063
(713) 977-0664

Computertex
2300 Richmond Ave.
Houston, TX 77006
(713) 526-3456

Interactive Computers
7646½ Dashwood Rd.
Houston, TX 77036
(713) 772-5257

Neighborhood Computer
Store
#20 Terrace Shopping Center
4902 - 34th Street
Lubbock, TX 79410
(806) 743-2787

The Micro Store
634 So. Central
Expressway
Richardson, TX 75080
(214) 231-1096

VIRGINIA

The Computer Systems
Store
1984 Chain Bridge Rd.
McLean, VA 22101
(703) 821-8333

Media Reactions Inc.
11303 South Shore Dr.
Reston, VA 22090
(703) 471-9330

The Home Computer Center
2927 Virginia Beach Blvd.
Virginia Beach, VA 23452
(804) 340-1977

WASHINGTON

Byte Shop Computer Store
14701 N.E. 20th Ave.
Bellevue, WA 98007
(206) 746-0651

The Retail Computer Store
410 N.E. 72nd
Seattle, WA 98115
(206) 524-4101

WISCONSIN

Madison Computer Store
1910 Monroe St.
Madison, WI 53711
(608) 255-5552

The Milwaukee
Computer Store
6916 W. North Ave.
Milwaukee, WI 53213
(414) 259-9140

CANADA

Trintronics
160 Elgin St.
Place Bell Canada
Ottawa, Ontario K2P 2C4
(613) 236-7767

First Canadian
Computer Store Ltd.
44 Eglinton Ave. West
Toronto, Ontario M4R 1A1
(416) 482-8080

The Computer Place
186 Queen St. West
Toronto, Ontario M5V 1Z1
(416) 598-0262

Pacific Computer Store
4509-11 Rupert St.
Vancouver, B.C. V5R 2J4
(604) 438-3282

Processor

Circle 355 on inquiry card.

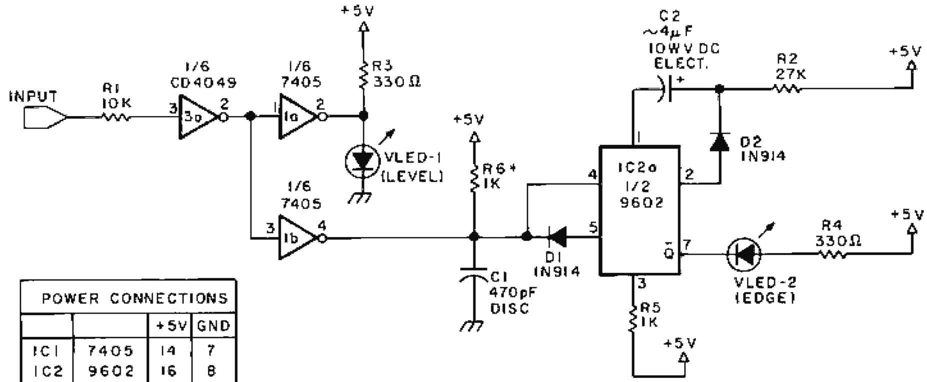
Letters

A CMOS LOGIC PROBE

This is in response to Tom Kryst's letter in the July 1977 BYTE, page 148, expressing a need for a low cost CMOS logic probe. I was faced with a similar requirement while working on a software development system for the RCA 1802 CMOS microprocessor. Not being a person who believes in reinventing the wheel, I borrowed Kurt Christner's TTL design from the January 1977 BYTE, page 82, adding a CMOS inverting buffer at the front end, as shown below. Please note that a CD4069 inverter may *not* be substituted for the more expensive CD4049, as the current sinking capability of the latter device is necessary.

I hope this modification may prove useful to some of your readers.

Frank A Weissig
343 NW 8th St, Apt 3
Corvallis OR 97330



POWER CONNECTIONS			
		+5V	GND
IC1	7405	14	7
IC2	9602	16	8
IC3	CD4049	1	8

*I added this resistor to Mr Christner's design because there's something about an OC gate without a pull up that makes me nervous.

KUDOS FOR THE LD-14

I'd like to make a few comments about one of your advertisers, namely Logic Design Inc of Laramie WY. I purchased their LD 14 Tutorial Training Computer which was written up in January 1977 BYTE. Although I have not finished wire wrapping the unit, I am confident that I am getting my $1200's worth. One thing that has impressed me about the company is the quick response to a problem I was having one evening because of a gross error on my part. Some of the wires melted together as the result of tying VCC to ground. I became a little nervous that maybe I damaged something, so I wrote to complain about their failure to give any guarantees that their product is idiot proof.

Within three days I received a very courteous and apologetic phone call from the company with an open invitation to call them collect in the event I experienced further difficulty. With this kind of response it would seem likely that they intend to back up their commitment to quality support as stated in their advertisement.

To a second party, my finished product may not appear to go beyond the Intercept Jr offered by Intersil, based on their IM6100 chip, which goes for around $450 with 1 K of programmable memory. This compared to the $1200 I have spent may be called to question when considering only the physical result. The difference, as I see it, is that I will have acquired an understanding of the PDP-8 computer that I would never have been able to by simply programming a prewired version. Beyond this, when I'm finished wire wrapping the PDP-8, I can unplug it from the logic designer front panel, give it a 5 V supply, and use the designer for my own creations.

For anyone interested in learning some of the state of the art do's and dont's of logic design, I will give this product a strong recommendation. Also, I would be interested in making contact with others who are building, or have completed, the LD 14 computer.

Phil Winninghoff
815 Plata Rd
Arroyo Grande CA 93420

ASIMOV'S PERSONAL COMPUTER ANTICIPATION, CIRCA 1950

In his article, "Why aren't there any Altairs on Arcturus II?" (April 1977 BYTE), H Melton states that, "I can remember only one old story that used pocket calculators. . . ." As an avid SF fan, and as an even more avid calculator (HP) freak, I must beg to differ with Mr Melton. In the *Foundation Trilogy*, by Isaac Asimov, one finds the following passage:

> *Seldon removed his calculator pad from the pouch at his belt. . . .Its grey, glossy finish was slightly worn by use. Seldon's nimble fingers. . .played over the hard plastic that rimmed it. Red symbols glowed out from the grey.*

How many of us have not performed the act just described? And the *Foundation Trilogy* was, I believe, written in the early 1950s. I must, however, agree with the general tone of the article: most of us (including SF writers) are just not able to predict the rate of technological progress.

W Gray Mansfield
5042 Guava Av
La Mesa CA 92041

PRINTING BARS ON SELECTRICS?

I see you're printing optical code. Is it possible to get an IBM ball (or keys) which prints letters and code simultaneously? If yes, where are they available?

Rob Loring
Twin Oaks Commmunity
Rt 4, Box 169
Louisa VA 23093

Yes, but it's expensive. Custom tooling is available from several sources for IBM style balls; thus, while one could design a bar code font, it would not be a project undertaken lightly.

LIGHTING THROUGH THE PAPER?

The idea of distributing software through PAPERBYTES is great. Why don't you publish PAPERBYTES on one side of a page only, leaving the other side blank? Light could then be shone from underneath the paper to illuminate the coded material.

This technique would greatly simplify the optical system needed for the bar code reader. It would eliminate the light source and lens in the reader (wand) and reduce specular reflection problems that cause errors.

I plan on trying this approach soon. Do you or your readers have any suggestions?

Andrew A Modla
108 Clemens Ct
Lansdale PA 19446

The main problem with the approach of backlighting is that it leaves one whole side of the paper blank, something which is less than optimal when it comes to publishing information in books and magazines on expensive paper.

PRINTING QUOTES ON AN HP9830 IN BASIC

As a user of Hewlett-Packard's 9830 BASIC machine, I have discovered one of its (very few) shining features. The BASIC interpreter has a statement pair WRITE and FORMAT, which FORTRAN users will recognize. This is a rather little known fact, as HP buries it in an obscure portion of their manual.

In order to get quotes out of it, something like this would be written:

```
10 FORMAT 3B
20 WRITE (15, 10) 34, "HELLO", 34
```

where 15 is the device code of the main printer. The output looks like this:

```
"HELLO"
```

What goes on is that B format will convert the decimal number given it (constant, variable, or whatever) and output it directly as ASCII; the quote mark is 34 in ASCII. This can also be used to print nonkeyboard characters; square brackets, reverse slashes, and so on.

The other way to do this requires the Advanced IO ROM, as HP sells their software as plug in black (or gray) boxes. The OUTPUT statement allows you to use a string as an output device. Example:

```
10 FORMAT B
20 OUTPUT (A$, 10) 34,
30 PRINT A$
```

with the result:

```
"
```

So, all you users of the HP9830, now you can print quotes and other goodies. This almost makes up for the crudity of HP's strings.

John Woods WB7EEL
6541 126th Av SE
Bellevue WA 98006

AS SALES EXPAND, SAFETY ASPECTS MUST BE IMPROVED

While experimenters look for the fastest, most powerful, least expensive microcomputer, they seldom look for the safest. From looking at ads for hobby computers, I learn that only OSI advertises their power supply as being listed by Underwriters Laboratories Inc. Some small computers have 117 VAC conductors that are exposed when the lid is off (as I learned one day by accident – no injury, just minor equipment damage); the same machines don't have fuses in the unregulated low voltage high current lines. Just wait until you've welded a probe to a power trace to find out how much you need that item. It's an industrial standard to avoid exposed high voltages whenever possible, and I see no reason to have an exposed, noisy

Continued on page 32

Rated G

The Best Game in Town.

Welcome to ComputerLand. An incredible adventure into the world of personal computers. A one-of-a-kind shopping experience.

Each ComputerLand store presents everything you ever wanted to know about computers. And then some.

Take our Game Room, for starters. You'll find excitement for the whole family in our endless variety of challenging computer games. You can battle the Klingons in an out-of-this-world game of Star Trek. Create an electronic work of art with a computer controlled TV. Test your skill in a game of computerized hangman.

You can even plot your biorhythm.

But we're more than just fun and games. Each ComputerLand store offers a knowledgeable and personable staff of professionals to serve you.

Plus the greatest available selection of micro components. Whether it's a data processing system for your business or a computer controlled sprinkler system for your home, you'll find whatever you need at ComputerLand.

Read on.

Genuine Service.

We want to supply you with the one system that's right. Rather than a complete system that isn't. Or a limited system that is.

That's why, at ComputerLand, you deal with real professionals who are also real people. People who speak your language . . . in addition to BASIC, COBOL or FORTRAN.

People, in short, who can offer both the novice and the old hand the same expert guidance in selecting the optimum system he or she needs.

Yet, assisting in the purchase is only the beginning of ComputerLand's service. If the kit you bought requires a little more do-it-yourself than you yourself can do, we provide assembly assistance.

If that complex program proves to be just that, we provide programming assistance.

And if your system breaks down, our in-store service department will get you back up and running.

Right now!

Great Selection.

Your first stop at ComputerLand may well be your last stop.

ComputerLand offers the finest quality and largest selection of all the major brand names. Like Apple Computer, Cromemco, DEC, Diablo, Hazeltine, ICOM, IMSAI, Lear Siegler, National Semiconductor, North Star, Texas Instruments, Vector Graphics and more.

Plus a complete inventory of tools, books and accessories.

What's more, at ComputerLand, we deal in product. Not promises. Our inventory is on our own shelves. Rather than the manufacturer's. So you can take delivery on tomorrow's components today.

Which means, simply put, that at ComputerLand, you get exactly what you want.

Exactly when you want it.

Be Our Guest.

Begin with the grand tour of our exhibit areas. "Test-drive" any of our individual systems.

Then tell us your needs. We'll sit down and talk about the system that's right for you. It's as easy as that at ComputerLand.

The great computer store. RATED G.

Call or write for the address of the ComputerLand store nearest you. Franchise opportunities available.

ComputerLand™

1922 Republic Avenue, San Leandro, CA 94577 (415) 895-9363


Introducing Apple II.™

You've just run out of excuses for not owning a personal computer.

Clear the kitchen table. Bring in the color TV. Plug in your new Apple II* and connect any standard cassette recorder/player. Now you're ready for an evening of discovery in the new world of personal computers.

Only Apple II makes it that easy. It's a

complete, ready to use computer, not a kit. At $1298, it includes video graphics in 15 colors. It includes 8K bytes ROM and 4K bytes RAM—easily expandable to 48K bytes using 16K RAMs (see box). But you don't even need to know a RAM from a ROM to use and enjoy Apple II. For example, it's the first personal computer with a fast version of BASIC permanently stored in ROM. That means you can begin writing your own programs the first evening, even if you've had no previous computer experience.

The familiar typewriter-style keyboard makes it easy to enter your instructions. And your programs can be stored on—and retrieved from—audio cassettes, using the built-in cassette interface, so you can swap with other Apple II users.

You can create dazzling color displays using the unique color graphics commands in Apple BASIC. Write simple programs to display beautiful kaleidoscopic designs. Or invent your own games. Games like PONG—using the game paddles supplied. You can even add the dimension of sound through Apple II's built-in speaker.

But Apple II is more than an advanced, infinitely flexible game machine. Use it to teach your children arithmetic, or spelling for instance. Apple II makes learning fun.

Apple II can also manage household finances, chart the stock market or index recipes, record collections, even control your home environment.

Right now, we're finalizing a peripheral board that will slide into one of the eight available motherboard slots and enable you to compose music electronically. And there will be other peripherals announced soon to allow your Apple II to talk with another Apple II, or to interface to a printer or teletype.

Apple II is designed to grow with you as your skill and experience with computers grows. It is the state of the art in personal computing today, and compatible upgrades and peripherals will keep Apple II in the forefront for years to come.

Write us today for our detailed brochure and order form. Or call us for the name and address of the Apple II dealer nearest you. (408) 996-1010. Apple Computer Inc., 20863 Stevens Creek Boulevard, Bldg. B3-C, Cupertino, California 95014.

Apple II™ is a completely self-contained computer system with BASIC in ROM, color graphics, ASCII keyboard, lightweight, efficient switching power supply and molded case. It is supplied with BASIC in ROM, up to 48K bytes of RAM, and with cassette tape, video and game I/O interfaces built-in. Also included are two game paddles and a demonstration cassette.

SPECIFICATIONS

- **Microprocessor:** 6502 (1 MHz).
- **Video Display:** Memory mapped, 5 modes—all Software-selectable:
  - Text—40 characters/line, 24 lines upper case.
  - Color graphics—40h x 48v, 15 colors
  - High-resolution graphics—280h x 192v; black, white, violet, green (12K RAM minimum required)
  - Both graphics modes can be selected to include 4 lines of text at the bottom of the display area.
  - Completely transparent memory access. All color generation done digitally.
- **Memory:** up to 48K bytes on-board RAM (4K supplied)
  - Uses either 4K or new 16K dynamic memory chips
  - Up to 12K ROM (8K supplied)
- **Software**
  - Fast extended integer BASIC in ROM with color graphics commands
  - Extensive monitor in ROM
- **I/O**
  - 1500 bps cassette interface
  - 8-slot motherboard
  - Apple game I/O connector
  - ASCII keyboard port
  - Speaker
  - Composite video output

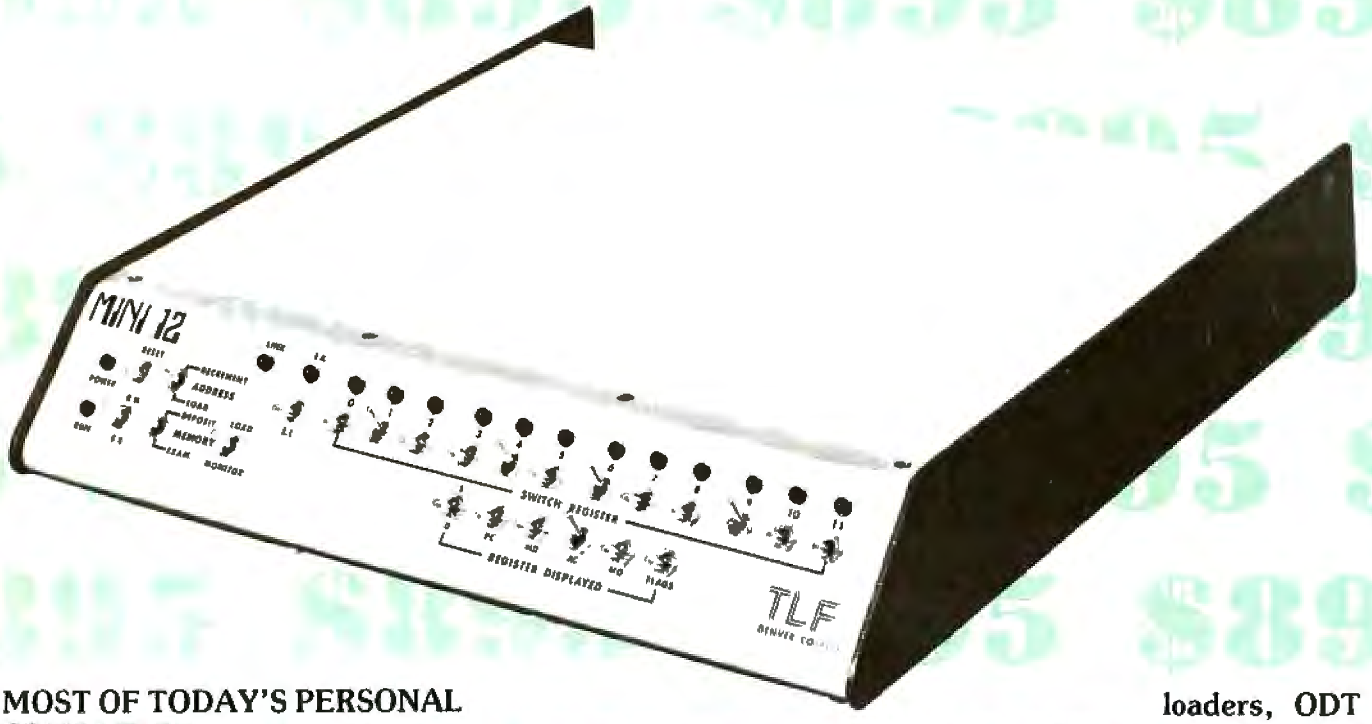
Apple II is also available in board-only form for the do-it-yourself hobbyist. Has all of the features of the Apple II system, but does not include case, keyboard, power supply or game paddles. $598.

PONG is a trademark of Atari Inc.

*Apple II plugs into any standard TV using an inexpensive modulator (not supplied).

apple computer inc.™

Why buy a micro, when you can buy a mini for less!



MOST OF TODAY'S PERSONAL COMPUTERS are microcomputers. Compared to our minicomputer they are slow and expensive.

Our mini "the MINI 12™" has been around for three years. We've been steadily selling it to industrial customers who have proven its reliability. Only recently have we paid much attention to the personal computing market. We've discovered that our computer is simply better, and most interestingly, LESS EXPENSIVE.

NONE OF THE MICROCOMPUTERS can offer you the range of software available to the MINI 12. Compatibility with the most popular minicomputer, DEC's PDP-8E, gives you a vast library of programs. You will also be pleasantly surprised when you experience the speed and sophistication of our EXTENDED BASIC compiler, or FORTRAN, ALGOL and FOCAL.

Unlike today's microcomputers, our Mini 12 is fully software supported by an advanced operating system, including...file handling text editor, symbolic assembler, loaders, ODT (octal debugging technique), batch processor, and system utilities.

The MINI 12 comes with 8K of memory and is easily expandable to 32K. Because the MINI 12 processes 12 bit words (instead of 8 bit bytes) you need far less memory. For most applications... using our optional MINI STORE digital cassette system with the BASIC compiler, you will never need add another single byte!

Because we want you to have the pleasure of using your minicomputer right away, our MINI 12 comes fully assembled, debugged and ready to plug in. It includes a full front panel, controller for eight digital tape drives, and built-in serial interface plus other technical niceties like a binary loader and monitor bootstrap in ROM memory.

Frankly, when compared with microcomputers, we're surprised at how good a deal we are. Write... or better yet, give us a call @303-794-1634 and let us help you get it all together!

TLF Post Office Box 2298, Littleton, Colorado 80161, Telephone 303-794-1634

PDP-8E and Focal are Trademarks of Digital Equipment Corporation

Circle 358 on inquiry card.

How to Write an Application Program

William B Jenkins
643 S Rte 83
Elmhurst IL 60126

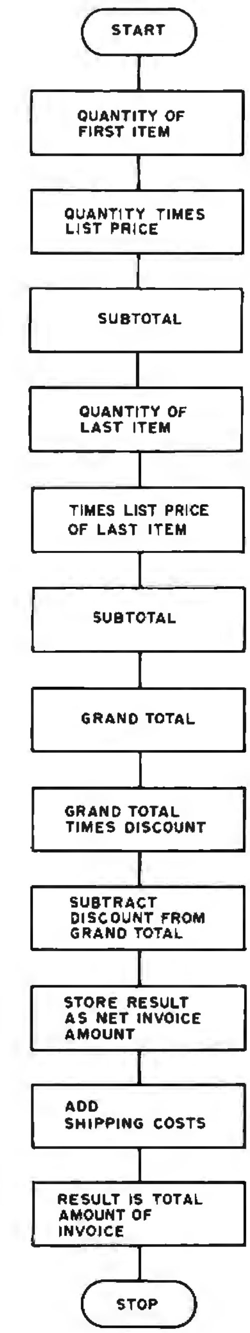


Figure 1: Basic flowchart outlining the procedure for completing an invoice. This is a first step in detailing precisely what the program should do.

The Texas Instruments SR-52 is a very impressive machine for its size. Several different libraries of programs are provided that include engineering, finance and games applications. The game programs do, of course, provide the easiest way to impress your friends with the power of the SR-52. This is no different from using game programs with any computer system. For personal use, though, games can quickly lose their glamour. When this happens, it is time to learn how to program.

As the owner of a small business, there are at least two tasks which consume a fair amount of my time. One is payroll and the other is invoice preparation. Of the two, invoice preparation is the easiest program to start with, although both programs offer good opportunities to learn many of the intricacies of programming. These techniques also provide a practical way of learning general programming methods. The result is the development of skills that include analyzing the problem and designing a logical way (flowchart) of identifying the best method for solving the problem. Next comes the conversion of the flowchart to the program language that your machine uses. To do this, let's start with the problem of developing a program to help prepare invoices.

Invoice Program

The first step is to think out what you would do to manually prepare an invoice. There are several ways of doing invoices. We are manufacturers of retail products and sell various quantities of many different items. Our invoices show the suggested list price for each item. After the items are totalled, a discount is calculated and subtracted from the total. The result is the net amount that the customer pays. There are other costs, such as shipping charges, which are also added. Figure 1 shows a flowchart for this first step.

Making a flowchart for this first step may seem to be unnecessary. What it does is give you a graphic description of the steps involved. From this, some of the next steps become more obvious. For example, in an invoice, we multiply one item by another and save the total. This basic step is repeated for as many items as required. This type of repetition is one of the things a computer does best. In the same way, it is the very thing that bores a human fastest.

The end steps require adding the individual totals and getting a grand total. The remaining steps are straightforward arithmetic.

The initial conclusion (one which we sort of knew from the start) is that this problem lends itself to solution by computer.

We will add one more set of calculations so that the final program can handle a second class of merchandise that does not have any discount applied. This addition is similar to the steps shown in figure 1, and we will include it in our next reiteration of the design.

SR-52 Features

What we have done so far has been independent of the system that we will be using. At this point, we have to learn a little about the specific hardware.

The SR-52 allows programs to be input using the calculator keyboard. The individual keys could be considered as a set of

Photo 1: SR-52 programmable calculator.

op codes for a high level language interpreter, with each key performing a set of fairly complex operations. The SR-52 provides several categories of key functions. It has the conventional arithmetic functions: +, −, x, ÷; and a set of memory functions that allow you to manipulate the 20 available memory areas. These functions include STORE, RCL (recall), SUM, PROD (product), and EXC (exchange). Trigonometric functions, Sin, Cos, Tan and other standard functions such as x^2, y^x, and $^1/x$ are also included. An important set of keys are those that allow the SR-52 program to make decisions. These can perform tests on program data in order to determine what the next program step might be. The SR-52 is shown in photo 1.

The SR-52 has a maximum of 224 program steps. To program the machine, you simply take the list of key sequences and push the buttons in that sequence until you are done. This is a little oversimplified, but it really isn't much more than that. If your program has been properly designed, the SR-52 will be able to do all of the finger crunching work when you put in the data. When you are satisfied with the program, you can save it on a magnetic card for future use.

Zeroing In

What we have done up to this point has given us a logical approach to the program solution of our problem, and an overall notion of how the SR-52 can do the job for us. Now we have to get a little closer to the SR-52 to know exactly how to translate our flowchart to key sequences.

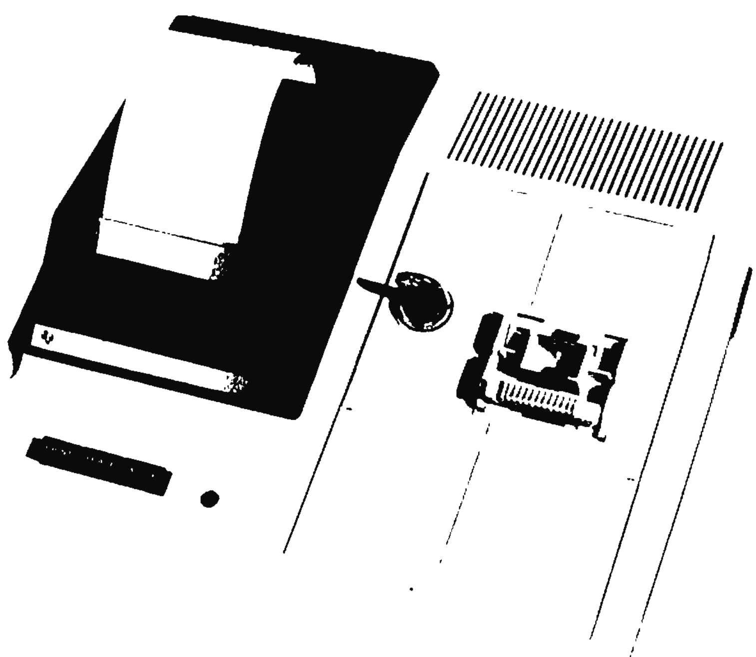
Our program should be easy for a nontechnical person to use. This means that entering the data should use very few steps. The program should do most of the work until the problem (invoice) is complete. To help accomplish this, the program will be written to use the optional SR-52 printer as a recorded output. The printer, shown in photo 2, in addition to providing a hard copy output, actually makes program development easier. By printing desired results as they occur we can avoid using too many of our precious 224 program steps to store and then later recall results. In some cases, programs that would be too long, more than 224 steps, without the printer can be handled on the SR-52. Additionally, since the printer can be operated under program control, some limited data formatting can be accomplished. Last of all, the printer allows the user to verify the accuracy of the data input.

Invoice Program Development

In the invoice problem we are working on, we can see from the initial flowchart that we are going to use several basic SR-52 functions. These are data storing, recalling, multiplying, subtracting, adding and printing.

We will now expand our initial flowchart

Photo 2: Printer attachment for SR-52.



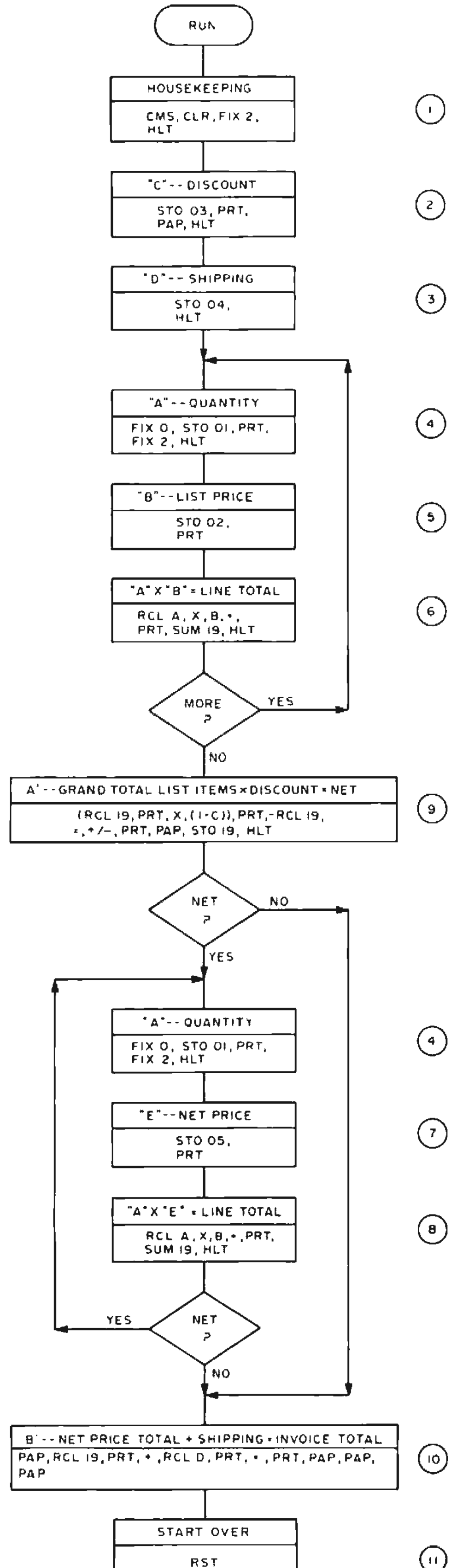


Figure 2: Expanded flowchart for invoice program includes net priced items. The actual programming steps for the SR-52 are also noted within the boxes. The circled numbers indicate the order in which the program was written and refer back to the text.

to include the actual steps we will ask the SR-52 to do for us. In this expanded flowchart we have added the second category of merchandise we want to price. One other design consideration that we want to include is the manner in which we will have the user input the data. The SR-52 has a set of five buttons that are labeled A, B, C, D and E. These same buttons can be set for a second use which adds five more functions labeled A′, B′, C′, D′ and E′. These buttons are called "user defined keys." Depending on the way you write the program, a given user button will do whatever you want. For our use, we will want to use these buttons to define to the program what our data means. Table 1 shows the definition we have given to these keys for this program. Looking at the work the SR-52 does when a user defined key is pushed gives you an idea of the power of the machine.

To get the invoice data into the program, the user keys in the numbers on the keyboard and presses the appropriate user defined key.

Now, we can expand our initial flowchart to what we see in figure 2. The numbers in circles adjacent to various boxes are the order in which we will write the program sequence. We will refer to these numbers in the following discussion of the new flowchart.

1. This is some of the housekeeping that any program should have. It assures you that the initial condition of the calculator or computer is what you expect it to be. Here we want to CLR (clear), CMs (clear data memories), FIX 2 (fix the decimal point of the computations to two places) and HLT (halt). We want the machine to stop and wait for us to input data.
2. In this step we are going to define key C to mean discount. We are going to STO (store) it in data memory location 03, print what we entered, advance the paper one line and HLT again.
3. Key D is defined to be the shipping amount of the order; it is stored in data memory 04. We then halt. We could have printed this value, but I chose not to because it will be used at the end of the program where, if it is the wrong amount, not much harm is done.
4. Key A is the quantity of merchan-

Continued on page 152

Look To The North Star HORIZON Computer.

HORIZON™— a complete, high-performance microprocessor system with integrated floppy disk memory. HORIZON is attractive, professionally engineered, and ideal for business, educational and personal applications.

To begin programming in extended BASIC, merely add a CRT or hard-copy terminal. HORIZON-1 includes a Z80A processor, 16K RAM, minifloppy™ disk and 12-slot S-100 motherboard with serial terminal interface — all standard equipment.

WHAT ABOUT PERFORMANCE?

The Z80A processor operates at 4MHZ — double the power of the 8080. And our 16K RAM board lets the Z80A *execute at full speed.* HORIZON can load or save a 10K byte disk program in less than 2 seconds. Each diskette can store 90K bytes.

AND SOFTWARE, TOO

HORIZON includes the North Star Disk Operating System and full extended BASIC on diskette ready at power-on. Our BASIC, now in widespread use, has everything desired in a BASIC, including sequential and random disk files, formatted output, a powerful line editor, strings, machine language CALL and more.

EXPAND YOUR HORIZON

Also available—Hardware floating point board (FPB); additional 16K memory boards with parity option. Add a second disk drive and you have HORIZON-2. Economical serial and parallel I/O ports may be installed on the motherboard. Many widely available S-100 bus peripheral boards can be added to HORIZON

QUALITY AT THE RIGHT PRICE

HORIZON processor board, RAM, FPB and MICRO DISK SYSTEM can be bought separately for either Z80 or 8080 S-100 bus systems.

HORIZON-1 $1599 kit; $1899 assembled.
HORIZON-2 $1999 kit; $2349 assembled.

16K RAM—$399 kit; $459 assembled; Parity option $39 kit; $59 assembled. FPB $259 kit; $359 assembled. Z80 board $199 kit; $259 assembled. Prices subject to change. HORIZON offered in choice of wood or blue metal cover at no extra charge.

Write for free color catalogue or visit your local computer store.

Circle 359 on inquiry card.

NORTH STAR ★ COMPUTERS
2465 Fourth Street • Berkeley, California 94710 • (415) 549-0858

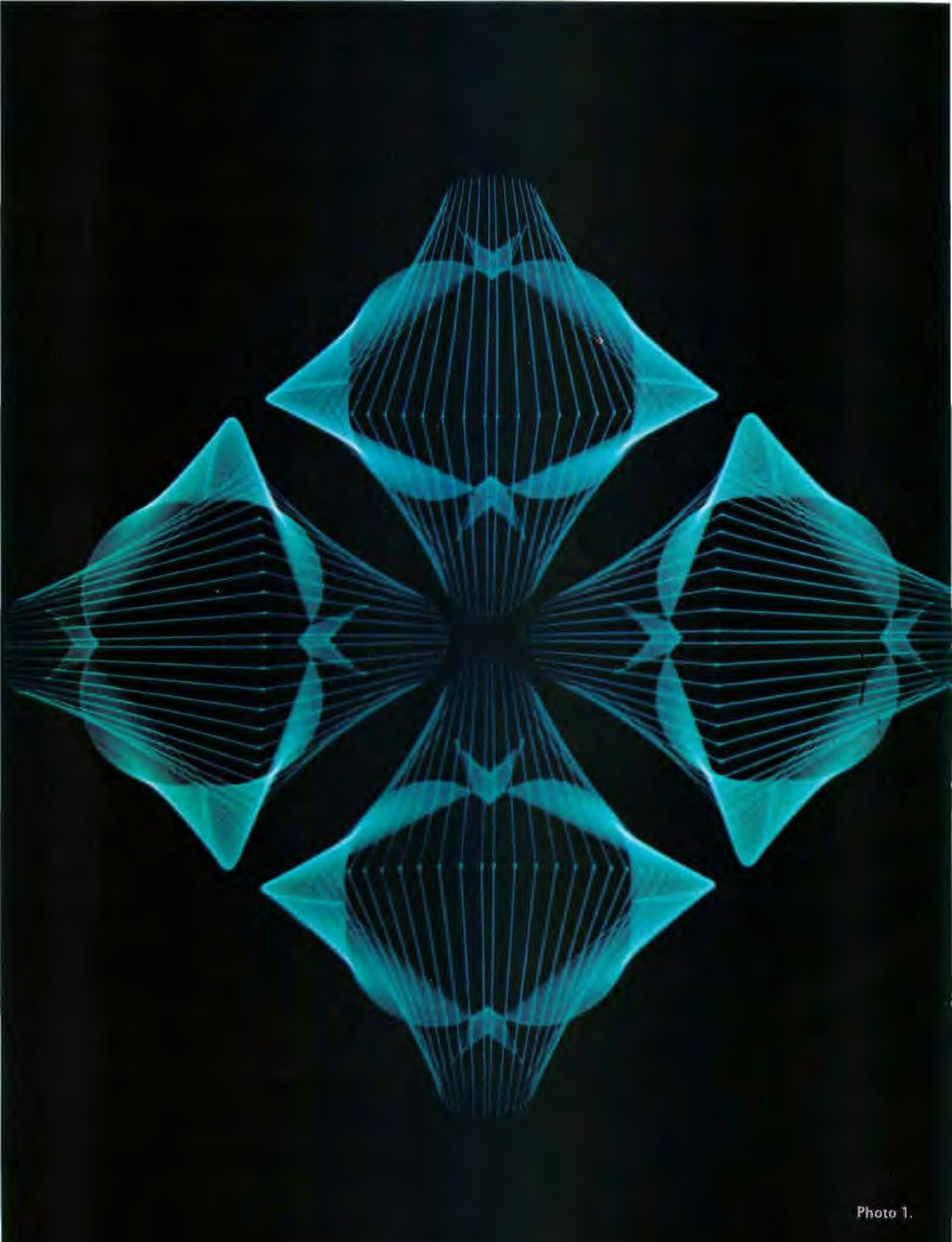


Photo 1.

About the Cover

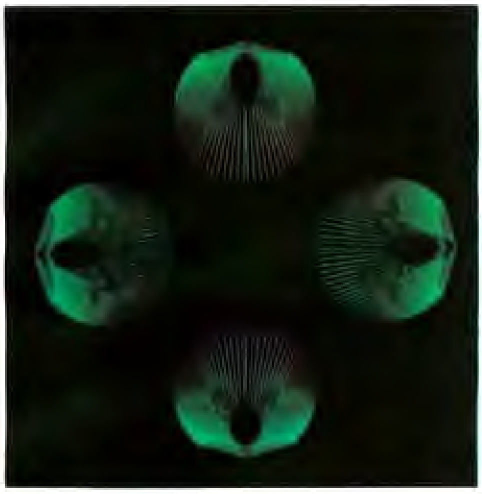


Photo 2.

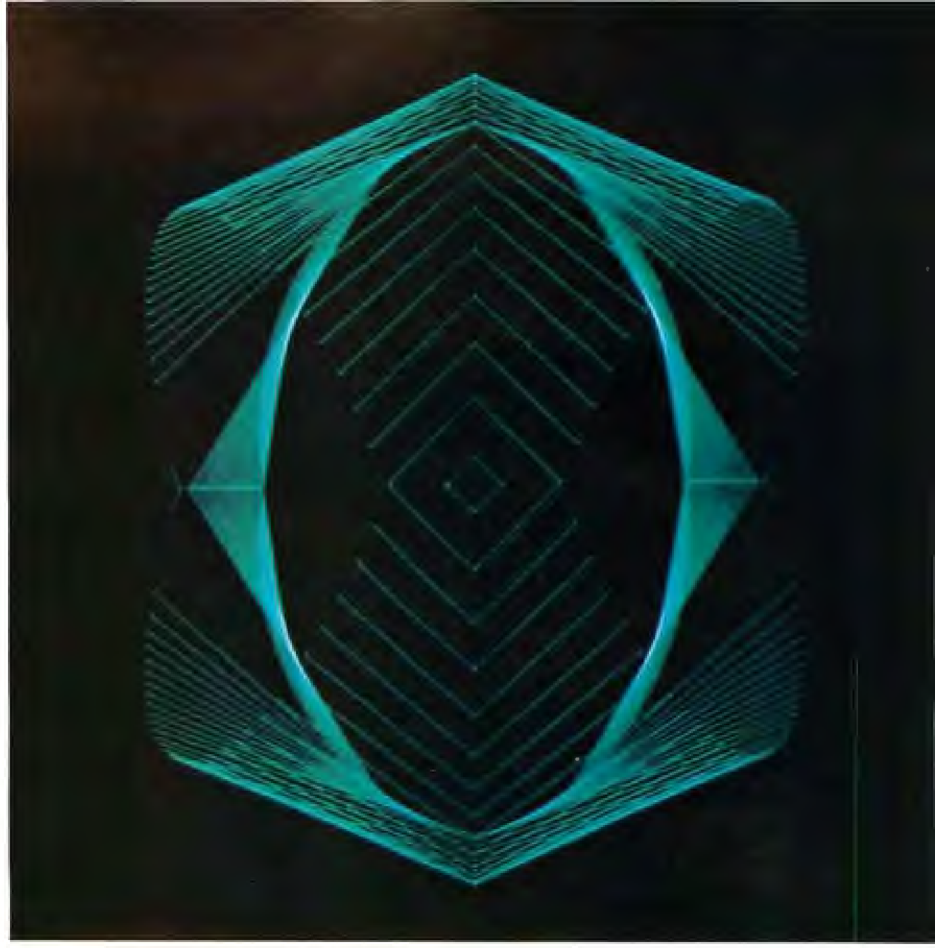


Photo 3.

. . . and Some More of the Same

Photo 4.

The artwork on our cover this month and the samples found on the following pages were created by Thomas A Defanti and Guenther Tetz of the University of Illinois at Chicago Circle. The system used to create these works used a high level "Graphics Symbiosis System" (acronym: GRASS) grafted onto a PDP-11/45 computer with a Vector General 3DR display scope, a data tablet and 30 channels of analog input and output for interactive control of the programmed parameters of the display. Consider it in some sense the visual equivalent of a Moog synthesizer's audio functions. A more complete description is found in a paper entitled "The Digital Component of the Circle Graphics Habitat" which was published on pages 195 to 203 of the proceedings of the 1976 National Computer Conference (a monstrously thick book available from AFIPS, the American Federation of Information Processing Societies, 210 Summit Av, Montvale NJ 07645).

Artists DeFanti and Tetz sent in several slides of their work as part of the art contest we ran last year in the September 1976 issue of BYTE. They report that work is progressing on a microprocessor based version

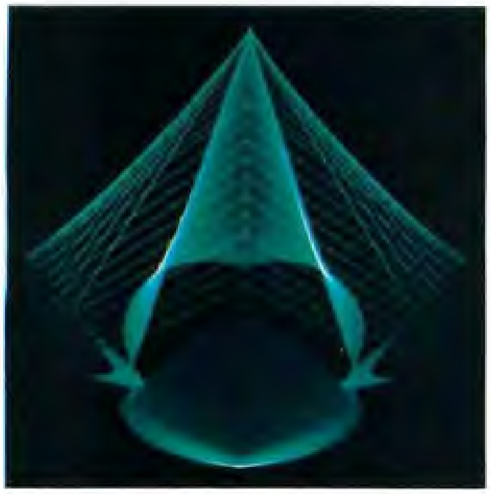


Photo 5.

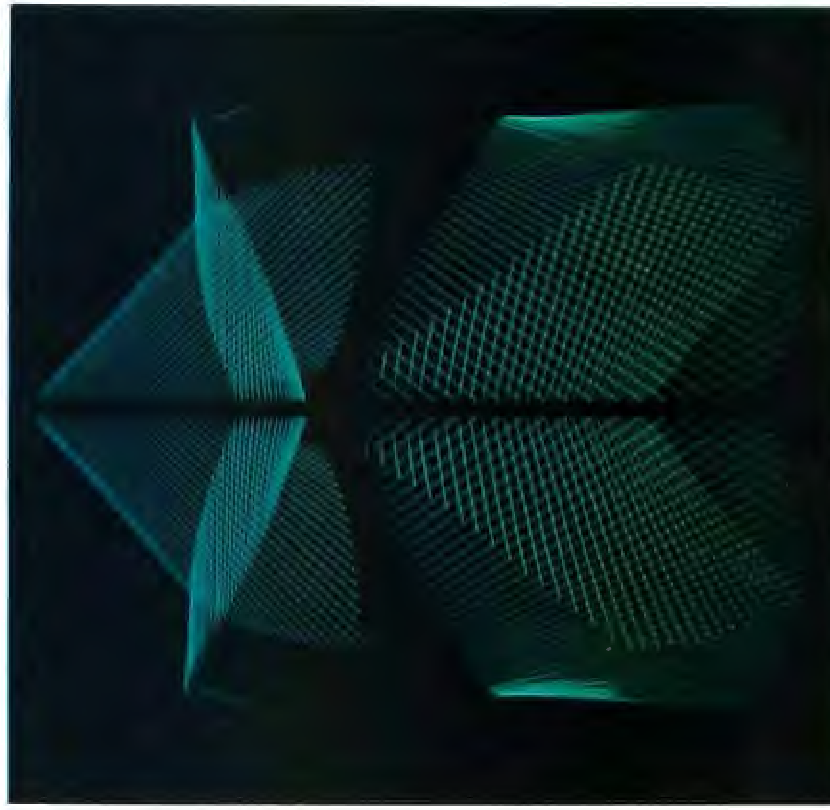


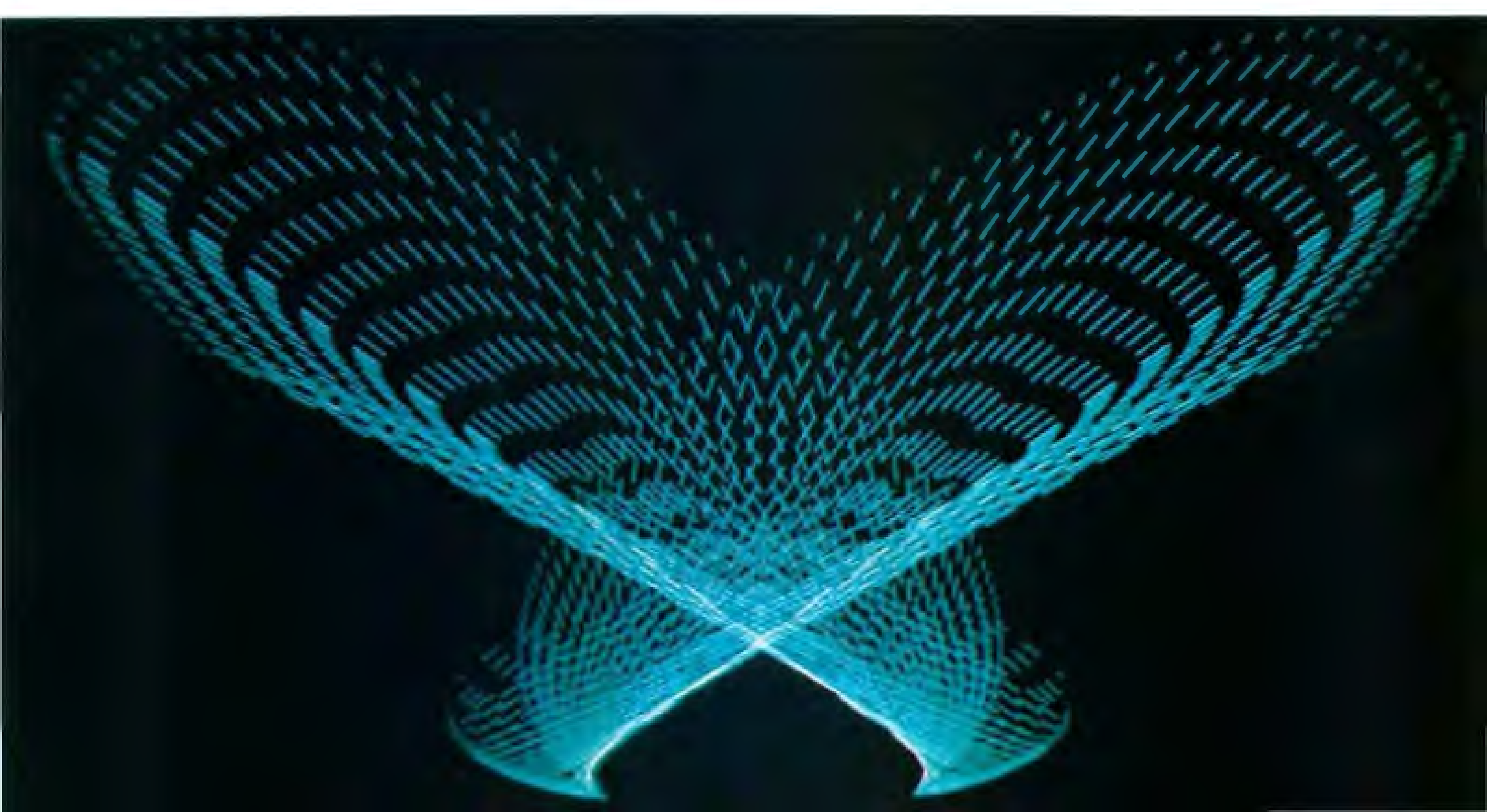
Photo 6.

of GRASS, work which may result in some heady progress in low cost graphics art using computers.

Around the BYTE offices, we tend to think of the cover as the "aquamarine-print" of the design of a house for the inhabitants of Arcturus IV. Whether this is true or not is open to some question.

The other photographs also came to us untitled, so we list them simply as photos 1 thru 8. Why? Well, it occurred to us that a bit of informal fun (and perhaps a bit revealing as well) could be had by leaving it to readers to suggest appropriate titles. Watch the letters columns of future issues to see what comes of this request for titles.■

Photo 7.



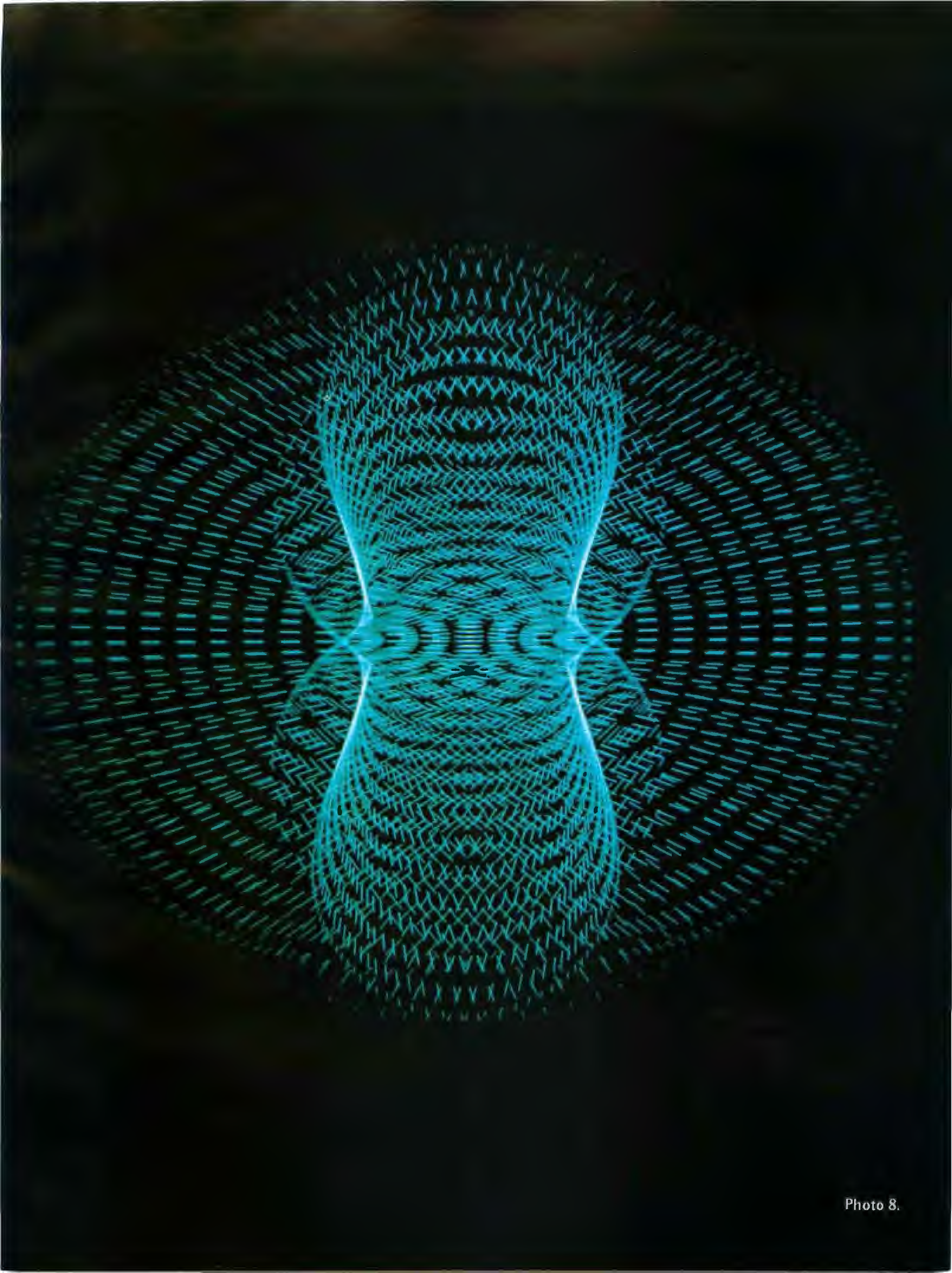


Photo 8.

Relocatability and the Long Branch

Robert J Borrmann PhD
Associate Professor of Electrical Engineering
Manhattan College
Riverdale NY 10471

A position independent relocatable program is a program that can be moved to any convenient place within your memory space, and executed without any changes. One of the very nice things about the Motorola 6800 instruction set is the relative addressing mode used in its branch (BRA) and branch to subroutine (BSR) instructions; instructions using this addressing mode are inherently relocatable without patching.

To illustrate relative addressing consider the 6800 instruction: 20 35 (hexadecimal). Hexadecimal 20 is the code for the branch (BRA) instruction, so 20 35 means jump to the address that is hexadecimal 35 bytes beyond the instruction which follows this branch instruction. Thus, if a program initially occupies hexadecimal locations 0080 to 0200, and includes at location 0100 the above branch instruction, this causes the processor upon encountering it to jump to location 0137. If the program is now copied from its original location 0080 to 0200 into a new location 1580 to 1700, the branch instruction (now at location 1600) will correctly cause the processor, upon encountering it, to jump to location 1637.

Unfortunately, the 6800's branch instructions allow only one byte to be used as the relative displacement. This allows a maximum branching range of only hexadecimal +7F (+ 127 decimal) to hexadecimal −80 (−128 decimal) bytes. What do you do if you have need for longer-range branching, as is the case in larger programs? If you use the jump (JMP) or jump to subroutine (JSR) instructions, which include absolute address references, the program is no longer relocatable without modification. It loses the position independence feature which makes a generalized program read only memory possible.

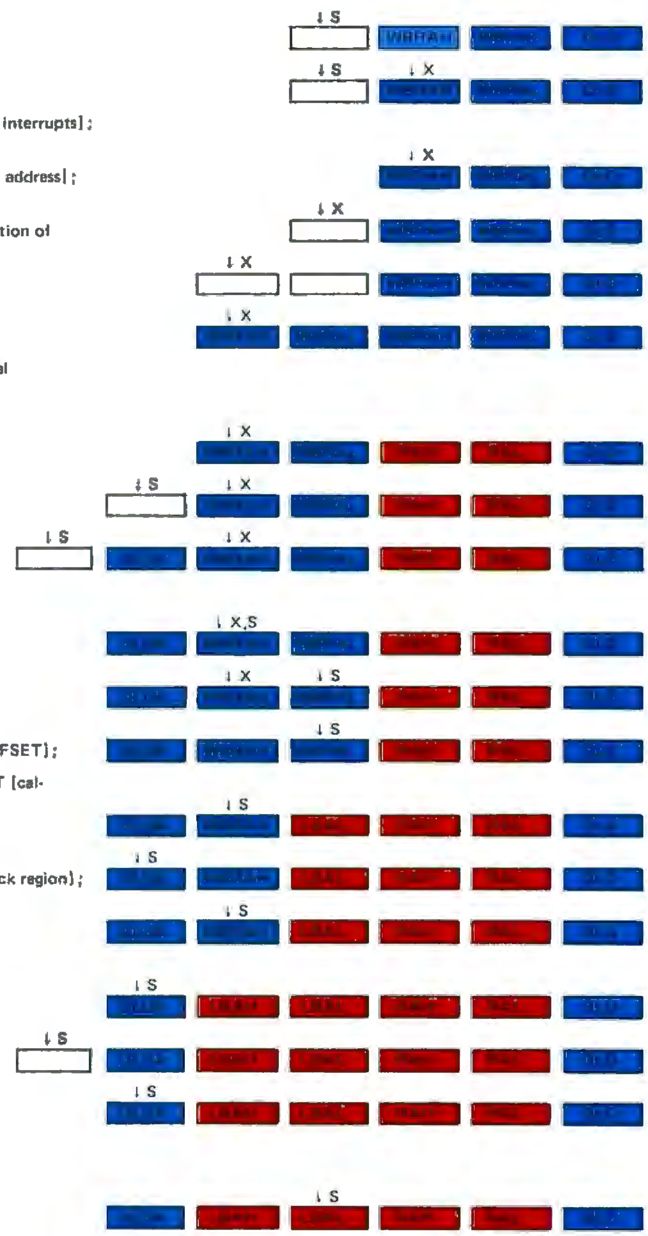
One solution to the problem of writing large relocatable programs is provided by the long branch (LONGBR) and long branch to subroutine (LONGBS) subroutines described here. Although the listings show starting addresses of 278E and 276B respectively, both routines are completely independent of memory address space location. They can be used in two ways.

The first way is to incorporate both routines into your own system monitor, which presumably occupies a fixed location in your memory space. This is the way I use them. Then, to execute a long branch within a program you are developing anywhere in memory, you simply execute a jump to subroutine LONGBR, and follow the JSR LONGBR instruction by a 2 byte adder which indicates how many bytes ahead or

Listing 1: A symbolic assembly language and object code representation of the long relative branch calculation routines for the 6800 processor. The addresses for the routines are picked based on the author's systems software. Since the routines are entirely position independent, they can be relocated without any modification, provided the code does not overlap the temporary storage area for the index register (XSTOR), which is referenced absolutely. The boxes at the right of the commentary in the listing signify the contents of the stack area. The blue color identifies contents based on values present on entry to the routine, and the red color identifies contents computed during the operation of the routine.

Hexa-decimal Address	Hexa-decimal Code	Label	Op	Operand	Commentary
276B	DF 20	LONGBS	STX	XSTOR	[save X for later];
276D	30		TSX		X := S + 1;
276E	0F		SEI		[turn off maskable (IRQ) interrupts];
276F	AE 00		LDS	0,X	S := @X [would be return address];
2771	09		DEX		X := X – 2 [point to location of
2772	09		DEX		WBRA copy];
2773	AF 00		STS	0,X	[copy WBRA];
2775	31		INS		S := S + 2 [calculate actual
2776	31		INS		return address];
2777	AF 02		STS	2,X	[patch the stack];
2779	35		TXS		
277A	36	APUSH	PSHA		
277B	31		INS		S := S + 2 [point stack
277C	31		INS		to WBRAL for PUL];
277D	32		PULA		A := WBRAL;
277E	EE 00		LDX	0,X	X := WBRA [point to OFFSET];
2780	AB 01		ADDA	1,X	LBA := WBRA + OFFSET [cal-
2782	36		PSHA		culate relative branch
2783	34		DES		target and insert in stack region];
2784	32		PULA		A := WBRAH;
2785	A9 00		ADCA	0,X	[complete the sum];
2787	36		PSHA		[finish patching stack];
2788	34		DES		
2789	32		PULA		[restore old A];
278A	DE 20		LDX	XSTOR	[restore old X];
278C	0E		CLI		[reenable interrupts];
278D	39		RTS		[go to address LBA];
278E	DF 20	LONGBR	STX	XSTOR	[save X for later];
2790	30		TSX		[set up for
2791	0F		SEI		branch without
2792	20 E6		BRA	APUSH	return];

Stack Area Manipulations



The following symbols are used for the values in the stack area:

WBRA = "would be branch address," a 16 bit value with high order byte WBRAH and low order byte WBRAL. This is the pointer to the relative branch OFFSET which follows the JSR to LONGBR or LONGBS.

OLD = old contents of stack, unchanged by these routines.

RA = "return address," a 16 bit value with high order byte RAH and low order byte RAL, computed by these routines.

LBA = "long branch address," a 16 bit value with high order byte LBAH and low order byte LBAL, the computed target of the long branch to subroutine or long branch operation.

OLDA = old value of the A accumulator, saved on the stack during computations.

The index (X) register and stack (S) values pointing to the stack area are noted above the content boxes in this representation.

behind you want to branch. For example, if you want to long branch from location 0100 to 0345, you would need the following code:

```
0100 BD 27 8E      JSR  LONGBR
0103 02 42         FDB $0242
```

The target address is computed by adding the offset (hexadecimal 0242) to the address of the byte following the JSR instruction (hexadecimal 0103):

```
 0242
 0103
 ----
 0345
```

To execute a long branch to subroutine the procedure is similar except that you jump to subroutine LONGBS instead of LONGBR. For example, if your program is at hexadecimal location 0240 and you want to branch to a subroutine that is located at 0050, you would need the following code:

```
0240 BD 27 6B      JSR LONGBS
0243 FE 0D         FDB $ FE0D
```

where as usual the target address 0050 is the sum of the offset (FE0D) and the address of the byte following the JSR instruction (0243). Upon encountering the JSR LONGBS instruction at 0240, the processor would go to LONGBS for some massaging of data on the stack, and from there would go to your subroutine at 0050. Upon entering the subroutine at 0050, it will have the same values of accumulators A and B, and index register and stack pointer, as it had when encountering the JSR LONGBS instruction at 0240. The condition flags would in general be different, however. Upon returning from your subroutine, execution would resume with the instruction at location 0245.

When relocating a program using JSR LONGBR or JSR LONGBS instructions you would, of course, leave such instructions unchanged (assuming that your monitor incorporating the LONGBR and LONGBS routines was not itself being relocated). If the location of each long branch or long branch to subroutine call is being changed by the same amount as the target addresses, the program will work the same in the new location, just as was the case with ordinary relative branches.

The second way to use the LONGBR and LONGBS routines is to build them into the long program you are writing. While LONGBR and LONGBS are themselves relocatable, there remains the problem of branching to these subroutines from all locations within the program that will require long branching. This branching can be accomplished by installing "stepping stones" thoughout your program which allows any location within your program to branch from "stone" to "stone," and thereby finally get to these subroutines.

To illustrate, suppose a program occupies memory locations 0200-04CF, including LONGBS at 0200 and LONGBR at 0223. Then the stepping stones might be:

```
        .
        .
        .
027E 20 80   LBS1  BRA  LONGBS
0280 20 A1   LBR1  BRA  LONGBR
        .
        .
        .
02FC 20 80   LBS2  BRA  LBS1
02FE 20 80   LBR2  BRA  LBR1
        .
        .
        .
037A 20 80   LBS3  BRA  LBS2
037C 20 80   LBR3  BRA  LBR2
        .
        .
        .
03F8 20 80   LBS4  BRA  LBS3
03FA 20 80   LBR4  BRA  LBR3
        .
        .
        .
0476 20 80   LBS5  BRA  LBS4
0478 20 80   LBR5  BRA  LBR4
```

To execute a long branch from anyplace within the program, simply execute a branch to subroutine (BSR) to the previous LBRn stepping stone. To execute a long branch to subroutine, simply execute a BSR to the previous LBSn stepping stone. These branches are then completely relocatable. (Of course this technique of stepping stones can be used directly just as well, without LONGBS or LONGBR; but one chain of stones has to be devoted to each branch target. If more than two subroutines [or branch targets] must be used, less code will be required if LONGBS [or LONGBR] is used.)

How It Works

The operation of the LONGBS subroutine can be understood by a study of the program listing and comments shown in figure 1. The entries at the extreme right of the listing show the contents of the stack region of memory at the conclusion of each step, which changes the stack contents or the pointers (index register X or stack pointer S) used to keep track of position within the stack.

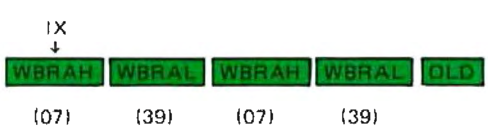
To help understand the operation of the program let us consider an example in which the processor encounters the following instruction sequence:

```
0736  BD  27  6B    JSR     LONGBS
0739  FC  1A        FDB     $FC1A
073B  86  02        LDA A #$02
```

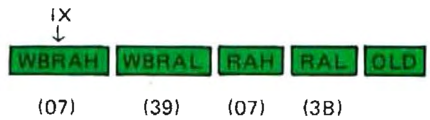
In executing the JSR instruction at 0736, the processor places the address 0739 (which would ordinarily be the return address) onto the stack, and jumps to 276B. However, 0739 as used here is not the actual return address, since it is the location of the 2 byte offset which will be used to form the actual target address. Thus, 0739 is more properly called the "would-be" return address (WBRA). Upon entering the subroutine LONGBS, then, the stack looks as follows:

where SP with the arrow denotes the position in the stack being pointed to by the stack pointer, WBRAH is the high order byte of WBRA, WBRAL is the low order byte of WBRA, and OLD denotes stack contents before executing the JSR LONGBS instruction; these "old" contents of the stack will not be disturbed.

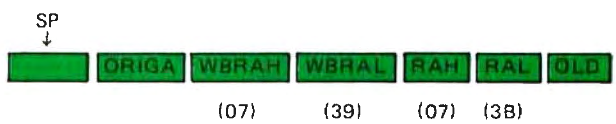
The first line of subroutine LONGBS stores the values of IX away for later retrieval. I happened to use low core location 0020 to 0021 for this purpose, but any available programmable memory location can be used instead. (You may prefer to use a spare 2 byte location in the programmable memory devoted to MIKBUG in many systems.) The next six lines of code transfer to IX the burden of keeping track of our stack location, while the stack pointer itself is used to copy the would-be return address into two additional stack locations. Since we are fooling with the stack pointer, it is necessary to prevent interrupts from occurring at this time. The SEI (set interrupt mask) locks out the maskable interrupts until further notice. We now have the following stack picture:



The next three lines (2775-2777) correct the would-be return address to the actual return address 073B:

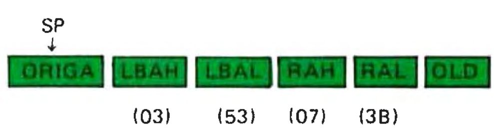


In line 2779 the stack pointer is returned to its usual role of keeping track of position in the stack. Since accumulator A is to be used for some data manipulation in subsequent instructions, its original value (the value it had when JSR LONGBS was executed) is stashed away on the stack for later retrieval. After executing this instruction at PSHA the stack appearance is:



The next ten lines (277B to 2787) cause the would-be return address WBRA to be increased by the value of the 2 byte adder; the result is the target address, denoted by LBAH (long branch address high order byte) and LBAL (long branch address low order byte).

The appearance of the stack now is:



The orginal values of accumulator A and of index register IX are now retrieved, interrupts are reenabled, and the processor jumps to 0353 by executing the RTS (return from subroutine instruction). Notice that because of the manipulation of the stack during subroutine LONGBS, the processor does not return to the program section which called it; instead it jumps to the target address (0353 in this case), because the effect of executing the RTS instruction is to place into the program counter the 2 byte address pulled from the top of the stack.

When the subroutine which begins at 0353 has been finished, the processor executes the RTS instruction which ends it, and this returns the processor to the actual return address 073B.

The operation of the long branch routine LONGBR is similar, except that the RAH and RAL bytes are not needed or wanted. Thus, the initial part of LONGBS which duplicated the value of WBRA on the stack is bypassed.■

Languages Forum is a feature which is intended as an interactive dialog about the design and implementation of languages for personal computing. Statements and opinions submitted to this forum can be on any subject relevant to its purpose of fostering discussion and communication among BYTE readers on the subject of languages. We ask that all correspondents supply their full names and addresses to be printed with their commentaries. We also ask that correspondents supply their telephone numbers, which will be printed unless we are explicitly asked to omit them.

Defining LIL, a Little Interpretive Language

Jack Cluff
34-57 73rd St
Jackson Heights NY 11372

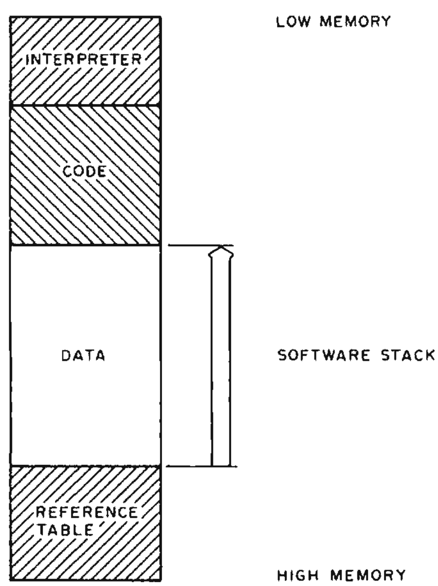


Figure 1: How memory is arranged in the LIL language.

```
code
C8 (hex)      = control byte
 RADL         reference address of
 RADH         descriptor of COUNT
 RADL         reference address of
 RADH         descriptor of numerical constant 1
 OAR          offset to address of routine "+"
 OAR          offset to address of routine "INTO"
 RADL         reference address of
 RADH         descriptor of COUNT
 OAR          offset to address of routine ";"

reference table
 DATA         data for numerical constant 1
 DES          descriptor for numerical constant 1 (integer 1 byte)
 DADH         data address
 DADL         for COUNT
 DES          descriptor of variable COUNT (say integer 3 bytes)

software stack
 DATA
 DATA         data for COUNT
 DATA
```

Listing 1: Linkages for the source statement COUNT +1 INTO COUNT; in the LIL language.

It is with great interest that I read the discussion by Donald J Stavely in the April 1977 Technical Forum. Mr Stavely has articulated a concept in which I have been greatly interested for the past several months, and in which I have invested a certain amount of work.

LIL, Little Interpretive Language, is a pseudocompiling language whose compiled code could be run on any microcomputer which has a LIL loader and a LIL interpreter. LIL is still in a state of flux, but may be spoken of in general terms. It is to consist of three separate programs: the compiler, the loader and the interpreter.

The compiler is to accept character keyboard input and build three structures: the symbol table (to be discarded at the completion of compilation) which contains the names of objects, the reference table which contains descriptors of objects and the memory addresses at which those objects may be found, and the compiled code which contains 2 byte descriptor addresses and 1 byte offsets to a table of function routines used by the interpreter.

The loader is to load reference tables and code into the proper areas of memory for processing by the interpreter, and is to modify all initial linkages established by the compiler so as to agree with the new positions of reference tables and code.

The interpreter is to interpret only addresses of descriptors and offsets to function routines. The interpreter is to allocate memory for data storage on a software stack; the hardware stack is to be used for data manipulation such as arithmetic functions.

The linkage for the source statement:

COUNT +1 INTO COUNT;

can be represented as in listing 1.

Control bytes serve the useful purpose of defining which of the following units of code are address pairs and which are singular offsets. The control byte in the previous example links to the code as follows:

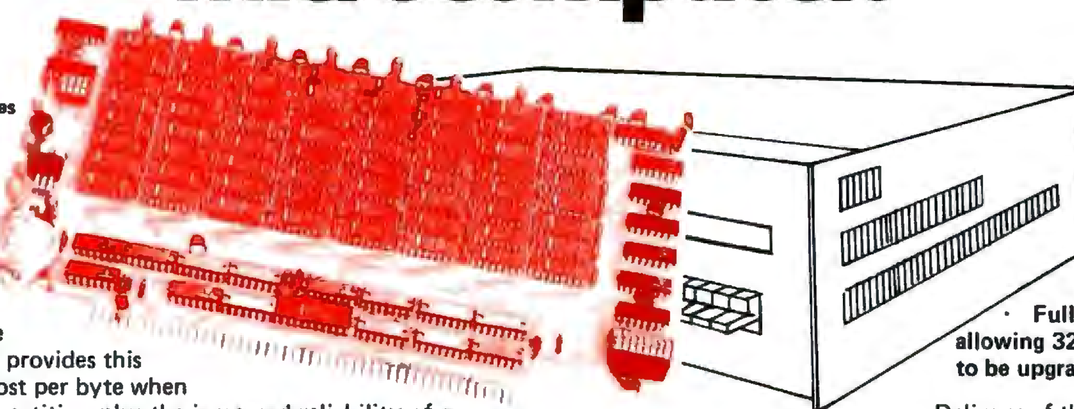
```
  1    1    0    0      1    0     X    X   (X = don't care)
 /    /    /     |       \     \
COUNT  1   "+"  "INTO"  COUNT  ";"
```

Continued on page 181

extensys corporation

64K for $1495

The most cost effective products for your microcomputer.



RM64
64K bytes

THE EXTENSYS RM64 MEMORY BOARD provides the most cost effective system memory found in the industry. The RM64 provides this because of our low cost per byte when compared to our competition plus the increased reliability of a single board over multiple boards containing less memory. The board is S-100 bus compatible making it usable in over a dozen different microcomputer systems including ALTAIR and IMSAI. The RM64 is available in three configurations: 32K, 48K or 64K bytes of memory all on ONE board. The board is completely assembled, checked out and burned in for at least 50 hours prior to shipment. This complete testing procedure allows Extensys to provide a one year warranty on parts, labor and materials (assuming no misuse of the board occurs).

On board hardware is provided for:

- **Individual memory bank address selection in 8K byte increments;**
- **Complete dynamic refresh logic without loss of processing efficiency while programs are running;**
- **Board select logic which allows more than one 64K byte board per system;**
- **S-100 bus compatibility including on-board voltage regulator;**
- **Memory overlap which allows memory sharing the same address space to coexist in the same system;**
- **Write protection in 16K blocks; and**
- **Fully socketed for 64K, allowing 32K and 48K versions to be upgraded at a later date.**

Delivery of the RM64 is 15 to 30 DAYS upon receipt of order. Prices for the RM64 include shipping and handling prepaid in the continental United States.

EXTENSYS Corporation is also announcing several other new highly cost effective products. These include a total floppy disk system based around File I/O board and a multiprocessor operating system. The other product, which interfaces with the RM64 memory board to create a megabyte or more of memory and adds full DMA capability to the File I/O board, is a Board Select/DMA board. Both of these products are S-100 compatible.

Circle 360 on inquiry card.

Contact your local computer store or order directly from EXTENSYS

B-4

extensys corporation
592 Weddell Drive
Sunnyvale, California 94086
(408) 734-1525

Please place my order for the following:

QTY	DESCRIPTION	AMOUNT
___	RM64-32K byte board @ $ 895 ea.	___
___	RM64-48K byte board @ $1195 ea.	___
___	RM64-64K byte board @ $1495 ea.	___
	Subtotal	___
	California residents add 6½% tax	
	TOTAL	___

Shipping and handling prepaid in continental United States

NAME ______

ADDRESS ______

CITY ______ STATE ______ ZIP ______ SIGNATURE: ______

PHONE(INCLUDE AREA CODE) ______

Please check method of payment:
- ___ Check Enclosed
- ___ BankAmericard No. ______ expiration date ______
- ___ Master Charge No. ______ expiration date ______

Continued from page 12

and hazardous 117 VAC power trace running long distances on a power board. The big boys have their equipment tested and listed by UL. It's a necessity if they expect commercial sales. If personal computer manufacturers want the kind of mass market you've been predicting in BYTE, they will have to follow suit. This stuff moves in interstate commerce, and if vendors sell a shock or fire hazard to some naive user, they can expect to be spanked by the Feds, something that tends to be a corporate hurt, and will damage this fledgling part of the computer industry. All the manufacturers who haven't, should clean up their acts before the Feds force something distasteful on them.

William R Hamblen
946 Evans Rd
Nashville TN 37204

A NEW SUBSCRIBER COMMENTS

How long have you been publishing BYTE? Are any back issues available, or reprints of articles in back issues? Do you publish an index to previous issues?

Could you recommend a book, or books, that are readily available, or list their sources on microcomputer soft and hardware, particularly software (from machine language level to high level programming languages)?

Daniel Owen Jenkins VIII
Box 201 RD#1 Clinton Grv
Weare NH 03281

We have been publishing since September 1975. No back issues are available; however, The Best of BYTE Volume I *at $12.95 contains most of the editorial materials from issues 1 to 12. An index is available for Volume I: send a self-addressed stamped envelope.*

PRODUCTS NEEDED

The following are some things we think are needed in this field:

1. A real time clock-calendar board for the S-100 bus with an IO port and recharging circuit which would display time, date and *day of week* on video monitor or other peripheral. Calendar would be at least 200 year. Would accept input to start clock or reset it, such as a 60 Hz line signal, or a time signal from WWV, or a manual signal. Could be programmed (EPROM) to exhibit holidays and special dates or times, or emit a signal to initiate some action at a present time and date. Would be designed to be highly resistant to transients in power supply. Would have 12 hour AM-PM or 24 hour switch selectable option, and a local standard time or daylight savings time or Greenwich time switch selection feature.
2. An ASCII-EBCDIC converter board for the S-100 bus.
3. A generalized converter board for the S-100 bus, under software control, which could enable the user to readily recode keyboards, printers, and video displays with no more than a few keystrokes. Could be used as an ASCII-EBCDIC converter, or could convert a keyboard from a standard QWERTY keyboard into a Dvorak Standard keyboard with a single command, or convert an ASCII into an APL or other special character set keyboard.
4. A convention to standardize the bus for the coming 9900 family of 16 bit processors before different buses proliferate.
5. A directory or clearinghouse of resources in this field, so that persons wishing to contact others doing or able to do things of a certain kind could do so.

Jon D Roland
Micro Mart
1015 Navarro
San Antonio TX 78205

FASTER MULTIPLY?

```
ENTRY: A=multiplier; E=multiplicand

ENTRY: A, E, C=unchanged; B, D =0; HL=product

Address  Op  Operand     Label Mnemonic     Commentary
006000   006 010               MVI B 010    Set B register for count
006002   041 000 000           LXI H 000000 Clear product registers
006005   124                   MOV D, H     Also clear D
006006   007           LOOP    RLC          Rotate left
006007   322 013 006           JNC SKIP     Skip on zero bit
006012   031                   DAD D        If nonzero, add multiplier
006013   005           SKIP    DCR B        Check counter
006014   310                   RZ           Exit if last time
006015   051                   DAD H        Arithmetic left shift of product
006016   303 006 006           JMP LOOP     Repeat
```

The quick and simple 1x1 byte multiplication routine of Christopher Glaeser (July 1977 BYTE, page 142) can be made even quicker and simpler by shifting the A register *left* instead of right. Using the RLC command instead of RAR preserves the value in the A register. The E register is also not altered. Thus, both multiplier and multiplicand are unchanged upon exit from the subroutine. This is an advantage in manipulating arrays, etc.

See box for register status.

The whole thing occupies only 16 bytes.

Leonard Morgenstern
POB 81
Rheem Valley CA 94570

COMMENTS ON SELECTRIC IO INTERFACING

As one who has designed an interface for both input and output between a Selectric 731 and my Z-80 system, I was interested to read Dan Fylstra's article "Interfacing the IBM Selectric Keyboard Printer," June 1977 BYTE, page 46. Dan is to be complimented on a very good technical description, especially when compared with the naive and often just plain inaccurate information which has been published recently on this subject.

There are several supplementary points which may be of interest to BYTE readers:

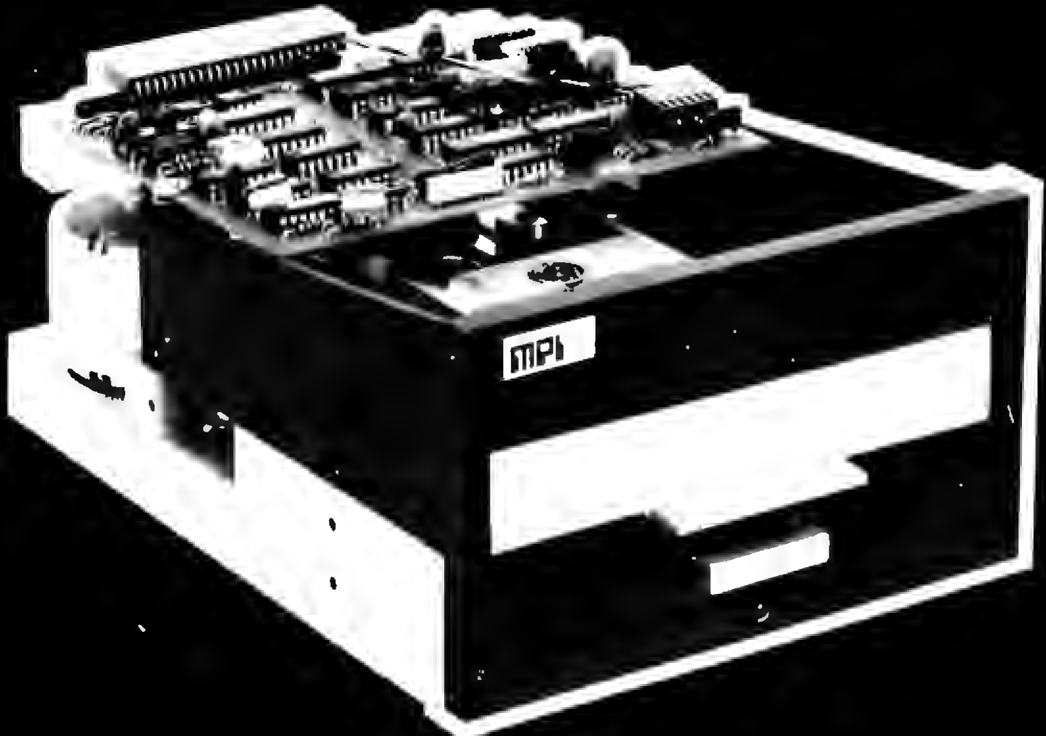
1. Dan gives an excellent description of the differences between a BCD and Correspondence coded 731 or 735. But there are three distinct versions of the Selectric which have been thought suitable for use with a computer and the differences between them are also worth noting:

a) The office Selectric is a light duty mechanism in the same sense in which the term is used for the Teletype Model 33. It won't fail for a long time but the office Selectric is less rugged mechanically than the "heavy duty" mechanisms designed for use as computer IO devices. In my opinion, little reason exists to attempt to convert an office Selectric for computer use since a mechanism designed specifically for that purpose is available at a comparable price.

b) The heavy duty Selectric IO mechanism was marketed by IBM in two basic forms. The Selectric IO mechanism itself (without any magnets, switches, wiring harness or covers) was made available to original equipment manufacturers (OEMs) to be incorporated into non-IBM terminals such as the Dura, Itel, Anderson Jacobson, Datel, etc. The designation for these mechanisms was 745 (15 inch carriage). They are often marked SER (meaning special engineering request) to denote modifications made by IBM at the manufacturing firm's request. These OEMs installed their own magnets, switches and interface electronics to achieve their own individual products. Many of these are now available on the surplus market. I have also seen some of these IO mechanisms which have been partially outfitted with IBM magnets for use as output printers only. Where OEM mechanisms have been used in a non-IBM product, IBM will service the Selectric mechanism only and not the magnets, switches or electronics.

c) The second basic form of the heavy-duty Selectric IO mechanism is the one which IBM outfitted for use as an electronically driven IO device by installing its

A STAR IS BORN!



MPI Flexible Disk Series B51 is a new generation of 5¼" flexible disk products employing a 'band driven' head positioner providing very fast, highly accurate access eliminating 'hysterisis'. A new generation of floppy's is born featuring:

Data Integrity Using the 'band drive' concept for precise head positioning, better error rates are realized. Decreased mechanical wear guarantees diskette interchangeability for the life of the device.

Reliability The Series B51 uses 30% fewer components than competitive devices and together with the low power Schottky TTL inherently provides you the user 10,000 hours MTBF and no preventative maintenance.

Low Cost Storage The Series B51 is very small. Its dimensions are 3.25 x 5.55 x 8.0 inches and weighs only 3 lbs. Using double density encoding techniques the device can store up to 250K Bytes (unformatted). Single quantity price is $320.

The S100 Bus Controller Available to the hobbyist is the R100D intelligent double density controller compatible in architecture to the S100 bus and the Series B51 or its larger family member the Series B82. It features board microprocessor, DMA, simple macro commands, multiple sector transfer, formatting, up to 4 drive control, and requires no overhead software. Single quantity price is $299.
Also available is a complete system featuring single or dual drives, power supply, cabinet and a S100 Controller with a disk operating system. Single Drive System $699. Dual Drive System $999.

MPI is your Company Follow the 'star' and you will get the best price/performance buy in the market. For further information or dealer/distributor packages, call or write Keith Ullal, **Micro Peripherals, Inc.,** 8724 Woodley Avenue, Sepulveda, California, 91343 (213) 894-4076.

Bank of Americard MasterCharge Personal Check Accepted

Circle 397 on inquiry card.

MPI
MICRO PERIPHERALS INC.

own magnets and switches. These are the Models 73, 731 and 735 (hereafter "73X") which are out of production in this country; I'm told that they are still produced in Europe. The 73Xs were used as the 360 console device, in MTSTs (with some wiring modifications) and were available to those who wanted a complete unit (less interface electronics) which IBM would service. Current maintenance agreement rates for 73Xs are about $150 a year. The 73Xs came in three flavors: BCD, Correspondence and MTST.

2. In considering which of the three kinds of Selectrics to use with a personal computer, the most important issue is feedback. Each time a character is sent to a Selectric mechanism (either by energizing magnets or by pressing a key), the cycle clutch is released; at the end of the print cycle, this clutch reengages to halt the cycle shaft after 180° of rotation. If a substantial volume of printing is to be done on a Selectric, the wear on this clutch becomes a serious problem. The only satisfactory solution is to send the next character at a time during the print cycle when the current character has been "processed" by the mechanism but before the cycle clutch has reengaged. IBM calls this the "closed loop" mode of operation. Characters which are sent to the Selectric during this 3 ms window operate the mechanism in its so-called "repeat cycle" in which the cycle clutch does not reengage and the maximum character speed is obtained. The timing required to operate in this way must be derived from switches which sense the state of rotation of the cycle shaft, the operational shaft and several other mechanism states such as whether a carriage return or tab operation has been completed. The time interval between successive characters or machine operations depends on many factors and is not constant. A different interval is required not only for each character and each machine operation, but these intervals vary with the age and condition of the individual mechanism.

There are two consequences of driving a Selectric mechanism without this feedback system: the speed of output is considerably reduced, in most cases below 10 characters per second because worst case timing delays must be introduced after each character or machine operation; and wear on the mechanism increases in proportion to the volume of output. On these points, IBM comments:

> *Feedback contacts are timed to permit initiation of a "next cycle" prior to the end of the "current cycle." This avoids completely stopping the machine between cycles and insures optimum hardware longevity by reducing the frequency of engagement and disengagement (under full power) of the IO mechanical clutches. . . .maximum speed, reliability and longevity is possible only with* [the closed loop] *mode of operation. . . .the IO should receive commands relative to machine degrees* [of shaft rotation]. *Reliable operation CANNOT* [emphasis in original] *be guaranteed when an OPEN LOOP mode of operation is employed. . . .occasional loss of a machine cycle is inherent in an open loop mode of operation.* (Service Manual: Selectric IO Typewriter, IBM Form # 241-5737-0, July 1973).

3. There is a great deal of confusion concerning the existence of a "print" magnet. Some OEMs have installed a magnet with this function, but the 73Xs don't have one. Dan correctly states that the cycle clutch is released whenever *any* of the magnet armatures is pulled down. But he then states "the trip mechanism is connected to a seventh magnet called 'check'." He should have said "also connected." The rule for 73Xs is that whenever any of the seven magnets is energized, a print cycle will occur. Energizing only the "check" magnet will initiate a print cycle without rotate or tilt, printing the home position characters.

4. The only piece of serious misinformation in Dan's article concerns his untested suggestion for closed loop operation. It's close but "no cigar"! Dan suggests generating a debounced TTL level signal from the switches connected between a and x (normally closed) and b and x (normally open) on the 50 pin plug. The switches connected in series this way are C2, C3, C4, C5 and C6. Dan correctly states that the magnets should be deenergized when the NO side makes and that the next character or machine operation should be initiated when the NC side remakes. But the carriage return and tab interlock contacts, which signal the longer (and variable) times required to complete those operations, have been left out of the chain! The correct procedure is to connect a to d externally (at the 50 pin plug) and use Y instead as the NC side. In addition, the figure 9 description of a debouncing circuit should not ignore the IBM specified current through the switches (10 mA @ 10 V minimum) which keeps the contacts clean.

5. Those who attempt the input side of 73X interfacing should know that there is one aspect of the IBM specifications which should not be followed (all else should definitely be respected). IBM specifies sampling of the character transmit contacts on the making of the normally open side of C1. This works fine when sampling means using the signals to switch relays (eg: in the console of the MTST unit). Typical closing time for these relays is 10 ms and, since C1 gates the common voltage onto the transmit contacts for only 15 ms, it is important to begin switching the relays at the earliest possible time. Furthermore, contact bounce is not important in that

Texas Instruments makes MOS EPROMs even more affordable.

TMS 2708 now $21.50*
The industry standard.

TMS 27L08 now $26.15*
The low power 8K.

TMS 2716 now $36.90*
The 2708 times two.

*100-piece prices

Remember a few months ago when EPROMs were expensive and hard to get? Due to TI's leadership they have become available microcomputer building blocks.

Prices have dropped dramatically; availability is excellent. Credit TI's high-yield, high-volume production.

TI's highly cost effective EPROMs feature a rugged, high-integrity ceramic package with sturdy gold-plated pins to withstand the repeated handling and insertions associated with reprogramming. And a gold-alloy-sealed lid for superior hermeticity.

TI offers a choice of three production EPROMs—all from stock.

- TMS 2708. The industry standard 8K EPROM. Fully TTL compatible.
- TMS 27L08. The industry first low power 8K EPROM fully compatible with the 2708. But less than one-half the power dissipation and 10% power supply tolerance.
- TMS 2716. A 2708 times two. Twice the memory (16K) in the same space. An economical plug-in upgrade for 2708s. And TI's 16K 2716 uses less power than a single 2708.

To order the affordable EPROMs, call your nearest authorized TI distributor listed to the left.

TEXAS INSTRUMENTS
INCORPORATED

93188A

Circle 362 on inquiry card.

application. Even though you may debounce the transmit contacts, I would recommend waiting the necessary 3 to 5 ms until the bounce from the making of C1 disappears before latching the state of the transmit contacts.

I am currently considering whether there is sufficient interest in a complete, plug-in-and-go IO interface to the 73X to warrant production of my design which meets all IBM specifications. I would offer a completely assembled and tested interrupt driven card for the S-100 bus with on card PROM for driver software, cable with connectors, and power supply for either the 24 V or 48 V magnets. The price would have to be $395 for the card (including software in the PROM) and about $100 for the 50 wire cable and power supply. I would appreciate receiving indications of interest from readers who would find such an interface attractive.

Beardsley Ruml II
3045 Ordway St NW
Washington DC 20008

CLOSED LOOP SELECTRIC MORE COMMENTS

This letter is in reference to Mr Fylstra's article on interfacing to Selectrics, which appeared in your June 1977 issue, page 46. Although the methods he is suggesting are simple, they have several serious drawbacks.

First, he does not use the keyboard lock mechanism during the time that the Selectric is under computer control. This is important to prevent accidental keyboard inputs which may very easily cause damage to the machine.

Also he suggests using a 5 V supply driving TTL for interfacing the contacts. IBM specifies a minimum voltage and current of 10 V at 10 mA for reliable operation of these contacts.

Lastly the method of driving he suggests is a form of open loop mode. Since the Selectric is not a synchronous device, output rates are only nominal. The importance of operating the machine in a closed loop mode cannot be overstressed. This is the only method recommended by IBM. Using the closed loop mode not only minimizes wear and tear on the Selectric but also provides output at the maximum rate of the machine.

I speak from experience, since our firm is in the business of manufacturing various forms of interfaces to Selectric mechanisms.

John Schwartz
IBEX
1010 Morse Av, Suite 5
Sunnyvale CA 94086

APL CHARACTER GENERATORS ARE AVAILABLE

I have noticed various letters in your letters column from APL enthusiasts. Our firm produces a CRT terminal with an APL character font and keyboard (see June 1977 BYTE, page 24). The APL character generator is a small piggyback board which can directly replace a 2513 and plug into its socket. (Bit 7 is wired through normally unused pin 23 of the 2513.) On the keyboard we use decals mounting on the front keycap face. These decals are white on clear, of excellent quality and durability.

We would be pleased to make these items available directly or through distribution. Interested persons may write me directly.

M C Volker
President, Volker-Craig Ltd
Waterloo Ontario N2J 321
CANADA

A NOTE FROM A EUROPEAN READER

I had my computer shipped air freight, and when it arrived, I found a very simple 50 Hz modification for my Southwest Technical Products CT1024. I cut the 60 Hz reference line. I assume this does bad things to a phase locked loop somewhere (my documentation isn't here either) but after it warms up it seems to work fine.

Bruce Turrie
Riverside Tower Esmoreitlaan 3
POB 3, Apt 68, B-2050 Antwerpen
BELGIUM

USING ADCs FOR TEST INSTRUMENTS?

I imagine this could have a great deal of interest to your readers. If you cannot develop a solution to this problem maybe your readers can.

I am working on a research project for Scott Community College. I am working on a hardware and software scheme that will make a color TV into an oscilloscope. I am using the TV Dazzler and the D+7A interfaces along with the IMSAI 8080. I want to do the rest of the task through software. I am having problems designing the software needed in conjunction with these interfaces. Any help that can be provided will be greatly appreciated. It need only cover audio frequencies. It need have no extended sensitivity nor multicolored display.

John Orvis
2125 Olympia Dr
Bettendorf IA 52722

A good idea for an article!

SELF-MODIFYING CODE

The letter "A Critique of Self-Modifying Code" in the June 1977 issue of BYTE will serve only, I predict, to discourage some readers from sending material to BYTE for publication. The BARC routines, applied within the context for which they were intended are certainly worthwhile.

Phil Rynes
6824 Roberts Dr
Woodridge IL 60515

NEED VA APPROVED COURSES?

Some time ago one of your readers wrote that he would be interested in taking a VA approved course in which the student would build a microcomputer. This subject has also been of interest to me since I am presently taking a course in digital electronics at The National Radio Institute in Washington DC. NRI is one of the leading VA approved correspondence schools in the country.

I wrote the president of NRI, J F Thompson, concerning the subject of microcomputers. The attached letter is the warm response I received to my inquiry. I thought some of your readers would be interested.

Robert N Smith
1617 Grunther Av
Rockville MD 20851

McGraw-Hill Continuing Education Center

3939 Wisconsin Avenue NW
Washington DC 20016
Telephone 202/244 1600

President

January 5, 1977

Mr. Robert N. Smith
1617 Gruenther Avenue
Rockville, MD 20851

Dear Mr. Smith:

I received your letter of December 19, 1976, and appreciate your interest in whether NRI is considering a course in Microprocessing with microcomputer kits. The answer simply is "yes". We believe that a technology program on microprocessors will help the computer industry bridge the shortage of service technicians.

Our thoughts at this time are to expand the Digital Computer Electronics course to include microprocessing and kit hardware. The microprocessor breadboarding kit will contain power supplies, a clock oscillator, a programmable read only memory, LED indicators, and control switches. When used in the course, it will serve as a teaching tool to reinforce the material in the lessons. It will provide real programming experience on a commercial microprocessor. Further, the design calls for its use with a number of peripheral devices. The specific devices, like CRT, video displays (TV), cassette recorder, discs, and others have not yet been determined.

We are confident that NRI will offer a microprocessing course that has high quality, a reasonable price, and be approved for veterans training. It is too premature now to provide you more specific information.

Again, thank you for taking the time to write me. Good luck in your studies for 1977.

Sincerely,

J. F. Thompson

JFT:mth

Disk Based Co
by Ohio

Any serious application of a computer demands a Floppy disk or hard disk because a disk allows the computer to access programs and data almost instantly instead of the seconds or minutes required with cassette systems. In real-world application of computers, such as small business accounting, a cassette based computer simply takes too long to do the job.

Ohio Scientific offers a full line of disk based computers utilizing full size floppy disks with 250,000 bytes of formatted user work space per disk. That's 3 to 4 times the work space of mini-floppies.

Challenger II

Challenger II is available with a single or dual floppy disk and a minimum of 16K of RAM instead of ROM BASIC. The disk BASIC is automatically loaded into the computer so there is no need for ROMs.

Ohio Scientific's powerful disk operating systems allow the computer to function like a big system with features like random access, sequential, and index sequential files in BASIC and I/O distributors which support multiple terminals and industry-standard line printers.

Challenger II's with disks can have the following optional features:

• 16 to 192K of RAM memory • Single or dual drive floppys • Serial and/or video I/O ports • Up to 4 independent users simultaneously • Two standard line printer options • Optional 74 Megabyte Hard disk • Much more

Challenger II disk systems are very economical. For example a 16K Challenger II computer with serial interface, single drive floppy disk, BASIC and DOS costs only **$1964.00** fully assembled.

mputer Systems Scientific

Challenger III

Ohio Scientific proudly announces the ultimate in small computer systems, the Challenger III. This computer has a 3 processor cpu board equipped with a 6502A, 6800, and Z-80.

This system allows you to run virtually all software published in the small computer magazines!

The Challenger III is fully software and hardware compatible with Ohio Scientific products and can run virtually all software for the 6800, 8080 and Z-80 including Mikbug® dependent 6800 programs!

Incredible as this is, Challenger III costs only about 10% more than conventional single processor microcomputers. For example a 32K Challenger III with a serial interface and a dual drive floppy disk (500,000 bytes of storage) costs only **$3481.00.** Fully Assembled, complete with software. Terminal not included.

☐ **Send me the Fall '77 Catalog. I enclose $1.**
☐ I would like to order directly from this advertisement. (Please allow up to 60 days for delivery)

NAME________________________

ADDRESS________________________

CITY__________ STATE______ ZIP______

To order: Payment by:
BAC (Visa)_____ MC_____ Money Order_____
Credit Card Account #__________
Interbank # (Master Charge)__________

_____ Model 500 Boards @ **$298.00** _____
_____ Challenger IIP @ **$598.00** _____
_____ Super Kit @ **$398.00** _____
_____ 16K Challenger II complete with serial interface, single drive floppy disk, BASIC and DOS @ **$1964.00** _____
_____ 32K Challenger III with serial interface, a dual drive floppy disk (500,000 bytes of storage) @ **$3481.00** _____

Ohio Residents add 4% sales tax _____
TOTAL CHARGED OR ENCLOSED _____

Order directly from: Ohio Scientific, 11679 Hayden St., Hiram, Ohio 44234 or your local OSI dealer
All orders shipped insured UPS unless otherwise requested.

Circle 365 on inquiry card.

OHIO SCIENTIFIC

11679 Hayden • Hiram, Ohio 44234

Continued from page 6

"show off" sequences which were viewable at the NCC show. These included false color processing of digitized pictures, as well as dynamically changing kaleidoscopic abstract patterns.

Why Would Anyone Want . . .?

The technological skeptics always pose a more or less unanswerable question of "why do such and so?". Some unnamed ancestor of western civilization probably received the same question in response to making the first wheeled carts, planting crops rather than gathering them wild, or tending and managing fires as the inspiration for the Prometheus myth. It is a repetitive pattern, uneasiness or fear when confronted with new ways of life.

I first ran into this form of question relative to the whole idea of personal computing when I seriously began exploring the possibility of building my own computer after the Intel 8008 first became available. I vividly remember a conversation with one of my professional associates at the time, which contained the question of *why* applied to personal computers: "Why would anyone ever want a personal computer?" Without answering the question, history has now shown that a lot of people do want such computers. At that time however, I attempted to weakly answer (my mind was made up, remember) with specific applications contrived on the spot as alleged justifications. I knew I wanted one, but I wasn't quite sure why. . .

Today, I would answer with a more general question and an equivalently general response. The more general question is "why would anyone ever want to use an arbitrarily chosen product of human technology (be it automobile, airplane, typewriter, computer,

Memory requirements (256 by 256 resolution, 8 by 8 point characters, 256 characters)

Character definitions	8192	bytes
Screen refresh	1024	bytes
Total dedicated memory	9216	bytes

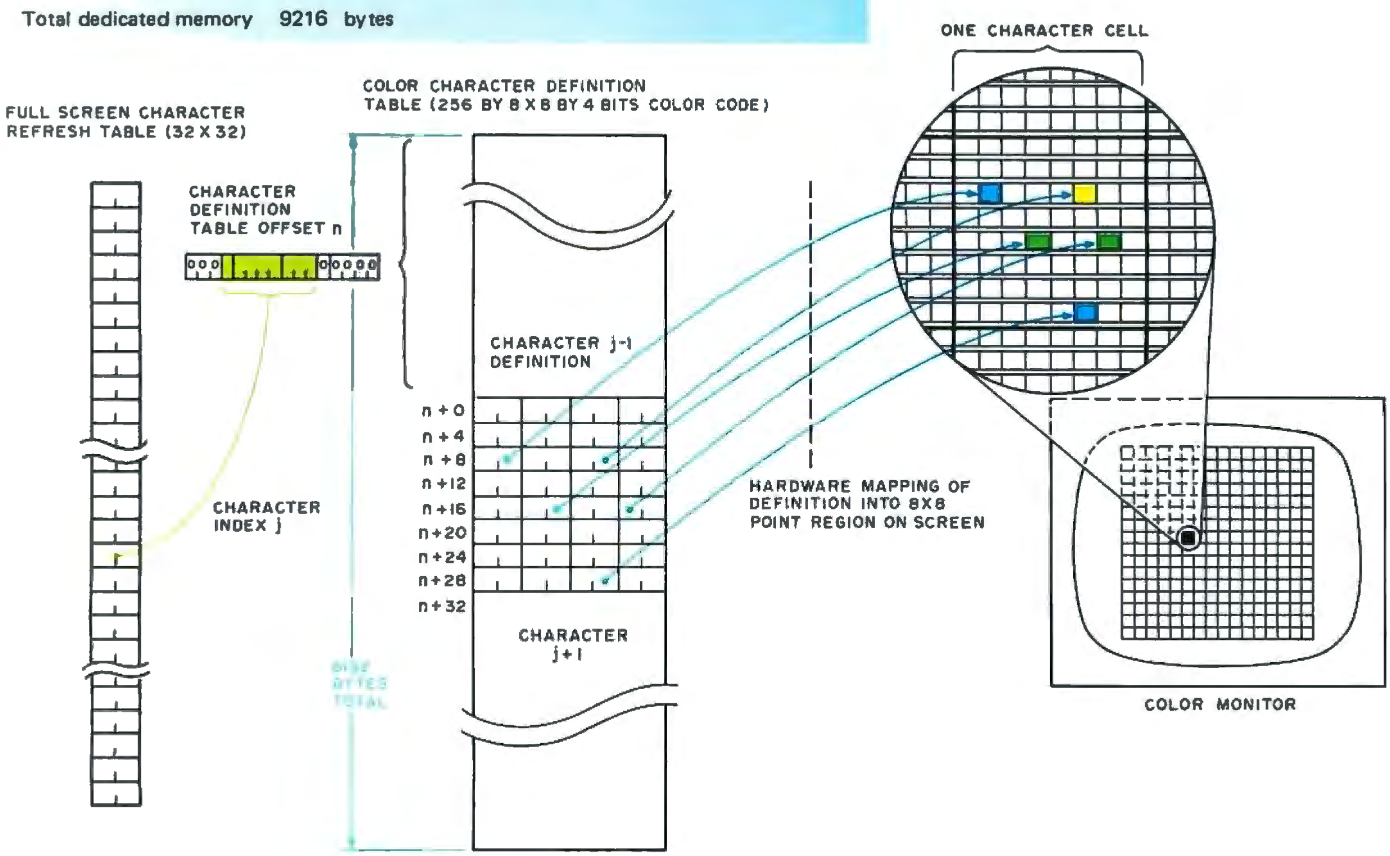


Figure 2: A less direct display method which uses considerably less memory to generate a picture is the reprogrammable color character generator technique. In its most general form, each logical character definition contains a matrix of 4 bit color values for picture elements within the character. A display is formed by filling the 1024 character refresh table entries with character values pointing to the programmable character generator matrix. This form of display in the configuration shown requires only 9,216 bytes compared to the 32,768 of figure 1, a saving which is achieved at a price of considerable flexibility.

piano, microwave oven, or television set)?" I would now reply that individual use of the products of technology has the psychological and emotional justification of personal satisfaction, derived from being able to choose from a wider range of activities with such inventions than without.

The More Specific Whys of Color Displays

The advent of high resolution color displays for personal use is near, as I'll demonstrate later in this essay. I have some rough ideas as to the practical consequences of inexpensive color displays in a personal computing context. Any such comments must be viewed in the same light as my responses to the personal computing skeptic several years ago. I am already committed mentally to the concept of the color display with computing intelligence behind it as an artistic and expressive medium for individuals. My attempt to detail my commitment's consequences with specific uses is necessarily a naive first try at explaining the fascination of the concept, just as my earlier attempt to explain the fascination of personal computing by examples was a weak argument at best.

Versatility and Flexibility

Perhaps a major attraction of the color display is the generality it provides. There is a certain fascination with the realization of the general purpose generic form of any program or hardware conception.

Carrying this attitude into the realm of visual displays, I claim that the color display with relatively high resolution is inherently desirable because it represents the most general form of the concept of a two dimensional display. Every other form of a display is a subset of the general ability to paint arbitrary colors in arbitrary patterns on a color display screen. Any given pattern is a subset of the general set of all possible patterns one could make with "n" color levels and an "i by j" pattern of picture elements.

The consequences of this versatility and flexibility are the possibility of color emphasis and detailed representations in conventional applications plus a whole new set of possibilities in the area of visual arts.

In an article we printed in December 1976, Margot Critchfield of the University of Pittsburgh's Project Solo illustrated the static artistic possibilities in several color renditions using a Cromemco TV-Dazzler and custom peripherals created by Thomas Dwyer and Leon Sweer. (The Dazzler is a first attempt at the personal color organ, a 64 by 64 element display with a limited range of eight colors and "off.") The renditions which she illustrated are examples of the static representations of visual art which are possible merely using the color display as the equivalent of the traditional artist's media of oil or acrylic and canvas.

This use of the color display as a medium of static visual artwork is a necessary first step toward exploring the practical consequences of high resolution color display technology. With use of appropriate inexpensive mass storage devices, libraries of high resolution images can be conceived. (A full size floppy can hold about 500 K bytes of information with contemporary "double density" recording technology, or a total of about 16 unencoded pictures with a 256 by 256 grid of 4 bit color picture elements.) In this mode of operation, the artistic user paints a picture on the display using a joystick or equivalent cursor control plus additional finger manipulated controls to select color, move patterns about with software, repeat patterns, etc. The key here is composition of a color image which is perceived statically. This is the mode of operation of the Project Solo Cybernetic Crayon mentioned earlier, and of the Cromemco Dazzler when it is used with a joystick.

Effective artistic use of this new form of visual imagery creation requires development of interactive software customized to the creator's tastes. The display by itself is not enough to make the facility complete, for it only becomes useful with the software equivalents of paint brushes and motion of the artist's arm. Distribution or copies of this form of the art can be done photographically, since the static image is what counts. In this sense, there is nothing startling or new about use of a color display and its computer backup for static images.

Art Forms Impossible Without Computer Controlled Imagery

Once the artist or experimenter (the two words are actually equivalent) learns to create a visual art work with computer aid the next step is to use the mechanism of the computer to produce effects which would previously have been difficult or impossible to achieve. Motion and change of images according to rules and techniques chosen by the artist are a part of the very act of composition in this new medium of dynamic visual art.

Here, we are talking about a dynamic and moving art form, the use of programming in a manner which can be directly perceived and understood by all viewers simply by observing it, just as music can be appreciated by anyone simply by listening. This is the true excitement of color imagery as an art form. The only visual antecedents are the use of film technology and choreography; but unlike much film and choreographic imagery, it is not constrained to images of human forms, since the display is general purpose and subject to various forms of abstractions and harmonies previously impossible with visual imagery. The key to this new art form is the time dependent algorithmic transformation of elements of the picture according to the artist's plan and implementation.

The time ordered nature of algorithmic visual arts which come from this source make a combination of the display imagery with music inevitable. If evolution of visual form with abstract or specific images is considered as a criterion, an artistic antecedent of this combination is found in the first experiments of Walt Disney in the form of the movie called *Fantasia.* (For those unfamiliar with that movie, it was a combination of classical music with cartoon technology which resulted in a feature length film.)

A later example of this combination of visual patterns with musical patterns is represented in the work of John Whitney over the past decade or so, using computers with photographic technology to make high resolution computer generated films which are synchronized to music. He gave an excellent taste of what can be done in his demonstration films and talks at the Personal Computing 76 show in 1976, and at the First West Coast Computer Faire this year. (He is continuing his work with equipment which has much in common with contemporary personal computing technology.) A major tenet of John Whitney's concept of dynamic visual arts is the idea of visual analogs to the harmonies and melodic evolutions in the musical forms. With the visual processors we can achieve with today's technology, it is possible for many more individuals to begin experimentation with dynamic progressions of forms of sensation which include both visual and aural components. The coming of the high resolution, yet not inordinately expensive, color graphic display opens up the wider use of this art form.

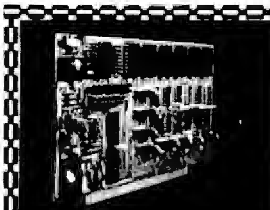
The simplest examples of the algorithmic visual arts have been seen at the various exhibitions. Nearly every computer store which carries the Cromemco TV Dazzler uses the color version of the game of Life as a customer attraction, one mode of algorithmic visual art. A similar display sometimes seen is the colorful forms of a kaleidoscope represented in that product's 64 by 64 color matrix on a TV screen. Another example of a simple, algorithmic art form is the Color Eater program which Apple Computer uses to illustrate the capabilities of its computer and its integral 40 by 40 patch color display generator. But better (ie: high resolution) color technology is sure to follow in the near future.

But Art Is Not All. . .

The programmable versatility of the color display concept is applicable to more than just artistic purposes. From the artistic point of view, the display's content, whether static or dynamically changing, is the object of the exercise. But using the display as a part of the information processing system which is the personal computer is attractive because colors can convey additional information.

Bordering on the concept of dynamic art is the concept of the color display oriented video game. Space War works just fine on a black and white display, for the realism of the simulation only demands points in a two dimensional projection of three space. But consider the possibility of cartoon style animation applied to simulation games. If the game involves a park like setting in which the simulation players move, use the color display to represent that setting, with programs appended for generation of players' movement in the setting. If the game involves simulating a plane landing, or an automobile race, use programs to generate the moving effects on the screen, and variations of the background information. Such suggestions involve significant software development and processor bandwidth when the degree of realism becomes high; but, given the color display and a given processor's capabilities, there can be considerable improvement in the types of displays used with games.

Why put up with a simple numeric time readout for your digital clock software? With a color display for the output device, a very realistic analog clock face could be displayed with moving hands and styling to the user's tastes. The 256 by 256 by 4 bit resolution capability suggested as a near term technological goal should be more than adequate for this task.



Then consider the use of color in highlighting data. For example, one purpose for which I'd like to use a high resolution color display in the future is highlighting the various melodic themes and chord structures in digital representations of music on the display. Here the color information is used to enhance the different segments of a complicated structure, a principle which has been applied at many points in industry and which is certainly just as useful for personal data displays.

Design Speculations . . . Color Displays

While it is unlikely that any personally affordable color display will soon match the facilities of the Comtal 8000-S which got me started on this subject, improved resolution color displays are a very real possibility for personal computing within the near future. By improved resolution in a personal computing context, I mean displays within the bandwidth range of a standard color television set adapted for direct video input. Taking into account memory prices, and the capabilities of the color television device, this means basically some form of raster scan color generator with a 256 by 256 picture element resolution and four bits of color level information per picture element. I see two methods in which this hardware can be implemented, with varying capabilities. The simplest brute force technique is to have the image generation equipment incorporate memory directly to refresh the display. For the 256 by 256 by 4 bit display this brute force technique requires 32,768 (32 K) bytes of memory. (At a 40 by 40 matrix resolution, the Apple II computer's video generator's color mode uses this brute force technique with a smaller amount of memory.) Figure 1 shows conceptually how each byte defines two picture elements' state of color through hardware which accesses the 32 K byte memory region to generate the display. When the price of a typical 32 K byte memory board is standing at about the $800 range assembled and tested for at least one product currently advertised (using 16 K dynamic chips), dedicating such a memory to a color display peripheral for perhaps $1600-2000 end user price is not unreasonable.

The second method of design I see coming is inspired by the video generator methods used in many video arcade games, and in particular the video generator of the Noval 760 computer, with which I became acquainted on a visit to Gremlin Industries in late 1976. This is the method of a reprogrammable character generator, which results in considerable compression of the amount of memory required to completely cover the entire plane of a display. (The inherent disadvantage, a trade off of function for less memory than the brute force technique, is that the picture must be represented as a closely packed array of subelements defined by the different character definitions, meaning that it is possible to run out of character definition space if complex patterns are involved.) Using this second method of indirect refreshing through a progammable character generator extended to include color information, a configuration which might be suitable for color is illustrated in figure 2. The particular design of figure 2 (which is not the only configuration possible by any means) uses 256 character definitions of 32 bytes per character, where each character has an 8 by 8 array of 64 picture elements of four bits per element. The hardware of such an indirect refreshing method scans the 1024 byte (32 by 32) array of characters in the refresh memory, then looks up the corresponding character definition in the user programmable color character generator. This form of the color raster display still gives resolution to a 256 by 256 grid, but requires only 9216 bytes compared to the 32,768 bytes of the brute force technique. As a result, one might expect to find this form appearing in less expensive color oriented peripheral products for personal computers.

With memory prices dropping consistently, I expect both general forms of color peripheral units to be appearing in the near future. The details will differ from these conceptual sketches, but the idea of the color display is here. There is no technological reason why such a display cannot be built and marketed to the readers of this magazine within the next one or two years.■

Attention Authors:

As a "how to" for reader construction, high resolution color displays and software drivers for them are certainly high priority items for future articles to be published in BYTE.

Have you been waiting for a hard copy printer that doesn't come with a high price tag or very limited performance and features?

Your wait is over. The new Integral Impact is precisely what you've been looking for. It's a micropriced, maxi-performance, full-size, full feature, impact printer that's ready to plug directly into your mini or micro-computer system. No special hardware or software interfacing required. Just connect to any RS 232 or current loop serial data source, and you're in business.

The Integral Impact is a complete professional printer that's fully assembled, tested and ready for use. No extra-cost options, nothing to assemble or wire, no special software to write . . . and no need, either, for expensive special paper. It prints the standard ASCII set of 64 characters using a 5 x 7 dot matrix at speeds to 120 characters per second . . .on full-size, 8½-inch paper in fan-fold or roll form.

You can even print multiple copies with up to 132 characters per line. With a long-life (10,000,000 character) ribbon that is automatically re-inked during printing, and a rugged print mechanism with few moving parts, the Integral Impact offers you outstanding service and reliability.

The Integral Impact has a microprocessor-based controller that features variable character pitch sizes, multiple line buffering and enhanced, double-width characters. Serial baud rates from 110 to 1200 bits per second are supported or, optionally, the printer may be used with a parallel interface. Operator controls for power, printer mode and off-line paper motion are conveniently located on the front panel. Internal switch settings determine serial baud rate, character pitch and number of characters per line.

The micro-price: $745.00 complete.

Where can you get it? Send the coupon below to Integral and we'll ship direct to you. Or have us send you full details by checking the appropriate box on the coupon.

Make big printer performance an integral part of your system . . . with the Integral Impact

Now. . . big printer performance at a mini-printer price.

Integral Data Systems INC

5 Bridge Street • Watertown, MA 02172 • (617) 926-1011

_____ Send more information

Name ______________________

Address ______________________

City ______________ State ______ Zip ______

I want big printer performance now.

Circle 371 on inquiry card.

Here's my check for $ ______________________

Bill my VISA ______ Mastercharge account ______

Account No. ______________ Expiration Date ______

M C Interbank No. ______________________

Signature ______________________

*Handling and shipping charge: $5.00. Mass. residents add 5% sales tax.

Commodore's New PET Computer

Photo 1: *Commodore's new PET computer at the NCC show in Dallas.*

One of the most interesting new products at the NCC show in Dallas was Commodore's new PET computer. Beneath its futuristic cover, the unit features 14 K bytes of read only memory containing an 8 K BASIC package, 4 K operating system, 1 K machine language monitor and 1 K diagnostic routine.

The display features 64 graphics characters as well as the standard 64 character upper case ASCII set. This gives the effect of high resolution when displayed on the built-in 9 inch (22.86 cm) video monitor. The keyboard is encoded so that by shifting to upper case, the user has access to the 64 graphics characters.

The 8 K extended BASIC package was designed by Microsoft, the people who have created a number of interpreters for personal computers. It features strings, integers, multiple-dimensioned arrays, 10 digit precision floating point capability, and "peek" and "poke" commands.

The price of the PET is $595 complete with 4 K bytes of programmable memory. The $795 version features 8 K bytes of programmable memory. All IO connections (excluding the built-in tape drive, keyboard and video display) are made via an IEEE-488 bus.

The PET is an excellent example of the true appliance computer: a neat, self-contained graphics oriented package designed for the mass market as well as for the serious experimenter.■

Photo 2: A closeup of the PET's unusual touch-sensitive keyboard with 73 keys and 64 shifted graphics symbols in addition to ordinary upper case ASCII characters.

Photo 3: A side view of the PET highlighting its modernistic lines.

Verbatim™

The whole message is quality.

You want to record your message verbatim—word for word—whether it's bits, bytes or "Dear Folks" translated into word processor language.

Our objective in manufacturing recording media for the electronics industry—digital tape cassettes, floppy disks, mag cards, computer cartridges—is to give you the finest, the best, the most dependable, the most cost-effective.

That means rugged, long-lived, abrasion-resistant recording media with superior magnetic qualities. If we made tires, they'd be steel-belted radials.

We delivered our first digital grade certified tape cassettes back in the beginning, 1969. We made the first commercial 3740-compatible floppy disks that didn't bear IBM's name. And the first Flippy® reversible flexible disks with *anyone's* name on them. The first mini data cassette is ours. And we've got the newest miniature flexible disk, the MD 525.

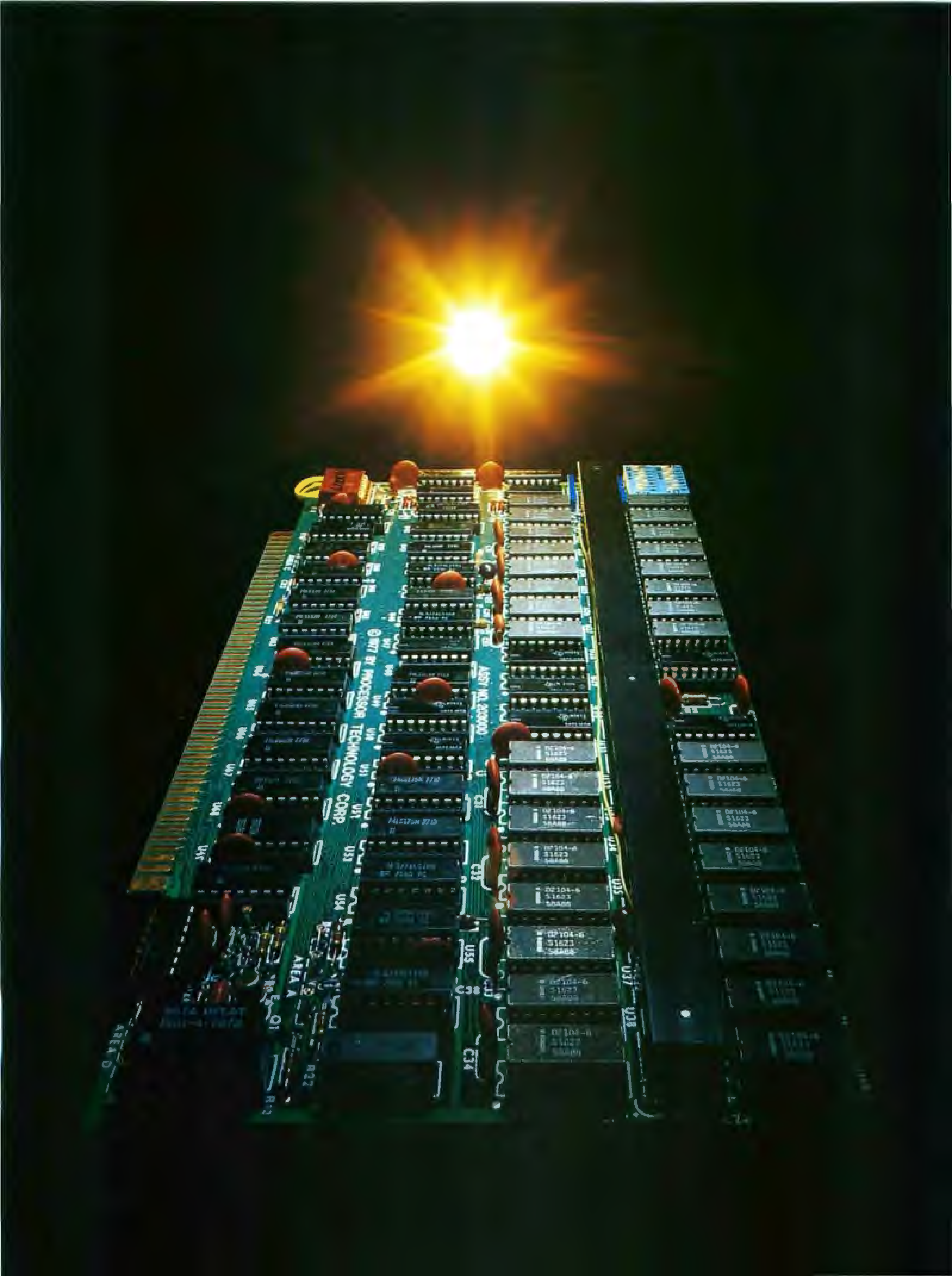
Now, **Verbatim** media. It's a new formulation of ferric oxides, an advanced macromolecular binder system to adhere it to the tough polyester film, and a process control system that demands over 200 separate quality checks before the material is cut, packaged, and certified to be 100% error-free.

The final quality check? "Make it pretty!" Our production people tell us that magnetic recording media is one of the rare instances in manufacturing where aesthetic appearance translates directly into final product quality. It has to look beautiful to work beautifully.

We have the formulas, the machines, the technology to make high quality recording media. But it takes the best people in the industry to deliver Verbatim disks, cards, cartridges and cassettes. You'll find them at your favorite retail computer store.

Circle 372 on inquiry card.

Information Terminals Corp.
Sunnyvale, California


© 1977 BY PROCESSOR TECHNOLOGY CORP.

The NCC:

By Chris Morgan, Editor

Over 36,000 people filled the Dallas Convention Center on June 13 to 16 to attend the biggest National Computer Conference ever. Virtually every manufacturer in the computing field was there either to exhibit or to take notes. But what made this year's NCC different from past shows was the NCC sponsored Personal Computing Fair and Exposition held concurrently in the Convention Center.

Photo 1: The main exhibit hall at the NCC.

Photo 3: Another view of the main exhibition hall.

Photo 2: The Personal Computing section of the NCC.

A Dallas Delight

Photographs by Charles Floto

Manufacturers of personal computers and related products turned out in force to display their wares before large crowds. Hopeful hobbyists and experimenters exhibited their home computer projects at the Personal Computing Fair (devoted entirely to noncommercial individual and group owned projects), and vied to win prizes.

Both sections of the show featured extensive seminars and panel discussions about present and future uses of computers; personal computing club representatives met to discuss the possibility of forming a National Club Congress.

It was gratifying to see long term NCC veterans mingling with hackers from all over the country in a congenial atmosphere filled with microcomputers that played games and dazzled the eye with elegant color graphics. One group learned from the other, and ideas were exchanged at extremely high rates.

Seeing everything at the show was quite a challenge: some 300 manufacturers were spread out over five football field's worth of exhibits upstairs. And the main floor exhibits were *big*: Data General, for example, brought an entire working planetarium controlled by a Nova minicomputer and capable of holding 50 people inside. Many of the "booths" were two stories

Photo 5: Deborah Ashman demonstrating her robot Joey, who moved back and forth to the tune, "If I Only Had a Brain."

Photo 6: Wes Stewart working at the cabinet desk he built for his SwTPC 6800 computer system. The program running is a golf simulation.

Photo 4: Commodore's new PET computer, a popular item at the show. The unit features 14 K bytes of read only memory including an operating system, diagnostic routines and an 8 K BASIC interpreter by Microsoft. It sells for $595 including 4 K bytes of programmable memory. A lengthy waiting list is predicted for the unit.

high; others took up areas of 30 by 60 feet or more.

There was much to see in the Personal Computing section downstairs, too: Commodore's new PET microcomputer, a variety of the latest video games programs (including a tank game for micros), new music programs, talking computers, software packages for FORTRAN, business programs, floppy disks, and so on.

Summarizing the entire show is almost impossible: we found it inspiring (and sometimes frustrating!) to look at some of the state of the art devices, such as high resolution video displays, on view in the main exhibit hall.

From the looks of this convention, the computer industry is in good shape, and we look forward to an even bigger and better show next year.■

Photo 7: Allen Isaacson demonstrates one of the exhibits at the Personal Computing Fair: a video game converted into a color graphics terminal that uses a Teletype instead of a joystick. The hardware was designed by Robert and Richard Benjamin, father and son, respectively.

Photo 8: Personal Computer Fair first prize winner Tom Aschenbrenner shown with the system he amassed over a five year period from surplus computer parts. A Texas Instruments 980A controls a system which stores and transmits messages for the benefit of Dallas area hams.

Photo 9: Photographer Floto reflects on Sperry Univac's shiny display.


XITAN

More on Inexpensive Plotters

Technical Forum is a feature intended as an interactive dialog on the technology of personal computing. The subject matter is open-ended, and the intent is to foster discussion and communication among readers of BYTE. We ask that all correspondents supply their full names and addresses to be printed with their commentaries. We also ask that correspondents supply their telephone numbers, which will be printed unless we are explicitly asked to omit them.

Michael W J Carmichael
Hanns Cottage, High St
Broughton, Hampshire
S021 8AE
GREAT BRITAIN

I was intrigued by the proposal to make "the world's least expensive plotter" at the end of Robert D Grappel's interesting article ("Give Your Micro Some Muscles," March 1977 BYTE, page 35). Eliminating the conventional XY carriage mechanism and using model aircraft servos looks like a good start.

May I put forward a somewhat different mechanical arrangement which avoids the two problems you describe in connection with your design, viz the weight of servo 2 sitting on the end of servo 1 arm, and the complexity of the trigonometrical equations requiring solution?

In figure 1 may be seen my alternative set-up in which the two servos are both fixed to ground and arranged concentrically. The inputs θ and Φ, rotations about the common pivot O, control the XY position of the pen. The moving links comprise a 4 bar linkage which has been dimensioned to approximately satisfy the following conditions:

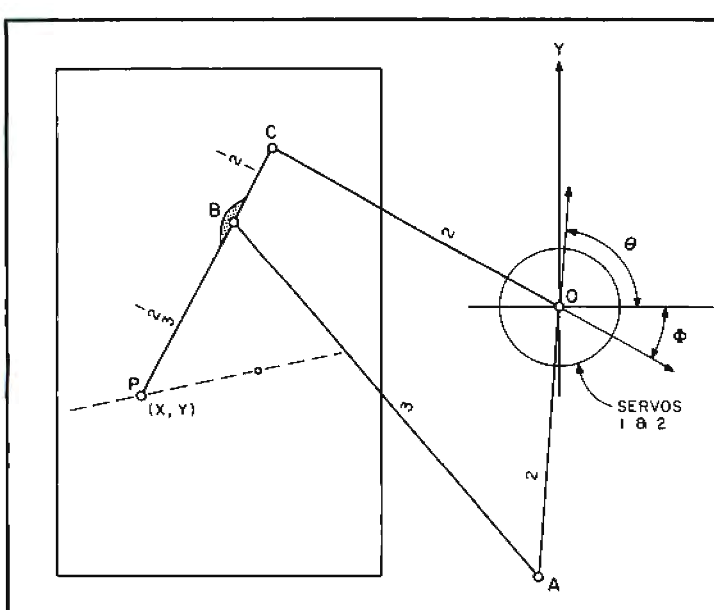
- When OA is fixed the point P traces a straight line locus.
- When produced this line passes through point O.
- The velocity ratio between the angular displacement of the link OC and the rectilinear displacement of the point P is constant.

The upshot of all this is that we now, in place of an XY plotter, have an Rθ plotter! The radial distance of P from the origin O is proportional to the angle between the links OA and OC ($\theta + \Phi$). The polar angle is simply proportional to the angular displacement θ of link OA. It is now a straightforward matter to express X and Y in terms of θ and Φ.

The professional may quibble about geometric distortion, but the homebrew man will discover an acceptable working range. I now confess that I have not actually built such a plotter myself. I am hoping someone else will be good enough to pursue the idea for me!

By the way I do enjoy your publication; it is most readable.■

Figure 1:



The suggestion you make is very attractive, especially when you consider the niceness of having the servos on concentric shafts. It is not clear, however, what the optimal geometry is for such a plotter. In figures 2a, 2b and 2c we did a paper and pencil exercise to look at a particular geometry chosen by trial and error among about three different extremes. In this geometry, we used a symmetrical quadrilateral for which dimensions of X and 2X were used, with the extension arm to the drawing point P located at a distance X from point B. (Subscript notation on the points is used to identify cases.)

In figure 2a, we've drawn this geometry in a case which looks very promising. For three different values of the angular parameter corresponding to Φ in figure 1, the construction produces an apparently straight line with θ held fixed at some value. However, in cases of figure 2b and 2c, with θ changed in either direction from its value in figure 2a, there is an apparent

Figure 2a:

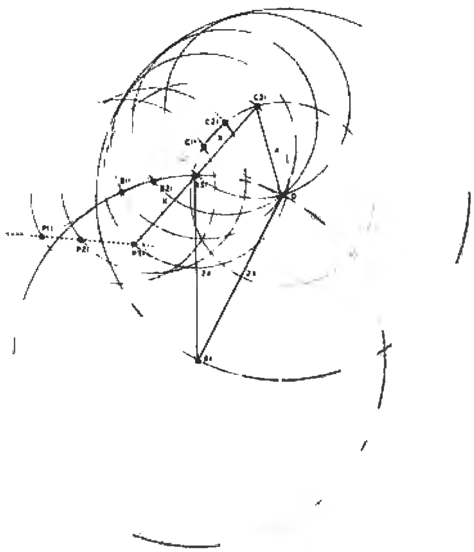
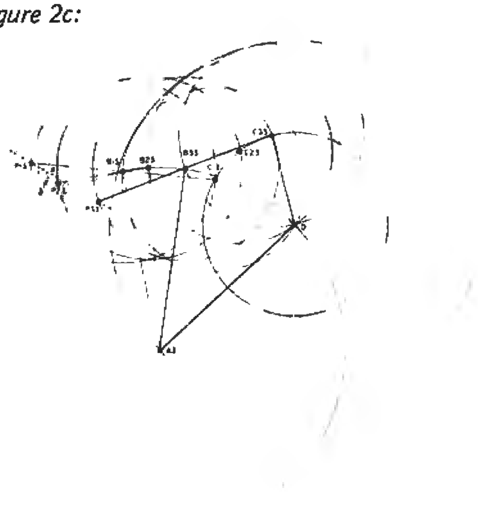


Figure 2b:

Figure 2c:



deviation δ from a straight line path for point P thru several states of angle Φ. Still, the construction looks quite promising if the problem can be set up analytically and parameters optimized for the smallest deviations as the angles are changed.

*Another point worth mentioning: the useful range of the angles in this type of construction is very small. In going from the example of figure 2b to figure 2c, the change in angle θ is approximately 45 degrees. This small angular change would require gearing to spread the useful control range of a model aircraft servo (180 degrees) over a smaller angular displacement. Note also that in our choice of parameters for this example, the locus of points P does not intersect the origin when extended as a straight line. If the constraint that distance PC equal distance OC is observed, this will guarantee that P intersects the origin. However, this criterion only guarantees that the locus of all points reachable by the moving point P will pass through the origin, and experiments with a compass and straightedge will verify that this locus is not necessarily a straight line through the origin, but is an arc of a curve. . .*CH

Othello, a New Ancient Game

Richard O Duda
590 Vine St
Menlo Park CA 94025

Othello is a 2 person board game based on the 100 year old game of Reversi. It resembles GO, but is much faster paced and easier to learn. If you are not already an Othello buff, this program will both teach you the game and provide an entertaining addition to your games library.

The game is normally played on an 8 by 8 checkerboard with pieces that are black on one side and white on the other. The game begins with four pieces making a checkerboard pattern in the center of the board as shown in figure 1. When it is your turn, you try to convert a run of your opponent's pieces, vertical, horizontal or diagonal, to your color. You can do this if you place your piece so that each end of the run is bounded by pieces of your color. Part of the action comes from the fact that you must either make such a conversion or forfeit your turn. The game is over after 60 moves or when both players must forfeit their moves.

```
(a)   A B C D E F G H     (b)   A B C D E F G H
  1   . . . . . . . .       1   X X X X X X X O
  2   . . . . . . . .       2   X X X O X X X O
  3   . . . . . . . .       3   X X X X X X X O
  4   . . . O X . . .       4   X X X X O X X O
  5   . . . X O . . .       5   X X X O X X X O
  6   . . . . . . . .       6   X X O O O X X O
  7   . . . . . . . .       7   X X X O X X X X
  8   . . . . . . . .       8   O O O O O O O X
```

Figure 1: These two example boards show the Othello game board at the beginning and end of a typical game. The game is always initialized to the configuration in figure 1a. Figure 1b shows the board at the end of a typical game. This game was played to completion with 60 moves. The computer was soundly beaten having only 20 pieces to the human player's 44.

The program has two strategies it can deploy against you, one elementary and the other simple. If losing to a computer would be too great a blow to your self-esteem, request the elementary strategy. After your confidence is fairly secure, you can risk a no holds barred encounter. Eventually you will tire of your prowess, and will want to modify the program to improve its play. Hopefully, the comments in the program will make its operation transparent and will invite modifications.

A few words about modifications that may be required to play at all. A random tie breaking strategy is used to provide variety, but it is not essential. If your BASIC does not provide random numbers, you can delete lines 1310 and 1320 with no significant change in performance.

At the outset the program asks if you want it to pause before making its move. This pause is most helpful with a display terminal, since it keeps the output from flashing by too quickly. The output of the board is done by the last subroutine in the program. It requires the ability to display at least 11 lines of 19 characters, and you will probably want to modify the program if your TVT cannot handle the 26 lines of 32 characters needed to see two boards and some accompanying output simultaneously.

Of course, if you use a relatively slow hard copy printer with this program, it will take a long time to type out all the boards. The obvious solution is to suppress this printout except on request, and use an actual board to record the positions of pieces for your own use.

Good luck, and remember: Othello was so named because it is a game of dramatic reversals.

Listing 1: A BASIC listing for Othello.

```
00010 REM   OTHELLO                    AUTHOR:  RICHARD O. DUDA
00020 REM   PLAYS THE GAME "OTHELLO" WITH TWO STRATEGIES:
00030 REM     1.  TAKE THE MAXIMUM NUMBER OF PIECES
00040 REM     2.  ADD A BONUS FOR OUTSIDE POSITION
00050 REM   BOARD IS THE ARRAY A, BOUNDED BY 0'S (BLANKS)
00060 REM    A = 0 FOR EMPTY SQUARE
00070 REM    A = B FOR BLACK SQUARE -- X  (INTERNALLY -1)
00080 REM    A = W FOR WHITE SQUARE -- O  (INTERNALLY +1)
00090 REM   I AND J ALWAYS USED FOR ROW/COLUMN INDICES
00100 REM   I4 AND J4 STORE INCREMENTS TO THE 8 NEIGHBORS
00110 REM   C$ AND D$ STORE CHARACTERS A-H,X,.,O FOR OUTPUT
00120 DIM A(9,9),I4(8),J4(8),C$(8),D$(2)
00130 REM   INITIAL GREETING
00140 PRINT "GREETINGS FROM OTHELLO"
00150 PRINT "DO YOU WANT INSTRUCTIONS (Y OR N) ";
00160 INPUT X$
00170 IF X$ = "N" THEN 390
00180 IF X$ <> "Y" THEN 160
00190 PRINT
00200 PRINT "OTHELLO IS PLAYED ON AN 8 X 8 CHECKER BOARD,"
00210 PRINT "ROWS NUMBERED 1 TO 8 AND COLUMNS A TO H."
00220 PRINT "THE INITIAL CONFIGURATION IS ALL BLANK, EXCEPT"
00230 PRINT "FOR THE CENTER FOUR SQUARES, WHICH FORM THE"
00240 PRINT "PATTERN"
00250 PRINT "              O X"
00260 PRINT "              X O"
00270 PRINT
00280 PRINT "TRY TO PLACE YOUR PIECE SO THAT IT 'OUTFLANKS'"
00290 PRINT "MINE, CREATING A HORIZONTAL, VERTICAL, OR"
00300 PRINT "DIAGONAL RUN OF MY PIECES BOUNDED AT EACH END"
00310 PRINT "BY AT LEAST ONE OF YOURS.  THIS WILL 'FLIP' MY"
00320 PRINT "PIECES, TURNING THEM INTO YOURS."
00330 PRINT "NOTE: YOU MUST CAPTURE AT LEAST ONE OF MY"
00340 PRINT "PIECES IN THIS WAY IF IT IS AT ALL POSSIBLE."
00350 PRINT "IF IT IS NOT POSSIBLE, YOU FORFEIT YOUR TURN BY"
00360 PRINT "ENTERING 0,0 FOR YOUR (ROW,COL) MOVE."
00370 PRINT
00380 REM   INITIALIZE
00390 PRINT "SHOULD I WAIT BEFORE MAKING MY MOVES (Y OR N) ";
00400 F2 = 0
00410 INPUT X$
00420 IF X$ = "N" THEN 460
00430 IF X$ <> "Y" THEN 410
00440 F2 = 1
00450 PRINT "OK.  TYPING ANY CHARACTER WILL LET ME GO."
00460 PRINT "SHOULD I PLAY MY BEST STRATEGY (Y OR N) ";
00470 S2 = 0
00480 INPUT X$
00490 IF X$ = "N" THEN 520
00500 IF X$ <> "Y" THEN 480
00510 S2 = 2
00520 B = -1
00530 W = +1
00540 D$(B + 1) = "X"
00550 D$(0 + 1) = "."
00560 D$(W + 1) = "O"
00570 FOR K = 1 TO 8
00580 READ I4(K)
00590 NEXT K
00600 DATA 0,-1,-1,-1,0,1,1,1
00610 FOR K = 1 TO 8
00620 READ J4(K)
00630 NEXT K
00640 DATA 1,1,0,-1,-1,-1,0,1
00650 FOR K = 1 TO 8
00660 READ C$(K)
00670 NEXT K
00680 DATA A,B,C,D,E,F,G,H
00690 REM   SET UP A NEW GAME
00700 FOR I = 0 TO 9
00710 FOR J = 0 TO 9
00720 A(I,J) = 0
00730 NEXT J
00740 NEXT I
00750 A(4,4) = W
00760 A(5,5) = W
00770 A(4,5) = B
00780 A(5,4) = B
00790 C1 = 2
00800 H1 = 2
00810 N1 = 4
00820 Z = 0
00830 REM   HUMAN'S CHOICES
00840 PRINT "DO YOU WANT TO HAVE X OR O ";
00850 C = W
00860 H = B
00870 INPUT X$
00880 IF X$ = "X" THEN 920
00890 IF X$ <> "O" THEN 870
00900 C = B
00910 H = W
00920 PRINT "DO YOU WANT TO GO FIRST (Y OR N) ";
00930 INPUT X$
```



```
00940 IF X$ = "N" THEN 1020
00950 IF X$ <> "Y" THEN 930
00960 REM   PRINT INITIAL BOARD
00970 GOSUB 3100
00980 GO TO 1690
00990 REM   COMPUTER'S MOVE
01000 IF F2 = 0 THEN 1020
01010 INPUT X$
01020 B1 = -1
01030 I3 = J3 = 0
01040 T1 = C
01050 T2 = H
01060 REM   SCAN FOR BLANK SQUARE
01070 FOR I = 1 TO 8
01080 FOR J = 1 TO 8
01090 IF A(I,J) <> 0 THEN 1380
01100 REM   FOUND A BLANK SQUARE
01110 REM   DOES IT HAVE AN OPPONENT AS A NEIGHBOR?
01120 GOSUB 2620
01130 IF F1 = 0 THEN 1380
01140 REM   FOUND AN OPPONENT AS A NEIGHBOR
01150 REM   HOW MANY OF HIS PIECES CAN WE FLIP?
01160 REM   (DON'T DO IT NOW)
01170 U = -1
01180 GOSUB 2820
01190 REM   EXTRA POINTS FOR BOUNDARY POSITION
01200 IF S1 = 0 THEN 1380
01210 IF (I - 1) * (I - 8) <> 0 THEN 1230
01220 S1 = S1 + S2
01230 IF (J - 1) * (J - 8) <> 0 THEN 1260
01240 S1 = S1 + S2
01250 REM   IS THIS BETTER THAN THE BEST FOUND SO FAR?
01260 IF S1 < B1 THEN 1380
01270 IF S1 > B1 THEN 1340
01280 REM   A TIE; RANDOM DECISION
01290 REM   THE NEXT TWO EXECUTABLE STATEMENTS CAN BE DELETED
01300 REM   FOR A VERSION OF BASIC WITHOUT RANDOM NUMBERS
01310 R = RND(1)
01320 IF R > 0.5 THEN 1380
01330 REM   YES
01340 B1 = S1
01350 I3 = I
01360 J3 = J
```

Listing 1, continued:

```
01370 REM   END OF SCAN LOOP
01380 NEXT J
01390 NEXT I
01400 REM   COULD WE DO ANYTHING?
01410 IF B1 > 0 THEN 01480
01420 REM   NO
01430 PRINT "I HAVE TO FORFEIT MY MOVE"
01440 IF Z = 1 THEN 2190
01450 Z = 1
01460 GO TO 1690
01470 REM   MAKE THE MOVE
01480 Z = 0
01490 PRINT "I WILL MOVE TO ";
01500 PRINT I3;
01510 PRINT ", ";
01520 PRINT C$(J3)
01530 I = I3
01540 J = J3
01550 U = 1
01560 GOSUB 2820
01570 C1 = C1 + S1 + 1
01580 H1 = H1 - S1
01590 N1 = N1 + 1
01600 PRINT "THAT GIVES ME ";
01610 PRINT S1;
01620 PRINT " OF YOUR PIECES"
01630 REM   PRINT OUT BOARD
01640 GOSUB 3100
01650 REM   TEST FOR END OF GAME
01660 IF H1 = 0 THEN 2190
01670 IF N1 = 64 THEN 2190
01680 REM   HUMAN'S MOVE
01690 T1 = H
01700 T2 = C
01710 PRINT "YOUR MOVE -- (ROW, COL) ";
01720 INPUT I, X$
01730 IF I < 0 THEN 1720
01740 IF I > 8 THEN 1720
01750 IF I <> 0 THEN 1820
01760 PRINT "ARE YOU FORFEITING YOUR TURN (Y OR N) "
01770 INPUT X$
01780 IF X$ <> "Y" THEN 1710
01790 IF Z = 1 THEN 2190
01800 Z = 1
01810 GO TO 1000
01820 FOR J = 1 TO 8
01830 IF C$(J) = X$ THEN 1870
01840 NEXT J
01850 GO TO 1720
01860 REM   CHECK IF BLANK
01870 IF A(I,J) = 0 THEN 1910
01880 PRINT "SORRY, THAT SQUARE IS OCCUPIED; TRY AGAIN"
01890 GO TO 1720
01900 REM   CHECK FOR LEGAL NEIGHBOR
01910 GOSUB 2620
01920 IF F1 = 1 THEN 1970
01930 PRINT "SORRY, YOU ARE NOT NEXT TO ONE OF MY PIECES;"
01940 PRINT "TRY AGAIN"
01950 GO TO 1720
01960 REM   CHECK IF LEGAL RUN
01970 U = -1
01980 GOSUB 2820
01990 IF S1 > 0 THEN 2030
02000 PRINT "SORRY, THAT DOESN'T FLANK A ROW; TRY AGAIN"
02010 GO TO 1720
02020 REM   EVERYTHING LEGAL; MAKE HUMAN'S MOVE
02030 Z = 0
02040 PRINT "THAT GIVES YOU ";
02050 PRINT S1;
02060 PRINT " OF MY PIECES"
02070 U = 1
02080 GOSUB 2820
02090 H1 = H1 + S1 + 1
02100 C1 = C1 - S1
02110 N1 = N1 + 1
02120 REM   PRINT OUT BOARD
02130 GOSUB 3100
02140 REM TEST FOR END OF GAME
02150 IF C1 = 0 THEN 2190
02160 IF N1 = 64 THEN 2190
02170 GO TO 1000
02180 REM   END OF GAME WRAPUP
02190 PRINT
02200 PRINT "YOU HAVE ";
02210 PRINT H1;
02220 PRINT " PIECES AND I HAVE ";
02230 PRINT C1;
02240 PRINT " PIECES -- ";
02250 IF H1 = C1 THEN 2290
02260 IF H1 > C1 THEN 2310
02270 PRINT "SORRY, I WON THAT ONE."
02280 GO TO 2320
02290 PRINT "A TIE !!"
02300 GO TO 2500
02310 PRINT "YOU WON !"
02320 C1 = C1 - H1
02330 IF C1 > 0 THEN 2350
02340 C1 = -C1
02350 C1 = (64 * C1) / N1
02360 PRINT "THAT WAS A ";
02370 IF C1 < 11 THEN 2490
02380 IF C1 < 25 THEN 2470
02390 IF C1 < 39 THEN 2450
02400 IF C1 < 53 THEN 2430
02410 PRINT "PERFECT GAME."
02420 GO TO 2500
02430 PRINT "WALKAWAY."
02440 GO TO 2500
02450 PRINT "FIGHT."
02460 GO TO 2500
02470 PRINT "HOT GAME !"
02480 GO TO 2500
02490 PRINT "SQUEAKER !!"
02500 PRINT
02510 PRINT "DO YOU WANT TO PLAY ANOTHER GAME (Y OR N) ";
02520 INPUT X$
02530 IF X$ = "Y" THEN 700
02540 IF X$ <> "N" THEN 2520
02550 PRINT "THANKS FOR PLAYING."
02560 STOP
02570 REM
02580 REM   SUBROUTINE TEST-FOR-PROPER-NEIGHBOR
02590 REM   ASSUMES:
02600 REM         I,J LOCATES A BLANK SQUARE
02610 REM         YOU HOPE TO SEE AN ADJACENT T2 (= -T1)
02620 FOR I1 = -1 TO 1
02630 FOR J1 = -1 TO 1
02640 IF A(I+I1,J+J1) = T2 THEN 2710
02650 NEXT J1
02660 NEXT I1
02670 REM   NO T2 FOUND; FAILURE
02680 F1 = 0
02690 RETURN
02700 REM   SUCCESS
02710 F1 = 1
02720 RETURN
02730 REM   SUBROUTINE SCORE-AND-UPDATE
02740 REM   ASSUMES:
02750 REM      (I,J) IS A TENTATIVE PLACE FOR A PIECE T1.
02760 REM      WANT RUNS OF T2 = -T1, TERMINATED BY A T1.
02770 REM      IF U IS TRUE (1), MARK THOSE RUNS AS T1'S.
02780 REM      RETURN SUM OF ALL RUNS (T2'S ONLY) IN S1.
02790 REM      MAIN PROGRAM CONTAINS THE FOLLOWING ARRAYS:
02800 REM          I4:   0 -1 -1 -1  0  1  1  1
02810 REM          J4:   1  1  0 -1 -1 -1  0  1
02820 S1 = 0
02830 FOR K = 1 TO 8
02840 I5 = I4(K)
02850 J5 = J4(K)
02860 I6 = I + I5
02870 J6 = J + J5
02880 S3 = 0
02890 IF A(I6,J6) <> T2 THEN 3070
02900 REM   LOOP THROUGH THE RUN
02910 S3 = S3 + 1
02920 I6 = I6 + I5
02930 J6 = J6 + J5
02940 IF A(I6,J6) = T1 THEN 2970
02950 IF A(I6,J6) = 0 THEN 3070
02960 GO TO 2910
02970 S1 = S1 + S3
02980 IF U <> 1 THEN 3070
02990 REM   UPDATE BOARD
03000 I6 = I
03010 J6 = J
03020 FOR K1 = 0 TO S3
03030 A(I6,J6) = T1
03040 I6 = I6 + I5
03050 J6 = J6 + J5
03060 NEXT K1
03070 NEXT K
03080 RETURN
03090 REM   SUBROUTINE PRINT-BOARD
03100 PRINT
03110 PRINT "     A B C D E F G H"
03120 FOR I = 1 TO 8
03130 PRINT I;
03140 FOR J = 1 TO 8
03150 PRINT " ";
03160 PRINT D$(A(I,J)+1);
03170 NEXT J
03180 PRINT
03190 NEXT I
03200 PRINT
03210 RETURN
03220 END ■
```

An APL Interpreter for Microcomputers

Part 3: Mathematical Processing

Mike Wimble
6026 Underwood Av SW
Cedar Rapids IA 52404

Introduction

In part 2 of this 3 part series (see last month's issue) I covered the expression evaluation portion of an APL interpreter. This month's concluding installment deals with mathematical processing. Mathematical processing is a 2 part function: recognition followed by interpretation. Both of these topics are covered here, followed by a short summary.

The Great APL Interpreter Contest

As an incentive to those experimenters who would like to try writing their own APL interpreters based on this series of articles, BYTE announces the Great APL Interpreter Contest. We will award prizes for APL interpreters (suitable for publication with royalties to authors) based on Mike's flowcharts (or independent of them if you prefer).

Contestants are free to write their interpreters for any microprocessor they choose. Entries will, however, be judged on their suitability for use on small systems with a minimum of 16 K bytes of memory, as well as on programming elegance and efficient use of space. All of these factors should therefore be kept in mind.

Entries should be addressed to BYTE, attn: The Great APL Interpreter Contest, 70 Main St, Peterborough NH 03458, and must be postmarked no later than midnight, February 28 1978. Entries must be in the form of a publication quality manuscript which describes the implementation of the interpreter and which includes a listing of source code and object code. Contestants should also submit machine readable source and object code in the form of paper tape or cassette.

The winners (if any) will receive $1000 plus normal author payments, should the entry be chosen for publication in book form or as an article in BYTE. We reserve the right to choose more than one winner under the same terms.

Judging will be done by the editors of this magazine. Those seriously interested in entering this contest should call Carl Helmers or Chris Morgan at BYTE, (603)924-7217. May the midnight oil burn prosperously for all.

Recognition of a Valid Operator

The recognition process involves testing the current syllable in table SP for an appropriate noun. If the noun is found, then CODE will be set to reflect the appropriate specified operation. Figure 29 shows this process clearly for the recognition of monadic operators. The recognition of dyadic operators (see figure 30) is a little more involved, but the principle is the same. The extra processing involved is merely to recognize the two possible special forms of a dyadic operator: the inner product and the outer product.

CALL MOP(B)
B set true if a monadic operator found.
B set false otherwise.

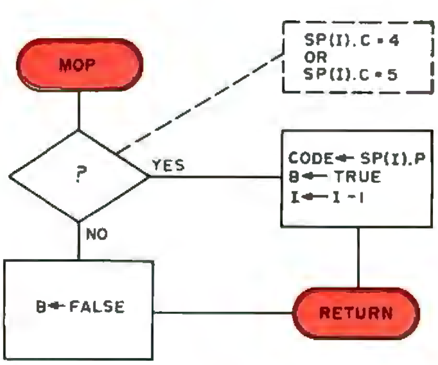


Figure 29: Recognition of a monadic operator.

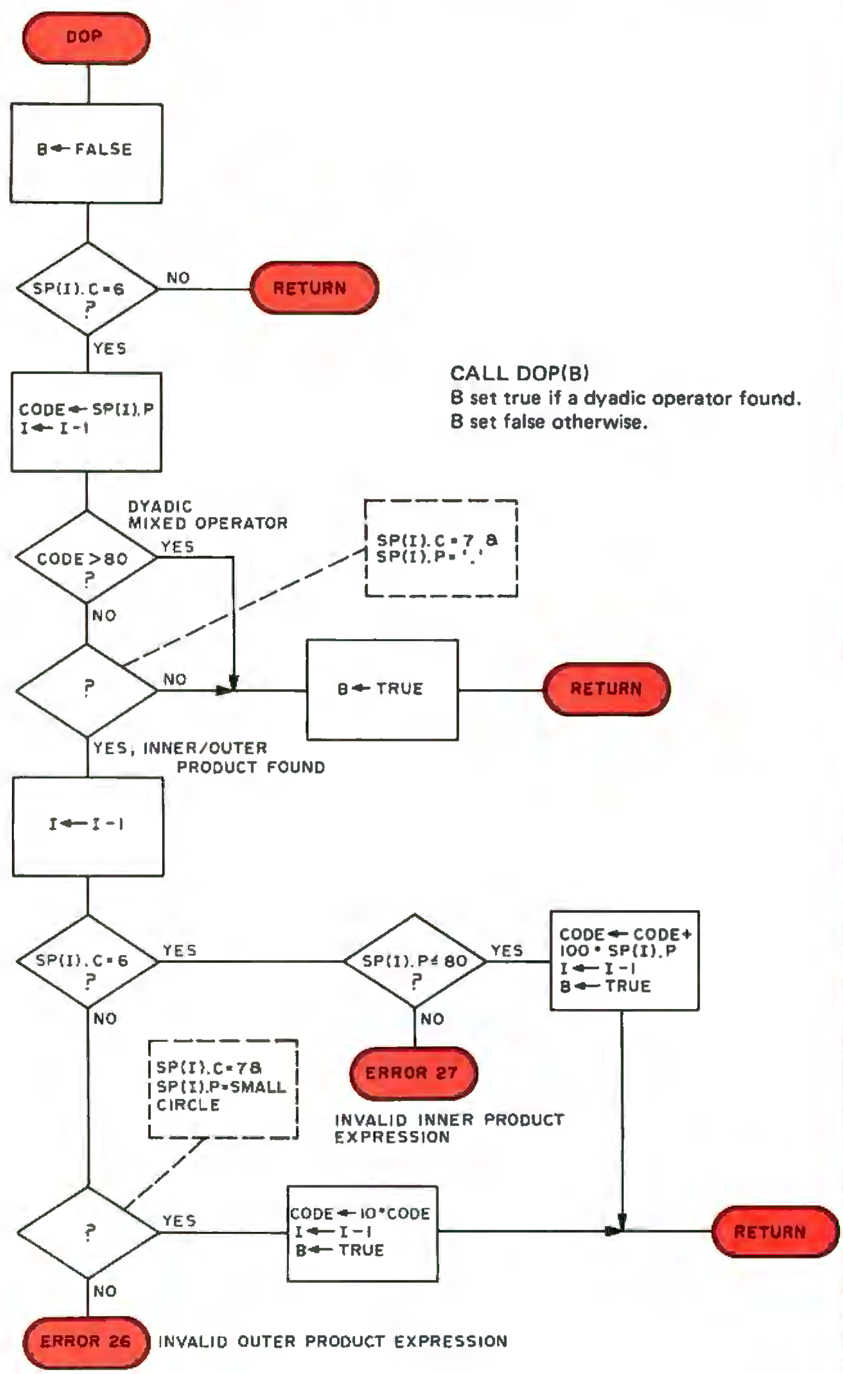
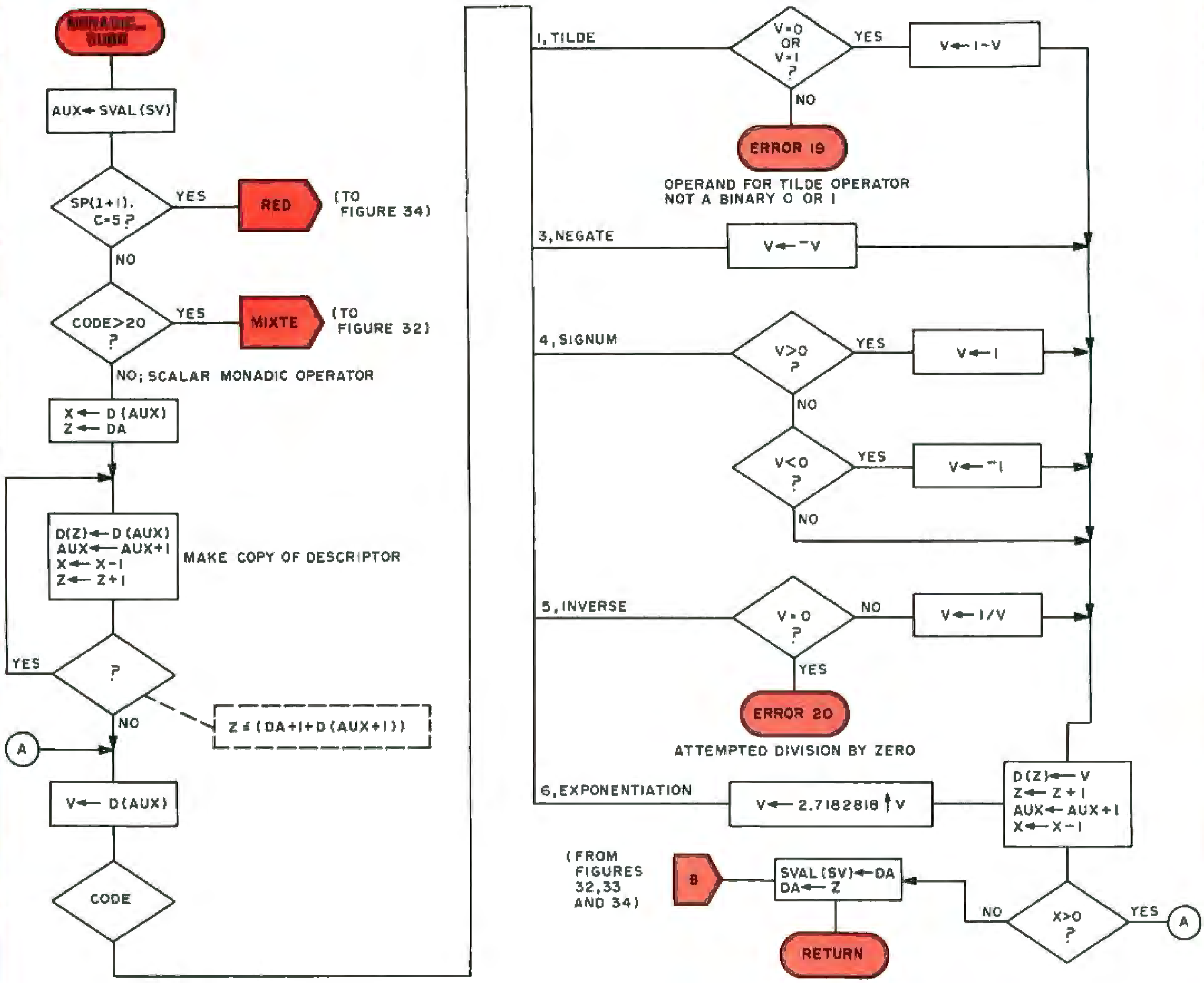


Figure 30: Recognition of a dyadic operator.

Figure 31: Interpretation of a monadic operator.



Interpretation of Monadic Operators

After a valid operator is recognized, it is interpreted. Interpretation involves unstacking pointers from table SVAL, which points to the operands; performing the operation to create a result in table D; and finally stacking a pointer to the result back in table SVAL. Figure 31 begins the description of monadic interpretation. Generally, a monadic operator generates a result having the same shape as its operand; in two cases, however, this is not true. The process of reduction produces a result with one less dimension than the operand. Also, the monadic operators iota, rho, and ravel may produce results having greater or fewer dimensions than their corresponding operands.

So, in figure 31, after saving a pointer to the argument for the monadic operator, a test is made for a reduction or mixed operator and the appropriate process is executed. The interpretation of nonspecial monadic operators is then straightforward. First, since the result will have the same shape as the argument, it is given the same descriptor as the argument. Next, the monadic operator is performed on each value in the argument (or on the single value for scalar argument) to produce each value in the result. Finally, a pointer to the result is stacked in table SVAL and index DA is updated to point to the next free space in table D.

GET THE BEST FROM... SEALS

MOST ORDERS SHIPPED WITHIN 10 WORKING DAYS

ORDER FORM [Write in each box the quantity of each part required]

KIT		ASSEMBLED	KIT		ASSEMBLED	KIT		ASSEMBLED
☐	4KROM	☐	☐	WWC	☐	☐	68EXT-L	☐
☐	8KSC-Z	☐	☐	88 EXT	☐	☐	68WWC	☐
☐	68KSC	☐	☐	68EXT-S	☐	☐	BBUC	☐
☐	8KSC	☐						

☐ ASSEMBLY AND OPERATING MANUAL $4.00

B-10-77

NAME ______________________________

PLEASE PRINT OR TYPE

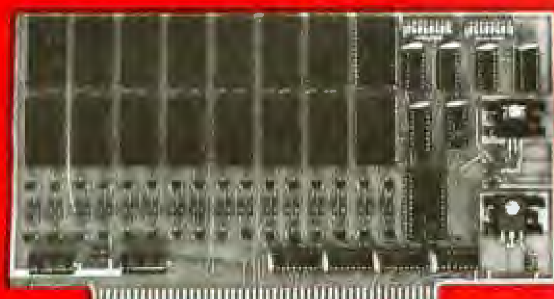
ADDRESS ______________________________

CITY ______________ STATE ________ ZIP ________

SEND CHECK • MONEY ORDER • COD'S ACCEPTED • CREDIT CARDS

SEALS ELECTRONICS, INC
P.O. BOX 11651
KNOXVILLE, TN 37919

EXPORT INFORMATION
Canadian and European community customers use our EXPORT DELIVERY SERVICE. Guaranteed delivery to your address. We take care of charges for customs clearance, duties, federal taxes and insurance. Add exactly 35% to the prices shown (45% in Canadian dollars). Allow 5 extra days for delivery. Minimum export order $100.00. DEALER INQUIRIES WELCOME.



4KROM (READ ONLY MEMORY)

- Will accept up to 16 ea. 1702-A or 5203 EProm Providing up to 4096 words of non-volatile memory for Boot Loads to Complete Programs.
- Programming Available at Factory for $3.00 per EProm when accompanied by binary formatted tape.
- Each 1702-A has its own Vgg clocked for Low Power Consumption.
- Will work with the weakest power supply based S-100 buss computer.
- Switch Selected Address in 4K Blocks.
- Switch selected wait states so that even the slowest 1702-A can work in your system. 0-8 wait states.
- Solder Masked on both sides of PC Board.
- Component Legend on Component Side of PC Board.

Kit Price. . . $109.00 Assembled Price. . . $119.00

68KSC (STATIC MEMORY CARD FOR SWTPC 6800)

- 8192 Words of Static Memory
- Access Time: 500 nsec. (250 nsec. on request)
- Memory Chip 91L02 APC or 2102AL-4
- Battery Standby
- Address Selected 8 Ea. SPST Dip Switch
- Low Power
- All Lines Buffered
- All IC's with sockets
- Solder Masked on both sides of PC Board

Kit Price. . . $259.00 Assembled Price. . . $268.00

8KSC-Z (STATIC MEMORY CARD)

- ALTAIR®, IMSAI® and S-100 buss compatible
- Access Time: 250 nsec max.
- Zilog speed Compatible up to 4 mhz.
- Memory Chip: 2102L-HPC or 2102AL-4
- Battery Standby: [illegible] to 4 volts
- Address Select: 8 ea. Spst. Dip Switch
- Wait States: None
- Current Req.: Less than 2.0 ma per 1K
- All Address, Control, and Data out lines fully buffered
- All IC's supplied with IC Sockets
- Solder Masked on Both Front and Back of P.C. Board

Kit Price. . . $285.00 Assembled Price. . . $295.00

Circle [illegible] on inquiry card

SEALS ASSEMBLED AND TESTED PRODUCTS AT KIT PRICES

Unit No.	Description	Kit Price	Assembled Price
[illegible]	[illegible] Static Memory Card [illegible]	$ 269.00	$ 269.00
WWC	Wire Wrap Card	$ 37.00	$ 47.50
88 EXT	88 Extender Card	$ 29.00	$ 38.00
68 EXT-S	Extender Card (Small)	$ 19.00	$ 25.00
68 EXT-L	Extender Card (Large)	$ 29.00	$ 39.00
68 WWC	Wire Wrap Card	$ 35.00	$ 45.00
BBUC	Battery Back Up Card	$ 65.00	$ 69.00

SEALS ELECTRONICS, INC.

ORDER DELIVERY: FROM STOCK TO 10 DAYS

TELEX NO. 55-7444 TELEPHONE NO. 615/693-8655

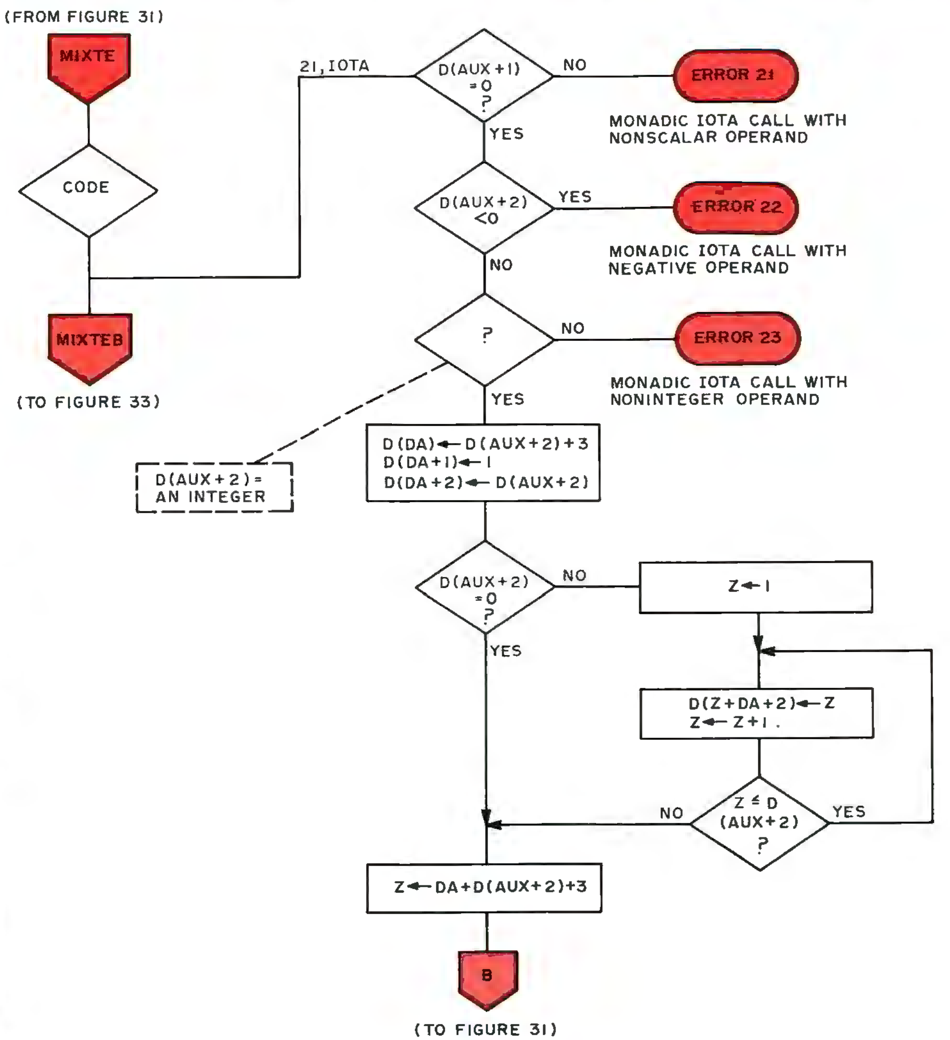


Figure 32: Interpretation of a mixed monadic operator.

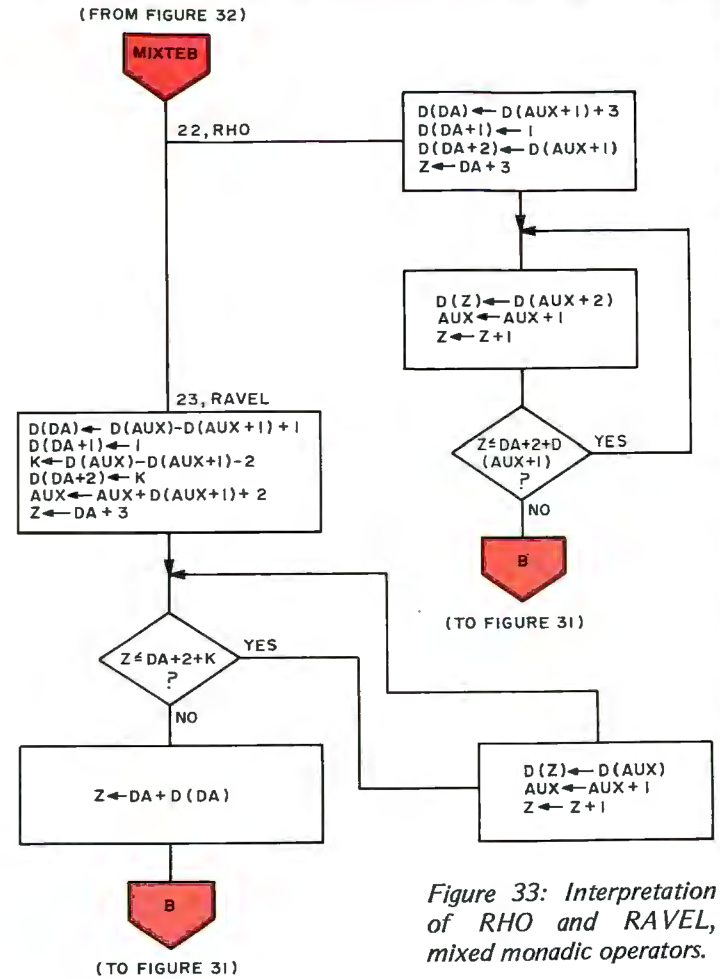


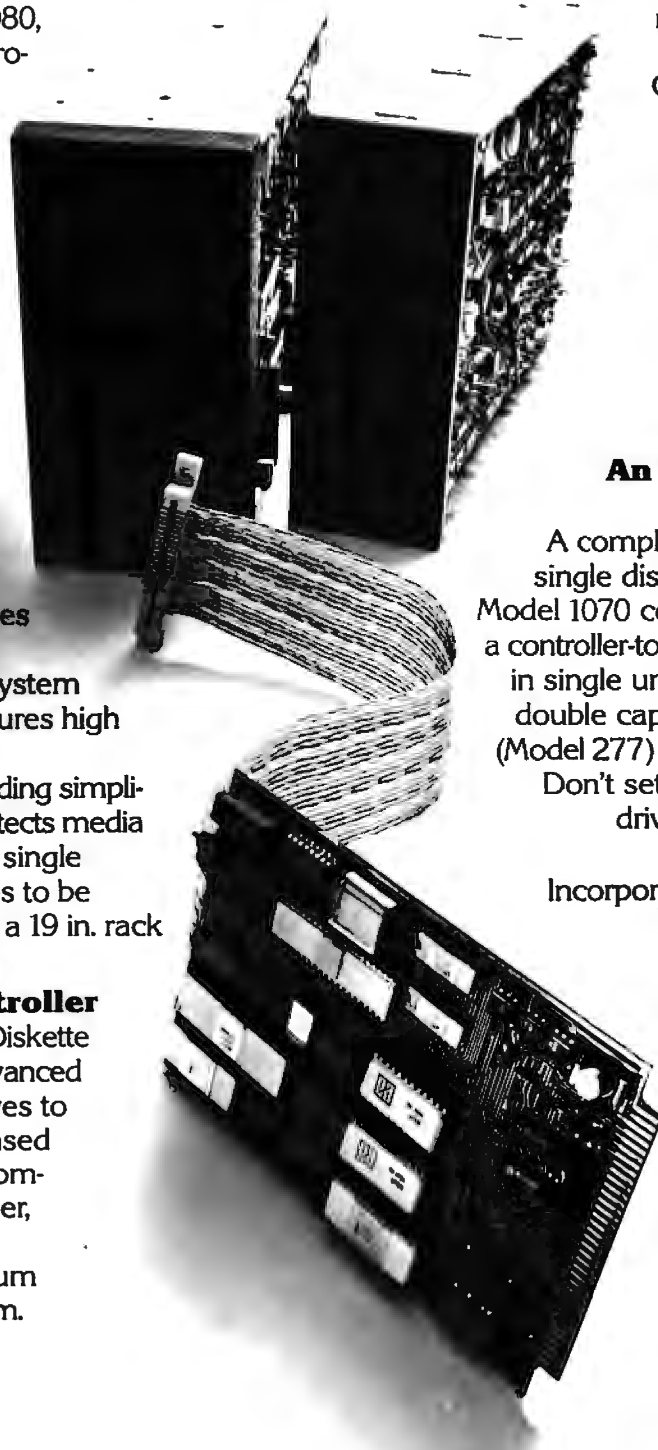
Figure 33: Interpretation of RHO and RAVEL, mixed monadic operators.

Interpretation of mixed mode monadic operators, as shown in figures 32 and 33, is a little more heuristic. For the iota operator the argument must be a nonnegative integer scalar and the result will be a vector. The operators rho and ravel take any form of argument and produce vector results. Again, after creation of the result in table D a pointer to the result is stacked in table SVAL and index DA is updated.

Continued on page 78

The new generation of Diskette Drives is here and under control.

PerSci has it—a family of diskette drives "design-years" ahead of competitive drives—now available in complete low cost subsystems for interface to 8080, 6800 and other major microprocessors.



The Highest Performance Diskette Drives:

PerSci diskette drives, both single and dual head units, offer a combination of performance features unique in the marketplace while still maintaining compatibility in existing systems:

- ☐ Voice coil positioning for access speeds seven times faster than competitive drives (76 tracks in 100 ms)
- ☐ A low power all DC system reduces cost and assures high reliability
- ☐ Automatic electric loading simplifies operation and protects media
- ☐ Small size permits 5 single drives or 4 dual drives to be mounted vertically in a 19 in. rack

The Most Powerful Diskette Drive Controller

The PerSci Model 1070 Diskette Drive Controller puts the advanced performance of PerSci drives to work in microprocessor based systems. An IBM format compatible, "intelligent" controller, the Model 1070 will handle from 1-4 drives with minimum demand on the host system.

In fact, with addition of a power supply and keyboard to the PerSci subsystem, the user can perform many floppy disk routines without additional hardware or software.

Controller features include:

- ☐ 8080 microprocessor
- ☐ Formatter/control IC
- ☐ 4K disk operating system
- ☐ 1K data buffer
- ☐ 8080 or 6800 microprocessor interfaces
- ☐ RS-232 serial interface optional
- ☐ IBM data format

An Economical Diskette Drive Subsystem

A complete subsystem including a single diskette drive (Model 70), the Model 1070 controller with interface and a controller-to-disk-drive cable is available in single units or OEM quantities. For double capacity, a dual diskette drive (Model 277) subsystem is also available.

Don't settle for yesterday's diskette drive. Get the new generation under control from PerSci, Incorporated, 12210 Nebraska Ave. W. Los Angeles, CA 90025 (213) 820-3764

PERSCI

Peripherals a Generation Ahead.

Circle 361 on inquiry card.

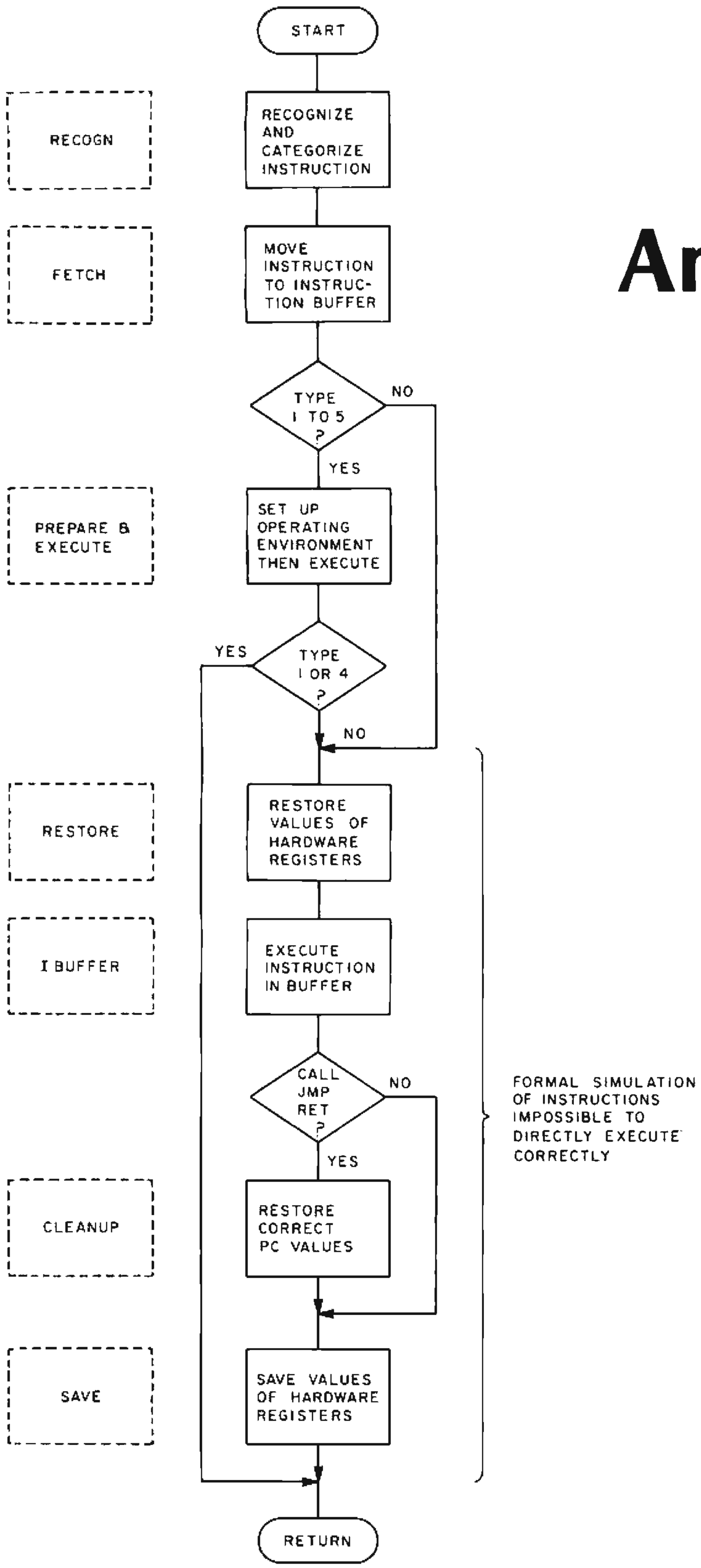


Figure 1: Basic flowchart for the simulator. Each of the square blocks constitutes a separate subroutine.

An 8080 Simulator

A simulator can be very useful to the personal computing experimenter for a number of reasons. The idea of a simulator is to perfectly duplicate the operation of a certain computing system using the instruction set of another computer. For example, suppose you have an 8080 program which you want to run on your 6800 system rather than completely write a new program. Since the simulator acts functionally like an 8080 processor to interpret the 8080 object code bit patterns, its hardware registers and program counter, it is possible for the 6800 system to run the 8080 program. Another attraction of a simulator is that in software it is possible to achieve a number of debugging and checkout functions as a part of the simulator itself, since the machine being run is always under total software control. When a simulator for machine X is run on machine X, these attractions can justify the use of the simulator.

A simulator usually has two drawbacks however: memory requirements and speed reduction. The speed reduction drawback is usually the more objectionable feature, although a fairly complex simulator can easily take up 8 K to 16 K bytes of memory space, depending on the complexity of the computer system being simulated. It is not unusual that thousands of instructions are executed before the functions of a single instruction can be simulated. This means that the simulator runs at a speed thousands of times slower than the actual computer does, which may be a difficulty in some cases.

Writing a simulator is a straightforward process as its functions are more or less well-defined by the documentation of the computer being simulated. All it has to do is to recognize the machine instruction, carry out its function, and record its effects. Of course you will have to keep track of the

Kin-man Chung
2114-204 S Orchard St
Urbana IL 61801

values of various registers, stack pointer, program counter, etc.

If, however, the computer being simulated is written on the same computer doing the simulation, eg: an 8080 simulator on an 8080 microcomputer or a Z-80 simulator on a Z-80, then a lot of work can be simplified. As a matter of fact, I have discovered a very neat trick for writing such a simulator that greatly reduces the program length and increases its speed. An 8080 simulator was written on my 8080 computer which uses about 350 bytes of memory. Using the simulator, a quite sophisticated debugging program was also written using another 1 K bytes of memory.

General Description

In a simulator the hardware registers, such as the program counter, stack pointer, index registers, conditional flags, etc, have to be simulated by memory locations. The instructions have to be recognized and executed, and the memory locations simulating the hardware registers have to be updated accordingly. Take as an example the INR A, increment accumulator, instruction in the 8080. The value 1 is added to the location simulating the accumulator, and the locations simulating conditional flags are updated according to whether the carry, sign bit, etc, are affected. This has to be done for all the instructions in the instruction set, using software. A major portion of the simulator program is actually "wasted" doing this kind of work. In my 8080 simulator, I still have to use memory locations to simulate hardware registers, but I do not simulate the instructions at all. How can this be done? The idea is actually quite simple. Instead of simulating the process of each instruction by elaborate software, I let the computer actually execute the instruction. Since the computer is simulating a computer whose language it speaks, this can always be done. Of course before we can do this, we have to fetch the instruction to be simulated into an instruction buffer, a section that is under simulator control. Under the correct operating environment, the simulator should remain in control right after executing the instruction in the instruction buffer. The various parts of the routine that simulate a single instruction cycle are illustrated in figure 1.

For those instructions that do not change the values of the program counter abruptly, there is no problem. After actually executing the instruction in the instruction buffer, the instruction next to it is the next instruction to be executed. The only problem which occurs is when there is a jump or call instruction. If this type instruction is actually executed in the instruction buffer, control would be forced to go to whatever location is specified in the instruction. The trick used is to change the call or jump operand of the instruction to a location under simulator control. The original address must be stored someplace, of course; more about this later.

There are five types of instruction that

Listing 1: The 8080 assembler listing for the 8080 simulator.

```
; *** 8080 SIMULATOR ROUTINE ***
SIM:    LHLD  PC          ; Get the value of PC.
        MOV   A,M         ; Load the instruction into acc.
        CALL  RECOGN      ; Call the recognizer.
        LHLD  NBYTE       ; Get number of byte and
        XCHG              ; load it into (D,E).
        LHLD  PC          ; Get the value of PC again
        DAD   D           ; and add these 2 num to get the next PC.
        SHLD  PC          ; Store it back at PC.
; *** FETCH ***
        MVI   D,4         ; Reg D, which keeps track of # byte to fill the
                          ; inst buff, is initially set to 4.
        LXI   B,IBUFF+2   ; (B,C) is set to the last byte of inst buff.
NXT:    DCX   H           ; (H,L) points to the last byte of the inst in user prog.
        MOV   A,M         ; Fetch the instruction from user program.
```

Listing 1, continued:

```
         STAX   B          ; and store at the inst buffer.
         DCX    B          ; starting from the last byte.
         DCR    D
         DCR    E          ; Rem: E contains the number of bytes of the inst.
         JNZ    NXT        ; Repeat until all bytes of the inst are moved.
         XRA    A          ; Set A=0, the NOP inst.
NXT2:    DCR    D          ; Decrement D.
         JZ     PREPARE    ; If inst buffer is filled, quit.
         STAX   B          ; else fill the rest of the inst buffer
         DCX    B          ; with NOP inst
         JMP    NXT2       ; until all done.
; *** PREPARE ***
PREPARE: LDA    TYPE       ; Now check type of inst for special preparation.
         DCR    A
         JNZ    NOT1       ; If it is not a type 1 inst, goto NOT1.
                           ; Type 1 – PCHL inst.
         LHLD   HL
         SHLD   PC         ; PC is loaded with the value of HL.
         RET               ; Return.
NOT1:    DCR    A
         JNZ    NOT2       ; if it is not a type 2 inst, goto NOT2.
                           ; Type 2 – CALL and Ccond inst.
         LHLD   IBUFF+1;   ; The operand of the call inst in the inst buffer
         SHLD   JCAD       ; is temporarily store at JCAD
         LXI    H,CAPT     ; and the address CAPT is put
         SHLD   IBUFF+1    ; into the instruction.
         JMP    LDSTK      ; Goto LDSTK directly.
NOT2:    DCR    A
         JNZ    NOT3       ; If it is not a type 3 inst, goto NOT3.
                           ; Type 3 – JMP and Jcond inst.
         LHLD   IBUFF+1    ; Similar to type 2 inst.
         SHLD   JCAD
         LXI    H,JPPT     ; except JPPT is used instead of CAPT.
         SHLD   IBUFF+1
         JMP    LDSTK
NOT3:    DCR    A
         JNZ    LDSTK      ; If it is not a type 4 inst, goto LDSTK.
                           ; Type 4 – RST inst.
         LHLD   PC         ; The next 8 inst push the value of PC into user stack:
         XCHG              ; Set (D,E) to PC value.
         LHLD   USESTK     ; Load address of user stack.
         DCX    H
         MOV    M,D        ; Move high value of PC.
         DCX    H
         MOV    M,E        ; Move low value of PC.
         SHLD   USESTK     ; Update user stack pointer value.
                           ; The next 6 inst set PC to its correct value.
         LDA    IBUFF+2    ; ACC now contains the RST inst.
         ANI    70         ; The higher 2 and the lower 3 bits are stripped.
         MOV    L,A        ; Set L = low address.
         XRA    A          ; high address is zero.
         MOV    H,A        ; Set H = high address.
         SHLD   PC         ; store (H, L) at PC.
         RET               ; Return.
LDSTK:   LXI    H,0        ; Store the simulator stack pointer value.
         DAD    SP
         SHLD   STACK      ; at location STACK,
         LHLD   USESTK     ; and set the stack pointer to the value stored
         SPHL              ; at USESTK, which contains user stack pointer value.
         DCR    A
         JNZ    RESTORE    ; If it is not a type 5 inst, goto RESTORE.
                           ; Type 5 – RET and Rcond inst.
         LXI    H,REPT
         PUSH   H          ; The address REPT is forced onto the stack.
; *** RESTORE ***
RESTORE: LHLD   BC         ; Restore values of reg B and C from loc BC.
         MOV    C,L
         MOV    B,H
         LHLD   DE         ; Restore values of reg D and E from loc DE.
         XCHG
         LHLD   SW         ; Restore values of acc and PSW from loc SW.
         PUSH   H
         POP    PSW
         LHLD   HL         ; Finally, restore values of reg H and L from loc HL.
; *** INSTRUCTION BUFFER ***
IBUFF:   DB     0,0,0      ; The 3 bytes of inst in the inst buffer are executed.
; *** SAVE ***
SAVE:    SHLD   HL         ; Values of reg H and L are saved at HL.
         PUSH   PSW        ; Values of acc and PSW are saved at SW.
         POP    H
```

need special preparation and clean up work. These are:

1) PCHL,
2) CALL and call with conditional execution,
3) JMP and jump with conditional execution,
4) RST,
5) RET and return conditional.

Detailed Description

The size of the instruction buffer is set to hold one instruction. It is a 3 byte memory location for an 8080. If the instruction to be simulated is less than three bytes, the buffer is filled with NOP instructions. A 2 byte memory location, which we call PC, is used as a pointer into the user program to be simulated. It always points to the first byte of the next instruction to be simulated. Memory locations are used for storing the stack pointer, accumulator, program status word and the other six 8080 registers. Since the value of the actual 8080 stack pointer is shared by the simulator and the user program, variables STACK and USESTK are used to store the simulator and user program stack pointer values during alternate use.

Below is a detailed description of each of the modules in the simulator routine. The program is listed in listing 1, and should be read in conjunction with the text.

Recognizer

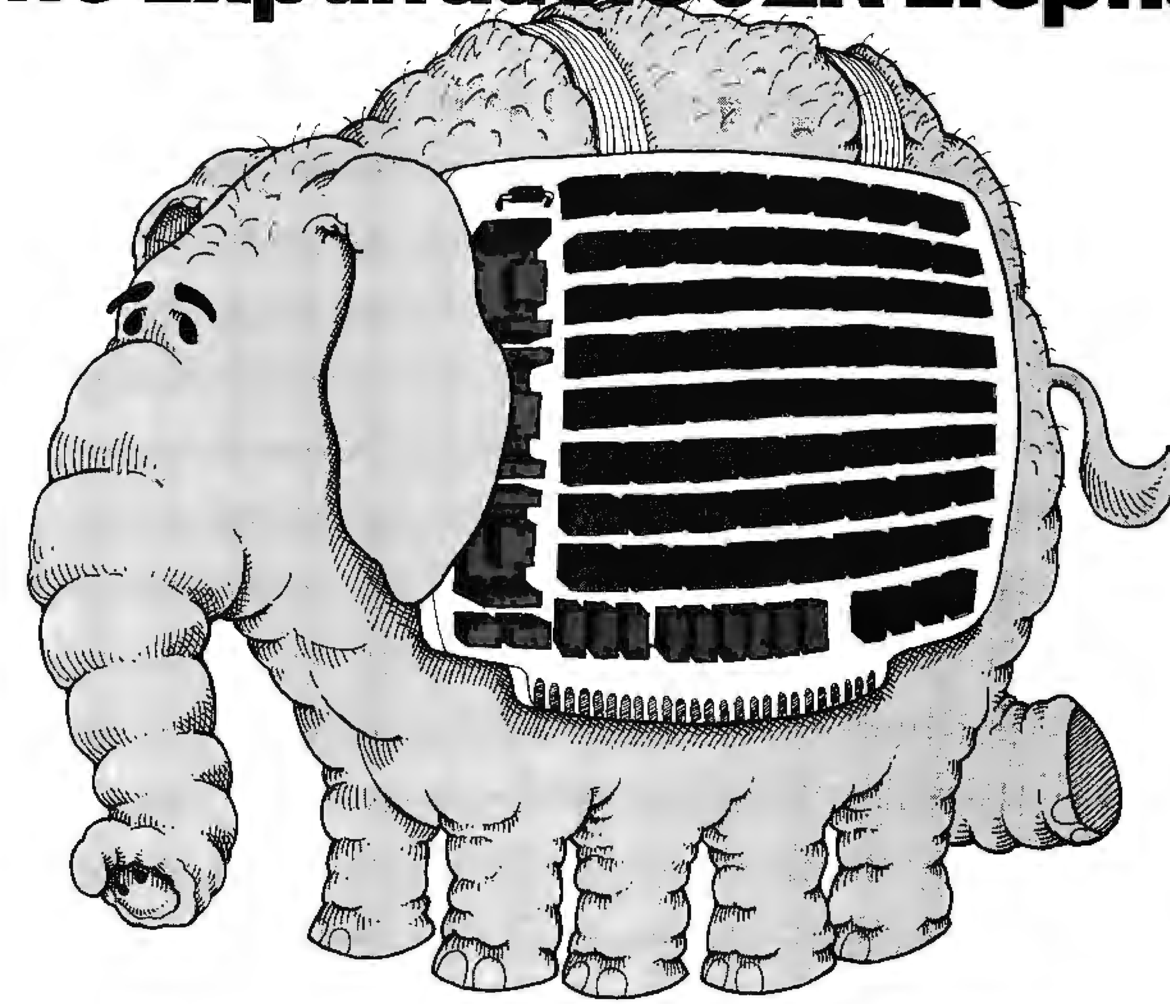
The recognizer routine determines the number of bytes used by the instruction and the type of instruction to be executed. This is done by the use of a table (TBL). Each entry in the table consists of four bytes, which we call b_1, b_2, b_3 and b_4. The algorithm used is (using a BASIC-like statement):

IF (((inst AND b_1) XOR b_2)=0)
THEN number-of-byte=b_3,
type-of-instruction=b_4

AND and XOR are the logical AND and exclusive OR operations to be carried out bit by bit. The logical operation for each of the entries in the table is checked until one is found to satisfy the logical condition. If no entry satisfies the logical condition, we assume the instruction is a type 6, a 1 byte instruction. This method has the advantage over a table of 256 entries in that it saves space and processing time. The box at the end of this article gives examples of how it works.

The simulator starts by loading the value of the instruction's address and calling the

ARTEC Introduces
The Expandable 32K Elephant

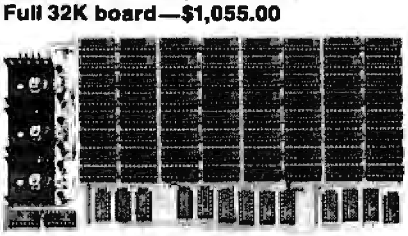


The 8K-32K Expandable Memory That Grows With Your System

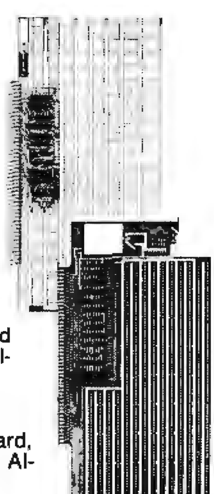
Now, for the first time, you can have a reliable true static memory that will grow with your system. Start with the board and 8K memory. Then add on one, two or three 8K increments of memory up to 32K. 250 ns access time. The Artec 32K Expandable Memory allows you plenty of room for memory and all necessary support hardware.

For five years Artec craftsmanship and reliability has been proven in tough industrial use. Now, you too can enjoy breadboards and memories that will work time after time. Boards like the GP 100 and the wire wrap WW-100. Send for an Artec Board, your order will be sent the same day as received.

Board & 8K of memory—$290.00
8K add on kits—$255.00 ea.
Full 32K board—$1,055.00



GP-100—$20.00
Maximum design versatility along with standard address decoding and buffering for S—100 systems. Room for 32 uncommitted 16 pin IC's, 5 bus buffer & decoding chips, 1 DIP address select switch, a 5 volt regulator and more. High quality FR4 epoxy. All holes plated through. Reflowed solder circuitry.



WW-100—$20.00
A wire wrap breadboard, similar to the GP 100. Allows wirewrap of all sizes of sockets in any combination. An extra regulator position for multiple voltage applications. Contact finger pads arranged for easy pin insertion.

TO ORDER: Use your Mastercharge or BankAmericard. Or just send along a money order. Your order will get same day service.

FOR MORE INFORMATION: For more information about these or any of Artec's complete line of circuit boards or for either industrial or personal use, please call or write. A catalogue will gladly be sent.

Please send me:
☐ 32K ☐ GP-100 ☐ WW-100
☐ I've enclosed a money order.
Bill my ☐ Mastercharge
☐ BankAmericard No. ______
Name ______
Address ______
City ______ State ______ Zip ______

10% discount for students & computer club members.

ARTEC ELECTRONICS, INC.

605 Old County Rd. • San Carlos, CA 94070 • (415) 592-2740

Circle 378 on inquiry card.

Listing 1, continued:

```
        SHLD  SW
        XCHG             ; Values of reg D and E are saved at DE.
        SHLD  DE
        MOV   H,B        ; Values of reg B and C are saved at BC.
        MOV   L,C
        SHLD  BC
        LDA   TYPE       ; Load TYPE into acc.
        CPI   5          ; Is it a type 5 inst?
        JNZ   STSTK      ; Skip the next inst if not.
        POP   H          ; Clean the garbage from the stack.
STSTK:  LXI   H,0        ; Store the user stack pointer value
        DAD   SP
        SHLD  USESTK     ; at location USESTK,
        LHLD  STACK      ; and set the stack pointer back to the value
        SPHL             ; at STACK.
        RET              ; Return.
; *** CLEAN UP ***
CAPT:   POP   H          ; For Type 2 inst, clean up the garbage on the stack
        LHLD  PC         ; and load correct return address onto
        PUSH  H          ; the stack.
JPPT:   LHLD  JCAD       ; For Type 2 and 3 inst, load correct execution address
        SHLD  PC         ; previous stored at JCAD and store at PC.
        JMP   STSTK      ; Resume processing at STSTK.
REPT:   POP   H          ; For Type 5 inst, the correct return address is on the
                         ; top of the stack
        SHLD  PC         ; which is popped and stored at PC.
        JMP   STSTK      ; Resume processing at STSTK.
; *** THE RECOGNIZER SUBROUTINE ***
RECOGN: LXI   H,TBL-2    ; Load the starting address of the table minus 2.
              .          ; Rem: acc contains the inst to be simulated.
        MOV   C,A        ; It is temporarily stored at C.
AGAIN:  MOV   A,C        ; The inst is restored from reg C.
        INX   H
        INX   H          ; (H,L) points to the 1st byte of entry in the table.
        MOV   B,M        ; Get AND mask.
        ANA   B          ; Perform AND operation.
        JZ    BYTE1      ; If result is 0; either NOP inst or end of table.
        INX   H
        MOV   B,M        ; Get XOR mask.
        XRA   B          ; Perform XOR with B.
        INX   H
        JNZ   AGAIN      ; If result is 0 then the inst is not recognized yet.
        MOV   A,M        ; else get and
        STA   NBYTE      ; store number of byte of the instruction.
        INX   H
        MOV   A,M        ; Also get
        STA   TYPE       ; and store type of instruction.
        RET              ; Return.
BYTE1:  MVI   A,1        ; This is a 1 byte inst.
        STA   NBYTE      ; Store 1 into NBYTE.
        MVI   A,6        ; This is also a type 6 inst.
        STA   TYPE       ; Store 6 into TYPE.
        RET              ; Return.
; *** VARIABLE AND DATA ***
NBYTE:  DW               ; Number of byte of the instruction.
TYPE:   DB               ; Type of instruction.
PC:     DW               ; Program counter for the user program.
STACK:  DW               ; Simulator stack pointer.
USESTK: DW               ; User program stack pointer.
SW:     DW               ; User program accumulator and status word.
BC:     DW               ; User program B and C registers.
DE:     DW               ; User program D and E registers.
HL:     DW               ; User program H and L registers.
JCAD:   DW               ; Temporary storage for Type 2 and 3 inst.
;The table used in the recognizer consisted of entries of 4 bytes each.
; Byte 1:  AND mask               Byte 2: XOR mask
; Byte 3:  Number of inst bytes   Byte 4: Type of inst.
TBL:    DB    377,351,1,1  ; PCHL inst
        DB    377,315,3,2  ; CALL inst
        DB    307,304,3,2  ; Ccond inst
        DB    377,303,3,3  ; JMP inst
        DB    307,302,3,3  ; Jcond inst
        DB    307,307,1,4  ; RST inst
        DB    377,311,1,5  ; RET inst
        DB    307,300,1,5  ; Rcond inst
        DB    317,001,3,6  ; LXI inst
        DB    347,042,3,6  ; STA and LDA inst
        DB    367,323,2,6  ; IN and OUT inst
        DB    307,006,2,6  ; MVI inst
        DB    307,306,2,6  ; 2 byte acc arith inst
        DB    0            ; end of the table
```

recognizer. The recognizer should return the correct values for NBYTE (number of bytes) and TYPE (type of instruction). The value of PC is then updated tentatively, by adding NBYTE to PC. We say "tentatively" because it may be altered later if the instruction happens to be a type 1 to type 5.

Fetch

The instruction is then moved from the user program to the 3 byte locations in the instruction buffer. The loop NXT moves the instruction, while the loop NXT2 fills the instruction buffer with NOP instructions to take care of short instructions.

Preparation

The first five types of instructions have to be specially processed before the instruction in the instruction buffer is executed. Type 1 and type 4 instructions actually bypass the instruction buffer.

Type 1: PCHL

The content of PC is set to the value stored at HL.

Type 2: CALL and Conditional Call

A CALL or conditional call instruction, a CZ ADDR for example, is changed to a CZ CAPT instruction, where CAPT is the starting address of a routine that takes care of the preparation for CALL and conditional call instructions. The original address ADDR is stored at JCAD for further processing at CAPT. When the instruction in the instruction buffer is executed, and in our example, if the zero flag is not set, the next sequential instruction is executed. If however the zero flag is set, then it would branch to location CAPT instead of the original location ADDR. Either way we are in control.

Type 3: JMP and Conditional Jump

A JMP or conditional jump instruction is treated similarly to a CALL instruction, except that JPPT is used instead of CAPT.

Type 4: RST

These instructions are simulated by first loading the value of PC into locations indicated by USESTK-1 and USESTK-2, and decreasing the value of USESTK by 2. This simulates the push stack operation. PC is then set to its correct value. The low PC value is decoded by using an AND operation. The high PC value is zero.

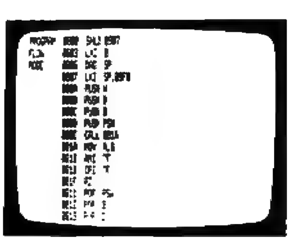
Before starting preparation work for

24 Channel **LOGIC ANALYZER**, complete with 2 cards and 3 sets of probes (only one set shown).

24 channel Logic Analyzer plugs into your S-100 Bus

Features

- 24 channels with 256 samples each.
- Display of disassembled program flow.
- Dual mode operation — external mode analyses any external logic system. Internal mode monitors users data and address bus.
- Selectable trigger point anywhere in the 256 samples.
- 0-16 bit trigger word format or external qualifier.
- 10MHz sample rate (50ns min. pulse width)
- Synchronous clock sample with coincident or delayed clock mode.
- User defined reference memory.
- Displays and system control through keyboard entry.
- TTL Logic level compatible (15 pf and 15 μa typical input loading).
- Includes annotated source listing.



Display of disassembled program flow.

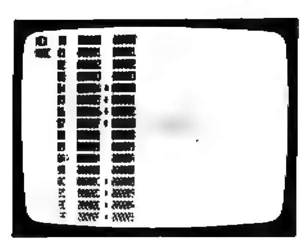
Circle 379 on inquiry card.

The DATALYZER

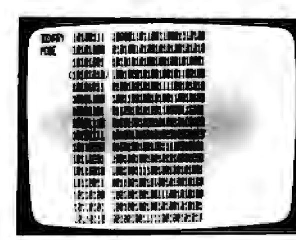
The Databyte Logic Analyzer (DATALYZER) is a convenient, flexible, high quality device. Efficient engineering has allowed a combination of features previously available in only the most expensive units.

Designed to plug easily into your S-100 Bus, the DATALYZER is a complete system –– for only $495. Display of disassembled program flow is a standard feature, not an extra. And the low price includes 30 logic probes, so you can hook up immediately, without additional expense.

The DATALYZER is available in kit form ($495), and as a fully assembled device on two PCB's ($595). Four-week delivery, a substantial warranty, and the Databyte, Inc. commitment to service make the DATALYZER a worthwhile investment. Begin debugging by sending the coupon now.



Displays in Hex



Displays in Binary

Please send me the 24 Channel **LOGIC ANALYZER**
☐ Kit – (manual included) $495.00 (Wis. res. add 4%)
☐ Assembled and Tested (manual included) $595.00
☐ Operators' manual only $7.50
Delivery of all items in four weeks to:
Name ____________
Address ____________
City ____________ State ____________ Zip ____________
Telephone ____________
Payment Enclosed: ☐ Check ☐ Money Order
☐ BankAmericard ☐ Master Charge Exp. Date ____________
Number ____________
Signature ____________

Databyte, Inc.

P.O. Box 14
7433 Hubbard Avenue
Middleton, Wisconsin 53562
Tel: (608) 831-7666

return instructions, we have to save the simulator stack pointer value and set the stack pointer to the user program stack pointer value. There is no instruction in the 8080 processor that directly transfers the value of the stack pointer to registers. We have to use a DAD SP instruction by setting register pair HL to zero first.

Type 5: RET and Conditional Return

The return address (REPT) is first pushed on to the stack. The philosophy here is similar to the call instructions. If the conditional test for the instruction is satisfied, the program will return to REPT instead of the original returning point. The extra unwanted entries on the stack should be cleared. The address REPT is within the simulator and the program is still under simulator control.

Restoration

Since the value of the user stack pointer has already been restored, we only have to restore the values of the accumulator, program status word, and the other six registers from memory locations where they were saved. The only way to obtain the program status word is to push PSW into the stack and pop it up again, making use of the user program stack pointer temporarily. This means that the location USESTK must be properly initialized for the simulator to work. Once the stack pointer is set to a specific value in the user program, that value is used for our temporary work.

Instruction Buffer

By now the instruction buffer should contain the desired instruction, either the modified one or the original one. When this instruction is executed, we should go either to REPT, or to CAPT, or to JPPT, or to the instruction next to it.

Clean Up

If we ever come to the location CAPT, it means that the conditional test for the CALL or conditional call instruction (if any) has been successful and a call has been made. The top of the stack now contains garbage, and should be replaced with the correct return address, the value stored at PC. PC is then set to its correct value, the one that we have previously saved at JCAD.

The clean up work for the JMP and conditional jump instructions is similar except that no stack is involved.

If we come to the location REPT, it means that the conditional test for the RET or conditional return instruction (if any) has

x AND y

y \ x	0	1
0	0	0
1	0	1

x XOR y

y \ x	0	1
0	0	1
1	1	0

Table 1: The results of the AND operation and the XOR operation.

Examples of the Use of TBL

The opcode for conditional jump instructions is 3d2 in octal, where d can be 0, 1, . . ., 7. Pick an instruction, say the JZ instruction:

```
JZ:          312 11001010
```

From the program we also know:

```
AND mask:    307 11000111
XOR mask:    302 11000010
```

Referring to the table for the logical bit operation, perform the required operation:

```
             11001010
AND          11000111
             --------
             11000010
XOR          11000010
             --------
             00000000
```

producing a zero. You can check that all the conditional jump instructions produce a resultant zero. Table 1 is an illustration of the results of the AND and XOR operations.

If you pick another instruction, say the RET instruction:

```
RET:         311 11001001,
```

and perform the same operation:

```
             11001001
AND          11000111
             --------
             11000001
XOR          11000010
             --------
             00000011
```

you will find a nonzero result. It is also obvious that any instruction other than conditional jump instruction would produce a nonzero result.

How do you get the AND mask and XOR mask in the first place?

As an example, look at SHLD, LHLD, STA and LDA instructions:

```
SHLD:        042 00100010
LHLD:        052 00101010
STA:         062 00110010
LDA:         072 00111010.
```

Combine these four instructions into one using an OR operation:

```
001dd010
```

where d stands for don't care, meaning that it can be either one or zero. Obtain the AND mask by assigning a 0 to d bits and a 1 to the others. Therefore:

```
AND mask =   11100111.

             001dd010
AND          11100111
             --------
XOR mask:    00100010
```

The resultant byte from the AND operation is then the XOR mask.

been successful and a return has been made. The top of the stack now contains the true return address. It is then popped and stored at PC.

Save

The values of accumulator, program status word, registers and the stack pointer are saved in memory locations, reversing the process in RESTORE.

This completes our discussion for the simulation of a single instruction cycle.

Conclusion

This article has demonstrated how an 8080 simulator can be written on an 8080 computer. Similar simulations can be done on any other microcomputer, using the same basic technique. Writing such a simulator can be greatly simplified by making use of the existing host computer. The scheme I have shown is a subroutine that simulates an instruction cycle. It can be called repeatedly to simulate continuous execution. Various tests can be made each time before calling the subroutine, and execution can be suspended at any time if necessary. A combination simulator and debugging system can be easily designed and implemented to include the following features:

- A monitor routine that parses, recognizes the commands and branches to the appropriate routines.
- A single step routine that calls the simulator routine once and returns.
- A set break point routine that allows the user to enter break point addresses into the break table.
- A dump register and load register routine that displays and loads the values of various registers.
- An examine, deposit, examine previous, examine next routine that simulates the functions of front panel switches.
- A run routine that repeatedly calls the simulator routine. Before calling the routine it compares the value of PC (the program counter) with the entries in the break table, checks whether the value of PC is within the designated limits, and stops execution or gives warnings if appropriate.

Such a combination simulator and debug system can be very useful in developing and debugging programs written in assembler or machine language. Such a project may turn out to be too complex for beginners, but it is one that is worth undertaking as a challenge by more experienced programmers.■

Continued from page 68

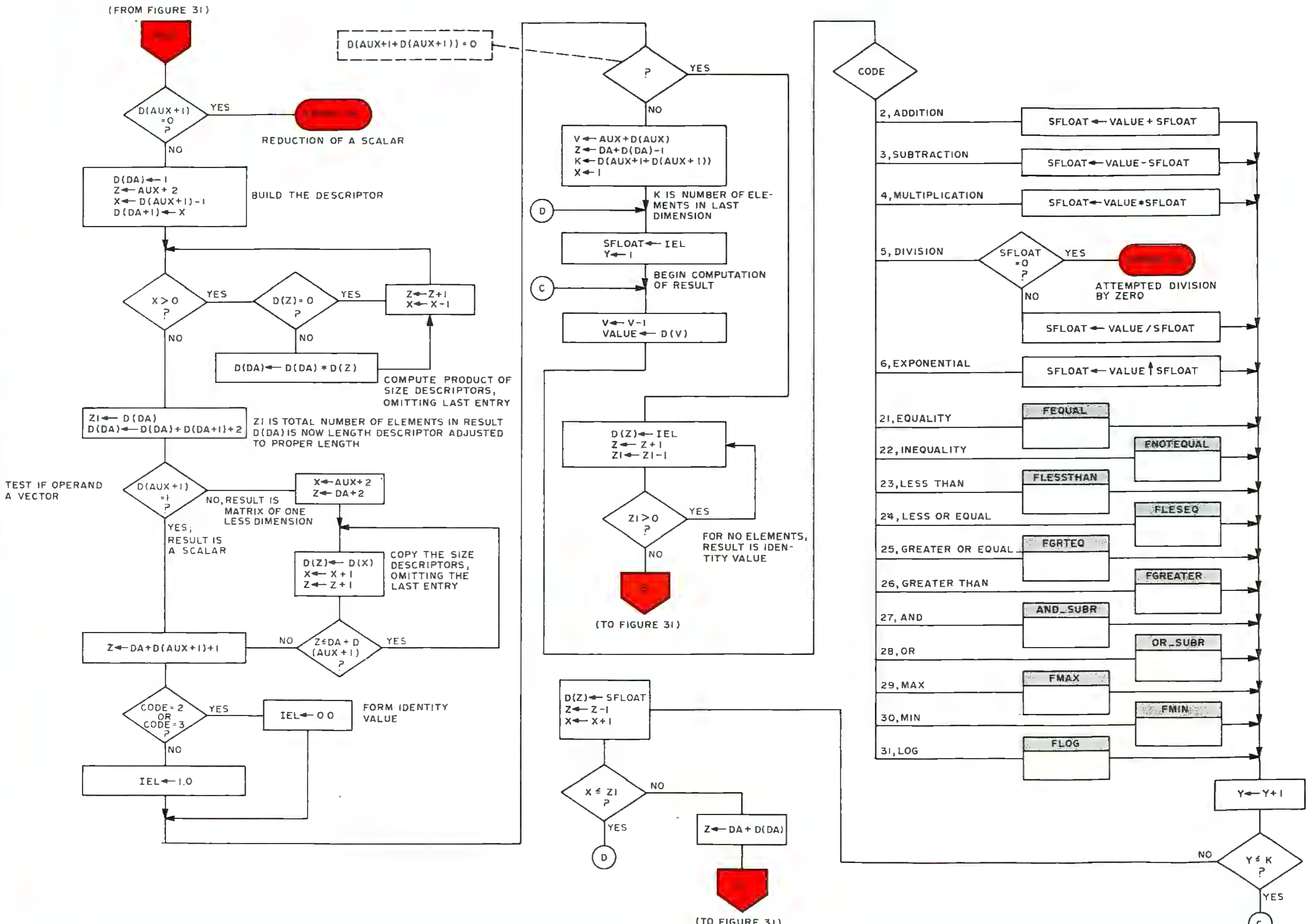


Figure 34: Interpretation of the reduction operators.

Interpretation of the Reduction Operators

Interpretation of the reduction monadic argument is one of the longer processes. The argument is any nonscalar value, and the result, which has one less dimension than the argument, is formed by applying the reducing operator iteratively to every element in the highest dimension.

Figure 34 illustrates the reduction process. First, the descriptor of the result is built. This descriptor will be the same as the argument descriptor except that the last dimension is omitted and, correspondingly, the length of the result may be smaller. Next, the identity value is determined by the type of reduction specified. If the last dimension of the input argument is zero, the reduction results in the identity value. Otherwise the elements of the result are computed by applying the monadic operator to the vector of elements along the last dimension of the input argument. Note that since reduction proceeds right to left across this vector, it actually results in the dyadic use of the monadic operator.

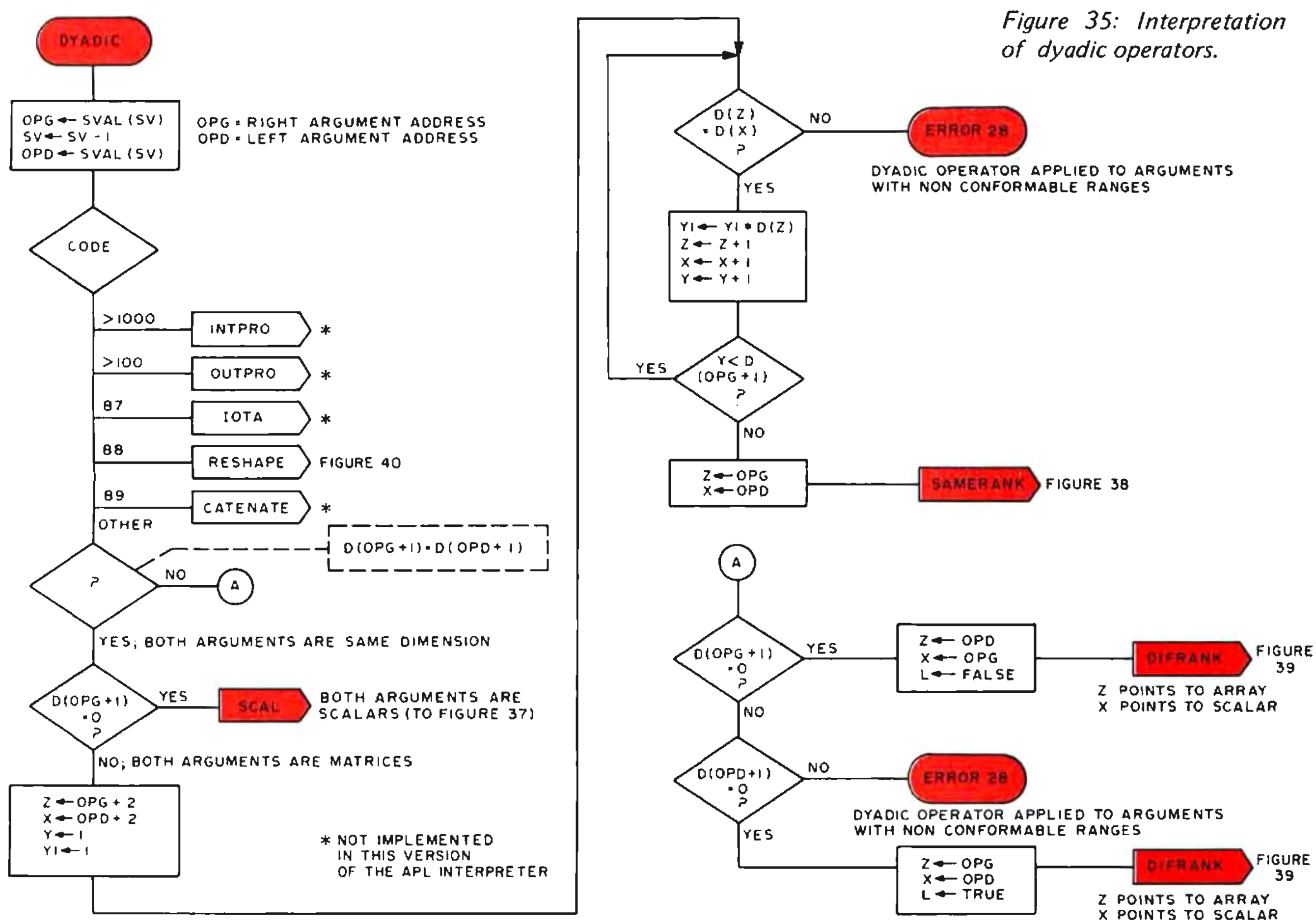


Figure 35: Interpretation of dyadic operators.

Interpretation of Dyadic Operators

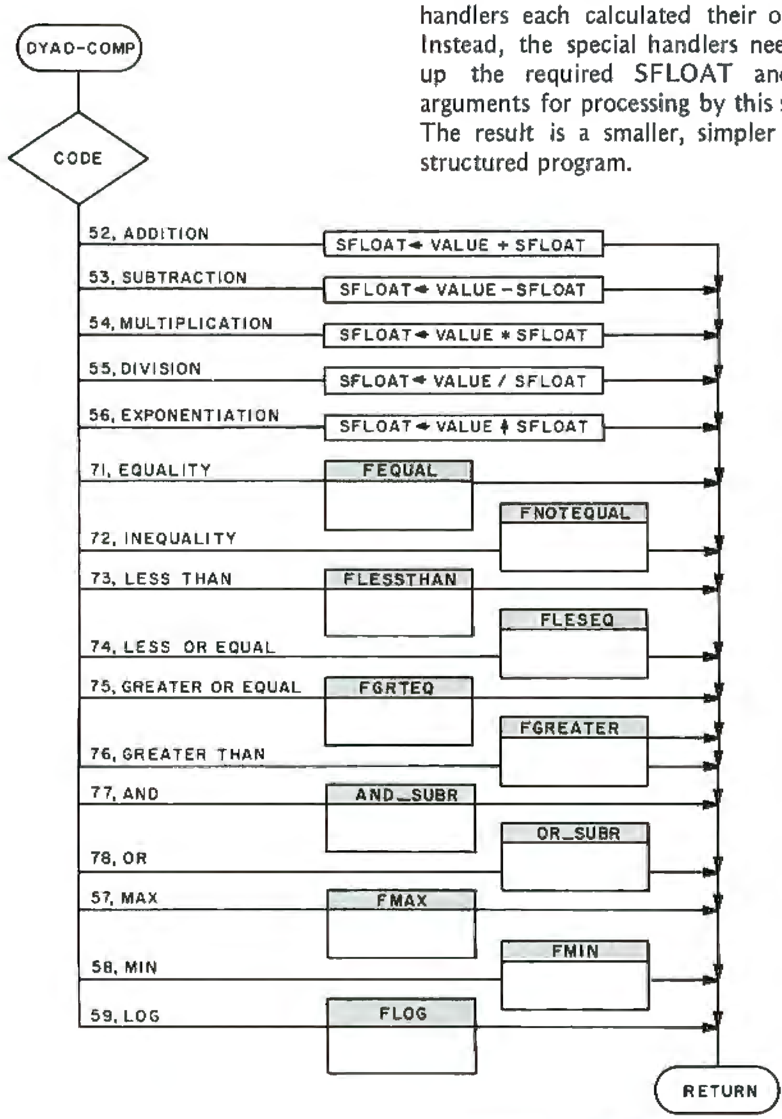
Any routine designed for the interpretation of dyadic operators must, in contrast to monadic operators, be able to handle a myriad of possibilities. Not only are there the special cases of inner and outer products, but iota, reshape and concatenate operators must be specially handled. Furthermore, the nonspecial cases may have scalar arguments, nonscalar arguments of the same size and shape, or a scalar argument and a nonscalar argument. Each case requires some extra processing.

Figure 35 begins the description of dyadic operator interpretation. Two pointers are popped from stack SVAL. OPG points to the right argument of the dyadic operator and OPD points to the left argument. Next, the various processing method requirements are determined as described above. Inner and outer product interpreters as well as IOTA and CATENATE interpreters were not implemented in this version of the APL interpreter and will not be described here. The special processing then continues at one of the following blocks: RESHAPE, SCAL, SAMERANK and DIFRANK.

Calculating a Numerical Dyadic Result

Before proceeding further, look briefly at figure 36. Whenever a numerical dyadic result is needed, this subroutine is invoked. This saves the redundant coding which would result if the various special dyadic handlers each calculated their own results. Instead, the special handlers need only set up the required SFLOAT and VALUE arguments for processing by this subroutine. The result is a smaller, simpler and better structured program.

Figure 36: Computation of a numerical dyadic result.



- ANY NUMBER OF FILES MAY BE OPEN (IN USE) AT ONE TIME
- THE NUMBER OF FILES AND SIZE OF FILES IS LIMITED ONLY BY THE SIZE OF THE DISC
- MERGING FILES REQUIRES NO EXTRA DISC SPACE
- NO WAITING FOR THE DISC TO RE-PACK
- LONGER DISC LIFE — MORE EVEN DISC WEAR

Our Basic Floppy Disc System (BFD-68) must, in all modesty be called superb. It comes completely assembled with a disc controller that is plug compatible with the SWTPC 6800. In fact all our products use the 6800 standard SS-50 (Smoke Signal 50) bus used by SWTPC. The cabinet and power supply are capable of handling up to 3 Shugart Mini-Floppy Drives. One drive is included in the price of the BFD-68 and others may be added easily at any time. Or you may save money by ordering the dual-drive BFD-68-2 or triple drive BFD-68-3 (pictured). Price: BFD-68 $795, BFD-68-2 $1139, BFD-68-3 $1479, SA-400 Drive $360.

The BFD-68 includes our Disc Operating System Software. Our software provides for a soft-sectored disc format consisting of 128 bytes per sector, 18 sectors per track and 35 tracks per disc. The software provides direct commands to name and rename files, transfer memory to disc and disc to memory and to automatically jump to the starting location of any program loaded from disc to memory. The direct command names are: RUN, GET, GETHEX, CLOSE, SAVE, DELETE, APPEND, RENAME, COPY, LIST, FIND, LINK and PRINT. In addition, the Disc File Management subroutines are available to create files under your program control.

A bootstrap PROM is included on the controller board to initiate the Disc Operating System which loads into a 4K memory board located at 7000 or optionally at D000. Thus, you can be up and running from a cold start in just a few seconds.

SUPER SOFTWARE

Free patches are provided for SWTPC BASIC version 2.0 and Co-Resident Editor/Assembler. These patches allow the SAVE and LOAD commands to work with the disc or the cassette at your option.

SUPER EDITOR: Smoke Signal Broadcasting now has its own editor. It is a content oriented editor with string search and block move capability. Changes may be made by referring either to line number or string content or a combination of references. Naturally, it is designed for file transfer to and from the BFD-68. Price: SE-1 $29 on diskette or cassette.

ALL OUR PRODUCTS EXCEPT THE PS-1 ARE COMPLETELY ASSEMBLED.

SUPER ASSEMBLER: Inputs source code from file on the BFD-68 disc system and outputs object code to disc file. Assembly listings include alphabetized and tabulated symbol table. Price: SA-1 $29 on diskette or cassette.

Complete source listing included for both editor and assembler. Order both for $53 and save $5.

SMARTBUG — A CURE FOR MIKBUGITIS: A super smart Motorola-Mikbug replacement that preserves almost all Mikbug entry locations so your present programs will run without modification. Uses ACIA instead of PIA and includes many additional features including a software single-step trace command. A SMARTBUG listing is included and object code is provided on a 2708 free with each P-38 series board purchased. Source listing available separately for $19.50.

NEED A FULL SIZE FLOPPY? Our P-38-FF is a plug-in interface card to the ICOM Frugal Floppy™. It includes all the features of the P-38-I plus one 2708 EPROM containing the ICOM bootstrap software. Just plug the P-38-FF into your SWTPC 6800 and your ICOM into the P-38-FF and you're ready to use the Frugal Floppy and ICOM's 6800 software package. Price $299.

Our P-38 is an 8K EPROM board containing room for 8 2708's. Or, you may use it to hold up to 7 2708's plus your Motorola Mikbug or Minibug II ROM. The P-38 addressing is switch selectable to any 8K location. Price $179.

The P-38-I contains all the features of the P-38 plus an interface to the Oliver Paper Tape Reader and our EPROM Programmer. Price $229.

The PS-1 Power Supply Kit provides plus and minus 16 volts required for the P-38 series boards. Also, it allows a wiring modification to be made to the 8 volt supply that will increase its output by one volt. Price $24.95.

Our M-16-A is a 16K single power supply STATIC RAM memory system. The M-16-A is fully buffered and requires only half the power of a similar size system using low-power 2102's. With the M-16-A, you can expand your system to 48K and still have room left for one of our EPROM boards. The M-16-A is switch selectable to any 4K starting address and hardware write protect is included. Quick delivery. Price. $529.

BANKAMERICARD, VISA AND MASTER CHARGE WELCOME.

SMOKE SIGNAL BROADCASTING

P.O. Box 2017, Hollywood, CA 90028 • (213) 462-5652

Circle 381 on inquiry card.

Special Cases for Dyadic Operator Arguments

When both arguments of the dyadic operator are scalars, processing is simple (see figure 37). Similarly, for nonscalar arguments with the same size and shape, the scalar process is more or less applied iteratively (see figure 38). Where only one argument is a scalar, interpretation is a hybrid of figures 37 and 38. This is shown in figure 39. Here the scalar value is applied iteratively to the vector or array.

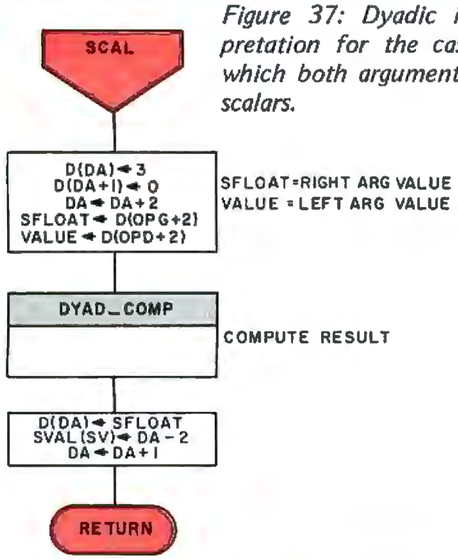


Figure 37: Dyadic interpretation for the case in which both arguments are scalars.

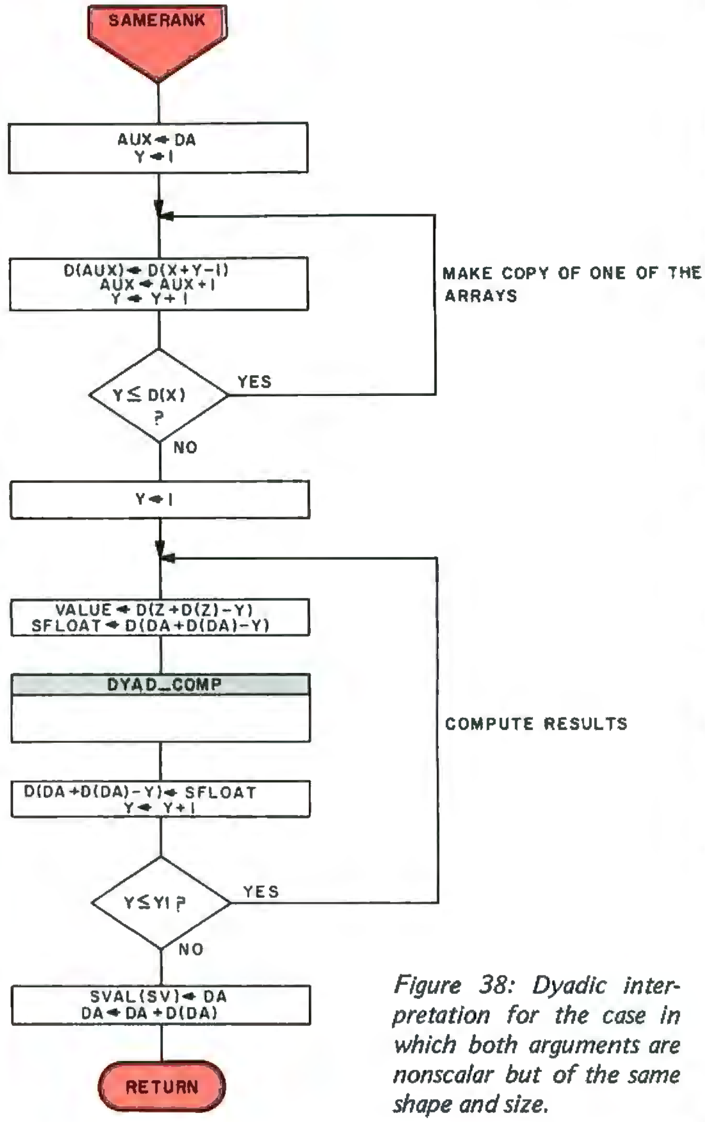


Figure 38: Dyadic interpretation for the case in which both arguments are nonscalar but of the same shape and size.

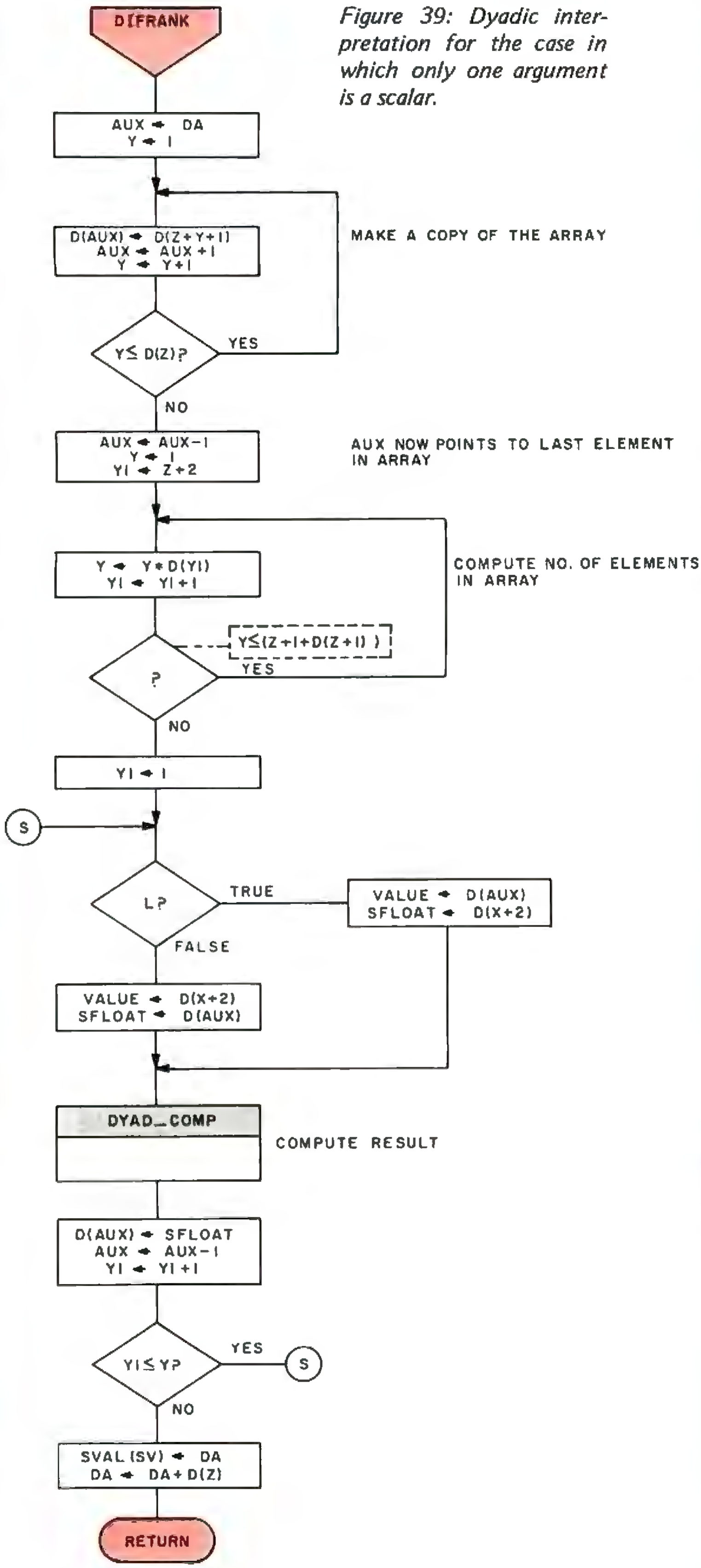


Figure 39: Dyadic interpretation for the case in which only one argument is a scalar.

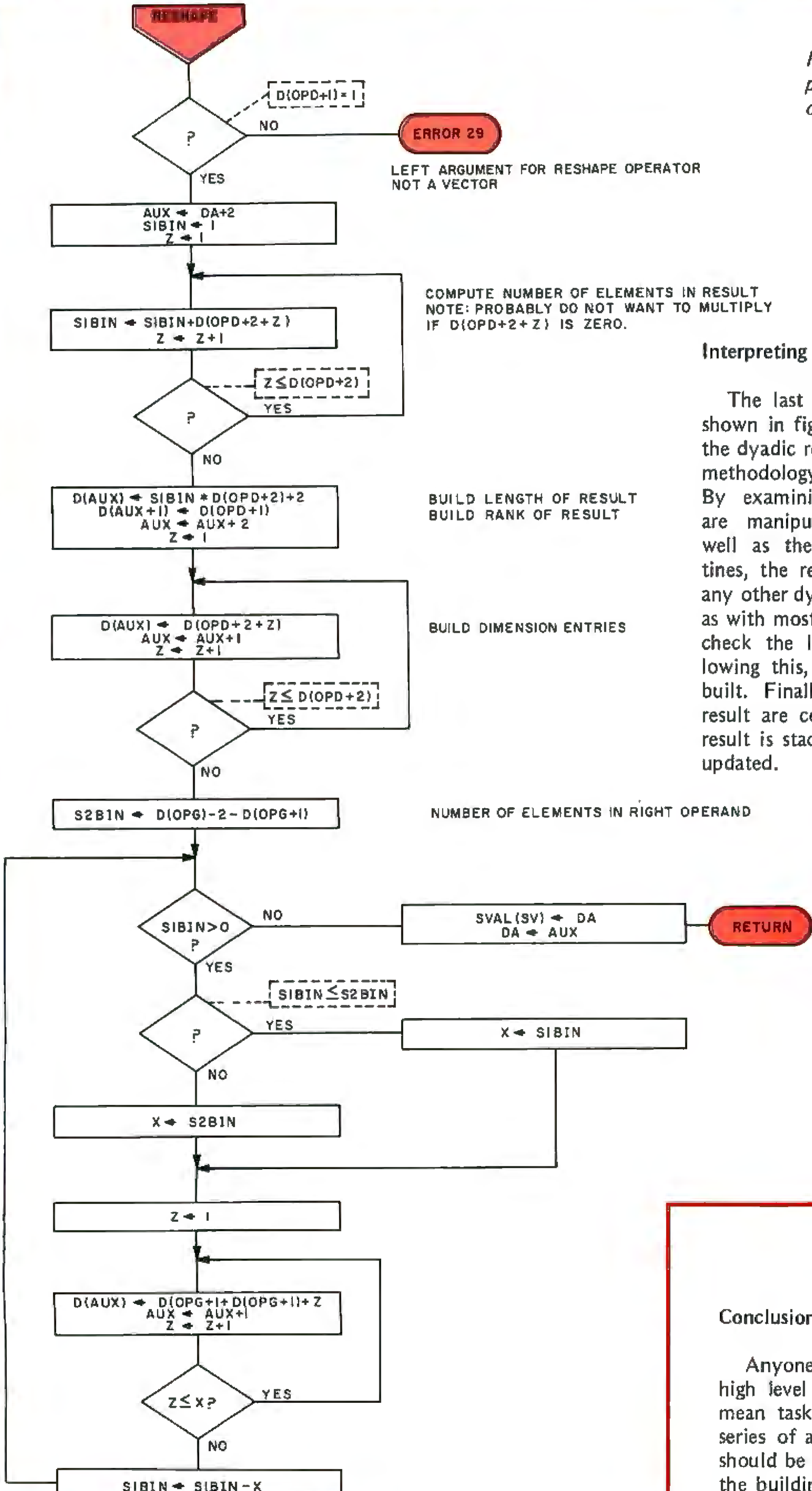


Figure 40: Dyadic interpretation of the reshape operator.

Interpreting the Dyadic Reshape Operator

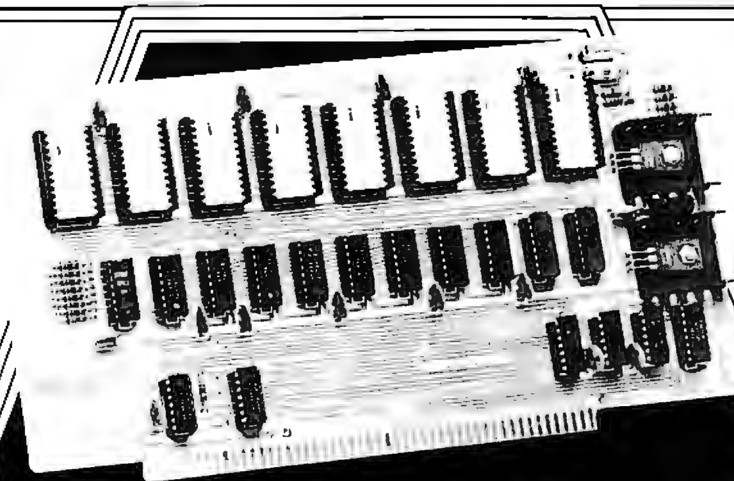
The last dyadic interpreter described is shown in figure 40. This process interprets the dyadic reshape operator, but the general methodology used is of more importance. By examining how the various pointers are manipulated in this subroutine, as well as the previously described subroutines, the reader should be able to design any other dyadic subroutine. The procedure, as with most numerical processors, is to first check the legality of the arguments. Following this, the descriptor of the result is built. Finally the value elements of the result are computed and a pointer to the result is stacked in SVAL and index DA is updated.

Conclusion

Anyone who has worked with powerful high level interpreters knows that it is no mean task to describe them in so short a series of articles as this. Nevertheless there should be enough information here to allow the building of a powerful APL interpreter. Admittedly, there is room for improvement in some areas of design and implementation. Using the experience I gained from my first implementation, I have incorporated many desirable features into several microprocessor implementations which are currently being designed.

Anyone designing an APL interpreter based upon this series of articles and encountering problems should first consult Mr Robinet's article, "Architectural Design of a Directly-Executed APL Processor" (National Technical Information Service, US Dept of Commerce, 5285 Port Royal Rd, Springfield VA 22161), and, failing resolution, should contact me. Finally, although I have attempted to verify the accuracy of all tables, charts and methods used herein, it is still possible that errors or omissions remain.

During the time that these articles have been prepared, I have been actively working on an 6800-based APL interpreter. This work has led to changes and improvements to the flowcharts. Readers who would like to have the latest information are welcome to send a stamped, self-addressed envelope to Mike Wimble, 6026 Underwood Av SW, Cedar Rapids IA 52404.■



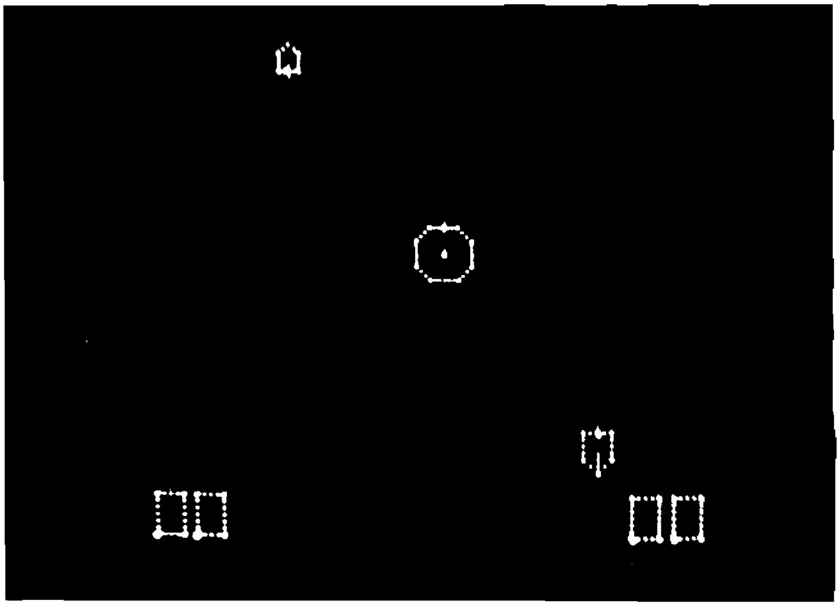


Photo 1: This is the video display at the start of a Space War game. The two spaceships are in their normal starting positions and accelerating slowly towards the sun. The sun image is 14 units across and centered on the screen, but its gravitational field permeates all of space in its vicinity (table 4).

How to Implement Space War

Dave Kruglinski
3256 Portage Bay Pl E
Seattle WA 98102

You say you'd love to go and play "Pong," "Tank" or "Shoot 'em" down at the corner tavern but they won't let you in because you're not old enough? Or maybe you learned to play Space War back in college on a turbocharged PDP 911/S megacomputer and you don't think your poor microcomputer could keep up? Or perhaps you hate BASIC but can't think of a good reason to mess with 50 pages of assembly language. Even if you don't fit into any of the above categories, merely reading this article will sufficiently disturb your mental equilibrium to start you on your way to being a Space War freak, making all other problems fade into insignificance.

Yes, it's true, despite what false impressions you may have held. You can play Space War on your micro: real Space War with gravity and torpedoes and thrust and explosions and hyperspace, all right there in front of your eyes on the screen. No message slowly banging out on the Teletype "THE KLINGONS ARE APPROACHING," but your own spaceship orbiting the sun, able to change direction, accelerate and fire at the enemy. There's even a score to keep track of how many times you've been hit. "Hold it!" you say, after pouring over the listings. "This doesn't conform to ANSI 74 Space War Standards. Alpha Centauri isn't at coordinates 137, 245 like it's supposed to be." Relax, just change the program to suit your taste.

It should be apparent by now that you need more than just a "naked micro" to do the job, but what you do need is not too unreasonable. Table 1 lists the equipment I found necessary to implement a quite realistic Space War game which is the subject of this article.

If you don't like the part about using an oscilloscope with its small screen, don't knock it if you haven't tried it. You just have to sit close. Besides, you can move up to a bigger display later. A vector graphics system would be an improvement, allowing more odds and ends on the screen before flicker sets in. The ideal, however, is a raster scan TV monitor system with resolution close to 256 by 256; anything less is just too coarse. Thomas R Buschbach's article in the November 1976 BYTE *["Add This Graphics Display to Your System," page 32]*

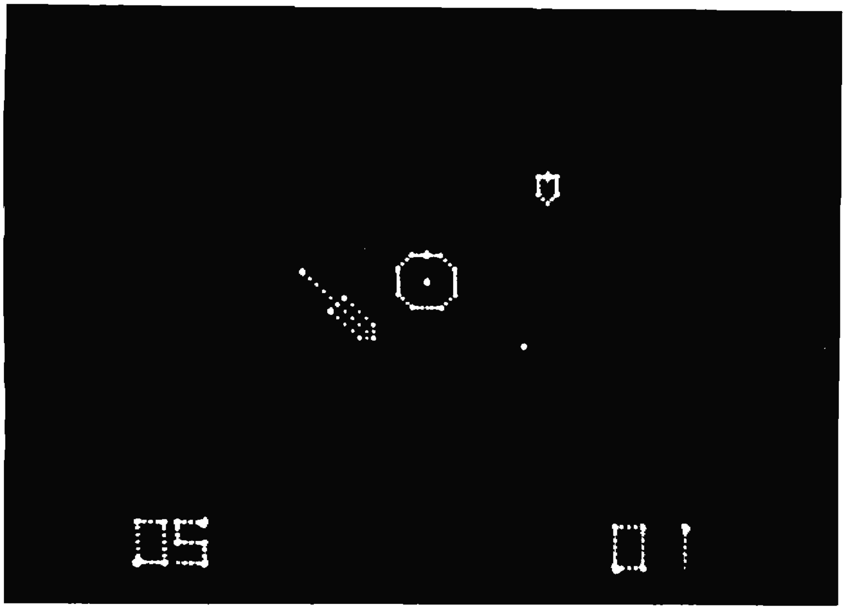


Photo 2: Here the larger of the two ships is accelerating towards the lower right of the screen. This information is relayed visually by the trail that is seen in back of the spaceship. The small dot which is positioned at roughly equal distances from both ships is a torpedo which has just been fired by the smaller spaceship.

(or Using Your Oscilloscope as a Telescope)

should be helpful here. Bear in mind that if you go the refresh memory raster scan route you will have to think about erasing things, not a problem with the analog refresh approach.

Meanwhile, back to the software. To start with, BASIC is out because it's far too slow. What's in is 5.5 K bytes of programs and tables, all in assembly language. "What's the secret?" you say. "Why has Space War only been seen on expensive systems up to now?" The answer is a special added ingredient called TLU. TLU stands for Table LookUp and eliminates the need for all multiplications and divisions, making your program run ten times faster. Together with simple difference equations you can use table lookups to generate gravity for pretty

TACTICAL MATERIEL NEEDED

- An 8080 type microcomputer with 8 K of memory and a way to load programs. An Altair 8800 or equivalent will do fine. An assembler and text editor will be necessary if you want to modify the program conveniently.
- A general purpose DC oscilloscope with X input. Bandwidth isn't too important, but it must be DC. The bigger the screen the better. You can get one surplus for about $100, and they're handy for other things, too. A raster scan continuously refreshed point display can also be used with appropriate modifications of graphic generation and object motion routines.
- A 2 channel digital to analog convertor. This device converts two bytes of digital data to two analog signals with a resolution of 1/256 (3.9×10^{-3}). It is the interface between the microcomputer and the oscilloscope, capable of displaying dots at any of 65536 locations on the screen in a 256 by 256 grid. The circuit is quite straightforward. The "Beer Budget Graphics Interface" described in November 1976 BYTE, page 26, will do fine. You could even skip the second latch on the Y strobe.
- Two or more hand held boxes, each with four push buttons. The two boxes together look like one memory byte or input port and are read by the program to determine what the player wants to do.
- Software functioning as described in this article.

Table 1: Basic facilities needed to implement Space War on a microprocessor.

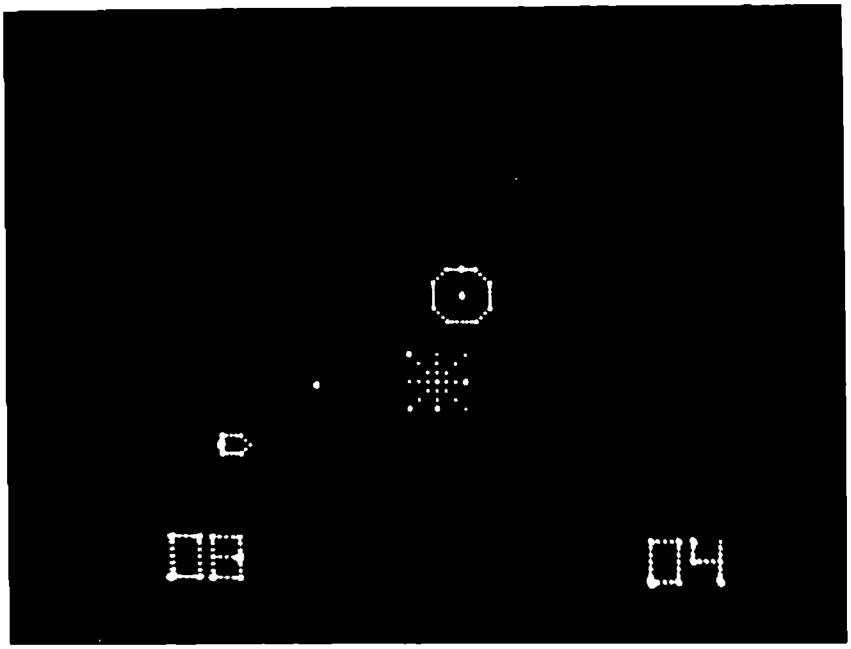


Photo 3: The larger of the two spaceships has been hit and destroyed. It had fired a torpedo just before it was hit.

respectable orbits for your spaceships and torpedoes. Another thing, after subtracting 2 K for the gravity table you have 3.5 K of program to manage. This translates to 55 pages of assembly listing with macros expanded (more on macros later). To do this and not go crazy you have to have a system, a way to organize the code into modules for ease of understanding, modification and debugging. But first some preliminaries.

The Game of Space War

As the game starts you will see a sun in the center of the screen, two scores of zero at the bottom, and two spaceships at opposite corners. You and your opponent each have a ship. Each ship is uniquely identifiable and is moving slowly towards the sun. Photo 1 shows the screen at the beginning of the game as I implemented it.

Each player has a hand held box as shown schematically in figure 1. The functions of the buttons are as follows:

CCW: The player's ship rotates counterclockwise as long as this button is held down. Rotation increments are 45° and the rate is about one rotation every 5 seconds.

CW: Same as a CCW except rotation is clockwise.

FIRE: A torpedo is fired from the player's ship when this button is depressed. The torpedo always originates from the front of the spaceship and travels in the same direction the ship is pointing. Its initial velocity is constant relative to the ship's. Each ship at any one time has only one active torpedo at its disposal and thus can't fire a second until the first has hit a spaceship, hit the sun, hit the screen edge, or timed out.

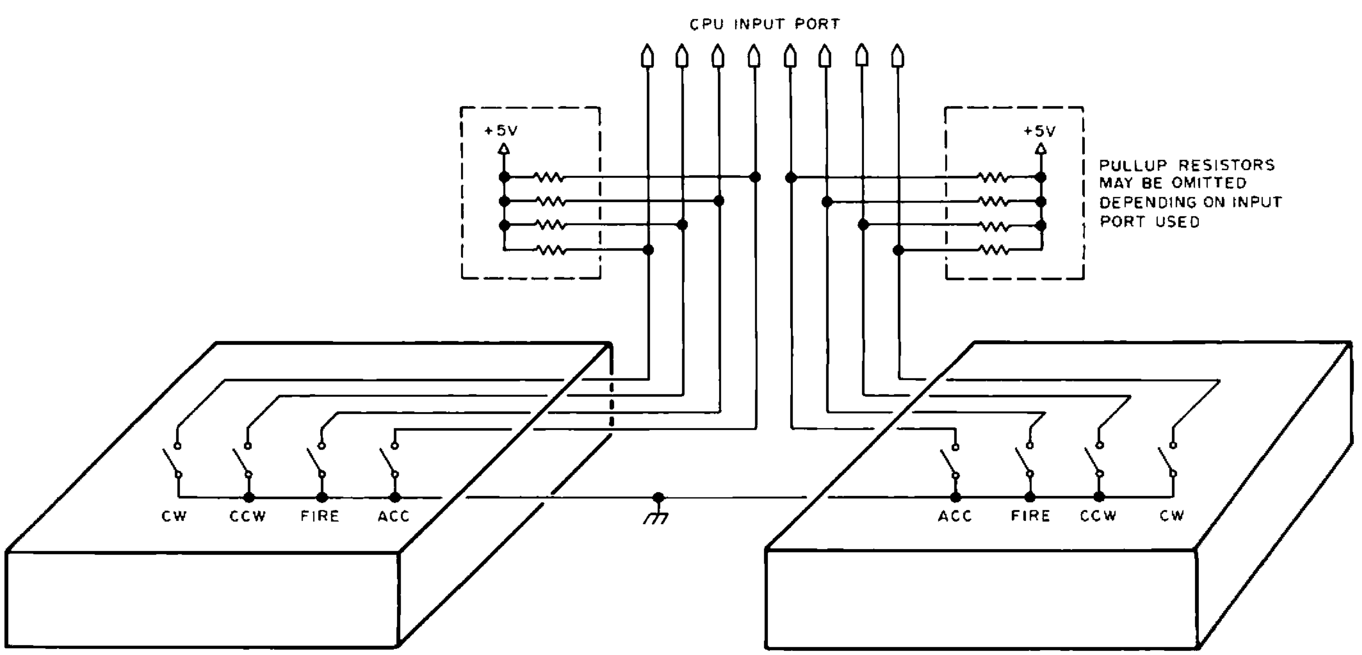


Figure 1: The player control boxes consist of eight switches, four per player, which define data for one input port to the microprocessor. The pullup resistors boxed by a dashed line may be omitted, depending on details of your input port.

ACC: Acceleration is applied to the player's ship as long as this button is held down. The acceleration is applied in the same direction the ship is pointed and is indicated visually by an exhaust trail at the rear of the ship.

The object of the game is to get your spaceship into a stable orbit around the sun and then shoot down the enemy. You have to, of course, watch out for the enemy shooting you down; and you have to be careful not to fall into the sun. If either of these two calamities should befall you, your ship will explode, your score will be incremented, points count against you, and a new spaceship will be delivered to the starting position.

You have a rather interesting option when the going gets tough. You can enter hyperspace by holding down both the CCW and CW buttons at the same time. Your ship disappears, then reappears at some random position with a random velocity. This happens not without risk; however, there is a definite probability, increasing as the game progresses, that your ship will explode when it comes back.

A word about orbits: both spaceships and torpedoes conform to Kepler's laws, traveling in elliptical orbits around the central sun, with revolution periods averaging about 15 seconds. In general, the further away from the sun the ship is, the slower it moves. Thus, if a ship is in a long elliptical orbit the effect will be a little like "crack the whip": the ship zips quickly around the back of the sun, then drifts slowly out into space, then back again. Torpedoes, usually moving faster than spaceships, are less affected by gravity and usually escape to the edge of the screen where they disappear. If they get close to the sun their trajectories may be bent as much as 90°. Due to limitations of the numerical method, the ships and torpedoes may not come closer than seven units (screen is 256 units wide) to the center of the screen. Thus the sun's radius is defined to be seven units and anything coming that close is destroyed. A spaceship which starts anywhere on the screen with zero velocity will move radially inward to the sun and be zapped.

Photos 1 to 5 show some scenes from a typical Space War game. Some are snapshots and two are time exposures showing the motion of spaceships and torpedoes.

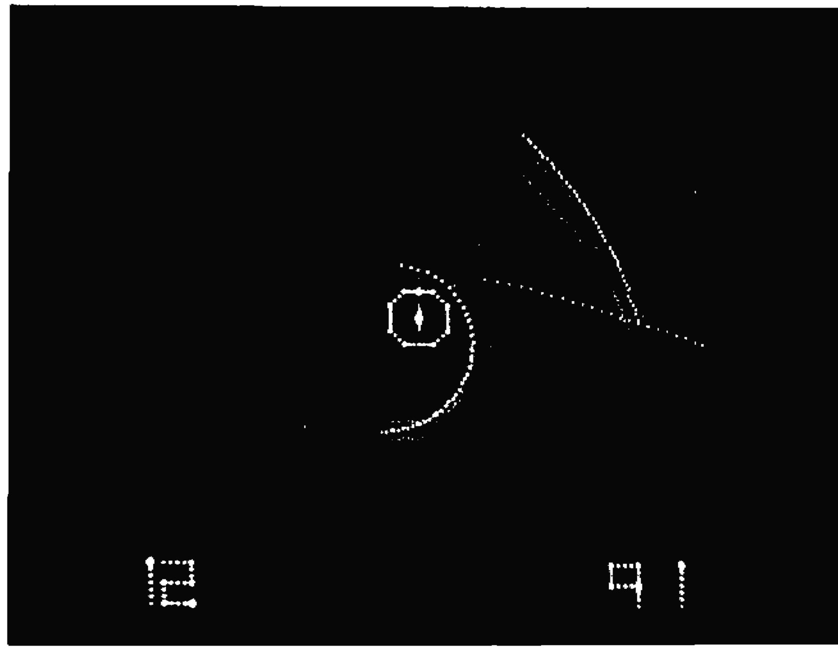


Photo 4: A time exposure showing the two ships in orbit and a near miss of a torpedo. The inner spaceship fired at the outer orbiting spaceship. Notice how the gravity effects of the sun curve the orbits of ships and torpedo.

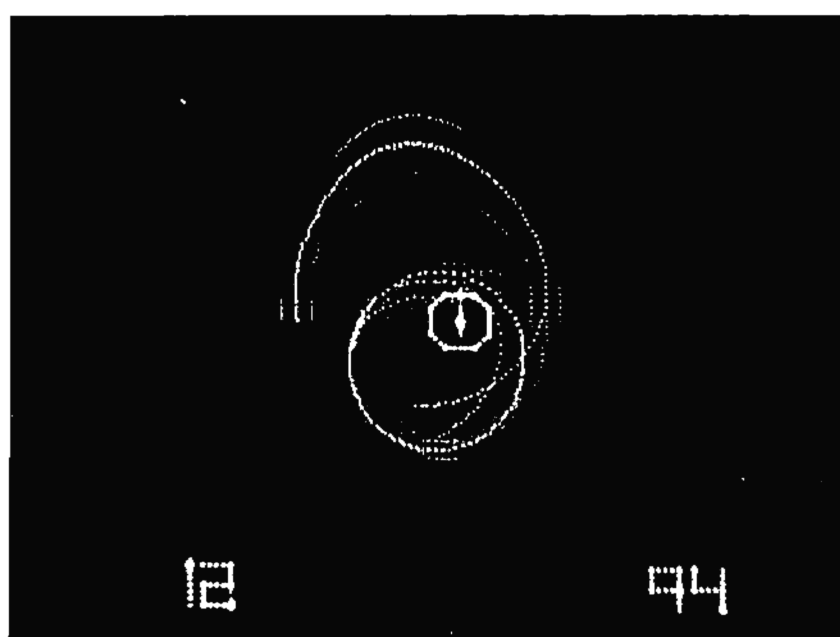


Photo 5: A very long time exposure showing the maneuvering of the two spaceships. Notice that Kepler's laws of motion are being followed, and the spaceship that is closer to the sun is orbiting faster than the spaceship that is farther away, since it has only traveled a portion of one revolution while the inner ship has completely circled the sun. These gravity effects encourage real time tactics and use of strategy in playing the game.

Software – General

Many of the techniques discussed here apply not only to Space War but also to other video games, display systems and real time applications. The only assumption made is that you know what an assembler

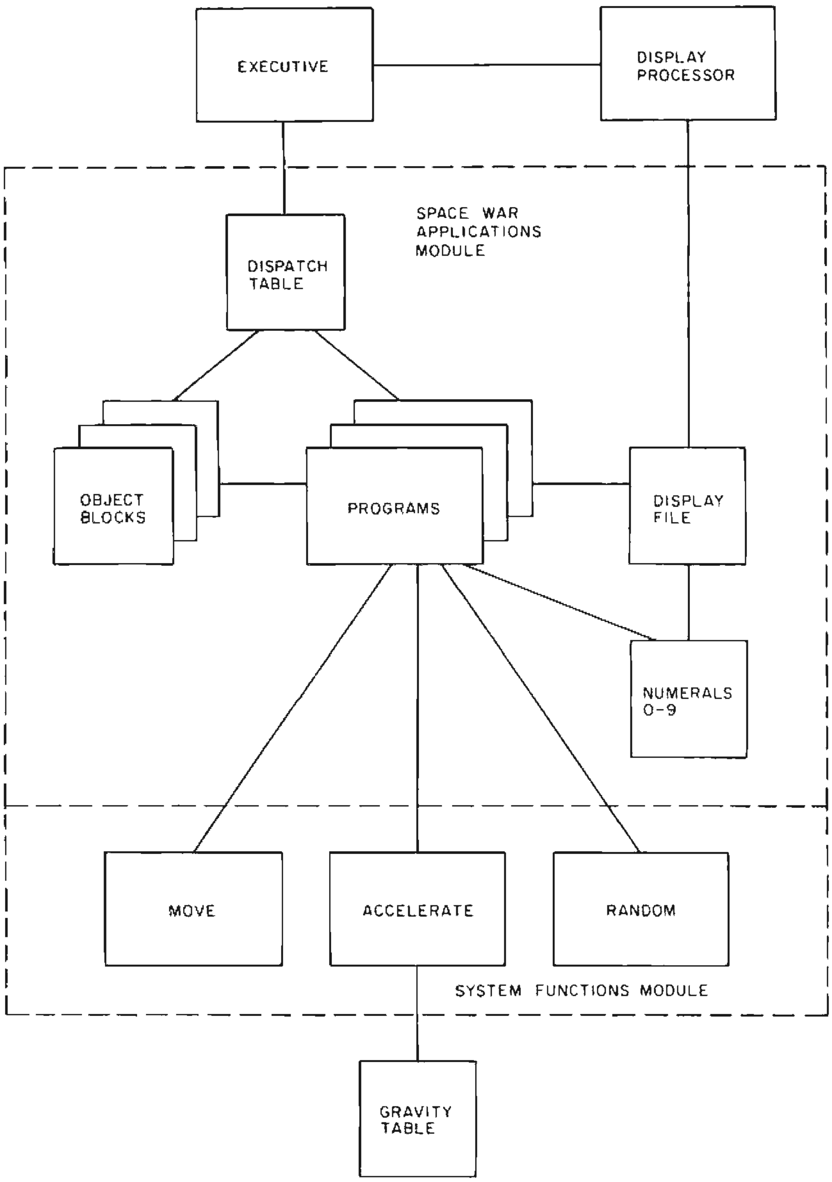


Figure 2: Block diagram of the software for Space War. The rectangles represent programs and the squares represent data. This general type of setup can be used in many other games besides Space War.

does and can read 8080 assembly language.

Since relocatable assemblers and loaders aren't generally available for microcomputers yet, chances are you will have to make the most out of the absolute variety. It is very useful to avoid having the entire 4 K of program in one source module. By breaking the code up into modules you generally need only to edit and assemble a portion of the entire program to make a correction or enhancement. Linkage between modules is made through a few fixed locations in low memory (0 to 80 hexadecimal). There are two categories of modules: system and application. System modules exist to support a variety of application modules. The application module in this case is the Space War video game. The systems modules needed here include: graphics display processor, interrupt handlers and real time executive, system functions. The applications module is a collection of programs and data related specifically to the game. Let's look at each of these modules in detail, while referring to the block diagram in figure 2.

Display Processor

In order to have a video game you have to be able to put pictures on the screen. Still pictures are a good place to start; it's easy to make parts of them move later. With the digital to analog converter and XY oscilloscope hardware configuration of the "Beer Budget Graphics Interface" *[by Peter Nelson, in November 1976 BYTE, page 26]*, it is necessary to paint the picture one dot at a time and also to continuously refresh any image put on the screen a minimum of 20 times a second. This suggests a loop consisting of a sequence of dot producing instructions executed over and over again. It would be very cumbersome, though, to have to write a new assembly language program every time you want a new picture. What about a single program which cycles through a table of XY coordinates? This is better, except it takes a lot of dots to draw a line or spaceship, and it's difficult to move something when you have to move all the dots together. A better solution is an interpretive "language" which you can use to describe pictures. A display file written in this language is interpreted by a special program called the display processor in order to provide the proper sequence of XY coordinates to paint the picture.

Understanding the display processor requires you to visualize the beam which defines a point on the face of the oscilloscope at position X,Y. A dot can be displayed at the beam position or at points relative to the beam position. This is very useful because you can define a spaceship, for instance, as a series of dots relative to the beam coordinates. To move the spaceship you merely move the starting beam position. Rather than define a series of dots individually, why not specify a vector or row of dots? This can be done neatly in one byte where you specify both the length and direction of the vector. Let's say that you have defined your spaceship, but that you want to display it more than once on the screen. Just as in any other computer program you use a subroutine. The subroutine in this case contains a series of relative vector commands and is called from several places in the display program, just after beam positioning commands.

You can pick up the details of the graphics language by studying the nine commands shown in figure 3. Figure 3 also shows the layouts of the graphics commands. In my assembler the mnemonics along with the arguments are put right into a program and the assembler produces the object code by means of macros. (A macro is a way of telling the assembler to substitute a group of instructions for a given symbol.) All the macros are defined at the start of the program with dummy arguments. The real arguments are sub-

Hexadecimal Op Code	Layout	Assembler Macro Form	Meaning
00	0 0 0 0 0 0 0 0 X coordinate Y coordinate	MBEAM x,y	The beam is moved from its previous position to the new coordinates x,y where x and y are between 0 and 255. This command does not cause a dot to be displayed, but is used as a setup for following commands.
02	0 0 0 0 0 0 1 0 X coordinate Y coordinate	MDISP x,y	Same as MBEAM except a dot is displayed at x,y.
04	0 0 0 0 0 1 0 0 X coordinate Y coordinate	LVEC x,y	A "long vector" is drawn from the previous beam position to the new coordinates x,y. The new beam position is now x,y.
06	0 0 0 0 0 1 1 0 E | length | O/F | drctn	SVEC SV length, drctn SVF length, drctn SVE length, drctn SVEF length, drctn E = escape O/F = beam on/off	The relative vector is called the short vector to distinguish it from the absolute long vector. The list is a sequence of length and direction pairs where the length is 0 to 7 dots and the direction is specified by a number from 0 to 7 where 0 is straight up, 1 is 45° in a counterclockwise direction, 2 is 90° counterclockwise, etc.
08	0 0 0 0 1 0 0 0 | scle | | orent	PARAM s,o	This command alters the effect of all following SVEC commands until the next PARAM statement. It is possible to change both scale (s) and orientation (o) of all following short vectors. If scale is set to 1, for example, all short vectors would be twice as long as they would have been if scale were 0. If orientation is set to 2 instead of 0, all short vectors appear rotated 90° clockwise. Thus a figure drawn entirely in short vectors can be enlarged and rotated by changing one command.
0A	0 0 0 0 1 0 1 0 low order address high order address	JUMP addr	Interpreter control is transferred to the command at address addr.
0C	0 0 0 0 1 1 0 0 low order address high order address	JUMPS addr	Interpreter control is transferred to the command at address addr and the address of the command following the JUMPS statement is saved. This is the graphics subroutine call.
0E	0 0 0 0 1 1 1 0	RETS	Return from the subroutine entered by JUMPS command. Subroutines may be nested.
10	0 0 0 1 0 0 0 0	EXEC	Control is transferred to the executive so that the proper applications programs are executed. When these are finished, control is returned to the graphics interpreter command following the EXEC command.

Figure 3: The nine graphic commands used by the display processor showing the op code, related mnemonic, memory allocation, and meaning for each command. These instructions and related programs can be used to implement a wide variety of animated video applications.

Entry	Byte	Data	Functon
1	0	1	look at last bit of TIME only
	1	0	execute on even TIME only
	2,3		object block 1 starting address
	4,5		ship fly program starting address
2	0	1	look at last bit of TIME only
	1	1	execute on odd TIME only
	2,3		object block 2 starting address
	4,5		ship fly program starting address

Table 2: This table design will allow the updating of the positions of two spaceships on alternating cycles. Byte 1 determines whether the updating takes place on an odd or even cycle. If it is set to zero, as in entry 1, updating will occur on an even cycle. If byte 1 is set to 1, as in entry 2, updating will occur on an odd cycle.

Bytes (Hexadecimal Offsets)	Function
0,1	X (n), present x coordinate with 16 bits, 0 at screen center.
2,3	X (n−1), previous x coordinate.
4,5	X acceleration.
6,7	Y (n), present y coordinate.
8,9	Y (n−1), previous y coordinate.
A,B	Y acceleration.
C,D	pointer to MBEAM instruction display file.

Table 3: A list of object parameters which are found at the head of an object block and used by the object move function. These coordinates are continuously updated as the program progresses.

stituted when the macro is actually used.

Now that you can describe a picture with a concise list of graphics commands you should be able to see how easy it is to animate the picture. All it takes is some other program in the system to periodically change parts of the display file. For example, if the MBEAM command preceding a spaceship subroutine call is given a label through the assembler, then the X and Y coordinates can be updated, causing the ship to move. To destroy the ship, replace the spaceship subroutine address with the explosion subroutine address.

See listing 1 for the display processor, and listing 2 for numeral subroutines 0 to 9.

Executive Program

The Space War program was originally designed to work with a real time clock producing 20 interrupts a second. It was discovered later that the update and refresh cycle was stable enough to make the clock unnecessary. This reduced the hardware requirements for the game. To reiterate, Space War runs on a 50 ms cycle. The first 15 ms are used to update positions of spaceships, test buttons, compute scores, etc. The remaining 35 ms are used to refresh the display, that is, process the display file from one EXEC command to the next. The executive:

- determines what update tasks are to be done.
- dispatches a control to these tasks.
- increments a variable called TIME.

A few words on tasks and dispatching are in order here. The applications module consists of a number of independent programs, each with a distinct function which is described in detail in the next section. When each of these programs runs, it references an object. For example, the ship fly program has to run once for ship 1 and again for ship 2. The combination of a program and a particular object is called a task. There are three pieces of information necessary to specify a task:

- frequency and phase at which the task is executed.
- address of the object block containing all parameters relating to a specific object.
- starting address of the program.

The dispatch table (table 2) in the applications module is a list of all the tasks. Each task has a 6 byte entry having the format:

byte 0: Mask which is logically ANDed with TIME and compared with the value in byte 1.
byte 1: Execution time. When this matches the masked value of TIME the task is executed.
bytes 2,3: Address of the object block for this task. It is passed to the program in register pair BC.
bytes 4,5: Address of the program to be called by the executive.

If, for example, you want to update the position of each of two spaceships on alternating cycles, you set up two entries as in table 2.

Most of the task schedules are set up just once when you assemble the applications program. However, it is possible to dynamically schedule one task from another. That scheduled task runs "right now" if both the mask and time bytes are set to zero. It runs "n cycles from now" if the mask is all 1 and the time is TIME+n. The dynamically scheduled task will, however, have to later deschedule itself by setting the mask to 0 and time to 1.

Now for a few words on how programs interface with object blocks. Normally a program references a number of fixed memory locations using direct addressing. This works fine, for example, if you have one program flying one spaceship. For two spaceships you could write two programs,

NOTICE
to all BYTE magazine subscribers and would-be subscribers

SAVE MONEY! As of January 1, 1978, we are raising our subscription rates to meet the increased costs of producing and mailing *BYTE* to our more than 110,000 monthly readers. The new domestic subscription rates, effective January 1, 1978:

Current Rates		Rates Effective January 1, 1978	
One year U.S.	— $12	One year U.S.	— $15
Two years U.S.	— $22	Two years U.S.	— $27
Three years U.S.	— $32	Three years U.S.	— $39

If you already subscribe to *BYTE* and still have several months to go before expiration of your subscription, you can still take advantage of current rates for renewal (even up to three years). Use coupon below. When your present subscription expires, your renewal order commences.

If you are not yet a subscriber to *BYTE*, the leading magazine for the creative home computer experimenter, don't delay . . . mail this coupon and start getting your own copies of this invaluable magazine, before new rates become effective.

USE THIS MONEY-SAVING COUPON TODAY

To new subscribers:

Read your first copy of *BYTE*, if it's everything you expected, honor our invoice. If it isn't, just write "CANCEL" across the invoice and mail it back. You won't be billed, and the first issue is yours.

Allow 6 to 8 weeks for processing.

BYTE Subscription Dept.79D • P.O. Box 361 • Arlington, Mass. 02174

PLEASE ENTER MY SUBSCRIPTION FOR: ☐ Renewal ☐ New Subscription

☐ One year $12 (12 issues) ☐ Two years $22 ☐ Three years $32

☐ Check enclosed (entitles you to bonus of one extra issue)

☐ Bill me ☐ Bill BankAmericard/Visa ☐ Bill Master Charge

Card Number: ☐☐☐☐☐☐☐☐☐☐☐☐☐☐ Expiration Date: ____________

Signature: ____________ Name *(please print)* ____________

Address: ____________

City: ____________ State/Country: ____________ Code: ____________

FOREIGN RATES FOR ONE YEAR: *(Please remit in U.S. Funds)*

☐ Canada or Mexico $17.50 (*Air delivered*) ☐ Europe $25 (Air delivered)

☐ All other countries except above: $25 *(Surface delivery)* Offer expires December 31, 1977

Air delivery available on request

Table 4: The 1024 by 2 byte gravity table used by the acceleration lookup function. Notice the maximum acceleration values of 999 in the upper left of the table which correspond to the position of the sun. The table can be rotated around its axis of symmetry to cover the entire screen in a minimum amount of memory. Lines drawn in the table illustrate several contours of roughly equal gravitational acceleration.

999	999	999	999	999	745	452	295	203	145	108	82	64	51	41	33	28	23	20	17	15	13	11	10	9	8	7	6	5	5	4	4
999	999	999	999	999	999	999	840	584	422	315	241	188	150	121	99	82	69	59	50	43	37	33	29	25	23	20	18	16	15	13	12
999	999	999	999	999	999	999	999	900	660	498	384	302	242	196	162	135	113	96	82	71	62	54	47	42	37	33	30	27	24	22	20
999	999	999	999	999	999	999	999	999	844	646	504	400	323	264	218	182	154	131	112	97	85	74	65	58	51	46	41	37	33	30	28
999	999	999	999	999	999	999	999	999	970	755	598	480	391	322	268	225	191	163	140	122	106	93	82	73	65	58	52	47	42	38	35
999	999	999	999	999	999	999	999	999	999	826	664	540	444	369	309	261	223	191	165	144	126	111	98	87	77	69	62	56	51	46	42
999	999	999	999	999	999	999	999	999	999	864	705	581	483	405	342	292	250	216	187	163	143	126	112	100	89	80	72	65	59	54	49
999	999	999	999	999	999	999	999	999	999	873	725	606	509	431	368	315	272	236	206	180	159	141	125	112	100	90	81	73	67	61	55
999	999	999	999	999	999	999	999	999	999	862	727	616	524	448	385	332	289	252	221	195	172	153	136	122	110	99	89	81	73	67	61
999	999	999	999	999	999	999	999	999	980	837	716	614	528	456	395	344	301	264	233	206	183	163	146	131	118	106	96	88	80	73	67
999	999	999	999	999	999	999	999	999	925	802	695	604	525	458	400	351	309	273	242	215	192	172	154	139	125	113	103	94	85	78	72
999	999	999	999	999	999	999	999	984	867	762	669	587	516	454	400	353	313	278	248	221	198	178	161	145	131	119	109	99	91	83	76
999	999	999	999	999	999	999	999	905	808	719	638	566	502	445	396	353	314	281	252	226	203	183	166	150	136	124	113	104	95	87	80
999	999	999	999	999	999	999	917	832	751	675	605	542	485	434	389	348	313	281	253	228	206	187	170	154	141	128	117	108	99	91	84
999	999	999	999	999	973	904	834	764	696	632	572	517	466	421	379	342	309	279	253	229	208	189	172	157	144	131	121	111	102	94	87
999	999	999	966	922	871	816	759	702	645	591	539	491	446	405	368	334	303	276	251	228	208	190	174	159	146	134	123	113	105	97	90
917	907	888	860	825	784	740	693	645	598	552	507	465	426	389	356	325	296	271	248	226	207	190	174	160	147	136	125	116	107	99	92
816	808	792	770	742	709	673	634	594	554	515	477	440	405	373	343	314	289	265	243	223	205	189	174	160	148	137	126	117	108	101	93
730	724	711	693	670	644	614	582	548	514	481	448	416	385	356	329	304	280	258	238	220	203	187	173	160	148	137	127	118	110	102	95
657	652	642	627	608	586	561	535	507	478	449	420	392	365	340	315	293	271	251	233	215	199	185	171	159	147	137	127	118	110	103	96
594	590	582	570	555	536	515	493	469	444	420	395	370	347	324	302	281	262	244	226	210	196	182	169	157	146	136	127	118	111	103	96
541	537	530	520	507	492	474	455	435	414	392	371	350	329	308	289	270	252	235	220	205	191	178	166	155	145	135	126	118	110	103	97
494	491	485	477	466	453	438	422	404	386	368	349	330	311	293	276	259	243	228	213	200	187	175	163	153	143	134	125	118	110	103	97
453	450	445	438	429	418	405	391	377	361	345	328	311	295	279	263	248	234	220	206	194	182	171	160	150	141	132	124	116	110	103	97
416	414	410	404	396	387	376	364	351	338	324	309	295	280	265	251	238	224	212	200	188	177	166	156	147	139	130	123	115	109	102	96
384	383	379	374	367	359	350	340	328	317	304	291	278	265	253	240	227	216	204	193	182	172	162	153	144	136	128	121	114	108	101	96
356	354	351	347	341	334	326	317	307	297	286	275	263	252	240	229	218	207	196	186	176	167	158	149	141	133	126	119	113	106	100	95
330	329	327	323	318	312	305	297	288	279	270	260	249	239	229	219	208	198	189	179	170	162	153	145	138	130	123	117	111	105	99	94
308	307	304	301	297	291	285	278	271	263	254	245	237	227	218	209	200	190	182	173	165	157	149	141	134	127	121	115	109	103	98	93
287	286	284	281	278	273	268	262	255	248	240	232	224	216	208	199	191	183	175	167	159	152	144	137	131	124	118	113	107	102	97	92
269	268	266	264	260	256	251	246	240	234	227	220	213	206	198	190	183	175	168	161	154	147	140	134	127	121	116	110	105	100	95	90
252	251	250	247	244	241	237	232	227	221	215	209	202	196	189	182	175	168	161	155	148	142	136	130	124	118	113	108	103	98	93	89

but this would be a pain in the asteroid. How about grouping all the variables together and defining a data layout common to all spaceships? The data includes coordinates, the address of the MBEAM instruction in the display file, orientation and about a million flag bits *[a bit of hyperbole . . .]*. The program knows which object block it's working with because it started out with the block start address in an index register, in this case the BC register pair of the 8080. All memory references then go through this index register. Don't let the fact that the 8080 microprocessor doesn't have real index registers bother you; you simply use assembler macros to invent your own indexed instructions as follows:

LOADX reg,offset: Loads data from object block relative location "offset" into register "reg."

STORX reg,offset: As above but stores.

LDBLX offset: Loads data from object block relative locations "offset" and "offset" + 1 into registers L and H respectively.

SDBLX offset: As above but stores.

Listing 3 shows the executive program and interrupt handlers.

System Functions

System functions are general purpose subroutines common to a number of applications programs. These include object move, acceleration look up, random number and a system call program which links with calling programs. To call one of the functions you invoke the macro SCALL n where n is 0 for move, 1 for acceleration and 2 for random number routine. This macro generates the code RST 7, DB n causing transfer of control through memory location 38 hexadecimal.

Looking at the first function, object move, you find a function fundamental to all video games. The function depends on a fixed layout of object coordinates at the head of the object block as shown in table 3. Calling object move updates these coordinates according to the following formulas:

$$X(n+1) = X(n) + X(n) - X(n-1) + Xacc$$
$$X(n\ \ 1) = X(n)$$
$$X(n) = X(n+1)$$

The same equations are used for Y.

The XY coordinates of the MBEAM command are updated with the new higher order values of X(n) and Y(n). It is important to note that the X and Y bytes in the MBEAM command are referenced to the lower left of the screen, but the 2 byte coordinates carried in the object block are referenced to the screen center.

These equations of motion represent what the math jocks call second order difference equations. Note the absence of multiplications and divisions. You can consider the difference between X(n) and X(n−1) to represent the X velocity of the object, and the corresponding Y difference to be the Y velocity. If you set both accelerations to zero and initialize the other coordinates appropriately, your object moves in a straight line with a constant velocity. If Y acceleration is a negative constant, the object falls in a parabolic trajectory like a thrown stone.

The object move function is sufficient for most video games, but not for Space War. For orbiting objects the accelerations are neither zero nor constant. Each point on the screen has unique values of X and Y acceleration. The acceleration lookup function finds these values for you, using the X(n) and Y(n) coordinates, so that when you alternately call it and the object move function, your spaceship or torpedo zips neatly around its orbit.

First generation Space War systems would calculate the accelerations each time they moved their objects according to the following formulas:

$$Xacc = cX/R^3 \text{ and } Yacc = cY/R^3$$

where c is a constant and $R = \sqrt{(X^2 + Y^2)}$. This requires a total of five multiplications, two divisions and one square root extraction for each update, clearly impossible for most of today's microprocessors.

Acceleration lookup, the heart of the Space War system, uses a 1 K by 2 byte table to find these accelerations, taking advantage of symmetry.

The third function, random number, returns an 8 bit pseudorandom number in the accumulator. This number is derived from a common shift register feedback scheme and has a repetition period of 255.

Listings of all system functions are shown in listing 4.

Gravity Table

Table 4 shows the 1 K by 2 byte gravity table used by the acceleration lookup function. The values were calculated by a FORTRAN program running on a larger computer. The table entries represent the absolute value of X acceleration in one quadrant. Y accelerations are found simply by transposing the indices. Because halving the distance from the sun causes the acceleration to increase by a factor of 4, the table can be magnified to produce more accurate values closer in. By proper shifting of indices and output values, the same table can be made to cover index ranges 0 to 16, 16 to 32, 32 to 64 and 64 to 128. Thus the maximum value of 999 applies only inside the sun.

Space War Applications Module

This module, occupying about 2 K bytes of user program memory, specifically defines the game of Space War. It is composed of constants, macro definitions, system linkages, the dispatch table, object blocks and programs. It interfaces with all the system modules described earlier. All programs execute once for each related object unless otherwise specified. Listing 5, the applications module, is divided into several sections which are described in a separate box entitled "A Guide to the Space War Applications Module."

Installation of Space War in Your System

The following steps might make it easier for you to get Space War up and running on an 8080 system:

- Make sure that your graphics output and button input work the way you think they do. Write short test programs if necessary.
- Modify the display processor module so it communicates with your particular graphics interface. The SHLD XYOUT instructions of the assembly listings are the ones you will want to look at carefully.
- Modify the ship fly and rotate programs in the applications module so that they read your buttons properly.
- Modify the keyboard handler (executive module) to accept interrupts from your keyboard. If you don't have keyboard interrupts you can periodically read your keyboard in the rotate program.
- Assemble all programs and load in the following order: display processor, executive, numerals, applications, system functions and gravity table.
- Temporarily eliminate the EXEC command in the display file, then start execution at hexadecimal location 100. This tests the display file and the display processor module, the correct result being a still picture which should make sense.
- Restore the EXEC command and deschedule all but the first task in the dispatch table, then start at 100 again. The system should remain in the still picture mode because not enough tasks have been enabled to support object motion.
- Enable tasks one at a time, thereby testing each. If the program bombs you will know exactly where to look.
- Fasten securely all loose objects in your computer room in anticipation of the large unruly crowds which will soon gather.

Conclusion

Listings 1-5 for Space War follow, concluding on page 111.

With Space War you will have come a step closer to making your computer The Ultimate Toy. In the process you will have learned some software fundamentals which will make you less afraid to put together some large systems of your own. Maybe you can think up some new and original video games which match or surpass the appeal of Space War. I'll be watching the pages of BYTE for the results.

A GUIDE TO THE SPACEWAR APPLICATIONS MODULE

Important Constants

All constants used in fine tuning the game are defined in this first section. These include the loading address, collision radii of the sun and torpedoes, acceleration constants and time-out values.

Macro Definitions

All macro definitions used in the module appear in this section. These include the graphics command macros and the load and store indexed macros.

System Linkages

Space War references two bytes in the system area: TIME, as described under the executive routine, and NUMS, the starting address of the list of graphics numeral addresses. The system modules, in turn, reference three points in the Space War module: the start of the display file, the dispatch table and the keyboard decode program.

Dispatch Table

The layout of this table is described in detail under the executive routine. The table is preceded by one byte indicating the number of entries, in this case 14. The ENTRY macro is used to define the entries.

Object Blocks

These include: ship 1, ship 2, torpedo 1, torpedo 2, score 1 and score 2.

Initialization Program

Executes: At start of game.
Function: Zeros score.
Schedules the ship start program to run immediately.
Deschedules itself.

Ship Start Program

Executes: When scheduled by initialization program.
A fixed time delay after a ship is destroyed.
Function: Puts spaceship subroutine call into display file.
Sets coordinates and orientation to starting values.
Deschedules itself.
Executes: On hyperspace return when ship is not set to destroy.
Function: Puts spaceship subroutine call into display file.
Clears hyperspace flag.
Deschedules itself.
Executes: On hyperspace return when ship is set to destroy.
Function: Puts explosion subroutine call into display file.

Ship Fly Program

Executes: On every clock cycle.
Function: Updates ship coordinates and acceleration values.
Adds acceleration and displays exhaust if the ACC button is held down.
Bounces ship off screen edge if necessary.
Replaces ship routine with explosion routine if:
ship is too close to any torpedo
ship is too close to the sun.
Releases torpedoes which get close enough to destroy spaceship.

Torpedo Fly Program

Executes: On every clock cycle.
Function: Updates torpedo coordinates and acceleration values.
Releases torpedo if:
torpedo hits edge of screen
torpedo gets too close to the sun.

Torpedo Fire Routine

Executes: When scheduled by the rotation program.
Function: Claims one torpedo belonging to the firing ship if it is not claimed.
Substitutes MDISP for MBEAM instruction at torpedo location in display file.
Computes initial position and velocity of torpedo from spaceship's coordinates and orientation.

Score Program

Executes: Every 16 clock cycles.
Function: Reads 1 byte binary coded decimal score value and converts it into addresses of two numeral subroutines which are inserted into display file.

Buttons Program

Executes: If hyperspace is entered (CCW and CW buttons both held down).
Function: Blanks spaceship.
Schedules ship start program after a time delay.
Indicates hyperspace destroy on random probability.
Executes: If CW button is held down.
Function: Increments orientation value.
Executes: If CCW button is held down.
Function: Decrements orientation value.
Executes: If FIRE button is down.

Function:	Schedules torpedo fire program.
Function during all executions:	Includes orientation value in appropriate PARAM instruction in display file. Releases any timed out torpedoes. Decrements counters for other torpedoes.

Keyboard Decode Program

Executes:	In response to interrupt from keyboard. Called from executive module.
Function:	Schedules initialization program twice when the C key is hit.

Listing 1: The display processor takes care of all the video display routines used for the game.

```
   5
; DISPLAY PROCESSOR MODULE
; COPYRIGHT 1976 D. KRUGLINSKI
;
; INITIALIZATION
STACK   EQU     0FFH    ;STACK ADDR
CINTC   EQU     0FFFAH  ;RTC CONTROL ADDRESS
GSTRT   EQU     100H    ;START ADDRESS
KINTC   EQU     0FFC0H  ;KBRD INT CONTROL ADDR
TIME    EQU     3FH     ;SYSTEM TIME
TOP     EQU     40H     ;DISPLAY FILE START ADDR
XYOUT   EQU     0FFFCH  ;CRT OUTPUT ADDR
        ORG     GSTRT
INIT:   LXI     D,0     ;ZERO XY (IN DE ALWAYS)
        LXI     H,TABLE
        SHLD    INCPT   ;INIT INCREMENT POINTER
        LXI     SP,STACK        ;INIT STACK POINTER
        MVI     A,10H
        STA     KINTC   ;TURN ON KB INT
        MVI     A,80H
        STA     CINTC   ;TURN ON RTC INTERRUPT
        EI
        LHLD    TOP
        SHLD    PNTR    ;INIT INSTRUCTION POINTER
; OPCODE DECODING
MLOOP:  LHLD    PNTR    ;ADDRESS OF OPCODE
        MOV     C,M
        MVI     B,0     ;OPCODE IN BC
        LXI     H,JMTAB ;BASE OF JUMP TABLE
        DAD     B       ;ADD OPCODE
        MOV     C,M
        INX     H
        MOV     H,M
        MOV     L,C     ;ADDR OF ROUTINE IN HL
        PCHL            ;JUMP TO IT
; JUMP TABLE FOR OPCODE PROCESSING
JMTAB:  DW      MBEAM   ;0
        DW      MDISP   ;2
        DW      LVEC    ;4
        DW      SVEC    ;6
        DW      PARAM   ;8
        DW      JUMP    ;A
        DW      JUMPS   ;C
        DW      RETS    ;E
        DW      EXEC    ;10H
        DW      SYNC    ;12H
; MOVE BEAM - DON'T DISPLAY POINT
MBEAM:  LHLD    PNTR
        INX     H
        MOV     D,M     ;GET X COORD
        INX     H
        MOV     E,M     ;GET Y COORD
        INX     H
        SHLD    PNTR    ;RESTORE POINTER
        JMP     MLOOP   ;GET ANOTHER INSTRUCTION
; MOVE BEAM AND DISPLAY POINT
MDISP:  LHLD    PNTR
        INX     H
        MOV     D,M
        INX     H
        MOV     E,M
        INX     H
        SHLD    PNTR
        XCHG            ;XY TO HL
        SHLD    XYOUT   ;WRITE TO CRT
        XCHG            ;SY TO DE
        JMP     MLOOP
; SET ORIENTATION AND SCALE
PARAM:  LHLD    PNTR
        INX     H
        MOV     C,M
        INX     H
        SHLD    PNTR
        MOV     L,C     ;NEW ORENT & SCALE IN HL
```

Listing 1, continued:

```
        MVI     H,0
        LXI     B,TABLE ;ADDRESS OF INCREMENT TABLE
        DAD     B       ;ADD ORENT & SCALF
        SHLD    INCPT   ;STORE IN INCREMENT POINTER
        JMP     MLOOP   ;GET ANOTHER INSTRUCTION
; JUMP TO A NEW LOCATION IN DISPLAY FILE
JUMP:   LHLD    PNTR
        INX     H
        MOV     A,M     ;1ST HALF OF ADDRESS
        INX     H
        MOV     H,M     ;2ND HALF OF ADDRESS
        MOV     L,A
        SHLD    PNTR    ;STORE IN INSTRUCTION POINTER
        JMP     MLOOP
; JUMP TO SUBROUTINE
JUMPS:  LHLD    PNTR
        INX     H
        MOV     C,M
        INX     H
        MOV     B,M     ;NEW ADDRESS IN BC
        INX     H
        PUSH    H       ;STORE OLD POINTER IN STACK
        MOV     H,B
        MOV     L,C
        SHLD    PNTR    ;ADDRESS IN INSTRUCTION POINTER
        JMP     MLOOP
; RETURN FROM SUBROUTINE
RETS:   POP     H       ; ESTORE POINTER FROM STACK
        SHLD    PNTH
        JMP     MLOOP
; SHORT VECTOR MODE
SVEC:   LHLD    PNTR    ;CURRENT INSTR INDEX
        INX     H
NEXT:   MOV     B,M     ;SHORT VECTOR INSTRUCTION
        INX     H
        SHLD    PNTR    ;RESTORE POINTER
        MOV     A,B
        ANI     7H      ;MASK DIRECTION BITS
        MOV     C,A     ;OFFSET IN C
        MOV     A,B     ;ORIG INST IN A
        MVI     B,0     ;ZERO IN B
        LHLD    INCPT   ;INCREMENT POINTER
        DAD     B       ;ADD DIRECTION OFFSET
        MOV     B,M     ;X INCREMENT
        INY     H
        INY     H
        MOV     C,M     ;Y INCREMENT
        XCHG            ;XY IN HL
        MOV     E,A     ;ORIGINAL INSTR
        ANI     70H     ;LENGTH BITS
        RRC
        RRC
        RRC
        RRC             ;SHIFT RIGHT 4
        MOV     D,A     ;IN D
        MOV     A,E     ;ORIGINAL INSTR
        ANI     8H      ;ON/OFF BIT
        JNZ     FLOOP   ;JUMP IF "OFF"
        SHLD    XYOUT   ;INITIAL DOT TO CRT
NLOOP:  MOV     A,B     ;Y INCREMENT
        ADD     H
        MOV     H,A
        MOV     A,C
        ADD     L
        MOV     L,A     ;NEW XY IN HL
        SHLD    XYOUT   ;WRITE TO CRT
        DCR     D
        JNZ     NLOOP   ;LOOP IF NOT DONE
        JMP     CKESC   ;CHECK "ESCAPE"
FLOOP:  MOV     A,B     ;X INCREMENT
        ADD     H       ;ADD TO X
        MOV     H,A
        MOV     A,C     ;Y INCREMENT
        ADD     L       ;ADD TO Y
        MOV     L,A     ;NEW XY IN HL
        DCR     D
        JNZ     FLOOP   ;LOOP IF NOT DONE
CKESC:  MOV     A,E     ;ORIGINAL INSTRUCTION
        XCHG            ;XY TO DE
        RLC             ;SET CARRY IF "ESCAPE"
        JC      MLOOP   ;MAIN LOOP IF SO
        LHLD    PNTR
        JMP     NEXT    ;GET MORE DATA IF NOT
; LONG VECTOR MODE
LVEC:   LHLD    PNTR
        INX     H
        MOV     B,M     ;NEW X POS
        INX     H
        MOV     A,M     ;NEW Y POS
        INX     H
        SHLD    PNTR
;
        MVI     C,0F8H  ;SPECIAL MASK FOR -
        SUB     E       ;DEL Y IN A
        JC      PLUSY   ;TEST SIGN OF DEL
        MVI     C,0     ;SPECIAL MASK FOR +
PLUSY:  RLC
        RLC
        RLC             ;DIVIDE BY 32
        MOV     H,A     ;SAVE IN H
        ANI     0F8H    ;SAVE 1ST 5 BITS
        MOV     L,A     ;L.O. YINC
        MOV     A,H     ;RESTORE
        ANI     7H      ;SAVE LAST 3 BITS
        XRA     C       ;XOR SPECIAL MASK
        MOV     H,A     ;H.O. YINC
        SHLD    YINC    ;STORE
;
        MOV     A,B     ;X POS
        MVI     C,0F8H
        SUB     D       ;DEL X IN A
        JC      PLUSX
        MVI     C,0
PLUSX:  RLC
        RLC
        RLC
        MOV     H,A
        ANI     0F8H
        MOV     L,A     ;L.O. XINC
        MOV     A,H
        ANI     7H
        XRA     C
        MOV     H,A     ;H.O. XINC
        SHLD    XINC
;
        MOV     B,D     ;X IN BC (H.O.)
        MVI     C,0
        MOV     D,E     ;Y IN DE (H.O.)
        MVI     E,0
        MVI     A,20H   ;32 POINTS IN VECT
;
LLOOP:  LHLD    YINC
        DAD     D
        XCHG            ;NEW Y IN DE
        LHLD    XINC
        DAD     B
        MOV     B,H     ;NEW X IN BC
        MOV     C,L
        MOV     L,D     ;XY IN HL (H.O.)
        SHLD    XYOUT   ;WRITE TO CRT
        DCR     A
        JNZ     LLOOP
;
        XCHG            ;RESTORE XY TO DE
        JMP     MLOOP
; TRANSFER CONTROL TO EXECUTIVE
EXEC:   LHLD    PNTR    ;BUMP POINTER
        INX     H
        SHLD    PNTR
        MVI     H,80H   ;BEAM TO SCREEN CENTER
        MVI     L,80H
        SHLD    XYOUT
        RST     3       ;XFER TO LOC 18H
        JMP     MLOOP   ;NEXT INSTRUCTION
; SYNCHRONIZE WITH REAL-TIME CLOCK
SYNC:   LHLD    PNTR    ;BUMP POINTER
        INX     H
        SHLD    PNTR
        MVI     H,80H   ;BEAM TO SCREEN CENTER
        MVI     L,80H
        SHLD    XYOUT
        LXI     H,TIME
        MOV     A,M     ;OLD TIME IN A
SLOOP:  CMP     M       ;OLD = NEW?
        JZ      SLOOP   ;YES - KEEP TRYING
        JMP     MLOOP   ;NO - NEXT INSTR
; TABLE FOR VECTOR ORIENTATION AND SCALING
TABLE:  DB      0
        DB      2
        DB      2
        DB      2
        DB      0
        DB      -2
        DB      -2
        DB      -2
        DB      0
        DB      2
        DB      2
        DB      2
        DB      0
        DB      -2
        DB      -2
        DB      -2
        DB      0
        DB      3
        DB      3
        DB      3
        DB      0
        DB      -3
        DB      -3
        DB      -3
        DB      0
        DB      3
        DB      3
        DB      3
        DB      0
        DB      -3
        DB      -3
        DB      -3
        DB      0
        DB      4
        DB      4
        DB      4
        DB      0
        DB      -4
        DB      -4
        DB      -4
        DB      0
        DB      4
        DB      4
        DB      4
        DB      0
        DB      -4
        DB      -4
        DB      -4
        DB      0
; THE FOLLOWING ARE STORAGE CELLS NOT TO BE IN PROM
; WORKING STORAGE
PNTR:   DW      0       ;POINTER TO NEXT INSTR
XINC:   DW      0       ;LVEC ONLY
YINC:   DW      0       ;LVEC ONLY
INCPT:  DW      0       ;POINTER TO INCREMENT IN TABLE
        END
↑Z*
```

FLIP OVER OUR FLOPPY

Only $750 from Peripheral Vision.

Peripheral Vision is a brand-new company that's dedicated to selling reasonably priced peripherals for various manufacturers' CPU's.

We think you'll flip over our first product.

It's a full-size floppy disk for the Altair-Imsai plug-in compatible S-100 BUS. And it's available for as low as $750.

Here are the features:

- 1 interface card supports 4 drives
- Stores over 300,000 bytes per floppy
- Bootstrap EPROM included—no more toggling or paper tape
- Completely S-100 plug-in compatible
- Interface cabling included
- Drive is from Innovex (the originator of the floppy concept)—assembled and tested
- Interface card design is licensed from Dr. Kenneth Welles and the Digital Group
- Disk operating system with file management system included on floppy
- Cabinet and power supply optional

Prices:	Kit	Assm.
Interface card kit and assembled and tested drive	$750	$850
Power supply— +24V at 2A	45	65
Cabinet—Optima, blue	—	85

Circle 387 on inquiry card.

Now, a little more about our company.

Peripheral Vision may be brand-new, but we have some old-fashioned ideas about how to run our business.

We know there are serious incompatibilities among the different manufacturers' peripherals and CPU's. We want to get them together. And, we want to bring significant new products to market—products consisting of everything from adaptation instructions/kits for hardware and software to major new products.

It's a tall order, but we feel we're up to the task. Peripheral Vision has already obtained a license from The Digital Group to adapt versions of some of their products to the S-100 BUS. And we're working on getting more from other companies.

Most important to our customers, Peripheral Vision is committed to helping you get along with your computer. We'll do all we can to make it easy.

Write us now for all the information on our company, our philosophy and our exciting line of products. And be prepared to flip over all of it.

P.O. Box 6267/Denver, Colorado 80206/ (303) 777-4292

Send me the works, and I just might flip over it!

Name________________________________

Address______________________________

City/State/Zip_________________________

Listing 2: Numeral display module contains the graphics dot code to draw the numerals 0 thru 9.

```
 $
;
; NUMERALS MODULE
; COPYRIGHT 1976 D. KRUGLINSKI
;
NSTRT   EQU     400H    ;LOAD ADDRESS
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; GRAPHICS MACROS
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
MBEAM   MACRO   X,Y
        DB      0
        DB      X
        DB      Y
        ENDM
;
MDISP   MACRO   X,Y
        DB      2
        DB      X
        DB      Y
        ENDM
;
LVEC    MACRO   X,Y
        DB      4
        DB      X
        DB      Y
        ENDM
;
SVEC    MACRO
        DB      6
        ENDM
SV      MACRO   LEN,DIR
        DB      DIR OR (LEN SHL 4)
        ENDM
SVF     MACRO   LEN,DIR
        DB      DIR OR (LEN SHL 4) OR 8H
        ENDM
SVE     MACRO   LEN,DIR
        DB      DIR OR (LEN SHL 4) OR 40H
        ENDM
SVEF    MACRO   LEN,DIR
        DB      DIR OR (LEN SHL 4) OR 48H
        ENDM
;
PARAM   MACRO   SCL,ORN
        DB      8
        DB      ORN OR (SCL SHL 4)
        ENDM
;
JUMP    MACRO   ADDR
        DB      0AH
        DW      ADDR
        ENDM
;
JUMPS   MACRO   ADDR
        DB      0CH
        DW      ADDR
        ENDM
;
RETS    MACRO
        DB      0EH
        ENDM
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; SYSTEM LINKAGES
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
        ORG     46H
        DW      ZERO
        DW      ONE
        DW      TWO
        DW      THREE
        DW      FOUR
        DW      FIVE
        DW      SIX
        DW      SEVEN
        DW      EIGHT
        DW      NINE
        ORG     NSTRT
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; NUMERALS 0-9
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
ZERO:   SVEC
        SV      6,0
        SV      4,2
        SV      6,4
        SV      4,6
        SVEF    6,2
        RETS
ONE:    SVEC
        SVF     4,2
        SV      6,0
        SVF     6,4
        SVEF    2,2
        RETS
TWO:    SVEC
        SVF     6,0
        SV      4,2
        SV      3,4
        SV      4,6
        SV      3,4
        SV      4,2
        SVEF    2,2
        RETS
THREE:  SVEC
        SV      4,2
        SV      6,0
        SV      4,6
        SVF     3,4
        SV      4,2
        SVF     3,4
        SVEF    2,2
        RETS
FOUR:   SVEC
        SVF     4,2
        SVF     3,0
        SV      4,6
        SV      3,0
        SVF     4,2
        SV      6,4
        SVEF    2,2
        RETS
FIVE:   SVEC
        SV      4,2
        SV      3,0
        SV      4,6
        SV      3,0
        SV      4,2
        SVF     6,4
        SVEF    2,2
        RETS
SIX:    SVEC
        SVF     3,0
        SV      4,2
        SV      3,4
        SV      4,6
        SV      6,0
        SV      4,2
        SVF     6,4
        SVEF    2,2
        RETS
SEVEN:  SVEC
        SVF     6,0
        SV      4,2
        SV      6,4
        SVEF    2,2
        RETS
EIGHT:  SVEC
        SV      6,0
        SV      4,2
        SV      6,4
        SV      4,6
        SVF     3,0
        SV      4,2
        SVF     3,4
        SVEF    2,2
        RETS
NINE:   SVEC
        SVF     4,2
        SV      6,0
        SV      4,6
        SV      3,4
        SV      4,2
        SVF     3,4
        SVEF    2,2
        RETS
        END
 ↑Z*
```

Listing 3: The interrupt handler and executive module covers the real time clock, keyboard scan routines, and dispatches object blocks on correct execution times.

```
 $
;
;INTERRUPT HANDLERS & EXECUTIVE MODULE
; COPYRIGHT 1976 D. KRUGLINSKI
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; SYSTEM LINKAGES
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
ESTRT   EQU     300H    ;LOAD ADDRESS
TADR    EQU     42H     ;ADDRESS OF # TASKS
;                        FOLLOWED BY DISPATCH TABLE
KINTA   EQU     10H     ;KBRD INT ADDR
CINTA   EQU     18H     ;RTC INT ADDR
KINTC   EQU     0FFC1H  ;KBRD INT CONROL ADDR
KDCOD   EQU     44H     ;KEYBOARD DECODER ADDR
TIME    EQU     3FH
        ORG     KINTA   ;KEYBOARD INTERRUPT
        JMP     KBENT   ;HANDLER ADDR
        ORG     CINTA   ;RTC INTERRUPT
        JMP     CENT    ;CLOCK HANDLER
        ORG     TIME
        DB      0
        ORG     KDCOD
        DW      KBDUM   ;DUMMY KB DECODER
        ORG     ESTRT
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; HANDLER FOR REAL-TIME CLOCK OR 'EXEC' COMMAND
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
CENT:   PUSH    PSW
        PUSH    B
        PUSH    D
        PUSH    H       ;SAVE ALL REGS
        LXI     H,TIME
        INR     M       ;INCREMENT TIME
        EI              ;ENABLE INTERRUPT
        CALL    EXEC    ;CALL EXECUTIVE
        POP     H       ;RESTORE REGS
        POP     D
        POP     B
        POP     PSW
        RET             ;RETURN TO GRAPHICS
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; HANDLER FOR KEYBOARD
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
KBENT:  PUSH    PSW
        PUSH    B
        PUSH    D
        PUSH    H
        LDA     KINTC   ;READ CHAR/CLEAR
        EI
        LXI     H,HRTN
        PUSH    H       ;SAVE RETURN ADDR
        LHLD    KDCOD   ;KEYBOARD DECODER
        PCHL            ;SIMULATED 'CALL'
HRTN:   POP     H
        POP     D
        POP     B
        POP     PSW
        RET             ;RTN TO INTERRUPTED PGM
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; EXECUTIVE -- SCANS DISPATCH TABLE, DISPATCHES
;  PROGRAM WITH OBJECT BLOCK WHEN TIMES MATCH
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
EXEC:   MOV     B,M     ;SAVE TIME IN B
        LHLD    TADR    ;# OF TASKS
        MOV     C,M
        INX     H       ;TOP OF DISPATCH TABLE
        LXI     D,5
ELOOP:  MOV     A,M
        ANA     B       ;MASK TIME FOR TASK
        INX     H
        CMP     M       ;COMPARE OBJECT TIME
        JNZ     NOTEQ   ;DISPATCH IF EQUAL
        PUSH    B       ;SAVE BC (TIME & #)
        INX     H
        MOV     C,M
        INX     H
        MOV     B,M     ;OBJECT BLOCK ADDR IN BC
        INX     H
        MOV     E,M
        INX     H
        MOV     D,M     ;PROGRAM ADDR IN DE
        INX     H
        PUSH    H       ;SAVE HL (POINTER IN LIST)
        LXI     H,ERTN
        PUSH    H       ;SAVE RETURN ADDRESS
        XCHG            ;PROGRAM ADDRESS IN HL
        PCHL            ;JUMP TO PROGRAM (CALL)
ERTN:   POP     H       ;RESTORE POINTER
        POP     B       ;RESTORE BC
        LXI     D,5
        JMP     ENDL    ;CONTINUE SCANNING TABLE
NOTEQ:  DAD     D       ;HL=HL+5
ENDL:   DCR     C       ;DECREMENT TASK CNT
        JNZ     ELOOP   ;TEST NEXT TASK
        RET             ;RETURN TO HANDLER
;
KBDUM:  RET             ;DUMMY KB DECODER
        END
 ↑Z*
```

the Processor Terminal.

A logical forward step in Microcomputer design

the **Processor Terminal.** A new design by **TEI** and look at what you get . . . a complete, self contained microcomputer system with display and mass storage, a full keyboard and plenty of slot space for additional boards. And that famous **TEI CVT** power supply that makes brownouts a thing of the past.

Display — A 15" high-resolution black and white video display with an optical filter face plate to reduce glare and improve type visability . . . **Keyboard** — Full upper and lower case ASCII detached keyboard with 8 programmable special function keys. Keyboard status indicators show computer BUSY or READY. And a 16-key numeric cluster pad set up calculator style . . . **Disk Drive & Controller** — A Shugart SA-400 mini-floppy disk drive. Soft sectored with a capacity of about 90 KB. IBM compatible format. Controller will handle 3 drives . . . **CPU** — 8080 based with a flexible design that allows you to implement a start up "jump to" operation to any dip switch selected byte address you choose. Merely turn on power or press RESET and you are off and running. Excellent for power failure automatic restart . . . **Memory** — 16K of static RAM memory. Low power chips. Selectable address assignment and memory protect features . . . **I/O** — 3P+3S input/output board. 3 parallel ports and 3 serial ports with selectable baud rates of 75 to 19,200. RS-232C and TTL outputs . . . **Video** — A video board provides the support for the video display functions . . . **Mainframe** — A 12 slot mainframe with a 17-amp CVT power supply, motherboard assembly, heavy duty aluminum cabinet, fan and washable filter. All edge connectors and card guides provided . . . **Software** — CP/M disk operating system and BASIC provided on disk.

the **Processor Terminal** (Model MCS-PT) fully assembled and tested is priced at $3495.00.

the **Processor Terminal** partially assembled (We build the cabinet, keyboard, monitor, power supply, disk drive and motherboard and you build the CPU, RAM, I/O, Video and Controller boards which we supply as kits). This partially assembled unit is priced at $2995.00.

TEI

SPECIAL SYSTEMS GROUP

MCS

MICROCOMPUTER SYSTEM

Contact your local TEI Dealer or if you are not near one of our dealers, write or call CMC Marketing Corp direct for more information.

CMC MARKETING CORP
7231 Fondren Rd, Houston, TX 77036 Telephone (713) 774-9526

See you at Pers. Comp. 10/27-29 Chicago Booth #53

Circle 388 on inquiry card.

Listing 4: The system function module takes care of all system calls, acceleration lookups, and random number generation.

```
   $
;
; SYSTEM FUNCTIONS MODULE
; COPYRIGHT 1976 D. KRUGLINSKI
; SYSTEM CALL, MOVE, ACCELERATION, RANDOM
;
MSTRT   EQU     ØC00H   ;LOAD ADDRESS
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; MACROS
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; COMPLEMENT HL
COMHL   MACRO
        MOV     A,H
        CMA
        MOV     H,A
        MOV     A,L
        CMA
        MOV     L,A
        INX     H
        ENDM
; LOAD HL INDEXED (BC=BASE)
LDBLX   MACRO   OFSET
        PUSH    D
        LXI     H,OFSET
        DAD     B
        MOV     E,M
        INX     H
        MOV     D,M
        XCHG
        POP     D
        ENDM
; STORE HL INDEXED (BC=BASE)
SDBLX   MACRO   OFSET
        PUSH    D
        XCHG
        LXI     H,OFSET
        DAD     B
        MOV     M,E
        INX     H
        MOV     M,D
        POP     D
        ENDM
; LOAD REG INDEXED (BC=BASE)
;  HL DESTROYED
LOADX   MACRO   REG,OFSET
        LXI     H,OFSET
        DAD     B
        MOV     REG,M
        ENDM
; STORE REG INDEXED (BC=BASE)
;  HL DESTROYED
STORX   MACRO   REG,OFSET
        LXI     H,OFSET
        DAD     B
        MOV     M,REG
        ENDM
;
        ORG     38H     ;RST 7 ADDRESS
        JMP     SYSCL
        ORG     MSTRT   ;LOAD ADDRESS
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; SYSTEM CALL FUNCTION
;       ENTERED ON RST 7 FOLLOWED BY FUNCTION #
;      DESTROYS HL ONLY
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
SYSCL:  POP     H       ;GET CALL # ADDR
        INX     H
        PUSH    H       ;RETURN ADDRESS
        DCX     H       ;CALL # ADDRESS
        PUSH    D       ;SAVE DE
        MOV     E,M
        MVI     D,Ø     ;CALL # IN DE
        LXI     H,CALTB ;CALL TABLE BASE
        DAD     D
        DAD     D       ;ADD #
        MOV     E,M
        INX     H
        MOV     D,M
        XCHG            ;ADDR IN HL
        POP     D       ;RESTORE DE
        PCHL            ;JUMP TO SUBROUTINE
CALTB:  DW      MOVE
        DW      ACCEL
        DW      RAND
        DW      Ø
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; GENERAL PURPOSE MOVE FUNCTION
; ASSUMES FIRST OBJECT BLOCK LOCATIONS AS FOLLOWS:
;       Ø       X(N)
;       1
;       2       X(N-1)
;       3
;       4       X ACCELERATION
;       5
;       6       Y(N)
;       7
;       8       Y(N-1)
;       9
;       A       Y ACCELERATION
;       B
;       C       POINTER TO 'MBEAM' INSTR
;       D
; CALL: SCALL Ø
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
MOVE:   PUSH    B       ;SAVE BC
        CALL    MOVI    ;UPDATE X
        CALL    MOVI    ;UPDATE Y
        LDAX    B       ;INST PNTR IN DE
        MOV     E,A
        INX     B
        LDAX    B
        MOV     D,A
        INX     D       ;X COORD ADDRESS
        POP     B       ;ORIG BC (TOP OF LIST)
        LOADX   A,1     ;X(N) H.O.
        ADI     80H     ;ZERO AT SCREEN CENTER
        STAX    D       ;X COORD
        INX     D       ;Y COORD ADDR
        LOADX   A,7     ;Y(N) H.O.
        ADI     80H     ;ZERO AT SCREEN CENTER
        STAX    D       ;Y COORD
        RET
; UPDATE EITHER X OR Y
MOVI:   PUSH    B       ;SAVE BC FOR X(N)
        LDAX    B       ;X(N) TO DE
        MOV     E,A
        INX     B
        LDAX    B
        MOV     D,A
        INX     B       ;-X(N-1) TO HL
        LDAX    B       ; AND DE TO NEW X(N-1)
        CMA
        MOV     L,A
        MOV     A,E
        STAX    B
        INX     B
        LDAX    B
        CMA
        MOV     H,A
        MOV     A,D
        STAX    B
        INX     H
        DAD     D       ;HL+DE+DE TO HL
        DAD     D
        INX     B       ;XACC TO DE
        LDAX    B
        MOV     E,A
        INX     B
        LDAX    B
        MOV     D,A
        DAD     D       ;HL+XACC TO HL
        POP     D       ;BC FOR X(N)
        MOV     A,L     ;HL TO NEW X(N)
        STAX    D
        INX     D
        MOV     A,H
        STAX    D
        INX     B       ;SETUP FOR NEXT BYTE
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; ORBITAL ACCELERATION LOOKUP FUNCTION
;       1Ø24-VALUE VERSION (2 BYTES/VALUE)
;       CALL:   SCALL   1
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
ATAB    EQU     ØEØØH   ;LOCATION OF ACC TABLE
XN      SET     Ø
XACC    SET     4
YN      SET     6
YACC    SET     ØAH
;   ADJUST X & Y FOR TABLE LOOKUP & SET SHIFT COUNT
ACCEL:  MVI     A,3
        STA     SHCNT   ;SET SHIFT COUNT (6 SHIFTS)
        LDBLX   XN      ;X VALUE
        MOV     A,H
        STA     HOXN    ;SAVE H.O. X
        CPI     Ø
        JP      POSX    ;ABS VALUE X
        COMHL
POSX:   XCHG            ;IN DE
        LDBLX   YN      ;Y VALUE
        MOV     A,H
        STA     HOYN    ;SAVE H.O. Y
        CPI     Ø
        JP      POSY    ;ABS VALUE OF Y
        COMHL
POSY:   MOV     A,H     ;IN HL
        ORA     D       ;COMBINE H.O. X & Y
        CPI     4ØH
        JP      GETAD   ;JUMP IF > OR = 4ØH
        DAD     H       ;DOUBLE Y IN HL
        XCHG
        DAD     H       ;DOUBLE X IN DE
        XCHG
        LDA     SHCNT
        DCR     A
        STA     SHCNT   ;DECREMENT SHIFT COUNT
        JNZ     POSY    ;LOOP IF > Ø
;   COMPUTE XACC TABLE OFFSET FROM X & Y (IN DE)
GETAD:  MOV     A,D     ;H.O. X
        RRC
        RRC
        RRC
        RRC             ;ROTATE RIGHT 4
        MOV     E,A
        ANI     7H
        MOV     D,A     ;H.O. TABLE OFFSET
        MOV     A,E
        ANI     ØCØH
        MOV     E,A
        MOV     A,H     ;H.O. Y
        RRC
        ANI     3EH
        ORA     E
        MOV     E,A     ;L.O. TABLE OFFSET
```

Talk to your computer for $299 with SpeechLab.™

Use SpeechLab to directly control any S-100 Bus Computer such as Sol, IMSAI, Altair and so on. SpeechLab can teach you almost as much as the Bell Laboratories know about voice recognition, voice control and computer input.

SpeechLab digitizes and extracts data from speech wave form and applies pattern matching techniques to recognize the vocal input. Response is real time. The system features 64 bytes of storage per spoken word and can handle up to a 64 word vocabulary. And recognition after very little practice is 95 percent or better.

$299* assembled and tested

When we talk price everybody's skeptical. And why not? We give you a complete hardware/software system, a 275 page laboratory manual, 95 page hardware manual and high fidelity microphone.

The lab manual includes 35 graded experiments with over 100 tables and graphs. In fact, it's the only introductory volume on speech recognition currently available.

Software includes SpeechBasic Basic programming language in source and paper tape, assembly language speech recognition program in source and paper tape, hardware self-test program in source and paper tape. SpeechBasic plot, correlation, recognition and advanced recognition programs are offered in source.

Hard to believe, you bet. True? A Los Angeles customer says, "I love your kit!!! I have 40 boards and 2 IMSAI's and your kit was the best documented of them all. I love the way you integrated the software and hardware together. I love your lab manual."

We loved those comments. They tell the story better than we ever could. The LA customer did ask who the founders of the firm were so he could relate better. They're a couple of gifted young engineers who got tired of the big firm, big technology trip and decided to take a chance with a better idea.

You can't get better quality
You can't get more performance

Sure, more complex, higher price equipment is available for about 50 times more money. It won't do much more than you can do with SpeechLab. And the quality and state-of-the-art engineering can't be any better. We use CMOS design for low power and ultimate reliability.

See SpeechLab at your nearby computer store

Selected computer stores have SpeechLab on display. Visit your nearest. If he doesn't have it, ask him to contact us or simply write us directly.

*Available in kit form for slightly less.

Circle 389 on inquiry card.

Heuristics, Inc.
Box B, 900 N. San Antonio Rd.
Los Altos, CA 94022, Phone (415) 948-2542

☐ Send me SpeechLab. I enclose $299.
California residents add sales tax.
☐ Send me more information.

☐ Master Charge
☐ BankAmericard (Visa)
Acct. No.________
Date Card Expires________
Interbank No.________
(Master Charge only)

Name ________
Street ________
City________ State________ Zip________

In Canada:
Trintronics Ltd.
186 Queen St. W.
Toronto, Canada M5V 1Z1

Heuristics INC.

Listing 4, continued:

```
        PUSH    D       ;SAVE OFFSET FOR LATER
        CALL    RETRV   ;GET ACC VAULE
;   MAKE XACC SIGN AGREE WITH -Y COORD
        LDA     HOXN    ;H.O. Y VALUE
        CPI     0       ;TEST SIGN
        JM      XMINS
        COMHL           ;POS - COMP XACC
XMINS:  SDBLX   XACC    ;STORE IN OBJ BLK
;   COMPUTE YACC TABLE OFFSET FROM YACC OFFSET
        POP     H       ;YACC OFFSET IN HL
        MOV     A,L
        RRC
        RRC
        RRC
        MOV     E,A
        ANI     7H
        MOV     D,A     ;H.O. TABLE OFFSET
        MOV     A,E
        ANI     0E0H
        MOV     E,A
        MOV     A,L
        ANI     0C0H
        ORA     H
        RLC
        RLC
        RLC
        ORA     E
        MOV     E,A     ;L.O. OFFSET
        CALL    RETRV   ;GET YACC VALUE
;   MAKE YACC SIGN AGREE WITH -Y COORD
        LDA     HOYN
        CPI     0
        JM      YMINS
        COMHL
YMINS:  SDBLX   YACC    ;STORE YACC IN OBJ BLK
        RET             ;RETURN TO CALLING PROGRAM
;
; SUBROUTINE TO RESTORE ACC VALUE FROM TABLE TO HL
;   AND ADJUST ACC BY SHIFT COUNT
;   INPUT: OFFSET IN DE
RETRV:  LXI     H,ATAB  ;TABLE BASE
        DAD     D       ;ADD OFFSET
        MOV     E,M
        INX     H
        MOV     D,M     ;DATA IN DE
        PUSH    B
        LDA     SHCNT   ;SAVE SHIFT COUNT IN B
        MOV     B,A
        CPI     0
        JZ      EXIT    ;SHIFT COUNT=0
;   SHIFT DE RIGHT 2 (DE < 1000)
        MOV     A,D
        RRC
        RRC
        MOV     D,A
        MOV     A,E
        RRC
        RRC
        PUSH    PSW     ;SAVE CARRY FOR ROUNDING
        ANI     3FH
        ORA     D
        MOV     E,A
        MVI     D,0
;   DECREMENT & TEST SHIFT COUNT
LOOP:   DCR     B
        JZ      ROUND   ;DONE SHIFTING
;   SHIFT E RIGHT 2 (D=0)
        POP     PSW     ;SAVE STACK
        MOV     A,E
        RRC
        RRC
        PUSH    PSW     ;SAVE CARRY
        ANI     3FH
        MOV     E,A
        JMP     LOOP
ROUND:  POP     PSW     ;RESTORE CARRY FROM LAST SFT
        MOV     A,E     ;L.O. ACC
        ACI     0       ;ROUND
        MOV     E,A     ;RESTORE E
EXIT:   XCHG            ;ACC IN HL
        POP     B       ;RESTORE INDEX REG
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; RANDOM NUMBER FUNCTION
;   GENERATES A NEW RANDOM NUMBER IN A AND 'RND'
;   CALL:       SCALL   2
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
RAND:   LXI     H,RND
        MOV     A,M
        ANI     8EH     ;FEEDBACK MASK,CLEAR CARRY
        JPE     CLEAR   ;XOR FEEDBACK BITS
        CMC             ;SET CARRY IF XOR TRUE
CLEAR:  MOV     A,M     ;RESTORE RND
        RAL             ;SHIFT IN CARRY
        MOV     M,A     ;IN MEMORY
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; WORKING STORAGE
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
RND:    DB      23H     ;RANDOM NUMBER
HOXN:   DB      0       ;H.O. Y VALUE
HOYN:   DB      0       ;H.O. Y VALUE
SHCNT:  DB      0       ;ACC SHIFT COUNT
        END
```

Listing 5: The applications module, which is specified for a particular program, is described in detail on page 96.

```
;
; SPACE WAR APPLICATIONS MODULE
; COPYRIGHT 1976 D. KRUGLINSKI
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; IMPORTANT CONSTANTS
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
ADIRT   EQU     500H    ;LOAD ADDRESS
BADRI   EQU     0FFFEH  ;BUTTON READ ADDRESS
DLN     EQU     7       ;COLLISION RAD OF GUN
INTVL   EQU     10H     ;DURATION OF EXPLOSION
EPSLN   EQU     5H      ;COLLISION RAD OF TORPEDO
ACON    EQU     8       ;ACCELERATION CONSTANT
RMAX    EQU     12      ;TORPEDO TIMEOUT
VCON    EQU     300H    ;TORPEDO RELATIVE VELOCITY
HDLY    EQU     20H     ;HYPERSPACE EXIT DELAY
LIMIT   EQU     7AH     ;SCREEN EDGE
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; MACROS
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
; GRAPHICS MACROS
DBEAM   MACRO   X,Y
        DB      0
        DB      X
        DB      Y
        ENDM
;
MDIEP   MACRO   X,Y
        DB      2
        DB      X
        DB      Y
        ENDM
;
LVEC    MACRO   X,Y
        DB      4
        DB      X
        DB      Y
        ENDM
;
SVEC    MACRO
        DB      6
        ENDM
SV      MACRO   LEN,DIR
        DB      DIR OR (LEN SHL 4)
        ENDM
SVF     MACRO   LEN,DIR
        DB      DIR OR (LEN SHL 4) OR 8H
        ENDM
SVE     MACRO   LEN,DIR
        DB      DIR OR (LEN SHL 4) OR 80H
        ENDM
SVEF    MACRO   LEN,DIR
        DB      DIR OR (LEN SHL 4) OR 88H
        ENDM
;
PARAM   MACRO   SCL,ORN
        DB      8
        DB      ORN OR (SCL SHL 4)
        ENDM
;
JUMP    MACRO   ADDR
        DB      0AH
        DW      ADDR
        ENDM
;
JUMPS   MACRO   ADDR
        DB      0CH
        DW      ADDR
        ENDM
;
RETS    MACRO
        DB      0EH
        ENDM
;
EXEC    MACRO
        DB      10H
        ENDM
;
; ENTRY MACRO FOR DISPATCH TABLE
ENTRY   MACRO   MASK,TIME,OBJ,PROG
        DB      MASK
        DB      TIME
        DW      OBJ
        DW      PROG
        ENDM
;
; SYSTEM CALL MACRO
SCALL   MACRO   MODN
        RST     7
        DB      MODN
        ENDM
;
; MACRO TO CREATE 12-BYTE COORDINATE BLK
COORD   MACRO   XN,XM,XAC,YN,YM,YAC
        DW      XN
        DW      XM
        DW      XAC
        DW      YN
        DW      YM
        DW      YAC
        ENDM
;
; MACRO TO LOAD REG INDEXED (BC=BASE)
;   DESTROYS HL
LOADX   MACRO   REG,OFSET
```

How to get more bytes for your buck.

First, come to Micropolis. Where you get a MetaFloppy™ instead of an ordinary 5¼″ floppy disk system.

Then, forget all things average.

Because the Micropolis MetaFloppy delivers over four times the 70K (or so) bytes of storage you'd expect. A whopping 315K bytes per disk to be exact. All formatted—not raw. And all on a single disk side.

But that's not all. Because Micropolis delivers everything else you need, too. Power Supply. Interface cable. Extended BASIC software. Plus an S-100 controller for your Altair 8800, IMSAI 8080, COMPAL 80, or the like.

There's even a dual drive model for maximum convenience. And savings.

The Micropolis secret: We build *every* component we sell. In fact, we're the only totally integrated 5¼″ floppy disk system builder around. And where there's no middleman, there's no middleman to pay.

So, before you swallow someone else's price/performance story, find out how much your buck really buys and how much faster it buys it—at Micropolis.

For a free brochure and price sheet, write: Micropolis Corporation, 7959 Deering Ave., Canoga Park, California 91304 or phone (213) 703-1121.

MICROPOLIS™

Circle 390 on inquiry card.

Listing 5, continued:

```
        LXI     H,OFSET
        DAD     B
        MOV     REG,M
        ENDM
;
; MACRO TO STORE REG INDEXED (BC=BASE)
;   DESTROYS HL
STORX   MACRO   REG,OFSET
        LXI     H,OFSET
        DAD     B
        MOV     M,REG
        ENDM
;
; MACRO TO LOAD 2 BYTES FROM OFSET
; INTO HL (BC=BASE ADDR)
LDBLX   MACRO   OFSET
        PUSH    D
        LXI     H,OFSET
        DAD     B
        MOV     E,M
        INX     H
        MOV     D,M
        XCHG
        POP     D
        ENDM
;
; MACRO TO STORE 2 BYTES IN OFSET
; FROM HL (BC=BASE ADDR)
SDBLX   MACRO   OFSET
        PUSH    D
        XCHG
        LXI     H,OFSET
        DAD     B
        MOV     M,E
        INX     H
        MOV     M,D
        POP     D
        ENDM
;
;MACRO TO TAKE ABSOLUTE VALUE OF A
ABS     MACRO
        CPI     0
        JP      POS     ;JUMP IF A POS
        CMA             ;COMPLEMENT A
POS:
        ENDM
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; LINKAGE BETWEEN SYSTEM AND APPLICATION
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
TIME    EQU     3FH
NUMS    EQU     46H
        ORG     40H
        DW      START   ;DISPLAY FILE
        ORG     42H
        DW      NTSK    ;# TASKS + LIST
        ORG     44H
        DW      KDCD    ;KEYBOARD DECODE
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; DISPATCH TABLE + NUMBER OF OBJECTS
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
        ORG     ASTRT
NTSK:   DB      14      ;# TASKS
; DISPATCH TABLE
IENT1:  ENTRY   0,0,SH1,INIT
IENT2:  ENTRY   0,0,SH2,INIT
PENT1:  ENTRY   0,1,SH1,SSTRT
PENT2:  ENTRY   0,1,SH2,SSTRT
        ENTRY   0,0,SH1,SFLY
        ENTRY   0,0,SH2,SFLY
        ENTRY   0,0,BU1,BFLY
        ENTRY   0,0,BU2,BFLY
GENT1:  ENTRY   0,1,SH1,FIRE
GENT2:  ENTRY   0,1,SH2,FIRE
        ENTRY   0FH,2,SC1,SCORE
        ENTRY   0FH,8,SC2,SCORE
        ENTRY   7H,2,SH1,ROT
        ENTRY   7H,6,SH2,ROT
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; OBJECT BLOCKS
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;   SHIP OBJECT BLOCKS
SH1:    COORD   0,0,0,0,0,0         ;DYNAMIC COORDS
        DW      PPOS1   ;PNTR TO MBEAM INSTR
;   START POSTN
        COORD   -6F00H,-7000H,0,6000H,6000H,0
        DW      BU1     ;PNTR TO TORPEDO
        DW      PEXH1+1 ;CALL EXHAUST/ZERO
        DW      SHIP1   ;SHIP SUB
        DW      PSUB1+1 ;CALL SHIP/EXPLO
        DW      PDIR1+1 ;SHIP ORIENT
        DW      PENT1   ;TASK POINTER
        DW      GENT1   ;FIRE ENTRY
        DW      SC1     ;SCORE
        DB      01H     ;ACC BUTTON MASK
        DB      02H     ;FIRE MASK
        DB      04H     ;CW MASK
        DB      08H     ;CCW MASK
        DB      0       ;INHIBIT FIRE
        DB      2       ;ORIENTATION
        DW      BADR1   ;BUTTON ADDRESS
        DB      0CH     ;HYPERSPACE MASK
        DB      0       ;HYPERSPACE FLAG
        DB      0       ;INITIAL ORIENTATION
        DW      IENT1   ;BACK PNTR TO INIT PGM
SH2:    COORD   0,0,0,0,0,0
        DW      PPOS2
        COORD   6F00H,7000H,0,-6002H,-6000H,0
        DW      BU2
        DW      PEXH2+1
        DW      SHIP2
        DW      PSUB2+1
        DW      PDIR2+1
        DW      PENT2
        DW      GENT2
        DW      SC2
        DB      10H
        DB      20H
        DB      40H
        DB      80H
        DB      0
        DB      0
        DW      BADR1
        DB      0C0H
        DB      0
        DB      4
        DW      IENT2
;   TORPEDO OBJECT BLOCKS
BU1:    COORD   0,0,0,0,0,0     ;DYN COORDS
        DW      BPOS1   ;PNTR TO MDISP INSTR
        DW      BU2     ;PNTR TO NEXT TORPEDO
        DB      0       ;COUNTDOWN SINCE FIRING
BU2:    COORD   0,0,0,0,0,0
        DW      BPOS2
        DW      0
        DB      0
;   SCORE OBJECT BLOCKS
SC1:    DB      0       ;BINARY SCORE VAL
        DW      SC11+1  ;1ST DIGIT
        DW      SC12+1  ;2ND DIGIT
SC2:    DB      0
        DW      SC21+1
        DW      SC22+1
; WORKING STORAGE LOCATION
ORSAV:  DW      0
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; DISPLAY FILE
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
START:  EXEC
        MBEAM   80H,20H ;SUN
        PARAM   2,0
        JUMPS   SUNSB
PPOS1:  MBEAM   0,0E0H
PDIR1:  PARAM   1,0
PEXH1:  JUMPS   NULL
PSUB1:  JUMPS   SHIP1
PPOS2:  MBEAM   0,0C0H
PDIR2:  PARAM   1,0
PEXH2:  JUMPS   NULL
PSUB2:  JUMPS   SHIP2
BPOS1:  MBEAM   0,0
BPOS2:  MBEAM   0,0
        PARAM   1,0
        MBEAM   1,1
SC11:   JUMPS   NULL
SC12:   JUMPS   NULL
        MBEAM   0D0H,1
SC21:   JUMPS   NULL
SC22:   JUMPS   NULL
        JUMP    START
;
SUNSB:  SVEC
        SVF     6,0
        SV      3,2
        SV      3,3
        SV      6,4
        SV      3,5
        SV      6,6
        SV      3,7
        SV      6,2
        SV      3,1
        SVE     3,2
        RETS
;
SHIP1:  SVEC
        SVF     3,2
        SV      2,5
        SV      4,4
        SV      2,2
        SVF     2,6
        SVF     4,2
        SVF     2,1
        SV      2,3
        SV      4,4
        SVE     2,6
        RETS
SHIP2:  SVEC
        SV      3,2
        SV      2,5
        SV      4,4
        SV      2,2
        SVF     2,6
        SVF     4,0
        SVF     2,1
        SV      2,3
        SV      4,4
        SVE     2,5
        RETS
EXPLO:  PARAM   2,0
        SVEC
        SVF     3,0
        SV      5,4
        SVF     3,2
        SV      5,7
        SVF     3,4
        SV      5,2
        SVF     3,0
        SVE     6,5
        RETS
NULL:   RETS
```

Listing 5, continued:

```
EXHST:  SVEC
        SV      7,4
        SVEF    7,0
        RETS
;
; *********** PROGRAMS ************
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    INITIALIZATION PROGRAM
;  SCHEDULED AT START & BY CTL C
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
SCPNT   SET     28H
IEPNT   SET     35H
PEPNT   SET     24H
INIT:   LDBLX   SCPNT   ;GET SCORE ADDR
        MVI     M,0     ;ZERO SCORE
        LDBLX   PEPNT   ;GET ADDR OF SHIP ST
        INX     H       ;TIME
        MVI     M,0     ;SCHED SHIP START
        LDBLX   IEPNT   ;ADDR OF INIT ENTRY
        INX     H
        MVI     M,1     ;DESCHED SELF
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    SHIP START PROGRAM
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
DC      SET     0
STCRD   SET     0EH
FBPNT   SET     1AH
EXPNT   SET     1CH
PLPNT   SET     1EH
PCPNT   SET     20H
PDPNT   SET     22H
ORENT   SET     2FH
HFLAG   SET     33H     ;HYPERSPACE FLAG
IORNT   SET     34H     ;INITIAL ORIENTATION
; SELECT ACCORDING TO HFLAG
SSTRT:  LOADX   A,HFLAG
        CPI     0
        JNZ     HYPR
;   CASE--H=0--NORMAL START--NOT HYPERSPACE
        CALL    STRT    ;SHIP INST, DESCHED
        CALL    BEGIN   ;ORENT=0,START POS
        RET
HYPR:   JM      HDEST
;   CASE--H=1--HYPERSPACE RETURN--NO DESTROY
        CALL    STRT
        CALL    HCORD   ;RANDOM COORDINATES
        RET
;   CASE--H=-1--HYPERSPACE RETURN--DESTROY
HDEST:  CALL    DSTRY
        CALL    HCORD   ;RAND COORDS
        RET
; ENDSELECT
;
STRT:   LDBLX   PLPNT   ;NORMAL START
        XCHG            ;SHIP SUB ADDR IN DE
        LDBLX   PCPNT   ;CALL ADDR IN HL
        MOV     M,E     ;SHIP SUB -> CALL
        INX     H
        MOV     M,D
        LDBLX   PEPNT   ;ENTRY ADDR
        MVI     M,0     ;MASK
        INX     H
        MVI     M,1     ;DESCHED SELF
        RET
;
BEGIN:  LOADX   A,IORNT ;SET ORIENTATION
        STORX   A,ORENT
        LXI     D,STCRD ;START COORD
        DAD     B       ;  ADDR IN HL
        MVI     D,12    ;12 BYTES TO MOVE
PXLOP:  MOV     A,M     ; FROM START COORDS
        STAX    B       ; TO DYN COORDS
        INX     H
        INX     B
        DCR     D
        JNZ     PXLOP
        RET
;
HCORD:  SCALL   2       ;RANDOM BYTE IN A
        STORX   A,XN+1  ;H.O. X COORD
        STORX   A,XNM+1
        SCALL   2
        STORX   A,YN+1  ;H.O. Y COORD
        STORX   A,YNM+1
        SCALL   2
        STORX   A,XN    ;L.O. X (VELOCITY)
        SCALL   2
        STORX   A,XNM
        SCALL   2
        STORX   A,YN    ;L.O. Y
        SCALL   2
        STORX   A,YNM
        MVI     A,0     ;ZERO HYPERSPACE FLAG
        STORX   A,HFLAG
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    SHIP FLY PROGRAM
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
XN      SET     0H
YN      SET     6H
XNM     SET     2H
YNM     SET     8H
XACC    SET     04H
YACC    SET     0AH
INPNT   SET     0CH
NBPNT   SET     0EH
RCNT    SET     10H     ;TORPEDO TIMEOUT
FIPNT   SET     26H
ACCM    SET     2AH
FIREM   SET     2BH
CWM     SET     2CH
CCWM    SET     2DH
HYPM    SET     32H     ;HYPERSPACE MASK
FINH    SET     2EH
BAPNT   SET     30H
ACON1   EQU     (5*ACON)/7
SFLY:   SCALL   0       ;MOVE SHIP/EXPLO
        SCALL   1       ;ACCELERATE
; TEST BUTTONS ONLY IF NOT IN HYPERSPACE
        LOADX   A,HFLAG
        CPI     0
        RNZ
;   CHECK IF ACC BUTTON ON
        LDBLX   BAPNT   ;BUTTON ADDR
        MOV     D,M     ;BUTTON WORD
        LOADX   A,ACCM  ;ACC MASK
        ANA     D
        LXI     D,NULL  ;NO EXHAUST
        JZ      XHST    ;NO ACCELERATION
        MVI     D,0
        LOADX   E,ORENT
        LXI     H,XATAB
        DAD     D
        DAD     D       ;X ACC ADDR FOR ORENT
        MOV     E,M     ;XACC L.O.
        INX     H
        MOV     D,M     ;XACC H.O.
        LDBLX   XACC    ;ADD TO ORIGINAL
        DAD     D
        SDBLX   XACC
;   SAME LOGIC FOR YACC
        MVI     D,0
        LOADX   E,ORENT
        LXI     H,YATAB
        DAD     D
        DAD     D
        MOV     E,M
        INX     H
        MOV     D,M
        LDBLX   YACC
        DAD     D
        SDBLX   YACC
        LXI     D,EXHST ;EXHAUST
XHST:   LDBLX   EXPNT   ;INSERT EXHAUST/ZERO
        MOV     M,E     ;IN DISPLAY FILE
        INX     H
        MOV     M,D
;   TEST IF X NEAR EDGE
ETST:   LOADX   A,XN+1
        ABS
        CPI     LIMIT
        CP      SWAPX   ;YES
;   TEST IF Y NEAR EDGE
        LOADX   A,YN+1
        ABS
        CPI     LIMIT
        CP      SWAPY   ;YES
;   SHIP HIT SUN?
        LOADX   A,XN+1
        ABS
        CPI     SUN     ;H.O. Y < SUN RADIUS
        JP      BTST
        LOADX   A,YN+1
        ABS
        CPI     SUN     ;H.O. Y < SUN RADIUS
        JP      BTST
        CALL    DSTRY
        RET
;   TEST FOR CLOSE TORPEDOS
BTST:   LOADX   D,DC+1  ;HO X POS OF SHIP
        LOADX   E,DC+7  ;HO Y POS OF SHIP
        PUSH    B       ;SAVE SHIP BLK PNTR
        LXI     H,BUL   ;FIRST TORPEDO BLK PNTR
FLOOP:  MOV     B,H     ;HL -> BC
        MOV     C,L
        LOADX   A,RCNT  ;TORPEDO CLAIMED?
        CPI     0       ;TEST FOR ZERO
        JZ      NXBUL   ;NEXT TORPEDO
NBTST:  LOADX   A,DC+1  ;HO X POS OF TORPEDO
        SUB     D
        ABS
        CPI     EPSLN
        JP      NXBUL   ;NOT CLOSE ENOUGH
        LOADX   A,DC+7  ;HO Y POS OF TORPEDO
        SUB     E       ;SUBT SHIP POS
        ABS
        CPI     EPSLN
        JM      HIT     ;BOTH CLOSE ENOUGH
NXBUL:  LDBLX   NBPNT   ;NEXT TORPEDO PNTR
        SUB     A       ;ZERO A
        CMP     H       ;CHECK OF H 0
        JNZ     FLOOP   ;NO
        CMP     L
        JNZ     FLOOP   ;L NOT 0
        POP     B       ;SAVE STACK
        RET             ;BOTH H&L ZERO
HIT:    LDBLX   INPNT   ;TORPEDO POSTION
        MVI     M,0     ;NBEAM INSTR (BLANK)
        MVI     A,2     ;RELEASE TORPEDO
        STORX   A,RCNT
        POP     B       ;RESTORE SHIP BLK
        CALL    DSTRY
        RET
;   SUBROUTINE TO INCREMENT SCORE, SCHED
;   SHIP START AND REPLACE SHIP WITH EXPLOSION
;   ALSO USED IN HYPERSPACE PROCESSING
DSTRY:  LDBLX   SCPNT   ;RESTORE SCORE ADDR
        MOV     A,M     ;INCREMENT SCORE
        INR     A
        DAA
```

Listing 5, continued:

```
        MOV     M,A
;  SCHED SHIP START
        LDBLX   PEPNT   ;SHIP ENTRY ADDR
        MVI     M,-1    ;MASK = -1
        LDA     TIME
        ADI     INTVL
        INX     H
        MOV     M,A     ;TIME+INTVAL TO PL ST
;  REPLACE SHIP WITH EXPLOSION
        LXI     D,EXPLO ;EXPLO SUB
        LDBLX   PCPNT   ;SUB CALL
        MOV     M,E
        INX     H
        MOV     M,D
        RET
;
SWAPX:  LDBLX   XN      ;SWAP X COORDS
        XCHG
        LDBLX   XNM
        SDBLX   XN
        XCHG
        SDBLX   XNM
        RET
;
SWAPY:  LDBLX   YN      ;SWAP Y COORDS
        XCHG
        LDBLX   YNM
        SDBLX   YN
        XCHG
        SDBLX   YNM
        RET
;
;  ACCELERATION TABLE
XATAB:  DW      0
        DW      ACON1
YATAB:  DW      ACON
        DW      ACON1
        DW      0
        DW      -ACON1
        DW      -ACON
        DW      -ACON1
        DW      0
        DW      ACON1
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;  TORPEDO FLY PROGRAM
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
BFLY:   SCALL   1       ;ACCELERATION LOOKUP
        SCALL   2       ;MOVE TORPEDO
;  TEST IF X NEAR SCREEN EDGE
        LOADX   A,DC+1  ;HO X POS OF TORPEDO
        ABS             ;ABS VALUE OF Y
        CPI     LIMIT   ;SUBTRCT LIMIT
        JP      BLANK   ;NEAR EDGE IF + OR 0
;  TEST IF Y NEAR EDGE
        LOADX   A,DC+7  ;HO Y POS OF TORPEDO
        ABS
        CPI     LIMIT
        JP      BLANK   ;NOT NEAR EDGE
;  TORPEDO HIT SUN
        LOADY   A,XN+1
        ABS
        CPI     SUN
        RP              ;RETN IF NOT
        LOADX   A,YN+1
        ABS
        CPI     SUN
        RP              ;RETURN IF NOT
BLANK:  LDBLX   INPNT   ;MBEAN/MDISP INST
        MVI     M,0     ;MBEAN
        MVI     A,0
        STORX   A,RCNT  ;RELEASE TORPEDO
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;  TORPEDO FIRE PROGRAM
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
GEPNT   SET     26H
RCON    EQU     EPSLN*140H
VCON1   EQU     (5*VCON)/7
FIRE:   LDBLX   GEPNT   ;DISPATCH TABLE ENTRY FOR SHIP
        INX     H
        MVI     M,1     ;DESCHEDULE SELF
        LOADX   A,ORENT
        ADD     A       ;DOUBLE ORIENTATION
        MOV     L,A     ;SAVE FOR LATER
        MVI     H,0
        SHLD    ORSAV
        LDBLX   FRPNT   ;PNTR TO 1ST TORPEDO ->HL
        PUSH    B       ;SAVE SHIP BLOCK BASE
;  FIND FREE TORPEDO
        MOV     B,H     ;HL -> BC
        MOV     C,L
        LOADX   A,RCNT  ;TEST FOR CLAIMED TORPEDO
        CPI     0
        JZ      SHOOT   ;NOT CLAIMED
        POP     B       ;SAVE STACK
        RET             ;NO FREE TORPEDOS
;  SHOOT A TORPEDO
SHOOT:  LDBLX   INPNT   ;UNBLANK TORPEDO
        MVI     M,2     ; 'MDISP'
        MVI     A,RMAX  ;SET TIMEOUT COUNTER
        STORX   A,RCNT  ; TO CLAIM TORPEDO
        POP     H       ;SHIP OBJ BLK IN HL
        MVI     D,10    ;10 BYTES TO MOVE
XLOOP:  MOV     A,M     ;BYTE FROM SHIP COORDS
        STAX    B       ;  INTO TORPEDO COORDS
        INX     H       ;NEXT BYTE
```

```
        INX     B
        DCR     D       ;DECREMENT CNTR
        JNZ     XLOOP   ;NOT DONE YET
        LXI     H,-10
        DAD     B       ;RESTORE TORPEDO BLK
        MOV     B,H
        MOV     C,L
;  COMPUTE INITIAL TORPEDO COORDINATES
;  X(N-1) & Y(N-1) ARE SHIP'S + COLLISION RAD
;  X(N) & Y(N) ARE SHIP'S + VELOCITY + COLL RAD
        LHLD    ORSAV
        LXI     D,XVTAB ;X VELOCITY TABLE
        DAD     D
        MOV     E,M     ;XVEL IN DE
        INX     H
        MOV     D,M
        LDBLX   XN
        DAD     D       ;ADD VEL+COLL TO X(N)
        SDBLX   XN
;
        LHLD    ORSAV
        LXI     D,XRTAB ;COLL RAD TABLE
        DAD     D
        MOV     E,M
        INX     H
        MOV     D,M
        LDBLX   XNM
        DAD     D       ;ADD COLL TO X(N-1)
        SDBLX   XNM
;
        LHLD    ORSAV
        LXI     D,YVTAB
        DAD     D
        MOV     E,M
        INX     H
        MOV     D,M
        LDBLX   YN
        DAD     D       ;ADD VEL+COLL TO Y(N)
        SDBLX   YN
;
        LHLD    ORSAV
        LXI     D,YRTAB
        DAD     D
        MOV     E,M
        INX     H
        MOV     D,M
        LDBLX   YNM
        DAD     D       ;ADD COLL TO Y(N-1)
        SDBLX   YNM
        RET
;  COLLISION RADIUS TABLES
XRTAB:  DW      0
        DW      RCON
YRTAB:  DW      RCON
        DW      RCON
        DW      0
        DW      -RCON
        DW      -RCON
        DW      -RCON
        DW      0
        DW      RCON
;  H.O. VELOCITY + COLL RAD TABLES
XVTAB:  DW      0
        DW      VCON1+RCON
YVTAB:  DW      VCON+RCON
        DW      VCON1+RCON
        DW      0
        DW      -VCON1-RCON
        DW      -VCON-RCON
        DW      -VCON1-RCON
        DW      0
        DW      VCON1+RCON
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;  SCORE PROGRAM
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
SCRE    SET     0
SDIG1   SET     1
SDIG2   SET     3
SCORE:  LOADX   A,SCRE  ;SCORE VALUE
        ANI     0FH     ;RIGHT DIGIT
        RLC             ;*2
        ADI     NUMS    ;INDEX OF SUBR IN HL
        MVI     H,0
        MOV     L,A
        MOV     E,M     ;SUBR ADDR IN DE
        INX     H
        MOV     D,M
        LDBLX   SDIG2   ;PNTR TO DIGIT
        MOV     M,E     ;SUBR ADDR IN DISPL FILE
        INX     H
        MOV     M,D
        LOADX   A,SCRE  ;SCORE VALUE
        ANI     0F0H    ;LEFT DIGIT
        RRC             ;JUSTIFY & *2
        RRC
        RRC
        ADI     NUMS
        MVI     H,0
        MOV     L,A
        MOV     E,M
        INX     H
        MOV     D,M
        LDBLX   SDIG1
        MOV     M,E
        INX     H
        MOV     M,D     ;LEFT DIG IN DISPL FILE
        RET             ;RETURN
;
```

Listing 5, continued:

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    PROGRAM TO ROTATE SHIPS, INITIATE TORPEDO FIRE,
;    DO HYPERSPACE PROCESSING & CHECK
;    FOR SPENT TORPEDOS
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; DON'T CHECK BUTTONS IF ALREADY IN HYPERSPACE
ROT:    LOADX   A,HFLAG
        CPI     0
        JNZ     SPNCK   ;CHECK FOR SPENT BULLETS ANYWAY
        LDBLX   BAPNT
        MOV     D,M     ;BUTTON WORD
;   CHECK FOR BOTH CW AND CCW,
;   INDICATING HYPERSPACE
        LOADX   A,HYPM  ;MASK FOR HYPERSPACE
        MOV     E,A
        ANA     D
        XRA     E
        JNZ     CWCK    ;NO - CHECK FOR CW
;   BLANK SHIP
        LXI     D,NULL  ;NULL GRAPHICS SUB
        LDBLX   PCPNT   ;PNTR TO CALL
        MOV     M,E
        INX     H
        MOV     M,D     ;INSERT
;   SCHEDULE SHIP START AFTER HDLY
        LDBLX   PEPNT   ;SHIP START ENTRY
        MVI     M,-1
        LDA     TIME
        ADI     HDLY
        INX     H
        MOV     M,A
;   SEE IF WE NEED TO DESTROY SHIP LATER
        MVI     D,1
        SCALL   2       ;RANDOM # IN A
        CPI     0       ;> ZERO
        JP      NODST   ;YES--DON'T DESTRY
        MVI     D,-1    ;NO--DESTRY
NODST:  STORX   D,HFLAG ;INDICATE TO SSHIP START PGM
        JMP     SPNCK
;   END OF HYPERSPACE PROCESSING
;   CHECK FOR CW (CLOCKWISE) ROTATION
CWCK:   LOADX   A,CWM   ;MASK FOR CW
        ANA     D
        JZ      CCWCK   ;CHECK FOR CCW
        LOADX   A,ORENT ;OLD ORIENTATION
        INR     A       ;UP BY 1
        ANI     7
        STORX   A,ORENT
;   CHECK FOR CCW (COUNTERCLOCKWISE)
CCWCK:  LOADX   A,CCWM  ;MASK FOR CCW
        ANA     D
        JZ      DINS
        LOADX   A,ORENT
        DCR     A       ;DOWN BY 1
        ANI     7
        STORX   A,ORENT ;INSERT ORENT IN DISPLAY FILE
DINS:   LOADX   D,ORENT
        LDBLX   PDPNT   ;ORENTATION IN DISPLAY
        MOV     A,M
        ANI     0F8H    ;STRIP OLD ORENT BITS
        ORA     D       ;INSERT NEW
        MOV     M,A
;   CHECK IF FIRE BUTTON ON
FINCK:  LDBLX   BAPNT   ;BUTTON ADDR
        MOV     D,M     ;BUTTON WORD
        LOADX   A,FIREM ;FIRE MASK
        ANA     D
        JNZ     INCHK   ;SEE IF INHIBIT
        MVI     A,0     ;NO FIRE -- CLEAR INH
        STORX   A,FINH
        JMP     SPNCK
INCHK:  LOADX   A,FINH  ;CHECK INHIBIT FLG
        CPI     0
        JNZ     SPNCK   ;SET
        MVI     A,1     ;NOT SET -- SET IT
        STORX   A,FINH
        LDBLX   FIPNT   ;SCHEDULE TORPEDO FIRE
        INX     H
        MVI     M,0
;   CHECK FOR SPENT TORPEDOS
SPNCK:  LDBLX   FBPNT   ; TORPEDO POINTER
;   FIND CLAIMED TORPEDO WITH TIMEOUT
;   FOR THIS SHIP
        MOV     B,H     ;HL -> BC
        MOV     C,L
        LOADX   A,RCNT
        CPI     0       ;TEST FOR CLAIMED
        RZ              ;NOT CLAIMED
        DCR     A       ;DECREMENT TIMER
        STORX   A,RCNT  ;RESTORE IN BLOCK
        RNZ             ;NOT ZERO YET
        LDBLX   INPNT
        MVI     M,0     ;BLANK TORPEDO
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    KEYBOARD DECODE PROGRAM
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
CTLC    EQU     'C'
KBDCD:  MOV     C,A     ;SAVE   CHAR
        CPI     CTLC    ;"C"?
        RNZ             ;RETURN IF NOT
        LXI     H,IENT1+1         ;SCHED INIT1
        MVI     M,0
        LXI     H,IENT2+1         ;SCHED INIT2
        MVI     M,0
        RET
;
        END     ■
```

Also Available: the famous LA36 DEC Writer II Keyboard Printer Terminal only $1495 (Assembled)!

FREE!

HEATHKIT CATALOG

Read all about our exciting computer systems and nearly 400 other fun-to-build, money-saving electronic products in kit form.

Prices are mail-order FOB, Benton Harbor, Michigan.
Prices and specifications subject to change without notice.
Heath Company, Dept. 334-340 Benton Harbor, Mi 49022

HEATH
Schlumberger

Heath Company, Dept. 334-340
Benton Harbor, Michigan 49022

Please send me my FREE Heathkit Catalog.
I am not on your mailing list.

Name ______________________

Address ______________________

City ______________________ State ________

CP-126 Zip ________

Circle 394 on inquiry card.

Fundamentals of Sequential File Processing

Wayne D Smith
Dept of Business Administration
Angelo State University
San Angelo TX 76901

The newly available floppy disk drive units have opened many new areas of applications for microprocessor users. In the area of high utilization data, and fast interactive programs, the capabilities of random access disk storage devices are unequaled. Unfortunately, the price of such units often puts them outside the budget of personal computer users. Even when disk units are available, the advisability of using these devices for storing low utilization programs or data may be doubtful. You don't really need a $1000 auxiliary storage device to maintain a Christmas card list which is used once a year.

The new, special design digital cassette recorders have also generated a great deal of interest. These units allow software control of all tape functions, and provide for tape position sensing. While the price of these units is about half that of a single disk drive, they are still somewhat expensive. Like disk units, they also require fairly sophisticated interface hardware for proper operation.

Audio cassette units, on the other hand, are cheap and readily available. These units are rapidly becoming the standard auxiliary storage medium for small computer systems.

Audio cassette units do, however, have several shortcomings. Compared to the other devices, they are slow, and most standard audio units do not have provisions for sensing the relative position of the tape. Further, most such units use a single motor for all tape operations, and accomplish the fast forward and rewind functions through gearing between the motor and the capstans. The physical pressure required to engage these gear trains virtually prohibits software control of these functions.

There are applications in which these shortcomings can be minimized. These applications are those that involve the use of auxiliary file storage where the information stored is inherently sequential in nature. While audio cassettes cannot compete with either digital cassettes or disk files in either speed or flexibility, they can provide very satisfactory results in the processing of sequential files.

This situation closely parallels the case in the data processing industry, where sequential tape files still play a major role in auxiliary storage. The economic advantages of these units can often offset the shortcomings of this type of storage.

About the Author

Wayne D Smith is currently an assistant professor of computer science at Angelo State University, San Angelo TX. He received his PhD in Computer Science from the University of Illinois in 1976. Besides his professional and hobby interest in computers he also enjoys operating an amateur radio station and flying.

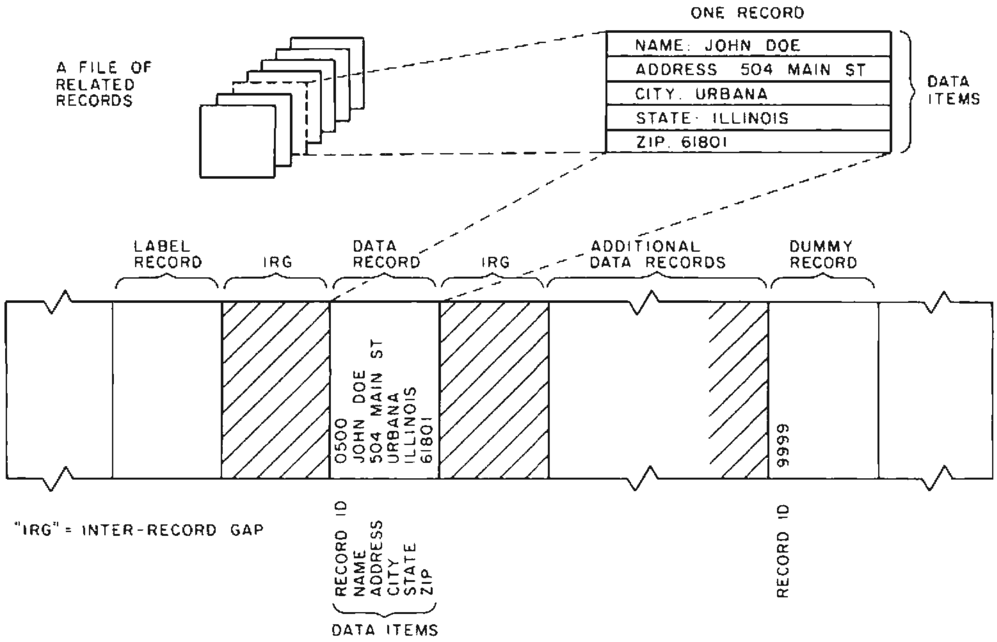


Figure 1: Relationship between files, records and data items in a sequential file.

Nature of Sequential Files

Figure 1 indicates the basic relationship between the components of a file oriented data processing system. In this type of system, information is arranged in sets or collections called files. The information within a file is usually related by the type of data contained in the file. For example, a small business might maintain separate files for employee records, customer accounts and stock inventory.

Files are then subdivided into smaller units called records. There is a single record for each entity within the file. In the case of the files mentioned above, there would be a record for each employee in the employee file, a record for each customer in the customer file, and a record for each stock item in the inventory file.

A record is simply a collection of related data items. In any file processing system, the user specifies the data items that constitute a record. For example, the records in a Christmas card list file would probably contain five data items. These items would be: name, address, city, state and zip code. These items are related within a single record in that they all pertain to the same individual. A collection of these records, whether on magnetic tape, magnetic disk, or in an address book, constitutes a file.

Files which are stored on magnetic tape are called sequential files because the records in such a file are stored one after the other on the recording medium. One of the major disadvantages of this storage type is the sequential arrangement of files. In order to obtain the information stored in the Nth record in such a file, the preceding N-1 records must be bypassed. Without tape position indicators, this means that the preceding N-1 records must be processed at normal tape read speed before the Nth record can be read.

A second, related problem concerns identifying individual records. Since many records may be read before the desired record is reached, some provision must be made for determining when the correct record has been found. In practice, this problem is usually overcome by providing each record in the file with a separate data item that can be used to identify that particular record. This identification must, of course, be unique for each record in the file.

Many types of records already contain data items which may be used for identification purposes. An employee file, for example, would probably contain a social security number in each record. This number is unique for each employee, and could be used for record identification. In other cases, an additional data item would have to be added to the record solely for purposes of identification. In the Christmas card list mentioned earlier, a sixth data item would have to be added in order to provide for record identification. For practical reasons, alphabetic data items, such as names, are seldom used for record identification.

While the records within a file do not necessarily have to be arranged in any special order, the search for a specific record is greatly facilitated if the records are stored in some orderly fashion. Sequential tape records are usually arranged in numerical order based on the identification number. In fact, this numerical ordering of records is a crucial requirement if efficient sequential file processing is to be possible. All the algorithms involved in processing sequential files are predicated upon this ordering.

The actual order of the records is immaterial. The records may be arranged in either ascending or descending order, depending on the preference of the user. Since ascending usually seems more natural to people, this arrangement will be assumed in the discussion which follows.

The sequential ordering of records has a large impact upon the speed with which a file can be processed. Suppose, for example, it becomes necessary to look up the addresses from five records in the Christmas card file mentioned above. If the records are unordered, each search must begin at the first record and search until the desired record is found. After listing this record, the tape must be rewound, and the search for the next record initiated from the beginning of the file. With this strategy the average number of records which must be read in order to find a specific record is one half the number of records in the file. Finding five records would require, on the average, reading 2.5 times the number of records in the entire file. In the worst case, it could require reading almost five times the number of records in the file.

If, on the other hand, the file is arranged in numerical order, the process is greatly simplified. In order to take advantage of the sequential ordering of the file, the numbers of the records to be found are also entered in sequential order. When this procedure is followed, processing time can be greatly reduced. After the first record is read and processed, the second search can begin without rewinding the tape. Since the second record has a higher number than the first one, that record must follow the first record. This means that the maximum number of records which must be read in order to process all five addresses cannot exceed the number of records in the file. This represents a considerable savings over the unsorted case.

Note, however, that the number of records read does not vary greatly regardless of the number of addresses to be found. That is, it requires about the same tape read time to process one record (average: one half the file length) as it does to process all the records in the file (full file length). This leads to the obvious conclusion that the type of files which are best suited to sequential storage are those in which a high percentage of the records are processed during each computer run. The number of records processed in a single run, divided by the number of records in the file, is called the activity ratio. As this value approaches one, processing efficiency approaches its maximum. A Christmas card list in which all records are listed for printing address labels has an activity ratio of 1.0. In this instance, sequential files offer a very efficient type of auxiliary storage.

Updating

As outlined in the preceding section, the processing of a sequential file does not differ greatly from the processing of any other type of file. The major differences appear when it becomes necessary to update, or change, the information which is stored in a sequential file. Updating may become necessary for any of several reasons. In some cases, data items may become inaccurate, as when a member of the Christmas card list moves. In other cases, it may become necessary to add or delete entire records from the file, as when a new employee is hired or an old one retires.

Due to the nature of magnetic tape drives, it is virtually impossible to position the tape head over a specific data item within a record. In fact, it is almost as difficult to position the tape head at the beginning of a specific record, since the record must pass the tape head before it can be read and identified. The process of adding new records in the correct sequential position presents additional difficulties. Inserting such records will necessitate moving all subsequent records down in the file. Minor errors in tape head position can result in erroneous or unreadable data.

To overcome these, and other, difficulties, the technique of using two tape units is employed. Instead of trying to change the information in the old file, a completely new file is created. This new file contains all the data and records from the old file that remain valid, plus any corrections, additions

Why you should buy a digital multimeter from the leader in digital multimeters.

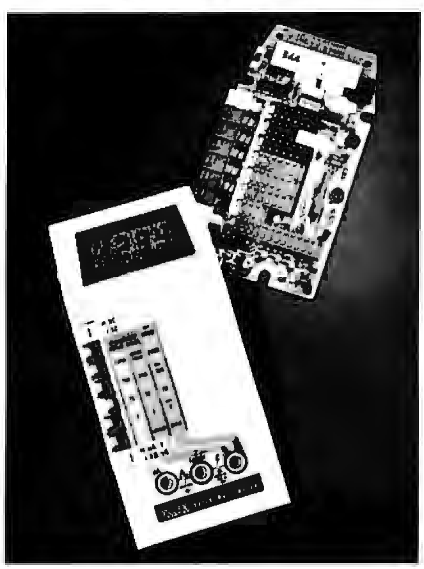
If you're shopping for your first multimeter, or moving up to digital from analog, there are a few things you should know.

First, look at more than price. You'll find, for instance, that the new Fluke 8020A DMM offers features you won't find on other DMMs at *any* price. And it's only $169.*

Second, quality pays. Fluke is recognized as the leading maker of multimeters (among other things) with a 30-year heritage of quality, excellence and value that pays off for you in the 8020A.

Third, don't under-buy. You may think that a precision 3½-digit digital multimeter is too much instrument for you right now. But considering our rapidly changing technology, you're going to need digital *yesterday*.

If you're just beginning, go digital.



Why not analog? Because the 8020A has 0.25% dc accuracy, and that's *ten times better* than most analog meters.

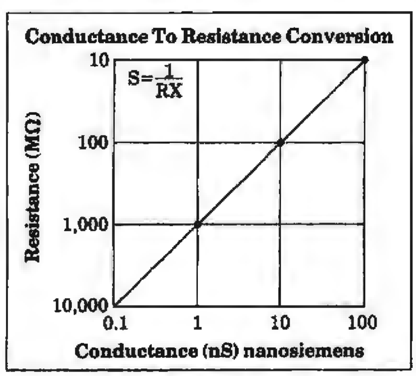
Also, the 8020A's digital performance means things like 26 ranges and seven functions. And the tougher your home projects get, the more you need the 8020A's full-range versatility and accuracy. The 8020A has it; analog meters don't.

If you're a pro.

You already know Fluke. And you probably own a benchtop-model multimeter.

Now consider the 8020A: smaller in size, but just as big in capability. Like 2000-count resolution and high-low power ohms. Autozero and autopolarity. And the 8020A is MOV-protected to 6000V against hidden transients, and has overload protection to 300V ac.

Nanosiemens?



Beginner or pro, you'll find the meter you now have can't measure nanosiemens. So what? With the 8020A *conductance* function, you can measure the equivalent of 10,000 megohms in nanosiemens. Like capacitor, circuit board and insulation leakage. And, you can check transistor gain with a simple, homemade adapter. Only with the 8020A, a 13-oz. heavyweight that goes where you go, with confidence.

What price to pay.

$169.*

Of course, you can pay more. Or less. In fact, you could pay almost as much for equally compact but more simplistic meters, and get far less versatility. And, the 8020A gives you the 'plus' of custom CMOS LSI chip design, and a minimum number of parts (47 in all). All parts and service available at more than 100 Fluke service centers, worldwide. Guaranteed, for a full year.

Rugged. Reliable. Inexpensive to own and to operate; a simple 9V battery assures continuous use for up to 200 hours.

Where to buy.

Call (800) 426-0361 toll free. Give us your chargecard number and we'll ship one to you the same day. Or, we'll tell you the location of the closest Fluke office or distributor for a personal hands-on feel for the best DMM value going.

*U.S. price only

Fluke 8020A DMM for Home Electronics Experts: $169

FLUKE®

Circle 395 on inquiry card.

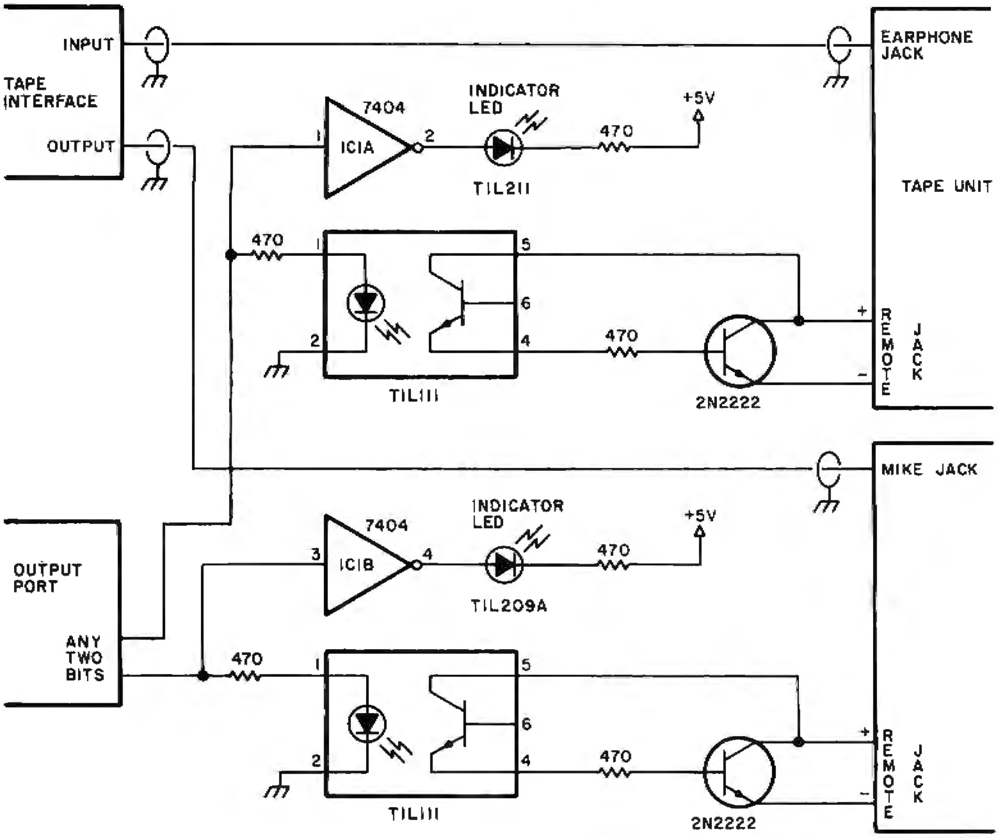


Figure 2: Circuit for controlling two tape decks for sequential file manipulation. The power pin connections to the 7404 (IC1) are +5 V to pin 7 and GND to pin 14. The 7404 inverter and indicator LEDs are not needed for the working of the circuit. Putting them in enables you to tell whether a read or write operation is happening. To make it easier to tell the difference from a distance, I have made the LED for writing red and the LED for reading green. Any equivalent LEDs will perform the same function. All resistances are in ohms and all resistors are 0.25 W.

or deletions that may be required. This is accomplished by using the old file as input (read only mode), while a new file is created in output (write only mode). Records which are to remain unchanged are transferred directly from the old file to the new file, via computer memory. Records which are to be changed receive part of the data items from the old file, but have other items updated from another input device (keyboard) before they are transferred to the new file. Records which are to be deleted from the file are read from the old file, but are not written on the new file.

Completely new records receive all the data items directly from the keyboard input device. After all data items have been entered, the new record is written on the new file. Program control insures that new records are entered in the correct sequential position in the new file. To facilitate the process of adding new records in sequence, all new files are initially created with substantial gaps between the record identification numbers.

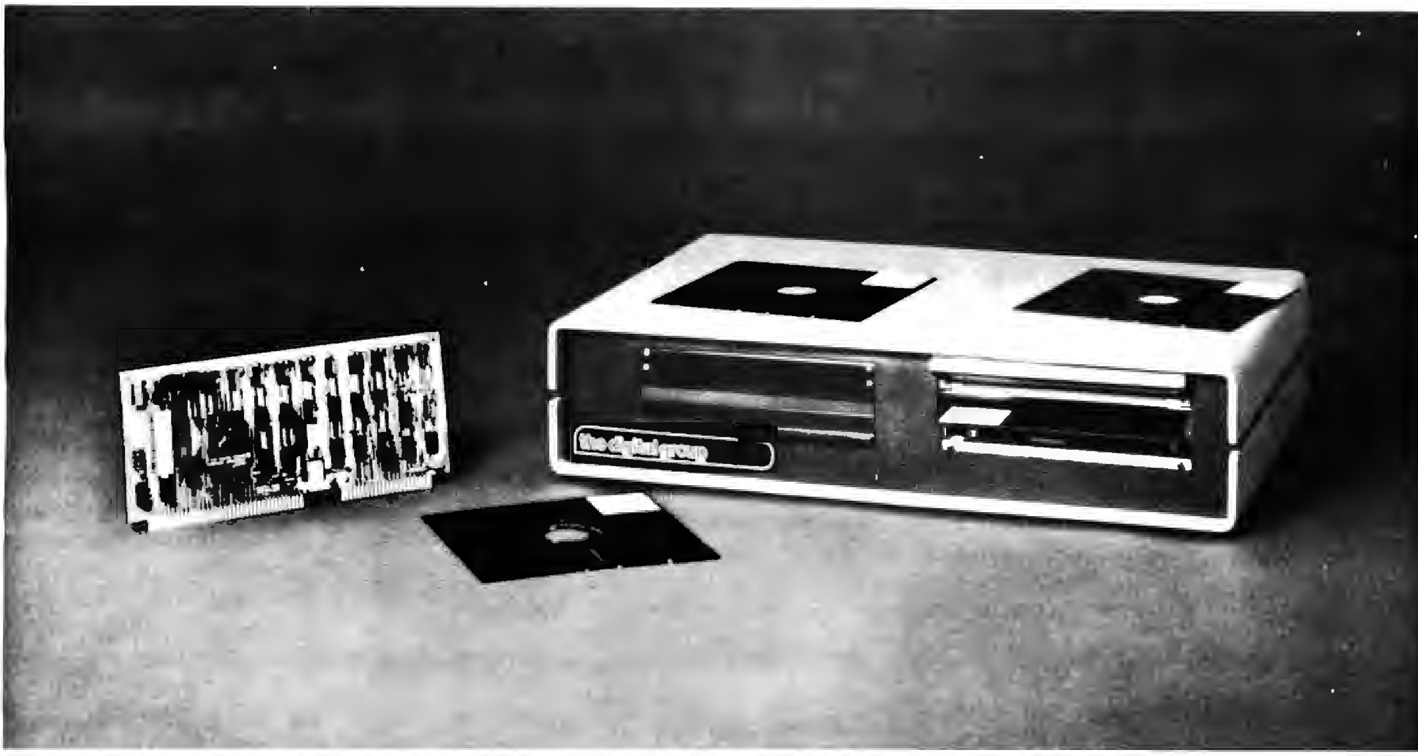
The technique of creating an entirely new file instead of trying to correct an old file has one major disadvantage. This disadvantage is, of course, the requirement for two separate tape units. There are, however, several advantages which far outweigh this single shortcoming. One significant advantage is that the exact positioning of the tape is not critical. Since the new records are written on blank tape, there is no problem with record alignment, and writing over old data cannot occur.

Another advantage is that each tape unit is operated in a single mode. Using audio cassettes, this means that one unit is always in the record mode, while the other is in the play mode. During the update process, the tapes operate in only this mode. The fast forward function is not required, and the rewind function is used only after all processing has been completed.

This single mode type of operation greatly simplifies the operation of the cassette units during the update. Since each cassette operates in only one mode, the function for each tape drive can be set up before processing begins. During processing, the tape units need only be turned on or off as records are read, processed and written. Since almost all cassette recorders have provisions for remote motor control, software can be utilized to activate the tape recorders as needed by commanding a suitable interface circuit such as that in figure 2.

One final advantage to the 2 tape system involves the protection of data. Since the old file is used only in the read mode, it will

NOW, DISKS TOO!



(And they're **Disk**-tinctly Digital Group.)

With the addition of our all-new Disk Subsystem, the Digital Group becomes the only manufacturer of microprocessors to offer you a real choice. Tape or disk Or both Whatever *you* want

We spent a lot of time designing and testing our new Disk Subsystem, and it shows in the result. The Subsystem is highly flexible. . . gives you direct access to data. . and completes your Digital Group System.

Here's what the Disk Subsystem consists of:

Disk Interface Card--provides the electrical interface between the disk drives and the computer.

Disk Power Supply--provides all power required for up to 2 drives

Disk Cabinet and Cables--cabinets provide housing for 1 to 2 drives plus the power supply; assembled cabling is provided for connection between all system components.

Disk Drives--the Digital Group Disk Subsystem supports standard 8" floppy disk drives from a number of manufacturers. Drives can be mixed, and in addition, mini-floppy drives are available for the function of a disk at minimum cost.

Documentation and Software--the Digital Group's standard documentation includes construction guides, maintenance manual, theory of operation and other useful information to help you get the fullest value from your new Disk Subsystem. And there's also software to provide initial support

The price? Low. It's become a tradition with the Digital Group to offer you high-quality, state-of-the-art equipment at the lowest possible prices. Our new Disk Subsystem is no exception.

For only $745, you can get a standard 8" floppy drive and an interface card. A complete single-drive Disk Subsystem with interface, power supply, cabinet, and cables is as low as $995.

Now the Digital Group system is really complete. If you don't own it all yet, take another longing look. In addition to our CPU, keyboard and video monitor, you can add an impact printer and the cassette storage system. All in our beautifully coordinated custom cabinets. All unmistakably Digital Group.

The Digital Group now has disks. And a lot more. Why not find out about all of our exciting products? Simply call or write today for direct information. From the Digital Group.

the digital group

box 6528 denver, colorado 80206 (303) 777-7133

Circle 396 on inquiry card.

remain intact even in the event that the updating program goes awry. If the user follows the practice of always maintaining the most recent version of the data, he or she will never be further than one update away from a current file.

Hardware Requirements

Assuming that a microprocessor already has a cassette tape interface, the implementation of a sequential file processing system requires very little hardware. As mentioned above, two cassette tape recorders are required, one for input, reading files, and one for output, writing files. The microprocessor tape input and output circuits are connected to the appropriate tape unit, as shown in figure 2. Even when a single circuit is used for both input and output, no problem arises, since only one cassette will actually be operating at any given time.

The cassettes can, of course, be operated manually if desired, but software control is easily implemented. Two latched bits of an output port are used to provide tape motor control. These output signals are used to activate the tape motors through opto-couplers (Radio Shack #276-1628, Texas Instruments TIL 111, or equivalent) and a single transistor.

The opto-couplers are used to prevent any polarity or voltage problems which might arise from direct interconnection of the microprocessor and tape motor circuits. In many cassettes, the positive terminal of the tape unit will be connected to the microprocessor ground terminal through the microphone and earphone cables. This arrangement would prohibit direct interconnection of the two circuits. Even when the electronics of a specific cassette would permit direct interconnection, the couplers are still worthwhile. Using the couplers will allow changing recorders at a later date without regard to the voltage or circuit ground polarity.

The only critical part of the circuit is the polarity of the transistor. The polarity of the remote jack should be checked before completing the wiring. Almost any NPN transistor will work, provided it can handle the current required by the tape motor. The current requirement for a specific tape motor is easily determined by connecting an ammeter across the remote jack with the tape in the play or record position. There is a fairly high current surge as the motor starts, but this is of short duration. A steady state current of 50 to 100 mA is about average.

The inverters and LEDs shown in figure 2 are optional. The indicator LEDs have proven to be very useful, since they are the only way to determine whether a tape unit is on or off (short of looking at the cassette). They can also provide a measure of amusement when a long tape program is running.

The operation of this interface system is quite straightforward. In order to process a tape file, the user first insures that the appropriate motor control is off. The cassette tape is then loaded and the tape function control is set for play or record as required. If a tape is to be recorded, it must be advanced past any leader on the cassette.

After the tape is loaded and the tape control is set, program execution can begin. The functions of reading or writing the tape are then under software control. If a tape record is to be read, the applicable motor control is turned on by writing a 1 in the appropriate bit of the output port. Normal tape input routines are then used to read the record. When the record has been read, the tape motor is turned off. For writing a record, the motor control is turned on, and normal tape dump routines are used to write the record on tape. A software delay is necessary in this case in order to allow the tape motor to come up to speed before recording begins. A second or two is usually sufficient.

Buffer Storage and Label Records

Processing sequential tapes requires that a certain amount of memory be allocated for storing the records as they are read or prepared for writing. This area is called a buffer, and is used to store the record currently being processed. The amount of memory required is minimal, since there is normally only one record in memory at a time. In one special case, when new records are being added to an existing file, storage for two records will be required. In all other cases, a single record length buffer is sufficient. Records are read into the buffer and then processed. Records may be written from the buffer onto a new file or to an appropriate output device as desired. For a generalized tape handling system, enough memory should be allocated to provide for twice the length of the largest tape record that will be processed. Not all of this area will be used by every program, but it should be provided in order to keep the system as general as possible.

In a sequential file processing system, a number of cassette tapes are required for storing the records of various files. More than one file may be stored on a single cassette, if desired, but this will result in

slower operation unless the files are very short.

Each file will contain two distinct types of records. All the records except the first are the data records that contain the actual file information. The first record in each file is a special type of record called the label, or header, record. This record should have exactly the same format on all files. The label record contains general information about the file itself. It does not contain data, and hence does not need an identification number. It is always the first physical record on the tape.

The purpose of the label record is to provide all the information which is required in order to process the file. Since this information changes from file to file, the obvious place in which to store it is in the file itself. Therefore, the first record in every tape file is dedicated to the label record.

The label record may be as simple or as elaborate as the user wishes to make it. At a minimum, however, it should contain:

- The file name or other identification.
- The date created (due to updating, several files of the same name may exist).
- The length of a data record.

In a more elaborate system, the label record could also contain:

- The number and type of data items.
- A description of all data items and their type (ie: binary, BCD, ASCII, etc).
- A security code (access can be denied without knowledge of this code).

All the remaining records on the tape are data records. The length and format of these records will vary from file to file depending on the amount and type of data in the file. The number of data records in a file will also vary.

Any program which processes tape files of this type must first read the label record. The information contained in the label record is then used to set variables that tailor the program for reading and processing the data records from that specific file. This procedure allows a single generalized program to process files of various types and lengths.

Utility Programs

In order to support the processing of sequential files, three general purpose utility programs are required. An optional fourth program may be provided if the user wishes a separate program for duplicating existing files. Running an update program without entering any changes will accomplish the same thing, however.

The first utility program is one which creates tape files from auxiliary input. This program performs many of the functions of a text editor. Data is entered via the keyboard, and is stored in the buffer until all the required data items have been entered. A sophisticated file creation program can be designed to query the user for data items one at a time until the record is complete. When all the data items have been entered, the record is written on the file. The process is repeated until the file is complete.

The file creation program must insure that the label record is written on the new file before any data is entered. As with the data records, the program can be designed so as to query the user for this information prior to writing the label record.

It may be desirable to provide the tape creation program with information about the number, length and type, ie: character, binary or binary coded decimal (BCD) of data in each data item. Information about the type of data would be used primarily for error checking, but could also be used to allow data packing. Packing of numeric data would allow two hexadecimal or binary coded decimal digits to be recorded as a single byte. This would require additional program complexity in converting from ASCII input to a packed format. This complexity may be warranted in cases where the packed format would facilitate arithmetic manipulation of the data directly from the tape file. Conversion back to ASCII would be required when listing the file.

As the data records are created, the user must supply the record identification number for each record. The user must insure that these are entered in the proper numerical sequence. In order to provide for the addition of new records at a later date, an initial file should have records numbered in increments of about 100.

The second required utility program is one which makes a hard copy listing of the data stored in a file. The label record may or may not be printed, depending on user preference. The simplest type program would perform an unformatted dump of the data exactly as it is read from the tape. A more elaborate program could separate the various data items into different positions or lines on the page. This type of program would allow printing the data in the form of a customer statement, a check or an address label.

By far the most complex of the utility programs is the update program. This pro-

Circle 398 on inquiry card.

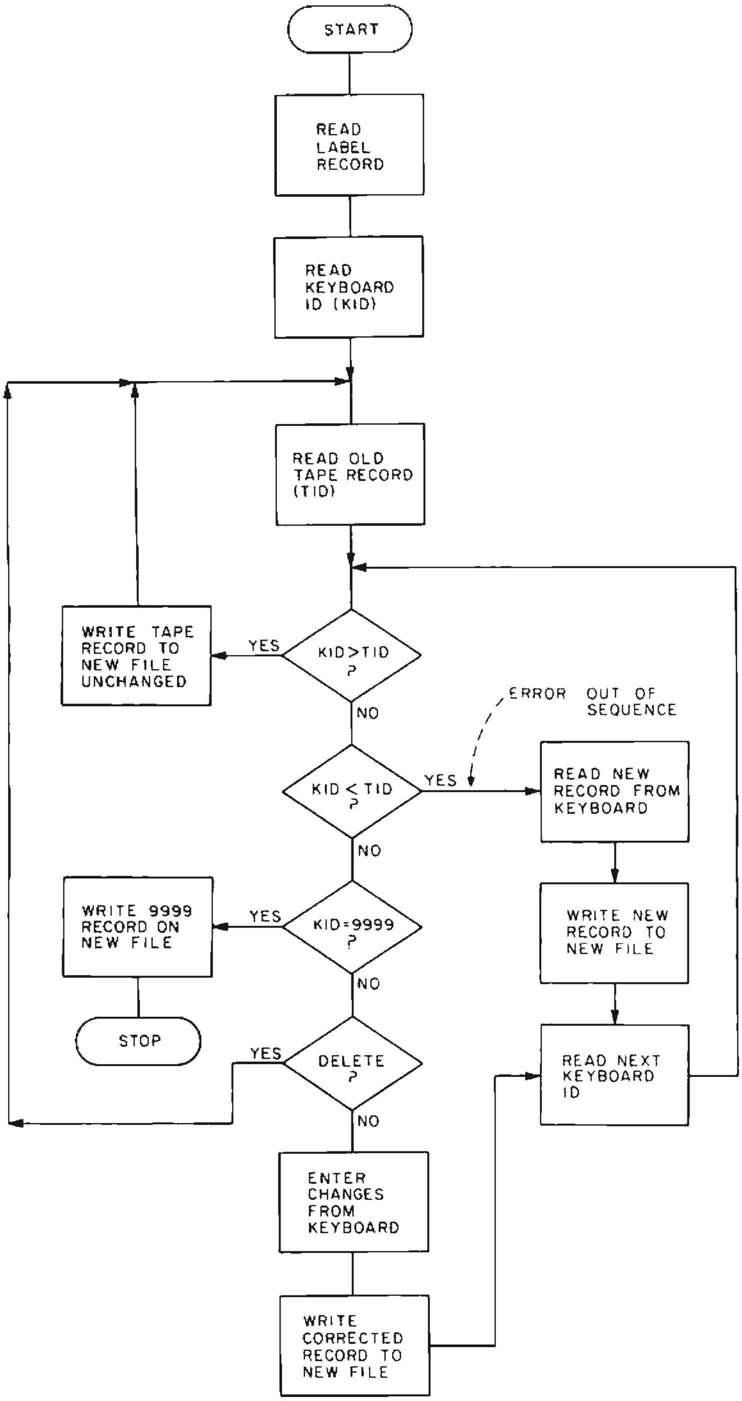


Figure 3: Sequential file updating algorithm allows the copying of records, deleting of old records, inserting of new records, and modifications of old records. This is a very generally applicable method, usable whenever serial format records are being processed.

gram is used for the correction of erroneous data in existing records, and also for adding records to or deleting records from the file. The logic for this program, which is shown in figure 3, is based on the assumption that the records in the file are arranged in ascending order, based on the record identification number. It is also assumed that the identification numbers of the records to be changed, added or deleted are also entered in this order.

The crux of the update program lies in the comparison of the keyboard entered identification number and the identification number from the tape record just read. When the tape ID is less than the keyboard ID, this indicates that the tape record just read is not to be changed. This record is copied to the new file exactly as it existed on the old file.

When the tape ID is equal to the keyboard ID, the tape record just read is one which is to be either changed or deleted. A keyboard entry determines which action is to be taken. If the record is to be changed, the new data items are entered from the keyboard into the buffer area, directly over the data that was read from the old tape file. When the changes have been completed, the corrected buffer is written onto the new file.

If a record is to be deleted, the program simply ignores the current tape record, and reads the next record into the buffer. The deleted record, therefore, does not appear on the new file.

When the tape ID is greater than the keyboard ID, this means that the program was unable to find a record ID on the old file which matched the keyboard ID. Therefore, the keyboard ID represents either a new record, or a typing error. When a new record is to be added, this record is entered into the second buffer area in memory. The use of this second buffer is necessary in order to prevent overwriting the last tape record read, which is currently stored in the first buffer. After the new record is entered, it is written to the new file from the second buffer. The added record now appears in proper sequence on the new file, while the last record read from the old file has been preserved. Processing then continues with the input of another keyboard ID number.

Whenever sequential files are processed, there is always a minor problem concerned with determining when the end of a tape file has been reached. In the program shown in figure 3, the old programmer's trick of using a dummy record with an ID number of all 9s is used to indicate the end of usable data in the tape file. The same code is used to signal the end of data entry from the keyboard. When this technique is used, the user must remember to provide this dummy record at the time a new file is created.

Examples

There are many instances in which the sequential file processing system outlined

above could be used. An obvious example is the Christmas card list mentioned earlier. This file could be created and then stored for use each December. By using a fairly simple listing program, mailing labels could be generated directly from the file. Sometime in January, the update program could be run to reflect any new addresses received with that year's cards. Persons from whom cards were received for the first time could be added to the file, and those from whom cards were not received could be deleted. The tape could then be stored until the following December. The January update is chosen as a point at which the activity ratio could be expected to be high. More frequent updates are possible, if needed.

An example which is not quite so obvious would be to use a sequential file as storage for a text editor which prepares input for an assembler. In this case, a record would consist of a single line of assembly language code. The data items would include: line number (record identification), label, operation, address and comments. Data items for location and machine code would be provided, but would be left blank by the file creation program.

After the file has been created, it is transferred to the input cassette for processing by the assembler program. The assembler then generates the location and machine code for each record in the file. As the machine code is generated, a new file is created on the output cassette. This file is identical to the input file except that the data items for location and machine code are filled in by the assembler. Optional features in the assembler could provide for loading the machine code for execution, or for creating a hard copy listing of the program as it is assembled.

If errors are detected during assembly or execution, the original file can be corrected through the use of the update program. Since each line represents a record, incorrect lines can be changed by changing the data items in that record. Extraneous lines can be deleted, and new lines can be added, in their proper positions, through the process of deleting or adding records to the file.

In using an assembler of this type, a 2 pass approach is required. This technique is required in order to prevent unresolved references at the time the machine code is being generated. Without prior knowledge of all label locations, forward reference branch instructions could not be assembled as they are read from the input tape. The 2 pass approach offers the additional advantage of dividing the assembler into two separate programs, only one of which must be in memory at any one time.

A particular advantage to using sequential storage with this type assembler is that it allows the assembly of very large programs. Due to the nature of sequential file processing, there is never more than one source language statement in the machine at any time. By processing the statements one at a time, more memory can be devoted to the assembler itself and its reference tables. In a machine with limited storage, this can result in a more elaborate assembler than would otherwise be possible.

A final example is of a more commercial nature, and involves the processing of an accounts receivable file for a small business. The basic file contains permanent customer information, such as name, address, account number, etc. It also contains data items for previous balance, current balance, payments, and a number of charges. When the file is originally created, these last data items are all set to zero.

During the first month, an update program is run, using this file as input. The file is updated by entering charges for that month. After all charges have been entered, the program computes the total charges and enters this value as the current balance in the new record. The record is then written on the new file. When all accounts have been updated, a listing program is executed, using the new file as input. This listing program, in effect, generates the first month's statements for all customers.

During the second and all succeeding months, a slightly different update program is used. This program first combines previous balance, payments and all old charges to form a new previous balance. Entries are then accepted for both payments and new charges. When all entries are completed, a new current balance is computed. The listing program is then used to generate customer statements for the new month.

Arguments Against Audio Cassettes

There are two arguments which are often advanced against the use of audio cassettes in the manner outlined in the preceding sections. The first argument is that leaving the tape drives engaged for long periods of time can flatten the capstans, which causes wow and flutter when tapes are read. In a system such as that proposed above, this would not present a problem. Even when large files are being processed, the motors would not be engaged for more than a few minutes at a time. In the case of file creation programs, where this time might extend to an hour or so, the tape unit is constantly being turned on and off. The probability of the capstan stopping in exactly the same position after each tape operation is quite small. Therefore, even when the capstan remains engaged for long periods of time, the exact position of the capstan is changing every few minutes.

Additionally, the capstan must be disengaged in order to remove the tape at the end of the program run. Since the tapes would normally be removed after processing, this would seem to preclude leaving the capstan engaged in the same position for a prolonged duration. For those people who are absentminded, the file processing programs could include a reminder to disengage the capstans after processing is complete.

The second argument against audio cassettes is their somewhat less than 100% reliability. Occasionally, a record will be written onto a file which cannot be read back. Even with good equipment, this can sometimes happen. In the file processing system described above, this presents only a minor problem, since it will seldom affect more than one or two records. If a record is unreadable, all the other records remain usable. Therefore, the tape update program can be used to add the unreadable record to a new file. Since the unreadable record does not "seem" to be on the old file, the process is treated as the addition of a new record by the update program. Even if several such records must be added the problem is of minor proportions, and represents more of a nuisance than a catastrophe.

Summary

It is readily admitted that sequential files are not a substitute for random access files. There are, however, many applications in which the disadvantages of sequential files can be minimized. In these instances, a substantial amount of file handling can be performed with very satisfactory results. In cases where disk files are not available, sequential files may be the only means of providing facilities for processing large volumes of data. Even in systems where disk files are available, the judicious use of sequential files can assist in conserving the more expensive disk storage facilities.

The software required to support a sequential file system is considerably less complex than that required by a disk file processing system, and the hardware costs are minimal. All in all, sequential files can provide a great deal of data processing capability at about one tenth the cost of a single disk storage device.■

SYNCHRO·SOUND

ENTERPRISES, INC.

One-stop shopping for Hardware and Software

Everything you need in small computer systems with special emphasis on TERMINALS! Look at these units...compare price, quality, delivery, service...and you'll see why you don't have to look anyplace else!

LEAR SIEGLER ADM 3A TERMINAL

- Full addressable cursor
- Display format—24 lines of 80 characters per line
- Communications rates—75 to 19,200 Baud
- Computer interfaces —EIA standard

ADM 3A Kit **$739.95**
ADM 3A Assembled .. **849.95**
Lower Case option **69.00**

ADM 1A TERMINAL

- Display Format —80 characters per line by 24 lines
- Full cursor control
- Edit operations: clear screen, clear unprotected character type-over. (Options are Character Insert/Delete, Line Insert/Delete, Erase to end of page, Erase Line/Field) and much more

ADM A1 Assembled **$1398.00**

ADM 2A TERMINAL

- 1920 character display (24 x 80)
- 16 function keys for 32 commands
- Separate keyboard—119 keys
- 10 key numeric pad
- Single key edit operations
- Page, field or line edit
- Security, protected fields and much more

ADM 2A Assembled ... **$1895.00**

OKIDATA MODEL 110 LINE PRINTER

- 110 CPS dot matrix

Friction Feed... **$1149.00**
Tractor Feed..... **1279.00**
RS 232C Serial Interface... **260.00**

OKIDATA MODEL 22 LINE PRINTER

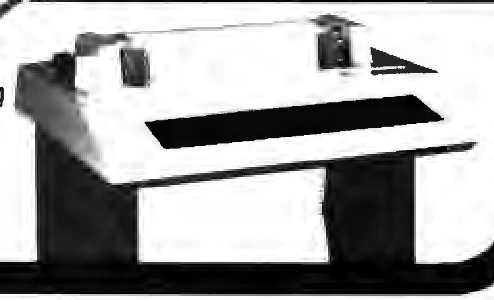
- 125 lines per minute 132-column print line
- Upper/lower case
- 8 different character sizes
- 12 IPS paper slew

Tractor Feed... **$2249.00**
RS 232C Serial Interface... **379.00**

DECWRITER II

- 132 column printing
- 10-30 CPS
- Full keyboard
- Tractor feed

$1695.00



C: A Language for Microprocessors?

J Gregory Madden
JGM Development Labs
POB 2345
West Lafayette IN 47906

I would first like to congratulate Peter Skye for his work on the high level language project! The interest created by his letters in recent BYTEs has helped to prod the emergence of a high level language. PL/I is a fairly popular language, and a good choice as an initial high level machine independent language. But what about other languages? Is there perhaps one which might be a better choice? Let's take a brief look . . . .

Enter FORTRANBASICSNOBOLPASCALCOBOLRPGCAPLALGOLPLI.

FORTRAN is a prehistoric beast from the 1950s, and though it has the highest "literacy rate" of any language in BYTE's 1976 survey, it is rather limited in its capabilities. BASIC is also popular. There are plenty of BASIC interpreters around and its capabilities (or, rather, lack thereof) are even less sufferable than FORTRAN's. Perhaps a BASIC compiler combined with an interpreter would be very helpful to speed up debugged programs, but let's continue. PASCAL is a reasonable candidate. COBOL would be fun to implement and it has a fairly high literacy rate; however, it is awfully wordy. Source programs take up lots of space and for microcomputers it is not a very good choice. ALGOL might be reasonable. As for RPG, FORMAC and TRAC, I know little about these languages; however, I don't believe they have any major advantages over others.

Enter C

What about this strange creature called C which has been vaguely mentioned here and there? Where did it come from, where is it going, and what's it all about?

C is a high level language based on an earlier language called B. It was written by several brilliant people* at the Bell Telephone Laboratories at Murray Hill NJ sometime around 1974. It is designed to run under the Unix operating system (also designed and written by these people) on a Digital Equipment Corporation PDP-11/45 thru 11/70 series machine. It now also runs on IBM equipment and is being rewritten for Interdata machines. The C users' rumor mill also mentions something about PDP-10 versions in the future. My purpose in writing this article is to convey the flavor of C and suggest its utility as a mode of expression for personal computing.

The language C offers a user the following features:

- Control structures which permit a flow of control by using the language, rather than having to program around the language.
- Expressions that eliminate most temporary variables and trivial statements, resulting in shorter, cleaner code.
- Pointers and character variables to do nonnumeric problems simply.
- Structures to enable easy implementation of complicated data concepts.

What does all of this mean? Basically, programming in C is easy, fun, and probably results in the fewest lines of comprehensible code per program of any general purpose programming language.

*Ken Thompson and Denis Richie.

Into the Briny Deep

What's C made of? Let's take a look at some of the basic constructs of the language to see what makes it tick.

Each section of the C program consists of what we will call a function, something like a FORTRAN function. Even the main program is a function with the name **main**. The format is as follows:

```
main( ) {
    statements
}
```

Arguments may be passed to a function by placing them between the parentheses in the definition and references to the function. C uses a call by value technique, rather than call by address for function arguments. This means that if a function changes the value of an argument variable, it does so only for that particular function call. How is this done? All variables in a function are stored on the stack rather than being assigned fixed locations. This means that storage space for variables is only allocated while they are being used, and is freed when control returns from a function. There are of course methods for changing the value of variables in the calling function during execution of the called function, but these methods are explicit exceptions to the call by value rule. Any C function can be used as a subprogram as well as its possible use as a main program. The example used **main** as a name, but as in many languages, there is a freedom to pick and choose descriptive names for C functions.

Variables are the usual: integer, character, floating point and double precision floating point. C variables are declared as in FORTRAN by a statement of the form:

type var1, var2, ... ;

Where *type* may be: **int**, **char**, **float** or **double**. Variables may be made to stick around (ie: storage is allocated permanently rather than temporarily using the stack) if the word static is included before *type* in the declaration. Then the value will not change between function calls unless explicitly modified. Normally, variables are dynamically allocated space on entry to a function.

All variables must be declared in C, and declarations must precede executable statements. Statements are ended with a semicolon (;) to indicate to the compiler what is the logical end of a statement.

mnemonic	C rep	description
BS	\b	backspace
NL	\n	newline
CR	\r	carriage return
HT	\t	tab
add	\add	octal byte
\	\\	backslash

IO in C

IO is done with basically two system supplied functions called **getchar** and **putchar**. Each gets or puts one character from standard input or output; **getchar** is normally used by equating a variable of type char to it:

```
c = getchar ( );
```

The variable c will have the ASCII value of the next character available from the input file; **putchar** is normally used by giving it the variable to print:

```
putchar (c);
```

The ASCII value of c would be output to the output file. The argument to **putchar** could be a single ASCII character:

```
putchar ('a');
```

in which case the character "a" will be output to the output file.

printf is another useful built-in function, similar to the FORTRAN WRITE and FORMAT statements. It actually calls **putchar** to do the IO. Its arguments are a string of characters in quotes, followed by the variables to be printed. In the string, format specifiers consist of a percent sign followed by an optional field size, followed by a letter indicating the format to output as: d for a decimal number, o for an octal number, h for a hex number, s for an ASCII string and c for a single character:

```
printf(
"The value of c is: %d"
,c);
```

If the new line is desired, it must be specified by putting a '\n' in the string at the appropriate point:

```
printf(
"The string is %s, and the value of a is %d.\n"
,string,a);
```

This inserts a carriage return and line feed when the '\n' is scanned during execution.

Special Characters

A note is in order before we go on about special characters in C. How does one represent a carriage return or a tab, or a line feed, etc? In general, anytime you want a special character, it can be generated by a backslash followed by a single nonnumeric character. New line is '\n', tab is '\t', end of file is '\0' (equivalent to 0). Also any ASCII character can be formed with a backslash and a 3 digit number: \040 is a space, etc. This enables all characters to be available for relational tests and use in computation.

The If Statement

The C **if** statement is of the format:

if (*expression*) *statement*

where *statement* may be any C statement, and *expression* may be any statement which has a value. The *expression* is true if its value is nonzero.

C relational operators may be selected from the following list:

==	equal to	!=	not equal to
>	greater than	<	less than
>=	greater than or equal to	<=	less than or equal to
&&	logical and	||	logical or
!	logical not		

Relational operators may be combined in any way, along with parentheses, to produce an expression.

The *statement* portion of the if may be any valid C statement, or combination of statements. The *statement* may be made arbitrarily complicated by enclosing a set of statements in braces, { } . These braces are similar to the BEGIN and END statements of several other languages.

The if may be associated with an else statement, which may also have an arbitrarily complex statement following it. Multiple elses can be used to construct logic that branches one of several ways:

if (*expression*) *statement*
else if (*expression*) *statement*
else if (*expression*) *statement*

While Statement

The *while* statement sets up a loop, whose general format is:

while (*expression*) *statement*

The *expression* is evaluated and if its value is true (not zero) does the *statement* and then starts again. The *statement* may of course be arbitrarily complex as in the if statement. Note that the *statement* is executed zero times if *expression* initially evaluates to zero.

Arithmetic

Arithmetic statements use the usual operators: +, –, * and / along with % which is the remainder (or "mod" operator in other languages). The syntax is a variable followed by an equal sign followed by an expression using the above operators. Multiple equivalences may occur on one line, an economy of notation which can be useful.

a = b = c = d = 25 * 2 / (c + 1);

One interesting feature is that when a variable is set equal to itself plus something, the statement may be abbreviated:

a = a + 5; is equivalent to:
a = +5;

Note that no space follows the equal sign in this form. In general the statement may be arbitrarily complex; however, note that one cannot assume a parenthesis, etc, in front of the implied variable.

Another very interesting feature is that any time a variable is used, it may have a prefix or a suffix of ++ or ––. The ++ means increment, the –– means decrement. If used as a prefix, the operation is done before the variable is used; if used as a suffix the operation is done after the variable is used.

These features help make C the supreme pleasure it is to use. They do however have a detracting feature: one must be wary when composing a statement lest it do something unintended!

Note that an if or while statement's expression may be a simple equivalence or it may contain calls to other functions! This can result in some of the cleanest, nicest code you have ever seen. It does sometimes take a minute to fathom the meaning of a statement, however, and it is easy to forget the double equals in an if test, resulting in an equivalence!

As an example, let's write a program to convert a file into lower case:

```
main( ){
    char c;
    while((c = getchar( )) ! = '\0'){
        if('A' <= c && c <= 'Z') putchar (c + 'a'    'A');
        else putchar (c);
    }
}
```

The program gets a character and assigns its value to the variable **c**. If the character is not zero (end of file, ie: \0) it executes the **if** statement. This checks to see if the character is upper case, and if so, converts it to lower case and outputs it. Otherwise it is already in lower case, and is output as is. The program continues with the next iteration of the while.

The Case and Switch Statements:

Another terribly useful C statement group contains the case and switch statements. They are used to replace tests of multiple options with **if** statements like:

if (c == 'a') . . .
else if (c == 'b') . . .
else . . . .

Our choice line of Personal Computing Books

NEW CHOICES

MICROPROCESSOR BASICS (Elphick) Selection and application info on 8 popular micro's, including the 8080, 6800, F8, IMP, and 6100. *#5763-6, paper, $9.95*

THE BASIC WORKBOOK: Creative Techniques for Beginning Programmers (Schoman) "Hands-on" learning of problem-solving using a computer. *#5104-2, paper, $4.25*

GAME PLAYING WITH BASIC (Spencer) Over 50 easy-to-learn and challenging games and puzzles for your personal computer. *#5109-3, paper, $6.95*

TELEPHONE ACCESSORIES YOU CAN BUILD (Gilder) Fully-illustrated, step-by-step instruction on building useful phone accessories at a fraction of the commercial cost. *#5748-2, paper, $3.95*

FUTURE CHOICES

STANDARD DICTIONARY OF COMPUTERS AND INFORMATION PROCESSING, Revised Second Edition, *#5099-2, Available Oct. '77.*

APPLIED COMPUTING: Putting Your Computer to Work, *#5761-X, Available Jan. '78.*

PROGRAMMING THE PROGRAMMABLE CALCULATOR *#5105-0, Available Jan. '78*

CRITICS' CHOICES

MICROPROCESSORS: New Directions for Designers (Torrero) ". . . a useful book for the electronics design engineer." BYTE MAGAZINE. *#5777-6, paper, $10.95*

FUNDAMENTALS & APPLICATIONS OF DIGITAL LOGIC CIRCUITS (Libes) "A great book for use as a reference by people who are learning digital electronics." PEOPLE'S COMPUTER COMPANY. *#5505-6, paper, $6.95*

BASIC BASIC: An Introduction to Computer Programming in BASIC Language (Coan) ". . . an excellent introduction . . . clearly written and well-organized." COMPUTING REVIEWS. *#5872-1, paper, $7.95*

ADVANCED BASIC: Applications & Problems (Coan) "This one rates well above average." DATA PROCESSING DIGEST. *#5855-1, paper, $6.95*

HAYDEN #1 in personal computing books!

AVAILABLE AT YOUR LOCAL COMPUTER STORE!

MORE CHOICES

GAME PLAYING WITH COMPUTERS, Revised Second Edition (Spencer) *#5103-4, cloth, $16.95*

COMPUTERS IN ACTION: How Computers Work (Spencer) *#5861-6, paper, $5.50*

COMPUTERS IN SOCIETY: The Wheres, Whys, & Hows of Computer Use (Spencer) *#5915-9, paper, $5.50*

PROGRAMMING PROVERBS (Ledgard) *#5522-6, paper, $6.50*

PROGRAMMING PROVERBS FOR FORTRAN PROGRAMMERS (Ledgard) *#5820-9, paper, $6.50*

COBOL WITH STYLE: Programming Proverbs (Chmura & Ledgard) *#5781-4, paper, $5.45*

MINICOMPUTERS: Structure & Programming (Lewis & Doerr) *#5642-7, cloth, $12.95*

DIGITAL SIGNAL ANALYSIS (Stearns) *#5828-4, cloth, $19.95*

FORTRAN FUNDAMENTALS: A Short Course (Steingraber) *#5860-8, paper, $4.95*

DIGITAL TROUBLESHOOTING: Practical Digital Theory and Troubleshooting Tips (Gasperini) *#5708-3, paper, $9.95*

DIGITAL EXPERIMENTS: Workbook of IC Experiments (Gasperini) *#5713-X, paper, $8.95*

COMPUTER MATHEMATICS (Conrad, Conrad, & Higley) *#5095-X, cloth, $13.95*

HAYDEN BOOK COMPANY, INC., 50 Essex Street, Rochelle Park, New Jersey 07662

The format of a is:

```
switch (variable) {
case 'a':
    statements;
    break;
case 'b':
    statements;
    break;
    .
    .
    .
default:
    statements;
}
```

The case statements label the various actions wanted; **default** is done if none of the other cases are satisfied. The **default** statement is optional. The **break** statement causes execution to continue with the next statement following the **switch** group. The **break** statements are optional, and if not used, execution falls through to the next case condition, which is often useful.

break may also be used with **for** and **while** statements, causing an immediate exit from the loop. (A **continue** statement may also be used to cause the next iteration of the loop to be started, in a **for** or **while** statement.)

Arrays

Arrays in C may be multidimensional and are subscripted by using square brackets. Array indexes begin at zero. Typical array declarations are:

```
int x[10];
int xy[10][20];
```

This example creates an array **x**, with ten elements, and an array **xy** with 200 elements. Subscripts can be arbitrary integer expressions.

Character Arrays (C Strings)

Character arrays in C are strings of ASCII characters terminated by a zero byte. This makes string handling simple. For example, when printing a character string using the **printf** format %s, **printf** prints the characters until it finds a zero byte.

Other Statements of Interest

The **for** statement is a generalized **while** of the format:

for *(initialization; expression; increment) statement*

where a typical example might be:

```
for(i=0; array[i] = getchar( ); i++);
```

which copies characters from the input file into **array**, until **getchar** finds a zero byte. The **for** is usually used when a variable must be initialized before it is used and then modified in a regular way each time the loop is reiterated.

Structures, Pointers, Pointers to Pointers, Pointers to . . .

Briefly, a structure in C is a grouping of data declared as follows:

```
struct {
    variable declarations
} {
    initialization section
} structure—name;
```

where the *initialization section* is optional. A structure member is referred to by:

structure—name.**member**

where **member** is one of the *variable declarations* of the declaration. Structures may be made to be arrays by following the *structure—name* with the array size in brackets. Note that the whole structure can be passed as an argument to a function, a feature most useful when complicated linkages are required.

Pointers are another way to refer to a member of a structure. Pointers in C are declared by preceding the variable name in the declaration with one or two (or more if you're a masochist) asterisks. One * means the variable is used as a pointer to something, two *s mean the variable is used as a pointer to a pointer (typically a pointer to an array of pointers). The address of a variable can be had in an expression by using the unary operator & in front of the variable. Pointers are usually used to "walk" along arrays efficiently. In fact, an array name represents the address of the zeroth element of the array, so it cannot be used on the left side of an expression, since you can't change the address of something by assigning to it.

The Crux of the Matter

It is difficult to learn much about a language by reading about it; I have only briefly skimmed the most important features of C in an effort to enable those who are not familiar with the language and are looking for a better high level language to gain some insight into it. It seems to take a bit of working with the language in order to be able to experience and understand the full power of it and to be able to effectively utilize that power.

XIMEDIA PRESENTS

The SOROC IQ120

CURSOR CONTROL. Forespace, backspace, up, down, new line, return, home, tab, PLUS ABSOLUTE CURSOR ADDRESSING.

TRANSMISSION MODES. Conversation (half and full Duplex) PLUS BLOCK MODE — transmit a page at a time.

FIELD PROTECTION. Any part of the display can be "protected" to prevent overtyping. Protected fields are displayed at reduced intensity.

EDITING. Clear screen, typeover, absolute cursor addressing, erase to end of page, erase to end of line, erase to end of field.

DISPLAY FORMAT. 24 lines by 80 characters (1,920 characters).

CHARACTER SET. 96 characters total. Upper and lower case ASCII.

KEYBOARD. 73 keys including numeric key pad.

REPEAT KEY. 15 cps repeat action.

DATA RATES. Thumbwheel selectable from 75 to 19,200 baud.

SCREEN. 12 inch rectangular CRT — P4 phosphor.

NEW LOW PRICE

Assembled $975.00

Specials of the Month

North Star **Micro Disk**
with power supply and cabinet ..... Kit — $699
Assembled — $799

IMSAI I-8080 with **TDL** ZPU ...... Kit — $825
Assembled — $975

Digital Systems FDS Disk Drive with
CP/M Software (assembled only) . Single — $1,750
Dual — $2,350

Mountain Hardware **PROROM** .... Kit — $145
Assembled — $210

Vector Graphic 8K RAM .... Kit — $235
Assembled — $285

NOW WE'RE TOLL FREE

800-227-4440

(in California, Hawaii, and Alaska, call collect: 415/566-7472)

XIMEDIA OFFERS A FULL RANGE OF PRODUCTS FOR THE PERSONAL COMPUTER ENTHUSIAST AND THE SMALL SYSTEM DESIGNER. LET US QUOTE ON ALL YOUR HARDWARE AND SOFTWARE NEEDS.

OUR RETAIL STORE — THE COMPUTERIST™ — IS NOW OPEN IN SAN FRANCISCO. CALL US FOR DIRECTIONS.

XIMEDIA

1290 24th Avenue, San Francisco, CA 94122
(415) 566-7472

Circle 405 on inquiry card.

COD orders freight collect. Orders with payment shipped prepaid. California residents add 6% sales tax. Please allow 3 weeks for delivery.

All of this is of no use to microprocessor users unless it can be implemented on a microprocessor. C was designed with the PDP 11/45 hardware directly in mind, and it takes advantage of the indexed addressing mode and the increment and decrement instructions. It turns out that most C statements result in about three machine code instructions!

The power of this language becomes more apparent with several notes. The Unix operating system was originally written in assembly language. During the summer of 1973, it was rewritten in C. A number of features were added including multiprogramming and the ability to share reentrant code among several user programs. The size of the new system was about one third greater than the assembly language version. The programmers "considered this increase in size quite acceptable." Several benchmark programs written to compare the Digital Equipment Corporation's IAS operating system BASIC and FORTRAN with the Unix C language gave results which placed C roughly ten times faster than FORTRAN and more than 15 times faster than the BASIC.

Why not PL/I?

I don't wish to put down PL/I, since it certainly is a useful language and has its place; however, I would like to make a few comments in closing.

Special characters are fine and fun, but create all kinds of problems when trying to implement them; for example, few terminals have many of the special symbols resulting in all kinds of weird escape sequences to get them. Unfortunately many keyboards in use are not even full ASCII. Hardware to do 5 by 7 matrix printing tends never to get built by the builder and most of the special symbols have no ASCII value. (On a personal note: I have met with the most amazing resistance to the idea of expanding ASCII to eight bits. People will nearly resort to physical violence against the idea. But why not? Most machines use eight bits anyway, and parity [how often is it really used?] can be implemented vertically instead of horizontally. Greek characters seem to be the rage [though how often have you used them?] so why not implement 256 characters? I can see the mail now. . .)

Mr Skye notes that he will update PL/SKYE as one might a dictionary. Have you looked at the size of the *Random House Dictionary* lately? It is all very well to intend that it won't be too big, but doing it is very much the question. What is too big? That depends on your system. As memory prices go down, too will the "big" threshold continue to go up and up. At this time, too big for a disk operating system seems to be about 16 to 32 K with perhaps four or five overlays (and that is very big), while on a tape system, too big is 16 K and that is very small if the program is to be in core, too.

What to do, what to do. . . Well, you probably shouldn't be running a high level language with a tape operating system unless it is a block oriented system (like the Digital Group Phi Deck system when implemented like a disk). So, perhaps we could set a limit at between 16 to 24 K for the compiler itself, leaving room for several 512 byte blocks for IO in a 20 to 32 K system.

As for writing the compiler in its own language, that is the only way to go.

Relocatable code is a must; and with index registers it becomes very simple. Here is where the processor becomes important. We really need a processor with indexed and relative addressing modes. That looks like the Z-80 processor (scream, rant, rave. . .). That does not however mean that it could not be implemented on a 6800, 6502, 8080, Micro-NOVA or whatever. Actually it turns out to be relatively easy to rewrite the code generation portion of a compiler to run on another machine, though at some expense in speed. (Can you imagine C on a Turing machine?)

IO? Yes, IO should be completely device independent. But that should be the concern of the operating system. Simply write your operating system in C, making it device independent. Actually, it is best to design the IO routines to be function (subroutine) calls in the compiler code, so the user can change it to suit his or her needs.

This brings up an interesting point. How does one distribute the code? It can't be in C source code; one can't run that. I hate binary object code since it is so machine and device dependent. Well, everyone should have an assembler for their machine, so why not have the compiler (or at least a version) generate machine source code. The IO sections of this can be modified if necessary using a text editor, assembled with the users' assembler and then it's ready to run. It turns out that it's not quite that simple, if you think about it or try it, but it's close. From then on, the compiler can compile itself when changes are made.

This has been just an introductory commentary on the virtues of what I consider an excellent language which should be adaptable to the range and scope of personal microprocessor based systems. As for when and if versions for the microprocessor user will appear, all I can do is close with the statement "wait and C."■

Simple Approaches to Computer Music Synthesis

Thomas G Schneider
706 Amherst SE
Albuquerque NM 87106

In order to produce a musical output, we must at least create a pitch output under control. This is but a starting point, since more complicated waveform and envelope generation is also useful in music.

The block diagram of a basic note and octave synthesis system is shown in figure 1. The top octave generator produces a square wave whose frequency (pitch) is determined by data sent out on the computer's data bus. Since the output of the top octave generator is a square wave, it can easily be divided by digital circuitry. Each time we divide the frequency by two, we end up with a note whose pitch is one octave lower than the input frequency. By using an ordinary TTL data selector as an octave selector, we can generate a musical scale covering many octaves, and we can also produce more than one pitch at a time, although these extra outputs will always be octave related to one another. The octave selector can also be controlled by data sent out on the computer's data bus, giving us more flexibility.

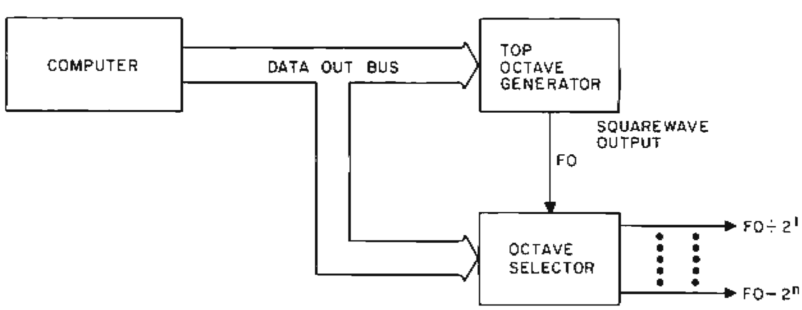
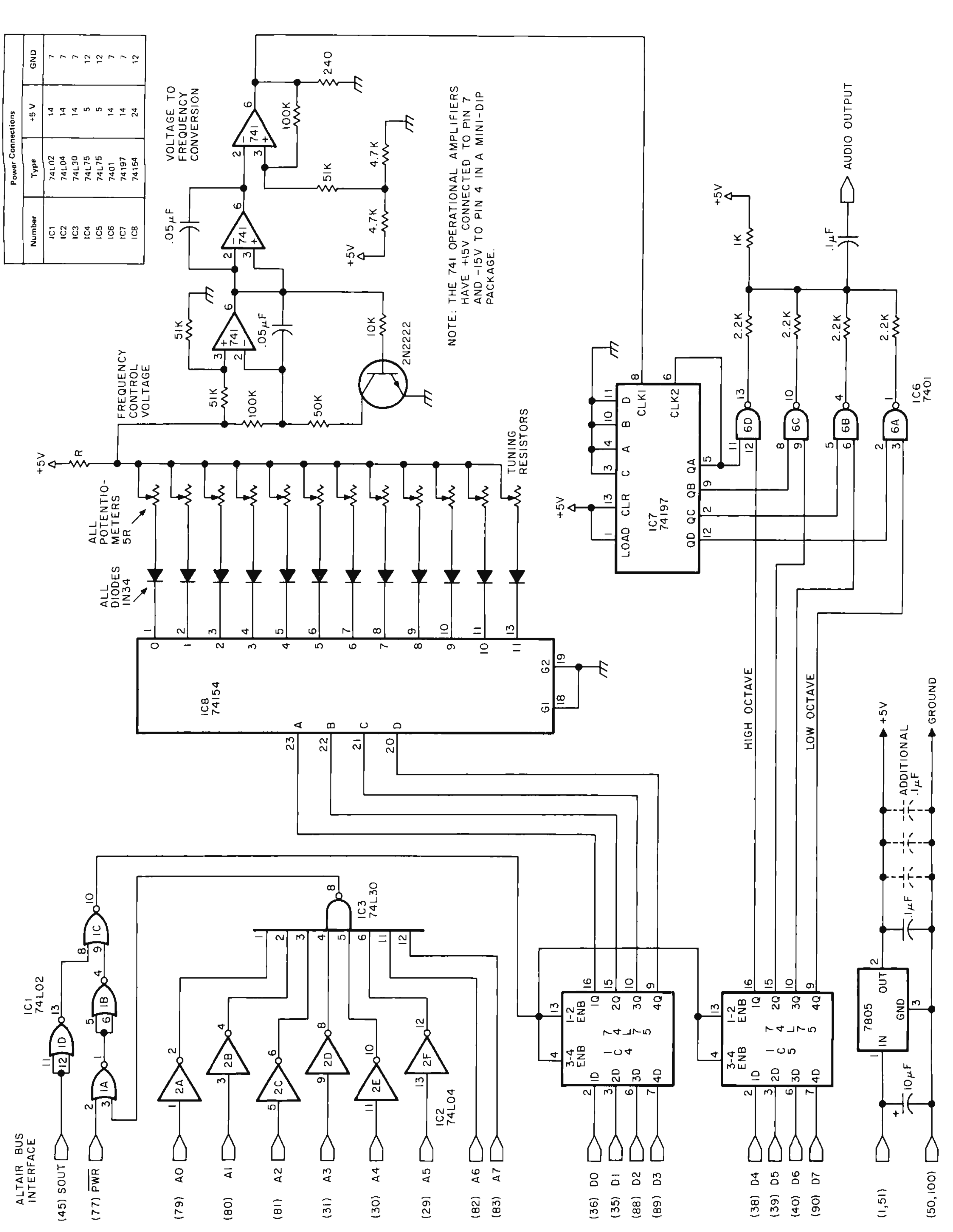


Figure 1: Block diagram of a pitch synthesis subsystem for use in electronic music experiments under computer control. The top octave generator produces a repetitive digital waveform selected under computer control from one of 12 well tempered pitches. This in turn drives octave generation logic consisting of a chain of toggles dividing frequency by two at each stage, and a selector to pick which of the octave related frequencies appear in the output.

The octave selector can be easily implemented using an n-stage divider and several NAND gates. However, there are several methods of generating the top octave. We need 12 notes to produce a 1 octave chromatic scale. These notes must be accurate in frequency and drift free in order to produce a true chromatic or "equally tempered" scale useful in music.

One way of synthesizing the top octave is to use a digital to analog (D/A) converter controlling an oscillator. An 8 bit converter limits both resolution and range so that we cannot produce an acceptably accurate chromatic scale. If we use a converter of 10 bits or more, resolution and range are suitable, but such units are expensive and require stable voltage controlled oscillators for this type of application. This method of pitch generation is shown in figure 1. The one nice feature of the digital to analog converter method is that we have a continuously variable output frequency. This permits nifty frequency sweeping effects (known as "portamento" or "glide" effects to the musician).

To save money we can construct a rather crude digital to analog converter which, in conjunction with the voltage controlled



Power Connections			
Number	Type	+5 V	GND
IC1	74L02	14	7
IC2	74L04	14	7
IC3	74L30	14	7
IC4	74L75	5	12
IC5	74L75	5	12
IC6	7401	14	7
IC7	74197	14	7
IC8	74154	24	12

Figure 2: An Altair bus interface and frequency selection logic for the tunable digital to analog conversion method of generation of pitches. This circuit can be constructed on a general purpose prototyping card for the Altair (S-100) bus.

Note	Octal Code	Hexadecimal Code
C	000	00
C#	001	01
D	002	02
D#	003	03
E	004	04
F	005	05
F#	006	06
G	007	07
G#	010	08
A	011	09
A#	012	0A
B	013	0B

Table 1: Octal and hexadecimal representations of note selections, in the low order bits, for these pitch generation circuits.

oscillator, will produce the 12 notes required for the full top octave. This method is shown in detail in figure 2. By using surplus 10 turn trimpots and the voltage controlled oscillator, we can construct an inexpensive top octave generator. However, this method has its disadvantages: tuning the trimpots is a critical operation, and once the pots are tuned, they can easily detune themselves because of vibration or temperature variations. My present synthesizer uses this method and needs to be retuned about every two months or so.

A good alternate method of generating the top octave is to use an integrated circuit top octave generator such as the MOSTEK MK50240P. This chip can be had for under $10 and is second-sourced by General Instrument Corporation as the AY-3-0215.

There are several advantages to be had by using this chip. The chip nominally requires a 2.000240 MHz reference frequency which is approximated by the central processor clock's circuitry of most Altair (S-100) bus systems. (The frequency is not exactly 2.000240 MHz, but will be close enough for this application.) This chip eliminates both the voltage controlled oscillator and digital to analog converter, and therefore puts an end to stability and tuning complications. The MK50240P generates the top 13 notes of the well-tempered music scale with an accuracy better than can be determined by the best musician.

Hardware Considerations for Two Working Circuits

The circuit used for the pitch generator, using the tunable digital to analog converter with an Altair bus is shown in figure 2. Bus timing and address decoding are performed by IC1, IC2, and IC3. IC4 and IC5 latch and hold data sent to the board on the data out bus. IC8 is a 4 to 16 decoder with active low outputs. These outputs select which 10 turn trimpot is selected as the bottom leg of the voltage divider whose top leg is resistor R. These trimpots should all have a value about 5 R. The voltage produced by this divider is connected to the input of the voltage controlled oscillator. The output of the voltage controlled oscillator is divided by IC7. The outputs of counter IC7 are gated onto the output bus by IC6, the quad open collector NAND gates. IC8 and the voltage controlled oscillator comprise the top octave generator and IC6 and IC7 comprise the octave selector.

The circuit using the MK50240P for top octave generation is shown in figure 3. The board address, bus timing and latch circuitry are identical to the circuit of figure 2. The octave selector is also identical to the one in figure 2. The MK50240P, IC6, is a 12 V device and requires input signal conditioning and output buffering. The 2N2222 transistor and associated resistors bring the TTL level clock signal from the bus up to the 12 V level required by the MK50240P. The outputs of this chip are buffered by IC7 and IC8 before going to IC9 which is the data selector and multiplexer. The MK50240P generates all the notes in the top octave simultaneously and IC9 selects any one of these outputs depending on what data is present at the outputs of IC4. The output of IC9 is then connected to the two chips (IC10 and IC11) comprising the octave selector. An additional voltage regulator, a 7812, has been provided to supply the 12 V needed for the MK50240P. Note that for additional music channels, additional copies of the note selector IC9 and octave selector can be driven off the buffered outputs of the MK50240P.

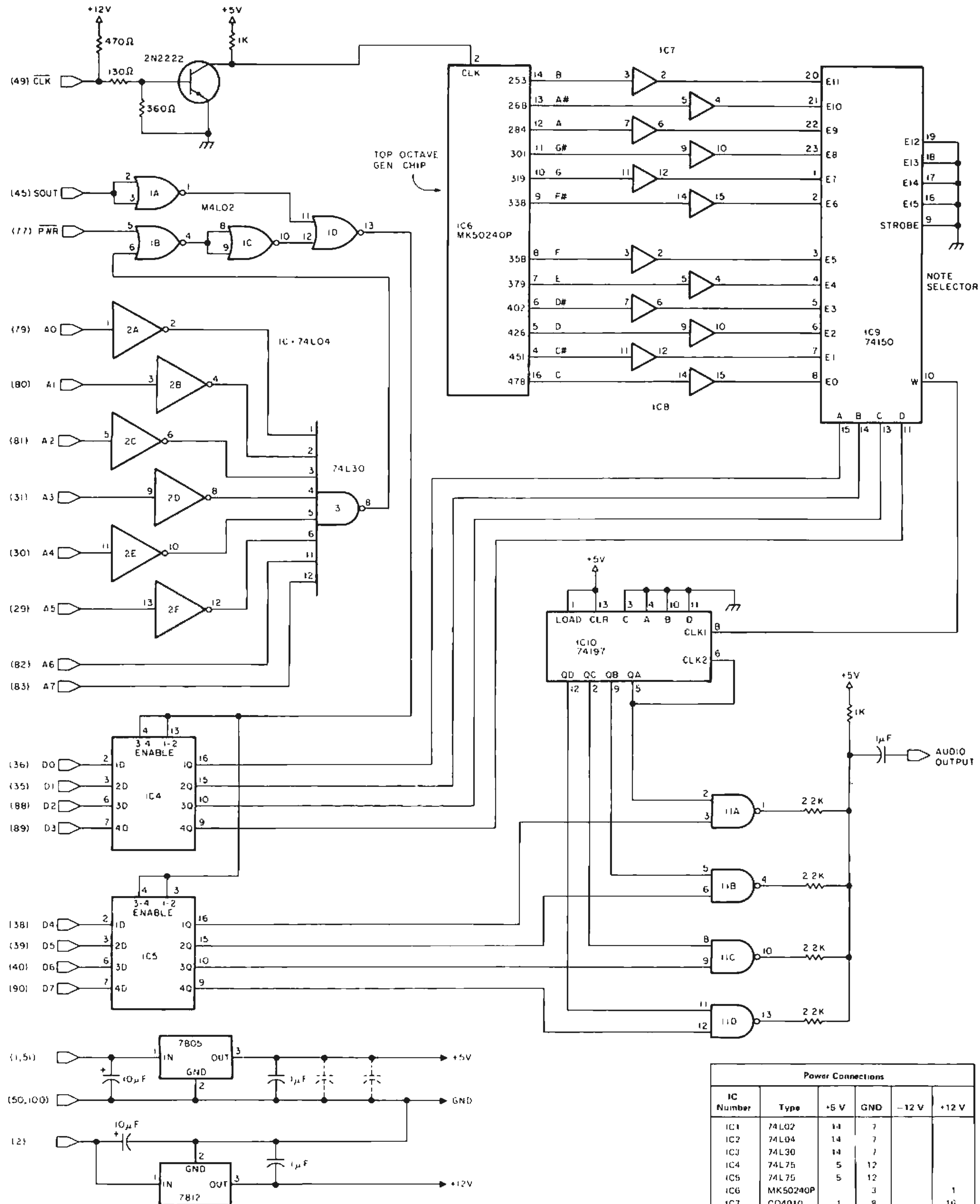
A word of caution: the audio output of the circuits in figures 2 and 3 swings about 2 V peak to peak and should be attenuated with a potentiometer before you plug it into your stereo system or amplifier.

Software Considerations

From a software point of view, the circuits of figures 2 and 3 are identical. Both circuits have an IO device address of 300 octal. Outputting the proper data to 8080 port 300 octal will cause the synthesizer to audibly produce the note and octave(s) represented by that data.

The synthesizer can be considered as having two input nybbles, each nybble containing four bits. The least significant nybble determines what note is to be selected and the most significant nybble determines what one of four possible octave(s) is to be selected. One byte contains all the information necessary to set up any note and octave(s).

Bear in mind that the synthesizer will continue to produce the note and octave(s)



Power Connections					
IC Number	Type	+5 V	GND	−12 V	+12 V
IC1	74L02	14	7		
IC2	74L04	14	7		
IC3	74L30	14	7		
IC4	74L75	5	12		
IC5	74L75	5	12		
IC6	MK50240P		3		1
IC7	CD4010	1	8		16
IC8	CD4010	1	8		16
IC9	74150	24	12		
IC10	74197	14	7		
IC11	7401	14	7		

Figure 3: The complete circuit for an equivalent of figure 2, which uses the top octave generator chip with an Altair bus interface. This method uses the top octave generator and a note selector to drive the octave selection logic, while the digital to analog converter method uses a voltage controlled oscillator with a diode resistance selection of pitch.

Octave	Octal Code	Hexadecimal Code
4 (highest)	020	10
3	040	20
2	100	40
1 (lowest)	200	80

Table 2: Octal and hexadecimal notation for octave enabling bits sent to the interface in the high order nybble of the 8 bit word. Note that one or more of the octaves may be enabled simultaneously simply by adding the codes together (or using a logical OR).

you have selected until you send it new data. To clear the synthesizer (no audible output) all you need do is output a 0 on data lines D4 thru D7, the most significant nybble.

Table 1 shows what the octal representations are for each note in the top octave. Table 2 shows the octal representations for each octave. To pack these two codes into one byte, they can either be added or ORed. Table 3 shows a series of codes that, when moved to port 300 octal in sequence, will produce a 12 note musical scale in the synthesizer's highest octave. However, if you wish to hear this scale, you must insert a software time delay in between each note. Otherwise all you will hear is a very short "click" because of the processor's high speed of program execution. If you wish to hear this scale in four octaves simultaneously, all you need to do is keep all four bits in the octave enabling nybble in the high state.

Armed with this information and some simple software routines, you and your trusty computer are now capable of synthesizing all of J S Bach's "B Minor Mass," to say nothing of the many new types of musical expression you now have at your fingertips. I myself have synthesized Haydn's "Minuet in G," parts of Bach's "Toccata and Fugue in D Minor," Henry Mancini's "Pink Panther Theme," and "Hot Rod Lincoln!"■

Octal	Hexadecimal
020	10
021	11
022	12
023	13
024	14
025	15
026	16
027	17
030	18
031	19
032	1A
033	1B

Table 3: Octal and hexadecimal representations of the notes of a chromatic scale in the highest octave of this interface. The codes for any given pitch can be generated by adding the note code of table 1 to one or more of the octave codes of table 2, providing a multiple frequency output in the summing resistor network of these circuits. In these examples, only one octave selection is enabled, the high octave.

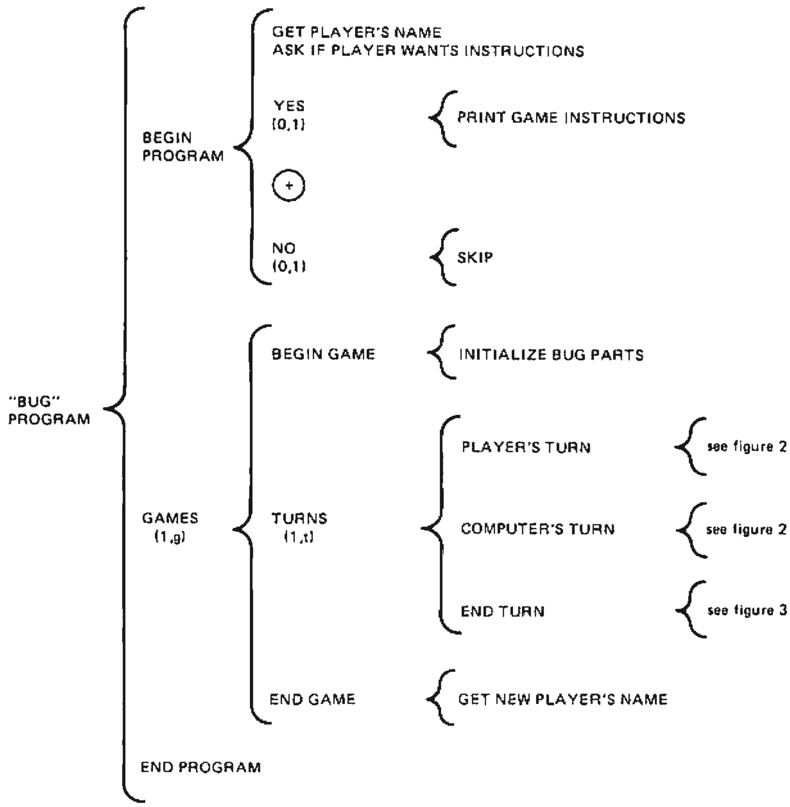


Figure 1: The Warnier-Orr diagram showing the basic structure of the BUG program.

Structured Program Design

David A Higgins
Langston, Kitch and Associates Inc
715 E 8th St
Topeka KS 66607

In the world of electronics, no experimenter in his right mind would build a circuit by throwing a few parts together with some wire and some hope, then attaching a line cord and plugging it in to see if it works. Not only are you likely to destroy some very expensive parts, but it is also a good way to get fried, or at least get a new hairdo.

Yet, after all the trouble that a serious microcomputer hobbyist will take to insure that his circuit is put together correctly before he ever turns it on, he will invariably try to program his new computer by using a technique analogous to the one above. That is why his programs almost never run right the first time, if indeed they ever manage to run right at all. It is also why many microcomputer buffs stay up until odd hours of the night drinking coffee by the gallon in an effort to find that one little bug.

But there is hope. I'm sure that nearly everyone involved with computers has heard something about structured programming in one form or another. It is not really a new technique, having been preached about for many years. However, the tools and methodologies available to design programs have changed radically over the years.

In the beginning there were flowcharts, which looked like five-dimensional octopi or the corporate structure of a conglomerate. Despite the absence of a consistent approach that would enable everyone to design a program using flowcharts, those programmers who did bother to work out their problem with a flowchart first usually seemed to have more luck in getting programs to run sooner and better than programmers who did not.

Structuring Tools

The development of mathematics would surely have been stymied if Roman numerals had been retained as our number system. In much the same manner, the science of structured program design would have been mired down if only flowcharts had been available for developing programs. It is not that calculus is impossible with Roman numerals, it's just that it's extremely difficult. Thus, over the years, a number of design and documentation tools were developed to better enable a programmer to understand the problem before going out to do battle with the program.

TOP-DOWN or GOTO-less programming, developed by Djkstra and others, was probably the first major attempt to solve the design versus coding problem. Djkstra simply observed that the more GOTOs that were in a program, the less likely it was to run correctly. Djkstra called such programs "spaghetti bowl" programs, because if you drew a line from each GOTO in the program to its destination, you ended up with a mess that looked like a bowl of spaghetti. He showed how any program could be written with just a few simple flow structures without any GOTOs. His techniques produced simple, readable code that was easy to test and maintain. So, the big push among design aficionados was to eliminate the GOTOs in their programming. Although TOP-DOWN programming was a big advancement over flowcharting, it was just that: programming. It was a technique for coding a program, not necessarily designing it.

Another technique, IBM's HIPO (and later HIPO-DB) entered the design field almost by chance, being primarily a documentation tool that was also being used for program design. The major drawback to HIPO techniques, besides the fact that they did not work well for designing a program, was their tendency to produce 50 pages of documentation for a 3 page program.

Warnier-Orr Diagrams — A New Approach

Within the last four years a new technique for program design has evolved from the work of Jean-Dominque Warnier (pronounced warn'-yay) in France, and Kenneth T Orr of Langston, Kitch and Associates in Topeka KS. The technique has foundations in set theory and Boolean algebra, and holds much promise for program design applications. Warnier-Orr diagrams, as we have called them here in the United States, allow programmers to design faster than ever before, to code programs with little or no effort, and produce programs that usually run correctly the first time. The approach is not limited to small programs. Nothing will make a believer out of someone quicker than a 20 page COBOL program which runs correctly the first time. The Warnier-Orr technique stresses design over coding and contends that once a problem is designed, it

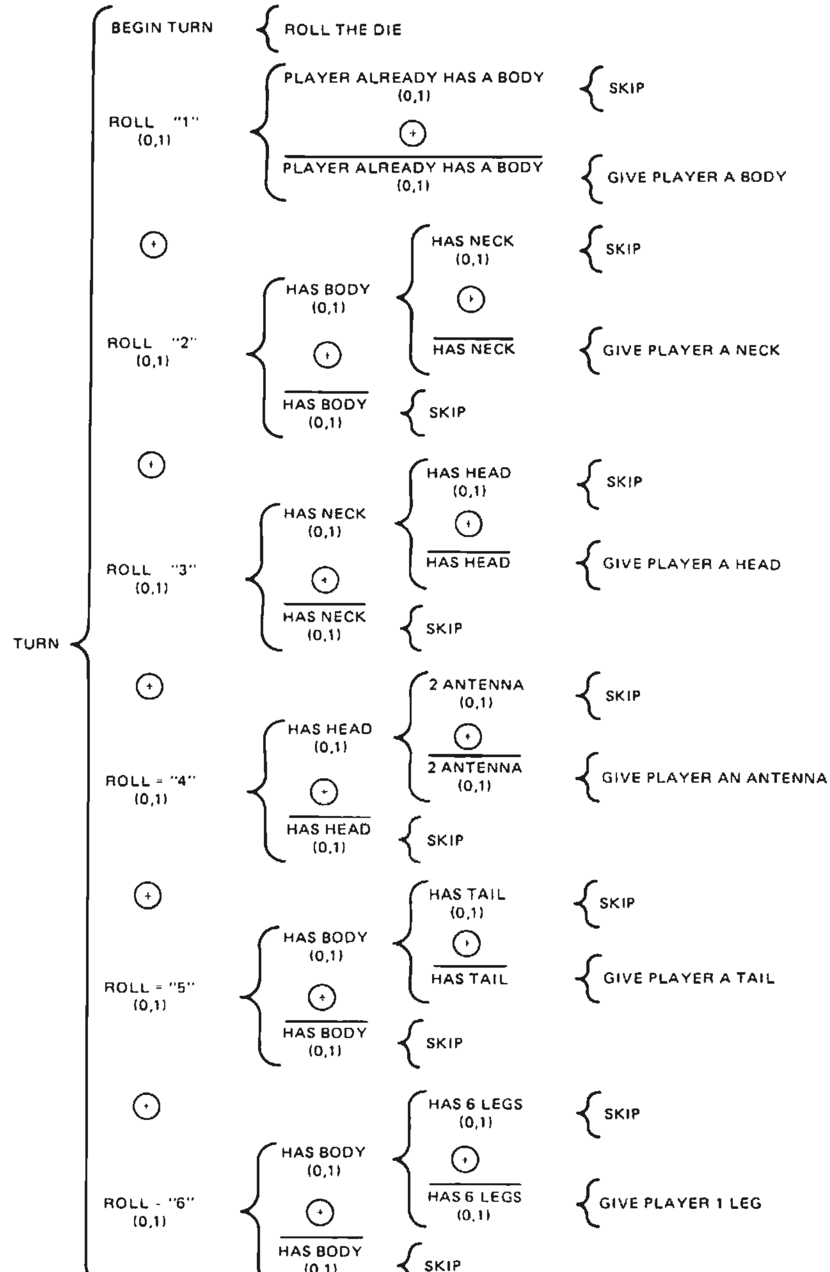


Figure 2: Diagram of the logic for the PLAYER and COMPUTER TURNS routines of the BUG program.

does not matter what programming language you code it in! At Langston, Kitch and Associates, people have used the technique to program in COBOL, PL/I, ALGOL, FORTRAN, BASIC, RPGII and assembler languages. It works equally well for all of them.

Warnier-Orr Diagram

The simplest way to learn about Warnier-Orr diagrams is to see examples of them. Warnier-Orr diagrams are very easy to learn

If the player rolls a 4, we first find the instructions to follow for a roll of 4 and check to see if the player has a BUG head. If he does, we then check to see whether or not the player already has two antennae. . . .

and use; however, be forewarned that this is a technique that is sometimes deceptively simple, but not as trivial as it often seems.

Let's consider the relatively simple game of BUG. In this game the computer rolls a die, once for itself and once for its opponent. Each number of the die corresponds to

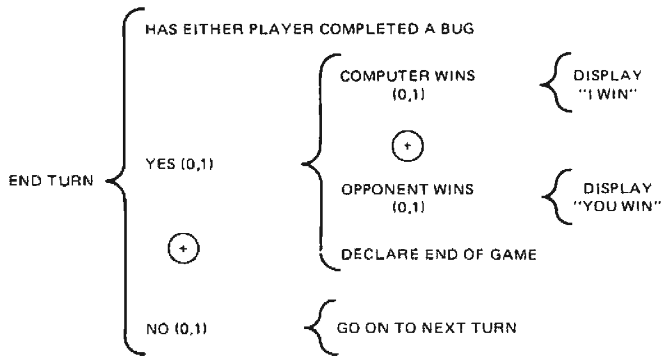


Figure 3: Warnier-Orr diagram for the ending of a turn or a game.

Listing 1: A structured BASIC program that was written using the Warnier-Orr diagrams of figures 1 thru 3. This code executed correctly the first time even though it was the author's first attempt at writing a BASIC program.

```
10   REM BUG PROGRAM
20   REM BEGIN PROGRAM
30   DIM HEAD(2), BODY(2), LEGS(2), TAIL(2), ANTE(2), NECK(2), CNT(2)
40   GOSUB 120
50   REM GAMES (1,G)
60   LET EPGM=0
70   GOSUB 200
80   IF EPGM=0 THEN GOTO 70
90   REM END PROGRAM
100  STOP
110  REM BEGIN PROGRAM SUBROUTINE
120  PRINT 'ENTER YOUR FIRST NAME'
130  INPUT :NAME$
140  PRINT 'DO YOU WANT AN EXPLANATION OF THE RULES; ENTER YES
     OR NO.'
150  INPUT ANS$
155  LET TEST = SCOMP ('YES' ,ANS$)
160  IF TEST = 0 THEN GOSUB 1200 ELSE ;
170  RETURN
180  REM GAMES SUBROUTINE
190  REM BEGIN GAME
200  GOSUB 290
210  REM TURNS (1,T)
220  LET EGAM = 0
230  GOSUB 390
240  IF EGAM = 0 THEN 230
250  REM END GAME
260  GOSUB 1150
270  RETURN
280  REM BEGIN GAME SUBROUTINE
290  LET BODY(1), BODY(2) = 0
295  LET CNT(1), CNT(2) = 0
300  LET NECK(1), NECK(2) = 0
310  LET HEAD(1), HEAD(2) = 0
320  LET ANTE(1), ANTE(2) = 0
330  LET TAIL(1), TAIL(2) = 0
```

a part of the BUG's anatomy: 1 = BODY, 2 = NECK, 3 = HEAD, 4 = ANTENNAE, 5 = TAIL, and 6 = LEGS. The object of the game is to finish your bug before the computer finishes its bug. Other rules: you must have a body before you can have legs, a neck or a tail; you must have a neck before you can have a head, and you must have a head before you can have antennae. One body, one neck, one head, one tail, six legs and two antennae are needed to complete a bug. Figure 1 is a Warnier-Orr diagram showing the basic structure of the BUG program.

The Warnier-Orr diagram is read left to right, top to bottom, just like conventional English text. The brackets enclose logically related operations, the largest of which is the program itself. The BUG program is composed of three logical sections:

- The BEGIN PROGRAM section, where the player's name is requested and there is an explanation of the game rules. Note that the + symbol between the modules YES and NO denotes the exclusive OR function, meaning that one or the other but not both of the modules will be performed. Observe also that this is reflected in the number of times that each module may be performed: 0 if the condition is false and 1 if the condition is true.
- The process section, GAMES, where the playing of the game actually takes place. The (1,g) denotes that the section is to be performed at least once, and possibly many (g) times.
- The END PROGRAM section, which in this case is empty, but which usually contains things such as the closing of files, the goodbye message, etc.

The rest of the brackets decompose in a similar fashion. The GAMES procedure breaks down into the beginning of the game, (BEGIN GAME), the turns that each player takes (TURNS), and the end of the game (END GAME).

Notice that logically there are things that only happen at the beginning of the program and things that only happen during the playing of the game itself. The Warnier-Orr diagrams allow you to see very easily just where and when a particular event must take

Whether managing accounts at the local general store or aiding in the development of the latest Space War, the **MiniTerm Dual MiniFloppy* System 80/2** can do the job best. Z-80 powered, and S-100 bus structured, the **System 80/2** mainframe with regulated supplies for two mini floppies* is also available without cards and drives.

Dealer Inquiries Invited.

MiniTerm Associates, inc.

Box 268, Bedford, Mass. 01730 (617) 648-1200

*Mini Floppy is a registered trademark of Shugart Assocates.

Circle 408 on inquiry card.

Listing 1, continued:

```
340   LET LEGS(1), LEGS(2) = 0
350   RETURN
360   REM TURNS SUBROUTINE
370   REM PLAYERS TURN
380   REM LET PLAYER START TURN
390   PRINT 'HIT RETURN TO ROLL DIE'
400   INPUT A
410   LET PLAY = 1
420   GOSUB 520
430   REM COMPUTERS TURN
440   LET PLAY = 2
450   GOSUB 520
460   REM END TURN
470   GOSUB 1060
480   RETURN
490   REM TURN SUBROUTINE
500   REM    PLAY=1;PLAYERS TURN—PLAY=2;COMPUTERS TURN
510   REM ROLL DIE
520   LET ROLL = FIX@ (((RND (0)) * 6.0)) + 1
530   PRINT : "ROLL IS A ", ROLL
540   IF ROLL = 1 THEN IF BODY (PLAY) #1 THEN GOSUB 690 ELSE ; ELSE ;
550   IF ROLL=1 THEN 650
560   IF ROLL = 2 THEN IF BODY (PLAY) = 1 THEN IF NECK (PLAY) # 1
      THEN GOSUB 760
570   IF ROLL=2 THEN 650
580   IF ROLL = 3 THEN IF BODY (PLAY) = 1 THEN IF NECK (PLAY) = 1
      THEN IF HEAD (PLAY) #1 THEN GOSUB 820
590   IF ROLL=3 THEN 650
600   IF ROLL = 4 THEN IF HEAD (PLAY) = 1 THEN IF ANTE (PLAY) # 2
      THEN GOSUB 880
610   IF ROLL=4 THEN 650
620   IF ROLL = 5 THEN IF BODY (PLAY) = 1 THEN IF TAIL (PLAY) # 1
      THEN GOSUB 940
630   IF ROLL=5 THEN 650
640   IF ROLL = 6 THEN IF BODY (PLAY) = 1 THEN IF LEGS(PLAY)
      # 6 THEN GOSUB 1000
650   LET A = 3
660   RETURN
670   REM BODY SUBROUTINE
700   IF PLAY = 1 THEN PRINT : NAME$, " 'S BUG HAS A BODY"
710   IF PLAY = 2 THEN PRINT : "COMPUTER'S BUG HAS A BODY"
720   LET CNT (PLAY) = 1
730   LET BODY (PLAY) = 1
740   RETURN
750   REM NECK SUBROUTINE
760   IF PLAY = 1 THEN PRINT : NAME$, " 'S BUG HAS A NECK"
770   IF PLAY = 2 THEN PRINT : "COMPUTER'S BUG HAS A NECK"
780   LET CNT (PLAY) = CNT (PLAY) + 1
790   LET NECK (PLAY) = 1
800   RETURN
810   REM HEAD SUBROUTINE
820   IF PLAY = 1 THEN PRINT : NAME$, " 'S BUG HAS A BODY"
830   IF PLAY = 2 THEN PRINT : "COMPUTER'S BUG HAS A BODY"
840   LET CNT (PLAY) = CNT (PLAY) + 1
850   LET HEAD (PLAY) = 1
860   RETURN
870   REM ANTENNAE SUBROUTINE
880   LET ANTE(PLAY) = ANTE(PLAY) + 1
890   IF PLAY = 1 THEN PRINT : NAME$, " 'S BUG HAS ",
      ANTE (1), " ANTENNAE."
900   IF PLAY = 2 THEN PRINT : "COMPUTER'S BUG HAS '" ANTE (2),
      " ANTENNAE."
910   LET CNT (PLAY) = CNT (PLAY) + 1
920   RETURN
930   REM TAIL SUBROUTINE
940   IF PLAY = 1 THEN PRINT : NAME$, " 'S BUG HAS A TAIL"
950   IF PLAY = 2 THEN PRINT : "COMPUTER'S BUG HAS A TAIL"
960   LET CNT (PLAY) = CNT (PLAY) + 1
970   LET TAIL (PLAY) = 1
980   RETURN
990   REM LEGS SUBROUTINE
1000  LET LEGS(PLAY) = LEGS(PLAY) + 1
1010  IF PLAY = 1 THEN PRINT : NAME$, " ' S BUG HAS ", LEGS (1), " LEGS."
1020  IF PLAY = 2 THEN PRINT : "COMPUTER'S BUG HAS ", LEGS (2),
      "LEGS."
1030  LET CNT (PLAY) = CNT (PLAY) + 1
1040  RETURN
1050  REM END TURN SUBROUTINE
1060  IF CNT (1) = 12 THEN 1090
1070  IF CNT (2) = 12 THEN 1110
1080  GOTO 1130
1090  PRINT : NAME$, " 'S BUG IS FINISHED' YOU WIN"
1100  GOTO 1120
```

place. After examining figure 1 carefully to make sure that you understand how the diagrams work, move on to the explanation of the PLAYER and COMPUTER TURNS section shown in figure 2.

In figure 2, we have represented the logic for each of the players' turns during the game. At the beginning of each turn, the die is rolled to determine the part of the BUG's body that the player may receive. Whatever the roll, we then have a logical path to follow. Again, please note that the presence of the ⊕ between each of the possible rolls denotes mutual exclusion, ie: only one of the paths may be selected. This particular structure is known as a case statement.

If the player rolls a 4, we first find the instructions to follow for a roll of 4 and check to see if the player has a BUG head. If he does, we then check to see whether or not the player already has two antennae. If he does, then we do nothing. If he does not have two antennae yet, we give him one antenna. If he does not have a BUG head, then again we do nothing. In a similar fashion, all of the possible rolls and their associated procedures are explained. Now let's move on to the Warnier-Orr diagram for the end of the turn, which is shown in figure 3.

If either player has won the game at the end of a turn, the computer declares the winner and ends the game. If neither player has won, the computer does nothing and cycles through for another turn.

Structured Programming

Having fully understood the problem, coding the BUG program is a simple and straightforward process. For this particular example I coded the program shown in listing 1 in a version of BASIC.

As you can see, each bracket of the original Warnier-Orr diagram roughly corresponds to a subroutine in the finished code: the process GAMES, for instance, becomes the subroutine at line number 180 which is called repeatedly by the branch at line 80 until EPGM equals 1, indicating that no more games are to be played; the process BEGIN PROGRAM is handled by the subroutine at line 110, and so forth. The resultant code is:

- easy to read and understand
- easy to change and maintain
- already documented
- logically correct.

It is also a program that will run correctly the first time, barring unforeseen syntax errors for those of us who can't type or spell. All of this is possible because the program was thoroughly designed before it was even partially coded.

Conclusion

Warnier-Orr diagrams are a giant leap in the right direction for structured programming. They represent an attitude which, for the first time since people have been playing with computers, can lead to consistently reliable software that is very easy to maintain. Currently, most data processing departments spend over 80% of their time and effort repairing old code that has suddenly gone bad. Warnier-Orr diagrams also provide the means to produce software of a quality that has never before been possible.

If you think that you are interested in using Warnier-Orr diagrams to help you solve some of your software headaches, by all means try them. But as I mentioned above, this technique looks deceptively simple, and you may not have much success. Understanding a diagram such as the one presented in this text is one thing; creating one from scratch is another.

If you do get bogged down, please feel free to write us for more information. If you try them, like them, and think you've done something exciting with them, again feel free to write us and tell us what you've done.■

Listing 1, continued:

```
1110  PRINT : "COMPUTER'S BUG IS FINISHED, I WIN"
1120  LET EGAM = 1
1130  RETURN
1140  REM END GAME  SUBROUTINE
1150  PRINT : "DOES ANYONE ELSE WANT TO PLAY"
1160  INPUT ANS$
1165  LET TEST = SCOMP (ANS$, 'YES')
1170  IF TEST # 0 THEN LET EPGM = 1
1180  RETURN
1190  REM EXPLANATION OF RULES SUBROUTINE
1200  PRINT "THE GAME OF BUG IS PLAYED AS FOLLOWS:"
1210  PRINT " A DIE IS ROLLED BY THE COMPUTER, AND EACH NUMBER"
1220  PRINT " ON THE DIE CORRESPONDS TO A PART OF THE BUG'S "
1230  PRINT " BODY: 1=BODY, 2=NECK, 3=HEAD, 4=ANTENNAE, 5=TAIL"
1240  PRINT " 6=LEGS. YOU NEED 1 BODY, 1 NECK, 1 HEAD, 2 ANTENNAE"
1250  PRINT " 1 TAIL, AND 6 LEGS TO COMPLETE A BUG."
1260  PRINT " THE OBJECT OF THE GAME IS TO BUILD YOUR BUG
      BEFORE"
1270  PRINT " COMPUTER BUILDS HIS."
1280  PRINT " —HIT RETURN WHEN YOU ARE READY TO PLAY."
1290  INPUT A
1210  RETURN
```

BIBLIOGRAPHY

1. Orr, Kenneth T, *Structured Systems Development*, Yourdon Press, New York, 1977.

2. Warnier, J D, *Logical Construction of Programs*, Van Nostrand Reinhold Company, New York, 1976.

Continued from page 20

dise purchased. We temporarily FIX 0 to print the quantity as a nondecimal integer and make reading the tape easier. We store the quantity in data memory 01, print, restore the SR-52 to FIX 2 and HLT.

5. Key B is the list price of the product. It is stored in data memory 02 and printed. This time we did not use a HLT instruction, and we will see why in the next step.
6. Without a HLT in step 5, the SR-52 program counter automatically moves down to the next step. It encounters the command to multiply AxB, or more exactly that data we stored in memory 01 times the data we stored in memory 02. We print this result and then sum the result into data memory 19. Since we cleared the SR-52 memories in our initial housekeeping instruction, we know that the initial content of data memory 19 is zero. After all this we HLT.

At this point in our flowchart we are at a manual or human decision point as shown by the diamond. If we have more merchandise to price, we can go back and put in the next quantity A and price B. We would do this until we were done. Then we would go to the A′ button. In our flowchart this is labeled as step 9. Putting A′ near the end of the program may appear to be out of sequence, but the SR-52 doesn't care. It will automatically go looking for A′ program instructions whenever that button is pushed. My programming style says that I like to finish defining the basic user defined key functions before getting involved in the detailed program steps. Since we haven't defined key E yet, that will be the next step. Incidentally, claiming a programming style as a reason for doing something is an easy way of avoiding a lot of "no win" programming arguments. If it's a matter of taste, who can complain?

Since the flowchart represents the actual sequence the user follows, we see that the A′ button is pushed to complete and print the totaling and discount of the list price merchandise. The details of this block will be discussed below. At this point we are only concerned with what we do next. The diamond again indicates a manual decision. Do we have any net price items on this invoice? If so, we will go to A again. This time, button A will be associated with quantities of the net price items. If there are no net price items we will push button B′. This will complete the invoice computation. At this point in the flowchart we define button E.

7. Key E is the net price of the merchandise. We will store this price in data memory 05 and print the price. Again as in step 5, since we don't have a HLT instruction, the SR-52 will automatically proceed to step 8.
8. This step will multiply AxE, or the quantity in data memory 01 times the net price in data memory 05. The result is printed and summed into data memory 19. Incidentally, the previous value of data memory 19 is stored with the net price computation that resulted when button A′ was pushed. The diamond after this block is another manual decision. If there are more net price items go back to A; otherwise go on to B′.
9. This block, at first glance, appears to be complex. But it really isn't. It provides the following instructions:

RCL 19	Recall the list price amount that was summed into data memory 19 and print (PRT) the amount.
1–C	Multiply the list price total by 1–C, the discount that was inputted, and stored originally with key C. Print the dollar amount of the discount.
–RCL 19	Subtract the list price amount from the discount; "+/–"; change the sign since the result is a negative number and print the result. This is the net price of the merchandise.
Pap	Advance the paper one line for format clarity.
STO 19	Store total net price in data memory 19 and HLT.

Not so bad, was it?

10. We now define the program instructions for the B′ button:

Pap	One line of paper advance.
RCL 19	Recall and print the total order net price that has been summed in previous

	steps into data memory 19.
+RCL D	Recall, add and print the shipping costs.
PRT=	Total the entire order and print.
Pap, Pap, Pap	Advance the paper three lines.
11. RST	Resets the program counter back to 000. This program address is the first one we used in our housekeeping steps. The program will automatically execute those housekeeping instructions and then HLT. The SR-52 is now ready for a new invoice routine.

The revised flowchart is now complete and at the same time we have defined virtually every program instruction step. The total development of this program has been a very straightforward translation of what we want to accomplish into precise terms understandable to our computer.

The last programming step is to set the SR-52 to receive program instructions. A key marked LRN (learn) is provided for this. Several other programming keys are also provided which dramatically simplify putting a new program into the machine. At this point we will only need the LRN button. With the SR-52 in the LRN mode, we begin with the first instruction which is CMs (clear memories). We push that button on the keyboard. The program counter moves to 001 and we push the CLR button. The SR-52 stores the clear instruction and the program counter moves to 002. We continue pushing the buttons that we defined in our program until we reach the last instruction which is RST. Our program is now stored in the machine. We take the SR-52 out of the learn mode by pressing the LRN button again. The program counter is manually reset to 000. Listing 1 shows the entire program list.

We are now ready for the critical test. The RUN button is pressed and, if the display is not flashing, there is a good chance that our program might work. We test it by putting in a sample calculation. If it does work, our tape printout would look like listing 2. If there are problems, the printer will be a great help in finding the bug.

The first thing that can be done is to ask for a list of the program steps. This output list can be compared to the work sheet we used to enter the program. Analyzing the program list shown in listing 1 is made easier if we mark each program address line with the key function name. *[In listing 1 this is done with typeset notations, but in SR-52 practice this would be done with handwritten notations ...CH]* In addition to doing this, groups of instructions should be bracketed so that they can be related to the flowchart in figure 2. The first three digits in the program list are the program counter address. The next two digits are the op code or the key codes for the SR-52 keyboard. If the program doesn't work, you may have omitted an instruction. The SR-52 provides an easy way to get to the point of omission so that we can insert the proper instruction. In a similar way, an incorrect instruction can be omitted.

The most painful debugging problem is where your program logic just isn't right. The printer will allow you to trace and print the steps the program actually is following as it attempts to solve the problem. (In our case, calculate the invoice.) If you are not too far off, the SR-52 has the facilities

Listing 1: Program listing for the SR-52 invoice program. The program is broken into sections corresponding to the flowchart of figure 2.

		PROGRAM COUNTER	KEY CODE
HSKPG 1	CMS	000	47
	CLR	001	25
	*FIX	002	57
	2	003	02
	HLT	004	81
DISCOUNT 2	*LBL	005	46
	C	006	13
	STO	007	42
	0	008	00
	3	009	03
	*prt	010	98
	*pap	011	99
	HLT	012	81
SHIPPING 3	*LBL	013	46
	D	014	14
	STO	015	42
	0	016	00
	4	017	04
	HLT	018	81
QUANTITY 4	*LBL	019	46
	A	020	11
	*FIX	021	57
	0	022	00
	STO	023	42
	0	024	00
	1	025	01
	*prt	026	98
	*FIX	027	57
	2	028	02
	HLT	029	81
LIST PRICE 5	*LBL	030	46
	B	031	12
	STO	032	42
	0	033	00
	2	034	02
	*prt	035	98
LINE TOTAL 6	×	036	65
	RCL	037	43
	0	038	00
	1	039	01
		040	95
	*prt	041	98
	SUM	042	44
	1	043	01
	9	044	09
	HLT	045	81
NET PRICE 7	*LBL	046	46
	E	047	15
	STO	048	42
	0	049	00
	5	050	05
	*prt	051	98
	×	052	65
	RCL	053	43
	0	054	00
LINE TOTAL 8	1	055	01
		056	95
	*prt	057	98
	SUM	058	44
	1	059	01
	9	060	09
	HLT	061	81
A' LIST PRICE TOTAL DISC AMT	*LBL	062	46
	*A	063	16
	*pap	064	99
	(	065	53
	(	066	53
	RCL	067	43
	1	068	01
	9	069	09
	*prt	070	98
	×	071	65
		072	53
	1	073	01
	−	074	75
	RCL	075	43
	0	076	00
NET 9	3	077	03
	)	078	54
		079	54
	*prt	080	98
		081	75
	RCL	082	43
	1	083	01
	9	084	09
	)	085	54
	•	086	94
	*prt	087	98
	*pap	088	99
	STO	089	42
	1	090	01
	9	091	09
	HLT	092	81
B' TOTAL NET SHIPPING, INVOICE AMT 10	*LBL	093	46
	*B	094	17
	*pap	095	99
	RCL	096	43
	1	097	01
	9	098	09
	*prt	099	98
	+	100	85
	RCL	101	43
	0	102	00
	4	103	04
	*prt	104	98
		105	95
	*prt	106	98
	*pap	107	99
	*pap	108	99
	*pap	109	99
11	*RESET	110	86
		111	00
		112	00
		113	00

DISCOUNT	0.60	PRT
QUANTITY	12.	PRT
LIST PRICE	5.63	PRT
LINE TOTAL	67.56	PRT
QUANTITY	18.	PRT
LIST PRICE	1.87	PRT
LINE TOTAL	33.66	PRT
QUANTITY	34.	PRT
LIST PRICE	5.63	PRT
LINE TOTAL	191.42	PRT
TOTAL LIST	292.64	PRT
DISC AMT	117.06	PRT
NET TOTAL	175.58	PRT
QUANTITY	18.	PRT
NET PRICE	0.42	PRT
LINE TOTAL	7.56	PRT
TOTAL NET	183.14	PRT
SHIPPING	2.83	PRT
INV AMT	185.97	PRT

Listing 2: Sample printout of the invoice program.

that allow you to easily implement your program fix.

Listing 3 shows the printer output in the trace mode. Here the left digits are the printed equivalent of the SR-52 display. The right side data is the function the SR-52 is performing on the display data.

The data shown in listing 3 is the same data used for the printout shown in listing 2. Marking and grouping the trace data in the way shown will simplify comparing the trace output with the design flowchart of figure 2. Each step in the trace should match what you designed in the flowchart. If it doesn't, then the machine is being asked to do something incorrectly. If your flowchart logic is faulty, review the SR-52 instruction book before making corrections. If your flowchart design has an improper premise, then the calculator will indeed proceed with perfect logic to an improper result.

Somewhere between the two "dump" methods, list and trace, lies the identification to the program problem. Listing shows program loading mistakes and tracing will reveal program logic errors.

I could go out on a limb with the truth and say that my invoice program was up and running on the first pass. But seriously, with the debugging facilities already discussed, it wasn't too difficult to find and correct the problem areas.

After the program has been debugged, you should make a new program list. Any flowchart changes that were necessary should also be documented. If you don't do this documentation, you will find that your memory will fail you miserably if you want to modify the program sometime in the future. The up-to-date list will also let you reproduce the program in case the magnetic program card ever becomes unusable.

```
                                  PROGRAM
                      DATA      INSTRUCTION
HOUSEKEEPING                       CMS
FOR                                CLR
THIS RUN               0.          FIX
(1)                                002
                      0.00
                                   HLT
"C"                   0.6          STO
DISCOUNT                           003
(2)                   0.60
                      0.60         PRT

                                   HLT
"D"                   2.83         STO
SHIPPING                           004
(3)                   2.83
                                   HLT
ITEM #1:               12.         FIX
                                   000
"A"                    12.
QUANTITY               12.         STO
(4)                                001
                       12.
                       12.         PRT
                       12.         FIX
                                   002
                     12.00
                                   HLT
"B"                   5.63         STO
LIST                               002
PRICE                 5.63
(5)                   5.63         PRT
                      5.63         x
                      5.63         RCL
                                   001
AxB                  12.00
(6)                    12.         =
                     67.56
                     67.56         PRT
                     67.56         SUM
                                   019
                     67.56
                                   HLT
ITEM #2                18.         FIX
"A"                                000
QUANTITY               18.
(4)                    18.         ST-

                       18.         HLT
                                   FIX
                                   000
                       34.
                       34.         STO
                                   001
                       34.
                       34.         PRT
                       34.         FIX
                                   002
ITEM #3.             34.00
                                   HLT
"B"                   5.63         STO
LIST                               002
PRICE                 5.63
(5)                   5.63         PRT
                      5.63         x
                      5.63         RCL
                                   001
AxB                  34.00
(6)                    34.         =
                    191.42
                    191.42         PRT
                    191.42         SUM
                                   019
                    191.42
                                   HLT
```

```
                    191.42         (
                    191.42         (
                    191.42         RCL
                                   019
TOTAL               292.64
LIST                292.64         PRT
PRICE               292.64         x
                    292.64         (
                        1.         -
                        1.         RCL
                                   003
                      0.60
DISC                   0.6         )
AMT                    0.4         )
                    117.06         PRT
                   117.056         -
                   117.056         RCL
                                   019
A'                  292.64
(9)                 292.64         )
                    175.58         PRT
NET
PRICE              175.584         STO
                                   019
                    175.58
                                   HLT
                       18.         FIX
                                   000
                       18.
                       18.         STO
                                   001
                       18.
"A"                    18.         PRT
QUANTITY               18.         FIX
(4)                                002
                     18.00
                                   HLT
"E"                   0.42         STO
NET                                005
PRICE                 0.42
(7)                   0.42         PRT
                      0.42         x
                      0.42         RCL
AxE                                001
(8)                  18.00
                       18.         =
                      7.56
                      7.56         PRT
                      7.56         SUM
                                   019
                      7.56
                                   HLT

                      7.56         RCL
NET                                019
PRICE               183.14
TOTAL               183.14         PRT
                   183.144         +
B'                 183.144         RCL
(10)                               004
                      2.83
SHIPPING              2.83         PRT
                      2.83         =
INVOICE             185.97
AMT.                185.97         PRT


(11)                               RSET
                                   CMS
HOUSEKEEPING                       CLR
FOR                     0.         FIX
NEXT RUN                           002
(1)                   0.00
                                   HLT
```

Listing 3: Another sample listing; this one is in trace mode. Trace mode outputs each program instruction and the data associated with it so you can check each individual program step. This is the SR-52's equivalent of a symbolic high level language in a larger machine.

Table 1: Functions of the user defined keys in the invoice program.

Key Name	Assignment when Button is Pushed
A	Data entered on key is the quantity of the merchandise purchased.
B	Data entered is the suggested list price of the merchandise. (Used with A.)
C	Data entered is the customers' discount.
D	Data entered is the shipping charge for the order.
E	Data entered is the net price (no discount) of the second category of merchandise. (Button A is also used with E.)
A'	This is the second function use of A. When pressed it tells the SR-52 to total the list price items, compute and subtract the discount and leave the net amount.
B'	This is the second function use of B. When pressed it tells the SR-52 to add the total of the net price items, if any, to the results obtained when A' was pushed. It then adds the shipping cost and results in the total for the order.

You should also write the user instructions at this time. Table 2 shows the user instructions for the invoice program. It is also helpful to the user if you attach a copy of a sample printout. The blank spaces on the magnetic program card should also be filled.

This invoice program, while being relatively simple, is certainly not a trivial program. It does involve many of the basic programming philosophies that would be applicable to writing a program for any system. With the prices of hand held calculators continuing to drop as they have in the past, the SR-52 and companion printer are not a bad way to get into the intricacies of computers. At the time of this writing, the combination can be bought for about $400. The fact that the SR-52 can be connected to the real life world (the printer), does hold a possible promise that a computer interface to the SR-52 might come along someday.

For those who have an SR-52 and need an invoice program, a listing of the program discussed in this article is provided in listing 1. The program uses 110 program steps, less than half of the total SR-52 capacity. This will allow space to modify the program to meet your specific needs.■

Step	Procedure	Press			Display
1	Load "A" side of card	CLR	2 nd	READ	Goes blank then 0
1a		run			
2	Enter discount (see note below)	C			Discount amt printed
3	Enter shipping	D			Shipping amt
4	Item quantity	A			Qty printed
5	Total list price of item Price printed Repeat 4 and 5 until all List price items entered	B			Individual and total price printed
6	Total of list items Prints list total Discount, net on list If there are net price items do 7 and 8 otherwise go to step 9	2 nd	A'		
7	Enter item quantity	A			Qty printed
8	Enter net item price Total net price printed Repeat 7 and 8 until done	E			Individual and total price printed
9	Net total printed, Then shipping and Invoice total printed	2 nd	B'		

Note: For 40% customers, enter .6;
For 50% customers, enter .5;
For 50+10+10, enter .405.

Table 2: User instructions for the invoice program. It is always necessary (for peace of mind) to write out these instructions, otherwise you may forget how to use a program.

Computer Information Arrangement

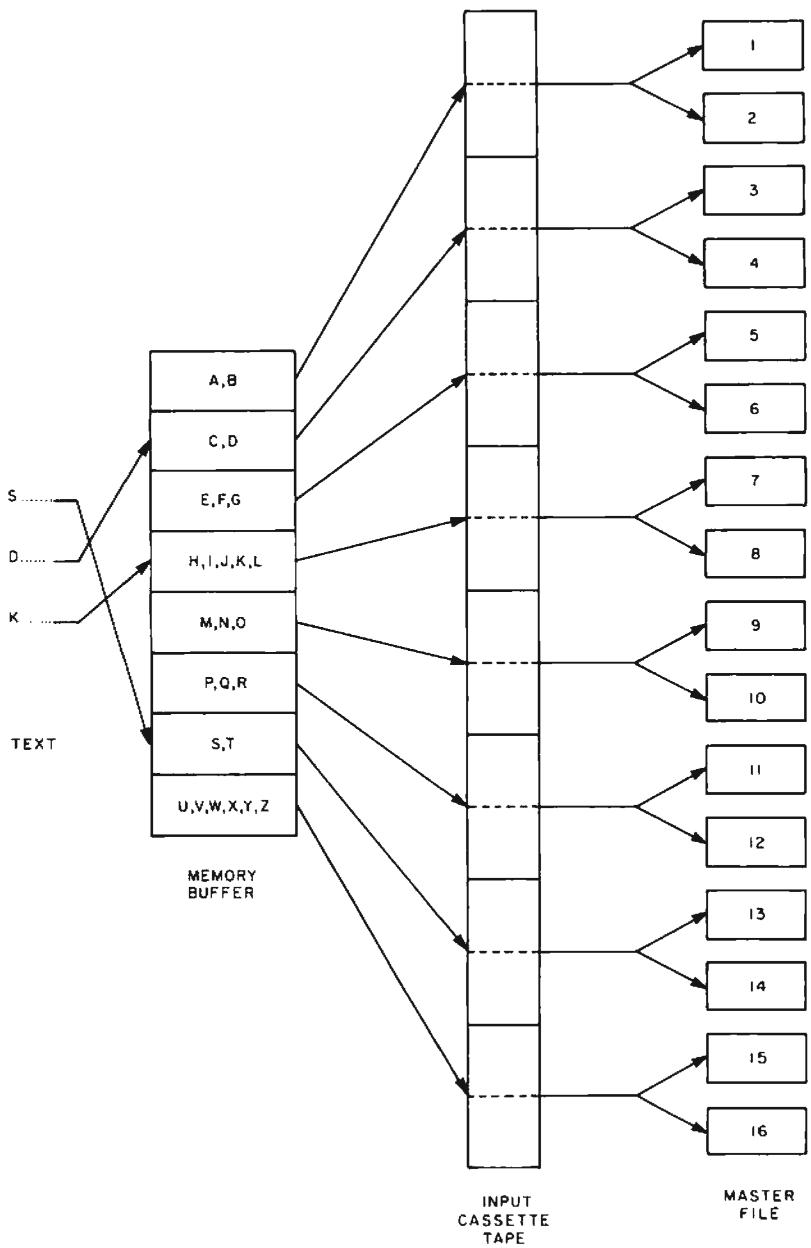


Figure 1: This is a basic diagram showing the input arrangement for the information retrieval system. The text is entered from a keyboard into a buffer area, and sorted alphabetically. In the example the three text strings start with S, D and K. When a certain buffer area is filled the information it contains is dumped to an input cassette tape. The information on the tape is sorted in alphabetical order. When any one tape is filled, or an updated master file is desired, the input tape is added to the master file.

David Holladay
225 Lakelawn Pl
Madison WI 53703

An examination of the small system computer field might lead the observer to take a limited view of the potential uses of small systems. This is unfortunate, because a computer, even a small one, can do more than play games or make lights blink.

One general application of computers is the information retrieval system. A classic goal of information retrieval is the construction of a system that absorbs the contents of books and can answer questions concerning the information contained in them. This goal has been unapproachable in even the largest of computer systems. The best approach is to put the burden of intelligence on the user's shoulders and make use of the computer's bookeeping ability. This reduces the program to a large scale sorting system tailored to a microcomputer's capabilities.

Small systems have limitations in memory size, data transfer rates and throughput. To cope with these limitations, I propose a mass information handling system called the Computer Information Arrangement, or CIA. The basic hardware required for this system includes a processor, 8 to 16 K bytes of programmable memory, keyboard, TV interface and several cassette interfaces with a data rate of at least 300 bps. One cassette drive has to be controllable by the computer in a manner beyond that of simple motor control.

The main storage memory for the huge data base is magnetic tape. Tape is slow and serial, meaning that only the information physically located near the tape head can be dealt with. However, it is cheap. For the moment, our data base will be a dictionary, ie: a list of definitions sorted alphabetically

Building a better computer wasn't easy. But we did it.

Introducing the MSI 6800 Computer System

When we set out to build the new MSI 6800 Computer System, we knew we had our work cut out for us. It had to be at least as good as the now famous MSI FD-8 Floppy Disk Memory System which is also pictured below. So, the first thing we did was analyze all the problems and drawbacks we had encountered with other 6800 systems, and then put our engineers to work on solutions. The objective: Build a better computer.

We started with power supply. We had big ideas, so we used a hefty 18 amp power supply. You can run full memory and several peripherals without the worry of running out of juice. We also put it in the front of the cabinet so it's out of the way.

The next step was the CPU Board. A separate baud rate generator with strappable clock outputs allows any combination of baud rates up to 9600. A separate strappable system clock is available and allows CPU speeds of up to 2 MHz. The new MSI monitor is MIKBUG software compatible, so you will never have a problem with programs. Additional PROM sockets are available for your own special routines and to expand the monitor. The CPU also contains a single step capability for debugging software.

When we got to the Mother Board, we really made progress. It has 14 slots to give you plenty of room to expand your system to full memory capability, and is compatible with SS-50 bus architecture. Heavy duty bus lines are low impedance, low noise, and provide trouble-free operation.

With all this power and potential, the interface had to be something special. So instead of an interface address in the middle of memory, we put it at the top . . . which gives you a full 56K of continuous memory. Interfaces are strappable so they may be placed at any address. An interface adapter board is compatible with all existing SS-50 circuit boards and interface cards. All MSI interface cards communicate with the rear panel via a short ribbon cable which terminates with a DB-25 connector. All baud rate selection and other strappable options are brought to the connector so they may be automatically selected by whatever plug is inserted into the appropriate interface connector. Straps may also be installed on the circuit board.

To complete the system, we used an MSI 8K Memory Board which employs low power 2102 RAM memory chips and is configured to allow battery back-up power capability. A DIP switch unit allows quick selection of a starting address of the board at any 8K increment of memory.

If you're one of those people who understands the technical stuff, by now you'll agree the MSI 6800 is a better computer. If you're one who does not understand it yet, you'll be more interested in what the system can do . . . play games, conduct research and educational projects, control lab instruments, business applications, or just about anything else you might dream up that a microcomputer can do. The point is . . . the MSI 6800 will do it better.

The MSI 6800 Computer System is available in either kit form or wired and tested. Either way, you get a cabinet, power supply, CPU board, Mother board, Interface board, Memory board, documentation, instructions, schematics, and a programming manual. Everything you need.

There is more to say about the MSI 6800 than space permits. We suggest you send for more information which includes our free catalog of microcomputer products.

Building a better computer was not easy. Becoming the number one seller will be.

See the MSI 6800 Computer System at Personal Computing '77 - Atlantic City.

Midwest Scientific Instruments

220 West Cedar • Olathe, Kansas 66061
913/764-3273 • TWX 910 749 6403 (MSI OLAT)

Midwest Scientific Instruments
220 W. Cedar, Olathe, Kansas 66061

NAME ____________________
ADDRESS ____________________
CITY ____________________
STATE ____________________ ZIP __________

0011077

Circle 411 on inquiry card.

by keyword. If the dictionary is closely packed on the tape, it will be difficult to add to it without shifting half of the data base. It would be more logical to spread out the entries on the whole tape to avoid later space problems. Unless the tape is getting full, the proper position of an entry is solely a function of keyword.

If entries are to be added in an efficient manner, close attention must be paid to differing data rates. A human can type two to five characters (bytes) per second, while a computer can take things on or off tape much faster. The typical personal computer can internally manipulate at least 250,000 bytes per second when programmed with an assembler. A video display can depict about 1,000 characters at a time, and can refresh itself 30 or 60 times a second, depending on the way the display handles the interlace. It will be the objective of the CIA system to put information onto the cassette tape in sorted order as fast as the user can type in the unsorted data. The user can therefore type the definition of "best" and of "machine" into our imaginary dictionary, and the computer will place both in their proper places on the tape.

A large part of main memory (at least 4 K bytes) is used as a buffer. As new data is typed in, it is added to the buffer and sorted on keyword. This sorting can be done by rearranging the data in the buffer in sequential order. An alternative is to keep items in unsorted order and maintain two pointers for each item, one pointing to the location of the item which is next in sorted order, the other pointing to the previous item in sorted order. The second system eliminates unnecessary searching in memory, but involves longer and trickier programming.

As the memory buffer accumulates data, it is important to keep track of how it is filling up. The alphabet is divided into eight sections for accounting purposes. The first section may be for words starting with A or B, etc. Eight counters keep track of how many bytes are taken up in the buffer by different ranges of the alphabet. When one counter exceeds certain limits, the cassette is moved to the region of tape corresponding to that range of the alphabet. Next, the data held in the buffer is transferred onto the tape in the proper location. Obviously the data would then be erased from the memory buffer to make room for more data. The end result is a cassette tape containing sorted information which generated at the same rate that the user is typing in the unsorted data (see figure 1). If the data is sorted as fast as it can be put in, what would be the advantage of greater throughput? The system works just as fast as it has to.

Notice how general the system is. It can be used to make huge mailing lists, keep track of books in a library, and so on. It has two principle limitations in addition to speed, size and the simple nature of the data that it can handle.

What can you do when you fill up a cassette? (It may take a while, since it is possible to fit as many as 500,000 characters on a digital cassette tape.) You could maintain a master set of 26 tapes, one for each letter of the alphabet. Once your tape is full, it would be merged with the master file, an unwieldy process at best. This procedure would mean putting master tape #1 in one cassette machine, your input cassette in another, and starting the merging program. After a while, the computer could signal that it had put all of the A entries onto master tape #1. Then you would take out tape #1, replace it with tape #2, and so on, up to tape #26. This process would happen rarely (or as often as you would require an up-to-date master file). But imagine how much data your system could hold. A friend of mine pointed out that if you were mechanically inclined, you could automate cassette manipulation. It would be a cross between a jukebox and large scale automated mass memory with media manipulation mechanisms. An automatic multimegabyte memory system for a few thousand dollars would be most impressive.

The question of data bases is a bit tricky. Data can be abstract, highly interrelated, and difficult to categorize. Your data will be interrelated in ways that the data base cannot show or represent. There are two approaches to follow. One way is to design very abstract data structures that show relationships inherently. The other is to maintain the simple dictionary alphabetic system, and add several cross-reference pointers. An example would be "Kennedy, Jackie: see Onassis, Jackie." By pursuing all the pointers listed under a keyword, and checking out all the pointers listed there, a tree structure is developed. A multi-cassette data system implies a significant amount of tape manipulation, unless you have built the jukebox system.

Although a pointer system is a bit crude, it can be handled automatically. The following example illustrates a typical entry. The original data entry: "Beethoven, Ludwig van, Symphony Number 3 (The Eroica)" would be filed under "Beethoven, Ludwig van". If the user wants to generate the cross reference pointer "Eroica Symphony, see Beethoven, Ludwig van", a special character could be typed before "Beethoven, Ludwig van" which the program would recognize. The program could then add "Eroica Symphony, see Beethoven, Ludwig van" to the text buffer. This would insure that the pointer and the data match, eliminating a problem with typographical errors.

Later, if it is necessary to eliminate the entry, you would know that the cross-reference pointer is also in memory because of the existance of the special character. Other special functions can be implemented by special characters, such as labeling the data source of facilitating tabular data. This is left as an exercise for the reader. The power of this information handling system is limited mainly by the size of programs that can be stored in memory, and by the speed of the tape recorder.

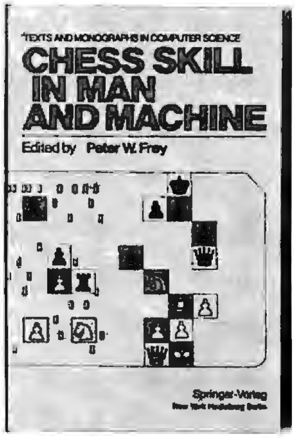
The Computer Information Arrangement needs five separate programs to work properly. Note that it is not necessary for more than one to be in memory at any time. Program 1, the input program, is the biggest and most difficult. It accepts characters from the keyboard, edits them, adds the cross-reference, puts them in the buffer, recognizes when the tape machine is idle or part of the alphabet range is getting full in the buffer, and spreads the data on the input cassette by means of a linear hashing formula. It may be necessary for the tape recorder to be controlled by a separate microprocessor and 1 K bytes of programmable memory shared by both processors, because of timing considerations. The second one is the merge program, which merges the input tape with the master set of cassettes. The third program, called the clean up, goes through tapes, "unbunches" data, and straightens out any local area that gets "out of sort." The fourth program is the display program. The user can tell it to display the Richard Nixon file, whereupon it will display all the references and pointers that are filed under the keyword Nixon, Richard. The last program does a crucial, but easily forgotten job: altering or deleting outdated or incorrect data from the input tape or from a master tape.

The CIA is a general computer information arrangement, an answer machine, or a list maker. Put in randomly ordered data and it comes out neat and organized. The arrangement has many applications: small business, journalism, research, or help for folks who have trouble organizing things. This is the type of program which will sell small systems to the world. ■

GLOSSARY

Alphabet range: part of the contiguous alphabet used to decide where to store alphabetically sorted data.
Buffer: section of random access memory used to temporarily store data until enough is collected to pass on.
Cross reference: a notation to look elsewhere in the data base for more information.
Data base: collection of information and the system used to organize it for use by a computer program.
Entry: A block of data that stays together during the sorting routine. The end of an entry is recognized by a special termination character.
File: Set of entries with the same keyword.
Input tape: cassette that accepts the sorted data. For larger data bases it must be merged with the master tapes.
Keyword: Word in an entry used to sort the entire entry into the data base.
Master tapes: Set of cassettes that make up the entire data base. Each cassette covers a portion of the alphabet range.

Book Reviews



Chess Skill in Man and Machine *edited by Peter W Frey, Springer-Verlag, New York, 1977. Hardcover $14.80.*

Chess Skill in Man and Machine is a series of eight essays dealing with four main topics, two major and two minor. These topics include the history and rules of computer chess tournaments, human chess skill, descriptions of the techniques of computer chess, and finally, an overview of the past, present and future of computer chess. The chapters of most interest to the computer chess enthusiast are those explaining human chess skill and the various computer chess techniques.

The first major topic is an analysis of chess skill in humans. Neil Charness in chapter 2 poses the question, "Should a computer be more like a human?" He describes the two schools of thought concerning this question. One school answers no, because computer hardware operates differently from the human mind and should therefore be operated in its most efficient mode, not like the human mind. The other school says yes, a computer should be like a human if it is to play and beat humans. The latter school believes that human thinking should be simulated in order to enable the computer to capitalize on any human miscalculations.

Charness believes that to write a better chess program, the programmer has to understand how humans solve the problem of choosing the best move. "Chess-specific" perception is the key. Searching through a tree of moves which consists of every possible move during an entire game of chess is an extremely inefficient way to play chess. An analysis of the legal moves for each side to a depth of six moves during a single turn of a player would involve over six billion different combinations of moves.

Becoming a good chess player demands that one learn to recognize types of positions. This chess-specific perception (or pattern recognition) can reveal weaknesses for further examination, thereby cutting down on the number of moves to be examined. This method of "pruning the search tree" is the human method; it should also be the computer method.

The second major topic deals with different theories of computer chess. Chapter 3, written by editor Peter W Frey, introduces opening move theory (whether to store standard openings or to simply play them from scratch every time), machine representation of the chessboard (using either bit patterns or the "mailbox" method), and move searching procedures.

There seem to be two general types of search strategies in use today: the A type (looking at all legal moves and countermoves at each position), and the B type (looking at only a small subset of the potential legal moves). Frey prefers the B type strategy over the A type because of the number of positions to be examined.

The B type suggests that a minimum of moves from any position should be evaluated using von Neumann and Morgenstern's minimax procedure. The idea of backward pruning using a heuristic method is also covered, involving evaluation of all capturing moves first, and so on down a list of types of moves. The various search strategies are given in detail in chapter 3 and are elaborated on in chapters 6 and 7.

Northwestern University's program, CHESS 4.5, is covered by the authors of that program. CHESS 4.5 is particularly relevant to the programmer who has a grasp of the theory and who is eager to design his or her own program. The creators of CHESS 4.5 discuss their older programs and the writing of their latest program. They discuss their particular searching strategy, how they came to decide upon their method of generating moves and evaluating them, and their reasons for reaching certain decisions. This discussion tends to bring one down from the theoretical level to the practical level of decisions concerning implementation of theories.

The book begins with a retrospective of past computer chess tournaments and their rules, and ends by asking the question, "Why program a computer to play chess?". I think there are two answers to this question. A computer chess program to simulate human methods of playing will greatly aid our understanding of human perception, learning and problem solving. Secondly, it can be fun solving a difficult problem in a creative, imaginative way.

Chess Skill in Man and Machine is an interesting book filled with the theoretical and the specific concerns of computer chess. It should give readers the background to get started, and its bibliography should provide a basis for further inquiry.

Dennis Barbour
2 Coliseum Av
Nashua NH 03060

The LATEST in Tape Systems

MODEL CC-8 $175.00

4800 BAUD CASSETTE RECORDER

An ASYNCHRONOUS NRZ type Recorder with remote motor start/stop. Error rate 10^8 at 4800 BAUD. Can be used from 110 to 4800 BAUD into a UART or "Bit Banger PIA" –no clocking required. This is not an audio recorder. It takes RS232 or TTL signals from the terminal or computer and gives back the same signals. No audio interface is used. Motor start/stop is manual or through TTL or RS232 signals.

Tape speed is 3.2"/second nominal; 1.6"/sec. optional. 110 volt, 60 Hz, 5 watts. (220 Volts on special order). Can use high quality audio cassettes (Philips Type) or certified data cassettes. Can be used in remote locations from a 12 Volt battery.

Recommended for DATA LOGGING, WORD PROCESSING, COMPUTER PROGRAM RELOADING and DATA STORAGE. Especially recommended for 6800 systems, 6502 systems, 1800 systems and beginners with the 8080 systems. Manual control except for motor start/stop. 6800 or 8080 software for file or record searching available on request with order. Used by major computer manufacturers, Bell Telephone and U.S. Government for program reloading and field servicing.

AVAILABILITY – Off the shelf.

---

50 KILOBAUD CARTRIDGE RECORDER

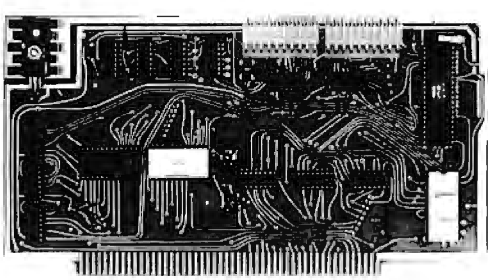
This is a self clocking (1/1) high speed recorder. Loads BASIC in under 2.0 seconds. Recording is done at 19.2 Kilobaud. Playback at 30 Kilobaud. File or Record searching is done at 50 Kilobaud and loading is automatic. Worst case access time about 2 minutes for up to 2 megabytes on the 3M Data Cartridge.

Tape speed 10"/sec. on record, up to 30"/sec. on playback. Records one clock track and one data track on each pass (2 passes). Recording is NRZ unencoded from RS232 or TTL signals.

This recorder requires one Parallel port for motor control, and one serial port for data and clock. (Cannot be used with UART's or UART boards such as the 3P+S). Used with USART's, ACIA's or other 1/1 clocking I/O devices under software control only. No manual controls. Software for 8080 and 6800 available. Power supply is built in, 110V, 60 Hz. 220 V, 50 Hz for export.

AVAILABILITY–90 days.

MODEL 3M3A $220.00



2SIO (R) CONTROLLER
$190.00 ($160.00 Kit)

PROVIDES MONITOR AND TAPE SOFTWARE in ROM TERMINAL and TAPE PORTS on SAME BOARD CONTROLS ONE or TWO TAPE UNITS (CC-8 or 3M3A)

This is a complete 8080, 8085, or Z80 system controller. It provides the terminal I/O (RS232, 20 mA, or TTL) and the data cartridge I/O, plus the motor controlling parallel I/O latches. Two kilobytes of on board ROM provide turn on and go control of your Altair or Imsai. NO MORE BOOTSTRAPPING. Loads and Dumps memory in hex on the terminal, formats tape cartridge files, has word processing and paper tape routines. Best of all, it has the search routines to locate files and records by means of six, five, and four letter strings. Just type in the file name and the recorder and software do the rest. Can be used in the BiSync (IBM), BiPhase (Phase encoded) or NRZ modes with suitable recorders and interfaces.

This is Revision 7 of this controller. This version features 2708 type EPROM's so that you can write your own software or relocate it as desired. One 2708 preprogrammed is supplied with the board. A socket is available for the second ROM allowing up to a full 2K of monitor programs.

Fits all S100 bus computers using 8080 or Z80 MPU's. Requires 2 MHz clock from bus. Cannot be used with audio cassettes without an interface. Cassette or cartridge inputs are RS232 level.

AVAILABILITY – Off the shelf.

NOW AVAILABLE: 6800 I/O with ROM, same as our 2SIO (R) but fits SWTP bus. $190 tested. ($160 kit)

NEW: SWTP bus 8085 MPU board. Use 8080 software in your SWP 800 system. $190 tested. ($160 kit)

OVERSEAS: Export Version 220 volt 50 hz. Write factory or: Megatron-Datameg, 8011 Putzbrunn, Munchen, Germany; Nippon Automation 5-16-7 Shiba, Minato-Ku, Tokyo, Japan; Hobbydata, FACK 20012, Malmo, Sweden; G. Ashbee, 172 Ifield Road, London SW 10-9ag: Trintronics, Ltd., 186 Queen Street W., Toronto, Ontario, Canada; EBASA, Enrique Barges 17, Barcelona 14, Spain; ARIES, 7, rue Saint Phillipe du Roule, 75008 Paris; Microlem 20131, Milano, Italy; Eagle Electric, Capetown, S. Africa.

For U.P.S. delivery, add $3.00 Overseas and air shipments charges collect. N.J. Residents add 5% Sales Tax. WRITE or CALL for further information. Phone Orders on Master Charge and BankAmericard accepted.

National Multiplex Corporation

3474 Rand Avenue, South Plainfield NJ 07080 Box 288 Phone (201) 561-3600 TWX 710-997-9530

Circle 413 on inquiry card.

Exciting lectures sponsored by BYTE

The Small Systems Journal

Personal Computing Expo is endorsed by BYTE magazine, whose staff has contacted prominent speakers for an exciting series of lectures.
Visitors will be able to attend these meetings **free of charge.** The lectures will not conflict with each other eliminating the worrysome choice among several equally important topics. In addition, they will be repeated on the next day to give you a second chance if you missed a topic.
Lectures are typically 30 minutes, often with demonstrations and an additional 15 minutes for questions.

Portia Isaacson Ph.D. . . . Saturday 11 AM and Sunday 12 Noon

Co-owner of the Micro Store, a personal computer store, in Richardson Texas actively engaged for 12 years in the computing field in industry and at universities.
Member of the ACM and IEEE, and chairman of the 1977 National Computer Conference.
Author of many articles in professional journals and magazines.
Received a Ph.D. in Computer Science from the Southern Methodist University.

1. Personal Computing: An Idea Whose Time Has Come

A review of what has happened so far in the personal computing field, and an outlook into future developments, including those in the computer assisted home. Slide demonstration.

Sol Libes . . . Friday 6 PM and Saturday 10 AM

President of the Amateur Computer Group of New Jersey
Teacher of electronics and computer programming at a community college
Author of 10 books (working on the 11th) and several hundred magazine articles in electronics and computing
Received an award for "The Outstanding Amateur Computer Hobbyist of 1976" in Atlantic City by Personal Computing 1976 show, and BYTE.

2. How to Get Started

A discussion of typical home computer systems and their essential hardware and software components.

John H. Dilks III . . . Saturday 1 PM and Sunday 2 PM

President of Personal Computing Inc. and tradefair director of the Personal Computing shows in Atlantic City in 1976 and 1977.
Experience with various computer systems since 1962.
Employed by Western Electric Co. Inc., division of the Bell System.
Member of the Amateur Computer Group of New Jersey and of the Philadelphia Area Computer Society.
Teaches microcomputer courses in an adult evening program at a vocational school.

3. Innovative and Unusual Computer Applications for the Home.

Discussion of "far-out" applications of microcomputers and electronic technology for home use, such as a child locator and warning device, a home security system, etc. Slide demonstration.

Robert S. Jones . . . Friday 7 PM and Saturday 2 PM

Publisher of Interface Age Magazine
Prior experience in sales and marketing for the semiconductor industry, including Intel, National Semiconductor and Analog Devices Inc.

4. Personal Computing for the Business Man

Evaluating business applications for micro computers, including slides showing selected applications.

Louis E. Frenzel . . . Saturday 3 PM and Sunday 3 PM

Director of Computer Products at Heath Company, involved in the planning of new computer products.
Prior to Heath, with McGraw Hill in product planning and design of educational electronic kits.
Prior experience including computer engineering for eight years
Author of several books, home study courses and numerous magazine articles in electronics and computers.
Received a BS in electronics from the University of Houston and a MEd from the University of Maryland

5. How to Build Personal Computer Kits

Tips for successful kit construction. Benefits of kit products for the personal computer user. Including slides showing selected computer kits.

Carl Helmers . . . Friday 9 PM and Saturday 4 PM

Editor-in-Chief and co-founder of BYTE magazine.
Obtained computing experience as a personal way to accomplish artistic and technological goals in music.
Graduated in 1970 with a BS in Physics from the University of Rochester, NY.
Worked for several years at Intermetrics, Inc. in Cambridge, Massachusetts on the NASA Space Shuttle Project.
Prior to working with BYTE, publication of a small computer newsletter on a part-time basis.

6. Computers and Music

How to create music with computers. Problems of performing electronic music, music under computer program control and computer music in conjunction with traditional instruments. Illustrations and examples from personal experience.

Jack L. Davies . . . Friday 8 PM and Saturday 5 PM

President of Pan Atlantic Computer Systems GmbH, a distributor of various micro computer systems in Europe.
Extensive experience in using minicomputers and microcomputers in the US Military Schools in Europe. Designed and developed numerous games and educational programs for students in these schools.

7. Microcomputers in Education

Discussion of the many possibilities of using microcomputer systems in schools. Effect of personal computers on students.

David Fylstra . . . Saturday 6 PM and Sunday 5 PM

Member of the research staff of the Telecommunications Sciences Center at Stanford Research Institute for more than two years. Specialized in microcomputer software and computer simulation of speech processing systems.
Graduated in 1974 with a BA in English and Psychology, Stanford University, Phi Beta Kappa.
Active in the research on communication systems and devices for the deaf.

8. Speech Analysis and Synthesis for the Amateur

Using the personal computer as a device to analyze the acoustical foundations of speech and to formulate rules for the control of the speech synthesizer.

Max Mathews Ph.D. . . . Saturday 7 PM and Sunday 4 PM

Director of Acoustical and Behavioral Research, Bell Labs
Author of **The Technology of Computer Music**, and numerous articles.
Scientific Advisor to the Institute for Research and Coordination of Acoustics and Music (IRCAM)
Dr. Mathews is often regarded as the "Father of Computer Music"

9. Pure Digital and Real Time Music Synthesis

The use of the digital computer as a musical instrument with which composers and performers create and play music. Slide and tape demonstration.

Carl L. Holder . . . Saturday 8 PM and Sunday 6 PM

Director of Product Management, Planning and Communications at Information Terminals Corp. for five years.
Prior experience, including Memorex Corp., in the area of magnetic media development and testing.

10. Present and Future Storage Devices

Survey and discussion of current devices and media, including latest technological developments like the charge coupled devices and magnetic bubble memories. Costs, advantages and disadvantages of these devices for the personal computer user. Accompanied by slides.

DAILY TUTORIALS

There will be two tutorials offered each day, one aimed at those participants who have little or no experience with microprocessors, and the other for those already experienced with microprocessor systems.

BASIC COURSES

1. Development of Microcomputer Systems for Business Use . . . Friday 10AM to 5PM
Sy Ratner, Citibank, N A.
— system design concepts
— economic efficacy of microcomputers versus large centralized computer systems
— distributed processing definitions
— advantages and problem areas
— network design and architecture
— data communications aspects
— case study: design of a stand-alone workstation for data entry and retrieval

2. Development of Microcomputer Systems for Home Use . . . Saturday 10AM to 5PM
Cary Ringel, Con Edison
— survey of simple microcomputer control systems for home use
— selection of hardware: IC's, boards, kits, development systems
— programming and software aids
— interfacing: A/D and D/A conversion
— examples: design of a home control system; microcomputers for a music synthesizer; computer TV games
— case study: use of the Motorola 6800 in design of a microcomputer system

3. Survey and Comparison of Microprocessors . . . Sunday Noon to 5PM
Donald Lewis, Standard Microsystems Inc and other speakers
— definitions and distinctions between ALU-chips, controllers, microprocessors, microcomputers
— current applications
— microprocessor architectures (bit-slice, 4-bit, 8-bit, 16-bit, minicomputer-type)
— vendor survey
— performance evaluation and criteria for selection

INTERMEDIATE/ADVANCED COURSES

4. Microprocessor Interfacing . . . Friday 10AM to 5PM
Donald Lewis, Standard Microsystems Inc.
— interface components (peripheral interface chips, UARTS, etc.)
— interface standards (IEEE 488, RE 232C, S-100, etc.)
— interfacing to keyboards
— interfacing to casette and floppy disk drives
— interfacing to display devices
— case study: how to design a CRT terminal

5. Microprocessor Programming and Software . . . Saturday 10AM to 5PM
Donald Lewis, Standard Microsystems Inc
— software design: flow-charting, setting breakpoints, documentation, etc.
— assembly language for the Intel 8080, 8085, Z-80, Motorola 6800
— instruction types and addressing techniques
— use of the stack
— interrupt handling and direct memory access (DMA)
— software development aids
— high level languages for microcomputers

6. Technology Analysis and Forecast of Future Microprocessor Structures . . . Sunday Noon to 5PM
Daniel R. McGlynn, U.S. Philips Corp. and Will Mathys, MOS Technology Inc.
— emergence of specialized computational elements (SCE)
— architectural evolution (stack processors, reconfigurable architectures, multi-level logic)
— resource management techniques
— software evolution (nano-programming, extensible instruction sets, structured programming modules, very-high-level languages)
— evolution of semiconductor technology of microprocessors
— microprocessor architecture at the chip level
— case studies: design of MOS Technology's new 8-bit and 16-bit processors

Personal Computing Expo to be produced by H.A. Bruno & Associates

H.A. Bruno & Associates, Inc., has been prominent in the exposition and promotion fields since 1923. Highly skilled in the production and promotion of consumer and trade shows, the company currently promotes the American Energy Expo, the National Boat Show, Auto Expo/New York. Promotion assistance also is currently rendered to the National Computer Conference and the Triennial IFIPS Congress in Toronto.

The show producer has promoted successful shows in the New York Coliseum every year since the building opened in 1957. Staff personnel are thoroughly familiar with the building, its services, management and labor.

Interesting, educational exhibits of Personal Micro Computers

The lectures and tutorials are not just theory! You will be able to see a multitude of micro computers. Try out various demonstrations: fascinating games, even in color; small business applications; computer produced speech; music; color graphics; and many more. Micro computers are not only small and portable, also you will need only a "micro budget" to be able to take them home.

Show Hours and Admission

Personal Computing Expo hours are as follows:
Friday, Oct. 28 — Noon to 10 p.m.
Sat. Oct. 29 — 10 a.m. to 10 p.m.
Sunday, Oct. 30 — Noon to 7 p.m.
General Admission: $5.00 (includes free BYTE lectures) per day.
Two-day Tickets: $9.00 (advance sale only)
Three-day tickets: $13.00 (advance sale only)

General Information

You may find it advantageous to purchase two or three-day admission tickets in advance. These are availble by mail only, no later than October 10, 1977. Use coupon below.

Group rates (10 or more persons) qualify for $1.00 off regular prices. Arrangements must be made by mail prior to October 10, 1977.

Special arrangements have been made if you desire to stay overnight. Our headquarters hotel, the Barbizon-Plaza, is located on Central Park South, two blocks from Columbus Circle. Single rooms available at $34.00 per night; $40.00 double, plus tax. There's a weekend plan: $22.95 daily, plus tax per person, double occupancy . . . includes breakfast (brunch on Sunday) and meal gratuities. Children under 14 in same room with parents, free.

For hotel reservations and information, call toll free (800) 223-5493. From New York State call (800) 223-5963.

For those traveling to New York by air, American Airlines offers a convenient service through arrangement with Personal Computing Expo. For information, call toll free (800) 433-1790. In Texas the number is (800) 792-1150. From the West Coast, round trip fare via American is only $227.00.

20,000 persons are expected to attend and view the more than 200 exhibits by personal computer manufacturers and retailers.

Personal Computing Expo will occupy the 4th floor of the New York Coliseum. It is located on 59th Street and Columbus Circle — the geographical center of New York City. Garage parking in the building is available.

For answers to any questions pertaining to your attendance at Personal Computing Expo, contact the Show Manager, Ralph Ianuzzi, at Area Code 212/753-4920.

Advance sale of tickets available . . . Three days $13 Two days $9 . . . General Admission $5 MAIL THIS CONVENIENT ORDER FORM NOW!

Personal Computing Expo admission is $5.00 per day. Advance reservation eliminates waiting in line. Order advance tickets with this coupon. Admission ticket includes access to exhibits, lectures and tutorials.

Please send me ________________ advance registration tickets for three days, October 28-29-30. Total cost $13.00 per person.
Please send me ________________ advance tickets for two days, October ______ and October ______. Cost is $9.00 per person.
Please send me ________________ advance tickets for one day, October ______. Cost is $5.00 per person.

Make all checks payable to PERSONAL COMPUTING EXPO, and mail to:
Personal Computing Expo, 78 East 56th Street, New York, N.Y. 10022.

Name________________________________ Amount enclosed $____________
Address ____
City____________________________State______________Zip____________

Analyze Your Car's Gas Economy with Your Computer

John P Bauernschub
14809 Clavel St
Rockville MD 20853

In a note accompanying this article, author Bauernschub reported: This program was developed to be displayed on a video monitor using the SwTPC CT-1024. However, I borrowed a Texas Instruments Silent 700 to generate the listing and was happy at the ease of interfacing it to the SwTPC MC-6800 and AC-30 cassette interface.

A few comments about myself. First, I had never put an electronic kit together before this. I spent six months studying the microcomputer market and decided on Southwest because they appeared to have "put it all together." It was enjoyable and fairly straightforward assembling the kits, and they all worked.

How well is your car performing? Is it matching the EPA miles per gallon ratings that the manufacturer posted on it? Is it time for a tune-up? Here is a system that will produce these answers and provide a worthwhile application for your personal computer.

The first step in this system is to gather data. What we're out to do is automate a familiar procedure: analysis of data about your car. But analysis requires data. Every time you purchase gas, record the quantity and the mileage. It helps to keep a small notebook in the glove compartment of your car for this. The system will be the most informative if the tank is filled each time. However, the fill-up is not necessary every time since the program totals the gallons and miles for an overall rate. You may also wish to note when you do an extraordinary amount of highway or city driving so you can identify its effect on your normal miles per gallon.

If you want to check your fuel performance and have not been recording detailed fill-up data, you can recover past data from gas receipts (on your credit card bills) and get a starting mileage from a shop maintenance record.

When the program is initiated you have the option of displaying the instructions. Next you are asked for the mileage before the first gas purchase. This is either the mileage when you first started recording the data or the last mileage used the previous time the program was run. For the greatest accuracy, the tank should be full for the first and last recordings. This value is stored by the program in the variable L to be used in the final statistics, and assigned to M2 for the first computation.

Column headings are displayed and you are requested to input mileage and gallons for the first fill after the mileage used as the start. The program will then display the mileage, the number of gallons, the number of miles since the last purchase and the miles per gallon for this data. You are again prompted for miles and gallons and the cycle repeats itself.

If you detect an error in the most recent entry, you can back it out by entering a negative number for the next mileage and zero for gallons. When this is done, the number of gallons for the most recent entry, G4, will be subtracted from the total gallons, T, (line 370); and the most recent mileage variable, M2, will be set equal to the second most recent mileage variable, M4. When you are prompted for mileage and gallons again, enter the corrected amount. If this is also in error, it can likewise be backed out. However, you can only back out the most recent entry. If you tried to back out multiple levels of entries by entering successive negative values for mileage, nothing would happen because after the first back out G4 was set to zero (line 380) and M2 equals M4.

After you have entered your last data and are prompted for miles and gallons, enter a zero for each and your computer will tell you how many miles you drove, how many gallons were used, and your average miles per gallon. This miles per gallon figure is computed using the total gallons and miles driven and not an average of each previously computed miles per gallon.

This program was written for a SwTPC 6800 and will run with either 4 K or 8 K BASIC. Its output was designed to fit the SwTPC CT-1024 video screen and will therefore appear squeezed on a wider screen or printer. This can be adjusted by changing the spacing of the column headings in line 700 and the tabs in line 320. The statement PRINT CHR$ (16); CHR$ (22); in lines 450, 520 and 690 are the computer cursor commands (home up, erase to end of frame) for the CT-1024 TVT and should be changed or left out if some other terminal is utilized. Also, all REM statements can be left out at the risk of losing some documentation value.

Carry this idea forward in your computer to help keep track of the performance of your car.■

Listing 1: The BASIC program for analysis of automobile mileage data. This program is written for the SwTPC 6800 system's 4 K or 8 K BASIC interpreters, and can easily be adapted to any interpreter which implements a minimum of decimal arithmetic and string output operations for formatting. This program is not recommended for use with "tiny" BASIC interpreters.

```
READY
LIST

0100 REM PROGRAM TO COMPUTE GAS MILEAGE
0110 REM BY JOHN F. BAGERT HOU., TX.
0120 REM APRIL 1977
0130 PRINT "INSTRUCTIONS (1=YES, 0=NO)";
0140 INPUT X
0150 IF X=1 GOSUB 520
0160 IF X=0 GOSUB 650
0170 LET T=0
0180 PRINT "M,G";
0190 INPUT M,G
0200 REM ERROR DETECTION
0210 IF M=0 THEN 420
0220 IF M<0 THEN 370
0230 IF M<=M2 THEN 720
0240 IF G<=0 THEN 750
0250 REM SAVE DATA FOR BACK OUT
0260 LET M4=M2
0270 LET G4=G
0280 REM MILES PER GALLON COMPUTATION
0290 LET M3=M-M2
0300 LET T=T+G
0310 LET R=INT(100*M3/G+.5)/100
0320 PRINT TAB(5);M;TAB(13);G;TAB(19);M3;TAB(25);R
0330 LET M2=M
0340 GOTO 180
0350 REM BACK OUT ERRORS FROM TOTAL
0360 REM GALLONS (T) AND LAST MILEAGE (M2)
0370 LET T=T-G4
0380 LET G4=0
0390 LET M2=M4
0400 GOTO 180
0410 REM FINAL STATISTICS
0420 LET M1=M2-L
0430 IF T=0 THEN 750
0440 LET R=INT(100*M1/T+.5)/100
0450 PRINT CHR$(16);CHR$(22);
0460 PRINT "YOU DROVE ";M1;" MILES ON"
0470 PRINT T;" GALLONS OF GAS"
0480 PRINT "FOR AN AVERAGE OF"
0490 PRINT R;" MILES PER GALLON"
0500 END
0510 REM INSTRUCTIONS
0520 PRINT CHR$(16);CHR$(22);
0530 PRINT "       COMPUTE MILES PER GALLON"
0540 PRINT
0550 PRINT "INPUT MILEAGE AND GALLONS FOR"
0560 PRINT "EVERY GAS PURCHASE WHEN ASKED"
0570 PRINT "FOR M,G"
0580 PRINT "IF AN ERROR IS MADE, ENTER ANY"
0590 PRINT "NEGATIVE NUMBER FOR THE NEXT"
0600 PRINT "M AND 0 FOR G.  THIS WILL"
0610 PRINT "BACK OUT THE PREVIOUS ENTRY"
0620 PRINT "FROM THE GRAND TOTAL."
0630 PRINT "TO FINISH, ENTER 0,0"
0640 PRINT "WHEN READY TO START...   "
0650 PRINT "INPUT MILEAGE BEFORE THE FIRST"
0660 PRINT "GAS PURCHASE...   ";
0670 INPUT L
0680 LET M2=L
0690 PRINT CHR$(16);CHR$(22);
0700 PRINT "   MILEAGE  GALS  MILES   MPG"
0710 RETURN
0720 PRINT "MILEAGE LESS THAN OR EQUAL"
0730 PRINT "PREVIOUS      ";
0740 GOTO 180
0750 PRINT "GALLONS MUST BE GREATER THAN 0"
0760 GOTO 180

READY
#RUN
INSTRUCTIONS (1=YES, 0=NO) ? 0
INPUT MILEAGE BEFORE THE FIRST
GAS PURCHASE...      ? 23400
   MILEAGE  GALS  MILES   MPG
M,G? 23620
? 19.8
      23620  19.8   220    11.11
M,G? 23795,17.8
      23795  17.8   175    9.83
M,G? 24007,19.6
      24007  19.6   212    10.81
M,G? 24218,18.4
      24218  18.4   211    11.46
M,G? 24366,14.1
      24366  14.1   148    10.49
M,G? 0,0
YOU DROVE 966  MILES ON
89.7  GALLONS OF GAS
FOR AN AVERAGE OF
10.77  MILES PER GALLON

READY
#
```

W Lloyd Milligan
8604 Maywood Dr
Columbia SC 29209

Mastermind

*Mastermind has become one of the most popular games around these days. (*Games and Puzzles *magazine gave it their highest rating.) Among other applications, it is used by a number of computer companies to test the programming skills of potential employees. The game was originated by Invicta Plastics; Mastermind is a trademark of that firm. In its original form, this game of skill requires a mind, but no computer.*

One of the most interesting conventional (ie: noncomputer) games on the market is "Mastermind," distributed by Invicta Plastics, Suite 940, 200 5th Av, New York NY 10010, and available in many local stores. Mastermind involves deductive logic, hypothesis testing and probabilistic inference. In Mastermind, the players take turns as "codemaker" and "codebreaker." The codemaker sets up a concealed row of four colored pegs from a set of **R**ed, **BL**ue, **BR**own, **G**reen, **Y**ellow and **O**range pegs. It is acceptable to use the same color or colors more than once. In version 2, a more advanced game, empty **S**paces are also permitted. (The bold face characters are abbreviations recognized by the BASIC program.)

The codebreaker has ten tries (rows) in which to discover the secret arrangement of colors in the concealed row. To input a row to the computer, type the legal abbreviation for the leftmost color on the first line, the next to left on the second line, and so on. After each row has been typed in, the program evaluates the try and types back the following message:

> YOU GET n BLACK AND m WHITE
> PEGS ON ROW i.

The number of black pegs corresponds to the number of correct colors in correct positions, and the number of white pegs gives the number of correct colors in wrong positions. An important rule is that no position in the try is counted more than once. For instance, if you have Green in columns 1 and 3, and the secret code has Green in 3 only, you get a black peg for Green in position 3 but no white peg for Green in 1. If you fail to break the code in ten rows, the codemaker program will tell you the answer and let you play again. It helps to study a missed game to see where you made faulty deductions.

```
MMM      12-JAN-77  BASIC V01B-02

10 REM MASTER MIND 'CODEMAKER'
20 REM CODED IN RT-11 BASIC
40 RANDOMIZE
60 PRINT 'MASTER MIND GAME'
110 REM INITIALIZATION
120 DIM A$(6),B$(3),C$(3),D$(3),S(3)
130 FOR J=0 TO 6
140 READ A$(J)
150 NEXT J
160 DATA 'R','BL','G','Y','O','BR','S'
210 PRINT 'WHICH VERSION (1 OR 2) ';
230 INPUT R
240 LET R=R+5\REM R WILL BE USED IN SUBSCRIPT COMPUTATION
270 REM CODE SELECTION
280 FOR J=0 TO 3
290 LET D$(J)=A$(INT(R*RND(J)))
300 REM THE SAME COLOR IN MORE THAN ONE POSITION IS LEGAL
330 NEXT J
340 REM GAME PLAY BEGINS HERE. I IS ROW COUNTER
345 REM B$ IS COPY OF D$ THAT CAN BE ALTERED TO PREVENT DUPLICATE
346 REM COUNTING. B$ IS REFRESHED BEFORE EACH ROW.
350 FOR I=1 TO 10
355 LET N=0\LET M=0
360 PRINT 'YOUR CHOICE FOR ROW'I
370 INPUT C$(0),C$(1),C$(2),C$(3)
372 FOR J=0 TO 3
374 LET B$(J)=D$(J)
376 NEXT J
380 GOSUB 480
390 IF N<>4 THEN 420 \REM TRY NOT PERFECT YET
400 PRINT 'CONGRATULATIONS! YOU HAVE BROKEN THE CODE IN'I'  ROWS'
410 GO TO 710
420 IF I<10 THEN 460
425 REM IF STILL IMPERFECT AFTER 10 ROWS, THEN REVEAL CODE
430 PRINT 'CORRECT WAS',D$(0),D$(1),D$(2),D$(3)
440 GO TO 710
460 REM
470 NEXT I
480 REM SUBROUTINE TO EVALUATE RESPONSE
490 REM COUNT BLACK FIRST
495 FOR K1=0 TO 3
500 IF C$(K1)<>B$(K1) THEN 510
505 N=N+1
510 NEXT K1
512 REM NOW COUNT WHITE
515 FOR K1=0 TO 3
520 FOR K2=0 TO 3
522 IF K1=K2 THEN 550
525 IF C$(K1)=B$(K1) THEN 550
530 IF C$(K2)=B$(K2) THEN 550
535 IF C$(K1)<>B$(K2) THEN 550
540 LET M=M+1
545 LET B$(K2)='X'\GO TO 555
547 REM 'X' IS DUMMY WRONG VALUE TO PREVENT DUPLICATE COUNT
550 NEXT K2
555 NEXT K1
590 PRINT 'YOU GET'N'BLACK AND'M'WHITE PEGS ON ROW'I
610 RETURN
710 REM SELECT ANOTHER GAME OR STOP
720 PRINT 'ANOTHER GAME -- Y = YES, N = NO ';
730 INPUT R$
740 IF R$='Y' THEN 210
800 END
```

Listing 1: Text of BASIC Mastermind Codemaker. Minor modifications may be required for other versions of BASIC.

```
RUN

MM2     19-FEB-77  BASIC V01B-02

MASTER MIND CODEBREAKER
PLEASE BE PATIENT. SOMETIMES I TAKE A FEW MINUTES ON MY MOVE
WHICH VERSION (1 OR 2) ?1

MY MOVE FOR ROW 1  IS
YELLOW       RED           YELLOW        GREEN
HOW MANY BLACK PEGS  ?1
HOW MANY WHITE PEGS  ?1

MY MOVE FOR ROW 2  IS
RED          RED           RED           YELLOW
HOW MANY BLACK PEGS  ?0
HOW MANY WHITE PEGS  ?1

MY MOVE FOR ROW 3  IS
BLUE         GREEN         YELLOW        BLUE
HOW MANY BLACK PEGS  ?0
HOW MANY WHITE PEGS  ?2

MY MOVE FOR ROW 4  IS
GREEN        YELLOW        GREEN         GREEN
HOW MANY BLACK PEGS  ?1
HOW MANY WHITE PEGS  ?1

MY MOVE FOR ROW 5  IS
YELLOW       BLACK         GREEN         BLACK
HOW MANY BLACK PEGS  ?3

MY MOVE FOR ROW 6  IS
YELLOW       BLACK         GREEN         WHITE
HOW MANY BLACK PEGS  ?4
THANKS FOR THE GAME

ANOTHER GAME  ?N

STOP AT LINE 900

READY
```

Listing 4: Sample run of BASIC Mastermind Codebreaker. Values in response to the queries "HOW MANY BLACK PEGS?" and "HOW MANY WHITE PEGS?" are typed by the player and correspond to correct colors in correct positions and correct colors in wrong positions in program's try.

strategy used by the program is main strength calculation involving little conceptual sophistication, it is nevertheless a powerfully effective strategy. Indeed it might be said that the BASIC Codebreaker program is a Mastermind Master.

After you have played against both of the programs you may want to link the two programs together so they play against each other. It would be interesting to watch the computer's strategy against itself.

Mastermind is marketed in several forms. MiniMastermind is identical to Mastermind except that only six rows are permitted for completing a game. The programs are easily adapted to play any version by altering index variables, subscripts, etc. It is recommended that a playing set be used when playing against the computer, as the game is somewhat difficult to play in your head. However, paper and pencil can be used if no playing set is available. Interesting computer games usually require long and complex programs. Mastermind is a logical and challenging game which can be programmed in a small system with minimum memory and a simple version of BASIC.■

BYTE's Bits

Multiplication

One of the applications of my homebrew 6800 based system requires a large amount of complex number crunching. A noticeable improvement in performance can be realized by reorganizing complex multiplication.

Conventional complex number multiplication of two numbers requires two additions and four multiplications. This can be reduced to three multiplications and five additions. Brute force multiplication yields the product directly as the cross product sum. That is:

$$(a+jb)\ (c+jd) = (ac-bd) + j(ad+cb)$$

However, these products can be expanded by the addition and subtraction of a constant to shorten the computation time.

$$\begin{aligned}(a+jb)\ (c+jd) &= (ac-bd+ad-ad) + j(ad+cb+ac-ac)\\ &= ((a-b)d + (c-d)a + j((a+b)c - a(c-d))\\ &= (y+x) + j(x-z)\end{aligned}$$

$$\text{Where: } \begin{aligned} x &= (a\ b)\ c\\ y &= (a-b)\ d\\ z &= (c-d)\ a\end{aligned}$$

William Jackson
24 Coldstream Dr
Munster Ontario
CANADA K0A 3PO■

Music and Poetry by Computer?

The Association of Computer Machinery (ACM) in cooperation with New York University is presenting a Conference on Computing in the Arts and Humanities to be held in Warren Weaver Hall at New York University, October 21 to 23 1977. The programs include: computer generated video tape and film works; a concert of music composed with the aid of computers; poetry by computer; a computer fashion show; literary analysis using computers; and more.

For registration information, contact Dr Naomi Sagar, Conference Chairperson, NYU Linguistic String Project, 251 Mercer St, New York NY 10012, (212) 598-2294, ext 5.■

An IEEE Hands-on Microcomputer Tutorial

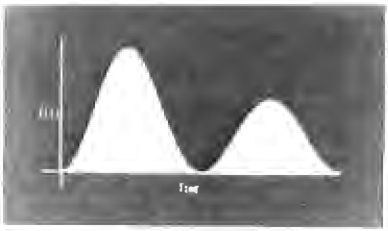
The Nuclear and Plasma Sciences Society of the IEEE, in conjunction with their annual symposium, are sponsoring a hands-on microcomputer applications tutorial. It will be held Saturday, October 22 1977 from 9:00 to 5:00 at the Sheraton Palace Hotel, San Francisco. The tutorial is slanted toward people with computer backgrounds who would like a hands-on experience on actual problems. Suitcase trainers based on the

8080 will be used. Eugene Fisher, Lawrence Livermore Laboratory, and Michael Maples, M and E Associates will head a group of microcomputer systems engineers, teachers and consultants in leading the workshop. Projects include a stopwatch timer display and a set point temperature controller. For further information on registration, write to A J Stripeika, Lawrence Livermore Laboratory, POB 808, Livermore CA 94550. The registration fee is $40 (lunch and coffee included). Material will be sent out for advance study. To obtain maximum benefit from the tutorial it is recommended that attendees study the material in advance. Class will be limited.■

The Microcomputer Connection

The 24th annual IEEE Fall Conference will be held at the Town House Convention Center in Cedar Rapids IA

Raster Scan Graphics



These pictures were created by David Koh MD, Dept of Cardiology RG-20, University of Washington, Seattle WA 98195, using a homebrew raster scan graphics interface based on Thomas Buschbach's article in the November 1976 BYTE, page 32. The circuit has been modifed to be compatible with the Southwest Technical Products bus and its 6800 processor signals. Timing signals are derived from a Southwest Technical Products CT-1024 terminal. Graphics and alphanumeric characters can be displayed either simultaneously or separately.

Resolution is 256 by 160, which

represents 5 K bytes of programmable memory at hexadecimal locations 6000 to 73FF. The cartoons were drawn freehand using a surplus Bolt Baranek and Newman X-Y digitizer purchased from Delta Electronics. This unit has a resolution of 128 by 128. For high resolution work, this region can be moved around under software control anywhere in the display area.■

---

on September 29 and 30 1977. The topic will be "The Microcomputer Connection" and 12 speakers will deliver talks on microcomputer applications in industry, government and the home. Also featured are product and equipment exhibits from a variety of manufacturers. The Fall Conference is open to the public; the fee is $4 for nonmembers, $2 for IEEE members and $1 for students.

For more information, contact Firooz Etemad, IEEE Fall Conference, POB 451, Marion IA 52302.■

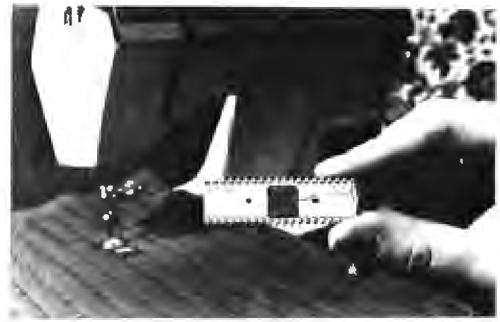
The Midwest Affiliation of Computer Clubs' Convention

Over 2500 people descended on Cleveland's Bond Court Hotel June 10, 11 and 12 to attend the Midwest Affilation of Computer Clubs' annual convention. Manufacturers were in abundance displaying their products. A number of people spoke at the technical sessions, including representatives of Intel and Texas Instruments, both of whom unveiled new products to receptive audiences. An excellent closed circuit video monitor system was also available for those who could not get into the well-attended lectures. All in all, it was a well-run and interesting convention.■

Ahem. AMI Makes a Sewing Machine LSI Control Chip

The new Singer Athena sewing machine features a novel LSI integrated circuit that has 24 different patterns permanently stored in its read only memory. The chip was designed by American Microsystems and has a capacity of 6 K bytes. It also controls the sequences of variable length pulses going to linear servo actuators that control needle and fabric movement.

The computerized appliance is here to stay, and we can look forward to many happy hours of digital darting and byte based basting.■



New Micro Interfacing Workshop

Wintek has added a 2 day "Hands on Interfacing Workshop" to their standard 3 day "Hands on Microprocessor Short Course with Free Take Home Microcomputer." The new interfacing workshop includes analog to digital and digital to analog conversion, signal conditioning, keyboard scanning and decoding, LED display driving, motor position, velocity control and related topics. Tuition is $299. The fall 1977 schedule includes workshops in Dallas, Houston, Washington DC, Melbourne FL, Denver, Palo Alto, San Diego, Indianapolis, Boston, Detroit, Chicago and Puerto Rico. For more information call (317) 742-6802 or write Wintek Corporation, 902 N 9th St, Lafayette IN 47904.■

Programming Tidbit

The normal mode of comparison for equality between two numbers, A and B, is simply the equivalent of the following statement in whatever language is used:

IF A = B THEN . . . .

However, in many contexts this is not necessarily the test to do. Suppose in a pattern recognition program, for instance, you merely want to find out if the two variables A and B are within the right "ballpark" of each other, which is the range DELTA. The semantics of the test computation must then be changed, and the first impulse might be to use the form:

IF (A-B) <DELTA THEN . . . .

However, what happens when B is greater than A and DELTA is a positive number? This would give a true result all the time, independent of the true magnitude of the difference between the two. Since DELTA is a positive number by assumption, we must force the difference being checked to be a positive value also. This is where it is handy to have a language with an absolute value function ABS to force a change of algebraic sign if its argument is negative. Rewriting this comparison gives:

IF ABS (A-B) <DELTA
THEN . . . .

If your interpreter or compiler has no absolute value function, all is not lost, however. You can still check for a differential range by coding the equivalent:

IF ((A-B <0
and (A-B >DELTA)
or ((A-B) >0
and (A-B) <DELTA)
THEN . . . .

Here the terms and and or have been used for logical functions relating different arithmetic expressions and parentheses are used for precedence of operations. Add this thought to your bag of programming tricks and techniques; it might be quite useful.■

BYTE's Bugs

IBM Selectric Interface

Regarding "Interfacing The IBM Selectric Keyboard Printer" by Dan Fylstra, in June 1977 BYTE, page 46, see a letter from Beardsley Ruml on page 32 of this issue, paragraph 4, for comments about:

1. Proper wiring of IBM 73x contacts for the purpose of implementing closed loop feedback. Mr Ruml states that Dan's suggestion of a proposed method for implementing feedback will not work. (Dan's interface was open loop, worst case timing.)
2. The IBM specified current and voltage used with operating the switches should not be ignored, since above the minimum 10 V, 10 mA specification there is enough current to "keep the contacts clean."■

Program Structure

The BASIC Mastermind Codemaker shown in listing 1 requires only 2 K bytes for the source code including remarks. RND and INT (random number and integer) are the only special functions used; no string functions are needed. This program can be easily modified to run under a version of BASIC that does not support string variables by substituting numbers for the color abbreviations. The secret row of colored pegs is set up by a call to the random number generator in line 290. Note that for version 1 games, R=6 and for version 2, R=7 permitting the assignment of empty Spaces. Black peg responses (exact correspondences) are counted from lines 490 to 510, and white peg responses (correct color, wrong position) are counted from lines 512 to 555.

An example of play against the computer program is given in listing 2. The first row of input is a guess. The second row is designed to test whether there are any repeated colors from the first try in the secret code. The codemaker's response tells us that one of the colors in the first try is repeated. In row 3 we hypothesize that BLue is the correct color in row 2 and that Orange is the repeated color from row 1. The codemaker's response constitutes a stroke of luck for us as we learn that BRown is the correct color on row 2 and that Red and Orange can be eliminated from row 1. In row 4 we hypothesize that Green is the correct color in the first row, but alas, we are mistaken. Now only two arrangements are possible. We have a 50% chance of getting it on row 5, but fate is cruel.

After playing the Codebreaker half of the Mastermind game you will learn how to get the most information out of each move. I leave these discoveries to the reader.

After trying to break the codes produced by the computer our next step will of course be to get even with it and see how good it is at breaking codes. The Mastermind Codebreaker program attempts to decode input codes. A word of warning; the Codebreaker program is an excellent player.

Listing 2: Example of play against BASIC Mastermind Codemaker. The text gives an analysis of the moves.

```
RUN

MM         13-JAN-77  BASIC V01B-02

MASTER MIND GAME
WHICH VERSION (1 OR 2) ?1
YOUR CHOICE FOR ROW 1
?R
?O
?G
?Y
YOU GET 1 BLACK AND 1 WHITE PEGS ON ROW 1
YOUR CHOICE FOR ROW 2
?BR
?BL
?BR
?BR
YOU GET 1 BLACK AND 0 WHITE PEGS ON ROW 2
YOUR CHOICE FOR ROW 3
?G
?BL
?O
?O
YOU GET 0 BLACK AND 0 WHITE PEGS ON ROW 3
YOUR CHOICE FOR ROW 4
?Y
?Y
?G
?BR
YOU GET 1 BLACK AND 3 WHITE PEGS ON ROW 4
YOUR CHOICE FOR ROW 5
?Y
?G
?BR
?Y
YOU GET 2 BLACK AND 2 WHITE PEGS ON ROW 5
YOUR CHOICE FOR ROW 6
?G
?Y
?BR
?Y
YOU GET 4 BLACK AND 0 WHITE PEGS ON ROW 6
CONGRATULATIONS! YOU HAVE BROKEN THE CODE IN 6    ROWS
ANOTHER GAME  - Y    YES, N    NO ?N

READY
```

Listing 3: Text of BASIC Mastermind Codebreaker. Minor modifications may be required for other versions of BASIC.

```
MM2      19-FEB-77  BASIC V01B-02

10 REM MASTER MIND 'CODEBREAKER'
20 REM CODED IN RT-11 BASIC
30 RANDOMIZE
40 DIM R$(9,3),S(9,1)
45 DIM A$(6),B$(3),C$(3),D$(3)
50 REM INITIALIZATION
60 FOR J=0 TO 6
70 READ A$(J)
80 NEXT J
90 DATA 'RED','BLUE','GREEN','YELLOW','BLACK','WHITE','SPACE'
100 LET L0=0
110 LET L1=0
120 LET L2=0
130 LET L3=0
140 PRINT 'MASTER MIND CODEBREAKER'
145 PRINT 'PLEASE BE PATIENT, SOMETIMES I TAKE A FEW MINUTES ON MY MOVE'
150 PRINT 'WHICH VERSION (1 OR 2) ';
160 INPUT V
170 LET V=V+5
180 REM ASSIGN COLORS AT RANDOM FOR ROW 1
190 FOR J=0 TO 3
200 LET R$(0,J)=A$(INT(V*RND(J)))
210 NEXT J
220 REM START MAIN PLAY OF GAME HERE
230 REM I IS THE ROW COUNTER
240 FOR I=0 TO 9
245 PRINT
250 PRINT 'MY MOVE FOR ROW'I+1' IS'
260 PRINT R$(I,0),R$(I,1),R$(I,2),R$(I,3)
```

Listing 3, continued:

```
270 PRINT 'HOW MANY BLACK PEGS  ';
280 INPUT S(I,0)
290 IF S(I,0)<>4 THEN 320
300 PRINT 'THANKS FOR THE GAME'
305 PRINT
310 GO TO 870
320 IF S(I,0)<>3 THEN 360
330 LET S(I,1)=0\REM IF 3 BLACKS THEN 0 WHITES
340 GO TO 380
360 PRINT 'HOW MANY WHITE PEGS  ';
370 INPUT S(I,1)
380 REM GENERATE HYPOTHESIS
390 FOR I0=L0 TO V-1
400 FOR I1=L1 TO V-1
410 FOR I2=L2 TO V-1
420 FOR I3=L3 TO V-1
430 LET D$(0)=A$(I0)
440 LET D$(1)=A$(I1)
450 LET D$(2)=A$(I2)
460 LET D$(3)=A$(I3)
470 REM CHECK ALL ROWS FROM FIRST TO CURRENT FOR CONSISTENCY
490 FOR R=0 TO I
500 FOR J=0 TO 3
510 LET C$(J)=R$(R,J)
520 LET B$(J)=D$(J)
530 NEXT J
540 REM USE ROW EVALUATION SUBROUTINE TO CHECK CONSISTENCY OF
550 REM HYPOTHESIS AGAINST EACH ROW
555 LET N=0\LET M=0
560 GOSUB 910
570 REM CHECK FOR AGREEMENT OF BLACK & WHITE COUNT
580 IF N<>S(R,0) THEN 700
590 IF M<>S(R,1) THEN 700
600 NEXT R
610 REM MAKE SURE THAT HYPOTHESIS ROW DOESNT DUPLICATE ROW 1
620 LET Z=0
630 FOR J=0 TO 3
640 IF R$(0,J)<>D$(J) THEN 660
650 LET Z=Z+1
660 NEXT J
670 IF Z=4 THEN 700
690 GO TO 820
700 NEXT I3
710 NEXT I2
720 NEXT I1
730 NEXT I0
740 PRINT 'I HAVE REACHED AN IMPASSE IN MY THINKING'
750 PRINT 'COULD YOU HAVE MADE AN ERROR?'
760 GO TO 870
770 LET L0=I0
780 LET L1=I1
790 LET L2=I2
800 LET L3=I3+1
810 REM DO NOT RECHECK ELIMINATED POSSIBILITIES
820 REM ASSIGN NEXT ROW
830 FOR J=0 TO 3
840 LET R$(I+1,J)=D$(J)
845 NEXT J
850 NEXT I
860 PRINT 'I AM STUMPED -- YOU WIN'
870 PRINT 'ANOTHER GAME  ';
880 INPUT R$
890 IF R$='Y' THEN 150
900 STOP
910 REM SUBROUTINE TO EVALUATE RESPONSE
920 REM COUNT BLACKS FIRST
930 FOR J1=0 TO 3
940 IF C$(J1)<>B$(J1) THEN 960
950 LET N=N+1
960 NEXT J1
970 REM NOW COUNT WHITES
980 FOR J1=0 TO 3
990 FOR J2=0 TO 3
1000 IF J1=J2 THEN 1080
1010 IF C$(J1)=B$(J1) THEN 1080
1020 IF C$(J2)=B$(J2) THEN 1080
1030 IF C$(J1)<>B$(J2) THEN 1080
1040 LET M=M+1
1050 LET B$(J2)='X'\REM DUMMY WRONG VALUE
1070 GO TO 1090
1080 NEXT J2
1090 NEXT J1
1100 RETURN
1110 STOP
2000 END

READY
```

Strategy

Two simple rules determine the Codebreaker program's strategy: row 1 is a random try (all possible arrangements are equally probable); each subsequent row is an arrangement which cannot be disproven on the basis of previous results. When playing Mastermind the human player may try an arrangement which is known to be incorrect in order to obtain specific information; this kind of strategy is not used by the BASIC program.

Program Structure

The BASIC Mastermind Codebreaker shown in listing 3 requires less than 3 K bytes for the source code, including remarks. RND and INT are the only special functions used. Play begins with a call to the random number generator for row 1 at line 200.

After the program types a row it requests feedback information about the try. The number of correct colors in correct positions is requested at line 270. If there are two or fewer black pegs for the try, the program asks for the number of correct colors in incorrect positions at line 360.

After receiving this information the program constructs a table of all possible arrangements, one row at a time (lines 390 to 730). Each row is internally hypothesized to be the hidden code and the question is asked: "How many black pegs and how many white pegs would have been awarded to each try from the first to the current row if the hypothesis were true?" These values are calculated internally by the subroutine at lines 910 to 1100. This is the same subroutine used by the Codemaker program. If there is a discrepancy between the calculated value and the actual value awarded for any row, the current hypothesis is rejected and the next row in the table is constructed. The variables L0, L1, L2 and L3 mark the program's place in the table of possible arrangements between tries so that rejected arrangements are not considered again.

Since row 1 is constructed randomly and therefore occupies an unknown position in the table, it is necessary to check each try to ascertain that it does not duplicate the first row. This check is conducted from lines 610 to 670. If the program exhausts ten tries without breaking the code, it prints the message at line 860. I would be interested to hear if this ever happens.

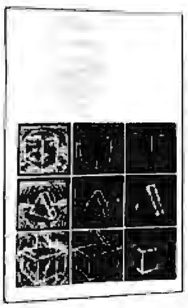
When evaluating the program's try it is necessary to count black and white pegs carefully. If you make a mistake counting the number of exact or inexact correspondences, the program may exhaust the table of all possible arrangements without finding a possibly valid try. In this event, the message at lines 740 to 750 is printed.

Summary

An example of the computer's play is given in listing 4. Note that although the

Mind Massagers from BITS

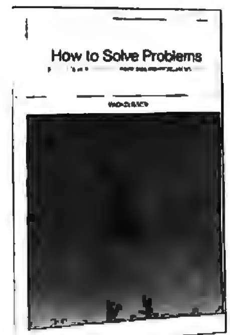
____The Thinking Computer: Mind Inside Matter by Bertram Raphael. Artificial intelligence, or AI, is the branch of computer science concerned with making computers "smarter." It is a growing, vital field that is, unfortunately, the subject of much popular misunderstanding. The Thinking Computer: Mind Inside Matter is a lucid introduction to AI that does much to overcome this misunderstanding. With a minimum of technical jargon, this book discusses the capabilities of modern digital computers and how they are being used in contemporary AI research. It discusses the progress of AI, the goals, and the variety of current approaches to making the computer more intelligent. $6.95.

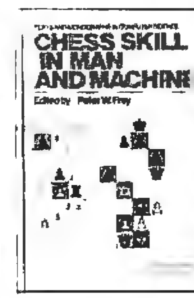


____Projects in Sight, Sound, & Sensation by Mitchell Waite. Dedicated "to all space cowboys." Detailed theory and practice of seven fascinating amateur electronics projects, along with a complete and detailed appendix on how to make PC boards. The projects included in this book are: The Syntheshape, an art generator that can be used to generate innumerable complex and beautiful patterns on the screen of an oscilloscope. An electronic music box that will play over 3000 possible melodies when the lid is lifted. A way to control muscle tension explained in chapter 4. A muscle-wave biofeedback monitor can be used to achieve deep relaxation. The laser-light show transfers light into fascinating patterns in a darkened room. Other projects include a Kirlian camera, a digital ESP machine, and neon-light randomizer. $5.25.

____The Great International Math On Keys Book, from Texas Instruments. Do you need a numerical algorithm for calculating exchange rates in a hotel on a foreign trip? Or figuring those mysterious "points" when facing the sanctimonious charisma of a banker at mortgage time? This book is a compendium of simple explanations and step by step procedures for accomplishing numerical solutions to numerous commonly encountered situations in daily life. Each entry is characterized by a statement of the problem, including the elements of theory required, and the keystrokes needed to solve the problem on an algebraic entry calculator. This is an invaluable sourcebook of information for the person who is inclined to manipulate numbers. $4.95.

____How to Solve Problems by Wayne A Wickelgren. When confronted by a problem, many of us spend more time puzzling over how to solve it than in actually doing so. This book analyzes and systematizes the basic methods of solving mathematical problems. The methods are described in terms of a modern theory derived from research in computer simulation of thinking. Examples illustrating these methods include chess problems, logical puzzles, and railroad switching problems frequently encountered in science and engineering. Whether your interest in solving problems is professional, recreational, or both, you will find this a helpful book. $6.50 softcover.





____Chess Skill in Man and Machine edited by Peter W Frey. This is a most fascinating book, concerning itself with the when, how, and why of computer chess. The when describes past ACM computer chess tournaments, with the details of more than a dozen games. The how consists of the basics of both human chess skill and computer chess theory. It includes a detailed description of the best computer chess program to date (Northwestern University's CHESS 4.5), an end game program called PEASANT, and of various search strategies and heuristic computer chess theory which should enable one to write his own chess program. The book ends with the why concerning the contributions, now and in the future, of computer chess to understanding artificial intelligence, human intelligence, and learning. The only difficulty for the hobbyist's computer chess program is the need for a large computer for the fast processing of search strategies and large core storage for the program and its results. $14.80 hardcover.

Send to:

BITS, Inc
70 Main St
Peterborough NH 03458

Total for all books checked $ ________
Postage, 50 cents per book for ________ books $ ________
Grand Total $ ________

Check payment method:
___My check is enclosed
___Bill my MC No. ____________ Exp. date ______
___Bill my BAC No. ____________ Exp. date ______

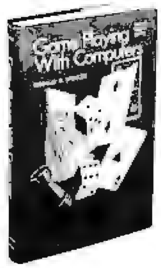
Name ____________
Address ____________
City ______ State ____ Zip ____
Signature ____________

Prices shown subject to change.
You may photocopy this page if you wish to leave your BYTE intact.

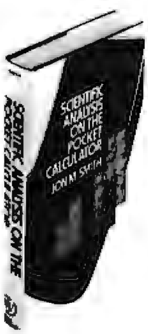
All orders prepaid.
In unusual cases processing may exceed 30 days.

Circle 383 on inquiry card.

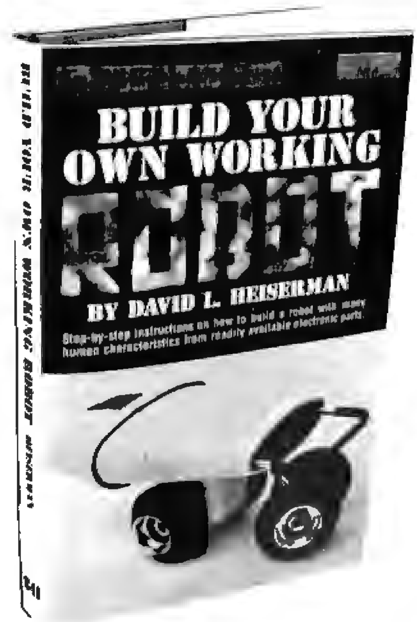
References and Robots from BITS



___Game Playing With Computers by Donald D Spencer, published by Hayden. What does it mean to play games using a computer? Read this book to get an introduction into numerous recreational uses of the computer to program and play mathematical and logical games. Topics include numerous mathematical problems, casino games, board games, unusual gambling games, and miscellaneous logic games. Numerous BASIC language programs and listings are included to show details. $16.95.

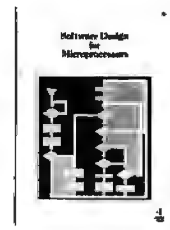


___Scientific Analysis on the Pocket Calculator by Jon M Smith, published by John Wiley & Sons. This book is another in a set of source books for mathematical analysis using the contemporary products of technology. It is oriented to the pocket calculator, yet it will provide you with algorithms and methods useful with any personal computer which implements the scientific and analytical functions found on a good pocket calculator. For a more complete description, see the book review on page 120 of the December 1976 BYTE; or order its 392 pages of detailed technical information and review its use for yourself. $13.75.



___Build Your Own Working Robot by David L Heiserman, published by Tab Books. This book will not tell you how to build Robbie, the robot of Forbidden Planet, or a classical android of science fiction. What it will introduce you to is the problems of making a robot mobile device called Buster III, using pre-microprocessor TTL integrated circuits for all logic functions. It is a must book for background reading, but much of the logic can be extremely simplified using today's microprocessor technology. Use this book as a first look at these problems from which you can build further and more elaborate solutions. Softbound, $5.95.

___A Dictionary of Microcomputing by Philip E Burton. In the opinion of BYTE's editor, Carl Helmers, "This is one of the best *designed* and executed dictionaries of computer related terms yet seen on the market. It is of particular relevance to those individuals who want a good general reference to numerous technical terms, broadly covering hardware and software fields as currently practiced." This new hardbound edition is part of the Garland Reference Library of Science and Technology. $12.50.



___Software Design for Microprocessors. This stand alone guide to microprocessors has been designed by the people at Texas Instruments to convey knowledge to the first time user of microprocessors. This excellent source book of computer concepts begins with an outline of the basic principles of the general purpose computer, its machine architecture, software, and methods of addressing. It proceeds to discuss how to build software, what is involved in documenting what you've done once you've done it, the mechanics of programming, and specific examples using the TI TMS-1000, TMS-8080, TMS-9900 and SBP0400 designs. You'll find a thick hardcover textbook filled with over 370 pages of useful information including a comprehensive glossary of microprocessor terminology, among several other detailed appendices. $12.95.

Send to:

BITS, Inc
70 Main St
Peterborough NH 03458

Total for all books checked	$ ________
Postage, 50 cents per book for ________ books	$ ________
Grand Total	$ ________

Check payment method:

___My check is enclosed

___Bill my MC No. ______________________ Exp. date ________

___Bill my BAC No. ______________________ Exp. date ________

Name ______________________________

Address ______________________________

City ______________ State ______ Zip ______

Signature ______________________________

Prices shown subject to change.

You may photocopy this page if you wish to leave your BYTE intact.

All orders prepaid.

In unusual cases, processing may exceed 30 days.

Clubs and Newsletters

Conducted by
Peter Travisano and
David Wozmak

Amateur Computer Club—Essex ENGLAND

In comparison to most hobbyist groups the English-based Amateur Computer Club is an ancient giant. They are now in their fifth year and are approaching 1000 members. Most of course are located in Great Britain, but many live in other European countries, and some in such remote corners of the world as Singapore and North Carolina.

Communication is via a newsletter which is published six times per year and contains articles on hardware, software and other pertinent technical data. The subscription rate is £2, although four American dollars would be equally acceptable.

A word about the club's name. When it was founded five years ago the only other computer group they had heard of was Stephen Gray's Amateur Computer Society. At that time they had no idea that interest in computers would take off to the extent that local clubs would become viable, so they thus have the distinction of being *the* Amateur Computer Club.

Write Amateur Computer Club at 7 Dordells, Basildon, Essex ENGLAND.

Apple Core Computer Club

According to Apple Core president Jerry Starzinski, AC^3 was formed in May of this year, can boast 14 members, two complete SwTPC 6800 systems, two partial SwTPC 6800s in need of terminals, and various individual members looking at some newer entries like Jupiter, Compucolor and Apple II.

The Apple Core Computer Club holds regular meetings at 7:00 PM on the last Wednesday of the month at J M Perry Technical Institute's Industrial Science Building, 2011 W Washington in Yakima WA. For more information write Jerry P Starzinski, 220 N 2nd St #17, Yakima WA 98901, or call (509) 248-1620 between 9 and 9:30 AM.

A Strange and Clandestine Group

An article by managing editor Joel Miller in Homebrew Computer Club's *Newsletter* has disclosed the existence of a new and potentially dangerous group of underground hackers. At least part of their covert activity centers around the publication of *The Pit.*

The text of Mr Miller's article reads as follows:

> *A few days ago, a mysteriously unsigned envelope was placed on my desk in the Fairchild Publications Department and, to my delight, it contained the last five issues of an informal underground microprocessor newsletter called the* Effate Pit, *"a journal for and by a small underground band of F8 users who can't afford a Formulator." An introductory statement informs us that "the chief users. . .are engineers working for a major semiconductor company," and that "you can't even get on the mailing list except by asking around in the right places." The issues contained a number of very interesting applications-oriented articles by such nefarious authors as "The Mole," "Captain Midnight," "The Scrounger" and "Poto."*
>
> *After a little detective work con-*

Would your club benefit from a write-up in BYTE's Clubs and Newsletters section? It's easy enough to arrange. Just drop a line to BYTE, 70 Main St, Peterborough NH 03458, attn: Clubs and Newsletters, with the information you'd like printed. Naturally, we can't cover every club every month, but we do make a special effort to mention new clubs and those which correspond regularly.

It is important to note that dated material must be submitted three months in advance of the event due to our production schedule.

Those people looking for a more complete listing of computer clubs should refer to January 1977 BYTE.■

tacting my numerous informants, snitches, and ratfinks scattered throughout the company, I finally made contact with The Mole, the newsletter's editor, who—despite my not knowing this week's password—was eager to talk. "Although F8 parts are easy to get here," says The Mole, "a lot of us are starting to fool around with other stuff, like the Z-80. (So we've) solved the problem of newsletter content by changing the name of the rag (to The Pit*) and moving further underground to avoid creditors and others of a similar interfering bent."*

The Pit *has published articles such as "A Twelve IC TV Typewriter," "Parallel Handshake Interface From An ASCII Keyboard," "A Simple-But-Slow Analog Output For The F8," "Dynamic RAM For The F8," "Keypad Data Loader" among others. An especially interesting article by The Mole uses the F8 as a controller for surplus Diablo printer mechanisms that are now available on the hobby market.*

Thanks for the story, Joel.

Should The Mole and his accomplices care to surface, BYTE would of course be happy to use its good offices to spread the word.

And Speaking of Homebrew. . .

Bay area hobbyists who haven't done so already should look into Homebrew, a inventive and well-established organization. Drop a line to Homebrew Computer Club, POB 626, Mountain View CA 94042 and find out more.

Nebraska Clubs

OMAHA, the Omaha Microprocessor Amateur Hobbyist Association, has recently celebrated its first anniversary. Club accomplishments to date include the financing of an Altair 8080, an audio cassette interface and an E & L Instruments MMD1—a micro designed for educational and experimental purposes. The group currently meets twice a month at Northern Natural Gas Company in Omaha.

One of OMAHA's board members, Lt Tom Smith, has formed a club in Bellevue NE called MACH (no explanation of the acronym was given). The club meets the second Tuesday of the month in the Commercial Federal Bldg, Hwy 73-75 and Galvin Rd, Bellevue NE.

Interested persons should contact

OMAHA, c/o Rita Bianchi, S&DP–4th Floor, Northern Natural Gas, 2222 Dodge, Omaha NE 68102, or MACH, c/o Thomas Smith, 2708 Calhoun St, Bellevue NE 68005.

Portland Computer Society

Portland Computer Society has recently been in touch. Meetings are held on third Saturdays at 1:00 PM at Portland Community College. For details write to PCS's new address, 3763 SE Division, Portland OR 97202.

Houston Amateur Microcomputer Club

Obviously there is strong interest in personal computing in the Houston area. One reflection of this is the growth of the Houston Amateur Microcomputer Club. Membership is now over the 100 mark, and according to their newsletter *Nybble,* they are preparing to make a significant contribution both in Houston and nationally. Contact HAMC, c/o Troxel Ballou, 3842 Grennoch, Houston TX 77025, or call (713) 661-6806.

Philadelphia Area Computer Society

Like Houston and countless other clubs PACS has grown dramatically with the personal computing boom. Their newsletter, *The Data Bus,* has doubled in size and improved in quality over the past four months. Club activity centers around the cosponsoring of the annual Trenton Computer Festival, an active group purchase effort and a major club project on 6502 applications.

Meetings are held on the third Saturday of the month at 2:00 PM at the LaSalle College Science Building, 20th and Olney St, Philadelphia; workshops, courses and subgroups meet prior to the club meeting, usually at 12:00.

You can write Philadelphia Area Computer Society at POB 1954, Philadelphia PA 19105, or call (215) 923-3299 or (215) 829-6745.

PET User Group

A user group has been formed for people interested in the Commodore PET 2001 computer. For those unfamiliar with the PET it is a compact computer with integrated keyboard, CRT with character and graphics capability, cassette drive, 14 K ROM operating system including full 8 K BASIC, and 4 K RAM user space.

The purpose of the PET user group will be to share and exchange applications, programs and hardware expansion techniques, and to provide general user feedback. The first year membership is $5 and will include the *User Notes* publication. Contact Gene Beals, POB 371, Montgomeryville PA 18936.

"Name the Users' Group" Contest

A 5 year membership is the prize in a "name the users' group" contest sponsored by computer hobbyists exchanging information on the use of Heathkits. Contestants should of course avoid submitting entries which have been registered as trademarks, since these will not be considered.

Entries and requests for further information should be sent to Charles A Floto, 267 Willow St, Apt 23, New Haven CT 06511.

Southern Nevada

Southern Nevada Personal Computing Society was formed in June of this year and as of this writing (July) was still in the embryonic stage with meeting times and places still unresolved. Membership is open to Clark county residents and nonresident students of Clark County educational institutions. More information should be available by the time this issue of BYTE reaches its readers. To learn more about SNPCS write 1405 Lucilee St, Las Vegas NV 89101, or call (702) 642-0212.

Hex Users

A new Hex Users Club has been announced for systems with 16 to 24 keys, EPA, Motorola Evaluation Kit II, restricted to 6800 based systems, for now. For more information write to Charles C Worstell, 36012 Military Rd S, Auburn WA 98002, enclosing a stamped, self-addressed envelope, or phone (206) 927-6038.

Inland Empire Computer Club—Spokane WA

The Inland Empire Computer Club is a 50 member organization. Meetings are held at 7:30 each fourth Wednesday at the Washington Trust Bank at 3830 E Trent. To find out more about the Inland Empire Computer Club write POB 1434, Spokane WA 99210.

IBM 5100 Users Group

Those interested in contacting an IBM 1500 Users Group are invited to write 5541 Parliament Dr, Suite 104, Virginia Beach VA 23462, or call (804) 490-0154.■

Continued from page 30

The memory management technique of LIL is to be as shown in figure 1.

Objects in a program may encompass the following: real simple variables, integer simple variables, homogeneous arrays (real or integer), strings, numerical constants, string constants, subprograms, functions, parameters, pointers and labels.

Numerical representation has been chosen so as to be as flexible as possible. Integer values are signed magnitude binary; real values have a binary coded decimal (BCD) mantissa with excess 64 binary exponent, a hybrid form which lends itself to the goal of memory conservation.

At present, the only segment of LIL which is actually implemented (on an MCS 6502-based KIM) is the arithmetic package, a mixed mode, variable word length package of great power and flexibility. Integer values may be from one to seven bytes in length; real values may be from two to 14 bytes in length. These word lengths may be curtailed to smaller values as LIL is actually implemented; much depends upon the demands made by language features for space in zero page.

As Mr Stavely suggests, LIL is being taken "one step at a time." However, it has been found that some of the more complex control features planned for LIL make it necessary to look ahead and allow for their implementation without "starting from scratch" with each new feature. LIL is to have control features which lend themselves to the concept of structured programming.

Object code, along with its accompanying reference table, is relocatable to any area of memory. Subprograms and functions may be compiled separately and used with different main programs. LIL has a COMMON statement through which global objects may be linked to a subprogram or function. All linkages in LIL are to take place through descriptor addresses; this implies execution time type checking for all operations. With the exception of labels, any global object may appear in a COMMON statement, including constants. Thus, heavily used objects may be included in the main program's COMMON statement and be repeated in a COMMON statement at the beginning of each subprogram or function. The COMMON statement of LIL is to cost zero additional bytes, no matter how lengthy the statement, with a maximum of 256 common objects.

The concept of which Mr Stavely speaks seems to be ideal for the limited resources of the microcomputer. This is the software simulated compiler, much in the same manner that all arithmetic operations more complex than single byte addition or subtraction must be software simulated.

As Mr Stavely rightly points out, the object code of programs may be shared between owners of different brand microcomputers; all that is necessary is that each person have a loader and interpreter which defines the proper "abstract machine." This is an exciting concept which bodes well for those microcomputer owners who wish to establish a shared program network, and interchange ideas about the abstract machine.

LIL is simply one version of the abstract machine. However, the concept of control bytes in the code seems to be a very general idea which would be useful for any abstract machine. A control byte is to appear as the first item of executable code (discounting a few overhead bytes such as "number of bytes in code"), and control bytes are to appear after each 8 unit subdivision of code, or after each END OF STATEMENT, whichever comes first. The interpreter is to pick up control bytes "on the fly" as it executes the code. A copy of the control byte is rotated left each time the program counter is incremented; thus, by examining bit 7 of the copy of the control byte, the interpreter can determine whether the next unit of code is a 2 byte address or a 1 byte offset. In LIL, END OF STATEMENT is a routine which checks for certain valid conditions, then turns control back over to the program monitor which loads the next control byte and proceeds merrily upon its way. Upon encountering code such as WHILE. . ., the program counter and current state of the control byte are pushed onto the hardware stack for preservation; by later popping this data off the hardware stack and restoring the values of the program counter and control byte, execution is returned to WHILE. . . .

Mr Stavely's article contains a most interesting concept which holds the promise of truly high level programming languages for the microcomputer, as well as portability of code from one brand of machine to another.

Please let us see more such stimulating discussion of software concepts in the pages of BYTE.■

Ask BYTE

WHO HAS THE MOST STORAGE?

I read your April 1977 article, "A Software Controlled 1200 bps Audio Tape Interface," page 40, and found it extremely refreshing to find magazine people really practicing low cost hardware methods with "nice" software. It really surprised me to see a machine state diagram. However, the most interesting comment is that about a 3M tape controller and drive. I am not really interested in working the electronics out from scratch as you are. But, like so many of your readers, I am very much interested in having a 9 track NRZI peripheral for my machine. My main interest is the huge software availability from "big machines." For me, this software is accessible to run on my machine. I certainly would be interested in constructing the interface hardware for a 9 track drive and would like to see some information on that subject. My machine at present is a Z-80 with 128 K of memory. Soon, like now that the price is right, fast fixed point multiply and divide is available and the microprocessor is now a decent minicomputer in performance. The next thing for me logically is multiprocessing when I add the other 128 K bytes.

R Plemich
307 Hatlen
Mt Prospect IL 60056

With a main memory region of that size, you need at least a 9 track drive or a multiplatter hard surface disk. Here is an interesting and informal contest: let each reader who reads this note send in a postcard giving the amount of memory presently integrated (ie: plugged in, checked out and used as a programming resource) in his or her system. Give a short description of the system in use as well as the type of memory technology broken down by regions in address space. Address your cards to BYTE Memory Census, 70 Main St, Peterborough NH 03458.■

10,000,000,000 BITS

Could you tell me when I might expect to be able to purchase a Philips/MCA optical video disk system with write and read capabilities suitable for use with a microcomputer?

Jack M Mann Jr
105 Yosemite Dr
Oxnard CA 93030

Rare circumstance falls in your favor. It happens that I sat next to Dr Jerry S Sullivan of Philips Laboratories on a plane flight from New York City to San Francisco April 19. We were both going to Monterey CA to attend the 1977 IEEE Asilomar workshop on microprocessors. One of the topics of the wide-ranging conversation was the application of video disk technology in a meaningful way to personal computing. The enticement is there: if a writing method could be achieved we'd have a media cost of \$5 to \$10 per disk with perhaps 10 billion (10^{10}) bits. Jerry could not predict an exact date of course, but Philips is indeed working on the concept of a field programmable digital version of the disk. It is conceivable that such a device might be ready within the next five years. The Philips/MCA video disk which is supposed to be on the market sometime this year is a read only device which will only be available programmed with movies and other visual materials. Functionally a digital version of the disk is similar to a fusible link read only memory or an ASR 33 paper tape with rubout: you can only write the information once. But with 10^{10} bits per disk and inexpensive disks, there is plenty of space to allow rewriting of files and updating by overwriting with the logical equivalent of "rubouts." Cost is an open question at this time, of course. . . CH■

TAKING STOCK

I want a small computer. I need it for stock market research. I can wangle the money. I'm satisfied I can learn to program. But how can I get stock market data into my memory? Are there services that provide stock market data for that purpose? Do you buy it on tapes? On disks? Can you tie your own computer into someone else's data book through a telephone connection?

Shouldn't a greenhorn locate his data source first before he can decide what type of computer will be best for him? Why don't I ever see any ads for data sources?

Charles M Larson
Investment Consultant
11513 Havenwood Dr
Whittier CA 90606

It is a good point that one should tie down the data source before assuming one can use it. Without any knowledge of the charges and contractual arrangements, I do observe that the stock brokerage office which is located in our office building in Peterborough uses

telecommunications equipment supplied by GTE Sylvania with a number of different video terminals and printers. It is all pretty standard communications equipment, although the size of the GTE equipment case in the hall (a full height 19 inch rack cabinet) suggests a dedicated minicomputer as the local satellite of a network. Based on this observation, it should in principle be technologically feasible for you to hang onto a stock market wire and accumulate data via an RS-232 port and modem as it is generated day by day. There is also a whole host of stock market analysis firms, many of whom may offer some form of machine readable data as a by-product. Consult the advertisements in the Wall Street Journal *or* Barron's. *If any readers can help by providing specifics in answer to Mr Larson's request, please write so that we can make the data available. Once the source of data is better detailed, it is possible to make some more specific comments.*

As for why you don't see any advertisements for data sources, perhaps it is because none thought of selling the data before. Perhaps your letter will help create a market.■

WHERE DO I START?

I am currently a college student enrolled at a community college. I have several issues of your magazine and have been reading about all the small systems of computers. I am very interested in owning and operating a small computer system except for one thing: how much do I need to know to set one up? Specifically, how much digital electronics, electronic assembly, computer science and programming knowledge do I need? Assuming I need a lot of this type of knowledge, how long might it take me to master it so I can set up my system?

I intend to use my computer for games, mathematical modeling, digital to analog devices and simulation. I would love to set up a system to do all these things, but I don't want to wait until I have a bachelor's degree in computer science to do it. I would greatly appreciate any guidance you could give me on what kind of knowledge or experience I would need in order to set up a workable system.

John Graffio
1094 Quail Creek Rd
Fallbrook CA 92028

The minimum "set up" time for a personal computer, by far, is through purchase of an "appliance" computer which comes ready to use, in the same manner as a stereo receiver comes ready to use. In this category are the latest calculators (such as the brilliant new SR-59 from Texas Instruments with its printer and ROM software attachments), or true general purpose computers such as the Apple-II, Commodore PET 2001, or Noval 760. All of these come complete and assembled. If you want kits, those products range in complexity from simple peripherals to complete systems. Kits naturally take more time to get into operation and debug. Kits such as the Heathkit computer products assume no initial familiarity with electronics, and provide tutorial materials to get the user into operation.■

CONVERSIONS?

As a newcomer to the field of computers, I am just beginning to understand the hardware, but my software problems still exist en masse. While thinking hexadecimal is nice, I have been brought up thinking in a base 10 numeral system (whatever that is?). I know not whether this is an extremely simple and trivial or a vastly complex question, but tell me, how does one get all those rows of 1s and 0s into groups of four BCD bits? I can understand actually counting up to the number with a high frequency counter (eight 7490s in BCD format paralleled by eight 7490s in binary format driven by a 10 MHz clock), some complex software method that applies tables, or even successively subtracting out exponents of 10. But I figure some genius, somewhere, has as algorithm. Can you help me?

Whit Smith
606 Brookwood Ln
Goldsboro NC 27530

Conversion of a binary number to a decimal number was presented in some detail for the 8080 processor in an article entitled "How to Do a Number of Conversions" by James Brown on page 50 of our September 1976 issue of BYTE. There is no magical short cut with most processors: you compare the integer successively with values of 10 raised to various powers (stored in a table of binary numbers) and derive each digit of the converted BCD number from the binary by successive subtraction of the appropriate power of 10 (or division if you have such on your processor). Conversion of the BCD form to an external ASCII representation is easier, and can be done by adding the hexadecimal constant 30 to each 4 bit BCD digit value, giving an 8 bit ASCII value from hexadecimal 30 to 39, which prints as the character values 0 to 9 respectively. Converting BCD inputs or ASCII inputs to binary internal forms is fairly simple as well: for each successive digit entered, multiply the previous entry sum by 10 (add the old value to the sum ten times for a less than optimal brute force technique), then add the new entry (carefully limited to values from 0 to 9) to the previous entry sum giving the new entry sum.■

Circle 445 on inquiry card.

Circle 446 on inquiry card.

Circle 447 on inquiry card.

Circle 448 on inquiry card.

Circle 449 on inquiry card.

Circle 450 on inquiry card.

Circle 451 on inquiry card.

Circle 452 on inquiry card.

Circle 453 on inquiry card.

E-Z-HOOK

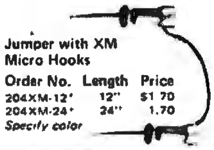
E-Z Hooks have been designed and field tested through the industry to save time and money in commercial electronic production and servicing. The spring-loaded hook attaches firmly, yet so gently it will not damage component — frees hands while testing. Durably constructed and fully insulated to a single contact point assuring true readings. Meets exacting laboratory and space age computer technology requirements. AVAILABLE IN 10 RETMA COLORS: Red, black, blue, green, orange, yellow, white, violet, brown or gray.

MICRO HOOK

XM Micro Hook (1.75" long <1 gram) for difficult IC Testing. Permits hookups to delicate wires where weight and leverage may damage component . . $.80 ea

Specify color. ORDER P/N XM

Jumper with XM Micro Hooks



Order No.	Length	Price
204XM-12*	12"	$1.70
204XM-24*	24"	1.70

Specify color

XM-S MICRO HOOK SET (Includes 1 ea. red, black, blue, green, orange, yellow, white, brown, violet and gray Micro Hook) At this low price you can afford more than one set

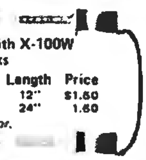
COMPLETE SET (10) MICRO HOOKS $7.95

MINI HOOK

X100W Mini Hook (2.25" long) combines rugged construction, miniature size and Finger-eze Hypo Action for all the best test connections. Hook is large enough for component leads, yet small enough to get into tight places . $.75 ea.

Specify color ORDER P/N X100W

Jumper with X-100W Mini Hooks



Order No.	Length	Price
204-12W	12"	$1.60
204 24W	24"	1.60

Specify color.

Jumper X-100W Mini Hook to Stacking Banana Plug

Order No.	Length	Price
201W	32"	$1.35

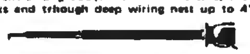
Specify color.

Jumper with X-100W Mini Hook

Order No.	Length	Price
201XM	32"	$1.40

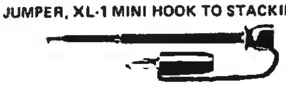
EXTRA LONG MINI HOOK

XL-1 Mini Hook (5.0" long) combines all the proven features of the X100W with an extra long body. It will make safe, short-free test connections in card racks and through deep wiring nest up to 4" ORDER P/N XL-1



Specify color $1.25

JUMPER, XL-1 MINI HOOK TO STACKING BANANA PLUG



Order No.	Length	Price
201XL 1	32"	$1.95

Specify color.

ANCRONA

Our Quality Is Best!

QUITE PORTABLE
VERY AFFORDABLE
AND UNBELIEVABLY
VERSATILE
the PET
computer
may very well
be a lifetime
investment

PET MODEL 2001

PET MODEL 2001 $595.00

OPT. 8K RAM ONLY $795.00

SPECIFICATIONS

DIMENSIONS LWD 18 1/2 x 16 1/2 x 14

WEIGHT 44 lbs

MEMORY

RAM (user) 4K (8K OPT.) exp to 32K
ROM (resident operating system) 14K
8K BASIC interpreter
4K Operating system
1K Diagnostic routine
1K Machine language monitor

VIDEO DISPLAY UNIT

9" black high resolution CRT • 1000 char display 40 col by 25 lines • 8x8 dot matrix for characters and continuous graphics • Auto scrolling from bottom of screen • Winking cursor with full motion control • Reverse field on characters • 64 standard ASCII char 64 graphic char

KEYBOARD

9 1/2" wide x 3" deep 73 keys • All 64 ASCII characters available without shift • Calculator style numeric key pad • All 64 graphic and reverse field characters accessible from keyboard (with shift) Screen Control: Clear and erase • Editing: Character insertion and deletion

CASSETTE STORAGE

Fast redundant recording scheme assuring reliable data recovery • Modified drive for higher reliability of recording and record retention • High noise immunity, error detection and correction • Uses standard audio cassette tapes • Tape files named

OPERATING SYSTEM

Supports multiple languages • Machine language accessibility • File management in operating system • Cursor control, reverse field and graphics under simple BASIC control • Cassette file management from BASIC • True random number generation or pseudo random sequence

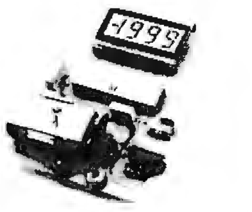
INPUT/OUTPUT

INPUT/OUTPUT

All other I/O supported through IEEE 488 instrument interface which allows for multiple intelligent peripherals • All I/O automatically managed by operating system software • Single character I/O with GET command • Easy screen line-edit capability • Flexible I/O structure allows for BASIC expansion with intelligent peripheral

BASIC INTERPRETER

Expanded 8K BASIC • Upward expansion from current popular BASIC language • Strings, integers and multi dim arrays • High precision (10 significant digits) • Direct memory access through PEEK and POKE commands



BUILD A WORKING DPM IN 1/2 HOUR WITH THESE COMPLETE EVALUATION KITS.

Test these new parts for yourself with Intersil's low-cost prototyping kits, complete with A/D converter and LCD display (for the 7106) or LED display (for the 7107). Kits provide all materials, including PC board, for a functioning panel meter.

ICL7106EV (LCD)	$29.95
ICL7107EV (LED)	$24.95

CAPACITORS

50V CERAMIC DISC $1.00 Per Package

5pf	9/pkg	300pf	7/pkg	.0047mfd	9/pkg
15pf	9/pkg	330pf	7/pkg	.005mfd	9/pkg
25pf	9/pkg	390pf	7/pkg	.01mfd	9/pkg
27pf	9/pkg	470pf	7/pkg	.015mfd	8/pkg
47pf	8/pkg	560pf	7/pkg	.02mfd	8/pkg
68pf	8/pkg	680pf	7/pkg	.022mfd	8/pkg
100pf	8/pkg	.001mfd	9/pkg	.033mfd	8/pkg
150pf	8/pkg	.0015mfd	9/pkg	.039mfd	7/pkg
220pf	8/pkg	.0022mfd	9/pkg	.047mfd	7/pkg
270pf	7/pkg	.003mfd	9/pkg	.1mfd	6/pkg

PLESSEY POLYESTER MINI BOX

MF	V	S	MF	V	S	MF	V	S
.001	1000	.14	.012	630	.14	.15	100	.20
.0012	1000	.14	.015	400	.15	.18	100	.21
.0015	1000	.14	.018	400	.15	.22	100	.23
.0018	1000	.14	.022	400	.15	.27	100	.26
.0022	1000	.14	.027	400	.15	.33	100	.30
.0027	1000	.14	.033	250	.15	.39	100	.33
.0033	1000	.14	.039	250	.15	.47	100	.36
.0039	630	.14	.047	250	.15	.56	100	.44
.0047	630	.14	.056	250	.15	.68	100	.47
.0056	630	.14	.068	250	.15	.82	100	.54
.0068	630	.14	.082	250	.17	1.0	100	.60
.0082	630	.14	.1	100	.17			
.01	630	.14	.12	100	.18			

MATSUO DIPPED TANTALUM

MF	V	$	MF	V	$	MF	V	$
.1	35	.33	2.2	20	.33	10.0	35	.90
.15	35	.33	2.2	35	.40	15.0	20	.45
.22	35	.33	3.3	35	.42	15.0	35	1.32
.33	35	.33	4.7	35	.45	22.0	18	.45
.47	35	.33	6.8	16	.40	33.0	20	1.32
.68	35	.33	6.8	35	.45	47.0	20	1.53
1.0	35	.33	10.0	16	.42	68.0	16	1.62
1.5	35	.40	10.0	25	.45			

1% MICA 500V

51 PF .29	220 PF .41	1500 PF .60
62 PF .29	240 PF .46	2000 PF .90
75 PF .29	300 PF .50	2200 PF .94
82 PF .29	390 PF .50	4700 PF 1.80
100 PF .30	470 PF .50	5100 PF 1.80
120 PF .32	620 PF .52	10000 PF 3.60
150 PF .35	820 PF .53	
180 PF .41	1000 PF .60	

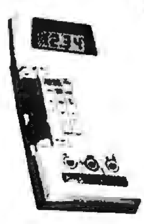
ALUMINUM ELECTROLYTIC (RADIAL LEAD)

MF	V	$	MF	V	$	MF	V	$	MF	V	$	MF	V	$
1	10	.12	10	16	.15	33	25	.22	100	35	.30	330	50	.55
1	16	.13	10	25	.18	33	35	.24	100	50	.38	470	10	.35
1	25	.14	10	30	.22	33	50	.30	220	10	.23	470	16	.40
1	30	.15	10	50	.28	47	10	.17	220	16	.29	470	25	.45
1	50	.18	22	10	.15	47	16	.18	220	25	.35	470	35	.60
5	10	.13	22	16	.18	47	25	.24	220	35	.40	470	50	.80
5	16	.14	22	25	.21	47	35	.29	220	50	.50	1000	25	.65
5	25	.17	22	35	.23	47	50	.32	330	10	.30	1000	35	.80
5	35	.20	22	50	.28	100	10	.18	330	16	.35	1000	50	1.70
5	50	.24	33	10	.19	100	16	.20	330	25	.40	2200	25	1.30
10	10	.14	33	16	.17	100	25	.26	330	35	.45	2200	35	1.60
												2200	50	2.90

NEW from FLUKE - MODEL 8020A

THE DMM FOR THE PROFESSIONAL

- 200 Hr Battery Life
- 26 Ranges for 7 Functions
- 2000 Count Resolution
- High Low Power Ohms
- Autozero and Auto polarity
- MOV-protected to 6000V against hidden transients and overload protection to 300 V AC
- Diode Test Function
- Conductance Function checks leakage resistance to 10,000 meg ohms
- Size: HWL (7.1x3.4x1.8 IN) (18 0x8.6x4.5 cm)
- Weight 13 oz.



ONLY $169.00

2N TRANSISTORS

				2N3646	14	2N4036	50	2N4921	60	2N5306	34	2N5824	25				
				2N3694	22	2N4037	66	2N4922	1 00	2N5308	40	2N5825	35				
2N508A	80	2N2904A	16	2N3702	18	2N4058	25	2N4923	70	2N5309	52	2N5826	36				
2N697	25	2N2905	14	2N3704	18	2N4060	20	2N4983	1 10	2N5321	60	2N5827	30				
2N699	80	2N2905A	16	2N3705	18	2N4061	20	2N4984	1 10	2N5322	70	2N5828	35				
2N706	35	2N2906	20	2N3706	25	2N4093	1 10	2N4987	1 50	2N5323	70	2N5828A	40				
2N711	40	2N2906A	21	2N3707	18	2N4123	20	2N4988	1 20	2N5354	34	2N6000	50				
2N718	40	2N2907	24	2N3708	20	2N4124	20	2N4989	1 60	2N5355	38	2N6001	52				
2N718A	40	2N2907A	28	2N3709	20	2N4125	20	2N4991	.75			2N6002	54				
2N918	40	2N2924	30	2N3710	20	2N4126	24	2N4993	1 80	2N5366	38	2N6008	56				
2N920	26	2N2925	27	2N3711	20	2N4142	18	2N4994	30	2N5367	50	2N6014	66				
2N930	28	2N2926	15	2N3721	20	2N4143	18	2N5086	30	2N5368	30	2N6015	68				
2N1132	35	2N3019	90	2N3724	50	2N4220A	1 00	2N5087	30	2N5369	30	2N6027	50				
2N1613	32	2N3053	30	2N3725	52	2N4224	1 25	2N5088	30	2N5370	30	2N6028	60				
2N1711	34	2N3054	60	2N3731	3 60	2N4228	40	2N5089	40	2N5373	35	2N6034	92				
2N1893	36	2N3055	1 00	2N3740	1 20	2N4235	1 80	2N5109	1 90	2N5375	28	2N6035	1 00				
2N2102	40	2N3376	6 30	2N3741	1 50	2N4237	1 20	2N5131	20	2N5381	30	2N6037	92				
2N2160	1 50	2N3391A	25	2N3771	2 40	2N4238	1 30	2N5133	18	2N5383	30	2N6076	70				
2N2218	26	2N3392	28	2N3772	2 50	2N4248	22	2N5134	20	2N5397	2 50	2N6099	80				
2N2218A	26	2N3393	27	2N3773	3 90	2N4249	18	2N5172	20	2N5418	30	2N6101	1 08				
2N2219	26	2N3414	20	2N3789	3 00	2N4250	24	2N5179	50	2N5419	35	2N6103	1 10				
2N2219A	26	2N3416	27	2N3819	40	2N4256	20	2N5190	80	2N5420	40	2N6121	66				
2N2221	20	2N3417	27	2N3823	70	2N4258	12	2N5191	80	2N5447	30	2N6122	68				
2N2221A	25	2N3440	1 00	2N3859	22	2N4348	2 10	2N5194	80	2N5448	30	2N6123	88				
2N2222	22	2N3442	1 80	2N3860	20	2N4395	1 30	2N5210	15	2N5449	30	2N6124	80				
2N2222A	26	2N3553	1 80	2N3866	1 10	2N4400	25	2N5219	20	2N5450	30	2N6125	68				
2N2368	28	2N3563	15	2N3877A	29	2N4401	25	2N5220	25	2N5451	25	2N6126	88				
2N2369	34	2N3564	16	2N3903	20	2N4402	30	2N5221	20	2N5457	60	2N6218	64				
2N2369A	26	2N3565	18	2N3904	22	2N4403	40	2N5223	20	2N5458	45	2N6219	60				
2N2475	60	2N3567	19	2N3905	20	2N4409	30	2N5225	18	2N5484	60	2N6220	68				
2N2484	30	2N3568	20	2N3906	22	2N4416	90	2N5237	24	2N5490	70	2N6222	30				
2N2605	50	2N3569	19	2N3955	2 70	2N4425	44	2N5232	32	2N5777	60	2N6223	30				
2N2608	2 00	2N3638	18	2N3957	1 50	2N4427	1 00	2N5232A	75	2N5778	55	2N6224	32				
2N2646	70	2N3638A	18	2N3958	1 60	2N4868	1 00	2N5249	40	2N5811	40	2N6225	34				
2N2647	95	2N3642	20	2N3962	35	2N4898	1 10	2N5294	70	2N5817	40						
2N2713	31	2N3643	20	2N3970	1 20	2N4918	60	2N5296	70	2N5818	40						
2N2904	35	2N3645	20	2N4032	80	2N4919	60	2N5298	70	2N5819	45						
										2N5822	58						

NEW from CSC

MAX 100 FREQ COUNTER
- 20 Hz to 100MHz
- 8 Digit Accuracy
- 6" LED Display

The max 100 is accurate enough for laboratory or field service applications and priced low enough for educational or hobbyist use.

Frequency Characteristics Range 20 Hz to 100 MHz (guaranteed) 110 MHz typical Gatetime. 1 sec. providing 1 Hz resolution through out freq range. Accuracy ± 1 count + Time Base error

Input Characteristics Impedance 1 MΩ shunted by 56 pf Connector phone jack Coupling AC Sine Wave Sensitivity 30 mVRMS, 10 Hz 50 MHz, 100 mVRMS 50 MHz to 80 MHz; 300 mVRMS, 80 MHz and above. Maximum Input 200 V Peak, 20 MHz-500Hz, 100V Peak, 500 Hz 1 KHz, 75 V Peak, 1 KHz 10 MHz, 50V Peak, 10 MHz and above.

Internal Time Base Characteristics Frequency. 3 579545MHz crystal oscillator Stability ± 3 ppm @ 25°C. Trimmer Adjustment ± 4 ppm. Temperature Stability Better than 0.2 ppm/°C. 0 to 50°C. Maximum Aging Rate 10 ppm/year.

MAX 100 $134.95

OTHER CSC PRODUCTS

SOCKETS AND BUS STRIPS

Plug in, wire test, modify or expand without patch cords or solder Snap together to form breadboard needed

P/N Description	Term's	Price
QT59S Socket	118	$12.50
QT59B Bus	20	2.50
QT47S Socket	94	10.00
QT47B Bus	16	2.25
QT35S Socket	70	8.50
QT35B Bus	12	2.00
QT18S Socket	36	4.75
QT12S Socket	24	3.75
QT8S Socket	16	3.25
QT7S Socket	14	3.00

PROTO BOARDS AND EXPERIMENTERS

Build and test circuits as fast as you think!

PB 6	$15.95
PB100	19.95
PB101	29.95
PB102	39.95
PB103	59.95
PB104	79.95
EX300	9.95
EX600	10.95

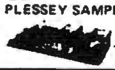
PROTO CLIP

PC14	14-pin	$4.50
PC16	16-pin	4.75
PC24	24-pin	8.50
PC40	40-pin	13.75

JAPANESE TRANSISTORS

2SB405	.85	2SC1973	1.30
2SC536	2/1.20	2SC1974	3.00
2SC828	2/1.50	2SC1975	3.10
2SC829	2/1.50	2SC2034	3.40
2SC929	2/1.40	2SD72	1.00
2SC930	2/1.30	2SD313	1.10
2SC933	1.00	2SD325	1.25
2SC1226A	1.25	3SK49	2.75
2SC1359	.90		

PLESSEY SAMPLER 300 METALIZED CAPACITORS 18 DIFFERENT STD. VALUES $26.00



SAMS BOOKS

TTL COOKBOOK No.21035	$8.95
IC OP AMP COOKBOOK No.20969	$12.95
TV TYPEWRITER COOKBOOK No.21313	$9.95
C-MOS COOKBOOK No.21398	$9.95
TRANSISTOR SUBSTITUTION HANDBOOK, 15th Ed. No 21169	$4.50
TRANSISTOR SPEC MANUAL, 7th Ed. No 21208	$5.95
SEMICONDUCTOR REPLACE GUIDE No.21092	$3.95
BUILDING AND INSTALLING ELECTRONIC INTRUSION ALARMS No.20929	$4.50
UNDERSTANDING IC OP AMPS No.20855	$3.95
HOW TO USE IC LOGIC ELEMENTS, 2nd Ed No.21081	$4.50
UNDERSTANDING CMOS IC's No.21129	$4.95

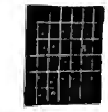
ZENER DIODES

½ WATT	V	PRICE	1 WATT	V	PRICE
IN5223B	2.7	.22	IN4728A	3.3	.44
IN5224B	2.8	.22	IN4729A	3.6	.44
IN5225B	3.0	.22	IN4730A	3.9	.44
IN5226B	3.3	.22	IN4731A	4.3	.44
IN5227B	3.6	.22	IN4732A	4.7	.44
IN5228B	3.9	.22	IN4733A	5.1	.44
IN5229B	4.3	.22	IN4734A	5.6	.44
IN5230B	4.7	.22	IN4735A	6.2	.44
IN5231B	5.1	.22	IN4736A	6.8	.44
IN5232B	5.6	.22	IN4737A	7.5	.44
IN5233B	6.0	.22	IN4738A	8.2	.44
IN5234B	6.2	.22	IN4739A	9.1	.44
IN5235B	6.8	.22	IN4740A	10	.44
IN5236B	7.5	.22	IN4741A	11	.44
IN5237B	8.2	.22	IN4742A	12	.44
IN5238B	8.7	.22	IN4743A	13	.44
IN5239B	9.1	.22	IN4744A	15	.44
IN5240B	10	.22	IN4745A	16	.44
IN5241B	11	.22	IN4746A	18	.44
IN5242B	12	.22	IN4747A	20	.44
IN5243B	13	.22	IN4748A	22	.44
IN5244B	14	.22	IN4749A	24	.44
IN5245B	15	.22	IN4750A	27	.44
IN5246B	16	.22	IN4751A	30	.44
IN5247B	17	.22	IN4752A	33	.44
IN5248B	18	.22	IN4753A	36	.44
IN5249B	19	.22	IN4754A	39	.44
IN5250B	20	.22	IN4755A	43	.44
IN5251B	22	.22	IN4756A	47	.44
IN5252B	24	.22	IN4757A	51	.44
IN5253B	25	.22	IN4758A	56	.44
IN5254B	27	.22	IN4759A	62	.44
IN5255B	28	.22	IN4760A	68	.44
IN5256B	30	.22	IN4761A	75	.44
IN5257B	33	.22			
IN5258B	36	.22			
IN5259B	39	.22			
IN5260B	43	.22			
IN5261B	47	.22			
IN5262B	51	.22			
IN5263B	56	.22			
IN5264B	60	.22			
IN5265B	62	.22			
IN5266B	68	.22			
IN5267B	75	.22			

DIODES

IN914	10/1.00
IN4002	10/1.00
IN4003	10/1.00
IN4004	10/1.00
IN4005	10/1.10
IN4006	10/1.20
IN4007	10/1.20

RESISTOR KITS 5% CARBON FILM RESISTORS



COMPLETE WITH STORAGE BIN

1/4 WATT KIT
42 Different Values (68Ω to 4.7 MΩ)
20 Each Value

1/2 WATT KIT
42 Different Values (68Ω to 4.7 MΩ)
20 Each Value

1/4W — $24.90 PER KIT
1/2W — $25.90 PER KIT

CARBON FILM RESISTORS (5%)

Only in Multiples of 100 pcs per value (ohms)

¼W . . . $1.69 per 100
½W . . . $1.79 per 100

10	100	1.0K	10K	100K	1.0M
11	110	1.1K	11K	110K	1.1M
12	120	1.2K	12K	120K	1.2M
13	130	1.3K	13K	130K	1.3M
15	150	1.5K	15K	150K	1.5M
16	160	1.6K	16K	160K	1.6M
18	180	1.8K	18K	180K	1.8M
20	200	2.0K	20K	200K	2.0M
22	220	2.2K	22K	220K	2.2M
24	240	2.4K	24K	240K	2.4M
27	270	2.7K	27K	270K	2.7M
30	300	3.0K	30K	300K	3.0M
33	330	3.3K	33K	330K	3.3M
36	360	3.6K	36K	360K	3.6M
39	390	3.9K	39K	390K	3.9M
43	430	4.3K	43K	430K	4.3M
47	470	4.7K	47K	470K	4.7M
51	510	5.1K	51K	510K	5.1M
56	560	5.6K	56K	560K	5.6M
62	620	6.2K	62K	620K	6.2M
68	680	6.8K	68K	680K	6.8M
75	750	7.5K	75K	750K	7.5M
82	820	8.2K	82K	820K	8.2M
91	910	9.1K	91K	910K	9.1M

RESISTOR ASSORTMENTS

Each assortment contains 5 pcs each of 10 different values
Values included are shown in (Ohms)
¼W,5% (50 pcs total). . .$1.75 / ½W,5% (50 pcs total). . .$1.85

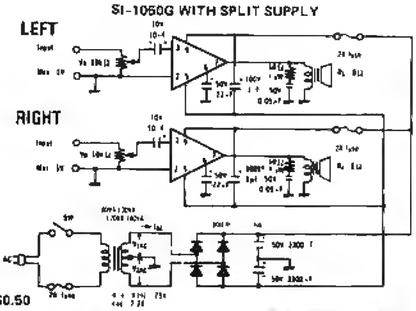
Asst.1:	10, 12, 15, 18, 22, 27, 33, 39, 47, 56 ohm
Asst.2.	68, 82, 100, 120, 150, 180, 220, 270, 330, 390 ohm
Asst.3:	470, 560, 680, 820 ohm, 1K, 1.2K, 1.5K, 1.8K, 2.2K, 2.7K
Asst.4:	3.3K, 3.9K, 4.7K, 5.6K, 6.8K, 8.2K, 10K, 12K, 15K, 18K
Asst.5:	22K, 27K, 33K, 39K, 47K, 56K, 68K, 82K, 100K, 120K
Asst.6:	150K, 180K, 220K, 270K, 330K, 390K, 470K, 560K, 680K, 820K
Asst.7:	1M, 1.2M, 1.5M, 1.8M, 2.2M, 2.7M, 3.3M, 3.9M, 4.7M, 5.6M (½W Assortment No 7 doesn't include 5.6M)
Asst.8R:	(All seven assortments above) ¼W 5% 350 pcs total $10.95 ½W 5% 350 pcs total $11.55

HYBRID AUDIO POWER AMPLIFIERS

SanKen

- Multi-purpose linear amplifiers for comercial and industrial applicacations.
- Less than 0.5% harmonic distortion at full power level.
- 1/2 dB response from 20 to 100,000 Hz.
- Single or split (dual) power supply.
- Rugged, compact and lightweight packages.
- Built-in current limiting for SI-1050G and efficient heat radiating construction.

TYPICAL CONNECTIONS
SI-1050G WITH SPLIT SUPPLY



SANKEN Series SI-1000G amplifiers are self-contained power hybrid amplifiers designed for Hi-Fi, stereo, musical instruments, public address systems and other audio applications. The amplifiers have quasi-complementary class B output. The circuit employs flip-chip transistors with high reliability and passivated chip power transistors with excellent secondary breakdown strength. Built-in current limiting is provided for SI-1050G and all devices can be operated from a single or split power supply

SI-1010G (10W output) . . . $ 6.90
SI-1020G (20W output) . . . $13.95
SI-1050G (50W output) . . . $26.80

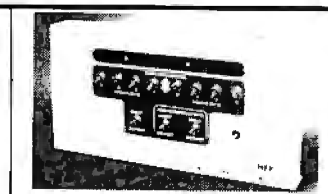
Data Sheet with Application Notes — $0.50

MICROPROCESSOR CRYSTALS

FREQUENCY (MHz)	HOLDER	PART NUMBER	PRICE
1 00	HC33	CY1A	$8.55
2.00	HC33	CY2A	6.75
3.2768	HC33	CY3B	4.80
4.00	HC18	CY3A	4.80
4.434	HC18	CY4C	4.80
5.00	HC18	CY7A	4.80
5.068	HC18	CY5B	4.80
5.7143	HC18	CB5C	4.80
6.00	HC18	CY6B	4.80
10.00	HC18	CY12A	4.35
14 31818	HC18	CY14A	4.35
15 00	HC18	CY15A	4.35
18.00	HC18	CY19A	4.35
20.00	HC18	CY22A	7.20
23.684	HC18	CY23B	4.35
27.00	HC18	CY27A	4 35
32 00	HC18	CY30B	4.35

MICROPROCESSOR & SUPPORT CIRCUITS

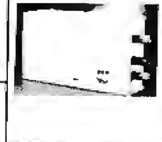
P8080A	12 50	AM8238PC	8.20	MH0026CN	1 00
C8080A	19 50	C8251	14 30	MH0026CH	5 50
C1702A	13 40	C8255	13 50	P2101	2.90
C2708	33 50	P2400	0 10	P2101 1	4.70
P8212	4 10	2517K	4 30	P2102	1 70
P8214	8 95	2524V	4 00	P2102 1	3 45
P8216	3 90	2525V	7 40	P2111	3.30
AM8224PC	6 20	2533V	4 00	P2111 1	3 70
P8228	3 90	N8T26B	3 10	P2112	2 90
AM8228PC	8 70	N8T97	1 60	P2112-2	3 70



MOTOROLA'S EDUCATOR II MICROCOMPUTER HEP KIT

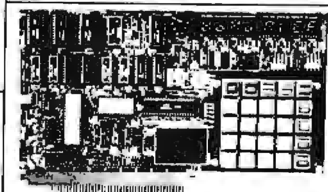
- Motorola M6800 Technology
- Test-as-you build in easy steps normally one evening assembly
- Completely self contained with all parts, cabinet and instructional manual

Get started in the fascinating world of Micro Computers for only $169.95



EDUCATOR II POWER SUPPLY KIT

- 5 V @ 1.0 Amps
- Designed specifically for Educator II Microcomputer Kit
- Complete Kit all parts, cabinet and construction manual $29.95



NEC TK-80 SINGLE BOARD MICROCOMPUTER FOR LEARNING 8080A
$395.00 MICROCOMPUTER SYSTEM

- Single board microcomputer system.
- On board displays and keyboard.
- Powerful monitor program in factory coded ROM.
- Easy system operation.
- CMOS RAM battery back up capability.
- Provisions for audio cassette tape for external storage.
- System expandability.
- Battery operation capability.
- Instructive program examples for learning.
- Informative documentations.

7400N SERIES

7400N	16	7453N	.22	74145N	1 20	74221N	1.10
7401N	20	7454N	.22	74147N	1 80	74247N	1 10
7402N	26	7459N	.25	74148N	1 70	74251N	1 10
7403N	.26	7460N	.22	74150N	1 50	74278N	3 10
7404N	22	7470N	48	74151N	1.10	74279N	1 10
7405N	.23	7472N	44	74152N	1 10	74293N	.90
7406N	.35	7473N	42	74153N	1 10	74298N	1 10
7407N	36	7474N	42	74154N	1 10	74365N	.80
7408N	28	7475N	60	74155N	1.10		
7409N	28	7476N	60	74156N	1 10		
7410N	24	7480N	80	74157N	1 10		
7411N	27	7481N	1.40	74158N	1.30		
7412N	38	7482N	1.40	74160N	1 20		
7413N	48	7483N	1.10	74161N	1 70		
7414N	1 16	7484N	1 20	74162N	1 80		
7416N	42	7485N	1 20	74163N	1.40		
7417N	42	7486N	44	74164N	1.40		
7420N	22	7489N	2.20	74165N	1.60		
7421N	28	7490N	.56	74166N	1.50		
7422N	58	7491N	80	74167N	3.90		
7423N	34	7492N	58	74170N	2.00		
7425N	40	7493N	62	74173N	1.60		
7426N	28	7494N	.88	74174N	90		
7427N	36	7495N	.90	74175N	1.20		
7428N	50	7496N	80	74176N	1 10		
7430N	25	7497N	4 00	74177N	1 10		
7432N	32	74100N	1 20	74178N	1 30		
7433N	44	74104N	.50	74179N	1.70		
7437N	36	74105N	60	74180N	1 10		
7438N	35	74107N	.42	75181N	3 10		
7439N	36	74109N	.56	74182N	1.00		
7440N	.26	74110N	60	74184N	2 70		
7441N	90	74111N	.80	74185N	2 90		
7442N	66	74116N	1.90	74188N	2.90		
7443N	74	74118N	1 90	74190N	1 40		
7444N	1.00	74121N	.48	74191N	1 20		
7445N	1 00	74122N	.48	74192N	1 10		
7446N	1.00	74123N	.80	74193N	1 10		
7447N	88	74125N	.54	74194N	1 10		
7448N	.98	74126N	.54	75195N	1.60		
7450N	22	74128N	.58	74196N	1.40		
7451N	22	74132N	1.40	74197N	1.30		
		74136N	.90	74198N	1.60		
		74141N	1.60	74199N	1 40		
		74143N	4.80	74200N	7 50		

74L'S

74LS00N	.30	74LS90N	.84	74LS258N	1 10
74LS01N	.44	74LS92N	.86	74LS260N	.44
74LS02N	.32	74LS93N	.86	74LS261N	1 90
74LS03N	.32	74LS95N	1 00	74LS266N	.72
74LS04N	.36	74LS96N	1 24	74LS273N	2 66
74LS05N	.33	74LS107N	.46	74LS281N	7 60
74LS08N	.38	74LS109N	.46	74LS283N	2 00
74LS09N	.44	74LS112N	.64	74LS290N	1 78
74LS10N	.32	74LS113N	.64	74LS293N	1.76
74LS11N	.37	74LS114N	.92	74LS299N	6 14
74LS15N	.44	74LS123N	1 10	74LS365N	.70
74LS20N	.32	74LS125N	.60	74LS367N	.70
74LS21N	.44	74LS132N	1 44	74LS374N	3 21
74LS22N	.44	74LS136N	.74	74LS375N	.82
74LS26N	.33	74LS138N	.99	74LS378N	1 60
74LS27N	.44	74LS139N	.99	74LS381N	4 56
74LS28N	.33	74LS145N	1 64	74LS386N	.72
74LS30N	.32	74LS148N	3 45	74LS399N	2 15
74LS32N	.36	74LS151N	.88	74LS670N	3 95
74LS33N	.56	74LS153N	.88		
74LS37N	.60	74LS155N	1 50		
74LS38N	.58	74LS157N	.88		
74LS40N	.44	74LS158N	.99		
74LS42N	.92	74LS160N	1 25		
74LS51N	.44	74LS161N	1 25		
74LS54N	.44	74LS162N	1 25		
74LS55N	.44	74LS163N	1 25		
74LS73N	.46	74LS164N	1.50		
74LS74N	.46	74LS170N	3 20		
74LS75N	.70	74LS174N	.80		
74LS76N	.50	74LS175N	.80		
74LS78N	.92	74LS181N	4.92		
74LS83N	1 38	74LS190N	1 50		
74LS85N	1 40	74LS191N	1 50		
74LS86N	.46	74LS192N	1 20		
		74LS193N	1 21		
		74LS194N	1 32		
		74LS195N	.99		
		74LS196N	2.76		
		74LS197N	2.84		
		74LS221N	1 76		
		74LS241N	3.00		
		74LS251N	1 10		
		74LS253N	1 10		
		74LS257N	.95		

74H

74H76	.42
74H101	.42
74H102	.42
74H103	.47
74H106	.48
74H108	.48
74H01	.32
74H04	.32
74H05	.32
74H08	.34
74H10	.32
74H11	.32
74H20	.32
74H21	.32
74H22	.32
74H30	.32
74H40	.32
74H50	.32
75H51	.32
74H52	.32
74H53	.32
74H54	.32
74H55	.32
74H60	.32
74H61	.32
74H62	.32
74H72	.54
74H73	.42
74H74	1 07

74S

74S00	.40	74S140	.62
74S01	.76	74S151	1 80
74S02	.44	74S153	2 70
74S03	.44	74S154	2 10
74S04	.50	74S157	1.80
74S05	.50	74S158	1 80
74S08	.44	74S160	3.60
74S09	.76	74S161	3.60
74S10	.44	74S172	8.00
74S11	.44	74S174	2.00
74S15	.44	74S175	2.00
74S20	.44	74S178	2 36
74S21	.50	74S179	2 36
74S22	.44	74S181	6 00
74S30	.50	74S182	2 16
74S32	.50	74S189	3 70
74S37	.60	74S194	2 00
74S40	.44	74S195	2 00
74S50	.76	74S196	2 66
74S51	.76	74S197	2 66
74S60	.76	74S251	1.80
74S64	.44	74S253	1.92
74S65	.76	74S257	1 92
74S74	.70	74S258	1 92
74S76	1 16	74S260	.50
74S78	1 10	74S280	4 00
74S85	2.50	74S289	3 60
74S86	.68		
74S112	.88		
74S113	.80		
74S114	2.08		
74S133	.44		
74S134	.80		
74S138	1 76		
74S139	1 80		

74COO

74C00N	.38	74C89N	14.00
74C02N	.38	74C90N	1 36
74C04N	.38	74C93N	1 36
74C08N	.38	74C95N	1 80
74C10N	.38	74C107N	1 90
74C14N	2 20	74C151N	3 82
74C30N	.38	74C154N	5.70
74C32N	.38	74C157N	3 42
74C42N	2 08	74C160N	2 04
74C48N	3 32	74C161N	2 04
74C73N	1 00	74C162N	2 04
74C74N	.86	74C163N	2 04
74C76N	1 00	74C164N	1 80
74C83N	2 74	74C165N	1 80
74C85N	2 74	74C173N	1 60
74C86N	.98	74C174N	1 66
		74C175N	1.66
		74C192N	2 04
		74C193N	3 74
		74C221N	2 04
		74C901N	1.00
		74C902N	1 00
		74C903N	1.00
		74C904N	1 00
		74C906N	1 00
		74C907N	1.00
		74C914N	2.20
		74C922N	5.70
		74C925N	12.00
		74C926N	12.00
		80C95N	1 16
		80C97N	1 00

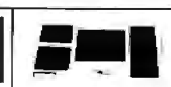
C MOS

4000AL	[illegible]	4016AE	.80
4001AL	.26	4017AE	1 20
4002AL	.26	4018AE	.90
4004AE	1 34	4019AE	.64
4006AL	1 30	4020AE	1 40
4007AE	.24	4021AE	1 40
4008AE	1 70	4022AE	1.20
4009AL	60	4023AE	.30
4010AF	50	4024AE	.90
4011AF	24	4025AE	.30
4012AF	28	4026AE	2.50
4013AE	27	4027AE	.60
4014AE	1 44	4028AE	1 50
4015AE	1 20	4029AE	1 60
		4030AE	.48
		4033AE	2 00
		4035AE	1 00
		4040AE	1 50
		4042AE	1 00
		4043AE	.90
		4044AE	.90
		4046AE	2 00
		4047AE	3 50
		4049AE	[illegible]
		4050AE	[illegible]
		4051AE	[illegible]
		4052AE	1 50
		4053AE	1 50
		4055AE	1 50
		4056AE	[illegible]
		4060AE	1 10
		4066AE	.60
		4068AE	.44
		4069AE	.44
		4070AE	.60
		4071AE	.45
		4072AE	.45
		4073AE	.45
		4075AE	.45
		4076AE	1 20
		4077AE	.55
		4078AE	.55
		4081AE	.45
		4082AE	.55
		4096AE	1 50
		4502AE	1 25
		4508AE	2 10
		4510AE	1 50
		4511AE	1 50
		4514AE	[illegible]
		4516AE	1 50
		4520AE	1 50
		4528AE	1 56
		4585AE	2 06
		4001AE	60

SIGNETICS 3000 DESIGNERS KIT

[illegible]

ORDER PART NO 3000KT100U $100 00



SIGNETICS 8080A EMULATOR KIT

This $299 Emulator Kit provides all that you need to run your 8080A system at bipolar speeds Includes PC board, all components and manual

$299.00

NEW ANCRONA DISCOUNT POLICY FOR ALL 7400 SERIES TTL IC'S

BUY 10 PER TYPE	5%	DISCOUNT
BUY 25 PER TYPE	10%	DISCOUNT
BUY 100 MIXED (10 PER TYPE)	15%	DISCOUNT
BUY 1000 MIXED (25 PER TYPE)	20%	DISCOUNT

MN6040 C-MOS PLL

a single chip phase locked loop intended for use in a frequency synthesizer in CB transceivers and other communications equipment

$7.90

LINEAR FET INPUT OP AMP

LF355H	1 00
LF355N	1 10
LF355AH	22 50
LF356H	1 10
LF356AH	22.50
LF357H	1.60
LF357N	1.10
LF357AH	25 50

MD4301 CMOS ALARM CIRCUIT IC $2.50

SANKEN HYBRID POWER OP-AMP — up to 50 w. (AC rms) power output — Built-in current limiting 1/2 dB response from DC to 100,000 Hz Terminal for external feedback SI-1050 GS $33.80

WIRE WRAP TOOL WSU-30 $5.95
Strips, wraps and unwraps 30 ga. wire on standard wirewrapping pins. 50' roll wire, specify color red, blue, white or yellow $1.98

WIRE WRAP KIT WK-2 $11.95
Includes WSU 30 tool, 50 ft roll of 30 AWG wire and 50 ft roll of 30 AWG wire and 50 pcs ea of 1", 2", 3" and 4" lengths of prestripped wire. Specify color red, blue white or yellow

LINEAR IC'S

		LM302N	1 90	LM709CH	.75
		LM304H	1 50	LM710CH	.90
LM101AH	3.50	LM305H	1 40	LM710CN	.90
LM101F	5.05	LM305AH	2.30	LM715H	6 00
LM105H	4.90	LM306N	1 50	LM715CH	2.90
LM105F	9.00	LM306H	6.80	LM715CD	8 40
LM106H	9.00	LM307H	.95	LM723CH	1 20
LM107H	3.95	LM307N	.90	LM723CD	1 40
LM108H	4 90	LM308H	1 50	LM723CN	.95
LM109K	8.00	LM308AH	6.30	LM725H	9 00
LM114	3 40	LM308N(8)	1 50	LM726CH	4.80
LM118H	14.50	LM309H	1 90	LM726CD	4 50
LM201H	.98	LM309K	1 90	LM733CH	1 70
LM201AH	3.00	LM310H	3.30	LM733CD	1 90
LM201AN	3.00	LM310D	3 80	LM733CN	1.80
LM202H	7.00	LM311D	4 20	LM741H	1.33
LM205H	6.80	LM311N(8)	1 30	LM741CH	.84
LM206H	12.50	LM311H	1 40	LM741CD	1 00
LM207H	3.50	LM312H	3 50	LM741CN(8)	.50
LM208H	6.00	LM316H	5.20	LM741CN	.50
LM209H	5 00	LM318H	3 60	LM747CH	2.60
LM209K	6.40	LM318N(8)	3.10	LM747CD	2 80
LM210H	5.70	LM319N	3 60	LM747CN	1.40
LM211H	3.45	LM324N	1 40	LM747D	4.80
LM218H	8.50	LM339N	1.40	LM748H	1.30
LM300H	1.50	LM380N	1.50	LM748CH	.90
LM300N	1.50	LM381N	3 30	LM748CN(8)	.84
LM301AH	.90	LM555CN(8)	.90	LM748CN	.80
LM301AN	.90	LM556CN	1 20	LM1458CH	1 50
LM302H	3 40	LM709CN	.75	NE536T	3.30

DIP SOCKETS

28 Pin	90	90	81
36 Pin	1 30	1 25	1 15
40 Pin	1 59	1 45	1 10

LOW PROFILE, SOLDER. TIN

	1-24	25-49	50 UP
8 Pin	16	15	14
14 Pin	19	18	17
16 Pin	21	20	19
18 Pin	28	27	26
20 Pin	34	33	32
22 Pin	36	35	34
24 Pin	37	36	35
28 Pin	44	43	42
36 Pin	59	58	57
40 Pin	62	61	60

STANDARD, SOLDER, TIN

8 Pin	25	23	22
14 Pin	27	25	24
16 Pin	30	27	25
18 Pin	36	32	30
22 Pin	70	65	60
24 Pin	49	45	42

STANDARD, SOLDER, GOLD

8 Pin	30	27	24
14 Pin	35	32	29
16 Pin	38	35	32
18 Pin	52	47	43
22 Pin	70	63	57
24 Pin	70	63	57
28 Pin	1 10	1 00	90
36 Pin	1 75	1 40	1 26
40 Pin	1 75	1 59	1 45

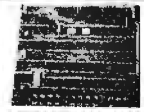
WIREWRAP GOLD (Level No. 3)

10 Pin	45	41	37
14 Pin	39	38	37
16 Pin	43	42	41
18 Pin	75	68	62
24 Pin	1 05	95	85
28 Pin	1 40	1 25	1 10
36 Pin	1 59	1 45	1 30
40 Pin	1 75	1 55	1 40

F8 ASSEMBLED KIT

A fully assembled and tested F8 Microcomputer System. It includes an F8 CPU, SMI, PSU (with Fairbug) and 1K byte RAM. This kit is supplied with F8 literature and a Programming Manual

$185



AM2900 EVALUATION AND LEARNING KIT

Contains all components, PC board and user's manual required to demonstrate the application of the architecture and operation of the AM 2901 and AM 2909

$289.00

AUDIO POWER AMPLIFIERS 5.5 WATT AN315

Designed for mobile radios, tape players, etc. Easy to use, High Gain — 53 dB (Closed Circuit). $3.90 ea.

SPECIAL FAIRCHILD LM741PC

3 for $1.00

TO-220 VOLTAGE REGULATOR FROM FAIRCHILD

7805C	5V	7815C	15V
7806C	6V	7818C	18V
7808C	8V	7824C	24V
7812C	12V		

2 for $1.20

INTERSIL 8038 PRECISION WAVEFORM GENERATOR AND VCO

For simultaneous sine, square and triangular wave forms < 001 Hz to 1 MHz.
PART NO. 8038CCPD $3.90

The KIM 1 Microcomputer System

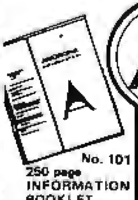
The KIM Family

KIM 1	$245.00	Microcomputer
KIM 2	170.00	4K Static Memory Module
KIM 3	289.00	8K Static Memory Module
KIM 4	110.00	Motherboard
KIM 5	80.00	ROM Expansion
KIMROM 1	150.00	Editor Assembler
KIMROM 2	40.00	KIMATH

KIM Documentation

$9 95*	KIM 1 User Manual
9 95*	Programming Manual
9 95*	Hardware Manual
9 95	KIM Math Subroutines
4.95	Cross Assembler
4.95	Terminal Interface Monitor
9 95	KIM Assembler Manual
9 95	KIM Text Editor

*Included in individual KIM 1 purchase at no charge



No. 101
250 page INFORMATION BOOKLET Only $4.95

ANCRONA

MAIL ORDER ADDRESS — P.O. Box 2208Y, Culver City, CA 90230. Send check or money order, COD, Master Charge and BankAmericard welcomed. Minimum Order: $10.00. Add $1.00 to cover postage and handling. California residents add 6% sales tax. TELEPHONE ORDERS: Call (213) 641-4064.

ANCRONA STORES DO NOT ACCEPT MAIL OR TELEPHONE ORDERS

CALIFORNIA ANCRONA	CALIFORNIA ANCRONA	ARIZONA ANCRONA	OREGON ANCRONA	GEORGIA ANCRONA	TEXAS ANCRONA	CANADA, B.C. ANCRONA
11080 Jefferson Blvd. Culver City, CA 90230 (213) 390-3595	1300 E. Edinger Ave. Santa Ana, CA 92705 (714) 547-8424	4518 E. Broadway Tucson, AZ 85711 (602) 881-2348	1125 N.E. 82nd Ave. Portland, OR 97220 (503) 254-5541	3330 Piedmont Rd., NE Atlanta, GA 30305 (404) 261-7100	2649 Richmond Houston, TX 77098 (713) 529-3489	Vancouver

Circle 454 on inquiry card.

What's New?

Digital Research Offers New Microcomputer Software

CP/M is a floppy disk operating system designed for use with IBM compatible floppy disk computer systems which employ the Intel 8080 processor. Functions of this software package include named dynamic files, program editing, assembly, debugging, batch processing, and instantaneous program loading. CP/M can be adaptable to any 8080 or Z-80 computer system with at least 16 K bytes of main memory and one or two IBM compatible disk drives. Price for a complete CP/M system in object form with documentation is $70; documentation (set of six manuals) alone is $25. For further information, contact Digital Research, POB 579, Pacific Grove CA 93950.■

Circle 597 on inquiry card.

Tychon Introduces New 8080 Editor/Assembler

Tychon's new co-resident editor/assembler, called TEA, is designed for use with 8080 systems and incorporates some useful features. The program resides in 5 K bytes of memory, and can accept either octal or hexadecimal values. The switch from one number base to the other can be made at any time through keyboard commands.

TEA can be relocated in memory using a built-in relocator feature. The price is $35 for a paper tape version of TEA plus the user's manual. The program is also available loaded into 1702A or 2708 programmable read only memory integrated circuits.

For more information, write to Tychon Inc, POB 242, Blacksburg VA 24060.■

Circle 598 on inquiry card.

BLOS and ZAPS

BLOS (rhymes with gloss) and ZAPS are two operating systems announced by Algorithmics for the 8080 and the Z-80 respectively. They are directly loadable on Digital Group systems via cassette. BLOS requires 12 K and ZAPS, 13 K of memory. ZAPS assembles all Z-80 instructions; BLOS assembles only instructions executable by the 8080. BLOS and ZAPS are identical otherwise. Programs written in the 8080 subset are transferable from one system to the other.

The text editor within BLOS and ZAPS is much like those used on timesharing systems. Because it is a full context editor, no line numbers are required. You may edit any of up to 16 files in memory. Only the actual data bytes are stored. Editor commands include input, insert, delete, replace, change, global change, up, down, top, bottom, string search, print and print current line number.

The assemblers feature user-defined commands, switchable number bases, and a full set of cassette operations.

Contact Algorithmics Inc, POB 56, Newton Upper Falls MA 02164.■

Circle 599 on inquiry card.

A Microprocessor Development Kit

Aivex Inc has a new product that should be of interest to those who work with the PDP-11 computer: the MAX-11 conversion kit. The package, which converts a PDP-11 into a microprocessor development system, consists of a cross assembler, simulator and PROM programmer. The cross assembler is said to feature a user defined macro library, local symbols, symbolic cross reference, listing controls and conditional assembly. The simulator features eight break points, single step with trace, external device simulation, interrupt simulation and real time cycle counter. The programmer accepts data from a PDP-11 disk through an RS-232C interface. The kit is presently available in 4040, 8080 and 6800 versions. The price for the simulator plus assembler is $1250; the progammer is $2300.

Contact Aivex Inc, 6 Preston Ct, Bedford MA 01730, (617) 275-2333.

Circle 600 on inquiry card.

A Hardware and Software Catalog

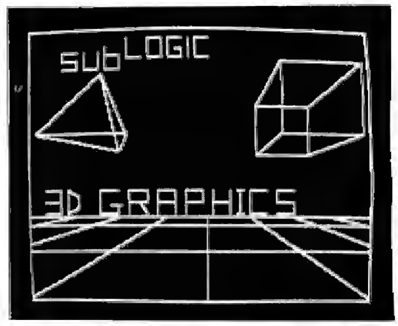
TDL's new catalog features a variety of hardware and software products, including the Z-16 memory module, a 16 K byte static programmable memory card. Maximum access time is 200 ns, and the kit price is $574.

Also described is the XITAN series of mainframes and the Zapple monitor and text editor. A relocating macroassembler is offered along with several BASIC packages and disk software.

For a free copy of the catalog, contact Technical Design Labs Inc, Research Park, Bldg H, 1101 State Rd, Princeton NJ 08540 (609) 921-0321.■

Circle 601 on inquiry card.

An In-Depth Software Package



Sublogic Company of Culver City CA has announced 3D graphics for microcomputers. The Sublogic 3D micrographics package allows the user to view two-dimensional perspective projections of three-dimensional scenes from any location in space. Driving and flying simulations, artistic projections, design projections, engineering analysis and advanced games are some of the applications.

Two versions of the graphics package are offered: a minimal subset BASIC version for general purpose slow speed graphics on any microcomputer system, and a 6800 assembly language version with dynamic graphic capabilities for advanced simulation and complex graphics.

Adaptation instructions, program listings, applications, interface and testing information are supplied with each package. The BASIC version costs $22; the 6800 package is priced slightly higher. Dealer inquiries are welcome. For further information contact Sublogic, POB 3442, Culver City CA 90230.■

Circle 602 on inquiry card.

LSM Engineering's New Text Editor and Operating System

Need a text editor or an operating system for your computer? LSM Engineering, POB 3243, Orange CA 92665, now offers both of these items.

EDIT is a text editor designed to edit source files for BASIC, FORTRAN, assembly language, etc, which utilize the standard 96 character ASCII set. Source files can be manipulated by character, string, line and page to create additions, insertions, replacements and deletions to text.

LSM's Small Operating System, called SOS, has commands for saving, loading and verifying data blocks in conjunction with a Tarbell cassette interface, as well as memory fill, memory block transfer, block verify and memory dump routines. SOS resides in 2 K bytes of read only memory and is supplied with a commented source listing and information to enable the user to patch in up to four new keyboard commands without reassembly.

EDIT is available for $22.50 on Tarbell block form cassette or paper tape, and SOS costs $15 on Tarbell block form cassette.■

Circle 603 on inquiry card.

D.R.C. ELECTRONICS

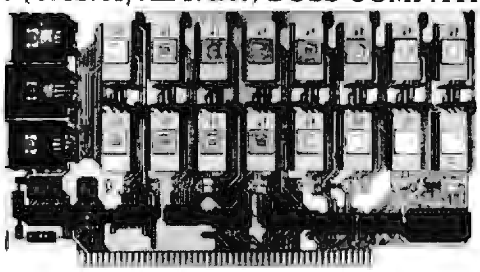
WE PAY POSTAGE!

P. O. Box 401247, Garland, Texas 75040 (214) 271-2461

16K E-PROM CARD

$69.95 (KIT)

S-100 (IMSAI/ALTAIR) BUSS COMPATIBLE



DEALER INQUIRES INVITED

IMAGINE HAVING 16K OF SOFTWARE ON LINE AT ALL TIME!

KIT FEATURES:

1. Double sided PC Board with solder mask and silk screen and Gold plated contact fingers.
2. Selectable wait states.
3. All address lines and data lines buffered!
4. All sockets included.
5. On card regulators.

USES 2708's!

KIT INCLUDES ALL PARTS AND SOCKETS! (EXCEPT 2708's)

ADD $25 FOR ASSEMBLED AND TESTED

SPECIAL OFFER: Our 2708's (650 NS) are $12.95 when purchased with above kit.

2708 1KX8 2708 EPROMS

Prime new units from a major U.S. mfg. 650 N. S. access time. Equivalent to four 1702A's in one package!

GOING INTO BUSINESS SALE!

$15.75 each

8K LOW POWER RAM KIT

(S-100 BUSS COMPATIBLE)

$149.00

(Kit of All Parts and Sockets)

The last word in RAM Kits. Uses 21L02-1 RAM's. All address and data lines fully buffered. PC Board is solder masked and silk screened and has gold plated contacts. Four regulators with heatsinks included.

ADD $30 FOR ASSEMBLED AND TESTED

OPCOA LED READOUT

SLA-1 Common Anode. .33 In. character size. The original high efficiency LED display.

$.75 each
4 FOR $2.50

741C OP AMPS

MINI DIP. Prime New Units. Has computer MFG's house number.

12 FOR $2
100 FOR $15

DISC CAPACITORS

.1 MFD 16 V. P.C. Leads Most Popular Value! P.C. Leads. By Sprague.

20 FOR $1

TANTALUM CAPACITOR

1 MFD. 35 V. Kemet. Axial Lead. Best Value.

10 FOR $1

T. I. ASCII CHARACTER GENERATOR

TMS 4103 JC. 28 PIN CER DIP. Has seven bit COLUMN Output for use with Matrix hard copy devices. With specs. **$3.50**

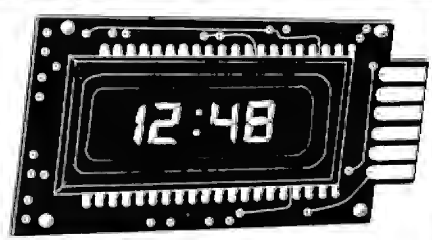
MOTOROLA 7805R VOLTAGE REGULATOR

Same as standard 7805 except 750 MA OUTPUT. TO-220. 5VDC OUTPUT.

$ .44 each 10 FOR $3.95

NATIONAL SEMI. MA1003 CAR CLOCK

Not a kit. Complete tested module. Works on 12 VDC, has on board time base. Sold by others at $24.95. Big .30" Bright Green Digits. Same as used by Detroit in new cars.



$19.95

EDGE CONNECTOR — $1.50

Z - 80 PROGRAMMING MANUAL

By MOSTEK, the major Z - 80 second source. The most detailed explanation ever on the workings of the Z - 80 CPU CHIPS. At least one full page on each of the 158 Z - 80 instructions. A MUST reference manual for any user of the Z - 80. 300 pages. Just off the press! A D.R.C. exclusive! **$12.95**

TRW POWER DARLINGTON

15 AMP 500V NPN. $2 Each

HIGHSPEED SCR

TO-66 5 AMP 250 PIV. By Hutson.

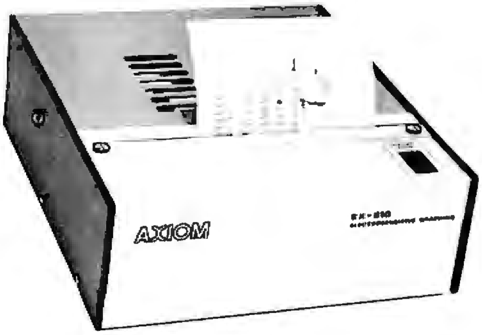
$.69 Each

TERMS: ORDERS UNDER $15 ADD $.75. NO C.O.D. WE ACCEPT VISA AND MASTER CHARGE CARDS. MONEY BACK GUARANTEE ON ALL ITEMS. TEXAS RESIDENTS ADD 5% SALES TAX.

Circle 455 on inquiry card.

ELECTRONICS P. O. BOX 401247 • GARLAND, TEXAS 75040 • (214) 271-2461

A New Plotter for Microcomputers



Axiom Corp has introduced a compact graphic plotter, called the EX-810, which is designed to operate with microcomputers. The unit prints 8,192 dots per second with up to 512 dots per row. It can function as an 80 column alphanumeric printer with a speed of 160 characters per second.

The EX-810 is designed to be driven by an 8 bit microprocessor and is equipped with a TTL compatible controller which takes care of all the internal timing functions necessary to drive the printhead and advance the paper. Printout is initiated by a single input command which causes the eight track non-impact printhead to move across the printing field in approximately 240 ms. Margin and printhead position marker signals are fed back to the microcomputer and may be used to synchronize an eight bit scanning raster to the printhead driver input. The user can vary the horizontal dot resolution by generating an external timing cycle for the input raster.

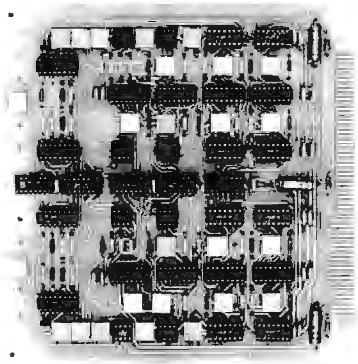
The EX-810 is a self-contained unit with case power supply, paper holder, infrared low paper detector and bell. The dimensions are 9.625 by 4 by 10.875 inches (24.45 by 10.16 by 27.62 cm), and 5 inch (12.7 cm) wide electrosensitive paper is used for printing. Applications include printing forms, tickets, logos, maps, pictures and charts.

The price is $795 in single quantities. OEM discounts are available.

Contact Simon Harrison, vice president of marketing, Axiom Corporation, 5932 San Fernando Rd, Glendale CA 91202. (213) 245-9244.■

Circle 604 on inquiry card.

Programmable Gain Amplifier Card



The Signal Laboratories Inc Programmable Gain Amplifier card (PGA) is a Zilog Z80 MCB compatible accessory offering two channels of computer controlled amplification. Each channel has filtered and unfiltered outputs. Under control of the Z80 MCB, the gain of each channel may vary from unity (0 db) thru 54 db (70 db optional) in 2 db increments, with gain accuracy of 0.1% (0.01% optional). Similarly, the Z80 MCB may select any one of eight user-selectable bandwidths for each channel's filtered output. Unfiltered outputs are 3 db down at 50 KHz. The Programmable Gain Amplifier (PGA) is one of a family of Zilog Z80 MCB compatible accessory cards available from Signal Laboratories Inc, 202 N State College Blvd, Orange CA 92668. Price for quantities under ten is $395, available from stock.■

Circle 605 on inquiry card.

A State(ment) of Flux

There are many sources of information available to the experimenter which teach the art of soldering, but few of these sources give a detailed explanation of the metallurgy and physics involved in the process. M W Dutton Company, Consumer Products Division, 350 Kinsley Av, POB 6205, Providence RI 02940, has bridged the solder gap with a well-written brochure entitled "Dunton's Nokorode Soldering Guide." Topics covered include the composition of solder, solder alloy phase diagrams, mechanical properties of solder and a discussion of solder flux. The brochure is free for the asking.■

Circle 606 on inquiry card.

National's New 16 Bit Microcomputer: the IMP-16L

Anyone looking for an alternative to the many existing 8 bit word microcomputers will be interested in National Semiconductor's new 16 bit microcomputer called the IMP-16L. The unit has a high speed asynchronous bus and a data controller that allows direct memory access (DMA) data transfer rates of up to 16 million bps. It comes complete with 4 K by 16 bits of programmable memory expandable to 64 K by 16 bits. A 20 mA interface is also provided.

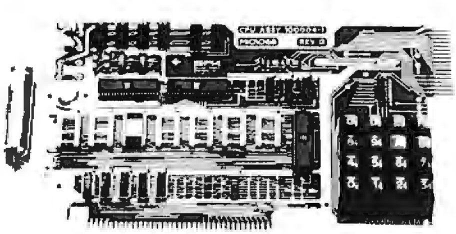
There are 60 general purpose instructions, including multiply, divide, double precision add and subtract, and numerous bit oriented instructions. A control read only memory is also provided in two different versions to speed calculations.

The IMP-16L has four general purpose registers and a 16 word last-in-first-out stack. IO lines include eight general purpose flags, one general interrupt, one vectored interrupt, and four general purpose jump condition inputs. Cycle time is 1.4 μs. Typical register to register addition time is 4.9 μs, and memory to memory addition time is 8.4 μs. One direct memory access transfer takes 1.05 μs.

Software support includes diagnostic routines, a resident assembler, plus a variety of cross-assemblers. The system's architecture permits the installation of up to three central processing units (CPUs) for multiprocessing. The unit costs $825 and up, depending on memory size and options, and is available either from distributors or directly from the manufacturer. For more information, write to National Semiconductor, 2900 Semiconductor Dr, Santa Clara CA 95051.■

Circle 607 on inquiry card.

EPA Micro-68a Computer Now Available in Kit Form



Electronic Product Associates Inc, 1157 Vega St, San Diego CA 92110, has announced the Micro-68 computer kit for $385, which includes power supply and cabinet.

This computer kit comes complete with 16 key hexadecimal keyboard and 6 character hexadecimal display. It uses the Motorola 6800 processor design. Sockets are provided for 768 bytes of volatile programmable memory (128 words supplied with kit). The MON-1 keyboard operating system is supplied in PROM form so that the computer is ready to run upon completion of construction. Commands include inspect and change, load user's program, run user's program, insert break points and save stack. Sixteen bits of IO are provided to the side connector, while the main bus is accessible at the front connector. Full bus buffering provisions are provided on the board. A piggyback Teletype/video display/audio cassette adapter is available.

The Micro-68a is fully compatible with the Micro-68b for later upgrading into a larger system and is available from stock.■

Circle 608 on inquiry card.

Classified Ads

FOR SALE: Assembled SwTPC CT-1024 TVT2 with computer controlled cursor, less power supply, $200. Assembled SwTPC TVT CT-L board, $15. Assembled IMS UCRI cassette interface, $40. Tarbell cassette tapes of 2 K MINOL 2.2 (a Tiny BASIC) or 8 K EMPL 1.0 (a micro APL for the 8080), $10 (user's manual included). Erik Mueller, 36 Homestead Ln, Roosevelt NJ 08555. (609) 448-2605.

FOR SALE: SwTPC CT-1024 with ASCII keyboard, memory board, power supply, and parallel interface for $150. KIM-1 with homemade power supply $200. Mike-2 8008 computer with commercial power supply, 2 K of RAM and ½ K of ROM for $200. Alan Feldman, 1114 Cameron Av, LaCrosse WI 54601. (608) 782-1921

FOR SALE: KIM 1 (MOS) computer board mounted in cabinet with built in power supplies. Excellent condition, $220. Roger Cox, 4327 N Chestnut, #18, Colorado Springs CO 80907.

WANTED TO BUY: Hewlett-Packard 9830A calculator, 4 K word memory preferred. Also, matrix ROM, strings ROM, mass memory ROM, and 9866A or 9871A printer. Consolidated Papers Inc, POB 50, Wisconsin Rapids WI 54494, Attn: Purchasing Dept.

FOR SALE: NATIONAL SC/MP LCDS Development System. Used once. Perfect condition, in original factory carton. Includes CPU card, LCDS, all manuals, $400. M C Devine, 31 Crestview Dr, Westerly RI 02891. (401) 596-7718.

FOR SALE: Several new 12 V stepping motors, with specs for $25. (Haydon part #9904 112 05001). These are normally sold for $49 in single quantities. Applications for stepping motors include robots, printers, XY plotters, NC machines, peristaltic pumps, etc. Mahesh K Shah, 4240 Clarendon, #63, Chicago IL 60613.

WANTED: Contact with persons who have a serious personal interest in using a microcomputer for stock and commodity market investment purposes. If covariance is more than just another word to you, send a brief note of your desires, qualifications and market experience/involvement to J Williams, 2415 Ansdel Ct, Reston VA 22091. An association is being formed.

6500: I have just bought an ECD Corporation MicroMind (based on the 6500 microprocessor family) and would like to get in touch with other people who have recently bought this system to exchange notes, problems, software, etc. I would appreciate anyone who wishes to (and this includes any 6500 family users, too) getting in touch with me to exchange addresses and, hopefully, information. Gregg Williams, 3439 Southern, #7, Memphis TN 38111.

FOR SALE: Five MITS 4 K dynamic memory boards. All in working order and run BASIC. $90 each. 1 ACR, working, $85. One ALTAIR 8800a. Has all power modifications including large capacitor, 14 edge connectors in mother board. $500. Danny L Quinton, POB 23496, Emory University, Atlanta GA 30322.

PROM PROGRAMMING: From Hex or Octal listing: 2708s ($16), 1702As ($5). From Hex/Octal paper tape: 2708s ($8), 1702As ($4). You supply the PROM. 48 hour turnaround. Quantity discount: 1 to 3 PROMs 0%, 4 to 7 PROMs 10%, 8 to 11 PROMs 15%, 12 or more 20%. They can all be different. I also have a few 1702As at $6 each. H S Corbin, 11704 Ibsen Dr, Rockville MD 20852. (301) 881-7571.

FOR SALE: Honeywell keypunch keyboard (reed-switched keys). No enclosure or electronics, $20. Richard Sims, 64 Charlesgate E Apt 75, Boston MA 02215.

FOR SALE: 1 absolutely brand new IMSAI 8080 system, 12 K memory. Also brand new 33 ASR and Panasonic tape recorder. All working and totally unused. Call toll free (800) 221-2628 or write Bernard Rome, 375 Park Av, New York NY 10022.

FOR SALE: Computer Automation 216 mini, complete and working with 8 K core, 16 bit processor, ITY Zoma controller, and miscellaneous electronics interfaced to Dr Suding 1100 baud cassette, software included: assembler, editor, debug, loaders, plus miscellaneous diagnostics. May consider trade up or down. Rich Adamson, 1517 Lucia, Fairmont MN 56031. (507) 235-6321.

FOR SALE: 3M DC300A data cartridges which work in the IBM 5100, TEKRONIX, DEC and similar equipment at a club price of $18 per data cartridge plus the cost of postage. IBM 5100 Users Group, c/o HITS Inc, 5541 Parliament Dr, Suite 104, Virginia Beach VA 23462. (804) 490-0154.

FOR SALE. Computer software to print signs. Print long horizontal signs in large block letters. Output letters are 100 print characters high by 50 print lines wide and are printed with characters of that letter. Complete program source text listing and instructions. Program is written in PL/I and is 517 statements long. Send only 517 cents plus 33 cents for mailing. David Sligar, 7091 Pickway Dr, Cincinnati OH 45238.

WANTED: Copies of book *Computer!* by J A Titus as published circa 1974 by *Radio-Electronics* Magazine. Also any other literature on 8008 microprocessor circuits, printed circuits layouts, etc. A C Acton, POB 31, Midland MI 48640.

WANTED: BYTE issues November 1975 and April, May 1976. Will pay almost any price for well preserved copies. Send price and condition information to Chris Sutter, RR 2 Box 370, Collinsville IL 62234. (618) 344-6552

WANTED: I have a Digital PDP-8/m computer, and would like to buy any or all of the following options: MM8-E 4 K core $200, MM8-E/J 8 K core $400, MP8-E memory parity $200, KL8-E serial interface $150, DK8-EA line clock $100, DK8-EC prog clock $150, KE8-E extended arithmetic unit $150, I am willing to bargain. T J Miles, 3970 Saanich Rd, Victoria BC CANADA V8X-1Y6. (604) 479-1752.

FOR SALE: SwTPC PR-40 alphanumeric printer Unit is assembled, tested, and adjusted. Used nine months without problems. All documentation and four rolls of paper tape provided, $350. Also one TVT1 with two pages of memory, keyboard, UART cassette tape IO, computer control interface. All contained in case. All documentation included, $200. Call or write Gary L Dickman, POB 41, Colby KS 67701 (913) 462-3439 after 6 PM CST.

FOR SALE: BASIC language program of Monopoly. Takes care of all bookkeeping and property detail for this popular board game. Play Monopoly between friends. Make deals and bankrupt your opponents. Runs in 22 K of programmable memory. Written in Data General's 12 K BASIC. Documentation $6, paper tape $12. Charles A Lovell, 4837 Clybourn Av, N Hollywood CA 91601.

FOR SALE: Teletype Model 32 ASR, complete, in good condition, $300 plus shipping. BYTE issues 14, 15, 16 for $3 each. Paul Enright, 2460C Waolani, Honolulu HI 96817.

CRYSTALS

THESE FREQUENCIES ONLY

Part #	Frequency	Case/Style	Price
CY1A	1 000 MHz	HC33 U	$5 95
CY2A	2 000 MHz	HC33 U	$5 95
CY2 01	2 010 MHz	HC33 U	$1 95
CY3A	4 000 MHz	HC18 U	$4 95
CY7A	5 000 MHz	HC18 U	$4 95
CY12A	10 000 MHz	HC18 U	$4 95
CY14A	14 31818 MHz	HC18U	$4 95
CY19A	18 000 MHz	HC18 U	$4 95
CY22A	20 000 MHz	HC18 U	$4 95
CY30B	32 000 MHz	HC18 U	$4 95

XR-2206KB Kit $29.95 **Special** XR-2206KA Kit $19.95

EXAR

WAVEFORM GENERATORS		MISCELLANEOUS		TIMERS	
XR-205	$8 40	XR-2211CP	$6 70	XR-555CP	$ 39
XR-2206CP	4 49	XR-4136	99	XR-320P	1 55
XR-2207CP	3 85	XR-1468	3 85	XR-556CP	1 85
STEREO DECODERS		XR-1488	5.80	XR-2556CP	3 20
XR 1310CP	$3 20	XR-1489	4 80	XR 2240CP	3 25
XR-1310EP	3 20	XR-2208	5 20	**PHASE LOCKED LOOPS**	
XR-1800P	3 20			XR-210	5 20
XR-2567	2 99			XR-215	6 60
				XR-567CP	1 95
				XR 567CT	1 70

CONNECTORS

PRINTED CIRCUIT EDGE-CARD

156 Spacing-Tin-Double Read-Out

Bifurcated Contacts — Fits .054 to .070 P.C. Cards

15/30	PINS (Solder Eyelet)	$1.95
18/36	PINS (Solder Eyelet)	$2.49
22/44	PINS (Solder Eyelet)	$2.95
50/100	PINS (Wire Wrap)	$6.95
50/100A (100 Spacing)	PINS (Wire Wrap)	$6.95

25 PIN-D SUBMINIATURE

DB25P	PLUG	$3.25
DB25S	SOCKET	$4.95

HEAT SINKS

205-CB | 680-.75A

205-CB	Beryllium Copper w-black finish for TO-5	$ .25
291-.36H	Aluminum for TO-220 Transistors & Regulators	$ .25
680-.75A	Black Anodized Aluminum for TO-3	$1.60
401-A	Black Anodized Aluminum — predrilled mounting holes for TO-3 — 4¾ x 1¼ x 2"	$1.75

DIP SWITCHES SPST Slide Action

#206-4 (8 pin dip) 4 switch unit $1.75 ea.
#206-7 (14 pin dip) 7 switch unit $1.95 ea.
#206-8 (16 pin dip) 8 switch unit $2.25 ea.

Etching Kits (cannot be shipped via air)

32 X A-1	P C Etch Materials Kit enough for 5 circuit boards	$29.95 ea.
27 X A-1	Etched Circuit Kit Complete kit — only add water	$ 9.95 ea.
Plugboards		
3662	6 5 X 4 5 X 1/16 Epoxy glass P-Pattern-44 P C Tabs-spaced 156"	$ 6.95 ea.
22/44	Mating connector for plugboard — 22 pin double readout	$ 2.95 ea.
8800V	Universal Microcomputer/Processor plugboard — Epoxy Glass — complete with heatsink and mounting hardware 5 313 x 10 X 1 16 copper clad	$19.95 ea.

Vector

1/16 VECTOR BOARD

	.1" Hole Spacing Part No	P-Pattern L	W	Price 1	2-Up
PHENOLIC	64P44 062XXXP	4 50	6 50	1 72	1 54
	169P44 062XXXP	4 50	17 00	3 66	3 32
EPOXY GLASS	64P44 062	4 50	6 50	2 07	1 88
	84P44 062	4 50	9 50	2 56	2 31
	169P44 062	4 50	17 00	5 04	4 53
	169P84 062	8 50	17 00	9 22	8 26
[illegible]	169P44 062C1	4 50	17 00	6 80	6 12

Vector

SLIT-N-WRAP WIRE WRAP TOOL

- Slits and opens insulation exposing bare wire
- No pre-cutting or pre-stripping.
- Comes complete with two - 100 ft spools #28 AWG wire

Model P180 $24.50

HEXADECIMAL ENCODER 19-KEY PAD

- 1 - 0
- ABCDEF
- Return Key
- Optional Key (Period)
- — Key

$10.95 each

63 KEY KEYBOARD $24.95

This keyboard features 63 unencoded SPST keys unattached to any kind of P C B A very solid molded plastic 13 x 4 base suits most applications

HD0165 16 LINE TO FOUR BIT PARALLEL KEYBOARD ENCODER CHIP $7.95

TOOLS

A97MS	— Diagonal Cutter - 4" semi-flush cut	$8.50 ea
A110MS	— Chain Nose Pliers - 4¾" long	7.50 ea
T-6	— Wire Stripper - #16 to #26 gauge	3.75 ea
55B	— Wire Stripper - #10 to #20 gauge	2.50 ea
CS-8	— Cutter-Crimper Tool - 8¼" long	8.50 ea
Nibbing Tool	— Cuts, Trims or Notches Metal up to #18 gauge	6.95 ea
Nibbing Tool Replacement Punch		3 75 ea

PERMACEL® P-29 PLUS Electrical Tape - All Weather • ¾" wide x 66 ft • Black vinyl

1-9 Rolls $.79 each 10-up Rolls $6.95/10 roll package

MICROPROCESSOR COMPONENTS

8080A	CPU	$19.95	8228	System Controller - Bus Driver	$10.95
8212	8 Bit Input/Output	4.95	MC6800	8 Bit MPU	22.50
8214	Priority Interrupt Control	15.95	MC6820	Periph Interface Adapter	15.00
8216	Bi-Directional Bus Driver	6 95	MC6810AP1	128 x 8 Static RAM	6.00
8224	Clock Generator Driver	10.95	MC6830L8	1024 x 8 Bit ROM	18.00
CDP1802	- with user manual	39.95	Z80	CPU	49.95

CPU'S			RAM'S			
8080	Super 8008	24 95	[illegible]	256 x 1	Static	$ 1 49
8080A	Super 8008	19 95	2101	256 x 4	Static	5 95
2650	8 Bit MPU	26 50	2102	1024 x 1	Static	1 75
			2107 5280	4096 x 1	Dynamic	4 95

SR'S						
2504	1024 Dynamic	$ 3 95	[illegible]	256 x 4	Static	6 95
2518	Hex 32 BIT	7 00	[illegible]489	16 x 4	Static	2 49
2519	Hex 40 BIT	4 00	4101	256 x 4	Static	6 95
2524	512 Dynamic	2 49	8111	256 x 4	Static	8 95
2525	1024 Dynamic	6 00	8599	16 x 4	Static	1 49
2527	Dual 256 Bit	3 95	[illegible]	1024 x 1	Static	2 25
2529	Dual 512 Bit	4 00	1200	256 x 1	Static	6 95
2532	Quad 80 BIT	3 95	93421	256 x 1	Static	2 95
2533	1024 Static	Special 5 95	MM5262	2K x 1	Dynamic	2 for 1 00
3341	Fifo	6 95				
74LS670	16 x 4 Reg	3 95				

UART'S			PROMS			
AY-5-1013	30K Baud	$5 95	[illegible]	[illegible]	Famos	$ 9 95
			[illegible]	[illegible]	[illegible]	[illegible]

ROM'S						
2513(2140)	Char Gen -upper case	$ 9 95	2708	8K	Eprom	29 95
2513(3021)	Char Gen -lower case	9 95	8301 1	1024	Tri State Bipolar	3 49
2516	Char Gen	10 95	8330 1	256	Open Collector Bipolar	2 95
			6331 1	256	Tri-State Bipolar	2 95

SPECIAL REQUESTED ITEMS

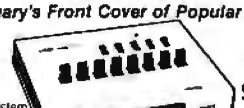
FCM3817	$ 5.00					4N33	3 95
AY-3-8500-1	$13.95	AY-5-2376	14 95	8374	1 95	8T20	7 50
MC3061P	3 50	CD4508	6 75	82S115	25 00	8T97	2 00
MC4016P (74416)	7 50	CD4515	6 50	5841	9 95	3341	6 95
MC14583	3 50	CD4520	2 70	MK5024D	17 50	8369	3 95
MC14562	14 50	MCM6571	17 50	11C90	19 95	MCM-6661	9 95
CD4099	9 95	MCM6574	17 50	DS0026CH	3 75	LD110 LD111	25 00 set
CD4070	55	MCM6575	17 50	TIL306	10 50	AY 5 9100	17 50 ea
MC14409	14 95	MC14419	14 95	ICM7208	22 00	95H90	11 95
MC14410	14 95	ICM7045	24 95	ICM7207	7 50	ICM7209	7.50
						HD0165	7.95

PARATRONICS

ALLOW 1 TO 3 WEEKS DELIVERY

Featured on February's Front Cover of Popular Electronics

Logic Analyzer Kit



MODEL 100A **$229.00/Kit**

- Analyzes any type of digital system
- Checks data rates in excess of 8 million words per second
- Trouble shoot TTL, CMOS, DTL, RTL, Schottky and MOS families
- Displays 16 logic states up to 8 digits wide
- See ones and zeros displayed on your CRT, octal or hexadecimal format
- Tests circuits under actual operating conditions
- Easy to assemble — comes with step-by-step construction manual which includes 80 pages on logic analyzer operation

Some applications are:
— Troubleshooting microprocessor address, instruction, and data flow
— Examine contents of ROMS
— Tracing operation of control logic
— Checking counter and shift register operation
— Monitoring I/O sequences
— Verifying proper system operations during testing

PARATRONICS TRIGGER EXPANDER - Model 10

Adds 16 additional bits Provides digital delay and qualification of input clock and 24-bit trigger word — Connects direct to Model 100A for integrated unit)

Model 10 — $229.00 Baseplate — $9.95

BUGBOOK ®

Continuing Education Series

BUGBOOK I & II - Basic concepts of TTL Logic — over 90 experiments	$17.00/set
BUGBOOK IIa - Introduces UART — recommended for RTTY enthusiast	$5.00/book
BUGBOOK III - Explores 8080 chip — introduces Mark 80 Microcomputer	$15.00/book
555 TIMER APPLICATIONS SOURCEBOOK WITH EXPERIMENTS — over 100 design techniques	$6.95/book
CMOS-M-DESIGNERS PRIMER AND HANDBOOK a complete CMOS instruction manual	$6.00
INSTRUCTORS MANUAL Necessary for instruction on Bugbook I and II.	$3.00

Introductory Offer — all 6 books (worth $49.95) (does not include Instructors Manual) SPECIAL - $42.95

CONTINENTAL SPECIALTIES

PROTO BOARD 6 SPECIAL **$15.95** (6" long X 4" wide)

Other CS Proto Boards

PB100 - 4.5" x 6"	$ 17.95
PB101 - 5.8" x 4.5"	29.95
PB102 - 7" x 4.5"	39.95
PB103 - 9" x 6"	59.95
PB104 - 9.5" x 8"	79 95
PB203 - 9.75 x 6½ x 2¾	75.00
PB203A - 9.75 x 6½ x 2¾ (includes power supply)	120.00

PROTO CLIPS

14 PIN	$4.50
16 PIN	4 75
24 PIN	8.50

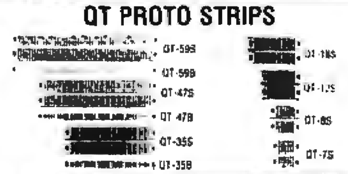
DESIGN MATES

DM1 - Circuit Designer 54.95
DM2 - Function Generator 69.95
DM3 - RC Bridge 59 95

MAX-100 Frequency Counter 20 MHz-100MHz — 8 digit — .6" LED Available September 15 1977 $134.95

Logic Monitor $84.95 for DTL HTL TTL or CMOS Devices

QT PROTO STRIPS



QT type	# holes	price
QT 59S	590	12 50
QT 59B	bus strip	2 50
QT 47S	470	10 00
QT 47B	bus strip	2 25
QT 35S	350	8 50
QT 35B	bus strip	2 00
QT 18S	180	4 75
QT 12S	120	3 75
QT 8S	80	3 25
QT 7S	70	3 00

Experimentor 300	$ 9 95
Experimentor 600	$10 95

$5.00 Minimum Order — U.S. Funds Only
California Residents — Add 6% Sales Tax

Spec Sheets - 25¢ — Send 35¢ Stamp for 1977A Catalog
Dealer Information Available

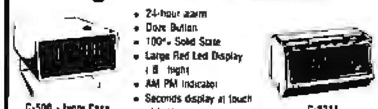
NOW OPEN SATURDAYS

James ELECTRONICS

1021-A HOWARD AVE., SAN CARLOS, CA. 94070
PHONE ORDERS WELCOME — (415) 592-8097
All Advertised Prices Good Thru October

Timeband

Digital Alarm Clocks



- 24-hour alarm
- Doze Button
- 100% Solid State
- Large Red Led Display (.8" high)
- AM PM indicator
- Seconds display at touch of button
- Power failure indicator
- One year factory warranty

C-500 - Ivory Case
C-500B - Ebony Case
$16.95

C-8211 Woodgrain Case **$19.95**

Timeband® by FAIRCHILD — LAMP CLOCK

- 24 hour alarm
- Doze button
- Alarm-on indicator
- .8" high Red LED Display
- AM/PM Indicator
- High Intensity lamp
- Lamp shuts off when collapsed

Model C-590 (Ivory) $29.95

ERC AM/FM 8-Track Stereo Receiver With BSR Changer

- PLL System
- BSR Record Changer
- Slide Controls
- Automatic AFC Control
- 4 Speaker Output
- Walnut finish vinyl covered wood veneer with smoke dust cover

Size: 20"W x 9⅜"H x 15¼"D

Model 6365 **$149.95**

DIGITAL STOPWATCH

- Bright 6 Digit LED Display
- Times to 59 minutes 59 59 seconds
- Crystal Controlled Time Base
- Three Stopwatches in One Times Single Event — Split & Taylor
- Size 4 5" x 2 15" x .90" (4½ ounces)
- Uses 3 Penlite Cells

Kit — $39.95
Assembled — $49.95
Heavy Duty Carry Case $5.95

Stop Watch Chip Only (7205) $19.95

IMC 3½ DIGIT DVM KIT

This 0-2 VDC 05 per cent digital voltmeter features the Motorola 3½ digit DVM chip set It has a .4" LED display and operates from a single +5V power supply The unit is provided complete with an injection molded black plastic case complete with Bezel An optional power supply is available which fits into the same case as the 0-2V DVM allowing 117 VAC operation

A. 0-2V DVM with Case	$49.95
B. 5V Power Supply	$14.95

JE700 CLOCK

The JE700 is a low cost digital clock but is a very high quality unit The unit features a simulated walnut case with dimensions of 6 x 2½ x 3 It utilizes a MAN72 high brightness readout and the MM5314 clock chip

115 VAC

KIT ONLY **$17.95**

Rejected ATARI GAME BOARDS

Over 60 Reusable IC's, misc. transistors, resistors, diodes, caps, crystals, switch, etc. (8½" x 15"). Limited Quantity.

$4.95 ea.

INSTRUMENT/CLOCK CASE

Injection molded unit. Complete with red bezel. 4½" x 4" x 1-9/16"

$3.95 ea.

JE803 PROBE

The Logic Probe is a unit which is for the most part indespensible in trouble shooting logic families TTL DTL RTL CMOS It derives the power it needs to operate directly off of the circuit under test drawing a scant 10 mA max It uses a MAN3 readout to indicate any of the following states by these symbols (HI) 1 (LOW) 0 (PULSE) - P The Probe can detect high frequency pulses to 45 MHz It can't be used at MOS levels or circuit damage will result

$9.95 Per Kit

printed circuit board

T²L 5V 1A Supply

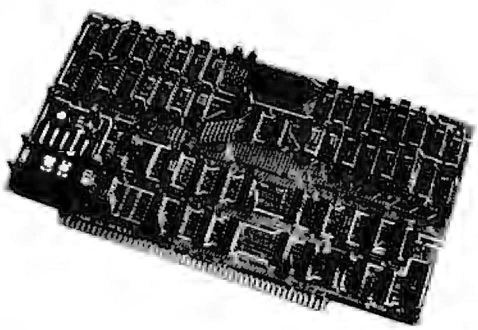
This is a standard TTL power supply using the well known LM309K regulator IC to provide a solid 1 AMP of current at 5 volts We try to make things easy for you by providing everything you need in one package including the hardware for only

$9.95 Per Kit

Circle 459 on inquiry card.

What's New?

IMSAI Offers New Microcomputer with One Megabyte Memory Capacity



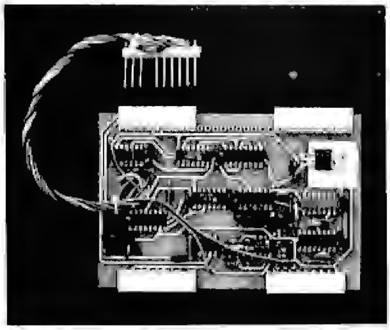
The new IMSAI 8080 Megabyte Micro is one manufacturer's answer to the ever increasing need for more programmable memory space in microcomputers. The unit's chassis has the capacity for one million bytes of programmable memory. The problem of addressing this much memory is solved by using a total of 20 address lines. Memory modules may be purchased in units of 16 K, 32 K or 64 K bytes of dynamic programmable memory.

The heart of the memory control logic is the Intelligent Memory Manager/Interrupt (IMM) board, which provides write protect for each 1 K byte block of programmable memory, read protection, fully vectored interrupts, and time of day clock. Memory expansion is implemented by increasing the number of address lines from sixteen to twenty, and using a form of block switching to control the four added lines. The extended address space is divided into 64 16 K byte blocks of which four may be on-line at any one time. The "switch" is implemented by defining a map which associates a state for address lines 16 through 19 with each state of address lines 12 through 15. The map defaults to 0000 out for all states of address lines 12 through 15 when the system is reset. It is modified and maintained under software control.

For more information, write to Michael Stone, IMSAI Manufacturing Corporation, 14860 Wicks Blvd, San Leandro CA 94577.■

Circle 571 on inquiry card.

Personal Computing Company Introduces ACI-33 Cassette Interface



The ACI-33 is an audio cassette interface designed primarily for the Southwest Technical Products Corporation SwTPC 6800, the control interface and a terminal. The unit will also operate with any RS-232 terminal and computer serial IO which can supply +5 V and ±12 V for the RS-232 interface. When used with the SwTPC, the ACI-33 supports all functions of the control interface.

The ACI-33 uses the self-clocking redundant Manchester scheme of encoding, sometimes called the "Byte Standard." The two logic states used for encoding are represented by 1200 Hz and 2400 Hz frequencies which are written onto and read from the tape.

Priced at $59.95 for an assembled model, the interface is available from Personal Computing Company, 3321 Towerwood Dr, Suite 101, Dallas TX 75234.■

Circle 572 on inquiry card.

MARQUIS, a New Video Display Unit from Dataview

The MARQUIS, announced by Dataview Inc, 23A Dana St, Malden MA 02148, is a new video display unit featuring both 20 mA current loop and RS-232 interfacing capability. The user can select any of 15 different data transmission rates ranging from 50 to 9600 bps. Full or half duplex operation and parity generation are included and are switch selectable. The unit displays 80 upper case characters per line, using a 5 by 7 dot matrix. Characters are taken from a 64 character ASCII set.■

Circle 573 on inquiry card.

A 3 Base Triple Play

Texas Instruments' new TI Programmer calculator could be a boon for the long suffering programmer: it performs arithmetic in octal, hexadecimal and decimal, and can be used to convert automatically from one of these bases to any other.

Typical applications of this calculator include converting memory addresses from hexadecimal to decimal, calculating relative address locations and so on. Expressions may have mixed bases within them, and parentheses may be nested up to four deep. Negative numbers are displayed in two's complement form in both the hexadecimal and octal bases: a one's complement key is also included.

A particularly useful feature on this unit is its ability to perform bit by bit operations on hexadecimal or octal numbers, including AND, OR, exclusive OR and SHIFT operations.

The TI Programmer sells for $49.95 complete with 8 digit LED display, rechargeable battery, AC adapter and carrying case, and is being offered initially by mail only from the company. For more information, write Texas Instruments Inc, Inquiry Answering Service, POB 5012, M/S 84 (Attn: TIP), Dallas TX 75222.■

Circle 574 on inquiry card.

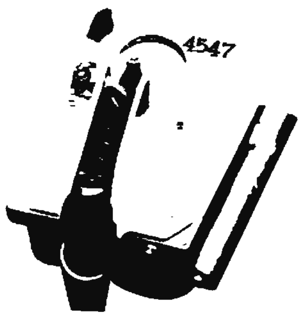
News of Interest to "Computors"

SEI Publications has announced a new monthly management report called the *Small Business Computor*. The term "computor" was coined by the company to describe the man who computes, rather than the machine. The report is to be in newsletter format with no advertising. Its goal is to educate and assist small businessmen and women in the use of computers, computing and data processing as they apply to small businesses.

Sample articles include "Motorcycles and Micros: Retail Success Story," "Control of Computer Based Information Systems," and "Computer Consultants: Who Needs Them?".

More information can be obtained by writing to SEI Publications, POB 145, Newington CT 06111.■

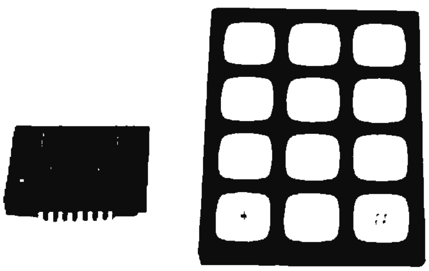
Circle 575 on inquiry card.



PDR-27 GEIGER COUNTER

Just released by the US Navy San Diego. They appear to be in excellent condition and come in fitted aluminum carrying case not shown. Batteries easily obtained except for 1.5 mercury cells which you can substitute with external AA cells, 4 ranges from 0.5 to 500 mr/hr, detects beta & gamma rays. Visual indication and if phonesa are purchased, audio as well. With no facilities to test, we are selling "as is" visually OK, with schematic.

Phones $5.00 Inst. Book $5.00 PDR-27 $35.00



TOUCH TONE ENCODER KIT $12.95

Simply solder the chip to the PC board on back of the touch pad. It's done. Add 9 volt battery, small speaker, and you have touch tone audio output. We provide specs and instructions.

SP-149-B $12.95

IR VIEWER $199.00 COMPLETE

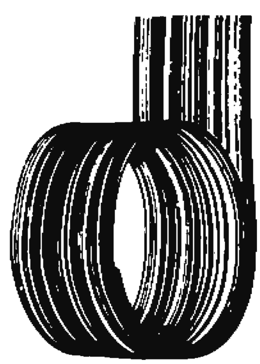
Custom made with manufacturers guarantee. Complete with built-in light source. Permits viewing in total darkness. Operates from 6 volt lantern battery. Great for scientific experimentation, viewing birds, animals, ciminal detection, just plain snooping. (We cannot ship to Calif. residents.

SPL-21 $199.00

SPECTRA FLAT TWIST

50 conductor, 28 gauge, 7 strands/ conductor made by Spectra. Two conductors are paired & twisted and the flat ribbon made up of 25 pairs to give total of 50 conductor. May be peeled off in pairs if desired. Made twisted to cut down on "cross talk." Ideal for sandwiching PC boards allowing flexibility and working on both sides of the boards. Cost originally $13.00/ft

SP-324-A $1.00/ft. 10 ft/$9.00



SP-234-A $1.00 ft 50 cond. 10 ft/$9.00
SP-234-B .90 ft 32 cond. 10 ft/$8.00

WIRE WRAP WIRE

TEFZEL blue #30 Reg. price $13.28/100 ft. Our price 100 ft $2.00; 500 ft $7.50.

MULTI COLORED SPECTRA WIRE

Footage		10'	50'	100'
8 Cond.	#24	$2.50	9.00	15.00
12 "	22	3.00	11.00	18.00
14 "	22	3.50	13.00	21.00
24 "	#24	5.00	20.00	30.00
29 "	22	7.50	28.00	45.00

Great savings as these are about 1/4 book prices. All fresh & new.

TOUCHTONE ENCODER CHIP

Compatible with Bell system, no crystal required. Ideal for repeaters & w/specs. $6.00

CHARACTER GENERATOR CHIP

Memory is 512x5 produces 64 five by seven ASCII characters. New material w/data $6.00

Meshna

Please add shipping cost on above. Minimum order $10

FREE CATALOG SP-9 NOW READY

P.O. Box 62, E. Lynn, Massachusetts 01904

Circle 460 on inquiry card.

A New Altair That Looks Different

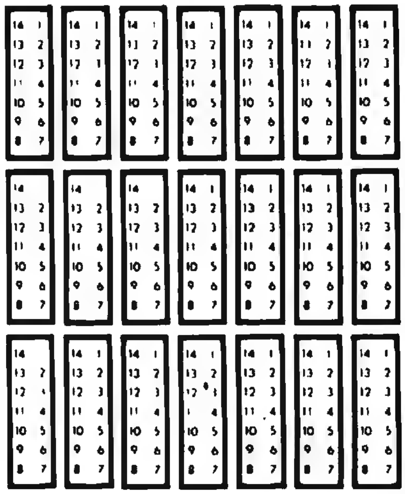
The new Altair Turnkey microcomputer is a significant departure in appearance from the rest of the Altair line. The front panel is virtually blank, sporting only two toggle switches, five LED indicators, and a key operated power switch. All the functional units of this Turnkey model of the 8800b are contained on one circuit board called the Turnkey Module board; this includes the central processing unit (CPU), programmable memory, sense switches and serial IO. All the hardware for this model is compatible with preceding Altair computers.

The Turnkey Module board consists of a serial IO channel that can operate a variety of peripheral devices, plus 1 K byte of programmable memory and provisions for another 1 K bytes of programmable read only memory and logic for the power-on-start feature.

The available software includes a multipurpose bootstrap loader and monitor, both resident in the programmable read only memory. The 8800b is software compatible with the rest of the Altair line. For more information, contact MITS, 2450 Alamo SE, Albuquerque NM 87106.■

Circle 576 on inquiry card.

A New Way to Label Integrated Circuit Pins



Here is an efficient method for labeling dual in line (DIP) integrated circuit pins on the pin side of printed circuit boards: Adtech Inc, POB 10415, Honolulu HI 96816, has announced DIP-1 DIP Pin Out Labels. The labels, which come in a variety of sizes from 8 pin to 40 pin, are printed on an 8½ by 11 inch sheet of clear mylar with adhesive backing. The accompanying reproduction illustrates a portion of the sheet. Each label is imprinted with the appropriate pin out pattern. The user cuts out the desired label and presses it onto the circuit board. Each 8½ by 11 inch sheet costs $5.95, postpaid, in the US. This technique can save a significant amount of time by cutting down on wiring errors and speeding up troubleshooting. We highly recommend this technique for prototyping, having used a crude manual approximation for years to help eliminate wiring errors.■

Circle 577 on inquiry card.

Power for Your Microprocessor

Century Industries, POB 348, Blue Bell PA 19422, has announced a new power supply kit called the Mighty 4000, designed for use with microprocessors. The unit features four regulated and short circuit protected outputs: +5 V at 5.5 A and −5 V at 2.2 A (both fixed output), +5 V at 1.9 A to +12 V at 3.3 A, and +12 V at 0.23 A to +28 V at 0.33 A.

The supply is 10.66 by 4.43 by 3.80 inches (27.07 by 11.25 by 9.65 cm) and weighs 8.5 pounds. The price of the kit is $99; it is also available assembled for $129.■

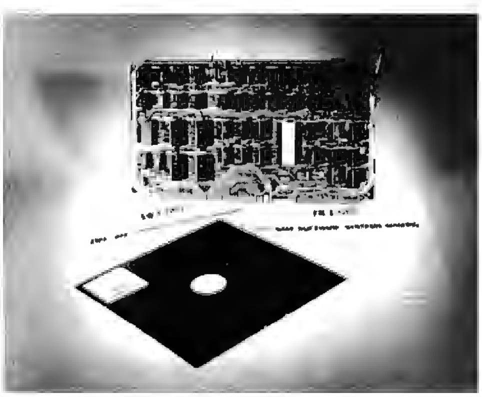
Circle 578 on inquiry card.

New Minicomputer Periodical Offered

Minicomputer Press has announced forthcoming publication of the first of several new periodicals devoted to minicomputer end users in business, industry and the home. *Minicomputer Workbook* is devoted to the interests of engineers, educators and professionals who are familiar with data manipulation and computer concepts, but who need a periodical that integrates the two areas together in a meaningful way. For more information, contact Charles Moore Associates, POB 6, Stump Rd, Southampton PA 18966.■

Circle 579 on inquiry card.

Industrial Users: Upgrade MDS-800 to Develop Z-80 Code



For designers wanting the additional capability of the Zilog Z-80 processor, but who presently have Intel MDS-800 hardware and 8080 software, Relational Memory Systems Inc (RELMS) announces the Z-80 System Adaptor Module, Z-80-SAM. Completely hardware and software compatible with the 8080, 8080A, 8085 and Z-80 microprocessors, the Z-80-SAM enables an MDS-800 to support and develop both 8080 and Zilog Z-80 software. The Z-80-SAM contains a single board and associated software and firmware. The SAM microcomputer board supplants the corresponding MDS-800 board.

Besides the hardware board, Z-80 SAM system monitor programmable read only memory (PROM) firmware replaces corresponding MDS-800 read only memory (ROM) firmware (on monitor ROM board).

The Z-80-SAM extends the useful life of the capital investment 8080 users have in their MDS-800 system while enabling the user to utilize the faster, more efficient Z-80 microprocessor in future hardware/software development.

Enhanced debugging capability is afforded by hardware features such as display LEDs and single cycle switch which allow the user to freeze the MDS-800 data bus, examine the bus contents, and single step the program.

The Z-80-SAM is priced at $1495 in single unit quantities which includes the Z-80 microcomputer board, monitor ROM board, and Z-80 SAM assembler on diskette.

For more information, contact Douglas B Kelley, director of sales and marketing, Relational Memory Systems Inc (RELMS), POB 6719, Santa Clara CA 95150.■

Circle 580 on inquiry card.

Now low-cost memory stacks up in reliability!

Introducing a new generation of ECONORAM* dynamics with SynchroFresh™ reliability

Meet ECONORAM* III with SynchroFresh™, the 8Kx8 dynamic memory for S-100 bus computers that really works. And uses less than half the power of static designs. And costs just $188 for an assembled 8K.

Unlike previous attempts at building a low-cost dynamic memory, ECONORAM* III is entirely reliable ... because of SynchroFresh™, a new approach to memory refresh that is simple, elegant and totally effective.

SynchroFresh™ was invented by George Morrow, designer of the original ECONORAM*. Instead of arbitrarily interrupting your CPU to perform memory refresh cycles, Morrow designed SynchroFresh™ to weave refresh invisibly into the natural timing of the S-100 bus. SynchroFresh™ circuitry simply monitors your computer's machine states, utilizing all of the normal opportunities for memory refresh. It's that simple.

And simplicity means reliability and dramatically lower cost. That's why a SynchroFresh™ design was chosen for the first ECONORAM* dynamic, to follow in the footsteps of the largest-selling static memories for personal computers.

ECONORAM* III with SynchroFresh™ is an 8Kx8 dynamic board, configured as two individually addressable 4K blocks for flexibility. It is available assembled, tested and warranteed for one full year for just $188. This unprecedented warrantee offers a full refund of purchase price if ECONORAM* III does not run reliably with your S-100 CPU—evidence of our confidence in its performance.

It is also available as a kit with complete assembly instructions and documentation for $159.

ECONORAM* III with SynchroFresh™, in assembled or kit form, may be ordered directly from Thinker-Toys™. Write 1201 10th Street, Berkeley CA 94710 or call (415) 527-7548. Call BAC/MC orders toll-free to 800-648-5311. Or ask your computer store to order it for you.

ECONORAM* III
with SynchroFresh™
$188
Assembled, tested & warranteed

A product of Morrow's Micro-Stuff for

Thinker Toys™

*ECONORAM is a trademark of Godbout Electronics.

Circle 461 on inquiry card.

Bally Introduces New Programmable Game Unit

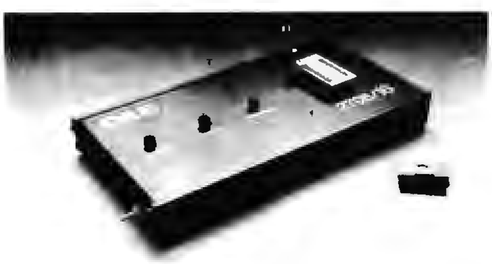
The new Bally Professional Arcade home TV entertainment center is a well-engineered example of the new breed of programmable game modules which use microprocessors for logic and control functions.

The unit features a printing keyboard calculator and can generate video games in color. The games are stored on cassettes; users can change games when desired by simply plugging a different cassette into the front of the unit. A large variety of games is available, including Checkmate, Dodgem, Seawolf, tennis, elementary math drills and poker; up to four players can participate. All games come complete with sound effects and music.

The unit sells for $299, which includes two game cassettes. Contact Robert E Wiles, director of marketing, consumer products, Midway Manufacturing Corporation, 10750 Grand Av, Franklin Park IL 60131.■

Circle 581 on inquiry card.

OAE Announces New PROM Programmer



OAE's new programmable read only memory (PROM) programmer should be of interest to the experimenter. This is the PP 2708/16, designed to program the 2708 and the 2716 PROMs which are available from a variety of sources. A parallel interface is used to connect the unit to any microcomputer; an internal address unit is built into the unit, plus timing and control logic and DC to DC power conversion from unregulated 8 VDC input.

The PP 2708/16 comes complete with a black anodized aluminum case, 5 foot ribbon cable with connectors, and software. The price is $249 for the kit, or $299 for the assembled and tested unit. Contact Oliver Audio Engineering Inc, 7330 Laurel Canyon Blvd, North Hollywood CA 91605.■

Circle 582 on inquiry card.

Turn Your Microcomputer Into a Programmable Scientific Calculator

From Artisan Electronics, 5 Eastmans Rd, Parsipanny NJ 07054, comes news of a new microcalculator called the Model 85. This new unit is the single board equivalent of a hand held scientific calculator but without a keyboard or case, designed to interface directly to a microcomputer.

Instruction entry to the microcalculator Model 85 is under microprocessor software control. The Model 85 accepts instructions, provides a means to detect busy status, and outputs the full 14 digit display back to the microprocessor for storage or display.

Software for controlling the Model 85 in both a read and write mode requires less than 256 bytes of microprocessor system memory.

The Model 85 has scientific calculation capabilities for handling scientific, engineering, mathematical or statistical problems. Problem solving capability includes transcendental functions, such as logarithms, sines and tangents; polar to rectangular coordinate conversions for handling complex arithmetic, vector; multiple storage registers, selecting operating mode and also constants for π and e are provided as well as four metric to English unit constants for conversions between inches and centimeters, liters to gallons, and so on. Statistical routines are also provided for calculation of means and standard deviations. The price is $189.■

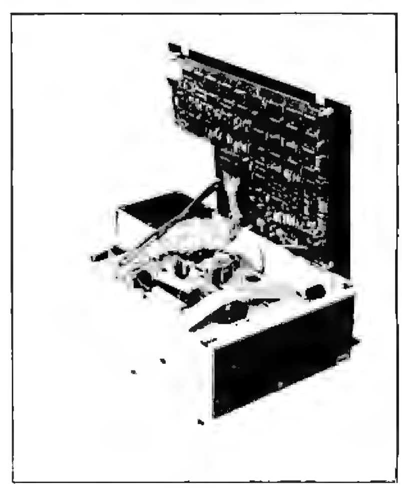
Circle 583 on inquiry card.

Innovex Introduces New Series 400 Floppy Disk Drives

Innovex Inc, 75 Wiggins Av, Bedford MA 01730, has announced their new series 400 floppy disk drives which feature automatic head unload and stepper motor timeouts, bidirectional write protect, six LED activity indicators, and 50 pin ribbon cable or twisted pair interfacing capability.

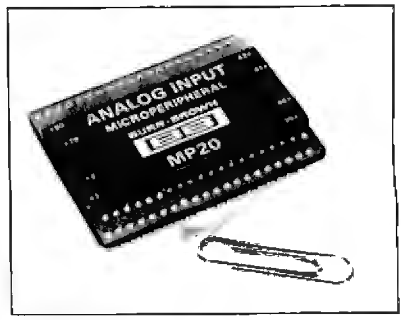
Both the Model 410, which is soft sectored and IBM compatible, and the Model 420, which is hard sectored, provide single and double density recording capability. A proprietary data separator design is said to offer improved data integrity margins in comparison with other available units.

Prices for the Innovex Series 400 start at $575 in single quantities. Contact Innovex for further information.■



Circle 584 on inquiry card.

Analog Input Module



Burr-Brown has announced their new MP20, a microprocessor compatible analog input which consists of a 16 channel analog multiplexer, high gain instrumentation amplifier, 8 bit analog to digital converter, plus all necessary address, data and control bus interfaces.

The MP20 is timing and logic level compatible with the 8080A and the 8008 processors. No external logic is needed. Gain and offset are internally laser trimmed, eliminating the need for external adjustments. Absolute accuracy is better than ±0.4% on the ±5 V or 0 to +5 V ranges.

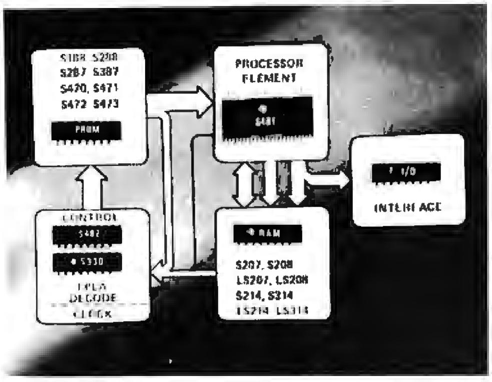
Low level signals such as thermocouple outputs can also be handled directly with reduced accuracy. The instrumentation amplifier can be programmed with a single external resistor to provide input signal ranges as low as ±10 mV full scale.

In actual use, the MP20 is treated as memory. Each analog input channel occupies one memory location. Any memory reference instruction can be used to access data. Thus, one LDA instruction will input data from one channel to the accumulator. Two adjacent input channels can even be acquired with one LHLD instruction. The MP20 can also be interfaced as IO.

Power requirements for the unit are ±15 VDC and +5 VDC. The price is $195 in single unit quantities. Contact C R Teeple, product manager, Burr-Brown, International Airport Industrial Park, Tucson AZ 85734.■

Circle 585 on inquiry card.

Nybbling Thinly Sliced Bytes



Texas Instruments has announced a new set of 4 bit slice microcomputer circuitry which will be of great interest to experimenters wanting to try their hands at designing a microcomputer from scratch.

The S481 chip set is a new series of Schottky TTL microprogrammable building block integrated circuits which are said to offer up to ten times the throughput of conventional microprocessors.

The S481 chip set can select and operate on two operands, generate status and store results in 100 ns. The set consists of one or more 4 bit slice processors, one or more 4 bit slice controllers, an appropriate number of field programmable logic arrays (FPLAs), and a variety of Schottky programmable read only memories and programmable memories.

The experimenter can choose word size to be any multiple of four, making this a very versatile system. The number of general purpose registers which can be created is limited only by the number of directly accessible memory locations. A memory to memory system places the register file function in main memory; this system can be designed to handle interrupts or context switches faster than conventional register to memory systems.

For more information, contact Texas Instruments Inc, Inquiry Answering Service, POB 5012, M/S 308 (Attn: S481), Dallas TX 75222.■

Circle 586 on inquiry card.

Periodical Guide for Computerists

The *Periodical Guide for Computerists* is a new 20 page book that indexes over 1000 personal computing articles from 15 magazines (including BYTE) for the period from January thru December 1976. The articles are indexed under more than 100 subject categories. Indexed are magazine articles, letters from readers, book reviews and editorials from both hobbyist and professional publications.

The books are available from E Berg Publications, 1360 SW 199th Ct, Aloha OR 97005, for $2.50 each, postpaid, and also from local computer stores.■

Circle 587 on inquiry card.

A New Product Guide from RCA

RCA has announced the MPG-180, a new 40 page guide to their COSMAC family of microprocessor products. Hardware products described include the CDP1800 series of CMOS integrated circuits. This series features central processing units (CPUs), programmable memories, read only memories, IO circuits and interfacing circuitry. The MPG-180 describes the CDP1802 microprocessor and covers its features, architecture, ratings, characteristics, timing diagram and instruction summary.

Among the additional items described are a Microtutor which teaches programming techniques, a floppy disk system, a binary arithmetic package, timeshare software development packages and a cross-reference guide to other manufacturers' integrated circuits.

The free MPG-180 product guide may be obtained by writing to RCA Solid State Division, POB 3200, Somerville NJ 08876.■

Circle 588 on inquiry card.

Is Your Computer System Half-Baked?

We received the following information from The Computer Center Inc, 321 Pacific Av, San Francisco CA 94111: A new *Microcomputer Recipe Book* is available free from the Computer Center, listing everything from soup to nuts that you need to put together your own microcomputer operating system for personal, business or scientific use. Items listed under "Ingredients" include computers, semiconductor and floppy disk memories, video displays and printers. Suggested menus for complete systems range from the "BIG MAC," a simple 8080-based computer used in conjunction with the family TV set to the "Beef Wellington," which includes a Processor Technology Sol-20 microcomputer, dual floppy disk drives, a DECwriter II printer and a video monitor. Technical books dealing with hardware design, software development and "How To Do It" books are also offered.■

Circle 589 on inquiry card.

A Hand Held 4 Channel Logic Analyzer

A problem with multichannel digital logic analyzers for most computer experimenters is cost. Digital Broadcast Systems Inc, 4306 Governors Dr, Huntsville AL 35805, has provided a relatively low cost solution to this problem with their new 45-B Logic Analyzer. The 45-B is a hand held instrument which attaches to any conventional single trace oscilloscope having an external trigger to produce four simultaneous digital waveforms. A typical application would be to display the clock from a given circuit on one channel, and selected logic gate inputs and outputs on the other three channels. Complete instructions are printed on the analyzer itself.

The unit can be used with TTL, DTL, RTL or CMOS logic families. Two potentiometers control waveform amplitude and multiplexed display rate; once these controls are set, the scope time base control operates normally. The price, including a 9 V battery and AC adapter, is $149.95.■

Circle 590 on inquiry card.

Continental Specialties' New Catalog

Continental Specialties Corporation, 44 Kendall St, POB 1942, New Haven CT 06509, have announced their new 16 page catalog. It contains a variety of breadboarding aids such as integrated circuit test clips, solderless breadboards (some with self-contained power supplies), and logic probes. A line of instrumentation is also featured, including a function generator, RC bridge and pulse generator. There is a question and answer section in back of the catalog which answers frequently asked breadboarding questions. The catalog is free.■

Circle 591 on inquiry card

A New Floppy from Digital Systems

Digital Systems announces a new floppy disk system featuring Shugart Associates drives and Digital Systems FDC-1 controller. Disk formatting is IBM compatible, and diskette initialization capability is provided. The Digital Research CP/M disk operating system software is available as an option, as is an Altair bus compatible interface. The price for the assembled and tested unit is $1845. Contact Digital Systems, 6017 Margarido Dr, Oakland CA 94618.■

Circle 592 on inquiry card.

NEW COMPUTER INTERFACE BOARD KIT

Our new computer kit allows you to interface serial TTL to RS 232 and RS 232 to TTL. There are four of these supplied with the kit, so you can run up to four devices on one TTL or four separate TTL to RS 232 devices.

Typical use: You can use your computer ports to run an RS 232 printer, video terminal and two other RS 232 devices at once, without constantly connecting and disconnecting your terminals.

$49.00

Example: Out store to printer — Voltage requirement +5V and ± 5V or ± 12 V depending on your RS 232 device.

We supply — board, connectors, documentation and components. Sorry, we do not supply case or power supply.

WHERE IT MAKES SENSE, MAY BE USED WITH ANY 8080, 6800, Z80 or F8 COMPUTER

GENERAL PURPOSE COMPUTER POWER SUPPLY KIT

This power supply kit features a high frequency torroid transformer with switching transistors in order to save space and weight. 115V 60 cycle primary. The outputs with local regulators are 5V to 10A, in one amp increments, – 5V at 1A, ± 12V at 1A regulators supplied 6 340T-5 supplied.

$79.00

UNIVERSAL 4K MEMORY BOARD KIT $74.50

This memory board may be used with the F8 and with minor modifications may be used with KIM-1µp.

32-2102-1 static RAM's, 16 address lines, 8 data lines in, 8 data lines out, all buffered. On-board decoding for any 4 of 64 pages, standard 44 pin, .156" buss.

F8 EVALUATION BOARD KIT WITH EXPANSION CAPABILITIES

A fantastic bargain for only with the following features:

$99.00

- 20 ma or RS 232 interface
- 64K addressing range
- Program control timers
- 1K of on-board static memory
- Built in clock generator
- 64 Byte register
- Built-in priority interrupts
- Documentation
- Uses Fairbug PSU

FOR FAIRBUG 4K F8 BASIC ON PAPER TAPE $25.00

2708 8K EPROM	$20.95
2522 STATIC SHIFT REG	$ 1.95
2513 CHARACTER GEN	$ 9.95
2518 HEX 32 BIT SR	$ 3.50
2102-1 1024 BIT RAM	$ 1.29
5280 4K DYNAMIC RAM	$ 4.75
5202A UV PROM	$ 5.95
1101A-256 BIT RAM	$ .85
21078	$ 4.25
MK4008P	$ 1.95
1702A UV PROM	$ 4.85
5204 4K PROM	$10.95
AY 5-1013 UART	$ 6.95

MINIATURE MULTI TURN TRIM POTS 100, 500, 1K, 2K, 5K, 10K, 25K, 50K, 100K, 200K, 500K 1 Meg. $.75 each. 3/$2.00

MULTI TURN TRIM POTS Similar to Bourns 3010 style 3/16" x 5/8" x 1 1/4 50, 100 1K, 10K, 50K ohms $1.50 ea 3 $4.00

LIGHT ACTIVATED SCR's TO 18 200V 1A $ 1.10

TRANSISTOR SPECIALS

2N3585 NPN Si TO 66	$ .95
2N3772 NPN Si TO 3	$ 1.60
2N456A PNP GE	$ .75
2N4908 PNP Si TO 3	$ 1.00
2N6056 NPN Si TO 3 Darlington	$ 1.70
2N5086 PNP Si TO 92	4/$ 1.00
2N4898 PNP TO 66	$ .60
2N404 PNP GE TO 5	5/$ 1.00
2N3919 NPN Si TO-3 RF	$ 1.50
MPSA 13 NPN Si TO 92	3/$ 1.00
2N3767 NPN Si TO 66	$ .70
2N2222 NPN Si TO 18	5/$ 1.00
2N3055 NPN Si TO 3	$ .70
2N3904 NPN Si TO 92	5/$ 1.00
2N3906 PNP Si TO 92	5/$ 1.00
2N6298 NPN Si TO-220	$ .50
2N6109 PNP Si TO 220	$ .55
2N3638 PNP Si TO 5	5/$ 1.00
2N5517 NPN TO 92 Si	3/$ 1.00

C/MOS (DIODE CLAMPED)

74C10	22	4012	.22	4023	22	4042-	78
74C193	1.50	4013	40	4024-	75	4046	2.25
4001	22	4015	95	4025	22	4049	40
4002	22	4016	40	4027-	40	4050	40
4006	1.20	4017	1.05	4028	88	4055	1.50
4007	22	4018	1.00	4029	1.10	4066	80
4009	42	4019	25	4030-	22	4071	.27
4010	42	4020	1.05	4033	1.50	4076	1.05
4011	22	4022	95	4035-	1.10		

1N4148 ... 15/$1.00

... $1.50

LED READOUTS

FND 359 C.C.	4"	$ .50	HP 7740 3" C.C.		$1.25
FND 70 C.C.	4"	$ .55	MAN 7 3" C.A.		$ .95
FND 503 C.C.	5"	$ .85	NS 33 3 dig array		$ .75
FND 510 C.A.	5"	$ .85	DL 747 C.A.	6"	$1.95

PRINTED CIRCUIT BOARD

4-1/2" x 6-1/2" SINGLE SIDED EPOXY BOARD 1/16" thick unetched $.60 ea 5/$2.60

7 WATT LD-65 LASER DIODE IR $8.95

2N 3820 P FET	$ .45
2N 5457 N FET	$ .45
2N2646	$ .45
ER 900 TRIGGER DIODES	4/$1.00
2N 6028 PROG UJT	$ .65
8 PIN DIP SOCKETS	$ .24
14 PIN DIP SOCKETS	$ .25
16 PIN DIP SOCKETS	$ .28
18 PIN DIP SOCKETS	$ .30
24 PIN DIP SOCKETS	$ .40
28 PIN DIP SOCKETS	$ .50
40 PIN DIP SOCKETS	$ .60

VERIPAX PC BOARD

This board is a 1/16" single sided paper epoxy board, 4½"x6½" DRILLED and ETCHED which will hold up to 21 single 14 pin IC's or 8, 16, or LSI DIP IC's with busses for power supply connector. $4.00

MV 5691 YELLOW GREEN BIPOLAR LED	$ .90
FP 100 PHOTO TRANS	$ .60
RED, YELLOW, GREEN or AMBER LARGE LED's	6/$1.00
IL-5 (MCT-2)	$ .75
MOLEX PINS	100/$1.00
	1000/$8.00
10 WATT ZENERS 3.9, 4.7, 5.6, 8.2, 12, 15, 18, 22, 100, 150 or 200V	ea. $ .60
1 WATT ZENERS 4.7, 5.6, 10, 12, 15, 18 or 22V	ea. $ .25
MC6860 MODEM CHIP	$9.95

Silicon Power Rectifiers

PRV	1A	3A	12A	50A	125A
100	.06	.14	.30	.80	3.70
200	.07	.20	.35	1.15	4.25
400	.09	.25	.50	1.40	6.50
600	.11	.30	.70	1.80	8.50
800	.15	.35	.90	2.30	10.50
1000	.20	.45	1.10	[illegible]	12.50

SILICON SOLAR CELLS

2¼" diameter

.4V at 500 ma. $4.00 / .2V at 200 mils $2.00

REGULATORS

309K	$ .95	340K-5, 12, 15 or 24V	$ .85
723	$ .50	340T 5, 6, 8, 12, 15, 18 or 24V	$1.10
LM 376	$ .60	78 MG	$1.35
320K-5 or 15V	$1.40	79 MG	$1.35
320T-5, 12, 15 or 24V	$ .85		

RS232 CONNECTORS	
DB 25P male	$2.95
DB 25S female	$3.50

TANTULUM CAPACITORS

.22UF 35V	5/$1.00	10UF 20V	$ .25
.47UF 35V	5/$1.00	22UF 25V	$ .40
.68UF 35V	5/$1.00	15UF 35V	3/$1.00
1UF 35V	5/$1.00	30UF 6V	5/$1.00
2.2UF 20V	5/$1.00	33UF 15V	$ .40
3.3UF 35V	4/$1.00	47UF 20V	$ .35
4.7UF 15V	5/$1.00	68 UF 15V	$ .50
6.8UF 15V	4/$1.00		

7001 ALARM CLOCK CHIP $5.75

NATIONAL MOS DEVICES

MM1402	1.75	MM5057	2.25
MM1403	1.75	MM5058	2.75
MM1404	1.75	MM5060	2.75
MM5013	2.50	MM5061	2.50
MM5016	2.50	MM5555	4.75
MM5017	2.70	MM5556	4.75
MM5055	2.25	MM5210	1.95
MM5056	2.25	MM5260	1.75

TTL IC SERIES

7400	14	7445-	.55	74151-	.60
7401-	14	7446-	.85	74153-	.60
7402-	14	7447-	65	74154-	.95
7403-	14	7448-	65	74155	70
7404-	.18	7450-	15	74157-	58
7405-	18	7472-	29	74161	.85
7406	25	7473-	29	74163	.80
7407-	25	7474-	29	74164-	.95
7408	18	7475-	45	74165-	.95
7409-	17	7476-	30	74173-	1.20
7410	14	7480-	35	74174-	.95
7411-	.20	7483-	62	74175	82
7412	.20	7485-	87	74176-	75
7413	39	7486-	30	74177-	75
7414-	63	7489	1.85	74180	65
7416-	25	7490-	42	74181-	1.90
7417-	25	7491-	58	74190-	1.00
7420-	14	7492-	43	74191-	1.00
7425-	25	7493-	45	74192	83
7426-	.22	7494-	70	74193-	83
7427-	25	7495-	65	74194-	85
7430-	.14	7496-	.65	74195-	52
7432-	25	74107-	28	74196-	.86
7437-	21	74121	.33	74257	1.25
7438-	21	74123-	66	74279	.87
7440-	14	74125	40	75324	1.75
7441-	70	74126	.40	75491	65
7447	40	74150-	.90	75492	65

MINIATURE DIP SWITCHES

CTS 206-4 Four SPST switches in one mini dip package $1.75

CTS 206-8 Eight SPST switches in a 16 pin DIP package $1.95

5 HV SPST ... reed relay ... normally open ... 3500 ohm coil resistance $.75 ... $2.00

ALCO MINIATURE TOGGLE SWITCHES

MTA 106 SPDT	$1.20
MTA 206 DPDT	$1.70

Full Wave Bridges

PRV	2A	6A	25A
200	.75	1.25	2.00
400	.95	1.50	3.00
600	1.20	1.75	4.00

SANKEN AUDIO POWER AMPS

Si 1010 G 10 WATTS	$ 7.95
Si 1020 G 20 WATTS	$15.95
Si 1050 G 50 WATTS	$27.95

WSU 30-Hand wire wrap tool used to wrap, unwrap & Strip #30 wire $5.50

#24, EIGHT CONDUCTOR SPECTRA FLAT CABLE 10'/$1.50 100'/$13.50

DIP SOCKETS

8 PIN	22	24 PIN	.40
14 PIN	25	28 PIN	.50
16 PIN	28	40 PIN	.60
18 PIN	30		

#30 WIRE WRAP WIRE SINGLE STRAND 100'/$1.40

74LS SERIES / LINEAR CIRCUITS

		74LS125	65		
74LS00	23	74LS126	65		
74LS02	23	74LS132	80	LM 101	75
74LS03	23	74LS136	39	LM 301 748	31
74LS04	28	74LS138	72	LM307	30
74LS05	28	74LS139	72	LM 308	95
74LS08	23	74LS151	98	LM 311	95
74LS09	33	74LS153	93	LM 318	1.35
74LS10	23	74LS155	1.40	LM 319	95
74LS11	.23	74LS156	95	LM 324	1.05
74LS13	50	74LS157	98	LM 339	1.10
74LS15	28	74LS160	1.02	LM 358	1.40
74LS20	23	74LS161	1.02	LM 377	2.50
74LS21	23	74LS162	1.02	LM 380	95
74LS22	23	74LS163	1.02	LM 381	1.25
74LS26	33	74LS168	1.10	LM 382	1.35
74LS27	27	74LS169	1.10	LM 537	2.50
74LS30	23	74LS170	1.72	LM 553	2.50
74LS32	33	74LS173	1.39	LM 555	.44
74LS37	37	74LS174	1.05	LM558	85
74LS38	37	74LS175	1.22	560	2.00
74LS40	33	74LS279	75	565	1.10
74LS51	.23	74LS190	1.50	566	1.50
74LS54	23	74LS191	1.50	567	1.50
74LS74	49	74LS192	1.75	703	90
74LS90	95	74LS193	1.75	709	35
74LS42	88	74LS195	1.75	710	35
74LS73	40	74LS196	99	711	35
74LS74	40	74LS197	99	741C or V	31
74LS76	40	74LS221	1.25	747	65
74LS90	89	74LS257	1.35	LM 1310	2.60
74LS92	85	74LS258	1.38	1456	95
74LS93	85	74LS365	66	1458	50
74LS109	50	74LS366	66	CA3046	75
74LS112	43	74LS367	68	CA3047	95
74LS113	43	74LS368	68	3900	49
74LS114	43	74LS390	2.20	8038CC	3.90

TRIACS / SCR'S

PRV	1A	10A	25A	1.5A	6A	35A
100	40	70	1.30	40	50	1.20
200	70	1.10	1.75	60	70	1.60
400	[illegible]	1.60	2.60	1.00	1.20	2.20
600	[illegible]	[illegible]	[illegible]		[illegible]	[illegible]

Terms: FOB Cambridge, Mass. Send Check or Money Order. Include Postage, Minimum Order $5.00, COD'S $20.00

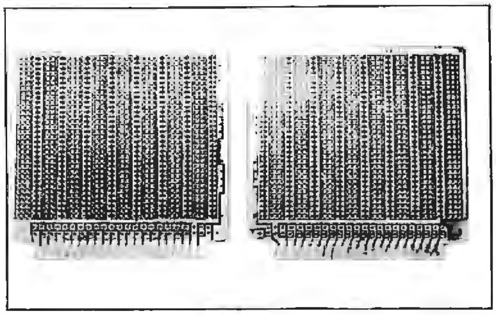
Send 25¢ for our catalog featuring Transistors and Rectifiers 145 Hampshire St., Cambridge, Mass.

SOLID STATE SALES
P.O. BOX 74B
SOMERVILLE, MASS. 02143 TEL. (617) 547-4005

WE SHIP OVER 95% OF OUR ORDERS THE DAY WE RECEIVE THEM

Circle 463 on inquiry card.

A New Printed Circuit Board for the Experimenter



Midgard Electronics Inc, 26 Walnut St, Watertown MA 02172, has announced the UMB, an expoxy glass board with 1380 plated through holes in a two-sided foil pattern. There are four power distributions, and pad spacing is 0.1 inch (0.254 cm).

All four power distributions are bused to each integrated circuit position to provide short interconnection distances. A reverse foil pattern minimizes characteristic impedances.

The UMB will accomodate all dual in line integrated circuit packages as well as discrete components. Exterior connection is provided by a 22 pin edge connector compatible with the Amphenol type 225-2221-101 connector. The price is $17.15 in quantities of one to six.

Circle 593 on inquiry card.

Acoustic Coupler

Anderson Jacobson announces the new A 242A acoustic coupler which operates at data rates up to 450 bps in the send mode.

The A 242A features:

- Flush mounted acoustic cups to lock in handset.
- A vibration isolation technique utilizing direct microphone handset coupling.
- Quartz crystal control of both transmitter and receiver.
- A carrier detector designed to sense valid data regardless of the carrier level.
- RS-232 and 20 mA interfaces provided.
- A rubber foot design for vibration isolation.

The A 242A is priced at $365 in single quantities. Contact Anderson Jacobson Inc, 521 Charcot Av, San Jose CA 95131, (408) 263-8520. ■

Circle 594 on inquiry card.

A New Business Software and Hardware Package

Computer Mart of New Jersey Inc, 501 Rt 27, Iselin NJ 08830, has announced a new business oriented small computer system. It stores information on floppy disks for random access, and features both video display and hard copy printing of business reports. A custom software package allows the user to communicate in simple English phrases. Contact the Computer Mart for further details.■

Circle 595 on inquiry card

A Lower Case Character Kit

Northern Valley Systems announces the LC01, an ADM-3 lower case option kit. This kit will allow Lear Siegler's ADM-3 and ADM-3A video display terminals to display lower case characters with no modifications to the PC board. The kit includes refresh memory chips as well as a custom programmed character generator. It can be installed in approximately ten minutes. The price is $27.50; dealer and group purchase discounts are available.

The LC01 is available from Northern Valley Systems, POB 687, Englewood, NJ 07631.■

Circle 596 on inquiry card.

Reader Service

To get further information on the products advertised in BYTE, fill out the reader service card with your name and address. Then circle the appropriate numbers for the advertisers you select from this list. Add a 9 cent stamp to the card, then drop it in the mail. Not only do you gain information, but our advertisers are encouraged to use the marketplace provided by BYTE. This helps us bring you a bigger BYTE

Reader Service Number	Page Number
441	AAA Chicago Computer Center p. 185
382	Alpha Digital p. 82
454	Ancrona pp. 188, 189
391	Anderson Jacobson p. 107
357	Apple pp. 14, 15, 16
378	Artec Electronics p. 73
446	AVTAR Systems p. 187
383	BITS pp. 83, 144, 151, 175, 176
392	BPI p. 109
409	Byte Shop East p. 155
451	Byte Shop of Miami p. 187
*	BYTE Subscription p. 93
374	Canada Systems p. 59
388	CMC Marketing Corp p. 101
398	Compumart Inc p. 123
363	Computalker p. 36
433	Computer Age p. 185
444	Computer Corner p. 186
452	Computer Creations p. 187
449	Computer Depot p. 187
400	Computer Enterprises p. 127
356	Computerland pp. 12, 13
431	Computer Mart of NH p. 184
442	Computer Playground p. 186
415	Computer Room p. 171
437	Computer Systems Store p. 185
353	Cromemco pp. 1, 2
379	Databyte p. 75
423	Data Search p. 179
396	Digital Group p. 119
410	Digital Micro Systems p. 155
438	Disc 3 p. 185
455	DRC p. 191
351	ECD Cover III
450	Economy Terminals p. 187
386	E & L Instruments p. 97
432	EDP Professionals p. 184
436	Electravalue p. 185
464	Electrolabs p. 206
425	Electronic Control Technology p. 180
440	Electronic Discount Sales p. 185
360	Extensys p. 31
419	Floto, Charles p. 177
456	Godbout p. 193
*	Hayden Books p. 135
394	Heath pp. 112, 113
389	Heuristics p. 103
372	Information Terminals p. 51
371	Integral Data Systems p. 49
439	Intelligent Business Machines p. 185
417	International Data Systems p. 172
435	IOR p. 185
416	Ithaca Audio p. 171
458	Jade p. 195
352	James Electronics Cover IV
459	James Electronics pp. 196, 197
395	John Fluke Mfg p. 117
453	JVR Computer Store p. 187
407	Kent-Moore p. 145
460	Meshna p. 199
380	MicroAids p. 80
414	Micronics p. 167
397	Micro Peripherals p. 33
390	Micropolis p. 105
369	Microware Systems Corp p. 47
443	Microtex Inc p. 186
465	Mikos p. 207
408	MiniTerm p. 149
461	Morrow's Micro Stuff p. 201
421	mpi p. 178
411	MSI p. 157
412	Mullen Electronic Boards p. 159
413	National Multiplex p. 161
359	North Star Computer pp. 21, 77
384	Objective Design p. 85
365	Ohio Scientific pp. 38, 39, 40, 41
457	Oliver Audio Electronics p. 194
399	Osborne & Associates p. 125
434	Outer Products p. 185
428	PAIA Engineering p. 182
418	Parasitic Engineering pp. 174, 181
*	Per Com Data pp. 11, 173
354	PERCOMP 78 p. 5
387	Peripheral Vision p. 99
420	Perri-White Assoc p. 178
361	Per-Sci p. 69
*	Personal Computing Expo pp. 162-165
*	PolyMorphic Systems p. 7
355	Processor Tech pp. 8, 9, 10, 52, 53
368	Quantronics p. 45
*	Radio Shack of Fort Worth p. 43
406	RHS Marketing p. 139
430	Rotundra Cybernetics p. 184
377	Scelbi Computer Consulting p. 121
*	Scientific Research pp. 44, 63
462	S D Sales p. 203
376	Seals Electronics p. 67
381	Smoke Signal Broadcasting p. 81
427	Software Exchange p. 182
375	Software Records p. 61
447	The Software Store p. 187
364	Solid State Music p. 37
463	Solid State Sales p. 205
445	Sunny Computer Store p. 187
350	SWTPC Cover II
429	Sybex p. 183
401	Synchro Sound pp. 128, 129
448	Szerlip Enterprises p. 187
426	Tarbell Electronics p. 181
373	Technical Design Labs p. 57
403	Technical System Consultants p. 133
402	Technico Inc p. 131
388	TEI p. 101
362	Texas Instruments pp. 34, 35
358	TLF p. 17
466	Urban Instruments p. 207
393	Vector Electronics p. 111
385	Vector Graphic p. 85
422	Vectron p. 179
424	Worldwide p. 180
405	Ximedia p. 137
370	Xybek p. 47

**Reader service inquiries not solicited. Correspond directly with company.*

BOMB
BYTE's Ongoing Monitor Box

ARTICLE NUMBER	ARTICLE	PAGE NUMBER
1	**Jenkins: How to Write an Application Program**	18
2	**Borrmann: Relocatability and the Long Branch**	26
3	**Duda: Othello, A New Ancient Game**	60
4	**Wimble: An APL Interpreter for Microcomputers, Part 3**	64
5	**Chung: An 8080 Simulator**	70
6	**Kruglinski: How to Implement Space War**	86
7	**Smith: Fundamentals of Sequential File Processing**	114
8	**Madden: C: A Language for Microprocessors?**	130
9	**Schneider: Simple Approaches to Computer Music Synthesis**	140
10	**Higgins: Structured Program Design**	146
11	**Holladay: Computer Information Arrangement**	156
12	**Bauerschnub: Analyze Your Car's Gas Economy with Your Computer**	166
13	**Milligan: Mastermind**	168

July BOMBworks

The results of the July 1977 BOMB voting found the winner to be James R Boddie's "Speech Recognition for a Personal Computer System," page 64. James' article placed first in the voting by BYTE readers, at a distance of 1.75 standard deviations above the mean vote for 14 articles in the July voting. Second place was a tie between W Douglas Maurer's "How to Pick Up a Dropped Bit," page 72, and Robert Grappel's "Give Your Micro a Megabyte," page 78. Bonus checks of $100, $50 and $50 were sent to authors Boddie, Maurer and Grappel respectively.■

WE'RE OUT OF OUR MINDS!

OUR MICROMIND COMPUTOR HAS MET WITH SUCH CONSUMER ENTHUSIASM THAT WE'RE BUSY PARING DOWN A SIX MONTH PILE OF ORDERS....

PLEASE BE PATIENT.... IT'LL BE WORTH IT!

FOR INFORMATION ON MICROMIND CALL (617) 661-4400

MicroMind

off... and rolling!

ECD

ECD CORP.
196 BROADWAY, CAMBRIDGE
MASS. 02139

INNOVATIONS IN ELECTRONICS

Circle 351 on inquiry card.

ATTENTION DEALERS:

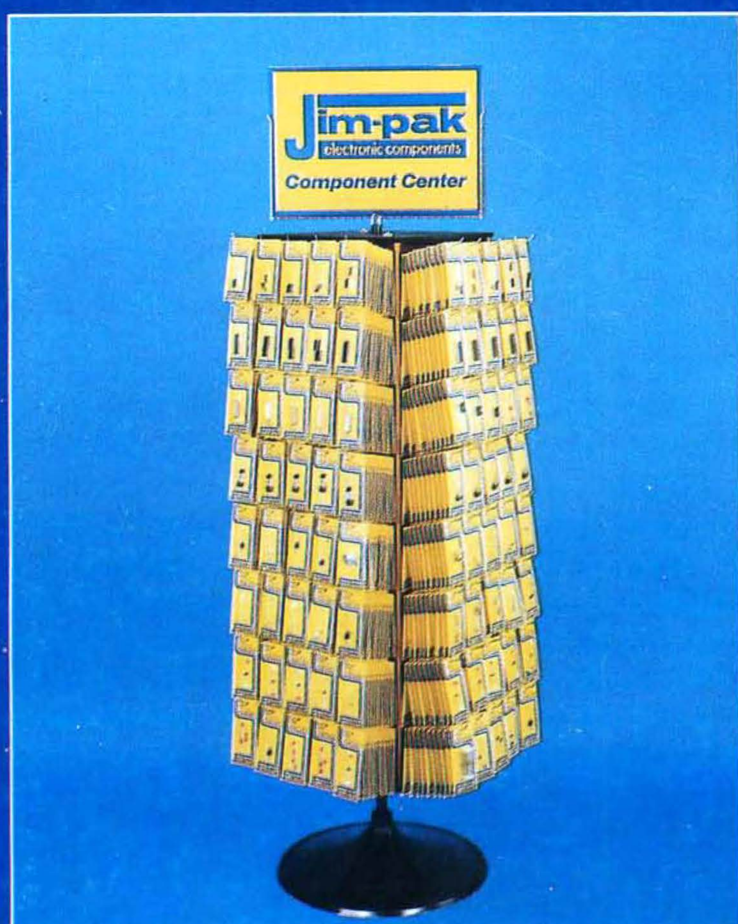
Announcing

Jim-pak®

electronic components

One-Stop Component Center

- ★ Over 200 quality items including integrated circuits, resistors, diodes, transistors, capacitors, connectors, switches, sockets, LEDs and Data Books covering all JIM-PAK® items.
- ★ Immediate delivery on all orders
- ★ Store display racks available
- ★ Stock rotation and return policy
- ★ Direct mail program available from list of active electronic buyers in dealers' area.
- ★ National advertising campaign in leading electronics magazines to include list of qualifying dealers
- ★ Nationally known manufacturers' products at prices every dealer can afford
- ★ Guaranteed products
- ★ Standard industry part numbers



A component line of proven sellers developed for the independent dealer. Ideal for computer shops, school stores, electronic dealers, hobby shops, or any location where there is a potential market for electronic sales.

A product line which supplies most of your needs from one distributor with a reputation for fast and efficient service. Attractive and compact display racks make initial installation of the JIM-PAK® line easy.

Your customers deserve the best. Now you can profitably retail name brand components at competitive prices. Be the first in your area to announce and sell the JIM-PAK® line. Write or call today.

FOR MORE INFORMATION AND PRICING SCHEDULE CONTACT:

Jim-pak® electronic components, a division of James Electronics, 1021 Howard Avenue, San Carlos, California 94070, (415) 592-8097

Circle 352 on inquiry card.