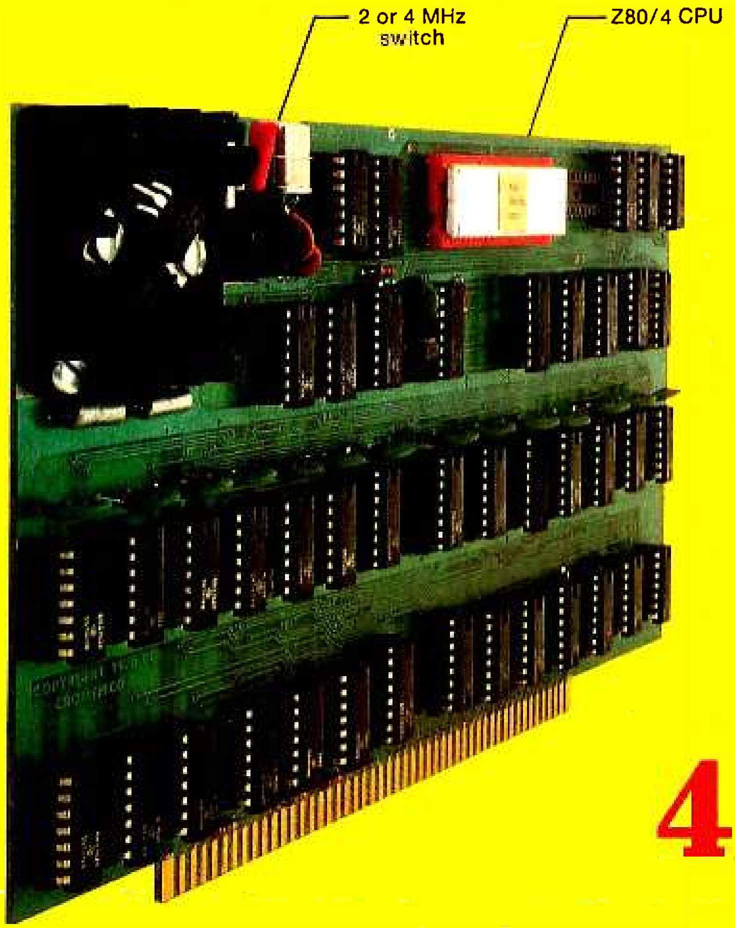
ISSUE NUMBER 14 $1.50 IN NORTH AMERICA ($2.50 elsewhere) OCTOBER 1976

BYTE

the small systems journal

PRINTED IN USA



The only CPU card to give you 4 MHz speed

2 - 5X MORE THROUGHPUT

Here is by far the most powerful CPU card now available.

It's Cromemco's new ZPU™ card.

It uses the slick new Z-80 chip — in fact, it uses the even faster Z80/4 high speed version of the Z-80 — and it's the *only* card that does. The Z80/4 is **certified** by its manufacturer for 4 MHz operation.

The Z80/4 has all the advantages of the 8080 and 6800 — and enormously more.

And Cromemco's new ZPU **does** enormously more.

4 MHz CLOCK RATE

First, the ZPU lets you choose either a 2 or 4 MHz crystal-controlled clock rate. Right away that means you can have twice the throughput. Cuts program running time in half. Then the instruction set of the Z80/4 reduces software even more.

The 2 or 4 MHz clock rate is switch-selectable as shown in the above photo.

POWER-ON MEMORY JUMPS

Cromemco's ZPU also has some neat design innovations of its own. For example, you'll like the simplified operation you get because upon power turn-on the ZPU will jump to any desired 4K boundary in memory. No switch flipping to go through to begin your program.

SELECTABLE WAIT STATES

Cromemco engineers have also arranged that your present systems will always be useful with the new ZPU. To do this, the ZPU has been designed to have jumper-wire-selectable wait states on the card.

These simplify interfacing with your present memory or I/O even at 4 MHz operation.

80 ADDITIONAL INSTRUCTIONS

You've probably heard that the Z-80 with its 80 new additional instructions is by far the most powerful chip around. It's true.

That means with the ZPU you will be able to devise much more powerful (as well as faster) software than before.

ALTAIR/IMSAI COMPATIBLE WITHOUT MODIFICATION

Yes, the new ZPU is plug-compatible with the Altair 8800 and IMSAI 8080. Just remove the existing CPU, plug in the ZPU card, and you're up and running.

Further, the Cromemco ZPU is the only card guaranteed to work with all present and future Cromemco peripherals. (Cromemco manufactures the popular BYTESAVER™ memory, the TV DAZZLER™, the D+7A™ analog interface board, a joystick console, and others.

INCLUDES FREE SOFTWARE

The ZPU comes with our powerful Z-80 monitor, complete documentation, source code, and paper tape object code. The monitor is also available in PROM ($75) for use in our BYTESAVER memory board.

STORE/MAIL

The new ZPU is available as a kit or assembled. Look into it now because you can see demand will be strong. Present delivery is 30 days.

ZPU kit (Model ZPU-K) $295
ZPU assembled
(Model ZPU-W) $395

California users add 6% sales tax.

Cromemco

Specialists in computer peripherals

2432 Charleston Rd., Mountain View, CA 94043 • (415) 964-7400

About the Cover

As a way to highlight the history of electronic digital signalling, we dug up a picture of one of Joseph Henry's original telegraphy keys, circa the early 1800s. Robert Tinney then placed the key in the frame and wall setting you see on the cover, using a photo supplied by Brian McCarthy.

In the Queue is on page 7 this month.

In This BYTE

The problem of decoding arbitrary hand generated Morse code is not a trivial one. It requires some care and thought in the design of adaptive algorithms. As one contribution to this issue's sub theme of computerized Morse code, Lt William A Hickey, USN, provides some background information and suggestions on the subject.

•

W J Hosking, W7JSW, is an amateur radio operator in search of applications hardware and software. Read about **A Ham's Application Dreams** and find out how to implement one aspect of his dream with the Morse code input and output conversion technology described in detail in the balance of this issue . . . .

•

A theme of this October issue is the application of microcomputers to the decoding of Morse code. One approach to the problem is detailed in Robert Grappel and Jack Hemenway's article on MORSER . . . a program to read Morse code, implemented with a Motorola 6800 computer.

Lawrence Krakauer describes a technique to store Morse characters as a packed table of bit patterns for machine generated outputs — or for machine decoded inputs.

•

If Only Sam Morse Could See Us Now. He'd have a fistful of problems trying to copy radio transmissions at 1000 wpm generated by programs such as Wayne Sewell's CWBUFFER subroutine. But, using one of Wayne's set of sundry drivers for CWBUFFER, Mr Morse could potentially learn to copy — or at least have his computer copy — in a code practice mode.

•

One application of the Morse code problem solvers is documented in Bruce Filgate's article on **Morse Code Station Data Handler.** This is an application program which handles direct sending of Morse outputs, from character text, adaptive interpretation of Morse inputs, storing of fixed messages (eg: ' CQ CQ CQ DE W1AW ') in a message buffer for later transmission or repetitive transmission, etc. Bruce has put it all together in the form of a comprehensive 1536 byte program for an 8008.

Once you sit down and **Build This Mathematical Function Unit** as described in part one of R Scott Guthrie's two part article, the world of high level mathematical functions is opened to your microcomputer. In part two this month, the software needed to interface with the calculator is described, as well as several test loops used to adjust timing parameters with an oscilloscope. As a final illustration of the calculator's use, the author provides a program called CALCULA which enables a Teletype (or other ASCII) port to drive the calculator and print results, simulating the ordinary hand calculator level of operation.

•

National Semiconductor announced the PACE computer some time ago, but until recently it has been somewhat hard to obtain. Now that this 16 bit minicomputer is beginning to enter its volume production stage, we **Keep PACE With the Times** by offering Robert Baker's Microprocessor Update on this processor. If you missed the convenience of your familiar 16 bit minicomputer when you started reading about and "dry run programming" for personal computing, then the PACE processor might be a logical choice for a homebrew or kit system.

•

The advent of the personal system portends a fundamental change in the ways computers are used. In **Homebrewery vs the Software Priesthood,** David Fylstra and Mike Wilber make some comments about the impact of widespread use and knowledge of computers.

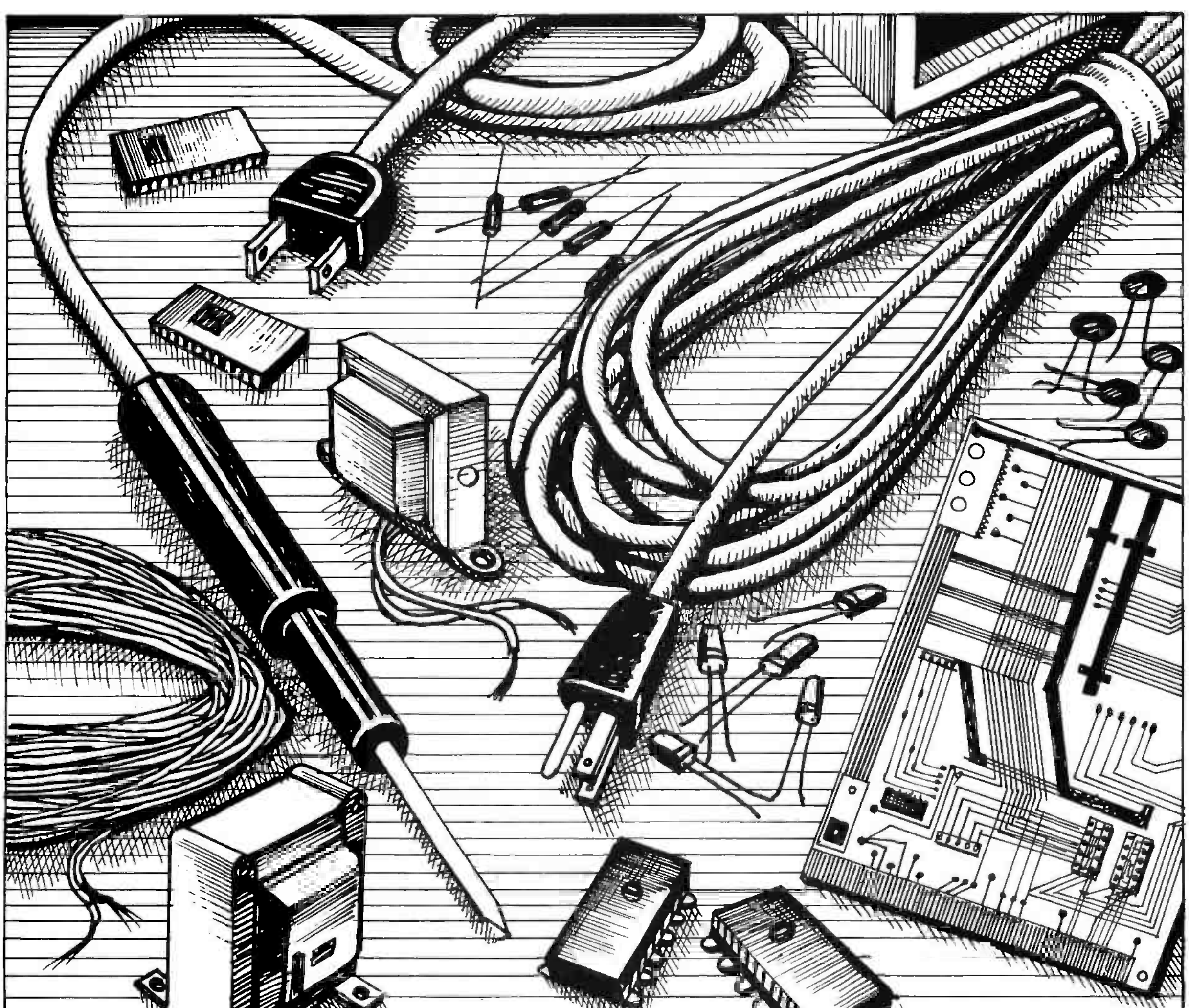
•

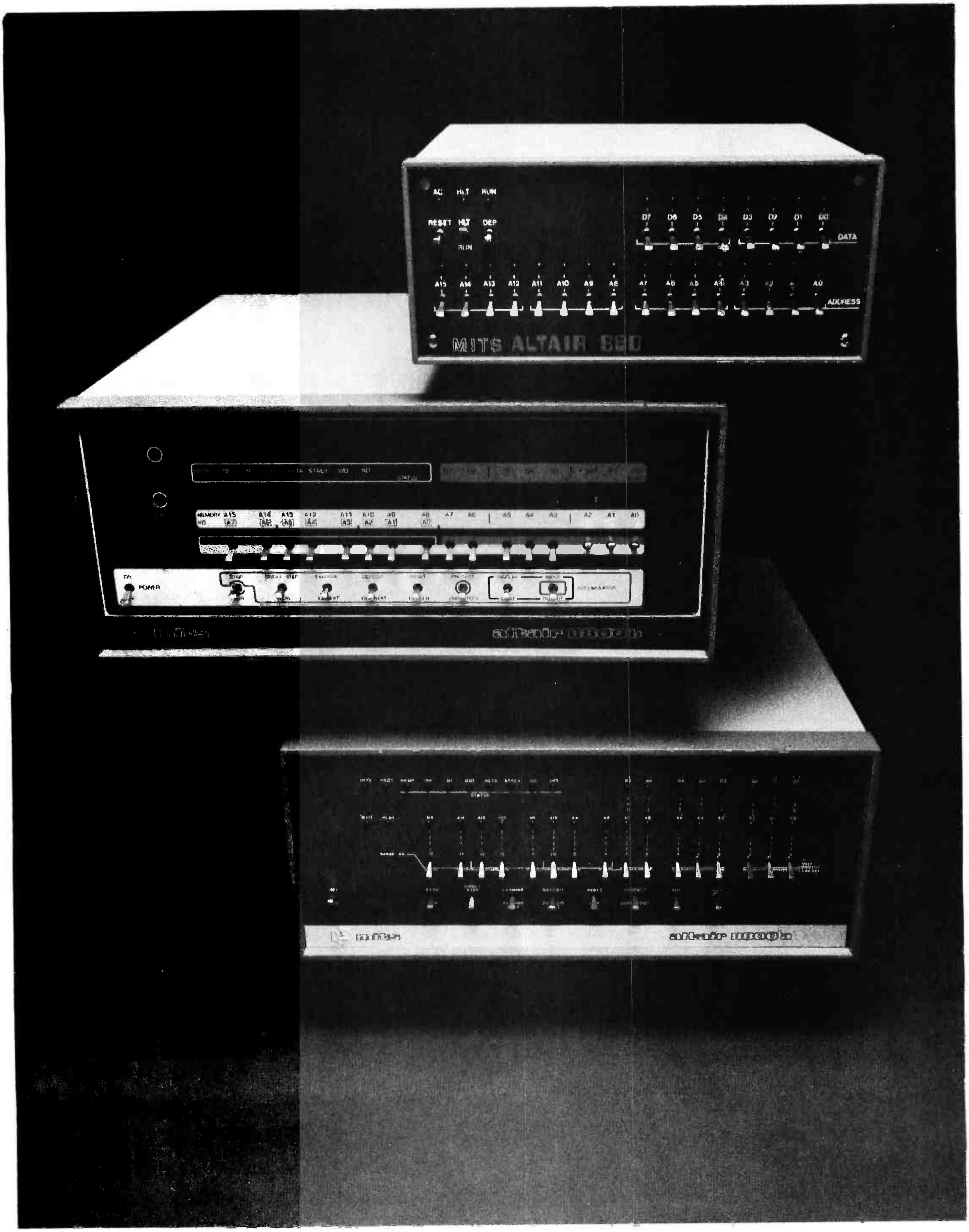
Looking for ideas for meetings of your local computer group? Dr Charles F Douds has a few suggestions to make in his background article on the subject this month.

KIT-A-MONTH

The Altair™ kit-a-month plan allows you to own an Altair mainframe without taxing your pocketbook. Mits has made it easy for you to purchase an 8800a, 8800b or 680b computer in monthly installments where you receive components with each payment. There are no financial charges because we have made each monthly shipment a kit in itself. This will give you time to read up on computers and/or gain knowledge from friends.

We have set up an Altair kit-a-month payment desk to service your needs. When writing or calling the factory for information about your shipment or account, just refer to the "kit-a-month payment desk."




AC HLT RUN
RESET HLT DEP
RUN
D7 D6 D5 D4 D3 D2 D1 D0
DATA
A15 A14 A13 A12 A11 A10 A9 A8 A7 A6 A5 A4 A3 A2 A1 A0
ADDRESS
MITS ALTAIR 680
POWER

OWN YOUR 680b IN FIVE EASY PAYMENTS

1st month	680b Manuals, Main and Display PC Boards, and membership in users group
2nd month	680b Case and Power Supply
3rd month	Parts for 680b Display Board
4th month	Parts for 680b Main Board less Integrated Circuits
5th month	680b Chips

$93.20 plus $2.00 per month make your payments $95.20 per month and you have your complete kit in five easy payments. This plan gives you the full 680b with memory and I/O. The BAUDOT Option is still $42.00 extra and may be purchased with Time Payment #4 or #5. Remittance of $95.20 will start you on your way to owning your own computer.

Alaska, Hawaii, APO and FPO customers include $3.00 per month for Air Parcel Post charges instead of $2.00 for regular Parcel Post shipment. This would make your payment $96.20 per month.

Canadian customers include $3.00 per each month for postage and handling fees.

NOTE: This plan does not apply to foreign sales other than Canada.

NEW ALTAIR 8800B

Available on Time Payment Plan

$105.00 payment per month plus $2.00 postage and handling for each kit makes an easy $107.00 per month to own the newest of the Altair processors.

Send in your first $107.00 money order and start receiving your 8800B Kit by August 1, 1976.

8800B Month #1	Manuals
#2	EC-18
#3	Power Supply Board & Parts
#4	Transformer
#5	Display Board & Parts
#6	Case
#7	Main Chips
#8	CPU Board & Parts Less 8080A

Alaska, Hawaii, APO and FPO customers please include $4.00 for shipping charges (making $109 per month payments) for Air Parcel Post shipment. Otherwise, shipment will come Parcel Post, not insured.

Canadian customers must accept month #6 Emery Airfreight Collect. All other months must include $4.00 postage and handling making monthly payments of $109.00.

$79.00 / Month
ALTAIR 8800A TIME PAYMENT PLAN

8800A Time Payment #1	8800A Manuals and Users Group Membership
#2	EC-18, PC Board and Hardware
#3	8800A Power Supply Kit
#4	8800A Case
#5	CPU PC Board and Bag of Parts less the main chip
#6	Main Processor Chip
#7	Display Control Board and Parts

The price of the Altair 8800A mainframe is $539.00. Seven easy payments plus $2.00 per month for postage and handling charges make this plan equal $79.00 per month. Upon receipt of your first $79.00 payment you are on your way to owning your own 8080A basic computer system. A list of available compatible peripherals is enclosed to let you plan your system as you learn about your microprocessor. By 8800A Time Payment #7 you're ready to go.

Alaska, Hawaii, APO and FPO customers please include $4.00 for shipping charges (making $81 per month payments) for Air Parcel Post shipment. Otherwise, shipment will come Parcel Post, not insured.

Canadian customers must accept month #4 Emery Airfreight Collect. All other months must include $4.00 postage and handling making monthly payments of $81.00.

KIT-A-MONTH

ORDERING INSTRUCTIONS

In order to smoothly and efficiently expedite your orders, we ask that you note the following helpful hints:

1. Send all payments other than BankAmericard or Master Charge in the form of a cashier's check or money order. Personal checks are acceptable, but clearance time will delay your order by 2-3 weeks.

2. The kit-a-month plan has been set up to proceed in order and we cannot deviate from that order. You can help us by noting with your payment what month you are on.

3. When calling or sending in orders, refer to your customer name on the original order and also your Mits order number.

4. If you change your address, keep your name as it is on the original order to keep records straight.

5. Please note special instructions for Alaska, Hawaii, APO, FPO and Canadian customers. If these are not followed, it could result in delays in processing your order.

6. The Kit-a-Month desk has been set up to help expedite your orders because of the overwhelming response we've had with previous time payment plans. Please feel free to use this service whenever you have questions. When writing letters to Mits, simply note "Kit-a-Month desk" on the outside of the envelope.

NOTE: Once you start the Kit-a-Month plan you are guaranteed the existing price at the time of your first order. You will not be affected by price increases.

Enclosed is my payment of ____________ for the first shipment of my Altair kit-a-month.

Master Charge #________________ or BankAmericard #________________

Altair 680b ☐ Altair 8800a ☐ ☐ Altair 8800b

NAME____

ADDRESS____

CITY________________________ STATE & ZIP________________

mits/2450 Alamo SE/Albuquerque, NM 87106 505-243-7821

Prices, specifications, and delivery subject to change.

mits/2450 Alamo SE/Albuquerque, NM 87106 505-243-7821

In the Queue

Foreground

Background

Nucleus

BYTE # 14

OCTOBER 1976

staff

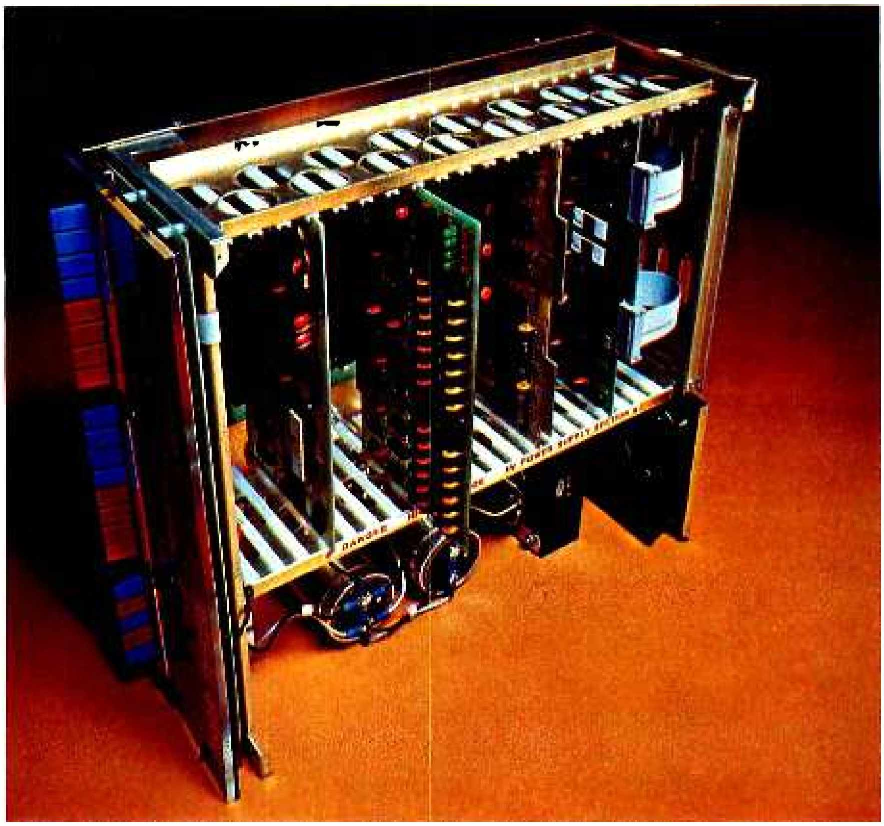
PUBLISHERS
Virginia Peschke
Manfred Peschke
EDITOR
Carl T Helmers, Jr
GENERAL MANAGER
Manfred Peschke
PRODUCTION MANAGER
Judith Havey
CIRCULATION
Deborah R Luhrs
DEALER CIRCULATION
Deena Zealy
PUBLISHERS ASSISTANTS
Cheryl Hurd
Julie Nelson
Carol Nyland
Kim Place
ADVERTISING
Elizabeth Alpaugh
Virginia Peschke
TYPOGRAPHY
Custom Marketing Resources, Inc
Goodway Graphics
Mary Lavoie
Taimi Woodward
PHOTOGRAPHY
Ed Crabtree
Custom Marketing Resources, Inc
ART
Matthew Arnold
Mary Jane Frohlich
Lynn Malo
Bill Morello
PRINTING
The George Banta Company
Custom Marketing Resources, Inc
ASSOCIATES
Bob Baker
Dan Fylstra
Harold A Mauch
Chris Ryland
PRODUCTS COORDINATOR
Floyd W Rehling
TECHNICIAN
Melvin G Weir
CONSTRUCTION
Walter Pence

BYTE magazine is published monthly by BYTE Publications Inc, 70 Main St, Peterborough New Hampshire 03458. Subscription rates are $12 for one year, $22 for two years, $30 for three years in North America. Elsewhere: $15 one year, $25 two years, $33 three years. Second class postage paid at Peterborough NH 03458 and at additional mailing offices. Phone 603—924-7217.

Address editorial correspondence to Editor, BYTE, 70 Main St, Peterborough NH 03458.

If you thought a rugged, professional yet affordable computer didn't exist,

think IMSAI 8080.



Sure there are other commercial, high-quality computers that can perform like the 8080. But their prices are 5 times as high. There is a rugged, reliable, industrial computer, with high commercial-type performance. The IMSAI 8080. Fully assembled, it's $931. Unassembled, it's $599. And ours is available now.

In our case, you can tell a computer by its cabinet. The IMSAI 8080 is made for commercial users. And it looks it. Inside and out! The cabinet is attractive, heavy-gauge aluminum. The heavy-duty lucite front panel has an extra 8 program controlled LED's. It plugs directly into the Mother Board without a wire harness. And rugged commercial grade paddle switches that are backed up by reliable debouncing circuits. But higher aesthetics on the outside is only the beginning. The guts of the IMSAI 8080 is where its true beauty lies.

The 8080 is optionally expandable to a substantial system with 22 card slots in a single printed circuit board. And the durable card cage is made of commercial-grade anodized aluminum.

The IMSAI 8080 power supply produces a true 28 amp current, enough to power a full system.

You can expand to a powerful system with 64K of memory, plus a floppy disk controller, with its own on-board 8080—and a DOS. A floppy disk drive, an audio tape cassette input device, a printer, plus a video terminal and a teleprinter. These peripherals will function with an 8-level priority interrupt system. IMSAI BASIC software is available in 4K, that you can get in PROM. And a new $139 4K RAM board with software

memory protect. For the ultimate in flexibility, you can design the system for low-cost multiprocessor, shared memory capability.

Find out more about the computer you thought didn't exist. Get a complete illustrated brochure describing the IMSAI 8080, options, peripherals, software, prices and specifications. Send one dollar to cover handling.

Call us for the name of the IMSAI dealer nearest you.

Dealer inquiries invited.

IMSAI

IMS Associates, Inc. Dept. B-10
14860 Wicks Boulevard
San Leandro, CA 94577
(415) 483-2093

The Concertina System

Editorial by
Carl Helmers

The often asked question of the personal systems cynic is "What on earth do people *do* with home computers?" In many ways this question is analogous to what might have been asked by automotive skeptics in the early part of this century: "What on earth do you expect people to *do* with automobiles?" Fifty to 60 years of history have answered the latter question in numerous concrete demonstrations, and one can only expect the coming decades of computer evolution to answer the former question in numerous ways.

Of course the simplest reply to the first question is "Compute!" and is as empty of content as the analogous reply for automobiles, "Drive!". Computing without a purpose is like driving without a destination, an intrinsically enjoyable pastime on occasion but hardly touching upon the set of possibilities inherent in digital computation and control. The key to a broadened perspective on the computer and its place in human activities is the concept of the application. An application for the computer is like a destination for an automotive trip. If I set in my mind the goal of driving down to Boston for an evening in Symphony Hall with Arthur Fiedler, the Boston Pops and company, my automobile has now acquired an "application." Similarly, if I decide to customize my computer system as a vehicle for editing and playing music, a very ordinary and garden variety Motorola 6800 plus memory and peripherals has acquired an "application" whenever I choose to use it for that purpose. (Like an automobile that can be driven anywhere without reason, the true general purpose computer need not be exclusively dedicated to one applications goal.)

The concept of music played using computers is an excellent focal point to demonstrate practical uses for personal computers. Here is a specific application of the computer technology for very human purposes, a concrete argument to throw at the skeptic and cynic. The choice of a musical application goal, like the decision to drive to a concert with an automobile, makes the technology come alive with human values.

The traditional concertina is a simple pneumatic acoustical instrument similar to an accordion. The pneumatic concertina is hardly a widely known or used instrument. In a hand held package with control buttons, it gives the player an ability to create a fairly rich timbre similar to a reed organ or a harmonica. Like all instruments, it requires an element of virtuosity to play at all well, but within its limitations it makes an interesting vehicle for musical expression.

The relative obscurity of the original concertina instrument, the harmonically rich nature of its timbre, noting its use by a single (good) artist in creating a polyphonic output, and most of all, falling in love with the smooth sound and etymological roots of its name leads me to propose the name "concertina system" for a musical instrument based on a personal computing system integrated with musical software and peripherals. The concept of the digitally controlled musical instrument is not new, but the technology which makes it possible at a reasonable price is as new as the whole LSI computer technology. For about the same price that you or I would pay for a virtuoso quality home electronic organ, it is possible to add a music playing peripheral to a computer system which will allow the owner to accomplish musical performance feats unheard of on an organ or traditional instruments. As of this writing, I know of two companies which are in the process of preparing products which can simply convert an existing computer system into a polyphonic synthesizer with the potential (with software) of becoming a truly playable concertina system. One company is located in Arizona and is said to be designing an Altair compatible plug-in card with several polyphonic channels of digitally controlled music output. Another company, ALF Products, 2130 Bell Ct, Lakewood CO 80215, is in the preproduction prototype stages of preparation to market a modular computer controlled synthesizer which interfaces to any existing computer by using the two programmable ports of a single PIA chip. (Most existing computers have provisions for a "PIA card" with one or more such "parallel interface adapters" together with appropriate plugs.) The ALF design has 8 fully programmable music channels with an option to add 8 less versatile "background" channels to achieve the potential for a truly orchestral sound.

There are undoubtedly additional individuals and companies working on similar systems and products which can simply and inexpensively (relative to costs two to three years ago) add a minimal concertina system capability to the typical home computer system. Readers will find more information on this subject as the products become better defined and reach the marketplace; we also expect to publish articles on the technology of computer controlled musical instruments (experimenters and potential authors: take note).

If you have a computer, you have 75 to 90% of such a "concertina system" already available. All the control and data management power needed to implement a relatively simple and quite functional polyphonic music interpreter is present in a microprocessor system using chips such as the 8080, Z-80, 6800 or 6502 with several K of applications program memory (I use 12 K myself for interpreter and text area, but it would certainly be possible to program a usable system with as little as 4 K memory.) The system also requires a video output display, ASCII text keyboard and a pair of audio cassette drives with motor control via relays and a data rate of at least 300 baud. Obtaining better mass storage peripherals such as floppy disks, 3M cartridge drives or high speed electronically controlled Philips cassette decks is of course highly desirable.

This use of the computer system, while requiring a dedicated peripheral, is completely consistent with the concept of the general purpose personal system, for when the system is not being used for music, programs with other purposes can be employed. Many typical uses require such a peripheral specific to the application; many other applications such as record keeping, calculation, text processing, mathematical and simulation games, and so on merely require the general purpose computing system composed of processor, programmable memory, text keyboard, video display and mass storage on magnetic media. The "concertina system" concept is but one of innumerable answers to the critic's question of "What do you really do with a home computer?"■

Word Gets Around . . .

The personal computing field is getting some attention as the amount of activity creates some micro ripples in the big pond of things people do. An article in the July 12 issue of *Business Week* featured Paul Terrell's Byte Shop computer store in Mountain View CA. Paul's shop is one of the largest retail outlets among the more than 250 stores coast to coast which regularly stock BYTE.

Also, Ivan Berger, electronics and photography editor of *Popular Mechanics* magazine, reports in phone conversation that he has scheduled a short feature article on home computers, their present and future prospects, in the September *Popular Mechanics*.

There have of course been several local and national newspaper stories lately. As products improve and the market expands, we should see more and more examples of public awareness of computers documented in the press and other media, a welcome trend indeed.■

Size and Finesse

Perhaps you've noted a moderate expansion in the size of your monthly mouthful of BYTE, along with the addition of some colorful spices to make each BYTE more flavorful.

Switching metaphors, a magazine such as BYTE is in many respects like a very large and complicated program design. The first concern was and is to fill a high quality technical magazine with good useful information and fun, once a month. This is the algorithm we have used very successfully and continue to use.

But, like the large program, although the basic algorithm design is not subject to major changes there are always new features, incremental improvements and parameters to adjust. Striving for the "best possible BYTE" is an ingrained part of our philosophy, where "best" is defined as serving the needs of our customers in this specialized field.

In the area of improving the product, recent increases in the size of each BYTE reflects a synergistic combination of subscriber and advertiser support. The added spice of color printing on interior pages is directly a result of support from advertisers, who make such support possible because of our readership. We're still experimenting with this new possibility of highlighting and enhancing technical articles but the presentation should continue to improve.■

And now minis too! All from a catalog, at discount prices.

Digital's Direct Sales Catalog — the first catalog to offer computer products by mail with off-the-shelf delivery — was such a success, we've come out with an expanded second edition that includes the PDP-8A, the newest member of the world's most popular minicomputer family.

We've also added the LA180 line printer and expanded the sections on microcomputers, logic modules, terminals, cabinets, connectors, accessories, and supplies from the first edition.

Of course, you still get our 5% catalog discount, plus another 4% for cash with your order. You also get a 10-day free trial period, plus our standard 90-warranty. (Since we're selling by mail, you do have to install the equipment yourself.)

If you were among the thousands who ordered our original catalog, you'll get the new one automatically by mail.

If you weren't, why not sign up now?

For your free copy, call 800-225-9480 (Mass. 617-481-7400 ext. 6608). Or write: Components Group, Digital Equipment Corp., MR2-2/M59, One Iron Way, Marlborough, MA 01752. (Catalog sales to U.S. only.)

The Computer... Versus... Hand Sent Morse Code

Lt William A Hickey USN
c/o US NSGA Edzell
FPO New York NY 09518

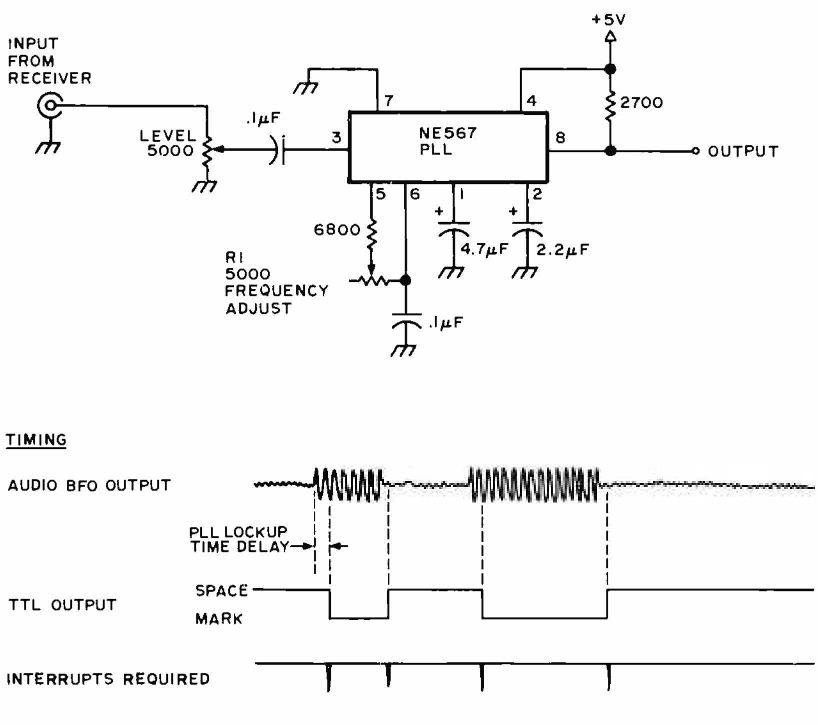


Figure 1: A suggested circuit to decode the audio output of an amateur radio receiver. The receiver produces an audio tone burst corresponding to the keyed continuous wave (CW) signal being received. This tone has a frequency which depends on the tuning of the receiver. The 567 PLL circuit is a tone decoder, which can have its center frequency adjusted by R1. The receiver output frequency and the NE 567 frequency should be the same for the desired signals. Noise and garbage (such as other stations nearby) will complicate the actual receiver output waveform.

So you've been reading all about these marvelous new microcomputers, and thinking about how nice it would be to have one that would translate Morse code for you. Well, it certainly sounds reasonable; it really depends upon what you expect from your computer. If you are expecting error free code translation under even the best signal conditions, you are in for a rude awakening.

First of all, a computer (by today's standards anyway) cannot beat or even meet the standards set by a good human operator receiving code. You say you'll concede that point? Why should it be so difficult to design a translator that would work most of the time? That's the purpose of this article! I am not trying to discourage all the code copying enthusiasts out there; I am trying to prepare you for some of the not so obvious problems you might expect.

For the purpose of this article, I will have to assume that you have somehow managed to translate the audio (code signal plus noise plus garbage) into a digital format of 1's and 0's. There are many ways to do this, but probably the most simple is shown in figure 1. This is a straightforward tone decoder using a 567 integrated circuit. (Remember that when a tone is present and decoded, the output is a zero.)

Now that you have this nice (hopefully free of noise) digital signal, what do you do with it? The answer to this one is simple.... Anything you want to do! In all seriousness, I will now branch into a discussion of hand sent Morse code characteristics. Assuming that you all know that a dot is assigned a relative time duration of (1.0), it follows that you also know that the dash is ideally (3.0), the letter space is (3.0) and the word space is (7.0). (If you are a ham, you do know that, don't you?) At this point I can safely say that the problem of translating machine sent Morse code is relatively trivial. I use the word trivial because it is really just one machine talking to another machine; the intervals are all fixed and are constant, making translation merely a matter of a table lookup assuming both machines use the same Morse code data rate.

At this point, it will be convenient to illustrate the hand sent Morse translation problem. Figure 2 is a histogram of a very short message sent with a hand key.

CQ CQ CQ DE OPERATOR NUMBER ONE

The largest single distribution in the shorter mark group has been assigned the relative time value of (1.0). Keeping in mind the very small size of this sample, you can see that there is a wide variation in mark lengths. (This variation will increase proportionately with an increase in sample size.) Space lengths are not as clearly defined, and it is very difficult to decide where the decision should be made between letter spaces and word spaces. This won't be a serious problem though, since an extra space or the lack of a space between words rarely damages the message context. The decision problem will become acute when the space interval within characters begins to spread toward the lower boundary of the space interval between characters. This will be one of the primary sources for decoding errors.

This article will not and should not specifically address such problems as: (1) gradual frequency drift from either the transmitter or receiver, (2) rapid transmitter frequency drift ("chirp"), (3) atmospheric fading, (4) noise from natural or human made sources, or (5) the presence of many other Morse and non-Morse (SSB, RTTY, etc) signals in the same receiver passband. The reason for this is simple: These factors are just too complex to be within the correction capability of simple algorithms. Remember that these problems are common to machine sent and hand sent Morse code signals.

Translating hand sent code really begins to get sticky when the sending operator gets sloppy. (He or she might send a "special" signal like --......-- [*73, a signoff greeting*] which tends to give receiving operators problems too!) Let's face it, there are a lot of really bad "fists" out there. Of course, there are some pretty good ones too, but frequently the contact you want falls into the bad group!

The reason automatic decoders are usually unsuccessful at decoding hand sent Morse code is: they are unable to adapt to the time

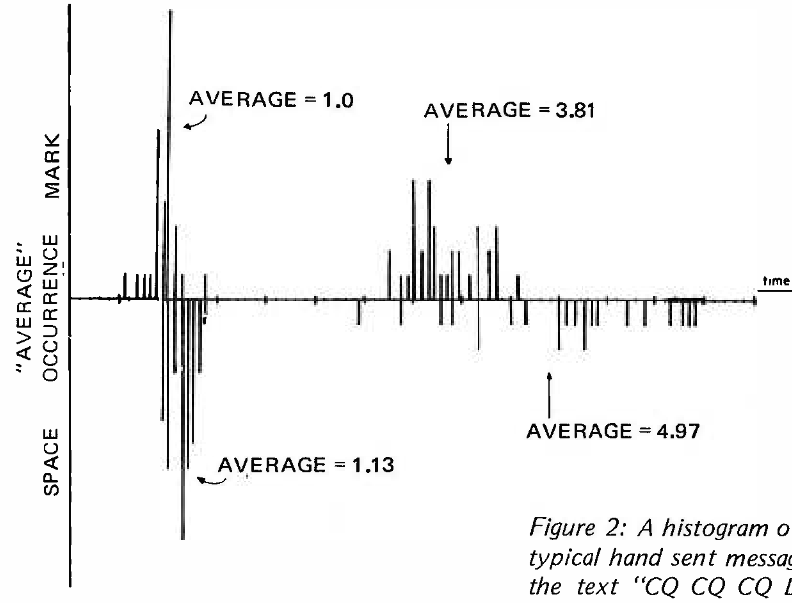
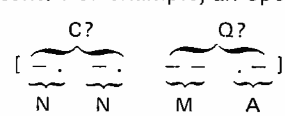


Figure 2: A histogram of a typical hand sent message, the text "CQ CQ CQ DE OPERATOR NUMBER ONE". The terms "mark" and "space" refer to the low and high TTL levels out of the detector of figure 1, respectively.

varying properties of the statistical Morse signal. (This means the signals change characteristics from time to time . . . . usually just as the machine was ready to adapt to the previous change.) These statistical changes are reflected only partly in the mark and space timing characteristics of different operators or those of the same operator over an extended time.

One of the advantages a human has is the ability to make contextual analysis on what is sent. For example, an operator might hear



and understand the signal ["CQ"]; the machine would translate NNMA (just as it was sent). Examples of this phenomenon are endless and are available on the airwaves daily.

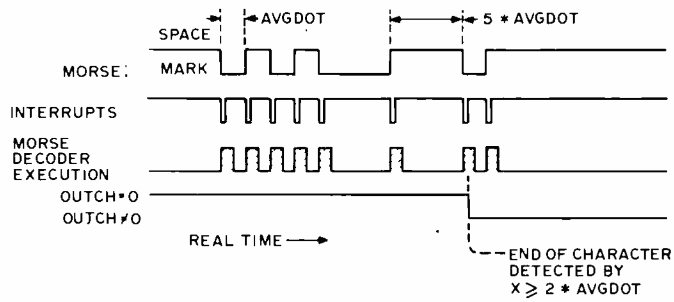
There are three primary approaches to a solution of the hand sent Morse translation problem:

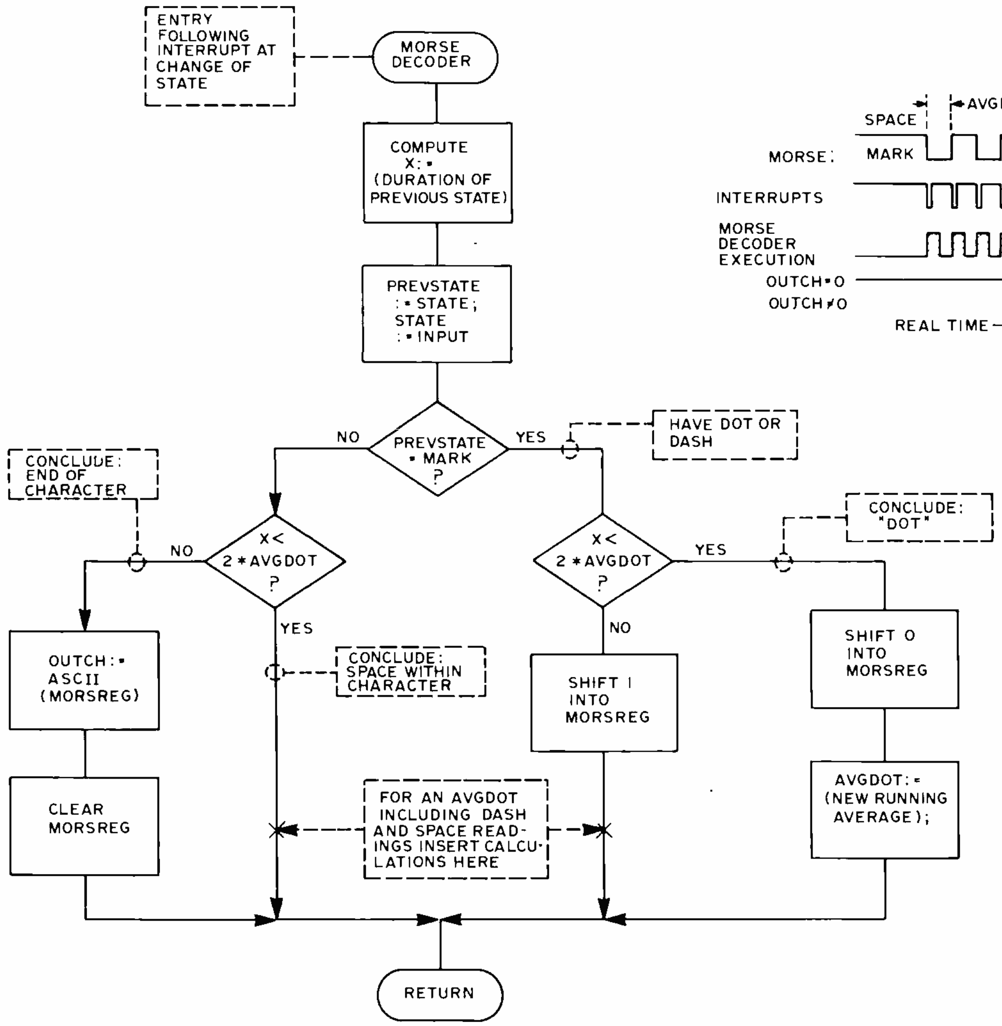
1. *Macro:* You can accumulate statistical information on a particular operator and use this data to make decoding decisions.
2. *Micro:* You can make your decisions on a mark to mark basis.
3. *Averaging:* You can compromise these two methods and come up with a hybrid algorithm.

Approach (1) requires a long sample time to develop the statistical information, and during this time the decoded output would probably be unacceptable. Even after the statistics become valid, the decoder would operate only on operators with similar

Data Definitions: 8 bit registers or programmable memory

X	Duration of the previous input state.	OUTCH	Output ASCII character temporary buffer.
AVGDOT	Running average of the previous "n" dot times.	MORSREG	Morse code pattern input shift register.
		PREVSTATE	Previous mark or space input state.
		STATE	Current mark or space input state





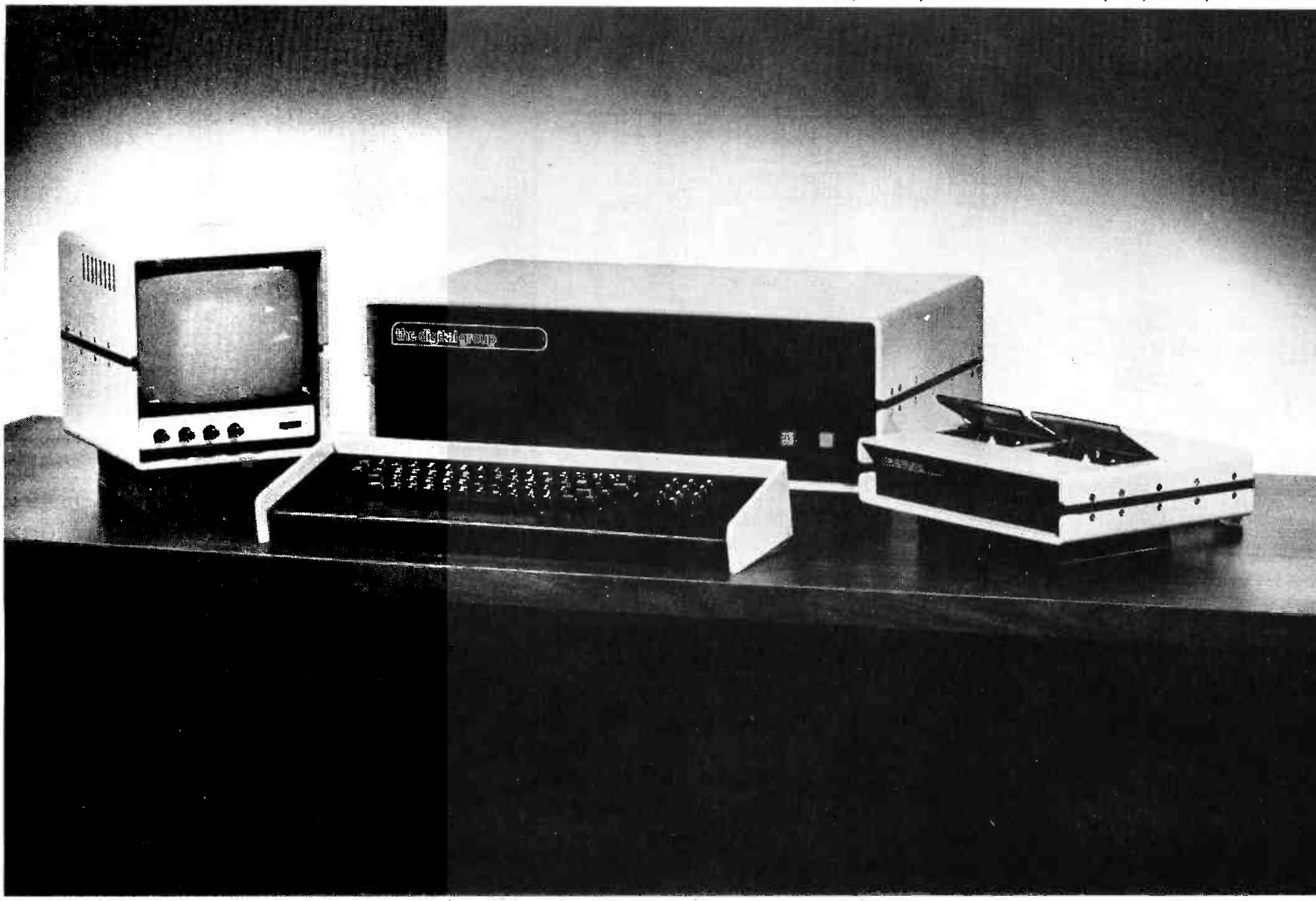
Note: This routine is intended to be executed following an interrupt generated by both the rising and falling edges of the TTL Morse code input from a detector. An application routine to receive the output of this routine must have a wait loop which continually tests OUTCH for zero. After an interrupt is completed, a non zero character will be in OUTCH and the application routine will leave its idling loop and can use the character after copying it and setting OUTCH back to zero.

Figure 3: The flow diagram of a relatively unsophisticated Morse decoder program. The adaptive features of this program are contained in the calculation of a new running average of the dot length whenever a dot is detected. The speed of the algorithm's response to a change in the keying rate of the Morse input is a function of the number of dots maintained in the running average. A more complex algorithm could take into account the nominal dash spacing of three dots as part of the average, as well as the spacing between signal elements within characters.

idiosyncracies. This is a nice idea, and it works well for individual operators; but it is not very workable for a broad collection of operator characteristics. Approach (2) is the easiest method — sometimes called the "ideal dot" method — but it is very susceptible to noise pulses and rapid code speed changes. (It tends to generate an excessive number of errors and is not really that good for decoded output.) For now, approach (3) seems to offer the best chance of working. Many individuals and commercial manufacturers have tried variations on approach (3); but they all boil down to: Sample from four to eight characters, average the lengths of the dots, and use that average to make decoding decisions. After the initial average is set, you can update the average each time a dot is detected; or you can average both dot and dash lengths and settle on a median

Continued on page 106

Cabinets clockwise from top: CPU, Dual-cassette drive, Keyboard, 9" Monitor.



The Digital Group covers up.

(Beautifully.)

For many months the Digital Group has been hard at work on the heart of our microcomputer system, insisting on quality where it counts in every product we've designed. Now, we have turned our attention to the outside and covered up . . . with a complete line of custom cabinetry that will enhance your Digital Group system for all the world to see. The result is beautiful.

Sleek and sophisticated, but rugged enough to take all the hard knocks you hand out, Digital Group cabinets are made to be used and not just admired. Extra-heavy-duty eighth-inch aluminum is utilized throughout with a special tough-texture commercial-grade paint in Computer Beige. All front panels are anodized aluminum in dark brown. Even the front panel switches are lighted.

The Digital Group offers a beautiful cover-up for each part of your system — from the CPU to the video monitor. What's more, every new Digital Group product will get covered, too, so each piece will maintain that unmistakable Digital Group image.

We're sure you've already fallen madly in love with our cover-ups, but just wait until you take a peek inside. That's where the real beauty lies.

Our video-based systems, including 8080, 6800 and the super new Z-80, are state-of-the-art, high quality and totally integrated designs. Digital Group systems are complete and fully featured and are specifically designed to be easy to use. Merely power on, load cassette and go! (And with our new cover-ups, you go in style.)

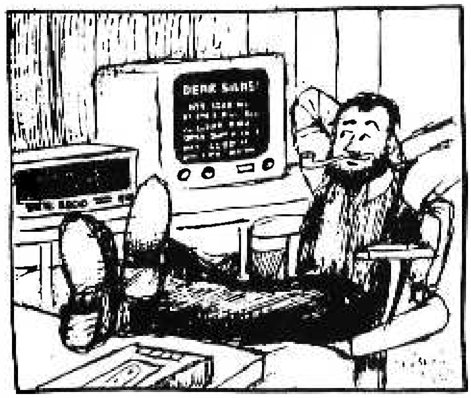
Best of all, Digital Group systems are available now. And affordable. Prices for a complete Z-80 based CPU start as low as $645, including the cover-up.

So write or call us for all the beautiful details. And then head for cover!

the digital group inc.

The Digital Group
P.O. Box 6528
Denver, Colorado 80206
(303) 777-7133

Letters



MORSE CONVERSION BACKGROUND INFORMATION

A letter from W A Hickey regarding Morse translators appeared on pages 92-93 of your July issue. For the further edification of your readers, perhaps including Mr Hickey, I provide the following additional references on this topic:

Althoff, W A, *An Automatic Radiotelegraph Translator and Transcriber for Manually Sent Morse*, NTIS AD-772 745, Dec 1973.

Ball, Edison L, *Processing of the Manual Morse Signal Using Optimal Linear Filtering, Smoothing, and Decoding*, NTIS AD-A019 493, Sept 1975.

Bedzyk, W L, *Machine Translation of Morse Code Using a Microprocessor*, NTIS AD-785 130, June 1974.

Guenther, J A, *Machine Recognition of Hand-Sent Morse Code Using the PDP-12 Computer*, NTIS AD-786-492, Dec 1973.

McElwain, D K and M B Evens, "The Degarbler — A Program for Correcting Machine Read Morse Code," *Information and Control*, March 1959.

McNaney, J T, and Richard R Tice, *System for Converting Telegraphic Code into Characters*, US Patent 2,840,637, June 1958.

Powers, B L, and F R Scalf, *The Design of a Morse-to-Teletype Signal Converter Using Integrated Micrologic Circuitry*, NTIS AD-840 255, June 1968.

Shenk, E R, and J C Phelps, *Automatic Code Signal Discriminating Device*, US Patent 2,534,388, Dec 1950.

Smith-Vaniz, W R, and E T Barret, "Morse to Teleprinter Converter," *Electronics*, July 1 1957.

Tevis, R, *Printing Telegraph Receiver*, US Patent 1,805,114, May 1931.

Thomas L A, *Morse Code Printing System*, US Patent 2,534,387, Dec 1950.

Winter, A C, *Code-Controlled Apparatus*, US Patent 2,384,513 Sept 1945.

I doubt that this list is complete; I have not been interested enough to do a really thorough literature search.

On a different topic: May I suggest that you provide the magazine name, the volume number, and the date, at the bottom of each page. Despite your predominantly hobbyist readership, this small professionalism would be useful.

E Douglas Jensen
Principal Research Engineer
Computer Systems Technology Section
Research Department
Honeywell Aerospace & Defense Group
Minneapolis MN 55413

The current issue of BYTE adds a bit to the applications literature. Thanks for sending along an excellent list of further sources.

MORE ON MAKING PC BOARDS

I would like to add a few things to James Hogenson's article on making printed circuit boards *[July 1976 BYTE, page 58]*.

Readers who wish to make their own PC boards will find that the spray resists are messy and often difficult to use. We have found that a dry film resist made by Dynachem Corporation, Santa Fe Springs CA 90670, works very well and is easy to use. The material is called Laminar and comes in various thicknesses. The one mil thickness is the best for general use. One of the nice things about it is that it is developed in a water solution of sodium carbonate (Arm and Hammer washing soda) made to a concentration of 2 to 3%. It is moderately sensitive to light and may be used in room lighting without difficulty. It is easily applied by heating the PC board and rolling it on with heating. A hard rubber roller works the best. These are available from art supply

stores. The Laminar also has protective plastic coatings on both sides. The soft, flexible coating is removed before laminating to the clean copper clad. The inflexible coating is removed before developing. The E I du Pont Company also has a similar type of resist material.

The PC boards must be very clean before resist is put down on the copper surface. A dip in dilute hydrochloric acid or muriatic acid followed by a scrubbing with Ajax or steel wool will prepare the surface. Hogenson's photo 9 looks like the result of resist put on a dirty board.

Printed circuit boards shouldn't be drilled with regular steel drills. They will wear out quickly and will leave ragged holes. Try carbide drills made for PC drilling. Most PC material houses should stock these. The type with an eighth inch shaft will fit in most tools.

Bishop Graphics Inc, 20450 Plummer St, Chatsworth CA 91311, has a wide variety of PC layout and tape-up aids. I would suggest that readers get a copy of their catalog since the last 32 pages are a technical manual on PC layout and related techniques.

I hope this will help your readers to improve their PC board technique.

Jonathan A Titus
Tychon Inc
POB 242
Blacksburg VA 24060

ON BLANKS, CHARACTERS AND WOMEN IN COMPUTING

If we credit Charles Babbage with the first programmable digital computer design, we should likewise consider Lady Lovelace as the first programmer.

I am glad to see BYTE is developing as a stable but flexible medium for the computer hobbyist. There are, however, three points which should be addressed early in your history:

1. Invent a "blank character." Variable spacing for uniform column width is fine for reading, but poor for showing significant blanks. Programmers have long used special symbols to represent spaces or blanks, much as zeros act as placeholders for Arabic numerals. Examples are the lower-case b with overstruck slash or dash, and a square-cornered U. If no special character is available, perhaps just a lower-case b would do (eg: "LIMb+1" to show a blank as necessary) [*Only necessary in the limited context of character text string examples . . .* CH]
2. Publish your character set. Testing the character set is often one of the first acts performed with a new medium (such as a keyboard, video display, or printer). It appears that you do not have, for example, Greek letters, although you have quite a variety of fonts and styles. (This suggests using a different typeface to spell out symbols that may be single characters in the original, such as THETA or UP-ARROW). Many of us are no longer limited to 47 computer symbols, but use 128 character ASCII. [*Our text character set is published by IBM, in its literature on Selectric Composer balls; for computer graphics we generally assume 7 bit ASCII unless noted.*]
3. While it is awkward in English to avoid masculine pronouns (he, him, etc), I do think we should try to avoid the masculine assumption about readers (eg: having a wife). This is a new field, which is developing at a time in our history when women (and men) are outgrowing their traditional roles. I think we can expect to see a gradual increase in the number of women interested in computers, and should encourage the trend.

In this light, it may be relevant that if we credit Charles Babbage with the first programmable digital computer design, we should likewise consider one of his chief advocates, the Lady Lovelace, as the first programmer. She wrote instructions for setting up the Analytical Engine to perform certain calculations. (This was, of course, working in the abstract, as the machine was never completed.)

Zhahai Stewart
POB 1637
Boulder CO 80302

As to the last point, we're all for it. But it is a fact that most BYTE readers are male. Where is the other 50% of the human race in computing? As a rule, we try to keep things relatively *free of stereotypes in the hopes that the other 50% will start finding out about the wonderful attractions of computers and computing. . . .* CH

IDENTITY CRISIS

This is just a short note to say that I am enjoying BYTE and to offer a suggestion in the form of a question:

"What do we call ourselves?"

Radio amateurs call themselves hams and I am sure other people have other names for them. However, in BYTE to date, I find that writers are grasping for words to use to describe the computer hobby. Perhaps you

could run a contest of sorts to promote a name for those active in the small computer hobby.

Looking through some past BYTEs I find words such as *microists, kluge, hacker, amateur computer, digital, analytical engine, cyber(nuts)* that may be altered, adapted or crossed to coin some new word to describe a computer hobbyist. Then we can say:

"Hi, I am a ?????????????????????"

Do you suppose this will help people understand what we are up to?

Bryan Patterson
Box 1726
Port Elgin
Ontario CANADA NOH2CO

Well, if we wanted to sound self aggrandizing, we could of course suggest "Byters" as a term. Actually, in spite of the negative connotations in computer science circles, I [CH] *tend to prefer the traditional term "hacker." At the start of amateur radio, "ham" as used previously also had somewhat negative connotations. (I make an etymological assumption here that the term as used in radio circles evolved from the tendency to "ham it up" on the air as in the usage of "ham actor.") I like hacker as a term for the serious amateur computer nut (who is also typically professionally involved as well) because it has implications of digging into the subject matter and really learning it at multiple levels of detail. The "compleat modern hacker" is the renaissance man (oops . . . person) of computing.*

WHO SAYS THEY AREN'T?

With respect to BYTE covers, you blew it. The phrase "Computers – the world's greatest toys" told it like it was and still is. Truly, the only difference between men and boys is the price of their toys.

A co-worker commented, "If computers are the world's greatest toys, then are programmers the world's biggest kids?" How can I argue with logic like that?

Julius T Marinaro
725 Cricklewood Dr
State College PA 16801

VIDEO TAPE AND COMPUTERS?

I was very pleased to find out about the existence of your journal. I have had an interest in small system uses for several years. At present I am working with video tape systems within a school district in Flint Michigan. I am interested in finding various small computing systems to use in video tape editing and special effects for CRT displays.

Enclosed is my check for a three year subscription to BYTE. I am looking forward to my subscription. Also I would like to ask for your assistance in answering two questions. I would like to find out if I could obtain back issues of BYTE since it was first published, and I would also like to find out if you can give me any information on the use of small computing systems with video tape systems. Any information that you can give in these areas would be greatly appreciated. Thank you for your time and trouble.

William D Wolverton
10320 Henderson Rd
Otisville MI 48463

September, October and November 1975 BYTE back issues are sold out at present, as is May 1976. Remaining back issues are now being serviced at a price of $1.50 per copy, plus 25¢ for postage and handling. Send in your requests, but send no money with your request. If we have what you're looking for, we'll bill you for what we ship.

As to the use of computers with video tape applications, it sounds like an excellent use. However, we have no articles in house on the subject . . . . yet. Perhaps you'd like to write about your results.

HELP!

I have some surplus ICs from our local IBM factory and would like to identify them. I hope you can help me. I have three types:

2709400 JUQ V 721304 7324FQ	M̂i
2709401 Q Korea 721304 7432FO	M̂i
2709170 JUK V 721186 733180	M̂i

They are all ceramic chip with 14 pins gold plated in each side and in the top a gold square with the numbers and a ground line to pin one like in the MOS devices.

Jose Vincente
Caiza Postal 764
13100 Campinas
S P Brazil

Rickey's tackling the SDK-80 microcomputer kit for his next science project.

Rickey likes soccer, lizards, hot fudge sundaes, skateboards and microscopes. He can't decide if he'd rather be Franco Harris, Bobby Fischer or Jonas Salk.

When his Dad brought home the Intel SDK-80 microcomputer systems kit, Rickey helped him put it together. It took only four hours. Everything was there. The 8080 CPU, RAM, PROM, programmable, I/O, a printed circuit board with all those capacitors and resistors and the other things that go with it. The best part was the instruction manuals. Every step was clearly explained. It was easy. The programming part looked especially interesting. So simple. Just imagine talking to a computer.

The big thrill came on Saturday when they went to his Dad's office to use a terminal. When they connected the SDK-80 to the teletypewriter they got a printout. That was exciting. Within an hour they were talking to the computer, then inventing games. They stayed all day.

Now Rickey is building a microcomputer of his own. He may be the first kid on his block with his own computer. Thanks to a $350 low interest loan from his Dad.

If you're interested in being the first on your block to have a microcomputer, contact your Intel distributor: Almac/Stroum, Component Specialties, Components Plus, Cramer, Elmar, Hamilton/Avnet, Industrial Components, Liberty, Pioneer, Sheridan, or L. A. Varah.

Microcomputers.
First from the beginning. **intel**®

3065 Bowers Ave., Santa Clara, California 95051.

A BIT OF CIRCULATION

Attention: Circulation
Dear Ms Luhrs:

Thank you very kindly for making available the lifetime subscription to BYTE magazine which I won at the First World Altair Convention in New Mexico. Of all the door prizes given, I firmly believe I won the best. It was generous of your firm to make it available. I would appreciate your conveying my gratitude to Mr and Mrs Peschke and Mr Helmers.

Since I already have a subscription to your fine magazine, I am presently receiving two copies. I am passing one along to non-subscribers and hopefully it will generate additional subscriptions.

I commend you on the many fine articles which have appeared in past issues. Being a novice in hardware applications, I particularly look forward to tutorial type hardware articles.

Gene Straub
5723 Shasta Cir
Littleton CO 80123

THE IEEE 488 BUS

Regarding your comments about a standard interface for microprocessors, etc, on page 96 of the April issue: I imagine you are familiar with the IEEE 488 which is being used by HP, Tektronix, Fluke and many others to interface microprocessors, calculators, disks, cassettes, DVMs, counters, etc. 488 is not as complex as the standards document would lead you to believe. It is achieving rapid acceptance because it is well suited to microprocessor manipulation. For various reasons it may not be ideal for personal computing, but it seems to be an excellent starting point. One drawback is that HP has the handshake patented. Their license fee is quite reasonable for some organizations but perhaps not for hobbyists. Other drawbacks include the question of common availability of the connector, etc. However it would be nice if there could be some degree of commonality between 488 and any hobbyist standard that might evolve.

Bob Huenemann
4209 Armand Dr
Concord CA 94128

Yes, 488 might be a good place to start. For those unfamiliar with the issue, the full name is IEEE Standard Digital Interface for Programmable Instrumentation, *published by the Institute of Electrical and Electronic Engineers, Inc, document number IEEE Std 488-1975. Quoting from the IEEE standards document, "The Hewlett-Packard Company has assured the IEEE that it is willing to grant a license under these patents on reasonable and nondiscriminatory terms and conditions to anyone wishing to obtain such a license." To obtain a copy of the 80 page standards document write IEEE Service Center, 445 Hoes Ln, Piscataway NJ 08854.*

TEXT PROCESSING OUTPUT VIA CONVERTED TYPEWRITERS?

Jim Lang's letter in the August BYTE on hard copy and IBM Selectric typewriters aroused a responsive chord. I too have been interested for some time in using the ubiquitous Selectric typewriter to obtain *high quality* hard copy. It seems to me that anyone familiar with both Selectrics and Teletypes would prefer the former — half again higher speed, much higher print quality, and both upper and lower case. The clincher would seem to be that most XYLs would look more favorably on the idea of spending half a kilobuck or so on something that can also be used as a good typewriter instead of something that can't.

As some readers are probably aware, there is a commercial firm that markets a (rather expensive) applique that attaches to a standard Selectric and makes it into a terminal. (Tycom Systems Corp, 26 Just Rd, Fairfield NJ 07006.)

In this connection, I'd like to point out that the US Patent Office is a veritable gold mine of technical information that can be obtained for a very modest price. For example, the patent documentation covering the Tycom system consists of 26 pages of diagrams and 50 fine print pages of descriptive text. An appreciable fraction of this consists of a *very* detailed description of the internal workings of the Selectric typewriter — it seems to collect details from the myriad of IBM patents on the Selectric into a single place.

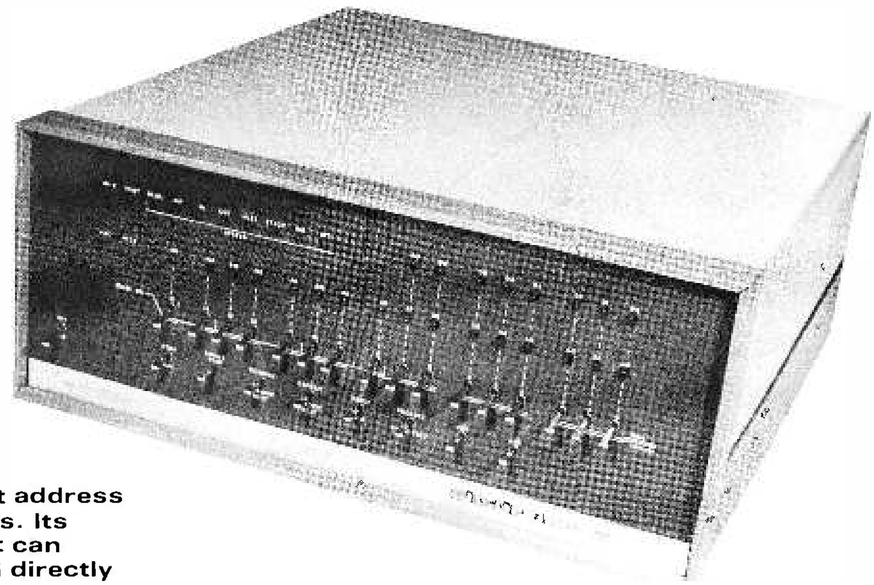
A copy of the printed version of this (or any) US patent can be obtained by specifying the patent number (#3,453,379 for the one mentioned above) and remitting 50 cents to "Commissioner of Patents, Washington DC 20231."

Obviously one cannot duplicate a patented item in making a product for sale. Nor, in this case, would one want to, since with new ICs and components, the circuitry involved is verging on obsolescence. Never-

Continued on page 136

altair™ 8800a

TECHNICAL INFORMATION



The Altair 8800a is a parallel 8-bit word/16-bit address computer with an instruction cycle time of 2 μs. Its central processing unit is the 8080 LSI chip. It can accommodate 256 inputs and 256 outputs, all directly addressable, and has 78 basic machine instructions. It is capable of directly addressing up to 65,000 bytes of memory.

As well as the LSI chip, the CPU board contains the two-phase clock, status latch, buffers and the various lines going to the bus. (The buffers are tri-state devices.)

The CPU contains six general-purpose registers, P counter, arithmetic unit, accumulator, stack pointer, instruction decoder, and miscellaneous timing and control circuits. The arithmetic unit contains the circuitry required to perform arithmetic in both decimal and binary forms. The stack pointer defines the current address of the external stack, which resides in memory. The stack is used to service interrupts and provides virtually unlimited subroutine nesting. The instruction decoder decodes the instructions and sets up the various registers, gates, etc., in the CPU for proper functioning.

There are 36 LED status indicators on the front panel, 16 of which are used for the address bus, 8 for the system status latches, and 8 for the data bus. The four remaining LEDs are used for indicating memory-protect, interrupt-enable, system-wait and hold status. Address line inputs A0 through A15, data lines D0 through D7, and the various status lines originate on the CPU board.

The front panel control board contains the circuitry for interfacing between the control switches located on the front panel and the CPU. In addition to the interconnections to the actual processor, this board accepts memory address switches A0 through A15 (also on the front panel). The first eight of these switches (D0 to D7) are used to put data into the CPU.

The front panel logic permits the following functions: STOP—stops the processor immediately after it completes the current instruction; RUN—starts the processor at the current address; EXAMINE—causes the data stored at the location (set by the switches) to be displayed in binary by LEDs; EXAMINE NEXT—steps the P counter once and displays the word stored at the next location; DEPOSIT—causes the information preset by the switches (A0-A7) to be stored in memory; DEPOSIT NEXT—steps the P counter and loads the memory; SINGLE STEP—steps the program one machine cycle; RESET—clears the CPU and sets up a starting address of 0; PROTECT/UNPROTECT—allows selective write protection of blocks of memory. When a block of memory is protected, it is impossible to write over that block, but its contents can be read out.

With proper adjustments, any memory speed can be used in the 8800a computer, although memory access time must be 500 nanoseconds or less if it is to be run without wait states. In addition to semiconductor RAMs, the processor will also service ROMs and PROMs.

NEW FEATURES

POWER SUPPLY

The power supply provides three voltages to the 8800a bus: +8V pre-regulated at 8 amps; +15V at 500mA; −15V at 500mA.

FAN

A fan has been mounted on the back panel of the 8800a to provide cooler operating temperatures.

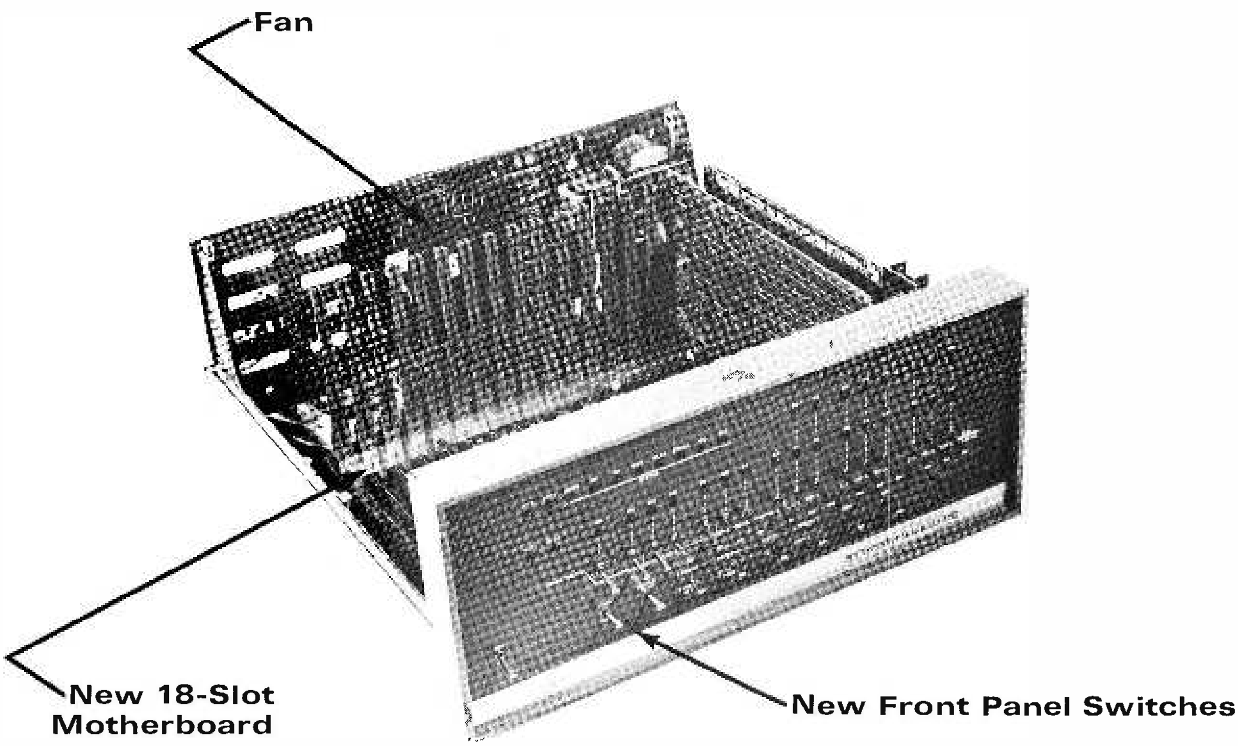
18 SLOT MOTHERBOARD

The four-slot expander cards in the Altair 8800 have been replaced with a single-piece 18-slot motherboard. The 18-slot motherboard contains the 100 solder lands that comprise the 100 pin bus.

FRONT PANEL SWITCHES

The front panel toggle switches have 50% longer handles that are flat (instead of round) for easier use.

An assembled Altair 8800a may be ordered with six, twelve, or eighteen sets of edge connectors. The Altair 8800a kits include an edge connector with every plug-in module purchased.



The four boards, along with the power supply, mount in an 18" deep x 17" wide x 7" high (45.7 x 43.2 x 17.7-cm) metal cabinet.

SPECIFICATIONS

Number of Boards	Up to 18
Microprocessor	
Model	8080A
Technology	NMOS
Data Word Size, Bits	8
Instruction Word Size, Bits	8
Clock Frequency,	2MHz
Add Time, Register to Register, Microsec. Per Data Word	2
Number of Instructions	78
Input/Output Control	
I/O Word Size, Bits	8
Number of I/O Channels	256
Direct Memory Access	Optional
Interrupt Capability	Std. one level
Vectored Interrupt (8 priority levels)	Optional
Software	
Resident Assembler	Yes
Cross Assembler	No
Simulator	No
Higher-level Language	BASIC
Monitor or Executive	Sys. mon.; text edit.
Software Separately Priced	Yes

mits
2450 Alamo S.E. Albuquerque, New Mexico 87106

A Ham's Application Dreams

W J Hosking W7JSW
8626 E Clarendon
Scottsdale AZ 85251

I'm writing this article for a selfish motive. I want to build some things and the construction articles aren't here yet, particularly in state of the art.

I just finished a television typewriter (TVT) which has alphanumeric character generation and storage capacity plus capability of serial or parallel ASCII (plus control) interface. Now that it is finished, I want more than a plaything. I would like to have the following capabilities:

a) Keyboard Morse code (CW) transmit and receive encoding with CRT display.
b) Keyboard RTTY transmit and CRT RTTY read.
c) Alphanumeric slow scan TV transmit and receive.
d) Computer terminal operation with a telephone coupled to a timeshared computer system.

Description

My envisioned system is shown in figure 1. Let's tackle that drawing block by block, considering the TVT and CRT to already exist and applying the constraint of a minimum hardware (low cost) implementation.

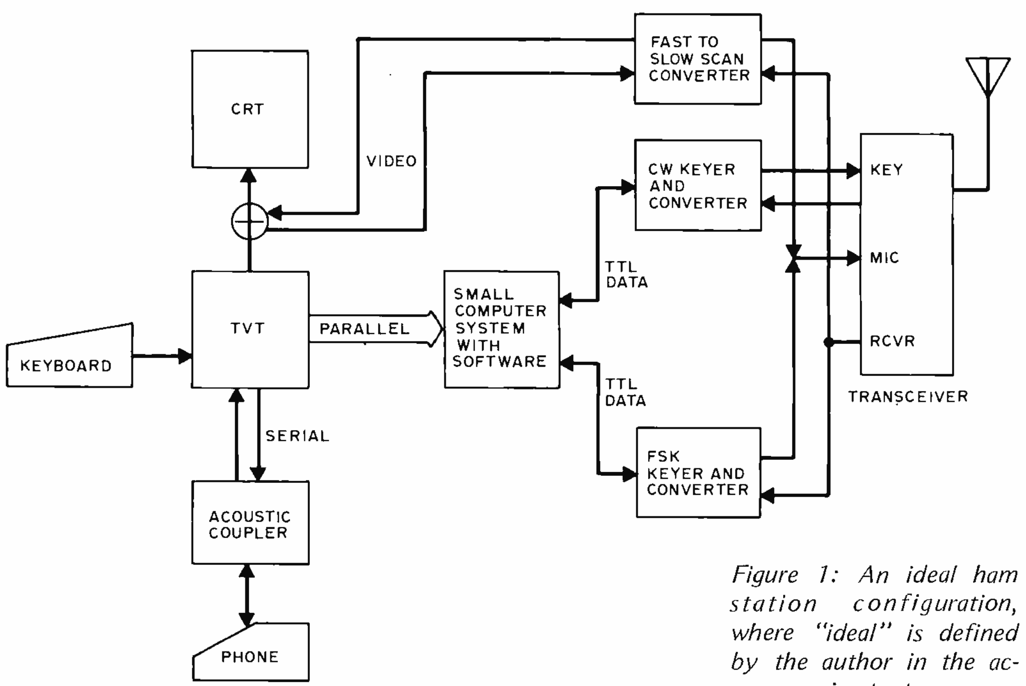


Figure 1: An ideal ham station configuration, where "ideal" is defined by the author in the accompanying text.

Morse (CW) Transmit and Receive

Looking back through previously existing magazines on electronics and amateur radio, I have found several articles on Morse keyboards, most using bulky diode matrices. The literature on Morse code readers is much harder to find, and as this is written I found nothing approaching state of the art.

It would seem that, with the rapidly dropping prices of microprocessors and their associated memories and peripherals, the way to go on this project would be a computer using minimum hardware implemented with an LSI microprocessor. I have presented a very basic approach to such a system in figure 2. All that would be required from the hardware standpoint would be the microprocessor, a read only memory for permanent program storage, a programmable random access memory for working storage, and a peripheral interface adapter (PIA) for input and output. Some other items such as power supplies and clocks are also required. I would hope that the microprocessor interface unit cost could be kept under $100.

Now for the required design work. This black box I am planning would have to accept a seven bit ASCII code (serial or parallel) plus a data present strobe. The box would convert the input character to serial Morse code and output the code over some suitable keying circuit to the station transmitter. Since we are talking about a small computer, the Morse code speed desired could be ordered by an appropriate input code sequence from the keyboard. The box then has to send a character or flag back to the TVT to say it is ready for a new character.

The reverse or receive mode is the Morse to ASCII conversion. However, here a special conversion device will be needed to change the audio out of the receiver to some kind of signal that the computer can recognize and convert. The software which drives the converter must also recognize intercharacter versus interword spacing and provide space characters to the TVT where required in the received text.

The end result would be Morse code sent by the keyboard through your transmitter and the Morse code heard by your receiver being displayed on the TVT CRT. This is a job for both hardware and software designers. [*See the articles elsewhere in this BYTE for technical details* . . . . CH]

Radioteletype (RTTY) Transmit and Receive

The same basic microprocessor described above could be used to provide the radioteletype function instead of or, with more

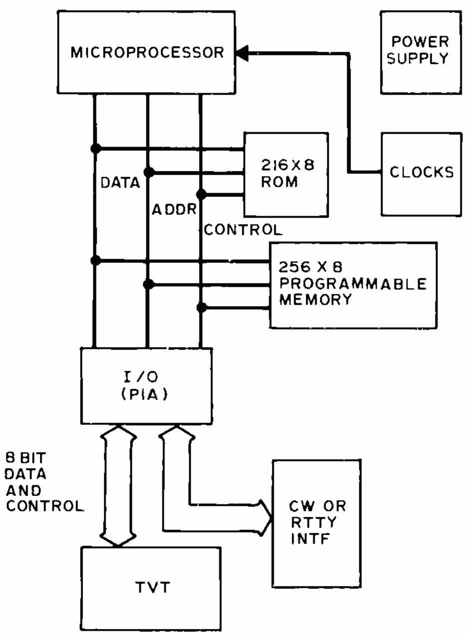


Figure 2: For a dedicated "black box" ham radio data processor, the use of a microprocessor system with as much as 2 K by 8 bits of program ROM with perhaps 256 bytes of programmable memory for data buffering. [Exact amounts of ROM will depend upon the complexity and features of the software loaded; 2 K should be reasonable for Morse and radioteletype support . . . . CH]

Editor's Note

Here is a short article on a theme of "wouldn't it be nice to have X" where X is defined as some automation applications for the amateur radio station. It is timely, in the context of this month's Morse code theme, in that one of the author's application goals is detailed in several different technical articles in this issue. There's still plenty of room for further explorations of computer application to amateur radio technology of course . . . . CH

ROM, in addition to the Morse code function.

Here, the transmit conversion would be from parallel ASCII (or serial) to serial Baudot at a transmission speed programmed into the computer. The system could easily include niceties such as automatic line feed, etc. The Baudot output would be serial signals sent to an FSK transmit terminal unit. When the FCC finally gets around to approving ASCII on the air, a simple ROM change would reconfigure the hardware to reflect this improvement.

On the receive side, a standard RTTY terminal unit would be used to convert the received FSK signal to a serial binary signal for the microprocessor. This keying signal would be routed to the processor which would convert it to either serial or parallel ASCII, whichever your TVT or terminal interface requires.

This system would work just as a regular teleprinter does, except that the received copy would be on a CRT instead of hard copy on a printer. For those desiring hard copy, a printer can easily be interfaced to the computer system. The development here is mostly software since really good terminal unit and keyer designs are readily available.

Slow Scan TV

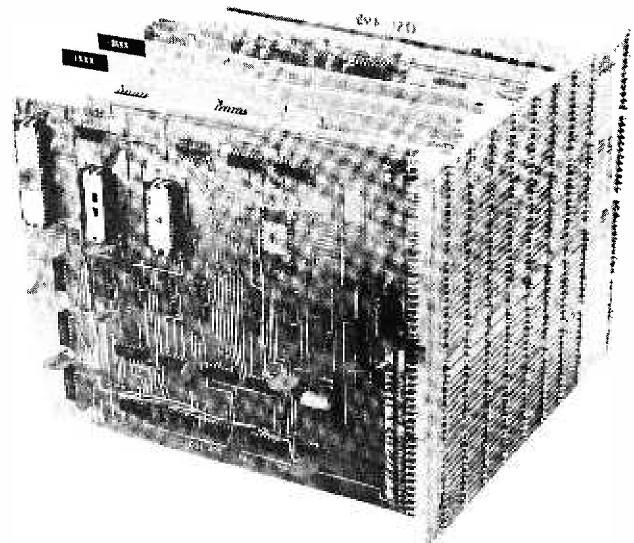
This is another area eagerly awaiting new developments. The TVT already has fast scan composite video as an output and the CRT involved accepts composite video. The problems and areas for new development are mostly in the area of conversion from fast scan to slow scan and reverse. It is highly likely that the microprocessor box we have already discussed can do at least part of this job for us. I think this one is really ripe for new breakthroughs.

Computer Terminal Operation

This is probably the easiest task. There are those out there who, had they the terminal, could make use of one or another timeshared computer system. This use requires coupling in one way or another to a telephone line. The easiest way to do this without angering Ma Bell is with an acoustic coupler.

Conclusion

As I said earlier, this was an idea article and now I'm going to sit back and eagerly await the neat ideas generated by all you experts out there. There's room for much development in both hardware and software. If you have designed something to do one of these jobs, then publish it for the rest of us.■



Add This 6800 MORSER to Your Amateur Radio Station

Robert D Grappel
148 Wood St
Lexington MA 02173

Jack Hemenway
151 Tremont St
Boston MA 02111

A great many amateur radio operators find Morse code operation a nuisance. It isn't easy to develop proficiency in copying code, and it often seems that those stations one wants to work are just those whose operators send code too fast to readily copy. There must be hundreds of Morse coding aids developed over the years, ranging from mechanical keys and paper tape transmitters to fancy code memories and typewriter like automatic transmitters. Nearly every advance in electronic componentry has spawned a new series of Morse code aids. Nearly all have been designed to help the operator send more effectively; the problem of reception has been more difficult to solve. Some complex circuits have been devised which can copy code, provided that it follows strict timing requirements. Truly general code followers, circuits which can copy code with performance approaching that of a skilled human operator, are rare. They are very complex, using dozens of integrated circuits, large diode matrices, etc. The recent advent of inexpensive yet powerful microcomputers can make the dream of a relatively simple yet very general code follower possible. Why a computer? First of all, it allows one to develop and improve an algorithm by simply changing program, instead of rebuilding complicated circuitry. Second, since the computer is not restricted to running only the Morse programs, one can use the computing power for any number of other uses, limited only by the operator's creativity. This article describes a code following computer program as implemented on a Motorola 6800 microcomputer. It can copy any code speed from 3 to 60 words per minute, and can adjust to the irregularities

Development of MORSER

The program for MORSER shown in this article was produced by a relocating assembler designed and written by Jack Hemenway and described somewhat humorously in the August BYTE article "Jack and the Machine Talk" [page 52] *by authors Grappel and Hemenway. The relocatability feature allows one to put assembled code anywhere in memory without reassembly, a feature which is most useful for building large programs. One drawback of this is that relocatable addresses (denoted "R" in the listing) are always two bytes, so the programmer cannot generally make use of Motorola's direct addressing mode, which requires addresses to be in the base page, the first 256 bytes of memory address space. Since MORSER was programmed so that both data and program code are relocatable, it is not the most compact form in which the algorithm could be expressed on a 6800. If one rewrote the program to keep all the variables in the first page of memory and used direct addressing wherever possible, about 75 bytes of storage could be saved.*

Photo 1: Author Robert Grappel, shown at the console of Jack Hemenway's computer system with a hand held switch used to test Morse code inputs to MORSER during the development of the program.

of hand sent code. A minimal amount of external hardware is needed, and the program only takes about 600 bytes of memory. The algorithm can be converted to run on almost any 8 bit microprocessor. Since you are still reading this, you are hooked. Let's begin to dissect the program.

MORSER consists of five segments: initialization, decoding, delay timer, sampler, and terminal driver. In the program listing, lines 001-068 are initialization, lines 069-200 are decoding, 200-213 form the delay timer, 214-250 form the audio sampler, and lines 251-277 drive the output terminal. Each segment will be described in turn.

The major function of initialization is to define the variables in the program and to give them appropriate initial values. The operating system of Jack Hemenway's machine performs some of the initialization automatically at loading time, such as clearing the peripheral registers and setting the stack pointer. If the program is to be run on a system without these features, then statements to perform these functions must be added to initialization. The values of DTIME and MAXCNT must be set, based on the computer running the program. DTIME adjusts the program timing to the processor cycle time, and MAXCNT adjusts the terminal driver to the data rate of the terminal in use for output. These values are not very critical, and the program comments list typical values for these constants. As assembled, MORSER assumes a 6820 PIA at addresses 8040-8043 (hexadecimal). The peripheral interface can be relocated to suit the particular hardware configuration in use. Only one input bit is needed; the rest of the PIA may be used for other functions. The listing also shows that an external subroutine, OUTCRT, is being used to drive a CRT terminal for output. This program is part of Jack Hemenway's system. The user of MORSER must provide a suitable routine for his or her own system. For example, the OUTEEE routine in Motorola's MIKBUG will work. The idea is that some way must be found to take a character from the A accumulator and place it appropriately on the output device. An automatic carriage return and line feed is required, as MORSER does not count the characters in a line.

Initialization also sets up the decoding table DECTAB. The ordering of this table is the heart of MORSER. The ASCII representation of a character is placed in DECTAB at an offset generated as follows: Generate a byte with a binary 1 for every dot and 0 otherwise; generate another byte with a binary 1 for every dash and 0 otherwise. For example, the letter A (Morse • —) generates 00000010 and 00000001 respectively. Multiply the dash byte by two with a left shift and add the bytes. The result is the character offset. Using this algorithm, it is seen that A is at an offset of 4. All other Morse characters are generated in the same manner, and the rest of the table is filled with blanks. It is the function of the remainder of MORSER to convert the incoming audio signal into offsets into DECTAB, and to transfer the character representations found there to the output device.

MORSER decides which inputs are dots, dashes or word spaces by sampling the audio

Listing 1: The MORSER program, as assembled for the 6800 at location 0000 hexadecimal using Jack Hemenway's assembler. The program is written in a relocatable fashion, so no data references are made to page 0 of the 6800 address space. In Jack's assembler, the column immediately following the hexadecimal code output sometimes has the letter "R" in it. This indicates a reference to a relocatable symbol.

input at intervals. The delay timer section controls the period of the sampling. This tiny section of code (only six instructions) actually consumes more than 75% of the running time of MORSER. In fact, the time spent in all other parts of MORSER is considered negligible in the design. A rough "rule of thumb" states that one word per minute of Morse code is equivalent to one dot length per second. All other code elements have lengths nominally equal to integer multiples of the dot length. MORSER is designed to sample each dot length time unit four times. Since the range of code speeds is 3 to 60 words per minute, this implies that the sampling period should range from 100 to 4 ms. The delay timer is adjusted by DTIME to count time in millisecond intervals. The variable TIMER determines how many milliseconds will be spent in the delay loops. This time is roughly the sampling period, since the longest path through the rest of the program is at most 0.3 ms. TIMER is adjusted in the decoding section to suit the speed of the code being processed. The formula 250/TIMER gives the approximate input code speed in words per minute, after the program has been running for several characters and has adjusted itself to the code.

The sampling section repeatedly tests the status of the audio input signal against its

```
0000 7E 00B5 R BEGIN JMP   INIT      BEGIN INITIALIZATION
                     *
0003 72F0        OUTCRT  EQU   $72F0  SYSTEM CRT HANDLER
0003 8040        PIA1    EQU   $8040  PERIPHERAL LOCATIONS
0003 8040        DDR1    EQU   $8040
0003 8041        CRR1    EQU   $8041
0003 8042        PIA2    EQU   $8042
0003 8042        DDR2    EQU   $8042
0003 8043        CRR2    EQU   $8043
                 *  AUDIO INPUT ON SIDE 1, BIT 0
0003 0020        BUFFT   RMB   32     DECODED OUTPUT BUFFER
                 *
0023 00          COUNT   FCB   0      OUTPUT TIMER
0024 FF          DFLAG   FCB   $FF    "DATA-READY" FLAG
0025 42          DTIME   FCB   66     DELAY LOOP TIMER
                 *   ADJUST DTIME FOR PROCESSOR CYCLE TIME
                 *   DTIME=165 FOR 1.0 USEC PROCESSOR
                 *   DTIME= 88 FOR 2.0 USEC PROCESSOR
                 *   DTIME= 66 FOR 2.5 USEC PROCESSOR
0026 FF          LDATA   FCB   $FF    LAST DATA
0027 FF          LMARK   FCB   $FF    LAST MARK TYPE
0028 0C          LMLEN   FCB   12     LAST MARK LENGTH
0029 09          LTRSP   FCB   9      LETTER-SPACE LENGTH
002A 14          MAXCNT  FCB   20     TERMINAL CONSTANT
                 *   MAXCNT=40 FOR 110 BAUD OUTPUT TERMINAL
                 *   MAXCNT=20 FOR 300 BAUD OUTPUT TERMINAL
                 *   MAXCNT= 8 FOR FASTER TERMINAL
002B 00          MLEN    FCB   0      MARK LENGTH
002C 00          OLDSTA  FCB   0      OLD MARK/SPACE STATUS
002D 00          OUTXCV  FCB   0      BUFFER POINTER
002E 0C          SPLEN   FCB   12     SPACE LENGTH
002F 00          STLDOT  FCB   0      DOT REGISTER
0030 00          STLDAS  FCB   0      DASH REGISTER
0031 01          TCNTR   FCB   1      TIME COUNTER
0032 14          TIMER   FCB   20     CLOCK ADJUST VALUE
0033 18          WDSP    FCB   24     WORD-SPACE LENGTH
0034 00          WDSXCV  FCB   0      BUFFER POINTER
                 *
                 *   DECODING TABLE GENERATED AS FOLLOWS:
                 *   TAKE MORSE CODE FOR CHARACTER...
                 *   GENERATE 6-BIT VALUE, 1=DOT, 0=DASH
                 *   GENERATE 6-BIT VALUE, 1=DASH, 0=DOT
                 *   SHIFT SECOND VALUE LEFT ONE PLACE
                 *   ADD VALUES...THE RESULT IS THE OFFSET
                 *   OF THAT CHARACTER IN DECTAB
                 *   BLANKS OCCUR WHERE THERE IS NO VALID
                 *   CHARACTER
                 *   REF: JORGENSON, HAM RADIO, MAY 72
                 *
0035 20          DECTAB  FCC   ' '
0036 45                  FCC   'ETIANMSURWDKGOHV'
0046 46                  FCC   'F LJP BXCYZQ  54'
0056 20                  FCC   ' 3   2       16 '
0066 2F                  FCC   '/      7   8 90 '
0076 20                  FCC   '   '
0079 04                  FCB   $04    END OF TRANS. CHAR.
007A 20                  FCC   '    '
007E 20                  FCC   '  ?         .   '
008E 20                  FCC   '               '
009E 3B                  FCC   ';   (     ,     : '
00AF 20                  FCC
                 *
                 *   CONFIGURE PERIPHERAL INTERFACE
                 *
00B5 86 04       INIT    LDA A #$04
00B7 B7 8041             STA A CRR1
                 *
                 *   END OF INITIALIZATION SECTION
                 *
                 *   DECODING ALGORITHM HAS FIVE RULES:
                 *   1. IF NEW MARK IS AT LEAST TWICE
                 *      THE LAST ONE, THEN IT IS A DASH
                 *   2. IF NEW MARK IS LESS THAN HALF
                 *      THE LAST ONE, THEN IT IS A DOT
                 *   3. IF MARK IS MORE THAN HALF, BUT
                 *      LESS THAN TWICE, IT IS SAME AS
                 *      THE LAST MARK
                 *   4. IF A SPACE IS MORE THAN 3/4 OF
                 *      LAST DASH LENGTH, IT IS LETTER SPACE
                 *   5. IF A SPACE IS MORE THAN TWICE THE
                 *      LAST DASH LENGTH, IT IS WORD SPACE
                 *   REF: BELL, BST, JAN 1971
                 *
00BA 7D 0024 R START   TST   DFLAG    NEW DATA READY?
00BD 2A 03             BPL   J10
00BF 7E 01C5 R         JMP   WAIT     WAIT FOR DATA
00C2 7D 0026 R J10     TST   LDATA    LAST DATA MARK OR SPACE?
00C5 2A 03             BPL   J11
00C7 7E 0166 R         JMP   SPACE
00CA B6 0028 R J11     LDA A LMLEN    MLEN .GE. 2*LMLEN?
00CD 48                ASL A
00CE B1 002B R         CMP A MLEN
00D1 2F 50             BLE   DASH
00D3 B6 0028 R         LDA A LMLEN    MLEN .LT. LMLEN/2?
00D6 47                ASR A
00D7 B1 002B R         CMP A MLEN
00DA 2A 05             BPL   DOT
00DC 7D 0027 R         TST   LMARK    LAST DATA MARK OR SPACE?
00DF 2B 42             BMI   DASH
               *
00E1 86 01     DOT     LDA A #1
00E3 B7 0027 R         STA A LMARK    PUT DOT INTO REGISTER
00E6 B6 002F R         LDA A STLDOT
00E9 F6 0030 R         LDA B STLDAS
00EC 48                ASL A
00ED 25 6F             BCS   LENERR   TOO MANY BITS FOR REGISTER
00EF 58                ASL B
00F0 25 6C             BCS   LENERR
00F2 4C                INC A          INSERT DOT
00F3 B7 002F R         STA A STLDOT
00F6 F7 0030 R         STA B STLDAS
00F9 F6 002B R         LDA B MLEN     REGENERATE CLOCK RATE
00FC C0 03             SUB B #3       BASED ON DOT LENGTH
00FE 2C 0D             BGE   NFAST    IF LESS THAN 3, CLOCK SLOW
0100 7A 0032 R         DEC   TIMER    IF MORE THAN 5, CLOCK FAST
0103 2E 0F             BGT   PUT      IF BETWEEN 3 AND 5, O.K.
0105 7F 0032 R         CLR   TIMER
0108 7C 0032 R         INC   TIMER
010B 20 07             BRA   PUT
010D C1 02     NFAST   CMP B #2
010F 2F 03             BLE   PUT
0111 7C 0032 R         INC   TIMER
0114 F6 0032 R PUT     LDA B TIMER    LIMIT TIMER TO 1-120
0117 C1 78             CMP B #120
0119 2D 02             BLT   RELOAD
011B C6 78             LDA B #120
011D F7 0032 R RELOAD  STA B TIMER
0120 7E 0150 R         JMP   STORE
               *
0123 B6 002B R DASH    LDA A MLEN     SET LTRSP=3/4*MLEN
0126 47                ASR A
0127 16                TAB
0128 57                ASR B
0129 1B                ABA
012A B7 0029 R         STA A LTRSP
012D 86 FF             LDA A #$FF     PUT DASH INTO REGISTER
012F B7 0027 R         STA A LMARK
0132 B6 002F R         LDA A STLDOT
0135 F6 0030 R         LDA B STLDAS
0138 48                ASL A
0139 25 23             BCS   LENERR   TOO MANY BITS FOR REGISTER
013B 58                ASL B
013C 25 20             BCS   LENERR
013E 5C                INC B          INSERT DASH
013F B7 002F R         STA A STLDOT
0142 F7 0030 R         STA B STLDAS
0145 B6 002B R         LDA A MLEN     SET WDSP=2*MLEN
0148 48                ASL A
0149 2A 02             BPL   J12
014B 86 7F             LDA A #127
014D B7 0033 R J12     STA A WDSP
               *
0150 B6 002B R STORE   LDA A MLEN
0153 B7 0028 R         STA A LMLEN
0156 86 FF     RESET   LDA A #$FF     RESET "DATA-READY" FLAG
0158 B7 0024 R         STA A DFLAG
015B 7E 01C5 R         JMP   WAIT
               *
015E 7F 002F R LENERR  CLR STLDOT     ELIMINATE BAD DATA
0161 7F 0030 R         CLR STLDAS
0164 20 F0             BRA RESET      GIVE UP ON CHARACTER
               *
```

```
0163 0166 B6 002E R SPACE  LDA A SPLEN   IS SPLEN .GE. LTRSP?
0164 0169 B1 0029 R        CMP A LTRSP
0165 016C 2D F8            BLT   RESET
0166 016E F6 0030 R        LDA B STLDAS  DECODE LETTER FROM REGS.
0167 0171 C4 3F            AND B #$3F    RESTRICT TO 6 BITS
0168 0173 B6 002F R        LDA A STLDOT
0169 0176 84 3F            AND A #$3F
0170 0178 58               ASL B
0171 0179 1B               ABA
0172 017A 84 7F            AND A #$7F    RESTRICT TO 7 BITS
0173 017C CE 0035 R        LDX   #DECTAB
0174 017F B7 0183 R        STA A INDEX1+1
0175 0182 E6 00     INDEX1 LDA B 0,X     RETRIEVE ASCII CODE
0176 0184 B6 0034 R        LDA A WDSXCV  BUFFER FULL?
0177 0187 81 20            CMP A #32
0178 0189 2C CB            BGE   RESET   IF SO, PUNT...
0179 018B BB 002D R        ADD A OUTXCV  GENERATE BUFFER POINTER
0180 018E 84 1F            AND A #$1F    RESTRICT RANGE
0181 0190 7C 0034 R        INC   WDSXCV
0182 0193 CE 0003 R        LDX   #BUFF1
0183 0196 B7 019A R        STA A INDEX2+1
0184 0199 E7 00     INDEX2 STA B 0,X     PUT CHAR. INTO BUFFER
0185 019B 7F 002F R        CLR   STLDOT  CLEAR REGISTERS
0186 019E 7F 0030 R        CLR   STLDAS
0187 01A1 B6 002E R        LDA A SPLEN   IS SPLEN .GE. WDSP?
0188 01A4 B1 0033 R        CMP A WDSP
0189 01A7 2D AD            BLT   RESET   NOT A WORD SPACE
0190 01A9 B6 0034 R        LDA A WDSXCV  INSERT BLANK FOR WORD SPACE
0191 01AC 81 20            CMP A #32     IS BUFFER FULL?
0192 01AE 2C A6            BGE   RESET   IF SO, PUNT....
0193 01B0 BB 002D R        ADD A OUTXCV  GENERATE BUFFER POINTER
0194 01B3 84 1F            AND A #$1F    RESTRICT RANGE
0195 01B5 7C 0034 R        INC   WDSXCV
0196 01B8 C6 20            LDA B #$20    ASCII BLANK CHARACTER
0197 01BA CE 0003 R        LDX   #BUFF1
0198 01BD B7 01C1 R        STA A INDEX3+1
0199 01C0 E7 00     INDEX3 STA B 0,X     PUT BLANK INTO BUFFER
0200 01C2 7E 0156 R        JMP   RESET
0201                ******************************************
0202                *
0203                *    END OF MAIN PROGRAM
0204                *    BEGIN DELAY TIMER
0205                *    GENERATE "TIMER" MSECS. OF DELAY
0206                *
0207                ******************************************
0208 01C5 B6 0032 R WAIT   LDA A TIMER
0209 01C8 F6 0025 R DLOOP1 LDA B DTIME
0210 01CB 5A        DLOOP2 DEC B
0211 01CC 26 FD            BNE   DLOOP2
0212 01CE 4A               DEC A
0213 01CF 26 F7            BNE   DLOOP1
0214                ******************************************
0215                *
0216                *    END OF DELAY TIMER
0217                *    TEST AUDIO INPUT HERE
0218                *    HANDLE OUTPUT TO TERMINAL
0219                *
0220                ******************************************
0221 01D1 B6 8040          LDA A PIA1    GET AUDIO INPUT
0222 01D4 84 01            AND A #$01    BIT ZERO OF PIA
0223 01D6 B1 002C R        CMP A OLDSTA  BIT=0 IS MARK
0224 01D9 26 10            BNE   DIFFER
0225 01DB 7C 0031 R        INC   TCNTR
0226 01DE 7D 0031 R        TST   TCNTR   OVERFLOW OF TCNTR?
0227 01E1 2C 05            BGE   OKAY
0228 01E3 C6 78            LDA B #120    FIX OVERFLOW
0229 01E5 F7 0031 R        STA B TCNTR
0230 01E8 7E 021C R OKAY   JMP   TRANS
0231                *
0232 01EB C6 01     DIFFER LDA B #1      SET "DATA-READY" FLAG
0233 01ED F7 0024 R        STA B DFLAG
0234 01F0 F6 0031 R        LDA B TCNTR   STORE ITEM LENGTH
0235 01F3 7D 002C R        TST   OLDSTA  A MARK OR SPACE?
0236 01F6 26 06            BNE   ZERO    SPACE
0237 01F8 F7 002B R        STA B MLEN
0238 01FB 7E 0201 R        JMP   NMARK
0239 01FE F7 002F R ZERO   STA B SPLEN
0240 0201 7D 002C R NMARK  TST   OLDSTA
0241 0204 26 08            BNE   NEWSP
0242 0206 C6 01            LDA B #1      SET LDATA TO OLDSTA
0243 0208 F7 0026 R        STA B LDATA   -1=SPACE, 1=MARK
0244 020B 7E 0213 R        JMP   SKIPA
0245 020E C6 FF     NEWSP  LDA B #$FF
0246 0210 F7 0026 R        STA B LDATA
0247 0213 B7 002C R SKIPA  STA A OLDSTA  RESET OLDSTA
0248 0216 7F 0031 R        CLR   TCNTR   RESET TIME COUNTER
0249 0219 7C 0031 R        INC   TCNTR
0250                *
0251 021C B6 0023 R TRANS  LDA A COUNT   CHECK FOR TERMINAL READY
0252 021F F6 0032 R        LDA B TIMER
0253 0222 C1 3C            CMP B #60     TRY TO SPEED TERMINAL
0254 0224 2D 01            BLT   S1
0255 0226 48               ASL A
0256 0227 B1 002A R S1     CMP A MAXCNT
0257 022A 2C 06            BGE   OUTCHK  TERMINAL READY
0258 022C 7C 0023 R        INC   COUNT
0259 022F 7E 00EA R        JMP   START   RETURN TO BEGINNING
0260                *
0261 0232 7F 0023 R OUTCHK CLR   COUNT
0262 0235 B6 0034 R        LDA A WDSXCV  ANY DATA IN BUFFER?
0263 0238 26 03            BNE   WDOUT
0264 023A 7E 00EA R        JMP   START   IF NOT, RETURN
0265 023D F6 002D R WDOUT  LDA B OUTXCV  GENERATE BUFFER POINTER
0266 0240 CE 0003 R        LDX   #BUFF1
0267 0243 F7 0247 R        STA B INDEX4+1
0268 0246 A6 00     INDEX4 LDA A 0,X     OUTPUT CHAR. IN ACCUM A
0269                *    USE SYSTEM SUBROUTINE TO OUTPUT CHARACTER
0270 0248 BD 72F0          JSR   OUTCRT
0271 024B 7A 0034 R        DEC   WDSXCV
0272 024E F6 002D R        LDA B OUTXCV
0273 0251 5C               INC B
0274 0252 C4 1F            AND B #$1F    RESTRICT RANGE
0275 0254 F7 002D R        STA B OUTXCV
0276 0257 7E 00EA R        JMP   START   BACK TO THE BEGINNING
0277                       END
```

previously sampled value. If no change is found, the time counter is incremented to indicate the increased length of the signal. If a change is found, then a series of processes are done. The data ready flag is set to tell the decoder that a pulse is complete, the time counter is stored into the mark length or space length variable, the data type (mark or space) is recorded, and the time counter is reset to one. A few notes are in order about sampling. First, the sampler assumes that an input level zero (ground) indicates the presence of an audio tone (mark), and that a one level input (high voltage) indicates no tone (space). This setup coincides with the audio processing hardware described later; but if one wishes to have a one input signify mark instead, only two simple program changes are necessary. Changing the BNE (branch on not equal to zero) instructions on lines 236 and 241 to BEQ (branch on equal to zero) instructions will accomplish the inversion. This illustrates the ease of modifying the system when it is based on a program instead of hardware. One other note: One must protect against overflows of the time counter. This occurs when long marks or long spaces cause the counter value to exceed the maximum value representable as a positive byte. MORSER checks for such occurrences, and resets the time counter to a large positive value whenever an overflow is detected.

We come now to the decoding section. The section is a software version of algorithms abstracted from several hardware designs. It can be described by a set of five decoding rules. The length (time counter value) of the last mark received is used to determine the type of the present mark. The length of the last dash received is used to determine the type of space being received.

RULE 1. If the new mark length is at least twice the length of the last mark received, then the new mark is a dash.

RULE 2. If the new mark length is less than one half of the length of the last mark received, then the new mark is a dot.

RULE 3. If the new mark length is more than one half but less than twice the length of the last mark received, then the new mark is the same type as the old mark was.

RULE 4. If the new space length is more than 3/4 of the last dash length received, then the new space is a letter space.

RULE 5. If the new space length is

> longer than twice the last dash length received, then the new space is a word space.

Any other space is an element space. These rules determine the processing path of each data item returned by the sampling section. The dots and dashes are stored in memory (STLDOT and STLDAS) until a letter space is detected. Then the memory contents are converted to an offset in DECTAB, following the process previously described. The character code found in DECTAB is then transferred to the output buffer. The buffer is arranged as a 32 character first in, first out store which allows the decoding to get ahead of the output device for short periods. The detection of a word space causes the latest letter to be decoded and an extra blank character is inserted into the output buffer to provide a space between output words.

The decoding section also adjusts the TIMER value each time a dot is detected. Dots are nominally four samples long. If a dot is declared shorter than three samples, the sampling period is reduced. If a dot is declared longer than five samples, the sampling period is increased. This mechanism helps MORSER to follow changes in code speed during a message or even within characters. The flexibility of the decoding rules will allow code far from the proper timing to be decoded correctly during the adjustment process. Most decoding errors will result either in no character at all being output, or a blank being substituted for the garbled character. MORSER can handle code with wild speed variations and weightings from 10% to 90%, but it can be fooled by sufficiently erratic code. So too, however, can most human operators.

MORSER uses a programming "trick" in conjunction with indexed addressing to facilitate the decoding and output process. This occurs on lines 174, 183, 198, and 267. The problem is to retrieve or store data at a particular location within a table or buffer. The starting address of the area is known, and the desired offset is calculated each time. MORSER uses the technique of modifying itself during execution. Since the program is stored in programmable memory, it can be changed just as the variables can be changed. In Motorola systems, the second byte of an instruction using indexed addressing stores an offset to be added to the index register contents to generate the final effective address for the instruction. MORSER loads the index register with the beginning of the desired table and then writes the calculated offset into the second byte of the indexed instruction. The processor adds the two, making the desired address. This works well, as long as the program is stored in programmable memory, and one is careful where one writes. If this type of trickery is to be avoided (for example, if the program is to be put into read only memory), the process of adding the offset to the base address must be done explicitly. The following code will perform the function, where X contains the starting address of the table, and accumulator A contains the offset:

```
INDSAV  RMB  2              Programmable memory area for
                            computation
INDEX   STX     INDSAV      Starting address
        PSH  B              Save contents of B accumulator
        CLR  B
        ADD  A  INDSAV+1    Add low order part of address with
                            offset in A
        ADC  B  INDSAV      Add high order part of address with
                            carry
        STA  A  INDSAV+1
        STA  B  INDSAV
        LDX     INDSAV      Put effective address into index register
        PUL  B              Restore contents of B accumulator
```

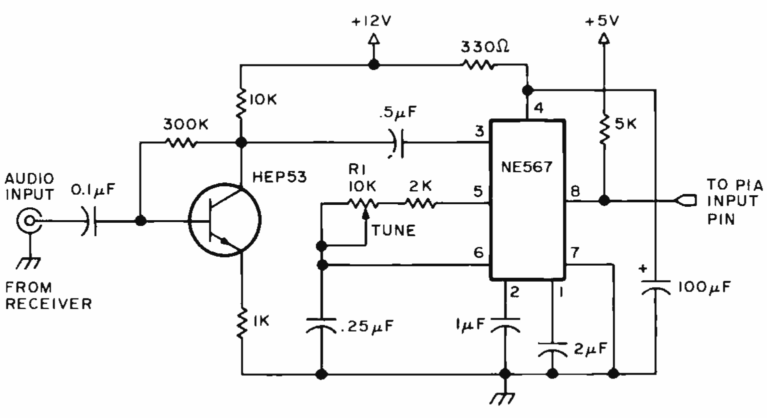


Figure 1: A suggested audio input processing circuit designed to be used with an amateur radio receiver. The tuning adjustment sets the frequency of the signal which is to be interpreted as a dot or dash by the program. The receiver tuning and BFO should be adjusted so that the desired station will have its dots and dashes at the frequency set by R1.

The output terminal driver makes use of the delay timer to prevent the program from exceeding the speed capabilities of the output device. A count is kept of the number of delay periods since the last character was sent to the terminal. The value of MAXCNT is set such that MAXCNT delay periods at a 60 words per minute Morse code speed are roughly equal to the recovery period of the terminal in use. If the code speed is less than 30 words per minute, then one half of MAXCNT is used to let the terminal run nearer its full speed. This counting ensures that the terminal will not receive characters faster than it is capable of handling them. The output buffer helps absorb speed variations. In cases of extreme speed variations with a very slow output terminal, a few characters may be lost. This would require input speeds of over 80 words per minute to be maintained for many characters if one used a 110 baud terminal. It is not likely that the audio processing hardware could switch at this rate, so the characters would be unreliable anyway. All slower speeds could be handled without trouble. The system subroutine used to drive the terminal is called in this section on line 270. It should not use interrupts or any timing loops, since these will upset the timing in MORSER. Since MORSER already does output timing, they are not needed anyway.

That is all there is to the MORSER program. The most critical part of the whole system is the audio input hardware. This circuit needs rapid response (since switching time for 60 words per minute code is about 13 ms), audio selectivity, immunity to noise, and immunity to varying signal levels. No optimum circuit is known by the authors. The circuit shown is a suggestion which shows promise. It uses a 567 phase locked loop tone decoder, tuned to a center frequency of about 1 kHz. The bandwidth is set at approximately 10% of the center frequency, or about 100 Hz. This circuit should switch fast enough for most code speeds, and the phase lock design gives noise immunity, for the circuit will require 10 or more cycles at the correct frequency before it will switch. An input level of about 200 mV seems to give the best immunity from interfering signals and noise. The output rests at +5 V, dropping to near ground when a proper frequency tone is detected. This voltage level is read by the sampler program through the peripheral interface. This signal must be very clean, since variations will probably be decoded as extra dots and dashes. Lots of 'E's and 'T's will indicate a problem in the audio processing.

MORSER works surprisingly well, and it will be able to decode almost anything that its operator can feed it. It does not know English or radio terminology, so it cannot guess what the characters should be. It simply decodes what it "hears," and may not correct some errors that a human would recognize. It does decode the input faithfully, and that can be quite a help. Of course if you use MORSER to listen to a station using automatic code generation technology (on a clear channel) you should get perfect copy once the speed adaptation is complete. ■

Efficient Storage of Morse Character Codes

Lawrence Krakauer
15 Orchard Ln
Wayland MA 01778

Since many of BYTE's readers are radio amateurs, a number of whom have expressed an interest in combining computing with their radio hobby, BYTE seems an appropriate place for this short note on a technique for storing Morse character codes in a minimal amount of memory space. It may not be immediately obvious that all of the possible Morse characters, up to six elements per character, can be stored using a maximum of only seven active bits per character stored.

A typical program using such a stored representation would be a programmed automatic keyer. As the user typed at a keyboard, the computer would receive the characters typed in some code (typically ASCII code). The computer would then use this code to index into a table of Morse character codes, retrieving a coded representation of the Morse character to be transmitted. With a suitable interface, the computer could then directly key the radio transmitter. A good program of this type would buffer the user's typing, allowing the operator to be some large number of characters ahead of the code actually being transmitted.

Another similar program would be a Morse code teaching machine, which could use a programmed pseudorandom number generator to create arbitrary "code groups." Such artificial text is customarily used for Morse code training, so that the student does not recognize words and thus anticipate the next letter to come. The computer in this case would key a practice oscillator, and the student would attempt to accurately receive what it sends. If the student types what he receives directly on the computer keyboard, the computer could notify him immediately of any errors. In addition, the computer could adjust its transmission speed for a fixed error rate, always sending at the upper limit of the student's reception speed, which would hopefully produce rapid learning.

A Morse character is a string of from one to six Morse elements, where each element is a dot or a dash. Since there are only two types of elements, Morse characters seem to

Table 1: Morse Code Patterns and ASCII Equivalents for Selected Graphics.

Graphic Char	ASCII Code Hex	ASCII Code Octal	Morse Pattern Hex	Morse Pattern Octal	Morse Pattern Binary
A	41	101	60	140	01100000
B	42	102	88	210	10001000
C	43	103	A8	250	10101000
D	44	104	90	220	10010000
E	45	105	40	100	01000000
F	46	106	28	050	00101000
G	47	107	D0	320	11010000
H	48	110	08	010	00001000
I	49	111	20	040	00100000
J	4A	112	78	170	01111000
K	4B	113	B0	260	10110000
L	4C	114	48	110	01001000
M	4D	115	E0	340	11100000
N	4E	116	A0	240	10100000
O	4F	117	F0	360	11110000
P	50	120	68	150	01101000
Q	51	121	D8	330	11011000
R	52	122	50	120	01010000
S	53	123	10	020	00010000
T	54	124	C0	300	11000000
U	55	125	30	060	00110000
V	56	126	18	030	00011000

Graphic Char	ASCII Code Hex	ASCII Code Octal	Morse Pattern Hex	Morse Pattern Octal	Morse Pattern Binary
W	57	127	70	160	01110000
X	58	130	98	230	10011000
Y	59	131	B8	270	10111000
Z	5A	132	C8	310	11001000
0	30	060	FC	374	11111100
1	31	061	7C	174	01111100
2	32	062	3C	074	00111100
3	33	063	1C	034	00011100
4	34	064	0C	014	00001100
5	35	065	04	004	00000100
6	36	066	84	204	10000100
7	37	067	C4	304	11000100
8	38	070	E4	344	11100100
9	39	071	F4	364	11110100
.	2E	056	56	126	01010110
,	2C	054	CE	316	11001110
?	3F	077	32	062	00110010
=	3D	075	8C	214	10001100
:	3A	072	E2	342	11100010
;	3B	073	AA	252	10101010
/	2F	057	94	224	10010100
—	2D	055	86	206	10000110

This table assumes that for machine generated Morse, data is shifted out through the most significant bit with 0 shifted into the low order. This process continues until the binary pattern 10000000 remains in the working memory location. For input, data is shifted in through the LSB. At end of character, a 1 is appended, and the code is left justified.

Note: *The representation of Morse characters described in this article is also used by Bruce Filgate in his extensive Morse code applications program described on page 52 of this issue. The information in this table (with some minor changes) is found at addresses 20/225 to 20/366 of listing 1 in Bruce's article.*

EXAMPLE

Sending an L . . . .

Morse Encoded in binary: 01001000

Iteration	Data After ASHL A	Data After ASHL CY	Morse Out
0	10010000	0	dot
1	00100000	1	dash
2	01000000	0	dot
3	10000000	0	dot
4	00000000	1	end char.

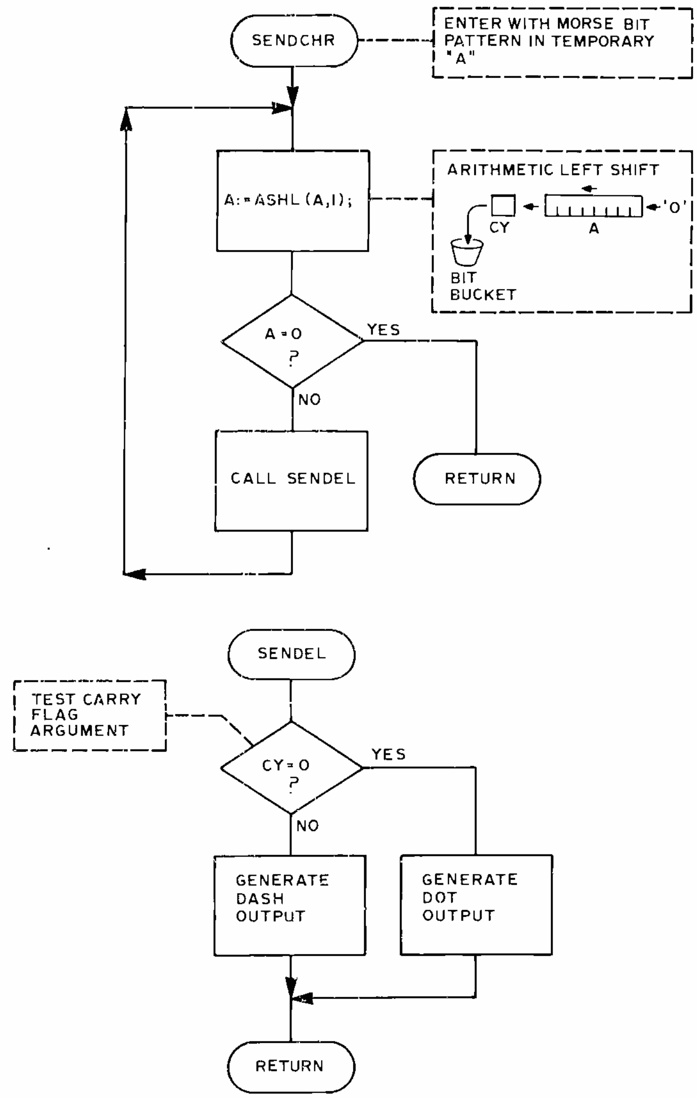


Figure 1: Flow chart of the SENDCHR routine and the SENDEL routine, used to generate Morse code outputs. The key element of the method is use of an arithmetic left shift operation to interpret the code which is presented to the routine. The codes are determined by a table lookup prior to calling SENDCHR, using an internal table equivalent to table 1. An example of the operation of the routine on a letter L is illustrated.

be a natural for a binary representation, simply letting 0 represent a dot, and 1 represent a dash. The catch is that all of the characters are not the same length, so that the number of elements in the character must also be included in its representation. However, the simple approach of directly including a number giving the length requires too many bits. Since there are up to six elements in a Morse character, we would need six bits for the character elements, plus three bits to represent the length, for a total of nine bits. Since most small computer systems are presently oriented around eight bit bytes, it would be convenient if all the Morse characters could be represented by eight, or fewer, bits each.

Let's try counting. There are two Morse characters of length 1 element (• and –), four characters of length 2 (•• •– –• and – –), and so on. So extending this sequence, the total number of possible characters of length six or less is:

$$N = 2 + 4 + 8 + 16 + 32 + 64 = 126$$

Since N is less than $2^7 = 128$, we ought to be able to represent all the Morse characters in seven bits. Perhaps it should be mentioned here that this includes all combinations of up to six elements, including those which do not actually represent legal Morse characters. Since we want a representation which will make it easy to transmit the character, we need a simple form that encodes all arbitrary combinations.

And, indeed, such a simple form exists.

Listing 1: The SENDCHR routine defined as symbolic assembly language for the 6800 and 8080 architecture; (a) shows the 6800 code, and (b) shows the 8080 code. These routines assume that a subroutine called SENDEL (see figure 1) is available which sends either a dot or a dash based on the state of the carry (CY) flag. Since temporary data is maintained in A, SENDEL must not alter the contents of A.

(a) 6800 Code:

Label	Op	Operand	Commentary
SCHRLOOP	JSR	SENDEL	Send dot or dash;
SENDCHR	ASLA		Entry point and shift left;
	BNE	SCHRLOOP	If more data then reiterate;
	RTS		Else return;

(b) 8080 Code:

Label	Op	Operand	Commentary
SENDCHR	ADD	A	Shift left by adding A to A;
	RZ		If no data then return;
	CALL	SENDEL	Send dot or dash;
	JMP	SENDCHR	Reiterate

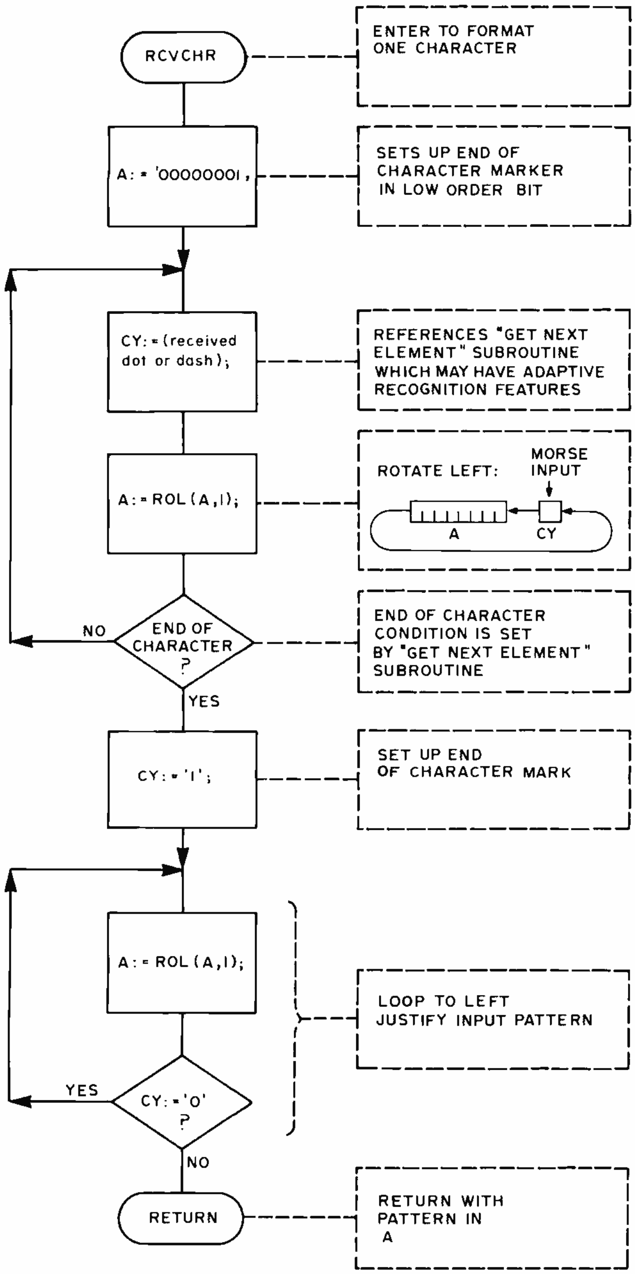


Figure 2: Flow chart of the RCVCHR routine which is used to format incoming dots (binary 0) and dashes (binary 1) into an 8 bit code. Reference is made to a routine loosely described as "get next element" which is used to receive a Morse code element. This "get next element" input routine will have adaptive speed interpretation features required if the program is to be used with hand sent Morse code. After the end of character indication is returned by the input routine, the RCVCHR routine must left justify the code so that it is in the same form as found in the code generation table (see table 1). Control returns with the input code contained in register A.

The Morse character is represented from left to right, one bit per element, using 0 for dot and 1 for dash. At the end of the character, after the last element, a 1 is placed as a stop bit. The remaining bits, if any, are filled out with zeros. Some examples are:

Character	Morse Code	Binary Representation
E	•	0100000
T	–	1100000
A	• –	0110000
L	• – • •	0100100
,	– – • • – –	1100111
=	– • • • –	1000110

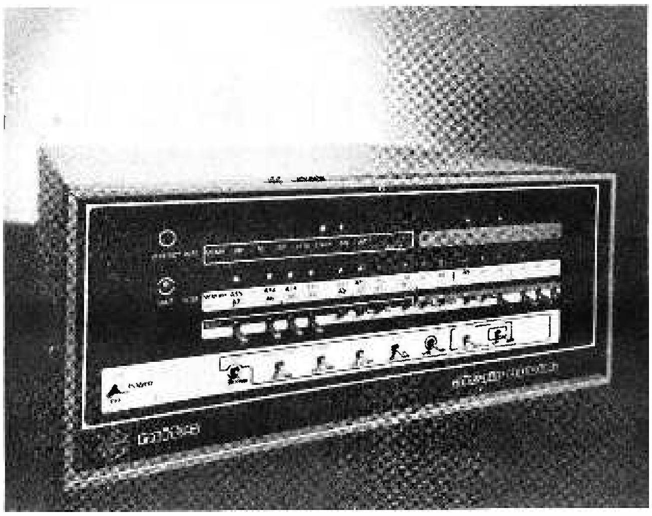
Table 1 is a complete list of alphabetic characters, numerals and selected special characters shown as hexadecimal and octal ASCII along with the equivalent hexadecimal, octal and binary code for the Morse pattern. Given the representation, transmission of the character is easy. Working from left to right, simply transmit each bit as a Morse element, until all that remains is a single 1 followed by all zeros, at which point the character has been completed. This operation is easily done using the "rotate" or "shift" instructions of the typical microcomputer. A flow chart for this procedure is given in figure 1. It is so simple that it reduces to only a four line subroutine in a typical microprocessor. Listing 1a gives a suggested program for the Motorola 6800 and listing 1b gives a program for the Intel 8080 (or 8008). These programs use an eight bit representation, which can express a code of up to seven elements, although no seven element characters are actually used in the Morse code.

Note that there are two binary representations which represent the null character; that is, they result in the subroutine returning without transmitting anything at all. These are the codes 00000000 and 10000000. If desired, they could be specially decoded. For example, one of them could be used to represent the special Morse character ...., which is used as a delete character. It would have to be transmitted as a special case, since it contains eight elements. If the all zero code were to be used for this purpose, it could be tested for by a branch-on-zero instruction at the subroutine entry point.

The representation proposed here is also easy to build up while receiving Morse code. A receive subroutine flow chart is given in figure 2.

The possibilities inherent in the combination of small computers and amateur radio are enormous. The challenging task of machine reception of hand sent code is now receiving much more attention due to the low cost of microprocessors. The possibilities of computer station control, amateur radio data communications networks, and microprocessor control of repeaters have barely begun to be explored. The large number of radio amateurs who are now beginning to experiment with small computers should produce some rather interesting results in the near future.■

altair™ 8800b
TECHNICAL INFORMATION



The ALTAIR 8800b computer is a general purpose byte-oriented machine (8-bit word). It uses a common 100-pin bus structure that allows for expansion of either standard or custom plug-in modules. It supports up to 64K of directly addressable memory and can address 256 separate input and output devices. The ALTAIR 8800b computer has 78 basic machine language instructions and is comprised of a power supply board, an interface board, a central processing unit (CPU) board, and a display/control board.

Power Supply Board

The Power Supply Board provides two output voltages to the ALTAIR 8800b computer bus, a positive and negative 18 volts. It includes a bridge rectifier circuit and associated filter circuit, a 10-pin terminal block connector, and the regulating transistors for the positive and negative 18 volt supplies.

Interface Board

The Interface Board buffers all signals between the display/control board and the ALTAIR 8800b bus. It also contains eight parallel data lines which transfer data to the CPU from the Display/Control board.

CPU Board

The CPU board controls and processes all instruction data within the ALTAIR 8800b computer. It contains the model 8080A microprocessor circuit, the master timing circuit, eight input and eight output data lines to the ALTAIR bus, and control circuits.

Display/Control Board

The Display/Control Board conditions all ALTAIR 8800b front panel switches and receives information to be displayed on the front panel. It contains a programmable read only memory (PROM), switch and display control circuits, and control circuits to condition the CPU.

NEW DESIGN FEATURES

Several new design features have been incorporated into the electronic and mechanical areas of the ALTAIR 8800b computer. Some of the new design features include additional front panel capabilities, redesigned power supply, and various electronic and mechanical design advancements.

New Front Panel Switches

Five new front panel switch positions have been added to the ALTAIR 8800b computer to expand the front panel capability.

1. SLOW position: Permits execution of a program at a rate of approximately 2 machine cycles per second or slower. The normal machine speed is approximately 500,000 machine cycles per second. The ALTAIR 8800b operates in the slow mode as long as the SLOW switch is depressed on the front panel.
2. DISPLAY ACCUMULATOR position: Displays the contents of the CPU accumulator register on the ALTAIR 8800b front panel.
3. LOAD ACCUMULATOR position: Loads the information present on the lower eight front panel address switches into the CPU accumulator register.
4. INPUT ACCUMULATOR position: Inputs the information present at an Input/Output device into the CPU accumulator register. The Input/Output device is selected on the upper eight front panel address switches.
5. OUTPUT ACCUMULATOR position: Outputs the contents of the CPU accumulator register to a selected input/output device. The input/output device is selected on the upper eight front panel address switches.

New Power Supply

The new power supply in the ALTAIR 8800b contains an 8 volt, 18 ampere tapped secondary supply which permits the addition of up to 16 printed circuit cards, and pre-regulated positive and negative 18 volt, 2 ampere supplies. A multiple tapped primary transformer provides for 110/220 volt operation and a 50/60 Hz operation.

Electronic Design Advancements

The electronic design advancements on the ALTAIR 8800b are in the CPU and front panel circuit boards.

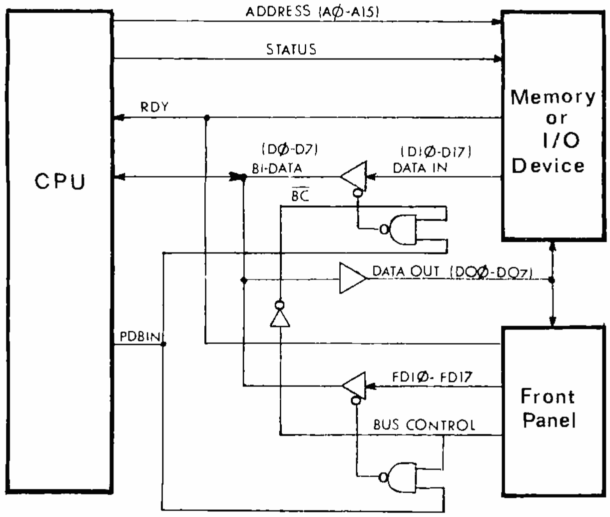
1. CPU. The new CPU circuit board uses the Intel 8224 clock generator integrated circuit (IC). The 8224 IC provides a specified clock frequency to the ALTAIR 8800b using an external crystal and dividing the crystal frequency down to 2MHz. Therefore, both the clock pulse widths and phasing (as well as frequency) are crystal controlled.
2. Front Panel. All front panel data lines are connected to an interface which buffers them from the rest of the ALTAIR 8800b. The front panel circuits also use a programmable read only memory (PROM) which contains programs for the following eight functions:
   EXAMINE
   EXAMINE NEXT
   ACCUMULATOR DISPLAY
   ACCUMULATOR LOAD
   DEPOSIT
   DEPOSIT NEXT
   INPUT ACCUMULATOR
   OUTPUT ACCUMULATOR

   The front panel circuits also have a wiring option which allows the CPU to perform a complete instruction cycle or a single machine cycle during the single step or slow operation.

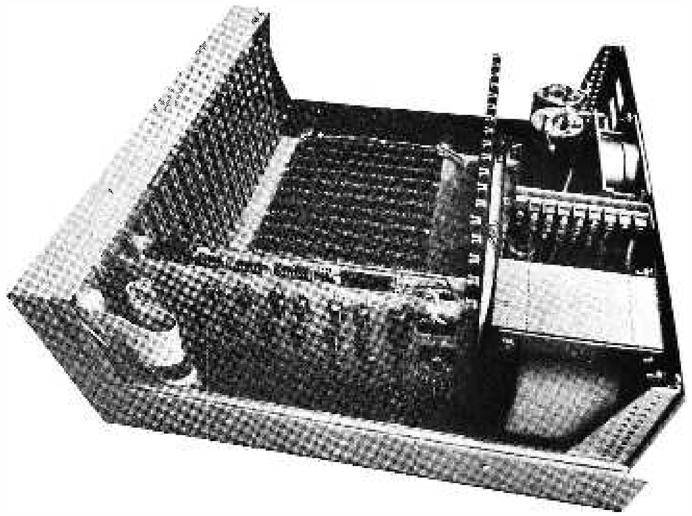
Mechanical Design Advancements

The mechanical design advancements on the ALTAIR 8800b are incorporated for ease of assembly and maintenance.

1. The wiring harness connection which exists on the front panel of the ALTAIR 8800 is replaced with ribbon cables. These ribbon cables connect the front panel circuits to the interface circuits.
2. The four slot expander cards in the ALTAIR 8800 have been replaced by a single piece 18-slot motherboard. The 18-slot motherboard contains 100 solder lands which comprise the 100 pin bus.
3. A new multi-color and redesigned dress panel is used in the ALTAIR 8800b. The front surface of the dress panel has a protective sheet of mylar to insure that the graphics are not rubbed or scratched off.



8800b BLOCK DIAGRAM



COMPATABILITY

8800b BLOCK DIAGRAM DESCRIPTION

The 8800b computer contains four main circuits: a Central Processing Unit (CPU), a Memory, an Input/Output (I/O), and a Front Panel. The CPU controls the interpretation and execution of software instructions, and the Memory stores the software information to be used by the CPU. The I/O provides a communication link between the CPU and external device. The Front Panel allows the operator to manually perform various operations with the 8800b. The 8800b block diagram description explains: A) the communication between the CPU and the memory or I/O circuits; and B) the communication between the CPU and the front panel.

CPU to Memory or I/O Operation

The Memory or I/O operation requires several main signals which allow for transfer of data to and from the CPU. The ADDRESS (AØ-A15) signal consists of sixteen individual lines from the CPU to the Memory or I/O device. This signal represents a particular memory address location or external device number which is needed to establish communications with the Memory or I/O Device. Once the ADDRESS (AØ-A15) data is presented to the Memory or I/O device, the CPU generates various STATUS signals. The STATUS signals either enable decoding of a memory address, or they condition the I/O device card to send or receive data from the CPU.

Data from the Memory or I/O device is presented on the DATA IN (DIØ-DI7) lines and applied to eight non-inverting bus drivers. The drivers are enabled by a PDBIN signal from the CPU and a BC (bus control) signal. The BC signal is LOW when the Front Panel is not in operation. The eight non-inverting bus drivers, when enabled, present the input data to BI-DATA (DØ-D7) lines which apply the data from the Memory or I/O device to the CPU.

Data to the Memory or I/O device is presented on the DATA OUT (DOØ-DO7) lines from the BI-DATA (DØ-D7) lines from the CPU. The RDY (ready) line either forces the CPU to a wait state while data is being transferred or allows the CPU to process data.

Front Panel Operation

The Front Panel Operation is very similar to the Memory or I/O operation. The Front Panel gains control of the CPU by producing a HIGH BC signal. The BC signal disables the DATA IN (DIØ-DI7) lines from a Memory or I/O device and enables the FDIØ-FDI7 lines. The FDIØ-FDI7 lines contain Front Panel data which is transferred to the CPU upon the occurrence of the PDBIN signal. All data from the CPU to the Front Panel is applied to the DATA OUT (DOØ-DO7) lines and displayed on the Front Panel.

Compatibility

All of the current 8800 software is compatible with the 8800b, and all the current plug-in circuit boards are compatible, with the exception of the 8800a CPU Board.

Memory Cards

1. 4K Dynamic RAM Memory Board
2. 4K Static RAM Memory Board
3. 16K Static RAM Memory Board
4. PROM Memory Board

Interface Cards

1. Serial Interface Board
2. Parallel Interface Board
3. Audio-cassette Interface Board
4. Disc Controller Board

ALTAIR 8800b
Specifications

Number of Boards	Up to 18
Microprocessor	
Model	8080A
Technology	NMOS
Data Word Size, Bits	8
Instruction Word Size, Bits	8
Clock Frequency	2M Hz
Add Time, Register to Register, Microsec. Per Data Word	2
Number of Instructions	78
Input/Output Control	
I/O Word Size, Bits	8
Number of I/O Channels	256
Direct Memory Access	Optional
Interrupt Capability	Std.
Vectored Interrupt (8 priority levels)	Optional
Software	
Resident Assembler	Yes
Higher-level Language	BASIC
Monitor or Executive	Sys. Mon.; text edit.
Complete Software Library Separately Priced	Yes

mits
2450 Alamo S.E. / Albuquerque, New Mexico 87106

Software Bug of the Month 5

Private Preston's Folly

Private Prescott Preston, Proud yet Prudent Programmer, is running his first sort routine, and it's a mess. See if you can straighten him out.

Private Preston has an array called PR, containing 50 elements. He reads in some data in unsorted fashion. The idea of the sort routine is to test whether two adjacent elements are in order. If they're out of order, they are interchanged.

A flag K is kept to tell whether any changes have been made in the current pass through the array PR (from beginning to end). If *no* changes have been made, we are done, because the table is now sorted (remember we've gone through the entire array). Whenever a change is made, however, the flag K is set. If the flag ever gets set, we have to make another pass through the array, and we keep on doing this until the array is sorted.

The program Private Preston wrote looked like this:

Step 1: Set K equal to zero.
Step 2: Set I equal to 1.
Step 3: If PR(I) is less than or equal to PR(I+1), then go to Step 7.
Step 4: Set K equal to one.
Step 5: Set PR(I) equal to PR(I+1).
Step 6: Set PR(I+1) equal to PR(I).
Step 7: Set I equal to I + 1.
Step 8: If I is unequal to 50, go to Step 3.
Step 9: If K is equal to zero, go to Step 1.

I won't even bother telling you what language that routine was written in. The first few times, of course, it had compilation errors. Now it compiles, but it still doesn't work. Why not?■

Solution in Next Month's BYTE

SOLUTION TO BUG OF THE MONTH 4

Did you try N = 1? Good for you. One of the surest ways a program like this can fail is in the low order cases. Unfortunately (for you), N = 1 works just fine. We discover that 1 is not a multiple of 2, and then we quit. So 1 is a prime number.

Now did you try N = 2?

You did? I think you saw this one somewhere before.

In fact, for N = 2 – and only for N = 2 – this program has a bug. It says that 2 is not prime, because 2 is a multiple of 2, when in fact it is prime.

How should the bug be fixed? One simple way, of course, is to test N for being less than 4; if it is, then it is prime. But this adds unnecessary inefficiency in what we shall presume are the bulk of our cases, namely N ⩾ 4. The real trouble is that we are using a "FORTRAN loop" – that must be executed at least once – rather than an "ALGOL loop" that might be done zero times. Thus, when N = 2, the first value of I, namely 2, is already larger than the maximum value, namely the square root of 2, and so no cases should be tested. We can make this into an ALGOL loop by putting a statement number – say 3 – on the last IF statement, and then writing GO TO 3 just before statement number 1.

There is one further acceptable answer. Suppose that we called PRIME(L, L). This does make sense; we are testing L to see if it is prime, and then, after we test, setting L equal to either zero or one. Unfortunately, the first thing that PRIME does is to set K = 1, which sets L equal to 1 (if parameters are called by reference). In other words, the program is now testing whether 1 is prime, not whether L is prime. This can be fixed by moving the statement K = 1 (which is otherwise perfectly good) down to the end of the program, just before the second RETURN statement.■

W Douglas Maurer
University Library Room 634
George Washington University
Washington DC 20052

If Only Sam Morse Could See Us Now

Wayne Sewell WB5NYC
9277 LBJ #253
Dallas TX 75243

When dealing with personal computers, it is necessary for a person to be familiar with his machine from both the hardware and software point of view. It has been my observation that hardware people often tend to stay in hardware and software people in software, each faction learning just enough about the other side to get along. I am guilty of this myself – I never do in hardware what could feasibly be done in software. Hopefully, this situation will change as my knowledge of microcomputing increases, but for the time being I am assuming that the computer runs because of the little elves pushing electrons around in the processor and putting most of my effort into programming.

This article is for people with a similar bent. There are several electronic Morse code keyers on the market, but rather than spend precious cash, why not use that little digital demon of a small computer that you paid all of that money for. You've probably been trying to find a valid use for it anyway. Think of the glory as you transmit a message in Morse code several hundred characters long at a speed of 1000 wpm. (While there is a legal speed limit for radio Teletype, to my knowledge Morse Code bandwidth is not explicitly mentioned in FCC regulations.) Then comes a real challenge: Find someone whose computer can receive at that speed. Finally, the ultimate goal: Read the response sent back to you at the same speed. [*See articles by Robert Grappel and Jack Hemenway, by Bruce Filgate, and by Lawrence Krakauer in this issue.*]

This article describes a software Morse code generator. The program listed is for a 6800 system (specifically the one sold by Southwest Technical Products); but a complete description of the generator is included to facilitate conversion to other systems.

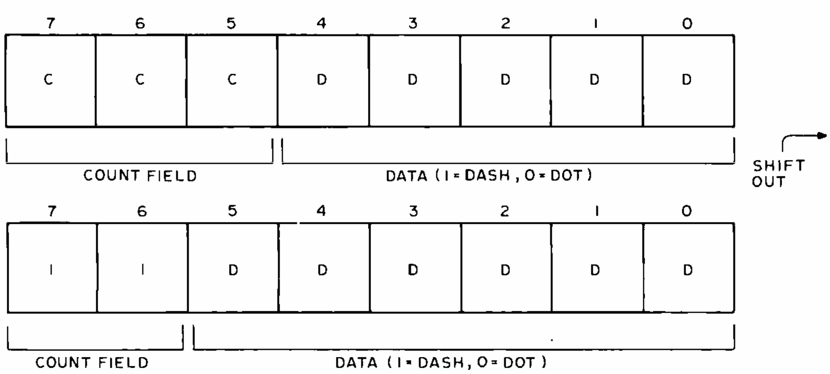


Figure 1: Data Formats. The number of ways to represent data within a computer can sometimes be equated to the number of programmers there are in the world. Here is another way to represent Morse codes internally, which differs from several other articles in this issue. The format is supported by table 3 which lists the equivalent hexadecimal codes for each Morse code graphic.

Data Format

The data format used in my Morse code generator makes it possible to pack any of the "dot" and "dash" combinations associated with a Morse code character into a single byte. The rightmost 5 or 6 bits (throughout this article, "rightmost" and "least significant" are synonymous) contain the pattern of dots and dashes, and the upper 2 or 3 bits contain a count which informs the program how many of the data bits are actually associated with the character. The remaining bits are set to zero arbitrarily. Out of all of the characters in international Morse code, there are only two which will not work in this system: space and the error code. It is not surprising that the space is a maverick in any data representation, since it is not really data at all, but a lack of data. It is simply a lengthening of the normal interval between characters to form an interval between words, and must be handled as a special case. Similarly, the error code is not really a character either.

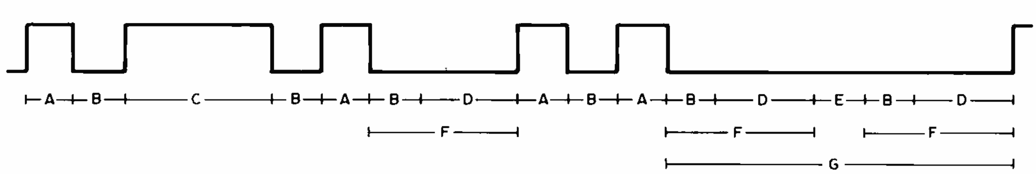
When an error is made during a Morse transmission, due to a spasmodic twitch of the operator's hand or similar cause, the errant operator sends a string of 8 dots, meaning: "Oh I'm sorry I made an error please disregard the last character I sent please will you huh." Both the space and the error code, since they don't fall in the normal scheme of things, are treated as special cases: Immediately upon detection, the program intercepts them, modifies them to a compatible format, and inserts them back into the main logic flow.

For all characters five elements long or less, the format is absurdly simple, as shown in figure 1a. The high order 3 bits contain a binary number from 1 to 5 corresponding to the number of dots and/or dashes in the character. The rightmost 5 bits contain the data elements themselves, where 0 is a dot and 1 is a dash. They are stored from right to left so that when the byte is shifted to the right, they are transmitted serially in the proper sequence. As an example, the Morse code character Q is transmitted as "— — · —". The element count is four, so a binary 100 string is stored in the leftmost 3 bits of the byte. Using the coding scheme above, the equivalent Morse bit pattern for Q (the binary string 1011) is stored in the rightmost 4 bits. In this case, bit 4 is not needed so it is a "don't-care" bit which is set to zero arbitrarily. Therefore, the code for Q stored in the table is the binary string 10001011 or hexadecimal 8B.

If all Morse code characters contained five elements or less, the coding system would be much simpler. However, many of the special characters, uncooperative cusses that they are, contain six elements. At first glance, this doesn't seem to be an obstacle, since the 3 bit count field can contain a binary number up to 7, inclusive. However, there are only 5 bits left over within the byte, which means that although the count can keep track of more than five elements, there is no place to store them. The one fact that keeps the entire system from collapsing into a pile of random logic is that there are no characters with seven elements. There are several special codes with six elements, but I have been unable to come up with one containing seven. Since the difference between a count of six, the binary string 110, and a count of seven (111) is one bit; and since there will never *be* a count of seven, then bit 5 becomes a "don't-care" and can be used as a data bit if bits 6 and 7 are both 1. We simply consider any count greater than five (leftmost 2 bits on) to be equal to six, shift our wild card bit right with the others, and the problem is solved, as seen in figure 1b. When the count is extracted in this case, bit 5 is forced to a zero, making the count equal to six.

When Morse transmission actually starts, the data bits are shifted right one at a time. If the rightmost bit is a zero, the weight factor for a dot is loaded; if the rightmost bit is equal to 1, the dash weight factor is loaded, normally exactly three times that of a dot. The key is turned on, and the timing subroutine, controlled by the value stored in ELESPEED, generates the proper delay for that element. The key is then turned off and the timing subroutine is reentered, this time to generate the space between elements, normally equal to a single dot time. This loop is repeated for every element in the character, shifting right each time. The space between elements is added to the last element in the character, also.

If a string of characters is being transmitted, rather than a single character, the



Time Interval	Explanation
A :	Dot, key on; basic timing element; duration is number of milliseconds contained in ELESPEED.
B :	Space between elements, key off; duration is one dot; also controlled by ELESPEED; generated automatically after every dot or dash.
C :	Dash, key on; duration is three times dot in normal weighting.
D :	Character space synchronizer, key off; generated automatically after every character; added to B to form total space (F) between characters; duration is twice the dot interval for normal spacing but can be increased for greater character separation; controlled by CHRSPEED.
E :	Space character (ASCII hexadecimal 20), key off; duration is one dot; added to character space (F) before and after to form total space (G) between words.
F :	Space between characters, key off; duration is B + D; if ELESPEED and CHRSPEED are synchronized this is equal to three dots; if not, duration is longer.
G :	Space between words, key off; duration = twice F plus one dot; if ELESPEED and CHRSPEED are synchronized, this is equal to seven dots; if not, duration is longer.

Figure 2: Time Relationships. The Morse characters RI sent in succession, followed by a space. The intervals identified by capital letters are explained in the notes below.

Listing 1: CWBUFFER Subroutine. This listing gives the complete absolute hexadecimal assembled code and symbolic representation for the CWBUFFER routine. Data in this listing was prepared using the author's SPUCA assembler program.

```
LOC    CODE        SOURCE STATEMENT
0040              CHRSPEED RMB   2            # OF MILISECONDS BETWEEN CHARACTERS
0042              ELESPEED RMB   2            LENGTH OF ELEMENT (DIT) DURATION
                  *                             IN MILISECONDS
0044              INITMASK RMB   1            OSCILLATOR CONTROL BYTE
0045              HOLDBYTE RMB   1
0046              MASK     RMB   1
0047              COUNT    RMB   1
0048              CWPTR    RMB   2            POINTS TO 1ST BYTE IN STRING
0100                       ORG   H'100
0100              CWBUFFER EQU   *
0100   A6 00               LDAA  I,0          GET CHAR
0102   81 03               CMPA  #3           IS THIS THE STOP BYTE?
0104   26 01               BNE   CONTINUE     BR IF NO
0106   39                  RTS                RETURN
0107              CONTINUE EQU   *
0107   08                  INX                INCRE PTR
0108   DF 48               STX   CWPTR        STORE POINTER
010A   8D 08               BSR   TRANSMIT     CONVERT THE CHAR TO CW AND TRANSMIT
010C   DE 40               LDX   CHRSPEED     LOAD DELAY CONSTANT FOR CHAR SPACING
010E   8D 46               BSR   MILDELAY     DELAY PROPER # OF MILISECONDS
0110   DE 48               LDX   CWPTR        GET ADDR OF CHAR STRING
0112   20 EC               BRA   CWBUFFER     GET NEXT CHAR
0114              TRANSMIT EQU   *
0114   8D 5C               BSR   CHARTOCW     CONVERT CHAR TO CW TRANSMIT CODE
0116   4D                  TST   A            A = 0 ?
0117   26 06               BNE   NOTSPACE     BR IF NO (VALID NON-BLANK CW CHAR)
0119   C6 01               LDAB  #1           SET NUMBER OF ELEMENTS AT 1, AND
011B   D7 47               STAB  COUNT          MAKE THE WEIGHT SAME AS A DIT
011D   20 25               BRA   SPACENT      BYPASS KEY-ON INSTRUCTION
011F              NOTSPACE EQU   *
011F   16                  TAB                SAVE A IN B
0120                       RPT   4
0120   59                  ROL   B            ROTATE BIT COUNT
0121   59                  ROL   B            ROTATE BIT COUNT
0122   59                  ROL   B            ROTATE BIT COUNT
0123   59                  ROL   B            ROTATE BIT COUNT
0124   C4 07               ANDB  #7           SET COUNT AT < OR = 7
0126   81 FF               CMPA  #H'FF        IS THIS SPECIAL ERROR CODE ?
0128   26 04               BNE   NOTERR       BR IF NOT
012A   4F                  CLR   A            SET DATA TO ALL DITS
012B   5C                  INC   B            INCRE COUNT FROM 7 TO 8
012C   20 06               BRA   COUNTOK      STORE COUNT
012E              NOTERR   EQU   *
012E   C1 06               CMPB  #6           BIT COUNT = 6 ?
0130   2D 02               BLT   COUNTOK      BR IF NO
0132   C4 06               ANDB  #6           SET COUNT AT 6
0134              COUNTOK  EQU   *
0134   D7 47               STAB  COUNT        STORE BIT COUNT
0136              BITLOOP  EQU   *
0136   D6 44               LDAB  INITMASK     INITIALIZE THE
0138   D7 46               STAB  MASK           OSCILLATOR MASK
013A   C6 01               LDAB  #1           LOAD WEIGHT FACTOR FOR DIT
013C   46                  ROR   A            ROTATE DATA BIT INTO CARRY
013D   24 02               BCC   SEND         BR IF CARRY CLEAR (DIT)
013F   C6 03               LDAB  #3           LOAD WEIGHT FACTOR FOR DAH
0141              SEND     EQU   *
0141   73 00 45            COM   HOLDBYTE     TURN ON KEY
0144              SPACENT  EQU   *
0144   8D 22               BSR   ELDELAY      GENERATE DELAY FOR ELEMENT LENGTH
0146   7F 00 46            CLR   MASK         CLEAR OSCILLATOR MASK
0149   7F 00 45            CLR   HOLDBYTE     TURN OFF KEY
014C   C6 01               LDAB  #1           LOAD WEIGHT FACTOR FOR BREAK
                  *                             BETWEEN ELEMENTS
014E   8D 18               BSR   ELDELAY      GENERATE DELAY FOR ELEMENT LENGTH
0150   7A 00 47            DEC   COUNT        LAST ELEMENT IN CHAR ?
0153   26 E1               BNE   BITLOOP      BR IF NO
0155   39                  RTS                RETURN
0156              MILDELAY EQU   *
0156   D6 46               LDAB  MASK         HOLD THE KEY OUTPUT CONSTANT
0158   D8 45               EORB  HOLDBYTE       AND FLIP-FLOP THE
015A   D7 45               STAB  HOLDBYTE         OSCILLATOR OUTPUT IF
015C   F7 80 04            STAB  L,H'8004               KEY IS ON
015F   C6 8F               LDAB  #H'8F        LOAD MILISECOND LOOP CONSTANT
0161              LOOP2    EQU   *
0161   5A                  DEC   B            COUNT DOWN
0162   26 FD               BNE   LOOP2        BR IF NOT FINISHED
0164   09                  DEX                DECRE MILISECOND COUNT
0165   26 EF               BNE   MILDELAY     BR IF NOT FINISHED
0167   39                  RTS                RETURN
0168              ELDELAY  EQU   *
0168   DE 42               LDX   ELESPEED     LOAD DELAY CONST FOR ELEMENT SPACING
016A   37                  PSHB               SAVE    # OF INTERVALS
016B   8D E9               BSR   MILDELAY     DELAY PROPER # OF MILISECONDS
016D   33                  PULB               RESTORE # OF INTERVALS
016E   5A                  DEC   B            DECRE   # OF INTERVALS
016F   26 F7               BNE   ELDELAY      BR IF NOT FINISHED
0171   39                  RTS
0172              CHARTOCW EQU   *
0172   84 7F               ANDA  #H'7F        STRIP PARITY BIT
0174   80 20               SUBA  #32          SUBTRACT OFFSET
0176   2D 17               BLT   INVALAS      IF INVALID CHAR OR = 0 BRANCH OUT
0178   81 40               CMPA  #LASTASCW    CHARACTER IN TABLE ?
017A   2C 0D               BGE   NOTFNDAS     BR IF NO
017C              GETENTRY EQU   *
017C   5F                  CLR   B            ALLOCATE AND ZERO OUT
017D   37                  PSHB                 A TEMPORARY 2-BYTE AREA ON
017E   37                  PSHB                   TOP OF THE STACK
017F   30                  TSX                POINT INDEX REGISTER TO IT
0180   A7 01               STAA  I,1          STORE TABLE OFFSET INTO SECOND BYTE
0182   EE 00               LDX   I,0          LOAD CONSTRUCTED POINTER INTO INDEX
0184   A6 00               LDAA  I,0          LOAD CW CODE
0186   31                  INS                DELETE TEMPORARY HOLD AREA
0187   31                  INS                  & REALIGN STACK
0188   39                  RTS                RETURN
0189              NOTFNDAS EQU   *
0189   80 20               SUBA  #32          POINT TO LOWER CASE CHARS
018B   81 20               CMPA  #H'20        BETWEEN UPPER & LOWER CASE ?
018D   2E ED               BGT   GETENTRY     BR IF NO (USE LOWER CASE AS UPPER)
018F              INVALAS  EQU   *
018F   4F                  CLR   A            SET A TO CW NULL
0190   39                  RTS                RETURN
```

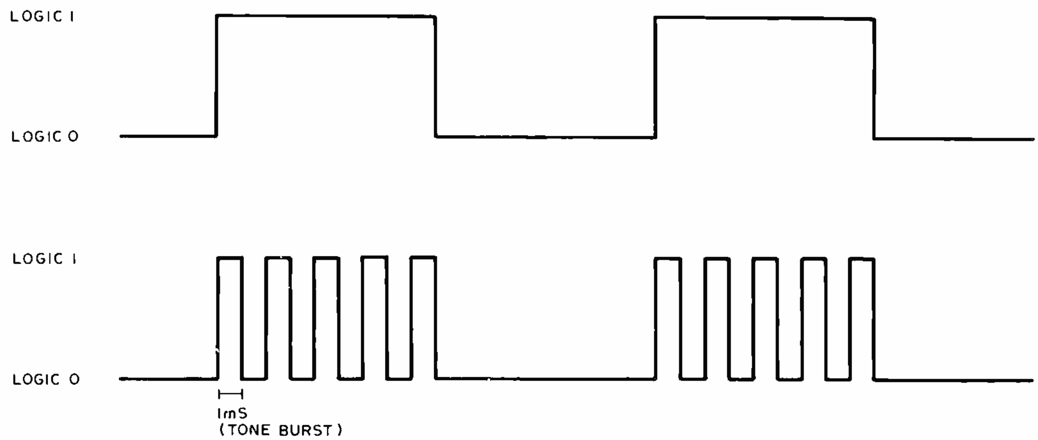


Figure 3: Output Signals from the Morse code generation program. This program will produce either or both of a tone burst "oscillating" output which can be used to drive a speaker for code practice, or a steady logic level output (during each dot or dash) used to control the keyer of a typical radio frequency transmitter.

space between them is generated according to the value stored at location CHRSPEED. If element and character speeds are synchronized, the delay caused by CHRSPEED is exactly twice the space between elements, which has already been generated after the last element. Combining the two causes the total space between characters to be exactly the length of a dash, the established spacing between Morse code characters (see figure 2).

If a space (in ASCII, hexadecimal 20) is detected, it is treated as a phantom character consisting of one dot length with the key off. When added to the 3 dot interval following the previous character, and the equal interval following itself, a total interval of 7 dot times is generated, the standard spacing between words. (ASCII codes with no Morse code equivalents are also treated as spaces.)

Speed Control

There are two separate controls for Morse code spacing: element speed and character speed. The element speed is the time duration of the smallest element of the Morse pattern. The time duration of dashes, dots, and the breaks in between are based upon it. If normal weighting is used, a dot is equal to one of these intervals, a dash is equal to three of them, and the space between dots and dashes within a character is equal to a single dot interval. The 2 byte field labelled ELESPEED contains a 16 bit binary number specifying the element delay as an integer number of milliseconds. With normal weighting, the duration of a dot will be exactly the number of milliseconds contained in ELESPEED, and the duration of a dash will be exactly three times that.

If you wish the element and character speeds synchronized (20 wpm characters sent at 20 wpm intervals, etc), the binary value stored in CHRSPEED should be exactly double the binary value stored in ELESPEED. In this configuration the program will generate perfectly synchronized Morse code according to the established standard (dash duration three times that of dot, duration of space between elements equal to one dot, duration of space between characters equal to that of one dash, and duration of space between words seven times that of one dot). When you change speeds, change the values of ELESPEED and CHRSPEED (once again, the value of the latter should be twice the value of the former.)

If you wish to lengthen the interval between characters without changing their internal speed, simply increase the value of CHRSPEED. For instance, if you wish to practice copying code, you can set the ELESPEED field at the value for 20 wpm and the CHRSPEED field to the 5 wpm value. This will cause 20 wpm characters to be sent at 5 wpm intervals. You can tweak either speed to any 16 bit value you want (except zero), but the value of CHRSPEED must *never* be equal to less than two times that of ELESPEED or the spacing will be demolished!

And It Comes Out Here

The Morse code generator program is designed to output the Morse signals through a parallel IO port. Two different types of output are available simultaneously: logical and oscillating. The logical output corresponds to the Morse signal as it is broadcast — the output is high during a dot or dash and low in the times between. This corresponds to the telegraph key itself and may be fed to the transmitter directly or via a relay or other driver. The oscillating output changes state once every millisecond while the logical output is high and is held low during the times the logical output is low. This output, when connected to a speaker, produces a 500 Hz tone and can be used as a sidetone or a code practice oscillator. Both types of output can be produced simultaneously, as seen in figure 3.

Which of the lines in your parallel port are to be used at logical outputs and which are oscillators is determined by the control byte labelled INITMASK. For every bit in INITMASK which is equal to zero, the corresponding bit in the parallel port is a logical output. Every bit in INITMASK which is in an on state causes the corresponding line in the output port to oscillate.

I am not a hardware type person; therefore, I am not going to attempt describing the interface necessary to take the Morse code output of the nonoscillating line and transfer it to your ham rig, especially since every transmitter has its own keying system. In a classic copout I say, "It is up to the user to take the logical output of the PIA (0=key off, +5 V=key on) and get it onto the air without blasting the PIA output driver."

Viva Southwest Tech

This Morse data generator was written for a Southwest Technical Products 6800 system, which uses the Motorola MC6830L7 ROM (MIKBUG Revision 9). A complete cross assembly of CWBUFFER is printed in listing 1. The program was written to be configuration independent, however, and will work for any 6800 system having programmable memory at locations hexadecimal 00 to 49, at least 190 bytes of programmable memory elsewhere, and one PIA. The ASCII to Morse conversion subroutine is completely relocatable and reentrant, although the main generator routines (CWBUFFER and TRANSMIT) are not.

All of the timing loops are calibrated for the Southwest Technical Products System, which has a 1.797 MHz master oscillator crystal. If your clock runs at a different speed, you may want to tweak the loop constant so that the output of one of the oscillating lines is exactly 500 Hz (each outer loop of the MILDELAY subroutine is supposed to last exactly one millisecond). The loop constant is at hexadecimal location 0160; incrementing or decrementing it by

Listing 2: Various and Sundry Drivers. Three different driver routines occupy the remainder of the assembled code prepared by the author. These drivers are ASDRIVER, SINGLECH, and CODEPRAC. ASDRIVER is used to load a buffer, then call CWBUFFER to transmit the data as Morse signals. SINGLECH is a simple routine to read a character from the keyboard then call the TRANSMIT subroutine (bypassing CWBUFFER) to output that character as its Morse equivalent. CODEPRAC is a routine used to generate a random series of Morse data for code practice. (Note that the instruction at 01D5 should be patched to reference a random number routine specific to your own computer system.)

```
LOC     CODE        SOURCE STATEMENT

0191                ASDRIVER EQU   *
0191    7F 00 45             CLR   HOLDBYTE     CLEAR OUTPUT
0194    7F 80 04             CLR   H'8004           PORT
0197    CE 01 F7             LDX   #BUFFER      LOAD BUFFER ADDRESS
019A                LOADLOOP EQU   *
                    *                           BRANCH TO A SUBROUTINE WHICH INPUTS
                    *                             AN ASCII CHARACTER AND PLACES IT
019A    BD E1 AC             JSR   H'E1AC         INTO ACC A-----FOR SYSTEMS USING
                    *                             MIKBUG THIS SUBROUTINE IS LOCATED
                    *                             AT HEX 'E1AC' (INEEE) ---
                    *                           NON-MIKBUG SYSTEMS SUBSTITUTE A
                    *                             COMPARABLE ROUTINE
019D    A7 00                STAA  I,0          STORE INTO BUFFER
019F    08                   INX                INCRE POINTER
01A0    81 03                CMPA  #3           IS THIS A CONTROL C  (STOP BYTE)  ?
01A2    27 10                BEQ   NONEED3      BR IF YES
01A4    81 0A                CMPA  #H'0A        IS THIS AN ASCII LINE FEED ?
01A6    27 08                BEQ   SENDIT       BR IF YES
01A8    81 08                CMPA  #H'08        IS THIS AN ASCII BACKSPACE COMMAND ?
01AA    26 EE                BNE   LOADLOOP     BR IF NO
01AC    09                   DEX                BACKSPACE POINTER 2 BYTES---1 FOR
01AD    09                   DEX                  DATA BYTE & 1 FOR BACKSPACE CHAR
01AE    20 EA                BRA   LOADLOOP     CONTINUE
01B0                SENDIT   EQU   *
01B0    86 03                LDAA  #3           LOAD AND STORE
01B2    A7 00                STAA  I,0              DELIMITER (STOP BYTE) INTO BUFF
01B4                NONEED3  EQU   *
01B4    CE 01 F7             LDX   #BUFFER      RELOAD BUFFER ADDRESS INTO INDEX
01B7    BD 01 00             JSR   CWBUFFER     CALL CW GENERATOR ROUTINE
01BA    20 D5                BRA   ASDRIVER     START CYCLE ALL OVER AGAIN
01CA                CODEPRAC EQU   *
01CA    7F 00 45             CLR   HOLDBYTE     CLEAR OUTPUT
01CD    7F 80 04             CLR   H'8004           PORT
01D0    CE 01 F7             LDX   #BUFFER      STORE ADDRESS OF BUFFER
01D3    DF 48                STX   CWPTR            INTO START-OF-BUFFER POINTER
01D5                GENLOOP  EQU   *
01D5    BD 00 00             JSR   *-*          CALL A ROUTINE TO GENERATE A 1-BYTE
                    *                             RANDOM NUMBER
01D8    36                   PSHA               SAVE ORIGINAL ASCII CHAR
01D9    BD 01 72             JSR   CHARTOCW     TEST FOR VALID CW CHARACTER
01DC    32                   PULA               GET  ORIGINAL ASCII CHAR
01DD    27 F6                BEQ   GENLOOP      BR IF NOT VALID
01DF    DE 48                LDX   CWPTR        LOAD BUFFER POINTER
01E1    A7 00                STAA  I,0          STORE CHARACTER INTO BUFFER
01E3    08                   INX                INCRE POINTER
01E4    DF 48                STX   CWPTR        SAVE NEW BUFFER POINTER
01E6    8C 7F FF             CPX   #BUFFEND     END OF BUFFER OR END OF MEMORY ?
01E9    26 EA                BNE   GENLOOP      BR IF NO, GET ANOTHER CHARACTER
01EB    86 03                LDAA  #3           LOAD AND STORE
01ED    A7 00                STAA  I,0              DELIMITER (STOP BYTE) INTO BUFF
01EF    CE 01 F7             LDX   #BUFFER      RELOAD BUFFER ADDRESS INTO INDEX
01F2    BD 01 00             JSR   CWBUFFER     GENERATE CW FOR ENTIRE BUFFER
01F5    20 D3                BRA   CODEPRAC     CONTINUE
01F7                BUFFER   EQU   *
7FFF                         ORG   H'7FFF
7FFF                BUFFEND  EQU   *
```

Table 1: Speed Control Values. CWBUFFER uses a 16 bit value, taken from this table, to control the basic number of milliseconds spent in each dot interval of the generated code, and each inter character interval. For standard Morse, the values loaded from the CHRSPEED column should be double the value of the corresponding ELESPEED entry of a given code speed. By increasing the CHRSPEED values, the spacings between characters can be lengthened while preserving the timing relationships for the data rate loaded into ELESPEED. These values are calculated assuming the Southwest Technical Products 6800 system is being used. For other processors with different rates for the master oscillator, a new table would have to be calculated.

Morse code rate (wpm)	Value Loaded into ELESPEED (hexadecimal)	Value Loaded into CHRSPEED (hexadecimal)
5	00D6	01AC
10	006B	00D6
15	0047	008E
20	0035	006A
25	002B	0056
30	0024	0048
35	001F	003E
40	001B	0036
50	0015	002A
60	0012	0024
70	000F	001E
80	000D	001A
90	000C	0018
100	000B	0016
200	0005	000A
500	0002	0004
1000	0001	0002

one will increase or decrease the interval which should correspond to a millisecond by six machine cycles. If you aren't concerned with perfect timing, you can compensate by loading different values into ELESPEED and CHRSPEED.

The parallel port address used in the program corresponds to the serial control interface used by MIKBUG, which is really a parallel interface that only simulates a serial interface via software. Connect a small 8 ohm speaker between connections RO and GND on the serial control interface. (Your computer will probably be unable to talk to the TVT or Teletype while the speaker is attached, due to loading problems.) In the program turn on the rightmost bit in INITMASK to produce the oscillating output.

There is only one output line normally available from the PIA on the Southwest Technical Products serial control interface, although it separates into two output systems, RS-232 and 20 mA current loop. Another line on the PIA's A side is used for MIKBUG input, which leaves six lines completely unused. These lines can be used for Morse output if you bring them out via jumpers from the backside of the PIA. Of course, MIKBUG has designated them as inputs in the A side data direction register; but if you OR a hexadecimal 7E into the DDR, you will reset them as outputs and leave the normal MIKBUG lines alone. You can use any combination of those lines for Morse output.

Sundry Drivers

The Morse Code generator program is designed to be used as a subroutine. It simply takes a character or string of characters in memory and outputs the Morse code equivalents. It is completely up to the user as to how this data is input, whether through the keyboard, read from tape, generated by a random number generator, or conjured up by evil spirits. All that is required is that the Morse code routine's controls (ELESPEED, CHRSPEED, and INITMASK) be set before the subroutine is entered, and, if the CWBUFFER entry point is used, that the index register contains the address of the first byte of the string and the stop byte, hexadecimal 03, follows immediately after the last byte of the string.

I have included three simple drivers which could be used for the 6800: one that generates one Morse character at a time from

Table 2. Special Codes and ASCII Graphics. There are several special case codes used in radio communications with Morse code. These are listed at the left with a short explanation. At the right are shown the ASCII character graphics and hexadecimal codes used by CWBUFFER to represent these special case codes.

To send:		Keyboard Entry	
$\overline{SK}$	End of work	'#'	ASCII hexadecimal 23
$\overline{BT}$	Break	'&'	ASCII hexadecimal 26
$\overline{AR}$	End of Message	'$'	ASCII hexadecimal 24
$\overline{KN}$	Invitation to transmit, specified station only	'+'	ASCII hexadecimal 2B
$\overline{AS}$	Wait	'='	ASCII hexadecimal 3D
ERROR CODE (8 dots)		'!'	ASCII hexadecimal 21 or
		DEL	ASCII hexadecimal 7F

the keyboard; a second that buffers characters until a delimiter character (line feed) is encountered, after which it sends out the characters it has stored in one brilliant blast of precision keying; and a third driver which fills up a buffer with random characters, then sends them out (for code practice). These drivers are shown as listing 2.

The single character driver (SINGLECH) lies dormant until a character is entered from the keyboard. When the character is received, it is passed immediately to TRANSMIT and the Morse code is sent out the PIA. The driver then goes back to sleep until a new character comes in.

The buffered driver loads incoming characters into a buffer in memory until either a line feed or ETX (Control C, hexadecimal 03) character is encountered. When either is received the starting address is loaded into the index register, a delimiter (ETX) is stored in the buffer after the last data byte; and the buffer is passed to CWBUFFER, which outputs the CW codes in a continuous stream until the end of the buffer is reached. Control then returns to the driver, which starts loading the buffer all over again. When a backspace command (Control H, hexadecimal 08) is received, the character immediately preceding the command is deleted from the buffer.

CODEPRAC, the code practice program, fills a buffer in memory with random characters, then passes them to CWBUFFER, which sends them out the PIA into your speaker (hopefully you have set INITMASK so that the line is an oscillating output). The speed and spacing are controlled by whatever you have loaded into ELESPEED and CHRSPEED.

This code practice application is where the dual speed controls really come into play. When a person is first learning code, he obviously has to start at an extremely low character rate. At this speed, the dots and dashes are extremely dragged out and sound

Table 3: ASCII to Morse Conversion Table. This table lists the relative offset, data, assembly language text, ASCII equivalent graphics and ASCII hexadecimal codes (H'xx' for code xx) for each character recognized and generated by CWBUFFER. To find the Morse encoded value of each ASCII graphic, subtract hexadecimal 20 from its ASCII hexadecimal value, add the difference to the first address location of ASCWTABL (0 in this assembly) and use the 16 bit result as the address of the desired Morse equivalent. Information in this table was prepared using the author's "SPUCA" cross assembler.

```
LOC     CODE      STMT    SOURCE STATEMENT

0000                 1    ASCWTABL EQU  *
0000    00           2             FCB  B'00000000'    SPACE        (ASCII H'20')
0001    FF           3             FCB  B'11111111'    ! (ERROR)    (ASCII H'21')
0002    D2           4             FCB  B'11010010'    "            (ASCII H'22')
0003    E8           5             FCB  B'11101000'    # (SK)       (ASCII H'23')
0004    AA           6             FCB  B'10101010'    $ (AR)       (ASCII H'24')
0005    00           7             FCB  B'00000000'    NULL         (ASCII H'25')
0006    B1           8             FCB  B'10110001'    & (BT)       (ASCII H'26')
0007    DE           9             FCB  B'11011110'    '            (ASCII H'27')
0008    ED          10             FCB  B'11101101'    (            (ASCII H'28')
0009    ED          11             FCB  B'11101101'    )            (ASCII H'29')
000A    00          12             FCB  B'00000000'    NULL         (ASCII H'2A')
000B    AD          13             FCB  B'10101101'    + (KN)       (ASCII H'2B')
000C    F3          14             FCB  B'11110011'    ,            (ASCII H'2C')
000D    E1          15             FCB  B'11100001'    -            (ASCII H'2D')
000E    EA          16             FCB  B'11101010'    .            (ASCII H'2E')
000F    A9          17             FCB  B'10101001'    /            (ASCII H'2F')
0010    BF          18             FCB  B'10111111'    0            (ASCII H'30')
0011    BE          19             FCB  B'10111110'    1            (ASCII H'31')
0012    BC          20             FCB  B'10111100'    2            (ASCII H'32')
0013    B8          21             FCB  B'10111000'    3            (ASCII H'33')
0014    B0          22             FCB  B'10110000'    4            (ASCII H'34')
0015    A0          23             FCB  B'10100000'    5            (ASCII H'35')
0016    A1          24             FCB  B'10100001'    6            (ASCII H'36')
0017    A3          25             FCB  B'10100011'    7            (ASCII H'37')
0018    A7          26             FCB  B'10100111'    8            (ASCII H'38')
0019    AF          27             FCB  B'10101111'    9            (ASCII H'39')
001A    C7          28             FCB  B'11000111'    :            (ASCII H'3A')
001B    D5          29             FCB  B'11010101'    ;            (ASCII H'3B')
001C    00          30             FCB  B'00000000'    NULL         (ASCII H'3C')
001D    A2          31             FCB  B'10100010'    = (AS)       (ASCII H'3D')
001E    00          32             FCB  B'00000000'    NULL         (ASCII H'3E')
001F    CC          33             FCB  B'11001100'    ?            (ASCII H'3F')
0020    00          34             FCB  B'00000000'    NULL         (ASCII H'40')
0021    42          35             FCB  B'01000010'    A            (ASCII H'41' OR H'61')
0022    81          36             FCB  B'10000001'    B            (ASCII H'42' OR H'62')
0023    85          37             FCB  B'10000101'    C            (ASCII H'43' OR H'63')
0024    61          38             FCB  B'01100001'    D            (ASCII H'44' OR H'64')
0025    20          39             FCB  B'00100000'    E            (ASCII H'45' OR H'65')
0026    84          40             FCB  B'10000100'    F            (ASCII H'46' OR H'66')
0027    63          41             FCB  B'01100011'    G            (ASCII H'47' OR H'67')
0028    80          42             FCB  B'10000000'    H            (ASCII H'48' OR H'68')
0029    40          43             FCB  B'01000000'    I            (ASCII H'49' OR H'69')
002A    8E          44             FCB  B'10001110'    J            (ASCII H'4A' OR H'6A')
002B    65          45             FCB  B'01100101'    K            (ASCII H'4B' OR H'6B')
002C    82          46             FCB  B'10000010'    L            (ASCII H'4C' OR H'6C')
002D    43          47             FCB  B'01000011'    M            (ASCII H'4D' OR H'6D')
002E    41          48             FCB  B'01000001'    N            (ASCII H'4E' OR H'6E')
002F    67          49             FCB  B'01100111'    O            (ASCII H'4F' OR H'6F')
0030    86          50             FCB  B'10000110'    P            (ASCII H'50' OR H'70')
0031    8B          51             FCB  B'10001011'    Q            (ASCII H'51' OR H'71')
0032    62          52             FCB  B'01100010'    R            (ASCII H'52' OR H'72')
0033    60          53             FCB  B'01100000'    S            (ASCII H'53' OR H'73')
0034    21          54             FCB  B'00100001'    T            (ASCII H'54' OR H'74')
0035    64          55             FCB  B'01100100'    U            (ASCII H'55' OR H'75')
0036    88          56             FCB  B'10001000'    V            (ASCII H'56' OR H'76')
0037    66          57             FCB  B'01100110'    W            (ASCII H'57' OR H'77')
0038    89          58             FCB  B'10001001'    X            (ASCII H'58' OR H'78')
0039    8D          59             FCB  B'10001101'    Y            (ASCII H'59' OR H'79')
003A    83          60             FCB  B'10000011'    Z            (ASCII H'5A' OR H'7A')
003B    ED          61             FCB  B'11101101'    (            (ASCII H'5B' OR H'7B')
003C    00          62             FCB  B'00000000'    NULL         (ASCII H'5C' OR H'7C')
003D    ED          63             FCB  B'11101101'    )            (ASCII H'5D' OR H'7D')
003E    00          64             FCB  B'00000000'    NULL         (ASCII H'5E' OR H'7E')
003F    FF          65             FCB  B'11111111'    DEL (ERROR)  (ASCII H'5F' OR H'7F')
0040                66    LASTASCW EQU  *-ASCWTABL
```

ASCWTABL	0000	LASTASCW	0040	CHRSPEED	0040	ELESPEED	0042	INITMASK	0044
HOLDBYTE	0045	MASK	0046	COUNT	0047	CWPTR	0048	CWBUFFER	0100
CONTINUE	0107	TRANSMIT	0114	NOTSPACE	011F	NOTERR	012E	COUNTOK	0134
BITLOOP	0136	SEND	0141	SPACENT	0144	MILDELAY	0156	LOOP2	0161
ELDELAY	0168	CHARTOCW	0172	GETENTRY	017C	NOTFNDAS	0189	INVALAS	018F
ASDRIVER	0191	LOADLOOP	019A	SENDIT	01B0	NONEED3	01B4	SINGLECH	01BC
SINGLOOP	01C2	CODEPRAC	01CA	GENLOOP	01D5	BUFFER	01F7	BUFFEND	7FFF

Table 4: Label Table. This table, also prepared with the author's cross assembler, gives the address (hexadecimal) for each symbol in CWBUFFER and the sundry driver programs.

completely different than they do at higher speeds. The Morse code neophyte should really learn to recognize the characters by listening to the total pattern, not by counting dots and dashes. However, speeding up the characters also speeds up the character rate if normal element to space ratios are maintained and an entire message has gone by while the beginner is still trying to recognize the first character. Therefore, the ideal situation is to retain the sound of the high speed characters and yet increase the interval between them. This is accomplished by leaving the element speed (ELESPEED) the same and increasing the duration of time between characters (CHRSPEED) to whatever length is desired.

Anyway, to get back to the code practice driver, it will continue to send characters at the CHRSPEED rate until the end of the buffer is reached, at which time it will generate another set of practice characters. Unfortunately, the random number generation routine itself is missing. The code practice driver was a last minute addition, and there was not time to develop one. (The program was tested using a kluge substitution.) However, there are several different versions floating around. The one you use should generate a one byte random number and return it in accumulator A. It should not destroy accumulator B or the index register. Load the address of the routine in the dummy jump to subroutine at hexadecimal location 01D5. The address at location 01E7 is the end of buffer address or your maximum memory location. Don't forget to leave a place for the random number generator subroutine.

The drivers in this article are very primitive, and are designed simply to get you running. If your system is a Southwest Tech one, you should be able to load all of the programs and the table into your system exactly as coded (CODEPRAC too, with the addition of a random number routine), connect a speaker between RO and ground on the control interface, load ELESPEED and CHRSPEED according to table 1, turn on the rightmost bit of INITMASK, branch to the starting address of the driver of your choice, and start typing. As you hear the speaker sing a 500 Hz aria with perfect 1 to 3 to 7 spacing, you may reflect that maybe programmers aren't such bad guys after all. Seriously, the sidetone output is a perfect way to ensure that the program is working with each of the various drivers before you try to tie it into your rig.

Other 6800 Systems

For non Southwest Tech 6800 systems, the installation of the Morse generator program is not much more difficult. The address of the PIA used for the Morse code output and that of the input routine are the primary concerns. If your system uses MIKBUG, which has the single character input routine at hexadecimal location E1AC and the serial control interface at hexadecimal 8004, so much the better – you shouldn't have to modify a single byte. If your configuration is different, substitute your own PIA address at hexadecimal locations 015D, 0195, 01C0, and 01CE, and your ASCII input routine address at locations 01C3 and 019B.

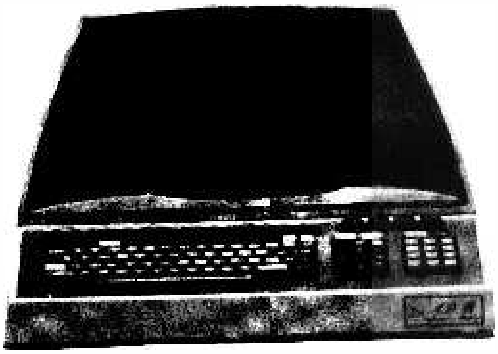
The programs in this article were not assembled on Motorola's assembler. They were run on SPUCA (Sewell's Psychedelic Universal Cross Assembler), a homemade cross assembler which runs on the IBM 370 and generates code for the 6800 and four other microprocessors. Listings 1 and 2, and tables 3 and 4 were generated by SPUCA.

The formats are almost identical, but there are some minor differences that should be pointed out for the sake of clarity. The major difference is in instructions using the indexed mode of addressing. Motorola places the symbol for the index register (X) after the operand with a comma in between, where SPUCA looks for an I *before* the operand, again with a comma in between. In other words, a Motorola indexed instruction looks like this: LDA A OPERAND,X and one read in by SPUCA is in this format: LDAA I,OPERAND. In addition, the programmer must specify explicitly whether direct (base page, one byte address) or extended (two byte) addressing is to be used. If the extended addressing is to be used, the operand must be preceded by an L, (for Long). Example: LDAA L,OPERAND.

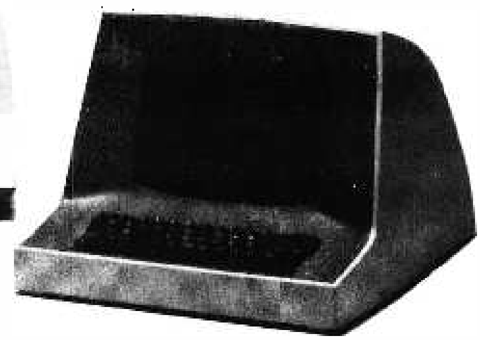
Most of the rest of the instructions are identical to Motorola's except that my assembler has no FCC (Form Constant Characters) directive and no separate column for the accumulator ID.■

In addition to the ADM-3 Dumb Terminal Kit we now offer you the complete ADM-SERIES

ADM-3



ADM-2



ADM-1

ADM-1 FEATURES

DISPLAY FORMAT: 960 characters arranged in 80 characters per line by 12 lines.
OPTIONAL SCREEN SIZE: 1920 characters on a 12" screen, 80 characters per line by 24 lines.
SCREEN: 12-inch Diagonal, P4 Phosphor. Etched face plate.
CHARACTER SET: 64 Alphanumeric USASCII characters.
CHARACTER FONT: 5 x 7 Dot Matrix.
CURSOR: Reverse Image (Block Cursor).
CURSOR CONTROLS: Skip, Backspace, Forespace, Down, Return, New Line, Up, Home, Read Cursor, and Cursor Addressing.
EDITING FEATURES: Clear Screen, Overstrike, and Absolute Cursor Addressing/Read.
FIELD PROTECT: Screen formatting of protected and unprotected fields displayed in dual intensity.
KEYBOARD: 60-Key TTY Standard configuration with cursor control keys, Tab, Home, and Clear keys.
INTERFACE: RS-232 point-to-point.
Data Rates — 110, 300, 600, 1200, 1800, 2400, 4800, 9600.
Parity — Even, odd, one, or zero.
EDIT OPTION: Character insert and delete, line insert and delete (when not in protected mode), clear to end of line/field/page, back tab, and message mode transmission.
OPTIONAL: Numeric 10-key pad with return and decimal.
TRANSMISSION OPTIONS: Polling-addressing, and RS-232 extension.
OPTIONAL: Serial Printer output.
SIZE: 12" high x 16" wide x 21" long.
LOGIC CARD SIZE: 19.2" x 13.3"
WEIGHT: 45 pounds.
POWER REQUIREMENTS: 115 Vac, 60Hz, 100 watts.

ADM-2 FEATURES

- Full 128 ASCII character set
- 1920 Character display
- 8 Transmission rates
- 16 Function keys for 32 commands
- 8 Status displays on the screen
- 5 Mode keys lighted
- 5 Block transmission modes
- Separate Keyboard — 118 keys
- 10-Key Numeric keyboard
- 5 Separate cursor control keys
- Dynamic control of Conversation/Block mode
- Program mode
- Single key edit operations
- Page, field, or line edit
- Security fields
- Protected fields
- Blinking fields
- Dual intensity
- Field tab
- Column tab
- Cursor addressing & cursor read
- Repeat: Repeat from the keyboard is 15 characters per second when a key is held down.
- Control Characters: Control characters are entered in memory if program mode is on or if they are preceded by ESCAPE.
- Interface: RS-232C point-to-point
  Optional 20mA current loop
  Optional RS-232C extension (multidrop)
  Optional RS-232C to a printer
- Size: 12" high x 20" wide x 24" long
- Weight: 50 pounds.
- Power 115 VAC, 60 Hz, 120 watts.

ADM-3 FEATURES

- Full- or half-duplex operation at selectable data rate (75, 110, 150, 300, 600, 1200, 1800, 2400, 4800, 9600, or 19,200 baud)
- 20mA current loop and EIA standard RS-232C interfaces
- Extension RS-232C interface port for hard copy printer, magnetic tape recorder or additional data terminals
- 59-key keyboard
- Bright, high contrast characters displayed in the familiar 5 x 7 dot matrix
- Bottom line data entry with upward page scroll
- End-of-line audible tone
- Options:
  24-line display
  Numeric keypad
  "Answer Back" capability
  Independently selectable
  Transmit and receive rates

Data Entry

New data enters on bottom line of screen; line feed causes upward scrolling of entire display with top-of-page overflow.
Automatic new line switch selectable

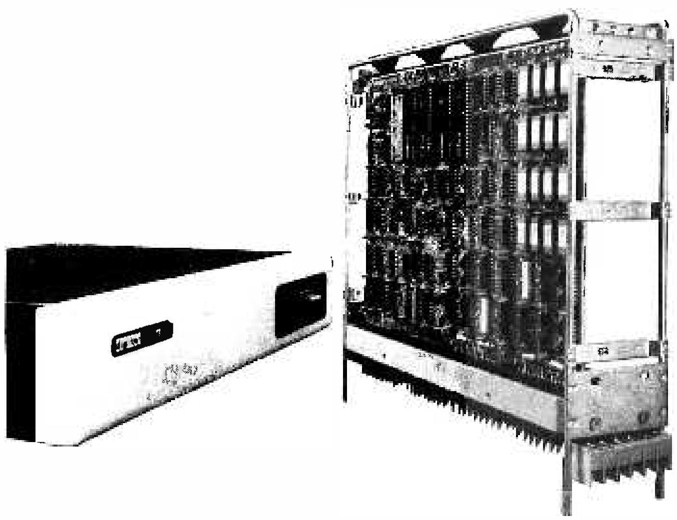
Refresh Rate

60 Hz standard; 50 Hz with input power option
Switch selectable

- Compact size: 15½" x 19" x 12½"
- Weight: 25 pounds.
- Power 115 VAC, 70 watts.

SYNCHRO

STEP UP TO DEC



PDP 11/03-LSI-11 Central Processing Unit

LSI-11 has. . . . . . .

4 K X 16 random access memory
16 bits I/O port DMA port
Power Fail/auto restart
Real time clock input
Automatic priority interrupt
Vector interrupt handling

PDP 11/03

Self contained version, with power supply, front panel switches, real time clock, card cage, including CPU & 4 K RAM.

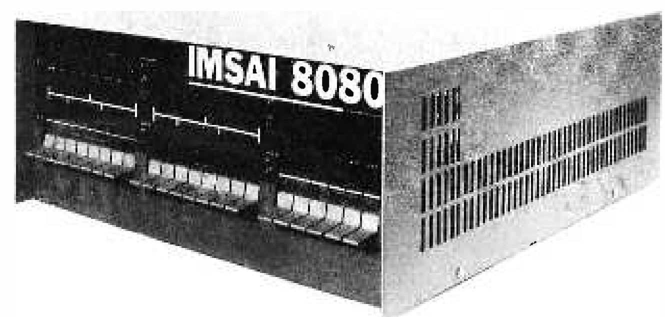
DISCOUNTS ON OTHER
IMSAI products, Lear Siegler Terminals and
DEC components.

We have the Z-80 also.

193-25 JAMAICA AVE.
HOLLIS, N. Y. 11432
(212) 468-7067 (9 AM - 5 PM)

SOUND

THE COMPLETE MICROPROCESSOR SYSTEM



IMSAI 8080 Microcomputer

POWERFUL • EASY TO USE • LOW COST

Please rush me the following items

☐ ADM-3K (Kit)	(24 X 80)	$ 849.95
☐ ADM-3 Assembled	(12 X 80)	$ 949.95
☐ ADM-3 Assembled	(24 X 80)	$1099.95
☐ ADM-2 Assembled	(24 X 80)	$2295.00
☐ ADM-1 Assembled	(24 X 80)	$1429.00
☐ IMSAI 8080 Microcomputer (Kit)		$ 559.95
☐ LSI-11		$ 840.00
☐ PDP 11/03		
☐ DECWRITER LA 36 II 30 cps printer		$1769.95
☐ Z-80		$ 242.00
☐ 8 K low power RAM board		$ 279.00

Enclosed is my . . .

☐ Cashiers check
☐ Money order
☐ Personal check
☐ Master Charge

$100 dollar deposit required for COD

AUTHORIZED DISTRIBUTORS

Name ______________________

Address ______________________

City ____________ State ____________ Zip ____________

New York residents add 8% sales tax

A Morse Code Station Data Handler

Bruce Filgate
Digital Equipment Corporation
Components Group Engineering
One Iron Way
Marlborough MA 01752

For some time, there has been a need in amateur radio for a machine that could both decode and generate Morse code; in addition, the decoder had to be capable of automatically tracking varying received code speeds. Although Morse code keyboards have been around in the amateur radio field for some time, decoders have not been so readily available. Since hardwired logic can be difficult to modify, I decided to implement the coder and decoder in software. Since a low price was desirable and high performance was not required, I used Digital Equipment Corporation's MPS Starter Set. This is an Intel 8008-1 based product which DEC has been marketing to the commercial world. This article describes my implementation using MPS.

Implementation

The main program consists of a few subroutine calls to the main tasks, as shown in listing 1. As can be seen in the source code of the monitor, there are two principal routines that provide for Morse input and output; these two routines are INPEND and OTPUT respectively. The other routines within the main modules provide housekeeping and data manipulation for these IO drivers.

It should be noted, on consulting the full program listing, listing 2, that there are two data tables: an ASCII table and a Morse data table. The only restrictions on modifying these tables are that both tables be the same size and in the same sequence. For instance, the ASCII table could be changed to reflect a non-ASCII code so that the console could be other than an ASCII terminal.

Listing 1: Monitor Entry and Supervisor Main Task. This listing shows the symbolic assembly language representation of the outer loop of the Morse code program. The detailed assembly is found in listing 2 along with the rest of the program.

```
RESTRT,   CAL   INPEND    try code input line;
          CAL   KYBD      try the keyboard task;
          CAL   PNTR      try the printer task;
          LHI   CMMND↑    test the mode byte;
          LLI   CMMND
          XRA             zero the A reg and flgs;
          ADM             mode byte to A reg and flgs;
          JFZ   CMMNDR    enter command mode;
          CAL   IDLE      non command character feedthrough;
          CAL   OUTPUT    anything morse to output?
          JMP   RESTRT    and loop (?) along;
```

Principle of Code Manipulation

A search of published literature available at the time this program was created showed an excellent method of representing Morse code in memory, particularly in eight bit wide memory. This method, found on page 13 of the July 1975 *QST* magazine, was adopted here for internal representation of Morse code. Binary ones represent dashes, binary zeros represent dots and a final binary one closes the character. For instance, the Morse character B(— • • •) would be represented by the octal number 210 (binary 10001000). Note that the binary strings are left justified with trailing zeros. It can be seen that if this representation is to be used to generate a Morse character B, the bit string should simply be shifted out to the left with each bit interpreted as a dash or a

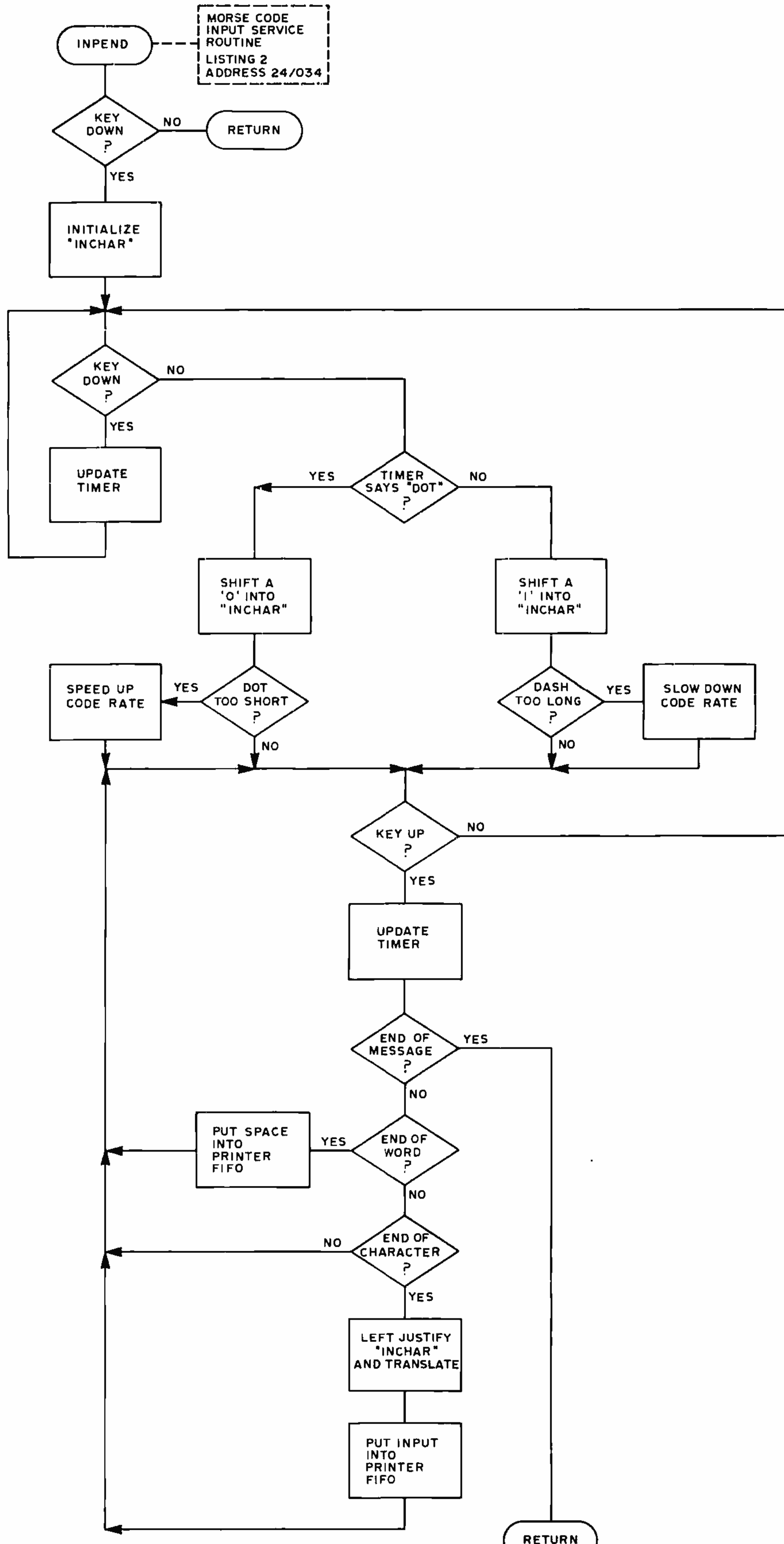


Figure 1: Flow Chart Detail of INPEND, the Morse code input service routine. This routine, beginning at address 24/034 in listing 2, is responsible for tracking and adapting to the variations in speed of human generated Morse code inputs. This routine also detects end of character and end of word gaps between Morse inputs. The character outputs are translated and sent to the printer buffer maintained by the program.

dot until a final result in the shift register is 200 octal. Thus 200 octal remaining in the processor's accumulator is used to signify the end of a character.

To transmit an ASCII keyboard character out in Morse code, all that is required is to look up the ASCII character in the ASCII table and compute its relative location within the table. Once the relative location is calculated, it is necessary to look in the Morse table in the same relative location and pull the Morse equivalent of the ASCII character out of the table. Once this code is available, it is loaded into a register and shifted out to the left as dashes and dots until the register contains octal 200. Conversely, data being received in Morse code starts a character by preloading of a register with octal 200 and then the shifting in of a bit at a time of dash and dot information in ones and zeros. When an intercharacter delay time is finally recognized, the character is considered "closed" and the data that is in the register is left justified and looked up in the Morse code table. When it is found in the Morse code table, the same relative address within the ASCII table contains the ASCII character equivalent. (See figure 1 for a flow chart of the code input service routine.)

Morse Code Speed Determination

When code is being transmitted by the program, the code output speed is selected by a keyboard control sequence. This control sequence consists of the escape (ESC) key followed by a 'W' and then a second character obtained from table 1. Once the

Table 1: The list of ASCII character graphics and their equivalent speed settings when used with the speed change command. The command sequence is: escape character, W, speed character. Thus the sequence escape, W, + sets the speed to 18 words per minute.

Speed Character	Rate (wpm)
SP	120
!	89
"	63
#	48
$	44
%	37
'	27
)	22
+	18
/	14
3	12
>	7.6
?	7.2

Software Availability in Machine Readable Form

The software for this program in source and binary form has been submitted to the DECUS library. DECUS is the Digital Equipment Corporation User's Society, managed by a board of directors composed of DEC equipment users. Free membership in the society is open to users of DEC equipment. The group periodically publishes indices of currently available programs with abstracts.

The documentation for this program is listed under the number 8-801. Since it is expected that some users will want to modify this program, the source in ASCII is available on paper tape from the DECUS library. To order copies of the ASCII source tape or the binary tape, write to:

Program Librarian
DECUS
146 Main St
Maynard MA 01754

as of the time of writing, the following prices apply:

8-801	Binary paper tape	$2
8-801	ASCII source paper tape	$8
8-801	DECUS writeup	$1
8-801	Assembly listing	$10

output speed is selected by this method it will remain constant until a new value is selected.

Since the DEC M7341 processor card uses a variable speed clock, table 1 is calculated assuming that the clock be operated at the same speed as the clock on the author's system. If the clock for the processor on which the software is running is operated at a different rate, then table 1 must be recalculated.

If it is desired that the user's clock be set at the same speed as the clock found in the author's processor, a method of calibrating the clock is offered here: If the program is set up to transmit a sequence of dots, for instance the character "five" which is five dots sequentially, a string of "fives" will generate five dots followed by three dot times between characters, five more dots for the next "five" and so on. This, in consideration with the equation:

$$\text{Speed (wpm)} = \frac{\text{dots per minute}}{25} = 2.4 * \text{dots per second}$$

can be used to compute the effective code rate in words per minute.

The code input speed is never actually calculated but instead a rather heuristic tracking technique is used to update what

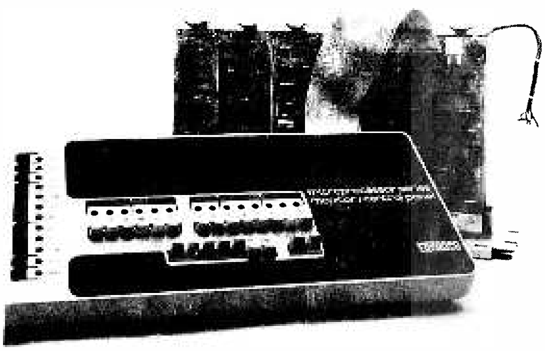


Photo 1: The Digital Equipment Corporation's MPS starter set used by the author for running the Morse code program. Further detail of the central processor board is shown in photo 2.

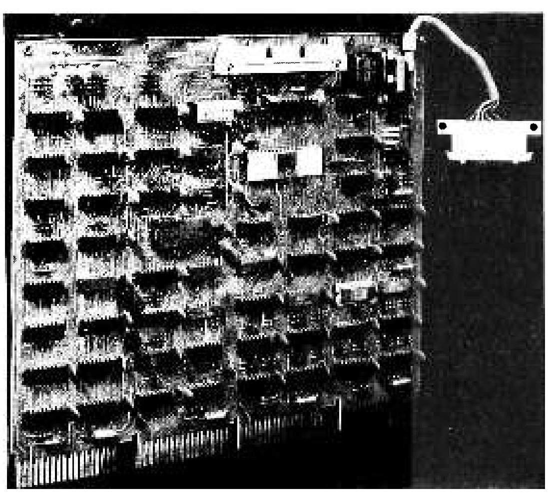


Photo 2: The Digital Equipment Corporation M7341 central processor board. The 8008 processor is socketed at the lower right. This board also includes a UART for serial interface and the random logic needed to buffer and drive an 8008.

Information about the Hardware Used

This program is designed to generate and decode Morse code. Although the program is intended to be executed on Digital Equipment Corporation's MPS M7341, it should also execute on almost any Intel 8008-1 based microprocessor with only slight modification. Photos 1 and 2 are supplied by Digital Equipment Corporation.

The following literature is available from:

Digital Equipment Corporation
Communications Services
444 Whitney St
Northboro MA 01532

General Interfacing Techniques for the M7341 Microprocessor Module
Interfacing The TU60 to the MPS M7341 Microprocessor
M7341 Processor Module Data Sheet
M7344-YA, –YB, –YC Read-Write Memory Module Data Sheet
M7345 Programmable Read Only Memory Data Sheet
M7346 External Event Detection Module Data Sheet
M7328 Evoke Decoder Module Data Sheet
M1501 Bus Input Interface Module Data Sheet
M1502 Bus Output Interface Module Data Sheet
Logic Handbook
KMP01A Microprocessor Series (MPS) Pre-wired Backplane Appl. Note
MR873 Microprocessor ROM Programmer Product Bulletin
1976 Direct Sales Catalog

the processor "judges" the current code rate to be. Therefore, the processor arbitrarily selects the initial code speed to be about 15 words per minute and, if it deems that the input code speed is other than that, a change will be made in the appropriate direction until the processor is able to synchronize against the incoming code speed. At this time, small changes in the code rate will be made to insure that the code speed remains within the tracking range. In addition to decoding the dot and dash times, the program must also be able to decode the times between characters, between symbols and even at the end of sentences. Additionally an arbitrary time is selected which is deemed to be an end of message; these times are set to be a function of a dot time. Thus, as the processor works to synchronize the code speed, all that is required is to keep track of a counter which represents the length of a dot. If the dot time is increased, thus decreasing the code speed, all the other times will be affected in a similar direction.

Keyboard Monitor

In general, characters typed on the keyboard are immediately translated to the Morse code bit string and then transmitted

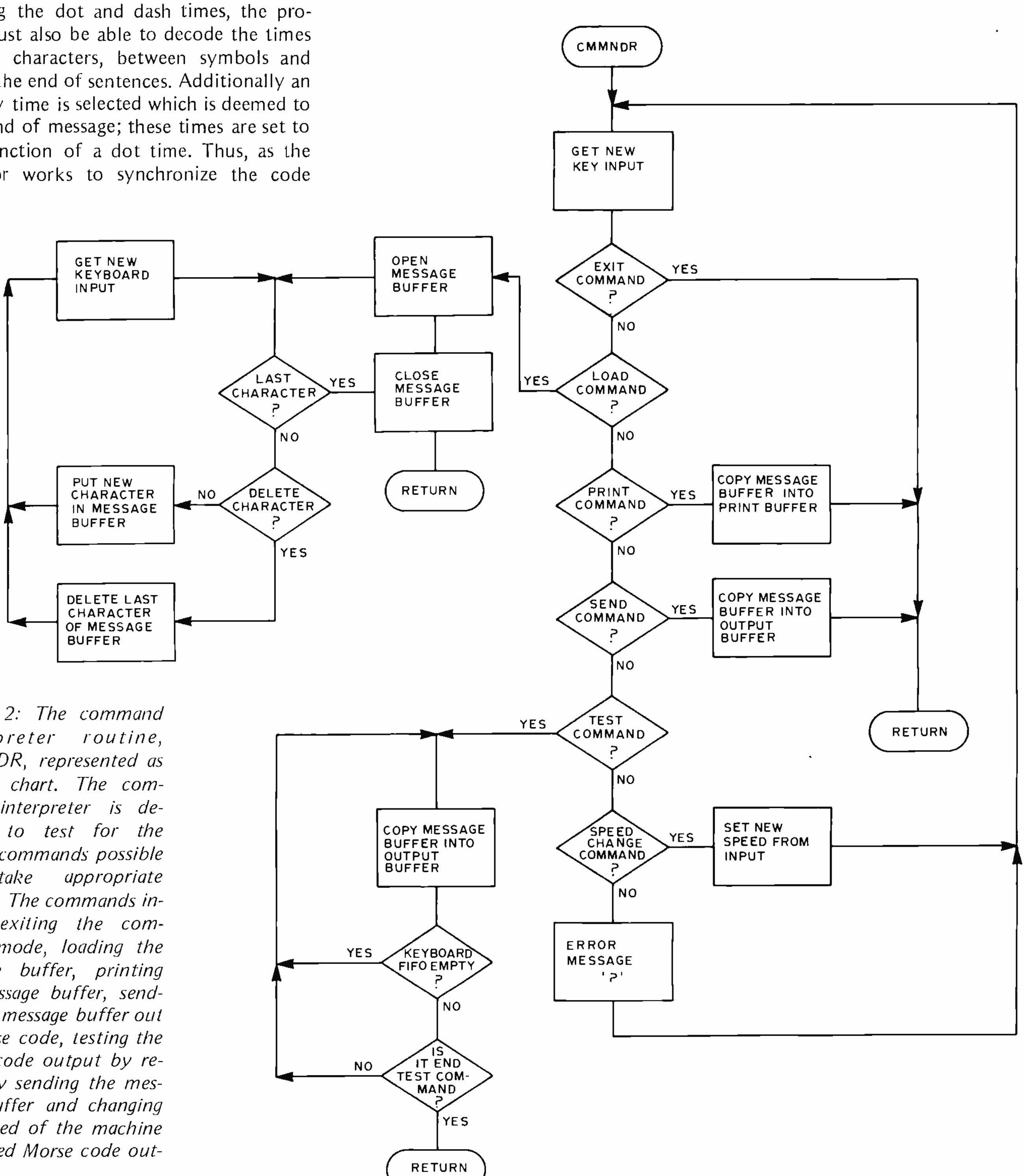


Figure 2: The command interpreter routine, CMMNDR, represented as a flow chart. The command interpreter is designed to test for the several commands possible and take appropriate actions. The commands include exiting the command mode, loading the message buffer, printing the message buffer, sending the message buffer out as Morse code, testing the Morse code output by repeatedly sending the message buffer and changing the speed of the machine generated Morse code outputs.

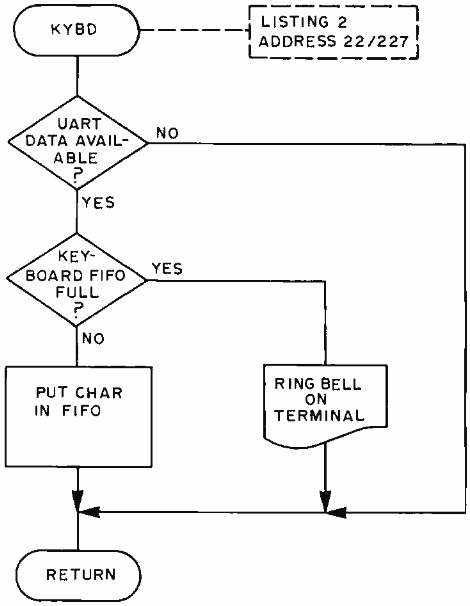


Figure 3: The keyboard service routine, KYBD, specified as a flow chart. This routine is simple: Look at the UART and see if data is available. If so, then stuff the data into the keyboard input buffer, or ring the bell and ignore if the buffer was full.

out in Morse code. There are some special commands interpreted as shown in figure 2 to modify this normal mode:

- <ESC> L is used to reload the message buffer until an escape (<ESC>). (LOAD command)
- <ESC> is used to return to normal keyboard mode. (EXIT command)
- <ESC> P causes the message buffer contents to be printed. (PRINT command)
- <ESC> S is used to ship the message buffer out, translated in Morse code. (SEND command)
- <ESC> T is a test to do <ESC> S until an escape is typed. (TEST command)
- <ESC> W loads a new rate constant for WPM output. A character obtained from table 1 sets the rate, and should follow the W. (Speed change command)
- The delete (rub out) can be used in a buffer loading mode to remove previous characters back to the beginning of the buffer.

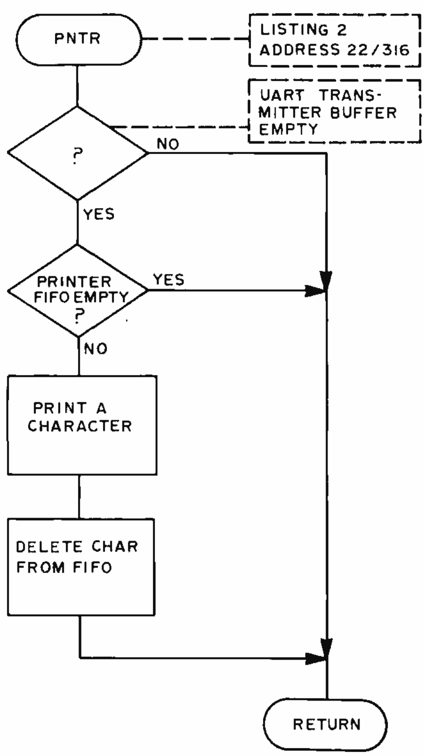


Figure 4: The printer service routine, PNTR, specified as a flow chart. This also is very straightforward: Look at the UART and see if it is ready to transmit. If so, then if the printer buffer is not empty send a character to the UART and remove the character from the printer buffer.

Software Buffer

This program uses overlaid IO to allow several operations to be done at the same time. First in, first out buffering is implemented to keep track of data transferred from or to the buffer locations. Four effective sources or destinations are involved: the message holding buffer, the code output buffer, the keyboard buffer (see figure 3 for the keyboard buffer service routine), and the printer buffer (see figure 4 for printer buffer service routine). Each buffer uses its first location as a byte count of the number of locations occupied by valid data. Since this is an eight bit machine, and single precision arithmetic is used, it is obvious that the maximum number of available data locations in a buffer can not exceed octal 377; any attempt by the user to exceed octal 377 locations or to exceed the maximum buffer size results in an error due to buffer overflow. When the program senses a buffer overflow condition, the terminal bell (or bell indicator) will be rung and the character that would have caused the overflow will be ignored. Since the user may choose to modify the sizes of these buffers, the following advice is offered: The message buffer should be as large as practically possible, followed by the keyboard buffer and then the printer buffer in priority.

Hardware Configuration

This piece of software has been designed and constructed so it can operate without modification on DEC Logic Products Starter Set 1 (KMP01 based). This starter set contains adequate memory, hardware, and interfacing to allow the software to execute properly. When executed on the starter set, the least significant bit of the input byte from input device 2 is used as the sense line for code input. The entire byte on output device 4 is used for code output (any particular bit on this output channel may be

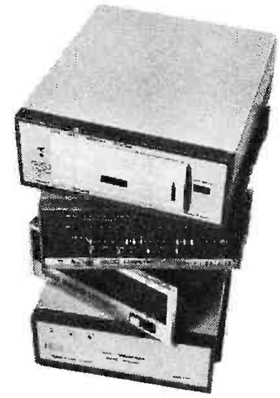
Text continued on page 70

Listing 2: Complete Assembly of the Morse Code Program for an 8008. This listing was prepared using a cross assembler available to the author at Digital Equipment Corporation. The listing is reproduced here in its entirety, with an absolute origin picked for the hardware available to the author. The listing is well commented and includes a symbol table found at the end.

```
                // THIS PROGRAM, CONSISTING OF A PROM SECTION OF
                /SUBROUTINES & MAINLINE TASKS AND A RAM
                /SECTION OF BUFFERS, GENERATES AND DECODES MORSE CODE.

                /THE DECODING SECTION IS DESIGNED TO BE SELF TRACKING
                /AS TO CHANGING SPEEDS.  DUE TO THIS SELF TRACKING
                /FEATURE, A FEW ILLEGAL CODES MAY BE DECODED AS THE
                /PROGRAM ATTEMPTS TO LOCK ONTO THE INPUT CODE SPEED.

                /THIS PROGRAM IS DESIGNED TO RUN ON THE INTEL 8008-1 BASED
                /MPS STARTER SET AVAILABLE FROM THE DEC COMPONENTS GROUP.



                / WRITTEN 1-29-76 BY BRUCE FILGATE OF COMPONENTS
                /APPLICATIONS ENGINEERING AT DIGITAL EQUIPMENT
                /CORPORATION IN MARLBOROUGH MASSACHUSETTS FOR THE
                /LOGIC PRODUCTS MPS PRODUCT LINE



                /WEIGHTING IS 1 DASH TO 3 DOTS.


                /JUST A FEW WORDS ABOUT THE STACK STRUCTURE.....
                        /THE STACKS USE THE FIRST LOCATION AS A COUNT OF THE
                        /NUMBER OF OTHER USED LOCATIONS IN THE STACK.

                /THE LSB OF THE INPUT BYTE FROM I/O CHANNEL #2 IS RESERVED FOR
                        /THE SENSE LINE FOR CODE INPUT.

                /THE BYTE ON I/O CHANNEL #4 IS USED FOR CODE OUTPUT

                /KEY SENSE FOR CODE INPUT AND OUTPUT IS GROUND FOR KEY DOWN
                        /CONDITION AND LOGIC HIGH FOR THE KEY UP CONDITION.


                /*******************************************
                /
                / PROGRAM SHOULD BE STARTED AT STRT
                /
                /       PROGRAM MAY BE RESTARTED AT RESTRT
                /
                /*******************************************


                /THE TERMINAL BELL WILL BE RUNG WHENEVER A BUFFER
                        /OVERFLOW IS CAUSED BY THE USER.  THE CHAR
                        /THAT WOULD HAVE CAUSED THE OVERFLOW IS TRAPPED
                        /AND DELETED.


                /NORMALLY THE KEYBOARD DATA IS TRANSMITTED OUT TRANSLATED
                /BUT ALT MODE (ESC) KEY USED FOR SPECIAL COMMANDS.
                        /L=RELOAD THE MESSAGE BUFFER UNTIL ESC
                        /ESC=RETURN TO NORMAL MODE
                        /P=PRINT THE MESSAGE BUFFER CONTENTS
                        /S=SHIP THE MESSAGE BUFFER TRANSLATED
                        /T=TEST BY DOING S UNTIL AN ESC IS TYPED
                        /W=LOAD BAUD CONSTANT FOR WPM OUTPUT
                                /       SP      120 WPM
                                /       !       89
                                /       "       63
                                /       #       48
                                /       $       44
                                /       %       37
                                /       '       27
                                /       )       22
                                /       +       18
                                /       /       14
                                /       3       12
                                /       >       7.6
                                /       ?       7.2

                /AND THE DELETE (RUBOUT) KEY IS USED TO EDIT BUFFER AS
                /WELL AS REPRESENT THE ERROR CODE IN THE IMMEDIATE MODE.



                        /NEED A FEW MORE INSTRUCTIONS HERE...
                        OPDEF   SENSE;105;0     /READ THE SENSE LINE
                        OPDEF   READ;101;0      /SERIAL INPUT
                        OPDEF   PRINT;121;0     /SERIAL OUTPUT
                        OPDEF   STATUS;103;0    /SERIAL STATUS
                        OPDEF   OUTPUT;131;0    /ENCODED OUTPUT
                /BUFFER SIZE SET UP
                        MSGSZ=377       /MESSAGE HOLDING
                        BUFOUT=60       /CODE OUTPUT
                        BUFSKY=377      /KEYBOARD
                        BUFSPN=60       /PRINTER

                /TERMINAL DEPENDENT CONSTANTS
                        WIDTH=110       /PRINTER WIDTH IN OCTAL
                        CR=15           /CAR RET CHAR (ASCII CR=15
                        LF=12           /LINE FEED CHAR (ASCII LF=12)
                        W=127           /LOAD NEW SPEED CONSTANT (ASCII W=127)
                        ERCHAR=7        /CONSTANT FOR ERROR CHAR (ASCII BEL=7)
                        ESC=175         /ENTER COMMAND MODE (ASCII ESC=175)
                        L=114           /LOAD MESSAGE BUFFER (ASCII L=114)
                        P=120           /PRINT THE MESSAGE BUFFER (ASCII P=120)
                        S=123           /TRANSLATE&SEND MESSAGE (ASCII S=123)
                        T=124           /TEST DO S UNTIL ESC TYPED (ASCII T=124)
                        ESCSYM=44       /ECHO A $ FOR ESC
                        QUEST=77        /QUESTION MARK FOR BAD COMMAND
                        DELETE=177      /CHAR THAT REPRESENTS THE DELETE
                        DELSYM=134      /PRINTABLE CHAR FOR A DELETE
                        ETX=3           /CONTROL C EXIT TO STRT .
                        C=103           /REQUIRED TO ECHO ETX, ^C
                        UPARRO=136      /REQUIRED TO ECHO ETX, ^C
                        BLANK=40        /ASCII SPACE CONSTANT


                        *20#120
                        /START UP TIME HOUSEKEEPING
20 120  056     STRT,   LHI     MSSGBF^ /CLEAR THE MESSAGE BUFFER
        026
20 122  066             LLI     MSSGBF
        044
20 124  370             LMA
20 125  056             LHI     BAUD^   /SET OUTPUT BAUD AT ABOUT 15 WPM
        024
20 127  066             LLI     BAUD
        301
20 131  076             LMI     35
        035
20 133  106             CAL     INCLH   /LIKEWISE WITH INPUT BAUD
        046
        021
20 136  076             LMI     35
        035
20 140  250     STRT1,  XRA             /RESTART FOR ^C AND CLEAR THE A REG
20 141  056             LHI     KYFIFO^ /CLEAR THE KYBD CHAR COUNT
        024
20 143  066             LLI     KYFIFO
        305
20 145  370             LMA
20 146  056             LHI     OTFIFO^ /CLEAR THE OUTPUT BUFFER
        025
20 150  066             LLI     OTFIFO
        364
20 152  370             LMA
20 153  056             LHI     CMMND^  /INITIALIZE THE MODE BYTE
        024
20 155  066             LLI     CMMND
        275
20 157  370             LMA
20 160  106             CAL     INCLH   /INIT THE CHAR COUNT
        046
        021
20 163  370             LMA             /SET FOR CRLF INITIALIZATION
20 164  056             LHI     PNFIFO^ /INITIALIZE THE PRINTER CHAR CNT
        025
20 166  066             LLI     PNFIFO
        304
20 170  370             LMA             /ZERO PRINTER CHAR COUNT
20 171  131             OUTPUT          /CLEAR THE TONE AND OTHER BITS


                /MONITOR ENTRY AND SUPERVISOR MAIN TASK
20 172  106     RESTRT, CAL     INPEND  /TRY CODE INPUT LINE
        034
        024
20 175  106             CAL     KYBD    /TRY THE KEYBOARD TASK
        227
        022
20 200  106             CAL     PNTR    /TRY THE PRINTER TASK
        316
        022
20 203  056             LHI     CMMND^  /TEST THE MODE BYTE
        024
20 205  066             LLI     CMMND
        275
20 207  250             XRA             /ZERO THE A REG AND FLGS
20 210  207             ADM             /MODE BYTE TO A REG AND FLGS
20 211  110             JFZ     CMMNDR  /ENTER COMMAND MODE
        150
        021
20 214  106             CAL     IDLE    /NON COMMAND CHARACTER FEEDTHROUGH
        360
        023
20 217  106             CAL     OTPUT   /ANYTHING MORSE TO OUTPUT?
        230
        023
20 222  104             JMP     RESTRT  /AND LOOP (?) ALONG
        172
        020



                /ASCII TABLE OF DATA
20 225  101     ASCTAB, BLOCK   32;101;1        /A THROUGH Z
        102
        103
        104
        105
        106
        107
        110
        111
        112
        113
        114
        115
        116
        117
        120
        121
        122
        123
        124
        125
        126
        127
        130
        131
        132
20 257  061             BLOCK   11;61;1         /1 THROUGH 9
        062
        063
        064
        065
        066
        067
        070
        071
20 270  060             DATA    60      /0
20 271  055             DATA    55      /-
20 272  056             DATA    56      /.
20 273  054             DATA    54      /,
20 274  077             DATA    77      /?
20 275  057             DATA    57      /SLASH
20 276  072             DATA    72      /:
20 277  050             DATA    50      /(
20 300  051             DATA    51      /)
20 301  047             DATA    47      /'
20 302  042             DATA    42      /"
20 303  012             DATA    12      /END OF MESSAGE(CR/LF)
20 304  012             DATA    12      /END OF WORK(CR/LF)
20 305  073     ASCEND, DATA    73      /;
```

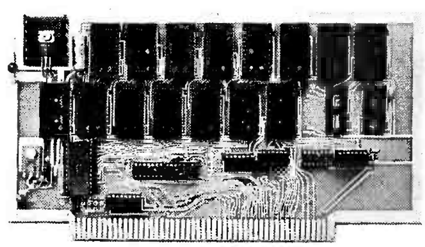
AN EDUCATIONAL EXPERIENCE...

EXPOSE YOURSELF



THE FD-8 FLOPPY DISK SYSTEM is designed for use with 8 bit microcomputer systems including the ALTAIR 8800, IMSAI, and SWTP 6800. **JUST COMPARE THESE FEATURES:**

- Full sector buffering for both read and write operations.
- Both single and double density models available.
- Up to 4 drives may be daisy-chained together.
- Complete software documentation for MSI-FDOS and disk driver routines provided for 8080 and 6800 systems.
- Resident I/O software for disk operation of assembler/editor.
- Easy interface, just two 8 bit parallel I/O ports required, or one PIA chip!
- Furnished complete with cabinet, power supply, drive, controller and cables.
- LOW COST, only $950.00 for this introductory offer.



THE MSI PROM/RAM BOARD WITH SYSTEM MONITOR SOFTWARE—for use in ALTAIR and IMSAI systems.

Free yourself from the drudgery of keying in bootstrap programs on front panel switches. Our monitor software contains all the common bootstrap loaders for assembler/editor software as well as basic. Just examine the proper starting address and press run! That's all there is to it! The monitor software takes over and all further selection of programs is done via your terminal. **The monitor contains many other time saving features which will allow you to program and debug with ease!**

- Examine and change memory contents using either octal or hex as desired.
- Dump and reload your programs to either paper tape or cassette using either hex or BNPF formats as desired.
- Execute user's program at any desired starting address.
- Trap functions display register contents at desired breakpoints inserted in user's program.
- List programs on your terminal.
- Look function scans a specified page of memory for a desired value or instruction displaying all memory locations where it is found.
- Relocator which moves a program from one area of memory to another, changing reference values thus allowing the program to run in its new location.
- Disassembler program which processes object code residing in memory displaying mnemonic with arguments, addresses, location values, and ASCII representation if applicable.
- Board accepts 15 1702A ROMs providing 3,840 bytes of Read-Only-Memory. 256 bytes of scratchpad RAM allow monitor software to be operated without the utilization of existing system memory.

PROM/RAM Board complete less PROMS.......... $ 95.00
Software listings only.............................. $ 45.00
PROM/RAM Board complete with
1702A PROMS containing monitor software........$245.00

```
                        /MORSE TABLE:1=DASH,0=DOT,WITH A 1 TO END THE CHAR
20 306   140   MORTAB,  DATA    140     /A
20 307   210            DATA    210     /B
20 310   250            DATA    250     /C
20 311   220            DATA    220     /D
20 312   100            DATA    100     /E
20 313   050            DATA    50      /F
20 314   320            DATA    320     /G
20 315   010            DATA    10      /H

20 316   040            DATA    40      /I
20 317   170            DATA    170     /J
20 320   260            DATA    260     /K
20 321   110            DATA    110     /L
20 322   340            DATA    340     /M
20 323   240            DATA    240     /N
20 324   360            DATA    360     /O
20 325   150            DATA    150     /P

20 326   330            DATA    330     /Q
20 327   120            DATA    120     /R
20 330   020            DATA    20      /S
20 331   300            DATA    300     /T
20 332   060            DATA    60      /U
20 333   030            DATA    30      /V
20 334   160            DATA    160     /W
20 335   230            DATA    230     /X

20 336   270            DATA    270     /Y
20 337   310            DATA    310     /Z
20 340   174            DATA    174     /1
20 341   074            DATA    74      /2
20 342   034            DATA    34      /3
20 343   014            DATA    14      /4
20 344   004            DATA    4       /5
20 345   204            DATA    204     /6

20 346   304            DATA    304     /7
20 347   344            DATA    344     /8
20 350   364            DATA    364     /9
20 351   374            DATA    374     /0

20 352   206            DATA    206     /-
20 353   126            DATA    126     /.
20 354   316            DATA    316     /,
20 355   062            DATA    62      /?
20 356   224            DATA    224     /(SLASH)
20 357   342            DATA    342     /:
20 360   266            DATA    266     /(
20 361   266            DATA    266     /)
20 362   172            DATA    172     /'
20 363   112            DATA    112     /"
20 364   124            DATA    124     /END OF MESSAGE(CR/LF)
20 365   026            DATA    26      /END OF WORK(CR/LF)
20 366   252   MOREND,  DATA    252     /;


               /SUBROUTINE TO PUT DATA IN A GENERAL STACK
                        /STACK POINTER IN H&L, DATA IN B, BUFFER SIZE IN C

                        /RETURNS WITH A=0 IF NO ERROR, =ERCHAR IF ERROR
20 367   307   ENTPAK,  LAM             /CHAR COUNT TO A
20 370   021            DCC             /COMPUTE CHARACTER LOCATIONS IN BUFFER
20 371   272            CPC             /DON'T OVERFLOW THE BUFFER
20 372   150            JTZ     ERROFL  /FULL...
         011
         021
20 375   004            ADI     1       /BUMP THE COUNT
         001
20 377   370            LMA             /CHAR COUNT UPDATE TO MEM
21 000   206            ADL             /LOW POINTER ADDED TO A
21 001   360            LLA             /GET L VALUE UPDATED TO L REG
21 002   100            JFC     OK      /IF A CARRY, FIX THE H
         006
         021
21 005   050            INH             /FIX H REG
21 006   371   OK,      LMB             /CHAR TO MEMORY
21 007   250            XRA             /CLEAR THE A REG
21 010   007            RET             /DONE
21 011   004   ERROFL,  ADI     1       /SET A REG NON ZERO ERR RETURN
         001
21 013   007            RET             /SYSTEM ERROR, SYSTEM MUST FIX!!!!


               /POP SUBROUTINE: ENTERED WITH POINTER IN H&L, SIZE IN B
21 014   307   POP,     LAM             /GET COUNTER TO A REG
21 015   024            SUI     1       /DECREMENT
         001
21 017   370            LMA             /RESTORE THE NEW COUNTER
21 020   104            JMP     POPY    /GET INTO POPLOP LOOP
         036
         021

                        /LOOP FOR POP
21 023   106   POPLOP,  CAL     INCLH   /POINT AT CHAR TO POP
         046
         021
21 026   327            LCM             /GET CHAR TO C REG
21 027   106            CAL     DCRLH   /POINT TO NEW LOCATION
         052
         021
21 032   372            LMC             /PUT CHAR IN MEMORY
21 033   301            LAB             /B TO A REG
21 034   024            SUI     1       /SUBTRACT 1
         001
21 036   053   POPY,    RTZ             /DONE?
21 037   310            LBA             /RESTORE B REG
21 040   106            CAL     INCLH   /RECOVER FROM THE DECREMENT POSITION
         046
         021
21 043   104            JMP     POPLOP  /NEXT PAIR POP
         023
         021


               /SUBROUTINE TO INCREMENT THE H AND L REGS
21 046   060   INCLH,   INL             /BUMP THE L
21 047   013            RFZ             /RETURN IF NO CARRY
21 050   050            INH             /BUMP THE H ON A CARRY
21 051   007            RET             /ALL DONE


               /SUBROUTINE TO DECREMENT THE H AND L REGS
21 052   306   DCRLH,   LAL             /L TO A
21 053   024            SUI     1       /DECREMENT THE A
         001
21 055   360            LLA             /RETURN IF NO BORROW
21 056   003            RFC             /NO BORROW, THEN RETURN
21 057   051            DCH             /FIX THE H AFTER A BORROW
21 060   007            RET             /DONE


               /SUBROUTINE TO WAIT A UNIT CODE TIME, DESTROYS A,B,C
21 061   056   TICK,    LHI     BAUD^   /POINT AT CONST
         024
21 063   066            LLI     BAUD
         301
21 065   026            LCI     50      /MULTIPLIER CONSTANT
         050
21 067   317   WAIT2,   LBM             /CONST TO B
21 070   011   WAIT1,   DCB             /COUNT IT DOWN DELAY
21 071   110            JFZ     WAIT1
         070
         021
21 074   021            DCC             /MULTIPLY IT
21 075   110            JFZ     WAIT2
         067
         021
21 100   106            CAL     KYBD    /OVERLAP WITH KEYBOARD INPUT
         227
         022
21 103   106            CAL     PNTR    /LIKEWISE WITH THE PRINTER
         316
         022
21 106   007            RET             /DELAY OVER!!!!


               /DELAY, USED FOR 8 SLICE DECODING OF INPUT CODE
21 107   056   TICKI,   LHI     BAUDI^  /POINT AT CONSTANT
         024
21 111   066            LLI     BAUDI
         302
21 113   026            LCI     5       /8 TIMES FASTER THAN TICK
         005
21 115   104            JMP     WAIT2   /FINISH TICKI IN TICK ROUTINE
         067
         021


               /SUBROUTINE TO GENERATE A DOT AND POST SPACE, DESTROYS A,B,C,H,L
21 120   006   DOT,     LAI     377     /SET ALL BITS IN THE A REG
         377
21 122   131            OUTPUT          /TURN ON THE KEY (DOWN)
21 123   104            JMP     FINDOT  /FINISH THE DOT IN THE DASH ROUTINE
         137
         021


               /SUBROUTINE GENERATES DASH ITS POST SPACE, DESTROYS A,B,C,H,L
21 126   006   DASH,    LAI     377     /SET ALL BITS IN THE A REG
         377
21 130   131            OUTPUT          /KEY DOWN
21 131   106            CAL     TICK    /DASH
         061
         021
21 134   106            CAL     TICK
         061
         021
21 137   106   FINDOT,  CAL     TICK    /ENTERED HERE TO FINISH A DOT
         061
         021
21 142   250            XRA             /CLEAR THE A REG
21 143   131            OUTPUT          /KEY UP
21 144   106            CAL     TICK
         061
         021
21 147   007            RET


               /MONITOR TASK SUBROUTINE FOR HANDLING COMMANDS FROM THE KEYBOARD
21 150   106   CMMNDR,  CAL     UNPAK   /GET CHAR FROM KEYBOARD
         164
         022
21 153   150            JTZ     CMMNDR  /IF NO CHAR, WAIT...
         150
         021
21 156   074            CPI     ESC     /WAS IT ANOTHER ESC?
         175
21 160   150            JTZ     CLRMD   /EXIT ON ESC
         153
         022
21 163   074            CPI     L       /WAS IT A LOAD?
         114
21 165   150            JTZ     LDNXT   /YES, GO LOAD A MESSAGE
         224
         021
21 170   074            CPI     P       /WAS IT A PRINT?
         120
21 172   150            JTZ     PRT     /YES GO PRINT THE MESSAGE BUFFER
         317
         021
21 175   074            CPI     S       /WAS IT A SEND THE BUFFER?
         123
21 177   150            JTZ     SNDNX   /YES, SHIP OUT THE MESSAGE
         060
         022
21 202   074            CPI     T       /WAS IT A TEST?
         124
21 204   150            JTZ     TEST    /YES, SEND THE BUFFER UNTIL ESC IS TYPED
         074
         022
21 207   074            CPI     W       /WAS IT A NEW WPM CONSTANT?
         127
21 211   150            JTZ     WPM     /YES, LOAD NEXT CHAR AS THE CONST
         133
         022
                        /IF HERE A BAD COMMAND
21 214   016            LBI     QUEST   /QUESTION MARK
         077
21 216   106            CAL     PPAK    /TO THE PRINTER FIFO
         304
         022
21 221   104            JMP     CMMNDR  /TRY FOR A VALID COMMAND CHAR
         150
         021

               /ROUTINE TO LOAD THE MESSAGE BUFFER
21 224   056   LDNXT,   LHI     MSSGBF^ /POINT AT THE CURRENT CHAR POINTER
         026
21 226   066            LLI     MSSGBF
         044
```

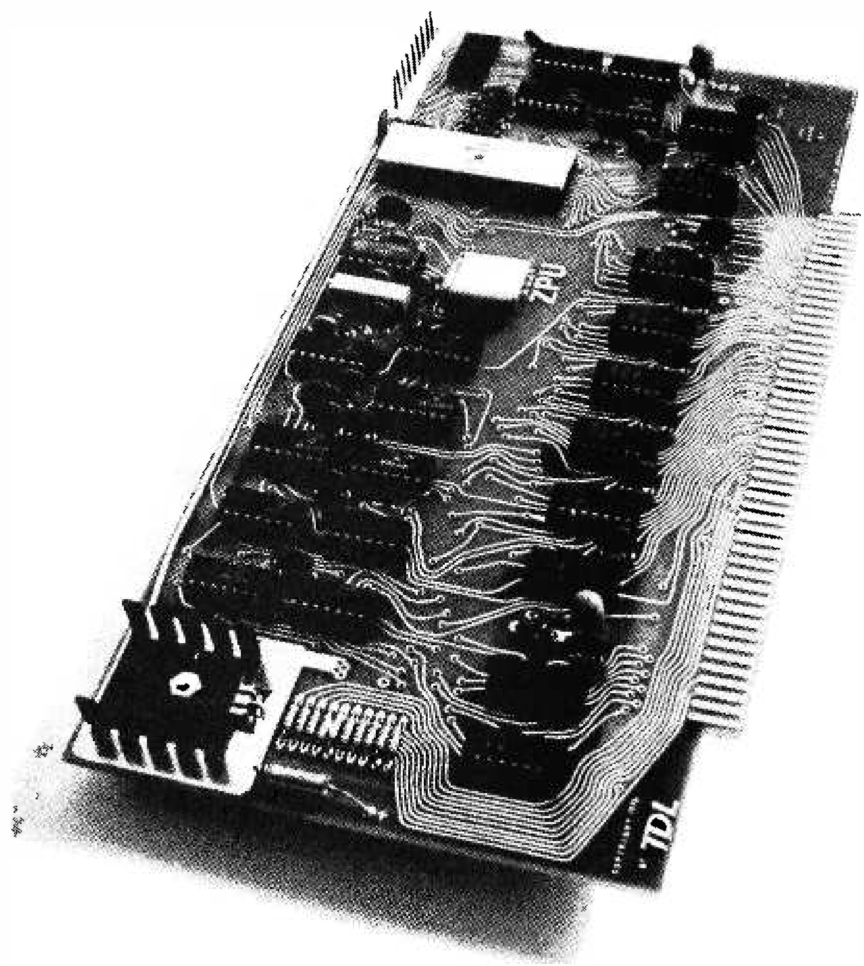
We've got it! Z-80 power for the Altair bus.

Here it is, TDL's ZPU™ the highest point of technology for an Altair/IMSAI system. Now, you can multiply your present capabilities without creating costly obsolescence. Take advantage of the wide range of existing hardware backup for your current system. The ZPU is compatible and dependable with many plus features you'll want to know about.

To further round out your system we also have available the fastest and lowest power static ram boards going (from 4 to 16K with expansion) and a system monitor board with a Z-80 monitor, powerful I/O and audio casette features.

As for software, TDL's user support program is unmatched in the micro-processor industry, currently available are the Z-80 monitor, basic, and the most sophisticated MACRO-ASSEMBLER yet developed.

Join the Z-80 revolution — it's more than just hardware!



ZPU Kit $269.00
Z16K Kit $574.00

Dealer Inquiries Invited.

HOW TO ORDER Just send check or money order, or use your BankAmericard or Mastercharge, and your orders will be shipped to you postpaid. COD orders must be accompanied by a 25% deposit. Your credit card order must include the serial # of the card, expiration date, and your order must be signed. New Jersey residents add 5% state sales tax. For more information, send for our free catalog.

TDL

(609) 392-7070

TECHNICAL DESIGN LABS, INC.
342 COLUMBUS AVENUE
TRENTON, NEW JERSEY 08629

```
21 230  076           LMI     0        /ZERO THE COUNT
        000
21 232  106  LDNXT1,  CAL     UNPAK    /GET KEYBOARD CHAR...
        164
        022
21 235  150           JTZ     LDNXT1   /WAIT FOR DATA
        232
        021
21 240  074           CPI     ESC      /END OF THE INPUT MESSAGE?
        175
21 242  150           JTZ     CLRMD    /EXIT END OF MESSAGE
        153
        022
21 245  310           LBA              /CHAR TO B REG FOR ENTPAK
21 246  056           LHI     MSSGBF^  /POINT AT THE MESSAGE BUFFER
        026
21 250  066           LLI     MSSGBF
        044
21 252  074           CPI     DELETE   /DELETE COMMAND?
        177
21 254  110           JFZ     LDNXT2   /NO
        272
        021
21 257  250           XRA              /YES, CLEAR THE A REG AND FLGS
21 260  207           ADM              /GET COUNT TO A AND FLGS
21 261  150           JTZ     LDNXT1   /BUFFER EMPTY, A NO NO
        232
        021
21 264  034           SBI     1        /DECREMENT COUNTER
        001
21 266  370           LMA              /RETURN COUNT TO MEM
21 267  104           JMP     LDNXT1   /LOOP
        232
        021
21 272  026  LDNXT2,  LCI     MSGSZ    /BUFFER SIZE SET UP
        377
21 274  106           CAL     ENTPAK   /CHAR TO BUFFER
        367
        020
                      /TEST FOR BUFFER OVERFLOW
21 277  112           CFZ     WHOOP    /TELL USER BUFFER FULL
        305
        021
21 302  104           JMP     LDNXT1   /LOOP UNTIL ESC
        232
        021


             /SUBROUTINE FOR USER ERROR IDICATION
21 305  103  WHOOP,   STATUS           /YES, GET PRINTER STATUS
21 306  044           NDI     20       /TBMT MASK
        020
21 310  150           JTZ     WHOOP    /WAIT
        305
        021
21 313  006           LAI     ERCHAR   /SET ERROR INDICATION
        007
21 315  121           PRINT
21 316  007           RET


             /ROUTINE TO PRINT THE MESSAGE BUFFER
21 317  056  PRT,     LHI     SOH^     /SET UP PRINT SUBMODE
        024
21 321  066           LLI     SOH
        300
21 323  076           LMI     0
        000
21 325  106           CAL     DMPSUB   /PRINT THE MESSAGE BUFFER
        333
        021
21 330  104           JMP     CLRMD    /EXIT TO THE SUPERVISOR
        153
        022

             /SUBROUTINE TO MOVE THE MESSAGE BUFFER CONTENTS TO LOCATION
             /DEFINED BY THE SOH LOCATION.  0=PRINTER    1=SENDER
21 333  056  DMPSUB,  LHI     MSSCNT^  /POINT AT TEMP CHAR POINTER
        024
21 335  066           LLI     MSSCNT
        277
21 337  076           LMI     0        /CLEAR THE POINTER
        000
21 341  106  PRT1,    CAL     PNTR     /TRY TO FINISH THE PRINTING
        316
        022
21 344  106           CAL     KYBD     /TRY TO FINISH THE KEYBOARD INPUT
        227
        022
21 347  106           CAL     OTPUT    /TRY TO FINISH THE TRANSMISSION
        230
        023
21 352  056           LHI     MSSGBF^  /IS THERE A MESSAGE?
        026
21 354  066           LLI     MSSGBF   /FIND CHAR CNTR AND CHECK FOR NON ZERO
        044
21 356  250           XRA              /CLEAR THE A REG
21 357  207           ADM              /ADD IN THE CHAR COUNT
21 360  053           RTZ              /NO MESSAGE, EXIT
21 361  056           LHI     MSSCNT^  /CHECK COUNT ON XFER CHARS
        024
21 363  066           LLI     MSSCNT
        277
21 365  277           CPM              /BUFFER ALL XFERED?
21 366  053           RTZ              /EVERYTHING XFERED, EXIT
21 367  317           LBM              /STILL HERE, BUMP THE CHAR COUNT
21 370  010           INB
21 371  371           LMB
                      /FETCH THE CHAR FROM THE MESSAGE BUFFER
21 372  006  PR4,     LAI     MSSGBF   /COMPUTE POINTERS
        044
21 374  207           ADM              /ADD IN THE BUFFER OFFSET
21 375  360           LLA              /SET UP THE L, H YET TO GO
21 376  056           LHI     MSSGBF^  /H SET IF NO CARRY FROM THE L
        026
22 000  100           JFC     PR2      /NO CARRY
        004
        022
22 003  050           INH              /FIX FOR THE L CARRY
22 004  317  PR2,     LBM              /CHAR TO THE B REG
22 005  250           XRA              /CLEAR THE A REG
22 006  056           LHI     SOH^     /GET THE SUBMODE
        024
22 010  066           LLI     SOH
        300
22 012  207           ADM              /SUBMODE IN THE A REG AND FLGS
22 013  110           JFZ     SOH1     /SEND SUBMODE
        024
        022
22 016  106           CAL     PPAK     /PRINT SUBMODE
        304
        022
22 021  104           JMP     PR3
        035
        022
22 024  056  SOH1,    LHI     OTFIFO^  /POINT AT OUTPUT BUFFER
        025
22 026  066           LLI     OTFIFO
        364
22 030  026           LCI     BUFOUT   /BUFFER SIZE TO C REG
        060
22 032  106           CAL     ENTPAK   /XFER TO THE BUFFER
        367
        020
22 035  150  PR3,     JTZ     PRT1     /LOOP IF NO ERROR
        341
        021
22 040  106           CAL     PNTR     /YES, OVERLAY THE I/O
        316
        022
22 043  106           CAL     KYBD
        227
        022
22 046  106           CAL     OTPUT
        230
        023
22 051  056           LHI     MSSCNT^  /RESET THE H REG FOR ERROR RECOVERY
        024
22 053  066           LLI     MSSCNT
        277
22 055  104           JMP     PR4      /TRY THE CHAR AGAIN
        372
        021


             /ROUTINE TO SHIP OUT THE BUFFER IN CODE
22 060  056  SNDNX,   LHI     SOH^     /SET UP SEND MODE FOR
        024
22 062  066           LLI     SOH      /DMPSUB ROUTINE
        300
22 064  076           LMI     1
        001
22 066  106           CAL     DMPSUB   /SEND THE MESSAGE BUFFER
        333
        021
22 071  104           JMP     CLRMD    /EXIT
        153
        022


             /ROUTINE TO TEST OUTPUT, SHIP UNTIL ESC IS TYPED
22 074  056  TEST,    LHI     SOH^     /SET UP FOR SEND MODE
        024
22 076  066           LLI     SOH
        300
22 100  076           LMI     1
        001
22 102  106           CAL     DMPSUB   /SEND THE BUFFER
        333
        021
22 105  106           CAL     UNPAK    /CHAR FROM KEYBOARD
        164
        022
22 110  150           JTZ     TEST     /NOTHING YET, DO IT AGAIN
        074
        022
22 113  074           CPI     ESC      /ESC?
        175
22 115  150           JTZ     CLRMD    /YES, EXIT
        153
        022
22 120  016  TEST1,   LBI     QUEST    /ILLEGAL CHAR FOR THIS
        077
22 122  106           CAL     PPAK     /NOTIFY THE USER
        304
        022
22 125  110           JFZ     TEST1    /IF OVERFLOW, TRY AGAIN
        120
        022
22 130  104           JMP     TEST     /AND LOOP....
        074
        022


             /CODE TO LOAD A NEW WPM CONSTANT INTO BAUD
22 133  106  WPM,     CAL     UNPAK    /GET CHAR FROM KEYBOARD
        164
        022
22 136  150           JTZ     WPM      /WAIT FOR CHAR
        133
        022
22 141  044           NDI     37       /MASK FOR 5 VALID BITS
        037
22 143  002           RLC              /MULTIPLY BY 2
22 144  064           ORI     1        /SET THE LSB
        001
22 146  056           LHI     BAUD^    /POINT AT BAUD LOCATION
        024
22 150  066           LLI     BAUD
        301
22 152  370           LMA              /CONSTANT TO BAUD LOCATION
                      /FALL THROUGH TO CLRMD
             /CLEAR THE FLAGS AND EXIT TO SUPERVISOR
22 153  056  CLRMD,   LHI     CMMND^   /ZERO THE MODE BYTE
        024
22 155  066           LLI     CMMND
        275
22 157  076           LMI     0
        000
22 161  104           JMP     RESTRT   /TO SUPERVISOR
        172
        020


             /ROUTINE TO GET CHAR FROM KEYBOARD FIFO
                      /RETURNS WITH CHAR IN A REG, A REG=0 IF NO CHAR
22 164  106  UNPAK,   CAL     PNTR     /TRY TO FINISH PENDING PRINTING
        316
        022
```

You'll Want to Nybble at these Byte Books

Where does the editor of a computer magazine turn when he must verify some author's hardware design? Information on a 75450 interface gate, or a 74147 priority encoder circuit does not spring forth by magic. Checking the information supplied by authors is part of BYTE's quality control program.

When you build a project, you need this same sort of information. All you find in the advertisements for parts are mysterious numbers identifying the little beasties . . . hardly the sort of information which can be used to design a custom logic circuit. You can find out about many of the numbers by using the information found in these books. No laboratory bench is complete without an accompanying library shelf filled with references – and this set of Texas Instruments engineering manuals plus Don Lancaster's *TTL Cookbook* will provide an excellent starting point or addition to your personal library.

- The **TTL Cookbook** by Don Lancaster, published by Howard W Sams, Indianapolis, Indiana. Start your quest for data here with Don's tutorial explanations of what makes a TTL logic design tick. 335 pages, $8.95.

- **The TTL Data Book for Design Engineers,** by Texas Instruments Incorporated. How does an engineer find out about the TTL circuits? He reads the manufacturer's literature. This 640 page beauty covers the detailed specs of most of the 7400 series TTL logic devices. No experimenter working with TTL has a complete library without **The TTL Data Book for Design Engineers.** Order yours today, only $3.95.

- The **Supplement to The TTL Data Book for Design Engineers,** by Texas Instruments Incorporated. What happens when you can't find a 7400 series device listed in **The Data Book for Design Engineers?** Before you start screaming and tearing your hair out in frustration, turn to the Supplement. The Supplement has 400 pages of additional information including a comprehensive index to both TTL Data Book volumes. To be complete (and keep your hair in place and vocal cords intact) you'd best order the supplement at $1.95 to accompany the main volume.

- The **Linear and Interface Circuits Data Book for Design Engineers,** by Texas Instruments Incorporated. When you run across one of those weird numbers like 75365 the immediate frustration problem occurs again. What kind of gate could that be? We won't tell in this ad, but you can find out by reading the specifications in **The Linear and Interface Circuits Data Book for Design Engineers.** You can interface your brain to the 72xxx (linear) and 75xxx (interface) series of functions by ordering your copy of this 688 page manual at only $3.95.

- The **Semiconductor Memory Data Book for Design Engineers,** by Texas Instruments Incorporated. Don't forget the importance of memories to your systems. Refer to this 272 page manual to find out about the TI versions of many of the popular random access memories and read only memories. Order your personal copy today, only $2.95.

- The **Transistor and Diode Data Book for Design Engineers,** by Texas Instruments Incorporated. You'd expect a big fat data book and a wide line of diodes and transistors from a company which has been around from the start of semiconductors. Well, it's available in the form of this 1248 page manual from TI which describes the characteristics of over 800 types of transistors and over 500 types of silicon diodes. This book covers the TI line of low power semiconductors (1 Watt or less). You won't find every type of transistor or diode in existence here, but you'll find most of the numbers used in switching and amplifying circuits. Order your copy today, only $4.95.

- The **Power Semiconductor Handbook for Design Engineers** by Texas Instruments Incorporated. To complement the low power transistor handbook, TI supplies this 800 page tome on high power transistors and related switching devices. Here is where you find data on the brute force monsters which are used to control many Watts electronically. Fill out your library with this book, available for only $3.95.

- **Understanding Solid State Electronics** by Texas Intruments Incorporated. This is an excellent tutorial introduction to the subject of transistor and diode circuitry. The book was created for the reader who wants or needs to understand electronics, but can't devote years to the study. This 242 page softbound book is a must addition to the beginner's library at only $2.95.

- The **Optoelectronics Data Book for Design Engineers** by Texas Instruments Incorporated. This 366 page book is a compendium of information on TI phototransistors, LEDs and related devices. Order yours at $2.95.

_____TTL Cookbook @ $8.95
_____TTL Data Book @ $3.95
_____Supplement to TTL Data Book @ $1.95
_____Linear and Interface Circuits @ $3.95
_____Semiconductor Memory Data @ $2.95
_____Transistor and Diode Data Book @ $4.95
_____Power Semiconductor Handbook @ $3.95
_____Understanding Solid State Electronics @ $2.95
_____Optoelectronics Data Book @ $2.95

Please add 75 cents for postage and handling. Please allow six weeks for delivery.

Send to: Name ______________________

Address ______________________

City ______________ State ________ Zip ________

☐ Check enclosed

☐ Bill MC #________________ Exp. Date ________

☐ Bill BA #________________ Exp. Date ________

Signature ______________________

BYTE PETERBOROUGH, NH 03458

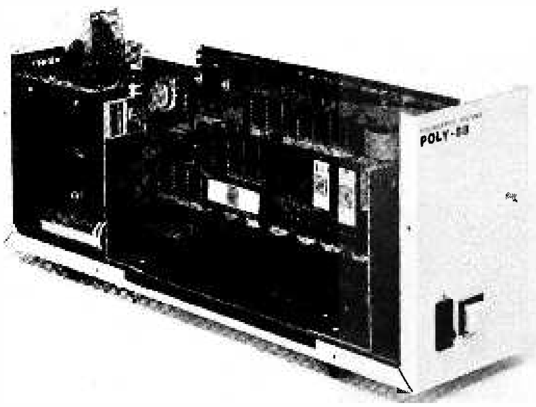
```
22 167  106          CAL     KYBD     /KEYBOARD HAPPY?
        227
        022
22 172  106          CAL     OTPUT    /OVERLAY THE CODE OUTPUT
        230
        023
22 175  056          LHI     KYFIFO^  /POINT AT KEYBOARD STACK
        024
22 177  066          LLI     KYFIFO
        305
22 201  250          XRA              /CLEAR THE A REG
22 202  207          ADM              /CHAR COUNT TO A REG
22 203  053          RTZ              /IF EMPTY, RETURN A REG=0
22 204  106          CAL     INCLH    /POINT AT CHAR
        046
        021
22 207  347          LEM              /GET CHAR TO E REG TEMP
22 210  106          CAL     DCRLH    /POINT AT KEYBOARD FIFO
        052
        021
22 213  106          CAL     POP      /OUT OF FIFO, RETURNS A=0
        014
        021
22 216  204          ADE              /CHAR TO A REG AND FLGS
22 217  007          RET              /RETURN WITH A REG=CHAR

              /SUBROUTINE TO WAIT FOR THE TBMT FLAG
22 220  103   WAITMT, STATUS          /GET TBMT FLAG
22 221  044          NDI     20
        020
22 223  150          JTZ     WAITMT   /WAIT....
        220
        022
22 226  007          RET              /OK, HAVE TBMT


                     /KEYBOARD HANDLER SUBROUTINE
22 227  103   KYBD,  STATUS           /GET THE SERIAL LINE STATUS
22 230  044          NDI     40       /MASK
        040
22 232  053          RTZ              /NEXT TASK
                     /PUT KEYBOARD CHARACTER IN KYBD FIFO
22 233  101          READ             /GET CHAR FORM KEYBOARD TO A REG
22 234  044          NDI     177      /GET RID OF ASCII PARITY BIT
        177
22 236  074          CPI     ETX      /CONTROL C
        003
22 240  110          JFZ     NETX     /NO
        267
        022
22 243  250          XRA              /YES, CLEAR OUTPUT
22 244  131          OUTPUT
22 245  106          CAL     WAITMT   /SEND OUT ^C
        220
        022
22 250  006          LAI     UPARRO
        136
22 252  121          PRINT
22 253  106          CAL     WAITMT   /WAIT FOR TBMT
        220
        022
22 256  006          LAI     C
        103
22 260  121          PRINT
22 261  106          CAL     WAITMT   /WAIT FOR TBMT
        220
        022
22 264  104          JMP     STRT1    /GO RESTART FROM ALMOST ZERO
        140
        020
22 267  056   NETX,  LHI     KYFIFO^  /POINT AT KYFIFO
        024
22 271  066          LLI     KYFIFO
        305
22 273  310          LBA              /CHAR TO B REG
22 274  026          LCI     BUFSKY   /BUFFER SIZE TO C REG
        377
22 276  106          CAL     ENTPAK   /PUT CHAR IN BUFFER
        367
        020
22 301  112          CFZ     WHOOP    /IF OVERFLOW, TELL THE USER
        305
        021
                     /FALL THROUGH AND RETURN IN NEXT ROUTINE


                     /PUT CHAR IN PRINTER FIFO FROM B REG
                     /RETURNS WITH A REG =0 IF NO ERROR, =ERCHAR IF ERROR
22 304  056   PPAK,  LHI     PNFIFO^  /POINT AT PNFIFO
        025
22 306  066          LLI     PNFIFO
        304
22 310  026          LCI     BUFSPN   /BUFFER SIZE TO C REG
        060
22 312  106          CAL     ENTPAK   /PUT CHAR IN FIFO
        367
        020
22 315  007          RET
                     /END OF THE KEYBOARD HANDLER TASK


              /PRINTER HANDLER SUBROUTINE TASK
22 316  103   PNTR,  STATUS           /GET THE PRINTER STATUS
22 317  044          NDI     20       /MASK FOR TBMT
        020
22 321  053          RTZ              /IF BUSY, TRY SOMETHING ELSE
22 322  056          LHI     TWIDTH^  /FIND PRINT POSITION
        024
22 324  066          LLI     TWIDTH
        276
22 326  250          XRA              /CLEAR THE A REG
22 327  207          ADM              /COUNT TO FLGS AND A REG
22 330  110          JFZ     PRT2     /NO LINE OVERFLOW
        347
        022
22 333  006          LAI     CR       /LINE OVERFLOW, FIX IT
        015
22 335  121          PRINT
22 336  106          CAL     WAITMT   /WAIT FOR TBMT
```

```
        220
        022
22 341  006          LAI     LF
        012
22 343  121          PRINT
22 344  076          LMI     WIDTH    /RESET PRINT POSITION COUNT
        110
22 346  007          RET
22 347  056   PRT2,  LHI     PNFIFO^  /POINT AT CHAR COUNT
        025
22 351  066          LLI     PNFIFO
        304
22 353  250          XRA              /CLEAR THE A REG
22 354  207          ADM              /CHAR COUNT TO A REG AND FLGS
22 355  053          RTZ              /NOTHING TO PRINT, NEXT TASK
              /IF HERE THERE IS PRINTING TO BE DONE!!!!!!!
                     /POINT AT CHAR TO PRINT AND PRINT IT
22 356  106          CAL     INCLH
        046
        021
22 361  347   NXTPNT, LEM             /CHAR TO E REG TEMP
                     /NOW UPDATE THE CHAR COUNT
22 362  066          LLI     PNFIFO   /POINT AT CHAR COUNT
        304
22 364  056          LHI     PNFIFO^
        025
22 366  106          CAL     POP      /RIPPLE THE FIFO
        014
        021
22 371  056          LHI     TWIDTH^  /UPDATE PRINT POSITION
        024
22 373  066          LLI     TWIDTH
        276
22 375  317          LBM              /COUNT TO B
22 376  011          DCB              /-1
22 377  371          LMB              /BACK TO MEMORY
23 000  304          LAE              /CHAR BACK TO A REG
23 001  074          CPI     LF       /IS IT A LF?
        012
23 003  150          JTZ     INCRLF   /YES, INSERT A CRLF
        031
        023
23 006  074          CPI     CR       /IS IT A CR?
        015
23 010  150          JTZ     INCRLF   /YES, INSERT A CRLF
        031
        023
23 013  074          CPI     DELETE   /IS IT A DELETE?
        177
23 015  150          JTZ     DEL      /YES, INSERT A BACKSLASH
        034
        023
23 020  074          CPI     ESC      /IS IT AN ESC?
        175
23 022  110          JFZ     PNT1     /NO
        027
        023
23 025  006          LAI     ESCSYM   /YES, SUBSTITUTE A PRINTABLE CHAR
        044
23 027  121   PNT1,  PRINT            /PRINT THE CHAR
23 030  007          RET              /DONE, PRINTED A CHAR
23 031  076   INCRLF, LMI    0        /SET FOR CRLF NEXT
        000
23 033  007          RET
23 034  006   DEL,   LAI     DELSYM   /SUBSTITUTE A PRINTABLE CHARACTER
        134
23 036  121          PRINT
23 037  007          RET              /END TO THE PRINTER TASK


              /THIS SUBROUTINE TRANSLATES A REG TO OUTPUT MODE
                     /COMPUTE THE DISPLACEMENT IN ASCTAB
23 040  056   XLATER, LHI    ASCTAB^  /POINT AT ASCTAB
        020
23 042  066          LLI     ASCTAB
        225
23 044  277   THISIT, CPM             /IS THIS THE CHAR?
23 045  150          JTZ     CONVT    /GO CONVERT THE CHAR
        074
        023
23 050  106          CAL     INCLH    /TRY NEXT CHAR
        046
        021
23 053  310          LBA              /SAVE THE CHAR IN B REG TEMPORARILY
23 054  006          LAI     ASCEND^  /GET HIGH LIMIT TO A FOR COMPARE
        020
23 056  275          CPH              /PAST END OF TABLE?
23 057  140          JTC     NTFUND   /PAST END OF TABLE AND NO MATCH
        122
        023
23 062  006          LAI     ASCEND   /GET LOW LIMIT TO A FOR COMPARE
        305
23 064  276          CPL              /PAST END OF TABLE?
23 065  140          JTC     NTFUND   /PAST END OF TABLE AND NO MATCH
        122
        023
                     /IF HERE, STILL IN TABLE.  TRY CONTENTS AGAIN.
23 070  301          LAB              /RETURN CHAR TO A REG
23 071  104          JMP     THISIT   /LOOP FOR NEXT TABLE ENTRY CHECK
        044
        023
23 074  006   CONVT, LAI     MORTAB   /COMPUTE REL DISPLACEMENT LOW
        306
23 076  024          SUI     ASCTAB
        225
23 100  100          JFC     OK1
        104
        023
23 103  051          DCH              /HANDLE THE BORROW
23 104  206   OK1,   ADL              /ADD IN THE LOW POINTER
23 105  100          JFC     OK2
        111
        023
23 110  050          INH              /HANDLE THE CARRY
23 111  360   OK2,   LLA              /L IS NOW POINTING IN THE OUTPUT TABLE
23 112  006          LAI     MORTAB^  /COMPUTE RELATIVE DISPLACEMENT HIGH
        020
23 114  034          SBI     ASCTAB^
        020
23 116  205          ADH              /ADD IN THE HIGH POINTER
23 117  350          LHA              /H NOW POINTS IN THE OUTPUT TABLE
23 120  307          LAM              /REPLACEMENT CHAR TO A REG
23 121  007          RET              /CODE IN A REG RETURN
23 122  006   NTFUND, LAI    200      /CHAR NOT FOUND LOAD OUT A 200
        200
23 124  007          RET              /ERROR RETURN
```

THE POLY 88 MICROCOMPUTER

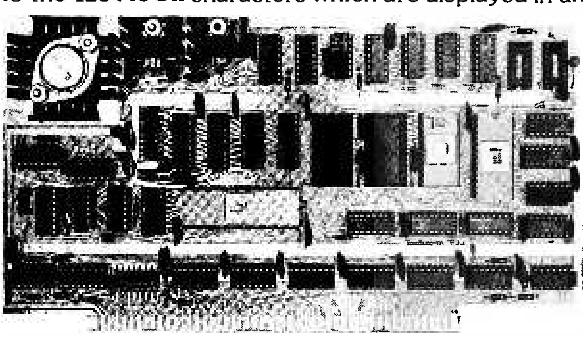
A Complete Microcomputer System with Keyboard Input and Video Output

The Hardware: The heart of the POLY 88 microcomputer, the CPU circuit card, features an 8080A central processor, 512-byte RAM, space for 3K of PROM, vectored interrupt and real time clock, a dual serial port with software-selectable baud rate,



and single-step logic that allows the processor to execute one instruction at a time.

The POLY 88 also includes our video terminal interface circuit card, which is both a video display device and an input port for a keyboard. Sixteen lines of up to 64 characters may be displayed on a standard TV monitor or modified receiver. In addition to the 128 ASCII characters which are displayed in an

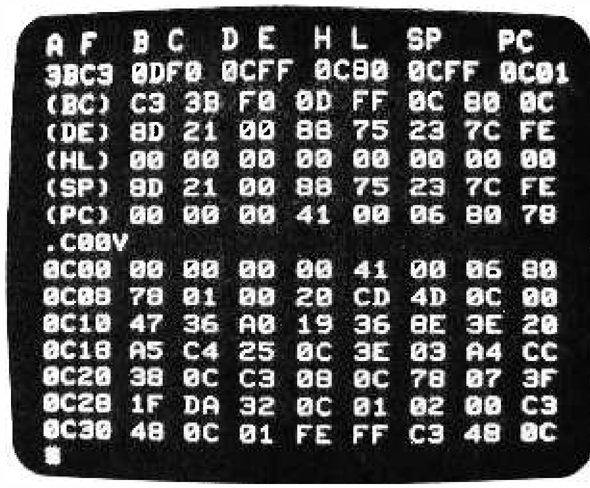


easy-to-read 7x9 font, there are 64 graphic characters available for plotting on a 48 x 128 contiguous grid.

These two circuit cards fit into a compact "Altair-compatible" chassis with room for up to five cards. A 6 amp power supply is built right on the mother board. The front panel consists of just two switches, on/off and reset; the monitor software with video screen and keyboard eliminates the need for a hardware front panel. As your system grows, you can plug chassis together for easy expansion. (The expansion connector also eliminates the need for an extender card.) Cassette, RS-232, and current loop interfaces are available which connect via ribbon cable to the CPU board and mount on the backpanel.

The Software: Supplied with the POLY 88 microcomputer is a 1024-byte monitor on ROM. The program is there when the power is turned on, so you don't need to key in a loader. The monitor is designed to use a keyboard entry and TV display, with a casstte or paper tape as the storage medium.

With the monitor you can: load data into memory in hex, display memory in hex, dump or read data from a storage device, and execute program one step at a time, displaying the contents of each of the 8080 registers as well as the values in memory at the address of each of the registers. All of these functions



are available the instant you turn on the power.

Future software available with this system will include BASIC, an assembler, and games.

Prices: Basic kit including chassis, CPU and video cards — $595, $795 assembled. Cassette option — $90 kit and $125 assembled. 8K of RAM — $300 in kit form or $375 assembled. We also sell the video and other "Altair-compatible" circuit cards separately.

Dealers: This system sells itself.

All prices and specifications subject to change without notice. Prices are USA only. California residents add 6% sales tax. Prepaid orders shipped postpaid. BankAmericard and MasterCharge accepted.

737 S. Kellogg, Goleta, CA 93017 (805) 967-2351

☐ Please send more information
☐ Order and check enclosed

Name ______________________

Address ______________________

____

BankAmericard ______________________

Master Charge ______________________

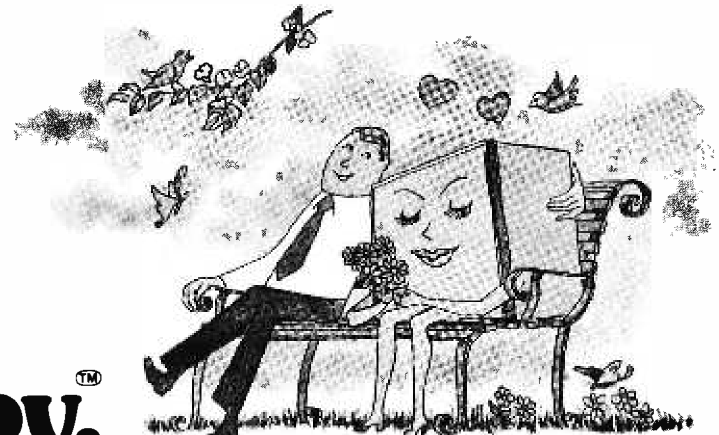
PolyMorphic Systems

```
                  /THIS SUBROUTINE TRANSLATES A REG TO PRINT MODE
23 125   074      XLAT,   CPI     377      /SET C FLG (GUARD BIT)
         377
23 127   022              RAL              /ROTATE
23 130   100              JFC     .-1      /IF NO LEFT GUARD, LOOP
         127
         023
                          /IF A REG CONTAINS A 000, ERROR CHAR WAS SEEN...
23 133   074              CPI     0        /SET FLGS
         000
23 135   110              JFZ     XLAT1    /NOT ERROR, TRANSLATE
         143
         023
23 140   006              LAI     DELSYM   /SET UP FOR DELETE SYMBOL
         134
23 142   007              RET              /EXIT
23 143   056      XLAT1,  LHI     MORTAB^  /POINT AT MORTAB
         020
23 145   066              LLI     MORTAB
         306
23 147   277      THIS,   CPM              /IS THIS THE CHAR?
23 150   150              JTZ     MCONVT   /YES, GO CONVERT THE CHAR
         177
         023
23 153   106              CAL     INCLH    /TRY NEXT CHAR
         046
         021
23 156   310              LBA              /SAVE CHAR IN B REG TEMP
23 157   006              LAI     MOREND^  /GET HIGH LIMIT TO A FOR COMPARE
         020
23 161   275              CPH              /PAST END OF TABLE?
23 162   140              JTC     NTFND    /YES, WITH NO MATCH
         225
         023
23 165   006              LAI     MOREND   /GET LOW LIMIT TO A FOR COMPARE
         366
23 167   276              CPL              /PAST END OF TABLE?
23 170   140              JTC     NTFND    /YES, NO MATCH
         225
         023
                          /IF HERE, STILL IN TABLE. TRY CONTENTS AGAIN
23 173   301              LAB              /RETURN CHAR TO A REG
23 174   104              JMP     THIS     /LOOP FOR NEXT TABLE ENTRY CHECK
         147
         023
23 177   006      MCONVT, LAI     ASCTAB   /COMPUTE REL DISPLACEMENT LOW
         225
23 201   024              SUI     MORTAB
         306
23 203   140              JTC     MOK1
         207
         023
23 206   051              DCH              /HANDLE THE BORROW
23 207   206      MOK1,   ADL              /ADD IN THE LOW POINTER
23 210   100              JFC     MOK2
         214
         023
23 213   050              INH              /HANDLE THE CARRY
23 214   360      MOK2,   LLA              /L IS NOW POINTING IN OUTPUT TABLE
23 215   006              LAI     ASCTAB^  /COMPUTE RELATIVE DISPLACEMENT HIGH
         020
23 217   034              SBI     MORTAB^
         020
23 221   205              ADH              /ADD IN THE HIGH POINTER
23 222   350              LHA              /H NOW POINTS IN THE OUTPUT TABLE
23 223   307              LAM              /REPLACEMENT CHAR TO A REG
23 224   007              RET              /ASCII CODE IN A REG RETURN
23 225   006      NTFND,  LAI     BLANK    /CHAR NOT FOUND, LOAD OUT A SPACE
         040
23 227   007              RET              /ERROR RETURN



                          /SUBROUTINE TASK TO OUTPUT CODE
23 230   056      OTPUT,  LHI     OTFIFO^  /POINT AT STACK
         025
23 232   066              LLI     OTFIFO
         364
23 234   250              XRA              /CLEAR THE A REG
23 235   207              ADM              /CHAR COUNT TO A AND FLGS
23 236   053              RTZ              /NEXT TASK, IF NOTHING TO DO
23 237   106              CAL     INCLH    /POINT AT THE DATA
         046
         021
23 242   307              LAM              /CHARACTER TO A REG
23 243   074              CPI     DELETE   /ERROR CHARACTER?
         177
23 245   110              JFZ     OTPUT1   /NO, GO TRANSLATE
         264
         023
23 250   036              LDI     7        /YES, DO 8 DOTS
         007
23 252   106      OTERR,  CAL     DOT      /1 DOT
         120
         021
23 255   031              DCD              /-1 THE COUNT
23 256   110              JFZ     OTERR    /NOT DONE....DO IT AGAIN
         252
         023
23 261   104              JMP     OUTEND   /DONE, POP AND EXIT
         342
         023
23 264   106      OTPUT1, CAL     XLATER   /TRANSLATE
         040
         023
23 267   330              LDA              /SAVE THE CHAR IN D REG
23 270   074              CPI     200      /IS IT A BAD CHAR (OR A SPACE)?
         200
23 272   110              JFZ     GOODCH   /CHAR OK, SO DO IT UP RIGHT
         311
         023
                          /IF HERE GENERATE A CHARACTER SPACE
23 275   036              LDI     6        /SET UP FOR 7 UNITS DELAY(UNITS-1)
         006
23 277   106      SPACE,  CAL     TICK     /WAIT ONE UNIT
         061
         021
23 302   031              DCD              /DECREMENT THE UNIT COUNTER
23 303   110              JFZ     SPACE    /LOOP UNTIL DONE
         277
         023
23 306   104              JMP     OUTEND
         342
         023
23 311   074      GOODCH, CPI     200      /IF A=200 THEN DONE
         200
23 313   150              JTZ     OUTEND   /DONE!
         342
         023
23 316   100              JFC     DSH      /DOT OR DASH?
         327
         023
23 321   106              CAL     DOT      /YES A DOT, SO KEY OUT A DOT
         120
         021
23 324   104              JMP     OTLOOP   /NEXT SYMBOL
         332
         023
23 327   106      DSH,    CAL     DASH     /MUST BE A DASH, SO KEY A DASH
         126
         021
23 332   303      OTLOOP, LAD              /GET THE CHAR BACK TO THE A REG
23 333   044              NDI     177      /THROW OUT THE USED BIT
         177
23 335   002              RLC              /ROTATE IN AN UNUSED BIT
23 336   330              LDA              /SAVE THE NEW IMAGE IN D
23 337   104              JMP     GOODCH   /LOOP FOR OTHER SYMBOLS
         311
         023
23 342   106      OUTEND, CAL     TICK     /INTER LETTER SPACE
         061
         021
23 345   106              CAL     TICK
         061
         021
                          /SET UP TO POP THE STACK
23 350   066              LLI     OTFIFO   /POINT AT STACK
         364
23 352   056              LHI     OTFIFO^
         025
23 354   106              CAL     POP      /POP THE STACK
         014
         021
23 357   007              RET              /NEXT TASK
                          /END OF THE OUTPUT ENCODED DRIVE TASK SUBROUTINE



                  /KEYBOARD DECODER FOR NON COMMAND MODE
23 360   056      IDLE,   LHI     KYFIFO^  /POINT AT CHAR COUNT
         024
23 362   066              LLI     KYFIFO
         305
23 364   250              XRA              /CLEAR THE A REG
23 365   207              ADM              /CHAR COUNT TO A REG
23 366   053              RTZ              /BUFFER EMPTY, TRY SOMETHING ELSE
23 367   106              CAL     INCLH    /POINT AT CHAR
         046
         021
23 372   347              LEM              /SAVE CHAR IN E REG TEMP
23 373   106              CAL     DCRLH    /POINT AT START OF BUFFER
         052
         021
23 376   106              CAL     POP      /POP THE CHAR OFF THE BUFFER
         014
         021
24 001   304              LAE              /CHAR TO A REG
24 002   074              CPI     ESC      /IS IT AN ESC?
         175
24 004   150              JTZ     IDLE1    /YES
         025
         024
24 007   056              LHI     OTFIFO^  /OUTPUT IN CODE
         025
24 011   066              LLI     OTFIFO   /SET UP FOR ENTPAK
         364
24 013   310              LBA              /DATA IN B REG
24 014   026              LCI     BUFOUT   /SIZE IN C REG
         060
24 016   106              CAL     ENTPAK
         367
         020
24 021   112              CFZ     WHOOP    /BUFFER FULL, TELL USER
         305
         021
24 024   007              RET              /DONE
24 025   056      IDLE1,  LHI     CMMND^   /SET FOR COMMAND MODE
         024
24 027   066              LLI     CMMND
         275
24 031   076              LMI     1        /MODE=1
         001
24 033   007              RET



                  /SUBROUTINE TO SERVICE MORSE CODE INPUT
24 034   105      INPEND, SENSE            /GET CODE INPUT LINE
24 035   044              NDI     1        /WE USE THE LSB
         001
24 037   013              RFZ              /NOTHING PENDING, EXIT
24 040   056              LHI     INCHAR^  /POINT AT HOLDING REG
         024
24 042   066              LLI     INCHAR
         304
24 044   076              LMI     1        /SET UP TO SHIFT IN MORSE
         001
24 046   056      INTIME, LHI     TIMER^   /POINT AT TIMER REG
         024
24 050   066              LLI     TIMER
         303
24 052   076              LMI     0        /INITIALIZE FOR TIME=0
         000
24 054   106      INSENS, CAL     TICKI    /WAIT FOR PART OF A BAUD (1/8)
         107
         021
24 057   056              LHI     TIMER^   /UPDATE TIMER
         024
24 061   066              LLI     TIMER
         303
24 063   317              LBM
24 064   010              INB              /+1
24 065   371              LMB              /TIMER+1
24 066   105              SENSE            /KEY DOWN?
24 067   044              NDI     1
         001
24 071   150              JTZ     INSENS   /WAIT FOR KEY UP
         054
         024
                          /IF HERE, KEY IS NOW UP
24 074   046              LEI     0        /SET E=0 FOR DOT, FIX LATER IF DASH
         000
```

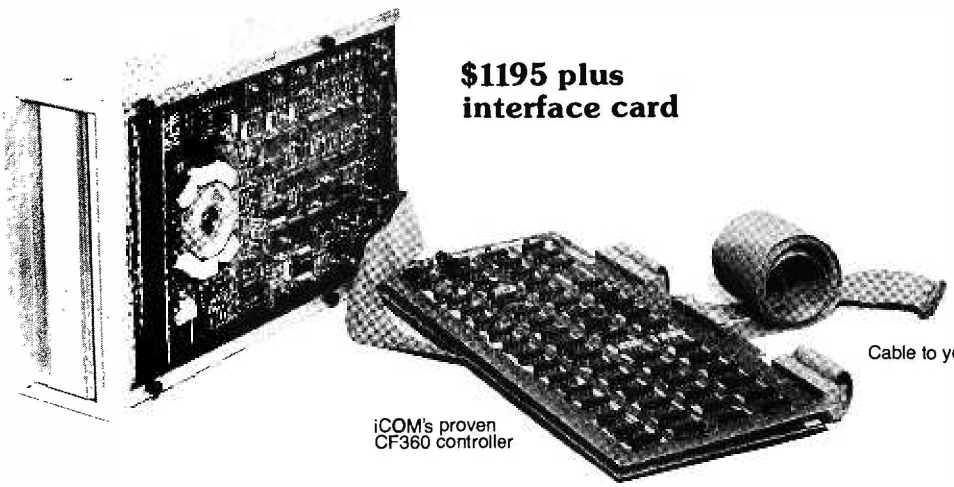
```
24 076   301              LAB              /TIMER TO A REG (B*# OF BAUD)
24 077   074              CPI     20
         020
24 101   100              JFC     INDASH   /IF DASH, SERVICE DASH
         131
         024
                          /SEE IF CLOCK MUCH TOO SLOW FOR DOT
24 104   074              CPI     6        /SHOULD BE A 10 IDEAL
         006
24 106   100              JFC     INPOK    /CLOCK IS GOOD ENOUGH
         156
         024
24 111   056              LHI     BAUDI^   /NOT GOOD ENOUGH, FIX WPM CONSTANT
         024
24 113   066              LLI     BAUDI
         302
24 115   307              LAM              /WPM TO A
24 116   074              CPI     2        /WPM TOO LOW TO TRACK?
         002
24 120   140              JTC     INPOK    /DON'T TRY TO FIX, ALREADY TOO FAST.
         156
         024
24 123   034              SBI     1        /-1
         001
24 125   370              LMA              /BAUDI-1
24 126   104              JMP     INPOK
         156
         024


                          /IF HERE, SYMBOL IS DASH, UPDATE BAUDI FOR TRACKING
24 131   056      INDASH, LHI     BAUDI^   /POINT AT INPUT WPM
         024
24 133   066              LLI     BAUDI
         302
24 135   301              LAB              /GET TIMER TO A REG AGAIN
24 136   074              CPI     34       /CLOCK TOO FAST?
         034
24 140   140              JTC     OKDASH   /NO.
         154
         024
24 143   307              LAM              /YES
24 144   074              CPI     376      /TIMER REALLY TOO TOO SLOW?
         376
24 146   100              JFC     OKDASH   /YES, BAIL OUT!
         154
         024
24 151   004              ADI     1        /+1
         001
24 153   370              LMA              /BAUDI+1
24 154   046      OKDASH, LEI     1        /SET E=1 FOR DASH
         001
24 156   056      INPOK,  LHI     INCHAR^  /POINT AT CHAR HOLDING REG
         024
24 160   066              LLI     INCHAR
         304
24 162   307              LAM              /GET PARTIAL CHAR TO A REG
24 163   022              RAL              /SHIFT UP ONE BIT
24 164   044              NDI     376      /JUNK THE OLD CARRY BIT
         376
24 166   264              ORE              /BRING IN NEW SYMBOL FROM E REG
24 167   370              LMA              /NEW PARTIAL CHAR TO INCHAR

                          /TIME THE INTERSPACE TO FIND WHAT TYPE IT IS.
24 170   056              LHI     TIMER^   /RESET THE TIMER
         024
24 172   066              LLI     TIMER
         303
24 174   076              LMI     0        /TIMER RESET
         000
24 176   106      UPTIME, CAL     TICKI    /DELAY 1/8 OF A BAUD TIME
         107
         021
24 201   105              SENSE            /GET THE KEY STATUS
24 202   044              NDI     1
         001
24 204   150              JTZ     INTIME   /KEY DOWN, GET NEXT SYMBOL
         046
         024
24 207   056              LHI     TIMER^   /UPDATE THE TIME
         024
24 211   066              LLI     TIMER
         303
24 213   317              LBM
24 214   010              INB              /+1
24 215   371              LMB              /TIMER+1
24 216   301              LAB              /GET TIMER TO A FOR COMPARE
24 217   074              CPI     377      /END OF MESSAGE?
         377
24 221   110              JFZ     NOTEOM   /KEEP LOOPING,
         235
         024
24 224   016              LBI     LF       /END OF MESSAGE, CR-LF-CR-LF
         012
24 226   106              CAL     PPAK
         304
         022
24 231   106              CAL     PPAK
         304
         022
24 234   007              RET              /EXIT TO MAINLINE
24 235   074      NOTEOM, CPI     60       /END OF WORD?
         060
24 237   110              JFZ     ENDLET   /NO.
         247
         024
24 242   006              LAI     1        /YES, OUTPUT A SPACE
         001
24 244   104              JMP     MPAK     /DO IT OUT RIGHT.
         263
         024
24 247   074      ENDLET, CPI     24       /END OF LETTER?
         024
24 251   110              JFZ     UPTIME   /NO, KEEP TIMING THE UP TIME
         176
         024
24 254   056              LHI     INCHAR^  /POINT AT HOLDING REG
         024
24 256   066              LLI     INCHAR
         304
24 260   307              LAM              /MORSE FROM HOLDING REG TO A REG
24 261   076              LMI     1        /RESET THE HOLDING REG FOR NEXT CHAR
         001
24 263   106      MPAK,   CAL     XLAT     /ASCII TO A EQUIV OF MORSE
         125
         023
24 266   310              LBA              /SET UP FOR PPAK
24 267   106              CAL     PPAK     /PRINT THE CHARACTER
         304
         022
24 272   104              JMP     UPTIME   /KEEP TIMING THE UP TIME
         176
         024


         /***ANYTHING BEFORE THIS POINT CAN BE IN PROM***
         /***EVERYTHING AFTER THIS POINT MUST BE IN RAM***

24 275   000   CMMND,  DATA    0        /0=NORMAL MODE, OTHERWISE COMMAND MODE
24 276   000   TWIDTH, DATA    0        /WHEN BYTE IS ZERO, GENERATE A CR/LF
24 277   000   MSSCNT, DATA    0        /TEMP CHARACTER COUNT FOR MESSAGE DUMP
24 300   000   SOH,    DATA    0        /SUBMODE FOR DMPSUB   0=PRINT   1=SEND
24 301   000   BAUD,   DATA    0        /WPM CONSTANT (SEE HEADING ON PROGRAM)
24 302   000   BAUDI,  DATA    0        /INPUT WPM VALUE (GETS MODIFIED)
24 303   000   TIMER,  DATA    0        /TIME BAUD *8 (10 OCTAL) COUNTER
24 304   000   INCHAR, DATA    0        /INPUT CHAR HOLDING REG

24 305   000   KYFIFO, HLT              /INPUT BUFFER

                       *KYFIFO+BUFSKY
25 304   000   PNFIFO, HLT              /PRINTER BUFFER

                       *PNFIFO+BUFSPN
25 364   000   OTFIFO, HLT              /OUTPUT BUFFER FOR CODE

                       *OTFIFO+BUFOUT
26 044   000   MSSGBF, HLT              /MESSAGE BUFFER
```

ASCEND	20 305
ASCTAB	20 225
BAUD	24 301
BAUDI	24 302
BLANK	00 040
BUFOUT	00 060
BUFSKY	00 377
BUFSPN	00 060
C	00 103
CLRMD	22 153
CMMND	24 275
CMMNDR	21 150
CONVT	23 074
CR	00 015
DASH	21 126
DCRLH	21 052
DEL	23 034
DELETE	00 177
DELSYM	00 134
DMPSUB	21 333
DOT	21 120
DSH	23 327
ENDLET	24 247
ENTPAK	20 367
ERCHAR	00 007
ERROFL	21 011
ESC	00 175
ESCSYM	00 044
ETX	00 003
FINDOT	21 137
GOODCH	23 311
IDLE	23 360
IDLE1	24 025
INCHAR	24 304
INCLH	21 046
INCRLF	23 031
INDASH	24 131
INPEND	24 034
INPOK	24 156
INSENS	24 054
INTIME	24 046
KYBD	22 227
KYFIFO	24 305
L	00 114
LDNXT	21 224
LDNXT1	21 232
LDNXT2	21 272
LF	00 012
MCONVT	23 177
MOK1	23 207
MOK2	23 214
MOREND	20 366
MORTAB	20 306
MPAK	24 263
MSGSZ	00 377
MSSCNT	24 277
MSSGBF	26 044
NETX	22 267
NOTEOM	24 235
NTFND	23 225
NTFUND	23 122
NXTPNT	22 361
OK	21 006
OKDASH	24 154
OK1	23 104
OK2	23 111
OTERR	23 252
OTFIFO	25 364
OTLOOP	23 332
OTPUT	23 230
OTPUT1	23 264
OUTEND	23 342
P	00 120
PNFIFO	25 304
PNTR	22 316
PNT1	23 027
POP	21 014
POPLOP	21 023
POPY	21 036
PPAK	22 304
PRT	21 317
PRT1	21 341
PRT2	22 347
PR2	22 004
PR3	22 035
PR4	21 372
QUEST	00 077
RESTRT	20 172
S	00 123
SNDNX	22 060
SOH	24 300
SOH1	22 024
SPACE	23 277
STRT	20 120
STRT1	20 140
T	00 124
TEST	22 074
TEST1	22 120
THIS	23 147
THISIT	23 044
TICK	21 061
TICKI	21 107
TIMER	24 303
TWIDTH	24 276
UNPAK	22 164
UPARRO	00 136
UPTIME	24 176
W	00 127
WAITMT	22 220
WAIT1	21 070
WAIT2	21 067
WHOOP	21 305
WIDTH	00 110
WPM	22 133
XLAT	23 125
XLATER	23 040
XLAT1	23 143

Your Altair, IMSAI, or Poly 88 will love our Frugal Floppy.™



It makes program development and loading 100 times faster!



The affordable floppy

You've made a big investment in your computer. Here's how to make that investment really pay off. Order iCOM's low cost Frugal Floppy™ Disk Subsystem. Just $1195 for the disk drive, field proven controller and cables. And you can order our Altair, IMSAI, and Polymorphic bus compatible plug-in interface, with on-board RAM, in kit form or fully assembled.

What it will do for you

With our Frugal Floppy,™ you can load and store programs hundreds of times faster than with paper tape, cassette, or teletype. Example—8K of memory can be loaded in just 7 seconds! Plus, you can store up to 256 programs on a single disk. That's why we say our Frugal Floppy will turn on your computer. And fast!

Here's what you get

The Frugal Floppy includes:

- Disk drive with daisy chain capability
- Proven IBM compatible controller
- Interface cable to your computer
- Controller-to-disk drive cable
- All required connectors

If you need a power supply and software, we've got that too.

FDOS-II software

iCOM's famous FDOS-II software is now available for the Altair/IMSAI compatible bus. There's nothing anywhere to compare with our disk based Intel compatible macro assembler and string oriented text editor.

With super features, such as named variable length files, auto file create, open and close, multiple file merge and delete. Use our software either in your development system or integrate it into your applications package. In either case, it's easy to do. And low cost too.

Fast delivery

See your iCOM dealer today to find out how our $1195 Floppy can turn on your micro—and fast. If you have any questions, or would like the name of your local dealer, phone us at 213-348-1391. You'll find that when it comes to Microperipherals™, iCOM has the answers. We should. We've delivered thousands of systems to date.

■ MICROPERIPHERALS™
iCOM
6741 Variel Avenue, Canoga Park
California 91303 • (213) 348-1391

DEALERS .... FOR COMPLETE INFORMATION ON HANDLING OUR LINE OF MICROPERIPHERALS™ PHONE OR WRITE iCOM TODAY

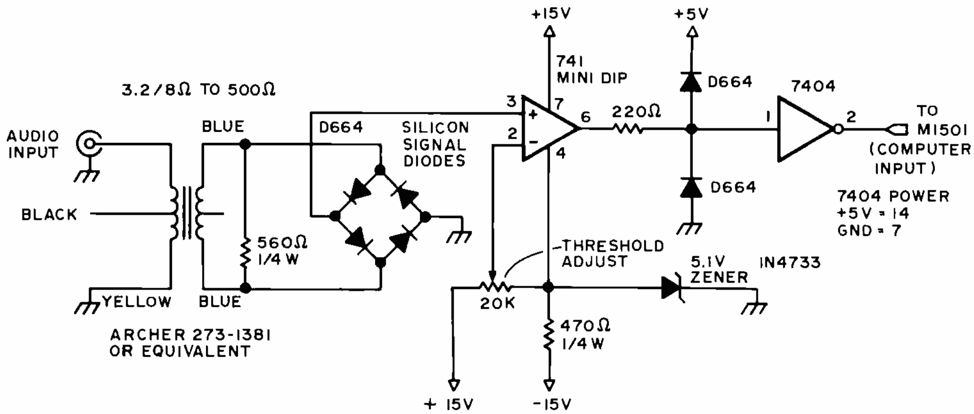


Figure 5: The input circuit used by the author in developing the Morse code interpreter. This circuit works but is not optimal. The operational amplifier saturates if any AC signal is present on the input to the bridge rectifier, so the frequency selectivity of this circuit is virtually nonexistent (ie: all stations heard in the pass band of your receiver will be logically "ORed" leading to garbled copy if you operate in a crowded band).

Some Thoughts on Improvements and Adaptations

The Morse code interpreter described here has been implemented and used by the author. As in any design, there is room for improvement and expansions of the capacity of the program. Here are some suggestions:

- Design a good input filter to pick up audio and output digital (phase locked loop?). See figure 6.
- Add multiple message buffers.
- Use multiple precision arithmetic to provide a larger message buffer.
- Modify the program for RTTY (Replace Morse table with Baudot table, add single byte flag to keep track of FIGS vs LTRS modes).
- Wire 6.3 VAC at 60 Hz into the DEC M7346 module and write a real time clock routine to keep track of time of day.

For individuals with 8080 processors, or the new Z-80, the source code for this Morse code interpreter (see listing 2) can be translated on a one to one basis into code for these newer computers. Such code will work without major changes, but will not make optimal use of the expanded instruction sets.

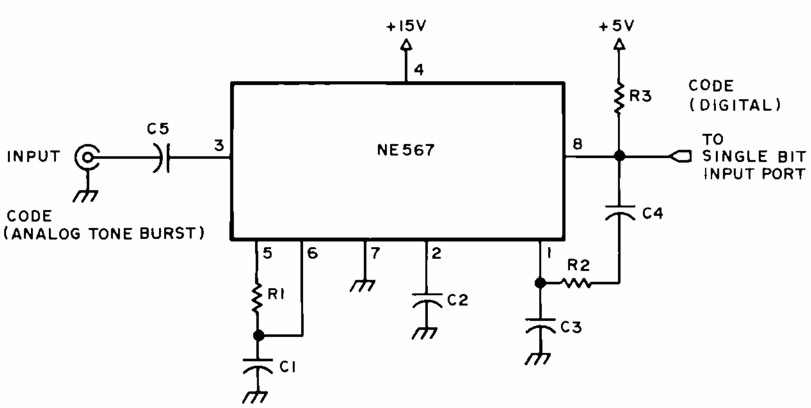


Figure 6: A suggested selective input filter for better performance. This circuit is adapted from the Signetics Catalog, page 6-97. The design equations are shown. The phase lock loop "latches up" when the signal is detected in the filter's band, typically after several tens of cycles. C3 determines the time taken for the filter to detect this condition. The time taken to unlatch after the signal disappears is determined by R2 and C4, with some effects from C3.

Text continued from page 56

used since they are all driven in the same sequence). In both of the above cases the keying sense for input and output is a TTL low level for a "key down" condition ("mark") and a TTL high level for a "key up" condition ("space").

This program has been tested on a starter set and has successfully operated in both PROM and programmable memory. When used in programmable memory, it should be noted that a DEC M7344YB (or an extra 1 K of programmable memory over the starter set M7344YA) is the minimum memory requirement. When the program is assembled and programmed into PROM, approximately 4 1/2 EROMs (1702A) are required.

Experience to Date

The program has been tested in generating Morse code over the speed range of 7.2 wpm to 120 wpm and appears to function properly. The program has been tested in receiving Morse code over the speed range of 7.2 wpm to 96 wpm: up to about 63 wpm the decoding function is fairly acceptable; at 89 wpm the number of erroneous characters is considered to be unacceptable by the author in this particular test. It is the author's opinion that the error rate at the higher code input speeds is probably related to the design of a particular input processing circuit that was used (see figure 5). In general, a phase locked loop, or a similar highly selective decoding scheme, would be useful, particularly to an amateur radio operator working the crowded bands of a field day type event. One such circuit is illustrated in figure 6. This would provide the amateur with a printout of communications in both directions from the station operating. In fact, with sufficient comment being transmitted to and from the stations involved, the printout from a hard copy terminal would provide a log for the field day events.■

Introducing A Remarkable New Microcomputer:

The Astral 2000

You're reading this ad, so you're obviously interested in getting a microcomputer. You're probably also a bit confused by the number of different microcomputer kits around today. So, think about the things you want in a microcomputer. *Ease of assembly, quality power supply, well designed cabinet and interconnect scheme, peripheral and memory options for an expanding system.*

Some Kit!

Although we have called the Astral a "kit", it actually arrives over 70% assembled. The power supply, processor board and RAM board are fully assembled, burned in and tested before shipment.

There is no complicated wiring harness. In fact, there is no front panel wiring harness at all. The front panel plugs directly into the backplane. Additional circuit boards are inserted through the rear of the chassis directly into the backplane.

Complete System

The Astral 2000 is shipped with power supply, cabinet, front panel components, mother board, processor board and one 8K RAM board. The processor is 6800-based and operates in serial and in parallel. Both RS-232 and 20mA current loop are provided by a serial I/O socket on the processor. This processor is shipped with our own 16K monitor ROM and has provisions for "cycle-stealing" DMA. The memory board contains 8K of low power, 500ns static RAM and uses less than 1.5A at 5V.

Lots Of Options

A computer isn't much fun if you can't talk to it. But you can talk to the Astral with the VID-80 video terminal board for only $189.95 unassembled ($245 assembled). The VID-80 has a selectable line length of 64, 72 or 80 characters per line. It displays 16 lines of upper case characters but gives you the option of installing a lower case character ROM as well.

We also have someplace for you to put your programs. Our 8K EPROM board ($59.95) is designed for the 5204 and will allow in-system program storage even during power-down. This board is assembled with all components except the EPROMs, however sockets are provided for the memory chips.

We've solved program loading, too. The I/O tape interface unit ($49.95) plugs into the I/O socket on the processor board and allows programs to be loaded from any inexpensive, non-digital tape deck. But if tape cassette isn't fast or big enough, a floppy disk with an Astral bus-compatible controller will be available for under $1,000 in the last quarter of 1976.

A New And Powerful BASIC

A unique and powerful version of BASIC with features never seen before in an 8K version has been designed especially for the Astral system. Astral BASIC contains all the features of competing BASICs and then some; Astral BASIC is also *very* fast.

With the *User Selectable* Floating Point package, the user chooses the degree of precision from the four choices of 6, 9, 13 or 16 digits. Fewer digits use less memory and is faster, however higher precisions are useful for scientific and mathematic applications.

The Astral BASIC's DO statement is unique; it has never appeared in any other version of BASIC. The DO statement is a simple and flexible way to subroutine without the restrictions of formal subroutines. DOs can be nested, too and — of course — Astral BASIC has all the other standard subroutine procedures as well.

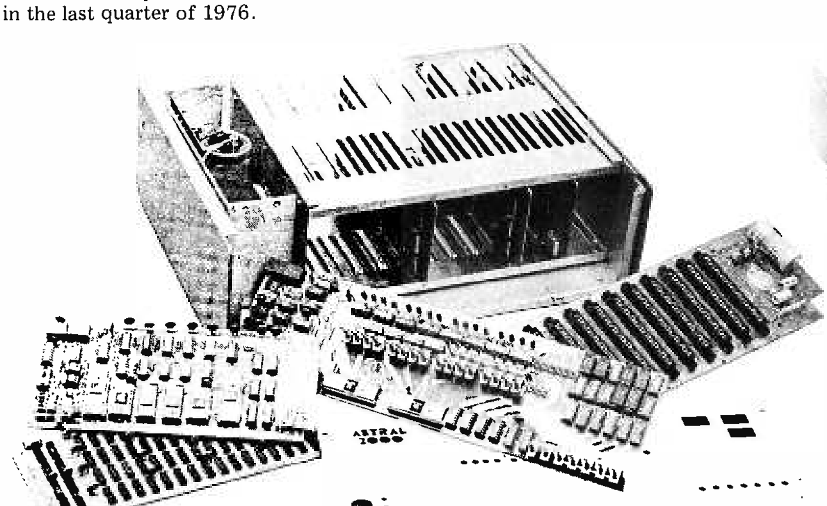
The Trace Mode is another feature rarely found in other BASICs. The Trace Mode is used in program debug to list statement line numbers as executed. This feature may be programmed to Trace On only for routines still needing check-out. Pressing the escape key halts the trace and returns control to the terminal.

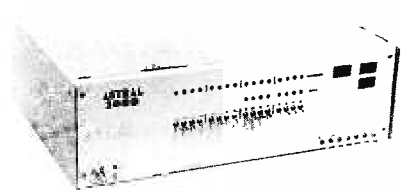
The Astral BASIC string facility permits *variable length strings of unlimited length* and includes the ability to search for a substring within another larger string, a particlularly useful feature for word processing applications.

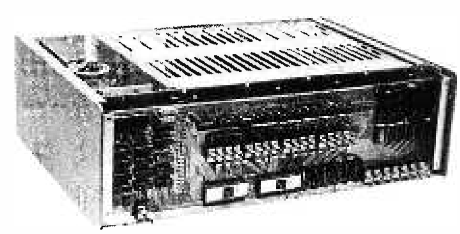
Powerful program editing capabilities allow loading, listing and saving of programs. Blocks of statements may be deleted or renumbered. The RENUMBER statement may be used to increment all specified line numbers and it automatically adjusts the numbering of any GO TOs, etc.

Another feature never seen before in an 8K version is the popular PRINT USING statement. PRINT USING permits floating "$", "+" and "–" signs as well as floating commas, so numbers such as $1,000,000+ can be printed in the standard accounting format.

m&r enterprises
P.O. box 61011, sunnyvale, california 94088







The Astral 2000 is $995 partially assembled ($1250 fully assembled) plus $14 for shipping and handling ($18 for Canadian orders). Additional 8K RAM boards are $245 each. California residents add 6% sales tax. The Software Package includes Astral Basic on magnetic tape cassette or paper tape, the game of Startrek, complete documentation and a free one year subscription to the Astral Newsletter, *all for $35. For more details, send a self-addressed, 8½ by 11 stamped envelope to M&R Enterprises, P.O. Box 61011, Sunnyvale, Ca. 94088. Allow approximately 8 to 12 weeks for delivery.*

Make a Giant of Y

3P+S Input/Output Module

Our 3P+S is a simple inexpensive answer to handling a variety of peripheral devices. It has **two 8-bit parallel I/O ports,** with full handshaking logic. Plus, it has a **serial I/O port,** with a data rate range from 35 to 9600 Baud. Simultaneously handle a paper tape reader, a keyboard, a TV terminal, a tape punch, **and** a telephone coupler! Available with premium grade low-profile IC sockets, **$149.** Owner's Manual, **$4.95.***

ALS-8 Assembly Language Operating System

Just turn on the switch, and instantly you have the power to **write, edit, assemble, de-bug, and run your own programs.** The **ALS-8** is the most **useful** software development tool available today. Optional firmware includes **SIM-1,** an interpretive simulator, that simulates programs without running them in real time. Thus, errors encountered during testing do not endanger your entire system. The **TXT-2** firmware adds the dimension of text editing. Insert, delete, move entire lines or single characters, and much more! **ALS-8** (assembled only), **$425. SIM-1, $95. TXT-2, $95.**

VDM-1 Video Display Module

If you're setting up just the system you really want, don't fall short by limiting its communicating ability. The **VDM-1** is an **ultra-high speed** video output device. Its **16 display lines** have **64 characters each, upper and lower case.** 1024 bytes of random access memory are on the card. It scrolls up or down, **even to 2000 lines per minute!** Any combination of the 1024 cursors can be displayed as black-on-white or vice versa. **Free terminal mode software is included,** along with premium grade, low-profile IC sockets. **$199.** Owner's Manual, **$4.95.***

8KRA Static Memory Module

We now offer a low-power static memory module, with a **full 8192 bytes of memory.** It has an exclusive KSET™ address selection DIP switch, so you can conveniently set address boundaries in increments of 1K. **Our low-power RAM's typically require one-third less power** than those commonly used by our competitors. They'll even retain memory for 4-5 hours when powered by two "D" flashlight cells. **On-board recharging circuitry and battery connectors** make it possible to protect your data against sudden power loss. Each RAM has its own IC socket, too, for easier assembly and repair. **$295.** Owner's Manual, **$4.95.***

2KRO EPROM Module

The **2KRO Read Only Memory** will accept up to eight 1702A or 5203 EPROM's (not included), providing 2048 eight-bit words of non-volatile storage for monitor, executive, loader and other programs. Programming services available from your dealer or write us for details. **$65.** Owner's Manual, **$4.95.***

4KRA Static Memory Module

The 4KRA (4096 bytes) was our first static memory module. It's still very popular, and uses the same **low-power static RAM's** as the 8KRA. Plus, we've added a **DIP switch,** and **every RAM now has its own premium grade, low profile IC socket.** On-board **recharging circuitry** (with battery backup) makes it possible to retain memory for 8-10 hours during power failure. **$159.** Owner's Manual, **$4.95.***

MB-1 Mother Board

Our single piece **Mother Board** for the Altair 8800 gives you 16-card capacity in one single installation. Available with 12 **(MB-12)** or 16 **(MB-1)** slots. **$70.** (Discontinued; limited to stock on hand.)

Wire Wrap Board

Do your own wire wrap prototyping with the **WWB Wire Wrap Board.** Up to 62 16-pin sockets or various combinations of 14, 16, 24 and 40-pin sockets. **$40.**

EXB Extender Board

The **EXB Extender Board** allows accessibility in servicing any 8080-compatible module. **$35.**

*Fully descriptive Owner's Manual available separately. Price refundable with purchase of kit.

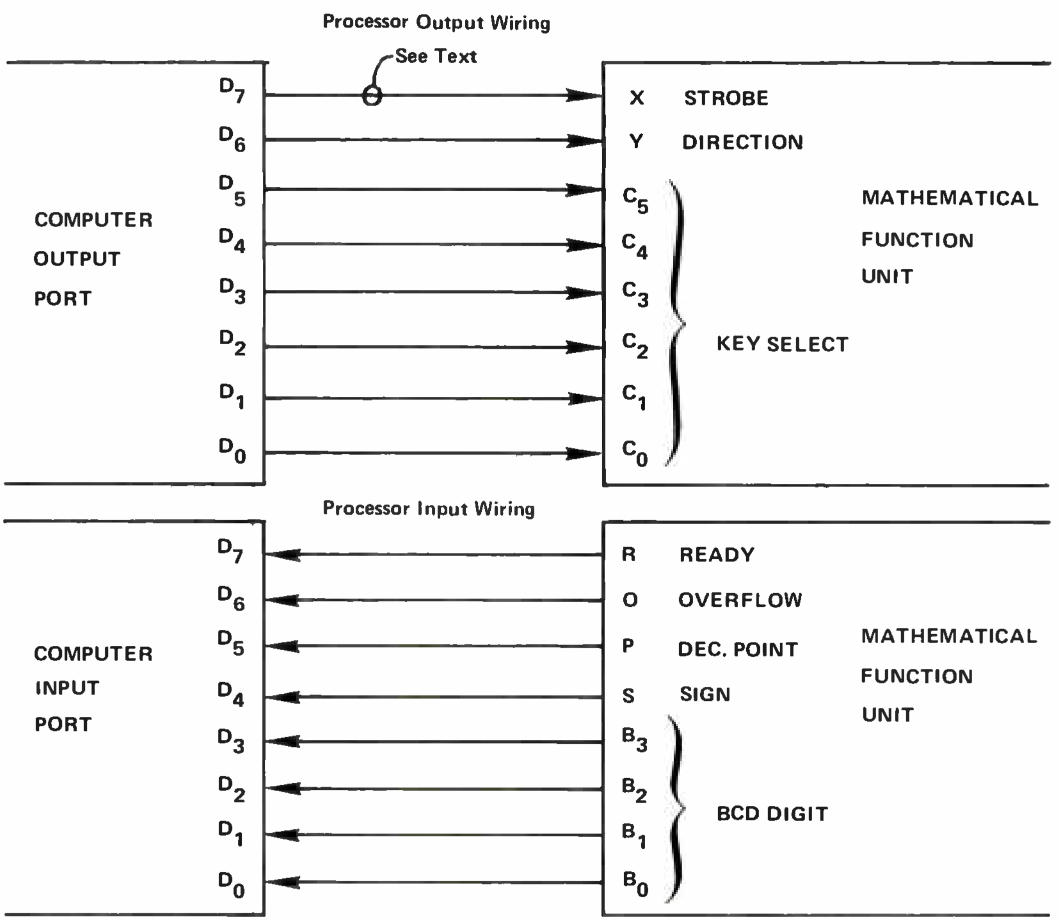
Build This Mathematical Function Unit

Part 2: Software

Figure 1: Wiring the Mathematical Function Unit to your computer's IO structure is accomplished by connecting the data input and output lines to appropriate pins of an 8 bit input and an 8 bit output port. For those homebrewing an interface, the 16 interface lines could be provided by a single dual port integrated circuit such as the Motorola PIA design. The software of this article reflects hardware for the input and latched output sides of a single Intel 8080 port.

Making Connections to the Computer

The connections needed to interface the Mathematical Function Unit to an 8 bit microprocessor are summarized in figure 1. In this interface diagram, only the signal wires are shown. A ground connection must also be made. If the wiring is done as shown here, the READY bit (bit 7) on the Mathematical Function Unit's output must be continually examined to determine completion of the calculating or data entering tasks. An alternative would be to attach the READY line to the computer's interrupt structure so the computer could be executing code other than the constant examination of the READY bit. This would lead to



R Scott Guthrie
1374 Franchere Pl
Sunnyvale CA 94087

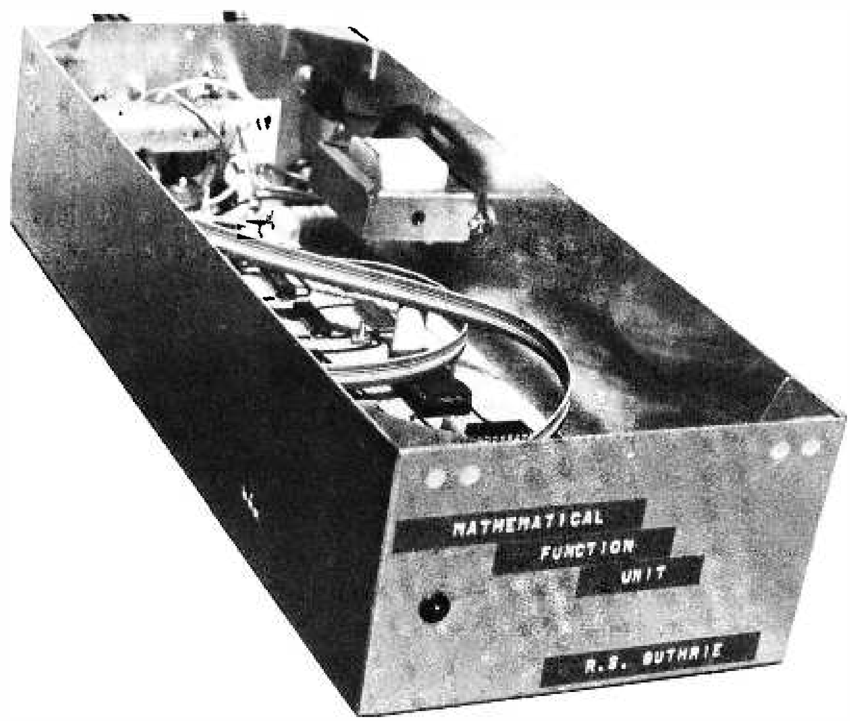
more efficient usage of the microcomputer and overall system, and will only create some fairly easy software changes from the examples shown in this part of the Mathematical Function Unit article. In either case, software toggling of the X (strobe) line is required to initiate operations.

The operation of the Mathematical Function Unit is governed by signals generated in the controlling computer system's software. The minimum required routines consist of an output procedure, an input procedure, and a short routine to check the status of the ready line, bit 7 of the input interface. The Intellec 8 Mod 80 made by Intel and belonging to the California Polytechnic State University Computer Science Department was used for the program development and all the controlling programs were written in Intel 8080 assembly language code as shown in the listings.

Calculator Entry Routine

The Mathematical Function Unit load procedure, called CAENTR (for CAlculator ENTeR) passes the code for the required operation to be performed from the microcomputer's main memory to the Mathematical Function Unit. The 8080 assembly language symbolic and absolute code for this routine is shown in listing 1, hexadecimal addresses 010D to 011F.

The CAENTR routine first saves the contents of the A register with a PUSH PSW instruction. Then it checks the Mathematical Function Unit's ready line, waiting if necessary for the unit to enter the ready state. The waiting is accomplished by a reference to the subroutine RDY at address 010F. The accumulator is then cleared to zero with an XRA A instruction, and this state is sent to the calculator via the 8080's output port 3, symbolically referenced as POUT. Note that in the Intellec system used for the prototype, the hardware inverts the state of the data. Thus all references to data from or to the IO ports have a complement operation



The purpose of this design project was to develop an economical and reliable method of performing simple arithmetic, trigonometric and logarithmic functions in microcomputer systems. Software routines for floating point arithmetic and transcendental functions involve extensive algorithms and complex programming on contemporary microprocessors (indeed, on any computer). A great deal of valuable memory area is often required even for some of the simpler mathematical routines. This is especially true when using the limited instruction sets of the contemporary 8 bit microprocessors. This makes the hardware approach a desirable alternative due to the powerful software available in some calculator integrated circuits on the market today.

In last month's BYTE [page 26], *we provided a description of the hardware for a unique Mathematical Function Unit built around the MOS Technology MPS-7529-103 calculator chip. The parallel IO structure of this interface can be used with any system which has 8 bits of parallel output and 8 bits of parallel input capability. In this month's conclusion to the project, software required to control the interface is described, along with a test program to emulate a hand calculator by use of a computer's hard copy or television display terminal. The software is given as developed for an Intellec 8 Mod 80 system, with notes concerning hardware idiosyncracies of that computer.*

associated with the transfer to compensate for the inversion.

Listing 1: Standard Input/Output Routines. The standard routines CAENTR, CAFTCH, and RDY are presented here in the Intellec 8 Mod 80 assembly language format. The hexadecimal address and object code is listed at the left, with symbolic assembly code at the right. CAENTR moves a command code to the calculator from DIN. CAFTCH reads the current calculator output to DOUT. RDY delays return until the calculator ready state is indicated, thus syncronizing a fast 8 bit central processor with a slow calculator. Prior to using the interface and calculator routines shown in this article, the stack pointer must be set to point to some area of programmable memory which is not in use.

```
                        ;MATHEMATICAL FUNCTION UNIT
                        ;STANDARD INPUT/OUTPUT ROUTINES
                        ;
                        ;*********************************
                        ;NOTE:  ALL INPUT AND OUTPUT DATA
                        ;IS COMPLEMENTED BY HARDWARE AND IS
                        ;RE-COMPLEMENTED BY SOFTWARE FOR
                        ;THIS REASON.
                        ;*********************************
                        ;
                        ;MFU LOAD PROGRAM    (CAENTR)
                        ;
                        ;PROGRAM DESCRIPTION:
                        ;PROGRAM TRANSMITS DATA AT LOC 'DIN'
                        ;TO MFU INPUT PORT
                        ;2 BYTES OF STACK SPACE USED
                        ;NO REGISTERS AFFECTED BY SUBROUTINE
0000                    ORG     010EH
010D            DIN     EQU     010DH
0101            DOUT    EQU     0101H
0003            PIN     EQU     03          ;PORT IN
0003            POUT    EQU     03          ;PORT OUT
010E F5         CAENTR: PUSH    PSW         ;SAVE REG. A
010F CD5201             CALL    RDY         ;TEST FOR READY
0112 AF                 XRA     A           ;ZERO REG. A
0113 2F                 CMA                 ; *** SEE NOTE ***
0114 D303               OUT     POUT        ;TOGGLE MFU
0116 3A0D01             LDA     DIN         ;GET CODE
0119 F6C0               ORI     0C0H        ;FORCE '11XXXXXX'
011B 2F                 CMA                 ; *** SEE NOTE ***
011C D303               OUT     POUT        ;SEND CODE
011E F1                 POP     PSW         ;RESTORE REG. A
011F C9                 RET                 ;RETURN
                        ;
                        ;MFU RETRIEVE PROGRAM   (CAFTCH)
                        ;
                        ;PROGRAM DESCRIPTION:
                        ;DIGITS 1(LSD) THROUGH 12(MSD)
                        ;ARE LOADED INTO MEMORY STARTING
                        ;AT LOC 'DOUT'.   THE INFORMATION
                        ;IS OF THE FORM "ROPSBBBB" WHERE:
                        ;R=READY BIT, P=DECIMAL POINT
                        ;O=OVERFLOW, S=SIGN, BBBB=BCD DIGIT.
                        ;6 BYTES OF STACK USED.
                        ;NO REGISTERS AFFECTED BY SUBROUTINE.
                        ;
0120 F5         CAFTCH: PUSH    PSW         ;SAVE A & PSW
0121 C5                 PUSH    B           ;SAVE B & C
0122 E5                 PUSH    H           ;SAVE H & L
0123 060C               MVI     B,0CH       ;LOAD 12 IN REG. B
0125 210101             LXI     H,DOUT      ;LOAD ADDR. OF SAVE AREA
0128 CD5201             CALL    RDY         ;TEST FOR READY
012B F3                 DI                  ;DISABLE INTERRUPTS
012C AF                 XRA     A           ;ZERO REG. A
012D 2F                 CMA                 ; *** SEE NOTE ***
012E D303               OUT     POUT        ;TOGGLE MFU
0130 3ECB               MVI     A,0CBH      ;DISPLAY RESTORE KEY
0132 2F                 CMA                 ; *** SEE NOTE ***
0133 D303               OUT     POUT        ;SEND CODE
0135 CD5201             CALL    RDY         ;TEST FOR READY
0138 AF         LOOP1:  XRA     A           ;ZERO REG. A
0139 2F                 CMA                 ; *** SEE NOTE ***
013A D303               OUT     POUT        ;TOGGLE MFU
013C 3E80               MVI     A,080H      ;SET TOGGLE BIT
```

The calculator command code to be sent to the Mathematical Function Unit is passed to the CAENTR routine in memory location DIN. This command code is combined with the bit pattern 11000000 using the ORI operation at address 0119. This forces both the X and Y control bits of the computer output port to be in a high state. The X line high starts the Mathematical Function Unit load sequence, and the Y line in a high state indicates that data is to be loaded into the calculator from the computer's port. The low order bits contain the bit pattern of the calculator key selection taken from table 1 on page 31 of September 1976 BYTE. Finally, the CAENTR routine restores the saved contents of the A register and returns to the calling routine.

In using the calculator interface, the procedure is quite simple: Set up a calculator control code from that table in the location DIN (hexadecimal 010D in this case) and then CALL CAENTR. One command to the calculator is transferred for each such CAENTR call.

Calculator Fetch Routine

The Mathematical Function Unit read routine, called CAFTCH (for CAlculator FeTCH) is responsible for controlling the transfer of data from the unit into the computer's memory.

This routine transfers all 12 digit positions of the calculator chip's output number into 12 sequential locations of the computer's memory starting at location DOUT, shown at location 0101 hexadecimal in listing 1. CAFTCH begins by saving the processor status. Then the subroutine RDY is called to delay until the calculator is ready for data transfer.

When the Mathematical Function Unit is ready, return from RDY is followed by disabling of interrupts with the DI instruction, so that complete transfer of all 12 bytes of calculator display output can occur without interruption. The calculator's "display restore" command *[see table 1, page 31, BYTE September 1976]* of 0B hexadecimal is then sent, in the form of the code CB (logical sum of 0B and C0). This insures that the Mathematical Function Unit output will be valid prior to reading the data into 12 bytes of memory. After another CALL RDY wait, the program enters a loop extending from address 0138 hexadecimal to 014A. This loop transfers 12 output digits in sequence into 12 consecutive memory locations starting at the location DOUT. (The initial address was set up by the LXI H at location 0125.) During this loop, the Y control line, bit 6 of the computer output port, is left at a logic 0 level. The X line, bit 7 of the computer output port, is toggled by the instructions at locations 0138 to 013F, setting up the next output transfer from the calculation. After waiting for the ready condition, the data is transferred into the computer at location 0144 with an IN PIN instruction, addressing the input port of the computer. The byte is then saved in memory, and the index register provided by the HL pair is incremented. A loop counter in register B is decremented, and if the loop is not completed, reiteration continues. After all 12 digits have thus been transferred, the interrupts are reenabled and the registers are restored prior to return.

The format of data recovered from the calculator in this transfer process was given in figure 2a found on page 27 of last month's BYTE. The offsets in memory for each byte of the data format were given in figure 2a.

Calculator Ready Routine

The subroutine called RDY is the final component of the basic set of interface driver software. This program is found at addresses 0152 to 015A of listing 1, and is a simple procedure to input from the computer input port (symbolically PIN, absolute value 3 in this case), and test the ready bit, bit 7 of the input pattern. If the ready state is indicated, the subroutine returns, otherwise it keeps reiterating. This routine is used by both CAENTR and CAFTCH.

If the interrupt structure is being implemented, this routine could return control to the operating system, or other programs in memory, allowing them to execute until being interrupted by the ready state of the Mathematical Function Unit.

Alignment Procedures

The three variable resistors, R5, R6, and R7 adjust the length of the following timing pulses:

R5	300 μs	Set Not-Ready delay
R6	50 ms	Key Pressed delay
R7	50 ms	Key Released delay

The use of three short timing loop programs, with the aid of an oscilloscope, allow these delays to be set. The programs are found in listing 2 and assume a 2 μs cycle time. Adjustments should be made if your 8080 processor runs at a different speed.

The 300 μs pulse can be set by connecting the scope probe to pin 13 of IC 1a while executing the 1 ms delay routine, D1MS. The scope's time base should be set at 50 μs per division and R5 adjusted until the pulse length is six divisions long. This pulse length is not at all critical and does not require exact setting.

The key pressed delay must be approx-

Listing 1, continued:

```
013E 2F                  CMA                ; *** SEE NOTE ***
013F D303                OUT     POUT       ;SEND CODE
0141 CD5201              CALL    RDY        ;TEST FOR READY
0144 DB03                IN      PIN        ;GET CODE
0146 2F                  CMA                ; *** SEE NOTE ***
0147 77                  MOV     M,A        ;SAVE DIGIT
0148 23                  INX     H          ;INCR H & L
0149 05                  DCR     B          ;DECREMENT B
014A C23801              JNZ     LOOP1      ;CHECK FOR DONE
014D FB                  EI                 ;ENABLE INTERRUPTS
014E E1                  POP     H          ;RESTORE H & L
014F C1                  POP     B          ;RESTORE B & C
0150 F1                  POP     PSW        ;RESTORE A & PSW
0151 C9                  RET                ;RETURN
                         ;
                         ;READY SUBROUTINE
                         ;THIS SUBROUTINE, WHEN CALLED
                         ;RETURNS TO CALLING ROUTINE ONLY
                         ;WHEN MFU IS IN THE READY STATE.
0152 DB03       RDY:     IN      PIN        ;GET MFU INFO
0154 2F                  CMA                ; *** SEE NOTE ***
0155 E680                ANI     080H       ;GET READY BIT
0157 CA5201              JZ      RDY        ;JUMP NOT READY
015A C9                  RET                ;RETURN
0000                     END
```

Listing 2: Alignment of the oneshots in this circuit is accomplished using timing loop programs which repeatedly toggle certain aspects of the interface. This listing shows three such timing loops, D1MS, D70MS and D140MS.

```
                         ;ALIGNMENT ROUTINES
                         ;1., 70., AND 140 MS. DELAY ROUTINES
                         ;FOR COMPUTER CYCLE TIME OF 2 US.
                         ;
0000                     ORG     200H       ;STARTING ADDRESS FOR 1 MS
0200 CD0B04     D1MS:    CALL    OUTPUT     ;OUTPUT STARTING SEQ.
0203 067C                MVI     B,7CH      ;SET FOR 1 MS. DELAY
0205 05         TLOOP:   DCR     B
0206 C20502              JNZ     TLOOP
0209 C30002              JMP     D1MS
                ;
020C                     ORG     300H       ;STARTING ADDRESS FOR 70 MS
0300 CD0B04     D70MS:   CALL    OUTPUT
0303 0623                MVI     B,35       ;SET FOR 70 MS. DELAY
0305 CD1604              CALL    DELAY
0308 C30003              JMP     D70MS      ;REPEAT
                         ;
030B                     ORG     400H       ;STARTING ADDRESS FOR 140 MS
0400 CD0B04     D140MS:  CALL    OUTPUT
0403 0644                MVI     B,68       ;SET FOR 140 MS. DELAY
0405 CD1604              CALL    DELAY
0408 C30004              JMP     D140MS     ;REPEAT
                         ;
                         ;SUBROUTINES
                         ;
040B 3E7F       OUTPUT:  MVI     A,07FH     ;SET X LINE LOW
040D 2F                  CMA                ;INVERT FOR OUTPUT
040E D303                OUT     03
0410 3EFF                MVI     A,0FFH     ;SET X LINE HIGH
0412 2F                  CMA                ;INVERT FOR OUTPUT
0413 D303                OUT     03
0415 C9                  RET                ;RETURN
                         ;
0416 3C         DELAY:   INR     A
0417 C21604              JNZ     DELAY      ; *DELAY*
041A 05                  DCR     B          ; *LOOP*
041B C21604              JNZ     DELAY
041E C9                  RET                ;RETURN
0000                     END
```

Listing 3: CALCULA, a Calculator Simulator. CALCULA demonstrates the use of the Mathematical Function Unit in an application program. The purpose of CALCULA is to drive the calculator interface as a printing calculator, interpreting ASCII codes from a keyboard as the key strokes on a typical hand calculator. The output of the calculator is displayed after every return operation. The original CALCULA was run with a Teletype for input and output.

```
                                ;MFU-CALCULATOR SIMULATOR
                                ;TTY KEYBOARD SIMULATES
                                ;A HAND HELD CALCULATOR.
                                ;
                                ;***********************************
                                ;NOTE:  ALL INPUT AND OUTPUT DATA
                                ;IS COMPLEMENTED BY HARDWARE AND IS
                                ;RE-COMPLEMENTED BY SOFTWARE FOR
                                ;THIS REASON.
                                ;***********************************
                                ;
0000                    ORG     200H      ;STARTING ADDRESS
010D           DIN      EQU     010DH
0101           DOUT     EQU     0101H
0120           CAFTCH   EQU     0120H
010E           CAENTR   EQU     010EH
                        ;
0200 21B302             LXI     H,HEAD    ;H,L -> HEADING
0203 061C               MVI     B,28      ;HEADING COUNT
0205 4E        PHEAD:   MOV     C,M       ;MOVE CHAR TO C
0206 CDA702             CALL    CO        ;PRINT CHARACTER
0209 23                 INX     H         ;INCR. MEM. PTR.
020A 05                 DCR     B         ;DEC COUNT
020B C20502             JNZ     PHEAD     ;JUMP NOT DONE
020E CD9C02    LOOP:    CALL    CI        ;GET INPUT CODE
0211 E67F               ANI     7FH       ;REMOVE 8TH BIT
0213 FE0D               CPI     0DH       ;COMPARE WITH RETURN
0215 C27102             JNZ     CODEIN    ;JUMP IF NOT RETURN
0218 CD2001             CALL    CAFTCH    ;GET DATA FROM MFU
021B 4F                 MOV     C,A       ;MOVE RETURN TO C
021C CDA702             CALL    CO        ;PRINT RETURN
021F 0E0A               MVI     C,0AH     ;MOVE LINEFEED TO C
0221 CDA702             CALL    CO        ;PRINT LINEFEED
0224 160C               MVI     D,12      ;DIGIT COUNT TO D
0226 210101             LXI     H,DOUT    ;H,L -> DOUT
0229 7E        DLOOP:   MOV     A,M       ;C(H,L) -> REG. A
022A E67F               ANI     07FH      ;FORCE '0XXXXXXX'
022C 77                 MOV     M,A       ;REPLACE # IN MEMORY
                        ;CHECK FOR OVERFLOW INDICATOR
022D E6BF               ANI     0BFH      ;FORCE  'X0XXXXXX'
022F BE                 CMP     M         ;SAME?
0230 CA3D02             JZ      CONT1     ;YES = JUMP, ELSE,
0233 0E2A               MVI     C,2AH     ;"*" -> REG. C
0235 CDA702             CALL    CO        ;PRINT "*"
0238 1601               MVI     D,01H     ;1 -> D  (NO MORE PRINT)
023A C35F02             JMP     CONT      ;BACK TO ROUTINE
                        ;CHECK FOR A NEGATIVE SIGN IN CODE
023D E6EF      CONT1:   ANI     0EFH      ;FORCE 'XXX0XXXX'
023F BE                 CMP     M         ;SAME?
0240 CA4B02             JZ      CONT2     ;YES = JUMP, ELSE,
0243 0E2D               MVI     C,2DH     ;"-" -> REG. C
0245 CDA702             CALL    CO        ;PRINT "-"
0248 C35302             JMP     DECPT     ;GO TO DECIMAL PT. ROUTINE
                        ;PRINT # FOUND
024B E60F      CONT2:   ANI     0FH       ;STRIP OFF D. P.
024D C630               ADI     030H      ;CONVERT TO ASCII
024F 4F                 MOV     C,A       ;C(A) -> C
0250 CDA702             CALL    CO        ;PRINT NUMBER
0253 7E        DECPT:   MOV     A,M       ;C(M) -> A
0254 E6DF               ANI     0DFH      ;FORCE 'XX0XXXXX'
0256 BE                 CMP     M         ;SAME?
0257 CA5F02             JZ      CONT      ;YES = JUMP, ELSE,
025A 0E2E               MVI     C,02EH    ;"." -> REG. C
025C CDA702             CALL    CO        ;PRINT "."
025F 23        CONT:    INX     H         ;INCREMENT H,L
0260 15                 DCR     D         ;COUNT DOWN CHARACTERS
0261 C22902             JNZ     DLOOP     ;CONTINUE DISPLAY
0264 0E0D               MVI     C,0DH     ;RETURN -> REG. C
0266 CDA702             CALL    CO        ;PRINT RETURN
0269 0E0A               MVI     C,0AH     ;LINEFEED -> REG. C
026B CDA702             CALL    CO        ;PRINT LINEFEED
026E C30E02             JMP     LOOP      ;DONE WITH OUTPUT
0271 210003    CODEIN:  LXI     H,TABLE   ;H,L -> TABLE
0274 1E35               MVI     E,53D     ;53 = # OF ENTRIES IN TABLE
0276 FE20               CPI     20H       ;COMPARE WITH SPACE
```

imately 40 ms long for the MPS 7529-103 Calculator Chip and is set using the 70 ms program loop, D70MS. With the scope's input connected to pin 5 of IC 1b and the time base set for 10 ms per division, R6 should be adjusted for five divisions (50 ms). This allows an extra 10 ms from the required 40 ms minimum for assurance that the data will be received in worst case situations.

The key released delay can be set using the 140 ms delay loop, D140MS. Since the 50 ms key released delay does not start until after the 50 ms key pressed delay, the end of this pulse should be around 100 ms from the initiation of the sequence. This can be seen on pin 13 of IC 19a and set by adjusting R7. The scope's external trigger can be connected to pin 5 on IC 1b and the time base set for 10 ms per division for a closer look.

It is not imperative that all pulse delays be exact; however data transmission errors may result if the two 50 ms delays are not set for at least 40 to 45 ms.

What's It Good For?

The application of the Mathematical Function Unit is appropriate wherever calculations must be done. To illustrate a specific case, listing 3 provides a very simple calculator program which will enable an 8080 to drive the unit through the routines of listing 1. A sample of the output is shown in listing 4. The purpose of the CALCULA program is to accept inputs from the normal ASCII keyboard of your computer (here assumed to be connected to a Teletype using input port 0 for data input, input port 1 for status, and output port 0 for data output) and use these inputs to set up command sequences to the calculator. After each "=" operation, the current display output of the calculator is read and printed.

The program is set up in a fairly straightforward manner, using a table located at hexadecimal location 0300 to store the conversion between ASCII input characters and calculator control characters. The list of ASCII codes and their corresponding calculator functions is found in table 1.

To use CALCULA, simply load memory address space from 0100 to 0334 (hexadecimal) with the content of listings 1 and 3, then start the CALCULA program by jumping to location 0200. This begins execution by printing out the heading MFU CALCULATOR SIMULATOR shown at the beginning of listing 4, after which an interactive input of various calculator commands from table 2 can begin. The sample of listing 4 shows uses of many of the calculator functions. The calculator replies with an asterisk (*) if an overflow occurs, as is the case with the attempt at 100! (100 factorial). Display output is normally printed following the input of a carriage return.

CALCULA is of course only the simplest of possible uses for this unit. For general programming, specialized routines could be written to execute sequences of calculator keystrokes when needed, using data kept in the 12 byte arithmetic format.

It should be fairly easy to implement a simple programmable calculator style interpreter to drive this interface, thus converting your personal computer into the equivalent of some fairly expensive desk top microcomputer packages being sold commercially. Ambitious readers will go even further and implement a BASIC interpreter or some other form of high level language referencing this machine both at compile time and in the run time software packages.

Some Parting Comments

The hardware problems encountered with this design were few. The major inelegancy of the design is probably the requirement for pull down resistors on the calculator chip's output lines. These resistors (R10 to R30) are required to supply the load normally supplied by digit and segment driver circuits,

Listing 4: CALCULA Program Sample Run. This Teletype listing was made using the CALCULA program to drive the Mathematical Function Unit. The command codes for operations are listed in table 1. A carriage return code (hexadecimal OD) is used to cue the display of calculator outputs after a transfer via CAFTCH.

```
MFU CALCULATOR SIMULATOR
$$
000000000.000
123.456 + 345.678 =
000469.134000
$   ((3+7)*(2+1)/10)*4+3!=
000000018.000
$   987.321M
000987.321000
$$  10↑14H + 26.579↑15H =
0001.26579-13
$$  45 S
00.7071068000
$   P
03.1415927000
$$  16. I
00000.0625000
$   69!
01.7112244098
$   100!
*
$$  1 E
02.7182818000
$   10 R
03.1622777000
$$  10 Y 2 I =
03.1622777000
$$  8 Q
000000064.000
$$  F
000987.321000
$Z
0.00000000.000
$   53C
-.0.9182828000
$   Z
000000000.000
$   1000. D
000001000.000
$   F
001987.321000
$$
000000000,000
```

Listing 3, continued:

```
0278 C28202            JNZ    AGAIN    ;JUMP IF NOT SPACE
027B 4F                MOV    C,A      ;MOVE SPACE TO REG. C
027C CDA702            CALL   CO       ;PRINT SPACE
027F C30E02            JMP    LOOP     ;BACK TO LOOP
0282 BE        AGAIN:  CMP    M        ;COMPARE C(A) WITH TABLE
0283 CA8E02            JZ     FOUND    ;JUMP IF FOUND
0286 1D                DCR    E        ;DECREMENT COUNTER
0287 CA0E02            JZ     LOOP     ;IF NOT FOUND TRY AGAIN
028A 23                INX    H        ;INCR H,L
028B C38202            JMP    AGAIN    ;KEEP TRYING
028E 4F        FOUND:  MOV    C,A      ;MOVE FOR PRINT
028F CDA702            CALL   CO       ;PRINT
0292 7D                MOV    A,L      ;MOVE CODE TO A
0293 320D01            STA    DIN      ;STORE CODE FOR MFU
0296 CD0E01            CALL   CAENTR   ;SEND CODE TO MFU
0299 C30E02            JMP    LOOP     ;BACK TO ROUTINE
                       ;
                       ;CHARACTER INPUT ROUTINE
029C DB01      CI:     IN     01       ;01 = TTY STATUS PORT
029E E601              ANI    01       ;01 = MASK FOR DATA AVAILABLE
02A0 C29C02            JNZ    CI       ;JUMP IF NO DATA
02A3 DB00              IN     00       ;READ THE CHARACTER
02A5 2F                CMA             ;*** SEE NOTE ***
02A6 C9                RET             ;RETURN TO PROGRAM
                       ;
                       ;CHARACTER OUTPUT ROUTINE
02A7 DB01      CO:     IN     01       ;01 = TTY STATUS PORT
02A9 E604              ANI    04       ;04 = MASK FOR TTY BUSY
02AB C2A702            JNZ    CO       ;LOOP UNTIL READY
02AE 79                MOV    A,C      ;MOVE FOR PRINT
02AF 2F                CMA             ;*** SEE NOTE ***
02B0 D300              OUT    00       ;OUTPUT CHARACTER
02B2 C9                RET             ;RETURN TO PROGRAM
                       ;
                       ;
02B3 0D        HEAD:   DB     0DH      ;RETURN
02B4 0A                DB     0AH      ;LINEFEED
02B5 4D465520          DB     'MFU CALCULATOR SIMULATOR'
02B9 43414C43
02BD 554C4154
02C1 4F522053
02C5 494D554C
02C9 41544F52
02CD 0D                DB     0DH      ;RETURN
02CE 0A                DB     0AH      ;LINEFEED
                       ;
                       ; *** END OF PROGRAM ***
                       ;
                       ;
02CF                   ORG    0300H
0300 30313233  TABLE:  DB     '0123456789AD'
0304 34353637
0308 38394130
030C 30303030          DB     '0000.+-*/Y=('
0310 2E2B2D2A
0314 2F593D28
0318 2950485E          DB     ')PH↑0000SCTL'
031C 30303030
0320 5343544C
0324 47524644          DB     'GRFDWZM$0000'
0328 575A4D24
032C 30303030
0330 49515845          DB     'IQXE!'
0334 21
                       ;
0000                   END    ;END OF TABLE
```

NOTE: A 10 by 5¼ inch (25.4 by 13.3 cm) printed circuit board with plated through holes is available from RSG Electronics, POB 13, Santa Margarita CA 93453. The price of this board is $24.95 plus $1.23 for postage and handling (California residents add 6% sales tax). For questions on obtaining the calculator chip or other parts, please write to the author.

Table 1: Keyboard Code Assignments for CALCULA. The CALCULA program interprets the ASCII keystrokes listed in the "keyboard code" column as the corresponding calculator key functions in the "function" column. The current content of the output displays is printed whenever an ASCII carriage return is input.

Keyboard Code	Function	Keyboard Code	Function
0	0	E	e^x
1	1	!	N! Factorial
2	2	Y	Y^x
3	3	P	Pi (3.1415927)
4	4	A	Arc
5	5	S	Sine
6	6	C	Cosine
7	7	T	Tangent
8	8	L	Natural Log
9	9	G	Log base 10
.	Decimal Point	Z	Degree to Radian
+	Plus		Mode Switch
—	Minus	M	Store in Memory
*	Multiply	F	Recall from
/	Divide		Memory
=	Equals	D	Add to Memory
(	Left Parenthesis	W	Swap X with Y
)	Right Parenthesis	$	Clear Entry
↑	Enter Exponent	$$	Clear All
H	Change Sign	Carriage	Output Current
I	Inverse I/X	Return	Contents of Dis-
Q	X^2		play to Main
R	Square Root of X		Memory and
X	10^x		Output

which are missing due to the use of MOS buffers in their place.

The Mathematical Function Unit has been overdesigned in several areas. The ready and timing circuitry has been designed for easy modification to accommodate some different calculator chips in place of the one used in this project. The timing pulses can be extended or shortened, or even bypassed if, for example, the key released delay is not needed. All modifications must, however, be carefully considered with the chip's specifications prior to altering the electronics.

The Mathematical Function Unit and the related software routines provide the microcomputer user with a very powerful calculation tool where little could easily be done before. The programming of this peripheral calculator has been found to be very straightforward and uncomplicated, requiring much less memory space (and design time) than would be required to perform mathematical routines with microcomputer software alone.

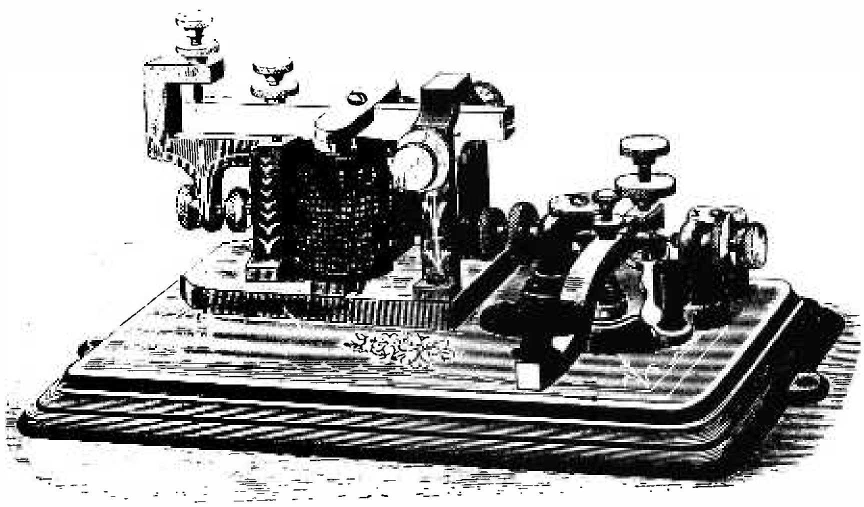
After considering the speed versus calculation ability tradeoffs, as well as the points mentioned above, I feel this project has been a complete success. I hope the microcomputer hobbyist as well as system designers will take advantage of this method for satisfying their mathematical function requirements in cases where high precision and low speed prove usable.■

BIBLIOGRAPHY

1. Intel Corporation: *Intel Data Catalog*, 1975, pages 6-1 through 6-64.
2. Intel Corporation: *Intellec 8/Mod 80 Microcomputer Development System Reference Manual*, 1974.
3. Intel Corporation: *Intellec 8/Mod 80 Microcomputer System Operator's Manual* (Preliminary Edition), June 1974.
4. MOS Technology Inc: "Specifications for Single Chip 40 Key Scientific Calculator Array MPS 7529-103" Rev 1, July 21 1975.
5. National Semiconductor Corporation: *CMOS Integrated Circuits*, August 1974.
6. National Semiconductor Corporation: *Digital Integrated Circuits*, January 1974.
7. National Semiconductor Corporation: *Interface Integrated Circuits*, April 1974.
8. National Semiconductor Corporation: *Linear Integrated Circuits*, January 1974.
9. Staken, Patrick H: "Using a Calculator Chip to Extend a Microprocessor's Capabilities," *Computer Design* September 1975, pages 98, 99.
10. Texas Instruments Inc: *The TTL Databook for Design Engineers*, First Edition, 1973.

SOME HISTORICAL NOTES ON COMMUNICATIONS AND PEOPLE

Source materials supplied by Claude Kagan, Western Electric Co.



"PRIVATE LINE" INSTRUMENT.

Courtesy of Western Electric Company from "Catalogue No. VII" of 1884.

The following text is taken exactly from a privately circulated English language translation of an excerpt (pages 519 to 523) of a French work, Traite De Telegraphie Electrique *by Abbe Moigno, published in Paris in 1852.*

Telegraph Lines in America

Electric telegraphy has not spread anywhere its mysterious network with as much speed and success as in the United States of North America; nowhere has one either conceived as promptly and executed with such admirable response the happy thought of placing commerce and private enterprise in possession of this all powerful means of communication.

The first American telegraph line was established in 1844, between Washington and Baltimore, on a length of 40 miles. It transmitted with such celerity and promptness the news relating to the election of presidential candidates, that all proclaimed, with a unanimous voice, the excellence and immense importance of these messages prompt as lightning; and one saw the formation at once, on all of the territory of the Union, private companies with the strong intention of making the entire country participate in the unexpected benefit of the electric telegraph. The line from Washington to Baltimore was immediately extended to Philadelphia and New York, over a distance of 250 miles; it reached Boston in 1845, and became the great Northern line on which two lines branched: one, a thousand miles long from Philadelphia to Harrisburg, Lancaster, Pittsburgh, Ohio, Columbo, Cincinnati, Louis Ville (Kentucky), and St. Louis (Missouri); the other, 1,300 miles long, from New York to Albany, Troy, Utica, Rochester, Buffalo, Erie, Cleveland (Ohio), Chicaga (Illinois), Milwankie (Wisconsin).

[The text goes on to describe further details and give a table totalling the telegraph mileage as of July 1849: 10,885 miles.]

What Was the Public Response to the Telegraphy Revolution?

[Continuing the excerpt] . . . Here is how the work is accomplished: the poles, twenty feet long, are cut in the forests and transported to the site by the neighboring farmers; one implants them into holes a foot and a half deep, one fastens at some distance from the top insulators or glass rings, on which the conductive wire rests. A few days suffice to see the rise, as by enchantment, of a long line of poles. The surveillance and protection of the poles and of the wire is entrusted to these same farmers, all provided with the small instruments required by such a simple task: a hammer, cutters, a vise, pliers, some screws, some rings and some nails.

The telegraph lines usually follow post roads, and their maintenance costs absolutely nothing; there is no instance that the wires, delivered to the good public sense, were broken through meaness. It is because in America all the wishes cooperate in concert for the public welfare, and that all useful invention is like a family heirloom that all wish to safeguard. Each farmer participates at the very least by his contribution of labor and materials, if not through his purse, to all the national undertakings; each thus becomes a shareholder and is interested in seeing them prosper.■

Microprocessor Update:

Keep PACE with the Times

Robert Baker
15 Windsor Dr
Atco NJ 08004

The new National Semiconductor IPC-16A/500D microprocessor commonly called PACE for Processing And Control Element is a single chip 16 bit microprocessor packaged in a standard 40 pin dual in line package. As a 16 bit general purpose computer it is an excellent candidate for personal computing applications.

Table 1: PACE Status and Control Flags. These flags are tested to control program execution as a result of various conditions.

Flag Register Bit	Flag Name	Description
0	1	Always logic 1, not used.
1	IE1	Flags IE1 to IE5 serve as interrupt enable flags for 5 of the 6 PACE interrupt levels. These interrupts are ignored when IE bit is low.
2	IE2	
3	IE3	
4	IE4	
5	IE5	
6	OVF	Set to the state of the 2's complement arithmetic overflow by arithmetic instructions.
7	CRY	Set to the state of the binary or decimal carry output of adder by arithmetic instructions.
8	LINK	Link flag is included in shift and rotate operations as specified by instructions. Is unaffected if not selected.
9	IEN	Master interrupt enable. This line simultaneously inhibits all five of lowest priority interrupt levels.
10	BYTE	Byte flag selects 8 bit data length when high and 16 bit when low.
11	F11	General purpose control flags. Drive PACE output pins to directly control system functions.
12	F12	
13	F13	
14	F14	
15	1	Always 1, addressing for interrupt 0 exit.

The Microprocessor Overview

PACE utilizes 16 bit instruction words to operate on software selectable, 8 or 16 bit data words for a wide range of applications. There are four 16 bit general purpose working registers available to the user as well as an independent 16 bit status and control flag register that automatically and continuously preserves system status. Table 1 lists the various bits of the status and control register and gives a brief description of the function of each bit. A ten word (16 bit) last in, first out, stack automatically saves return addresses for subroutine calls and interrupt servicing. A stack full or stack empty interrupt is provided to allow software stack expansion when required. A six level, vectored priority interrupt system provides automatic interrupt identification. More than one device may be placed on any given priority level using open collector, wired OR circuitry external to the processor. An individual interrupt enable is provided in the status register for each level and a master interrupt enable is provided for all five lower priority levels as a group. There are also four direct sense inputs and four control flag outputs to implement various single bit status and control functions. Figure 1 shows a functional block diagram of the PACE microprocessor while figure 2 shows the actual pin connections of the dual in line package.

Instruction Set

The PACE microprocessor instruction set consists of a general purpose mixture of 45 instruction types in eight classes as shown in

Table 2: PACE Instruction Set Summary. This table lists the mnemonic and a short description for all of the PACE instructions. For complete functional descriptions and bit patterns consult National Semiconductor's PACE Technical Description *document.*

Mnemonic	Operation (or meaning)
Branch Instructions	
BOC	Branch on condition
JMP	Jump
JMP@	Jump indirect
JSR	Jump to subroutine
JSR@	Jump to subroutine indirect
RTS	Return from subroutine
RTI	Return from interrupt
Skip Instructions	
SKNE	Skip if not equal
SKG	Skip if greater
SKAZ	Skip if And is zero
ISZ	Increment and skip if zero
DSZ	Decrement and skip if zero
AISZ	Add immediate, skip if zero
Memory Data Transfer Instructions	
LD	Load
LD@	Load indirect
ST	Store
ST@	Store indirect
LSEX	Load with sign extended
Memory Data Operate Instructions	
AND	And
OR	Or
ADD	Add
SUBB	Subtract with borrow
DECA	Decimal add
Register Data Transfer Instructions	
LI	Load immediate
RCPY	Register copy
RXCH	Register exchange
XCHRS	Exchange register and stack
CFR	Copy flags into register
CRF	Copy register into flags
PUSH	Push register onto stack
PULL	Pull stack into register
PUSHF	Push flags onto stack
PULLF	Pull stack into flags
Register Data Operate Instructions	
RADD	Register add
RADC	Register add with carry
RAND	Register and
RXOR	Register exclusive or
CAI	Complement and add immediate
Shift and Rotate Instructions	
SHL	Shift left
SHR	Shift right
ROL	Rotate left
ROR	Rotate right
Miscellaneous Instructions	
HALT	Halt
SFLG	Set flag
PFLG	Pulse flag

Diagrams and information, courtesy of National Semiconductor from their *PACE Technical Description*.

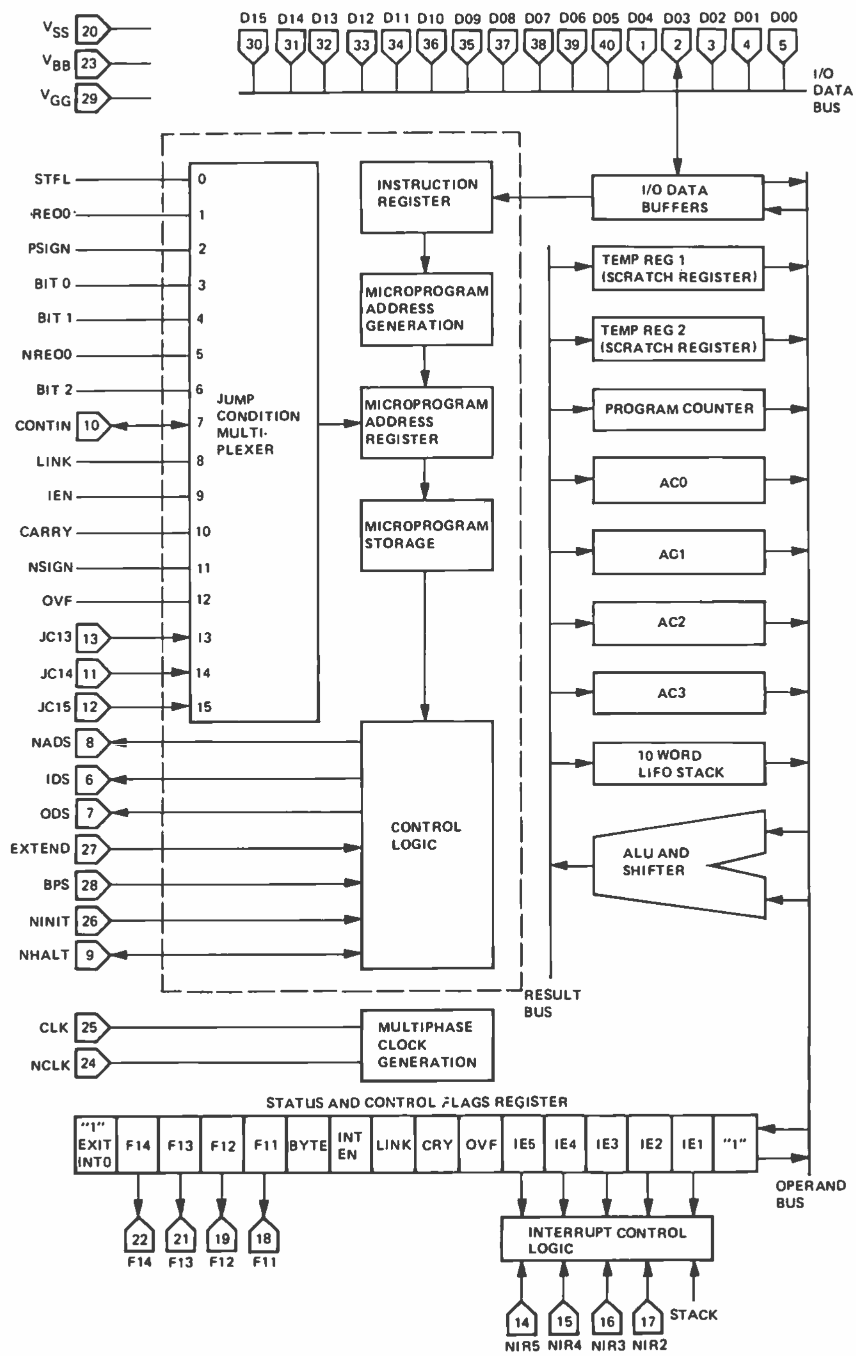
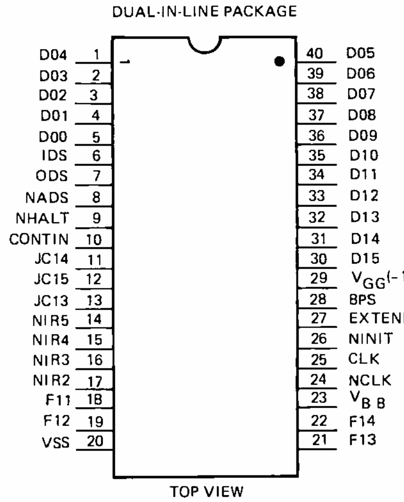


Figure 1: PACE Microprocessor Functional Block Diagram. This diagram shows the major internal sections of the processor, which are reflected in the instruction set. The numbered connections refer to the pins of the processor package, shown in figure 2. This diagram is reproduced courtesy of National Semiconductor from page 2-2 of the PACE Technical Description.

table 2. Conditional branches are implemented using the BOC instruction which allows testing any one of 16 conditions as shown in table 3. Additional testing capabilities are provided by the skip instructions which provide memory and register comparisons without altering data. Memory reference instructions use a flexible memory addressing scheme which provides three floating memory pages and one fixed page of 256 words (16 bit) each. Figure 3 shows the various memory reference instruction formats used to obtain the four different addressing modes.

Figure 2: PACE Microprocessor Pin Assignments. Fitting a 16 bit processor's pinout into a 40 pin package is a problem: If for example separate 16 bit address and data lines are used, 32 pins are occupied and only 8 are left for miscellaneous processor functions and power. But if only one 16 bit bus is used for address and data, with time multiplexing of the usage, then a large number of pins remain available for miscellaneous uses. National Semiconductor chose the latter course of action in the PACE design, leaving a large number of externally available inputs and outputs including four levels of external interrupts, three jump condition flags, and control lines. This diagram is reproduced courtesy of National Semiconductor, from page 2-3 of the PACE Technical Description.



PIN	DEFINITION
CLK, NCLK	True and complemented MOS clock input.
D00 to D15	IO MOS data bus lines.
IDS	Input data strobe, enables external devices to send data to PACE.
ODS	Output data strobe, enables external devices to accept data from PACE.
NADS	Address data strobe, clocks address from PACE into ALE.
EXTEND	Extended data transfer, increases time duration of data IO transfers for slow memory or peripherals without changing clock frequency.
NINIT	Initialize microprocessor functions.
CONTIN	Continue jump condition.
NHALT	Control panel halt.
BPS	Base page select, selects one of two possible base page addressing schemes.
J13 to J15	Jump conditions 13 to 15, user specified branch condition inputs to jump condition multiplexer (see table 3).
F11 to F14	Flags 11 to 14, status and control flag register general purpose control flag outputs.
NIR2-5	Interrupt requests 2 to 5.
Vbb	PACE substrate voltage input, derived from Vgg and Vss by STE.
Vgg	−12 V supply input.
Vss	+5 V supply input.

Table 3: PACE Branch Conditions. The BOC (branch on condition) instruction branches to the effective address if the selected condition is true. This table is reproduced courtesy of National Semiconductor, from page B-14 of the PACE Technical Description.

Condition Code (cc)	Mnemonic	Condition
0000	STFL	Stack Full (contains nine or more words).
0001	REQ0	(AC0) equal to zero (see note 1).
0010	PSIGN	(AC0) has positive sign (see note 2).
0011	BIT0	Bit 0 of AC0 true.
0100	BIT1	Bit 1 of AC0 true.
0101	NREQ0	(AC0) is nonzero (see note 1).
0110	BIT2	Bit 2 of AC0 is true.
0111	CONTIN	CONTIN (continue) Input is true.
1000	LINK	LINK is true.
1001	IEN	IEN is true.
1010	CARRY	CARRY is true.
1011	NSIGN	(AC0) has negative sign (see note 2).
1100	OVF	OVF is true.
1101	JC13	JC13 Input is true
1110	JC14	JC14 Input is true.
1111	JC15	JC15 Input is true.

NOTES: 1. If selected data length is 8 bits, only bits 0 through 7 of AC0 are tested.

2. Bit 7 is sign bit (instead of bit 15) if selected data length is 8 bits.

Applications

There is a complete family of support chips designed for PACE which are intended to interface directly to the microprocessor chip. The System Timing Element (STE) provides the necessary MOS clock signals and V bbo supply voltage for the processor chip as well as an optional TTL clock for the user's system. Figure 4 shows how the STE is connected to the PACE microprocessor for a typical application.

The Bidirectional Transceiver Element (BTE) provides single chip, 8 bit input and output buffering between the PACE MOS input bus lines and TTL devices. Figure 5 shows how three BTEs are utilized along with an STE and PACE chip to assemble a system providing a fully multiplexed address and data bus with a minimum chip configuration. The other two support chips, the Address Latch Element (ALE) and the Interface Latch Element (ILE), may be added for more complex systems as shown in the *PACE Technical Description* available from National Semiconductor as publication number 4200078A. This book also illustrates a priority encoder, a control panel, DMA capabilities, application cards, and various applications and features of the PACE system along with the information contained in this article.

User Group

COMPUTE, a National Semiconductor sponsored user group for microprocessor programmers, users, and technical experts publishes a newsletter called *The Bit-Bucket.* They can be contacted at:

COMPUTE/470
National Semiconductor Corp
2900 Semiconductor Dr
Santa Clara CA 95051
(408) 732-5000 X7183

Personal Computing with PACE

The design of the PACE computer is that of a minicomputer. Its instruction set presents a conventional minicomputer architecture (with a stack however, which was usually unheard of in the days of the mini). Looking at the PACE architecture is reminiscent of looking at a Data General NOVA. For those with a strong background in 16 bit minicomputer programming, a 16 bit processor like the PACE would provide an excellent place to take advantage of your present bag of tricks to save time while getting oriented. While no PACE personal computing kits or finished products are presently available (circa June

bits	15 … 10	9 … 8	7 … 0
	OPERATION CODE	INDEX (XR)	DISPLACEMENT (disp)

Index Field	Addressing Mode	Effective Address (EA)
00	Base Page	EA = disp
01	Program Counter Relative	EA = disp + (PC)
10	AC2 Relative (indexed)	EA = disp + (AC2)
11	AC3 Relative (indexed)	EA = disp + (AC3)

Key: (PC) = Contents of program counter
(AC2) = Contents of AC2
(AC3) = Contents of AC3

Figure 3: Memory Reference Instruction Format. The PACE processor's minicomputer-like instruction set has four addressing modes which are specified by the index field of an instruction. Addressing can be through an 8 bit signed or unsigned base page address selection, an 8 bit signed offset relative to the program counter, or an 8 bit signed offset relative to registers AC2 or AC3. This diagram is reproduced courtesy of National Semiconductor, from pages 2-11 and 2-12, PACE Technical Description.

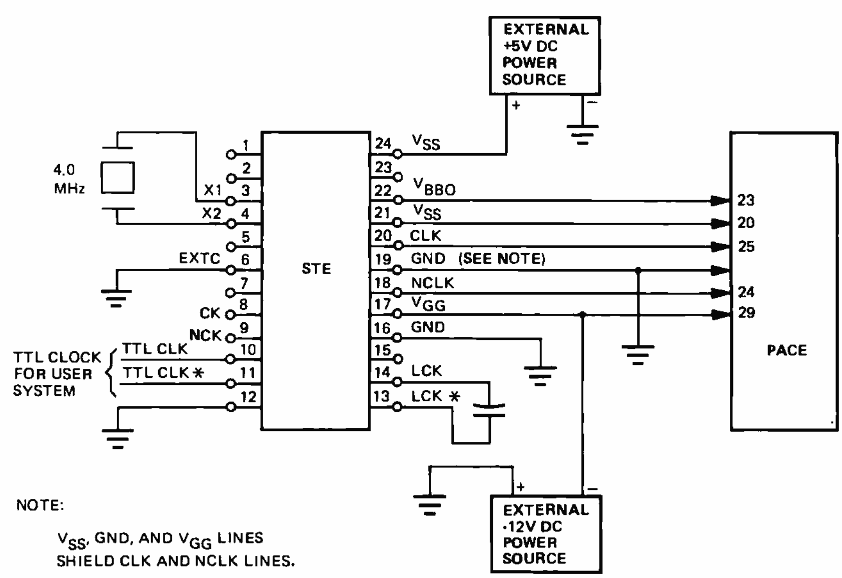


Figure 4: The PACE System Timing Element. In order to make a PACE work without undue engineering inconvenience, National Semiconductor also produces the STE chip which is an omnibus multifunction device including the master timing source, clock generator and driver, and one of the power supplies needed by the chip. The interconnections are shown in this diagram, reproduced courtesy of National Semiconductor from page 2-14 of the PACE Technical Description.

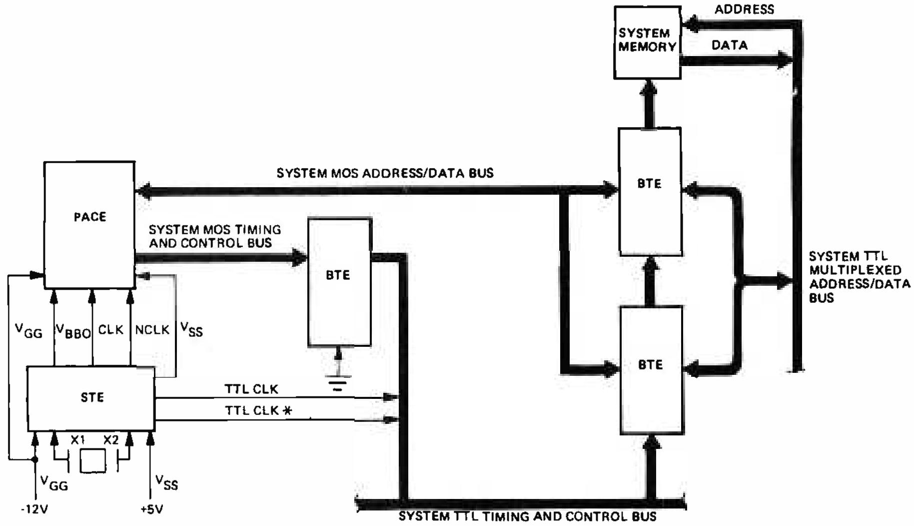


Figure 5: Minimum Chip Configuration. The minimum configuration for a multiplexed addressing and data interface to TTL logic is shown in this block diagram. The blocks labelled BTE are "bidirectional transceiver elements," members of the PACE chip set which are used to sense the MOS level outputs of the processor and drive TTL data bus structures, or in the opposite direction to convert TTL bus levels into PACE compatible input signals. This diagram is reproduced courtesy of National Semiconductor, from page 2-29 of the PACE Technical Description.

1976) for the personal computing market, evaluation kits and products like the Pacer (see below) will provide a convenient way for readers to obtain this 16 bit byte sized microprocessor.■

What's New?

So You Want to Keep PACE?

One way in which to evaluate and use PACE is to employ a system known as Pacer, which is manufactured by Project Support Engineering, 750 N Mary, Sunnyvale CA 94086. This product, which came our way recently for evaluation, is available either in kit, semi-kit or assembled versions, with a retail price in the $1000 range, more or less depending upon specific options chosen. Pacer 1H is totally unassembled, Pacer 2H is completely assembled, tested and burned in, and Pacer 3H is unassembled except for the tested and burned in logic cards. The pictures accompanying this description are of a Pacer 3H version which we recently assembled.

Photo 1: When assembled, Pacer is a neat desk top package with a molded plastic case and calculator style keyboard. The dot matrix alphanumeric display is here shown as it appears when the Pacer is initialized by pressing the INT button.

The basic unit as it comes (see photos 1-3) in its simplest version contains a motherboard with raw power supply, three main logic boards with on board regulators, and a control panel assembly. The three boards in the configuration shown in the photos are:

- The Pacer CPU board with the PACE processor, buffering elements, address decoding for system addresses, and clock logic. This is the board at the right in photo 3.
- Pacer control board which contains the Pacer executive ROM, executive programmable random access memory and initialization logic. This is the center board of the three boards shown in photo 3.
- Pacer memory board which contains slots for 1 K by 16 of 2112 memory chips and 512 by 16 bits of MM5204 programmable read only memory chips. (The unit comes with 256 words by 16 bits of programmable memory and no user ROMs.)

Also included is the Pacer control card, shown at the top of photo 2 with its ribbon cable running down to the motherboard. This control card contains the logic and switches needed to support the 8 digits of alphanumeric display and key board functions.

What Can You Do with Pacer as It Comes?

In the minimal configuration, you receive a powerful 16 bit PACE processor which basically resembles a Data General NOVA *[see Robert Baker's article]*. The executive features of this minimal configuration allow the user to enter and debug machine language programs for PACE, using a set of commands much more flexible and sophisticated than the toggle switches of an earlier era. These built-in checkout and debugging features include:

- Commands to set, modify and examine the hexadecimal contents of:
  - Program counter
  - Accumulators 0, 1, 2 or 3
  - Stack locations 0 through 9
  - Flag register
  - Any memory location selected
  - Any one of several break point addresses labelled 0 to 9
  - Value and mask fields used for search operations
- Extended command functions include:
  - Hexadecimal calculator: add, subtract display result
  - Memory search for value, with optional mask
  - Run, restart, initialize, single step the processor.

For initial checkout of machine language programs, this is more than adequate. The bus structure is completely documented in the manuals, and will prove quite expandable using prototyping boards which are also available from the manufacturer. The Pacer has 8 slots available on its backplane after the minimum three cards are inserted.

The Number One Expansion to Make to Pacer

One additional board came with our Pacer, one which both exhibits its expandability and will prove quite useful to anyone wishing to do software development. This is the optional PACE 2 printed circuit card. This addition to the system contains a parallel Teletype current loop interface, an RS-232 interface, and the PACE 2 read only memories.

The read only memories are the key to the usefulness of PACE 2. This board has a simple assembler (and dissassembler) program and powerful debugging executive, which takes over the system instead of the original read only memory executive of Pacer's minimal configuration. With the addition of PACE 2, the original front panel is essentially ignored with the exception of the restart and initialize buttons. The address space locations of the PACE 2 executive and assembler are contained in an 8 K block of memory which is completely protected by hardware in the standard boards as they arrive. (If desired, it is of course possible to defeat this feature through hardware modifications suggested by the logic diagrams supplied.) The hardware lockout involves write protection and automatic return of hexadecimal FFFF values when referencing the protected locations. In the words of the PACE 2 users guide, "As a result PACE 2 will remain operable even after a massive failure of the user's program . . . . "

The memory editing commands of PACE 2 allow the user to display and set memory contents from the terminal using ASCII, signed decimal, unsigned decimal or hexadecimal conversions. Memory listing commands allow the user to get an assembly language formatted output with symbolic references and absolute hexadecimal, sort of a built-in "disassembler" program. Debugging features include a sophisticated memory snap feature tied to break points, snap points or single step execution. This feature allows selected contents of registers, or memory to be dumped when execution

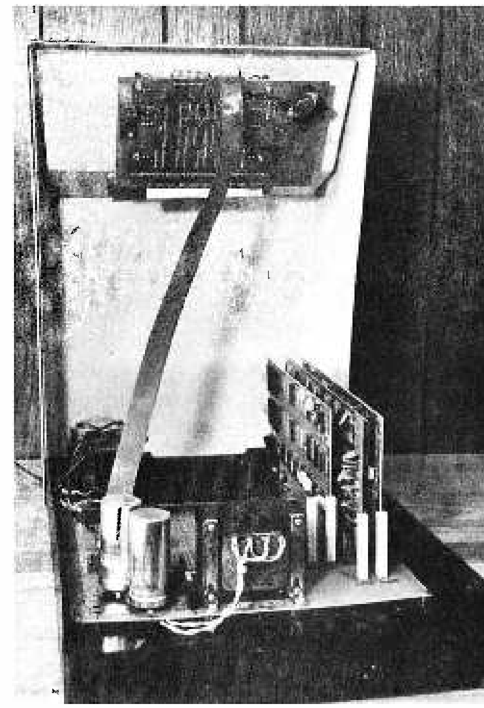


Photo 2: Removing four screws and tilting back the cover of the assembly shows the interior of the machine: a motherboard with raw power supply (unregulated voltages for on board regulators) and positions for eight boards in addition to the three boards of the basic Pacer kits. The control panel board and its ribbon cable connector are shown mounted on the cover near the top of the picture.

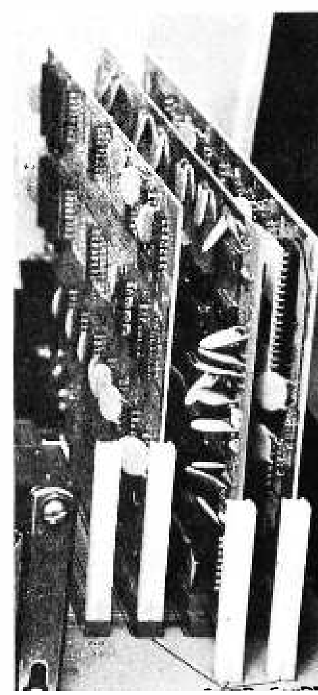


Photo 3: This detailed view shows the three boards of the basic Pacer kit product. The processor itself is the large chip on the rightmost board of the three shown. In the Pacer shown here, additional sockets and card guides which came with the kit are ready for the PACE 2 board and for prototyping boards.

reaches the desired points. As an assembler, the PACE 2 module in its "alter" command mode allows one to insert symbolic assembly language statements to define memory content word by word. The symbol table capacity built into the system when PACE 2 arrives is 26 symbols. By adding memory it is possible to extend the symbol table to a limit of 121 symbols. Each symbol has a 16 bit address value associated with it.

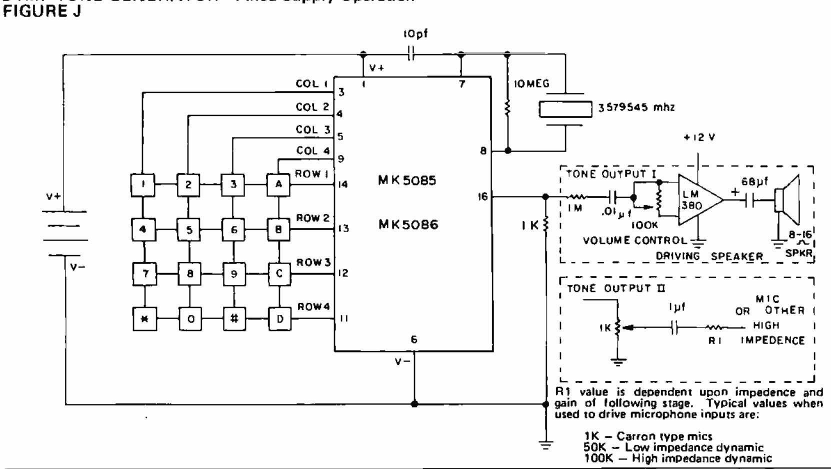
By putting the PACE 2 board together with the Pacer, the result will be an excellent combination of software development tools and expandable hardware which will prove attractive both to the engineer using PACE in systems work, and to an individual who wants a prepackaged 16 bit computer with good systems software support at a moderate price. For further information contact Project Support Engineering.■

The MOSTEK MK5085N/MK5086N Integrated Tone Dialers

This product provides the electronics needed to simply implement remote access via telephone to computer using a miniature hand held terminal which can be used with any telephone. In such a system, the computer end listens to audio from one of a number of Bell approved electronic tape answering service devices commercially available, and standard touch tone signals are sent to the computer after dialing into its phone port. This MOSTEK chip, built into a tiny case using the circuit described in the MOSTEK documentation, and reproduced here, is used to drive the remote telephone through a speaker you build into the remote package. The computer end must have a hardware or software tone decoder and a software command interpreter to carry out your remote instructions; for reliability, such software should be stored in ROM against the possibility of power failures. Eventually, commercially produced "universal signallers" using this technology should be available in calculator style packages, but until then you'll have to make your own. Contact MOSTEK at 1215 W Crosby Rd, Carrollton TX 75006.■

"Figure J" excerpted from MOSTEK's April 1976 data sheet for the Integrated Tone Dialer MK5085N/MK5086N.

DTMF TONE GENERATOR - Fixed Supply Operation
FIGURE J



SCELBAL, an 8008/8080 High Level Language

In our June 1976 issue [*page 82*], we had a product description article on the SCELBAL language, implemented for the 8008 and 8080 processors by Nat Wadsworth and Mark Arnold. Since that time, the language design has been published in detail in SCELBI Computer Consulting's book of documentation, available at $49 postpaid.

The documentation is very complete, and expresses Nat Wadsworth's excellent philosophy of giving all the information possible about the subject at a reasonable price. Quoting from the introduction:

> *. . . . It was known at the start that the program could not be developed to satisfy every potential user. Nobody has a system with that much memory available! Care was taken to provide a good fundamental selection of syntax statements and functions in the language. From that point, backed by the descriptions of the program's organization, general flow charts, and highly commented listings provided in this publication, it is felt that the user will be equipped to add extended capabilities depending on memory available, or willingness to sacrifice described functions.*
>
> *. . . . The extra measure of providing the information so that the user may go further if desired is the fundamental premise behind this publication.*

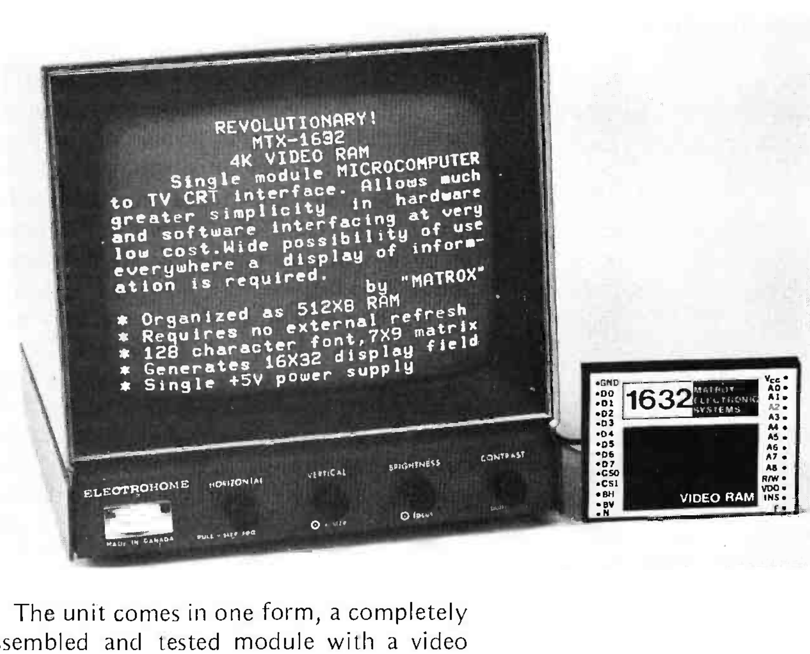
In the words of the sage, "right on!". This 368 page book contains an overall program logic description of the interpreter spread over the first 11 chapters, totalling some 162 pages. Of these pages, 51 alone are spent on chapter 6 which describes the statement interpretation logic of the program with flow charts, symbolic (commented) assembly language code and verbal descriptions. In the chapter 12, the detailed assembly of SCELBAL for an 8008 is given, and in chapter 13 the same information is repeated for an 8080 assembly. The detailed assemblies give absolute code for the program in octal, beginning at origin 01/000 for either version. The total memory required is the same in either version since the code is written for an 8008 and simply reassembled for the 8080 version. The actual program code covered by the assembly in either case is approximately 11.5 K bytes. The listings together occupy 164 pages. Chapter 14 contains operating instructions and chapter 15 contains "Suggestions for Program Tinkerers." The book is concluded with a "SCELBAL Labels Reference List" (ie: the symbol table of the assemblies), and several notes pages for patches. The final sheet of the book is a cardboard sheet with a personal "SCELBAL Registration Card," a change of address card, and a pocket reference card. Purchasers of SCELBAL return the registration card in order to be placed upon the update list for errors and patches (several of which are already present on the notes pages).

SCELBI is to be commended for this example of complete and thorough documentation for a high level language product. It will prove a useful volume for any person interested in a high level language like BASIC which has options for customization and extension. ■

The Visible Character Buffer Memory is Here

A firm called Matrox Electronic Systems, of Montreal, Quebec, recently came out with a unique line of inexpensive video display generators in "black box" module form. One of the products which will be of immediate interest to BYTE readers is the new MTX-1632 video display memory, a unit which interfaces to a three state bidirectional processor bus as if it were 512 bytes of programmable memory, and continuously generates a 16 line by 32 character upper and lower case ASCII display in the form of a standard video signal output. The unit is priced at only $198 in single quantities, and has a memory access time of 650 ns.

The three state bidirectional bus interface has specifications compatible with standard bus drivers such as the National Semiconductor DM8833 and similar chips, so with the addition of address decoding logic it should plug in to most systems with bidirectional buffered data buses. For readers using the Altair, IMSAI, Polymorphics, or Digital Group systems, some interfacing logic will be required to split the bidirectional bus into the "in" and "out" sections required by these systems.



The unit comes in one form, a completely assembled and tested module with a video output drive capability claimed sufficient for 25 standard television monitors to be wired in parallel. Contact Matrox at POB 56, Ahuntsic Stn, Montreal, Quebec CANADA H3L 3N5, (514) 481-6838. ■

Homebrewery vs the Software Priesthood

Mike Wilber
920 Dennis Dr
Palo Alto CA 94303

David Fylstra
PO Box 10051
Stanford CA 94305

The movement of computers into people's homes makes it important for us personal systems users to focus our efforts towards having computers do what we want them to do rather than what someone else has blessed for us.

The movement towards personalized and individualized computing is an important threat to the aura of mystery that has surrounded the computer for its entire history. Until now, computers were understood by only a select few who were revered almost as befitted the status of priesthood. The arts of designing and programming computers have long been regarded as sacred knowledge beyond the reach of the nonspecialist. Indeed, the journeymen of the trade have protected their privileged position by keeping their knowledge to themselves. These high priests and acolytes of the holy alliance of logicians (HAL) have dominated the field so far.

The movement of computers into people's homes makes it important for us personal systems users to focus our efforts toward having computers do what we want them to do rather than what someone else has blessed for us. If personal computing users freely share their hard-won information and even their programs, then this community of users can become quite adept at bending their computers to their own needs and wants.

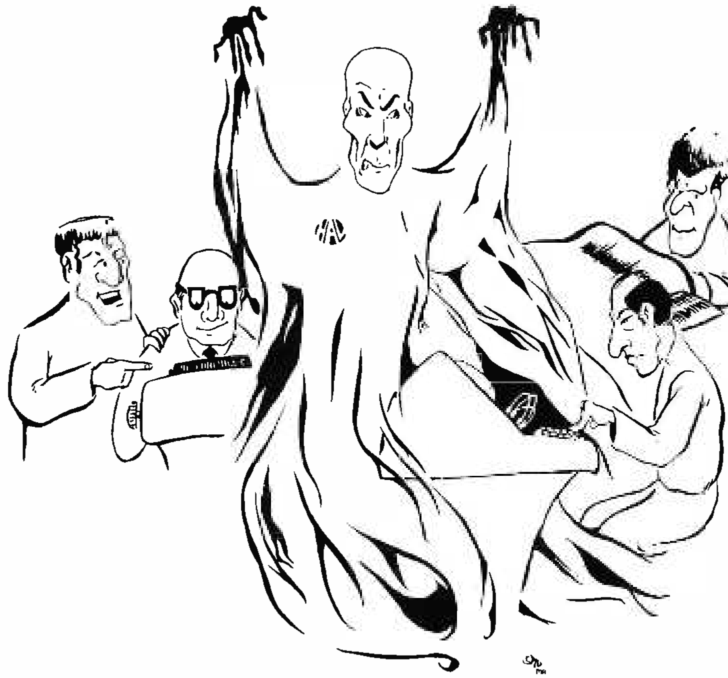
When computers move into people's homes, it would be most unfortunate if they were merely black boxes whose internal workings remained the exclusive province of the priests. It is fine to use them as black boxes as long as they do what you want, but the computer's owner should be able to modify its behavior to suit merely personal preferences. That is, computer literacy should be widespread. Now it is not necessary that everybody be a programmer, but the potential should be there. This understanding must be removed from the private reserve of the select few just because computers will be in the hands of the many.

Freely available software is perhaps the key to the independence of the homebrewer.

There are several indications that personal computing users are likely to be highly motivated to generate freely available software. One can be found in the amateur radio community: Hams have a long tradition of freely sharing their feats of engineering. In fact, it is quite natural to be proud of one's accomplishments, and that pride finds quite natural expression in telling all the details to anybody showing the slightest signs of interest. Already there are some signs that computer hobbyists have similar pride in their software achievements. Dr Dobb's *Journal* contains several examples of people contributing adaptations and extensions to Tiny BASIC. The mode of software development is likely to follow the example of program development among the world of paid programmers. For a while, people will contribute incremental improvements to a program (or a concept). During this first half of the cycle, the program becomes progressively more powerful and progressively messier until it is quite hard to understand or modify. Finally, somebody gives up in disgust and rewrites the program in the marvelously clear way that had been growing increasingly conspicuous by its absence. Then another cycle starts, but its starting place is far more advanced than the previous cycle had reached.

"...these are, and of right ought to be, Free and Independent...."

— John Hancock, et al



Personal computing people stand to be largely independent of the priesthood because they are strikingly sophisticated and because they freely share their ideas. A very good example of both these traits can be found in the nearly spontaneous generation of Tiny BASIC through the medium of the People's Computer Company and Dr Dobb's *Journal of Computer Calisthenics and Orthodontia*. One issue published some rough design notes for a machine independent Tiny BASIC, but that was only the beginning. The next few issues published refinements on the design and later ones included an implementation in an interpretive language and then both octal and annotated source programs realizing the interpreter and the entire system in 3 K of 8080 code. To top it off, the whole project was done by far-flung individuals in less than a year.

While Tiny BASIC is a very striking example of what amateurs can do when they work together, we cannot afford to ignore its extreme dependence on good fortune to bring it to pass. Your own copy of BYTE magazine is another example; it is the result of one man's frustration at making his own computer work and his desire to let others profit by his experience. We've been very lucky to have a few people with high ideals to point the way for us, but we would be ill advised to depend on having these fortunate circumstances continue. The time is ripe for the community of personal computing enthusiasts to start thinking seriously about supplying its own steam to back up the energies put out by a few people with strong motivations to help launch the personal computing movement. It's launched now, and we have to provide the impetus and direction to make sure it develops in a way beneficial to the community at large.

A good example of a means to distribute software which divides the effort fairly and in a way nobody seems to mind is the software exchange of the Homebrew Computer Club in the San Francisco Bay Area. At each meeting (every two weeks) there is a table covered with paper tapes of programs contributed by all and sundry. Anybody is welcome to take any tape at all, subject only to the proviso that each copy taken from one meeting be replaced by at least one

Software exchange is a two way street: He or she who uses an application or system's program from a community library is assumed to be willing and able to provide programs of equivalent value for others to use.

copy at the next meeting. A few cautions, however, must be repeated every few meetings: that people label the tapes they bring back, that they take reasonable steps to ensure the accuracy of their copies and that they only contribute software with the author's consent. Homebrewers have good intentions but still need occasional reminders to keep them from getting careless.

It is most unfortunate that some people give free distribution to software against the author's wishes. In fact it's usually illegal, and anyone caught could face a heavy fine because the crime is new enough that many think prominent examples should be offered to reduce its frequency. Until that lucky day, the main people being deterred are the people who have contributed the software that has immeasurably helped the whole hobbyist movement get started. Even though we have a tremendous potential for generating our own software, we still owe a tremendous debt (of gratitude besides the money) to those who have brought us Altair BASIC and 6800 BASIC, and who may have enough faith in us to bring us APL and some truly groovy text editors. Freely exchanged software should be truly free and untainted by ripoffs or by the appearance of ripoffs.

The extreme ease of software theft could present a real barrier to free interchange of good software because many valuable people could understandably be reluctant to become very deeply involved in a forum where such ripoffs are commonplace. Theft is so much easier with software than with hardware because software has two distinctive properties:

- A buyer cannot evaluate its benefit without extensive testing and use.
- The elementary operation in moving software is not to transport one copy but to generate another copy.

Another danger threatens free distribution of free software, and for some of the same reasons, a scarcity of documentation which is all too common in software. The temptation is strong: Somebody developing it understands it well and is concentrating on getting it to work at all and usually prefers not to be distracted by efforts to make it comprehensible to others. On the other hand, the task of documentation can easily take as much effort as the development itself. However, documentation is crucial to the value of a piece of software. Undocumented software is very hard to use and even harder to modify. We have no good solutions to either of these problems. The only thing to do is to repeatedly urge people to be mindful of the problems and to broaden their perspectives beyond the gains of the short term.

Telecommunications and the Community Information Exchange

Enthusiasts in this field can share software right now by banding together into clubs, but that medium limits sharing to small groups of people who live near one another. With telecommunications, people can share their programs with others living at long distances from one another. One vital ingredient to such remote communication is the ordinary telephone, which works wonders at spanning long distances between people. Telephones can have the same benefit for computers, if they are equipped with modems, which handle the translation between a computer's digital signals and the audio signals the telephone can handle. Then, one person can call another, and they can use that same call to connect their computers; one computer can run a special program to copy data from a cassette or memory to the phone line; and the other can run another special program to copy the information from the phone line onto its own cassette. Of course, the data being copied will likely be some program the two parties wish to share; and presto, you have an instance of software sharing at a long distance.

While that kind of person to person exchange is quite effective, it leaves room for improvement in several respects. It requires very close coordination between the two parties, and it requires one phone call for each interchange of data. However, there is another possible mechanism, which can permit widely scattered users to communicate far more freely and with much looser coordination while improving on the economy of phone line usage. That would be a sort of "Community Information Exchange," a computer that would be continually prepared to automatically answer the telephone and would expect a computer to be placing the call. It would be located where a number of individuals (for example members of a local club) could reach it with a local call; it would provide bulk storage facilities, and it would accept commands in a very concise, well understood format from the computer which had called it. Then one subscriber could leave a program in the bulk storage and invite all other subscribers to that Community Information Exchange to copy it to their own systems at their leisure.

This is a very powerful means of broadcasting software among a local community, but it has implicit in it a means of broadcasting opinions and news too. It requires a means of transmitting plain English text between people just so the people will know which programs they can or might wish to communicate to their computers. Once the individual subscribers can communicate words among themselves, they can communicate much more than news about the latest programs available and how to use them. They can also tell one another about the problems they are having with some program they recently picked up or even about problems they are having with some hardware they recently bought. The computer community can find a great deal of strength in freely sharing that kind of information, in addition to sharing their programs. Free communication of information of all types can greatly enhance the community's resistance to inferior products, and acceptance of superior products.

The Community Information Exchange is not limited to the local communication described thus far. In the dead of night, when telephone traffic is reduced and the transcontinental rates are low, a CIE in one locality can call a CIE in another locality. Then they can send programs and other data back and forth. Of course, they have to know just what should be sent where, and they could be told by their subscribers. The commands they will accept from their subscribers could direct them to copy a file to or from some remote CIE. Nor is it necessary for a CIE to directly call another to

Imagine a Community Information Exchange, complete with telecommunications access ports, mass storage and an accounting algorithm to keep track of operating expenses attributed to each user's activities.

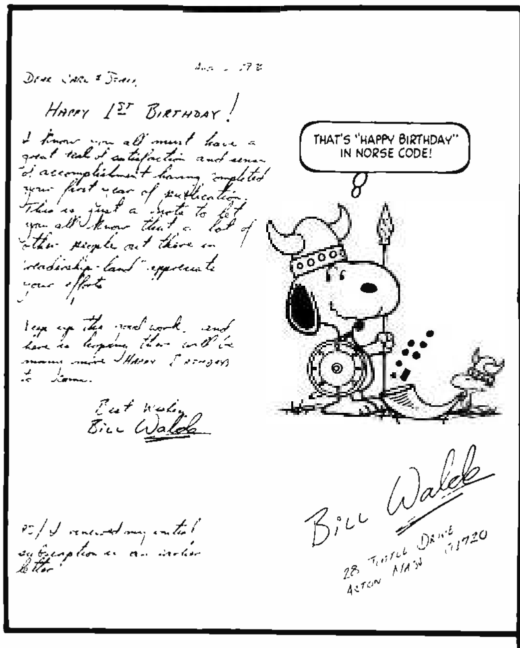
NOTE: The term "Community Information Exchange" was inspired by Michael Rossman.

have access to its file storage. Instead, commands and data could be relayed from one CIE to another until they finally reach their destination. Then, many isolated CIEs would behave like a vast network capable of transmitting software across the country overnight.

We are describing a communication network which can be very effective and which is highly decentralized. In fact, this decentralization is crucial to its effectiveness in promoting free communication between individual computer people. For example, computer manufacturers now organize their customers into user groups in order to provide a forum for communication of ideas and programs among their users. Most communication is channeled through publications controlled by the manufacturer, however; and this all too often results in the encouragement of software and viewpoints which are consistent with those of the manufacturer. On the other hand, it is not necessary to use a centralized forum to encourage people who need it: most of that interaction is on a person to person basis in almost any group.

All well and good, you might say, but just how does the personal computing user compare with the journeyman programmer? Well, the computer amateurs live in a world in which wholesale copying of programs is nearly inevitable. Actually, that represents a healthy trend in two ways: the person using the copy benefits by its availability; and the program's originator benefits by having helped spread a good example of the programmer's art. Of course, such freely exchanged software may well be worthless at best. That does seem a minor penalty, though, for all the advantages that stand to be gained by ready availability. That is one example of the difference between the two kinds of programmers. That is, a functioning program is required of the professional, while it is merely desired by the amateur user who figures he or she will have to patch and customize anyway. Someone paid to write a program has to make it do what the client wants; a personal computing user has only personal preferences to satisfy. Somebody paying for a program has a right to expect that it will be reasonably efficient in order to conserve the money spent on computer time, while your patience is a much stronger constraint for you than the cost of your computer's time. Somebody paying for a program is quite likely to need the documentation oriented toward people unskilled in the technicalities, while fellow enthusiasts won't need to have all the details explained to them. Finally, deadlines are quite firm in the commercial world but of considerably less importance among amateurs *[except for those who fill magazines once a month! . . .* CH*]*.

While the priests who market the old time software religion can help the personal systems user, such users should be mindful of the benefits to be gained from a healthy measure of independence. If you can get your hands on the symbolic form of freely exchanged software, you can revise it to suit your own needs. That's considerably easier than trying to convince somebody else who supplies a high priced package to make your favorite changes, especially since the other person may differ from you in values, priorities and notions of demand. For the traditionalist software source, a request from a single person could easily seem to represent too narrow an interest to motivate a change or patch. If program source listings are distributed freely and nearly every user has the requisite skills to make patches, then the person wanting something changed will probably be the person making the change. Of course, the end result is nearly the best of all possible worlds: Your home computer will do what you want it to do, and it will do it in the way you want it done. It will not do what somebody else decided it was reasonable for you to want it to do and in a way that it was convenient for somebody else to have it done.■



Dear Carl & Jean,

Happy 1st Birthday!

I know you all must have a great deal of satisfaction and sense of accomplishment having completed your first year of publication. This is just a note to let you all know that a lot of other people out there in "readership-land" appreciate your efforts.

Keep up the good work, and here is hoping there will be many more "Happy Birthdays" to come.

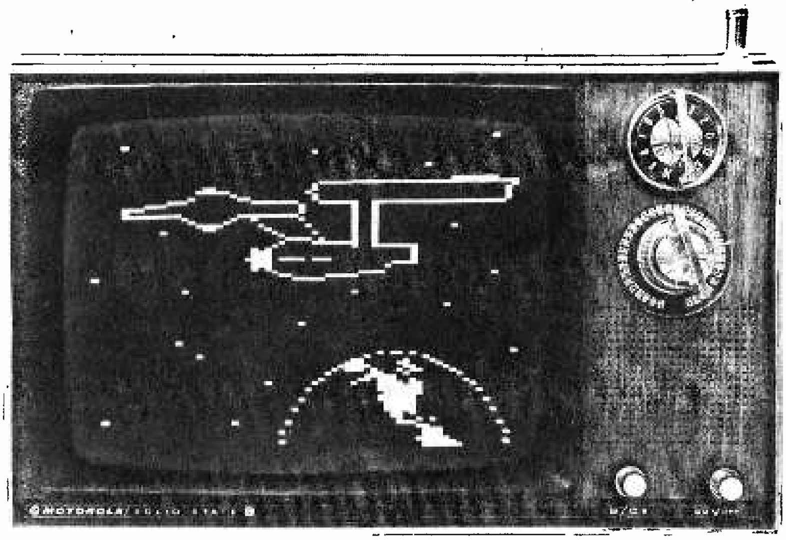
Best wishes,
Bill Walck

PS/ I renewed my initial subscription in an earlier letter!

Bill Walck
28 Tuttle Drive
Acton Mass 01720

BYTE'S BITS

What's Coming Up in BYTE . . . .



In the November BYTE, you'll find an article by John Deres of Southwest Technical Products Corporation, describing in detail the circuit and operation of the new GT-61 graphics display generator product which Southwest manufactures. Included in John's article is a 6800 program which is used to transfer stored images from the processor's memory to the display generator's memory through a parallel data port. One example given in John's article is the data needed to produce this display on the screen. Also scheduled for November are an article on APL by Mark Arnold, and an article on how to homebrew a 256 by 256 point array display interface written by Thomas R Buschbach.■

Networking, Anyone?

Walter Banks of the Computer Communications Network Group, University of Waterloo, Waterloo, Ontario, CANADA N2L 3G1, is interested in talking to radio amateurs and computer people interested in organizing and promoting computer telecommunications networks for amateur experimental use. The definition of a communications software discipline modelled after existing commercial and academic digital networks would be one goal of such activity. Such network activity could be supported "for free" on the radio bands using the OSCAR synchronous satellite when it gets launched, or using the dial-up capabilities of the phone network at the usual Bell rates for long distance calls. Walter can be reached by phone at (519) 885-1211, extension 2847.■

Classified Ads for Individuals and Clubs

Readers who have equipment, software or other items to buy, sell or swap should send in a clearly typed notice to that effect. To be considered for publication, an advertisement should be clearly non-commercial, typed double spaced on plain white paper, and include complete name and address information. These notices are free of charge and will be printed one time only on a space available basis. Insertions should be limited to 100 words or less. Notices can be accepted from individuals or bona fide computer users clubs only. We can engage in no correspondence on these and your confirmation of placement is appearance in an issue of BYTE.■

HELP! I need schematic & information on 3501A Asciscope S/N 14 by LEAR-SIEGLER. Will give almost anything to borrow, reproduce, or even glimpse. Emile Alline, 1119 Penn, Slidell LA 70458.

FOR SALE: KSR-33s, I have a limited number of KSR-33s on stands. We have checked them out on an Altair 8800 and they work like champs. $495 each while they last. Boyd Martin, 5130 Melvin Av, Tarzana CA 91356 eves and weekends, (213) 345-0903.

FOR SALE: Digital tape cassette transports, MFE model 250, outdated but new. Andruss Peskin Corp, POB 268, Natick MA 01760, (617) 653-3919.

FOR SALE: Disk controller, asking $800 or best offer, worth better than $2,600. IBM 2841 disk controller model 1 dual channel, controls up to 8 drives, power supplies, 4 K RAM, all cables and manuals available, maintenance available from IBM. Call Pete Arnett at (305) 671-5631 after 5 PM EST or write 7739 Liverpool, Orlando FL 32807.

FOR SALE: TVT-II (SWTP CT-1024), complete with SWTP power supply, manual cursor control board, parallel interface board, all cables and documentation. Includes wiring changes to make 64 characters per line. All ICs in sockets. Guaranteed working cond, $175. Also have MITS expander card (no sockets or card guides) $15. W W Crider, 4011 Oak Hill Dr, College Park GA 30337, (404) 767-6402.

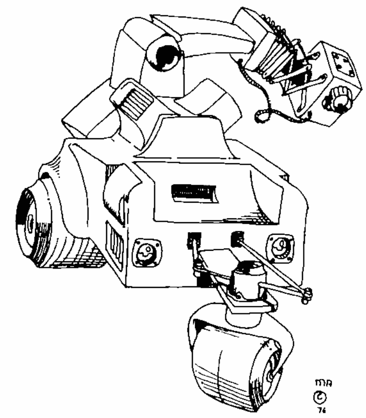
SWAP: Have first 8 issues of BYTE. Want ham radio or electronics magazines. FREE: Schematic & service data for almost any radio receiver or TV set. Make and model number a must; approx age, # of tubes, description helpful. Legal size SASE appreciated. Donald Erickson, 6059-K Essex St, Riverside CA 92504, (714) 687-5910.

FOR SALE: FLEXOWRITER Model SFD. Excellent condition. Has 8 level reader and punch, $300 or trade for TVT-I or TVT-II. Call or write; William Dawson, 316 21st St, Apt 4, VA Beach VA 23451, (804) 422-5921.

WANTED TO BUY: Video terminal (such as Sanders model 720 but any type would do) at a very reasonable cost. If desired, I would swap for my Busicom digital desktop calculator with 14 digit readout and memory. This is a very early digital calculator but is in good operating condition. Either way, tell me what you have. Rich Nicewonger, 24 Rosewood Rd, Edison NJ 08817.

FOR SALE: Disk drive, IBM 2311 type, 7.25 million bytes. Almost new, manuals and documentation available. Asking $750. P Arnett, 1730 S Bumby, Orlando FL 32806.

CLEANING HOUSE—If you want a list of items for sale, send SASE to Bob Baker, 15 Windsor Dr, Atco NJ 08004.



FOR SALE: Three 4 K Altair RAM boards @ $180. Assembled, tested, excellent cond; memory ICs in sockets, less 100 pin edge connector, postpaid. One 1 K static RAM board, also Altair. As above $100. John Martin, 808 Day St, Fairmont MN 56031.

FOR SALE: Paper Tape Splices – prepunched for any kind of 8 channel tapes, $3.50 per 100, postpaid. H Corbin, 11704 Ibsen Dr, Rockville MD 20852.

WANTED: Automatic card reader for demonstration project in the New York City Public School System. Please call: (212) 852-2957 or write: Alex Aderer, 166 Bergen St, Brooklyn NY 11217.

FOR SALE: Altair 8800 system w/ 8 K memory, 88SIO interface, 8 K BASIC complete and ready to run. $950. Also SWTP CT-1024 terminal complete with case $300 and Sanyo Monitor $150. Entire system with manuals guaranteed $1,400. D Smith, (714) 993-9939, Fullerton CA.

WANTED: MOS KIM-1 or OSI series 400 system, also TI SR-52. Joe Torzewski 51625 Chestnut Rd, Granger IN 46530.

WANTED: Information or documentation for SYNER Data total term. If you have same or know how to repair, please contact: Paul Massod, 338 N Warren Av, Brockton MA 02401, (617) 587-9035.

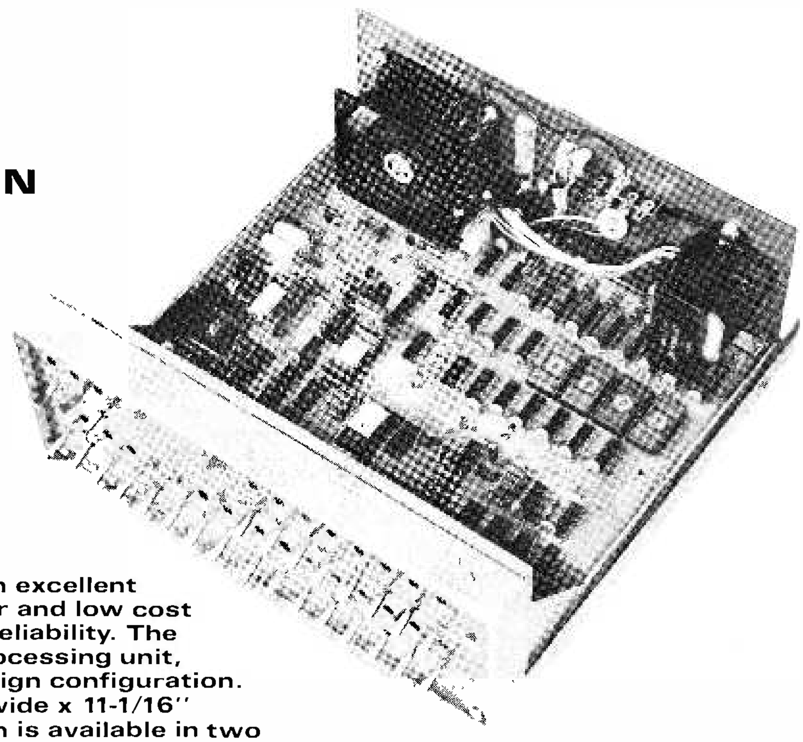
FOR SALE: MARK-8 CPU circuit board (assembled). Needs 8008 chip, some memory, address and IO latches to get it going. MARK-8 and Intel's 8008 manuals included, $20. Also Signetics Data Manual (1976) 1200 pages of TTL, ECL, linear, interface ICs, MOS memories, and their latest microcomps, $8. Frank Canova Jr, 725 Myrtle Av, Green Cv Spgs FL 32043, Phone (904) 284-3408.

WANTED: Software for Altair (BASIC) to keep track of club members, print zip code sorted mailing labels, and "Membership Directory." Write for details. Jack Hardman, 140 Forest Av, Glen Ridge NJ 07028, (201) 429-8880.

FOR SALE: DUMP THE MEMORY of your 6502 system with software system which generates BYTE Standard signal for recording onto cassette. Documented software listing for only $3. Timing variable to match your clock speed. Ideal for putting in PROM, with space for other programs. Dumps one page or many pages in one run. Don Rindsberg, 5958 S Shenandoah Rd, Mobile AL 36608.

altair[T.M.] 680b

TECHNICAL INFORMATION



The ALTAIR 680b microcomputer is an excellent compromise between computer power and low cost structure, without sacrificing design reliability. The system is based on the 6800 microprocessing unit, which adapts nicely to a minimum design configuration. The ALTAIR 680b measures 11-1/16" wide x 11-1/16" deep x 4-11/16" high. The basic system is available in two configurations, depending on the intended application.

Almost all of the 680b circuitry is contained on a single large printed circuit board, including memory and a built-in I/O port. The full front panel model contains all of the controls necessary to program and operate the computer and includes an additional printed circuit board, which provides all of the logic circuitry necessary to reset, halt or start the processor. Also located on this board are switches and associated LED indicator lights for each of the sixteen address lines and eight data lines. The front panel circuit board mounts directly to the main printed circuit board via a 100-contact edge connector. The power switch is located on the back panel of the unit for safety purposes. A "turn-key" front panel model, which eliminates all control except restarting the processor, is also available.

The basic ALTAIR 680b computer can be subdivided into five functional sections. These are the MPU and clock, the memory, an I/O port, control and indication, and the power supply. The first three of these sections, along with the power supply regulation components, are located on the main printed circuit board.

At the heart of the 680b system is the 6800 Microprocessing Unit, which is largely responsible for the overall simplicity of the 680b design. The 6800 MPU contains three 16-bit registers and three 8-bit registers. The program counter is a two byte register which keeps track of the current address of the program. The stack pointer is also a two byte register which keeps track of the current address of the program and contains the next address in an external, variable length push-down/pop-up stack. The index register is a two byte register used to store data or a memory address for indexed addressing operations. There are two single byte accumulators used for holding operands and results from the arithmetic logic unit (ALU). The 8-bit condition code register indicates the results of an ALU operation. In this register there are two unused bits, kept at a logic one. The remaining six bits are used to indicate the status of the following: carry; half carry; overflow; zero; negative; interrupt.

The 6800 has seven different addressing modes, with the particular mode being a function of both the type of instruction and the actual coding within the instruction. The seven modes include the following: Accumulator Addressing—one byte instructions, specifying either of the two accumulators; Immediate Addressing—two or three byte instructions, with the MPU addressing the location given in the 2nd or 2nd and 3rd bytes when the immediate instruction is fetched; Direct Addressing—two byte instructions which allow the user to directly address the lowest 256 bytes of memory in the machine; Extended Addressing—three byte instructions, the second two bytes referring to an absolute address in memory for the operation; Indexed Addressing—two byte instructions, the second byte being added to the 16-bit index register to give the address of the operand; Implied Addressing—one byte instructions and the instruction itself gives the address; Relative Addressing—two byte instructions where the second byte is added to the lower 8 bits, allowing the user to address memory + 129 to -125 bytes from the location of the present instruction.

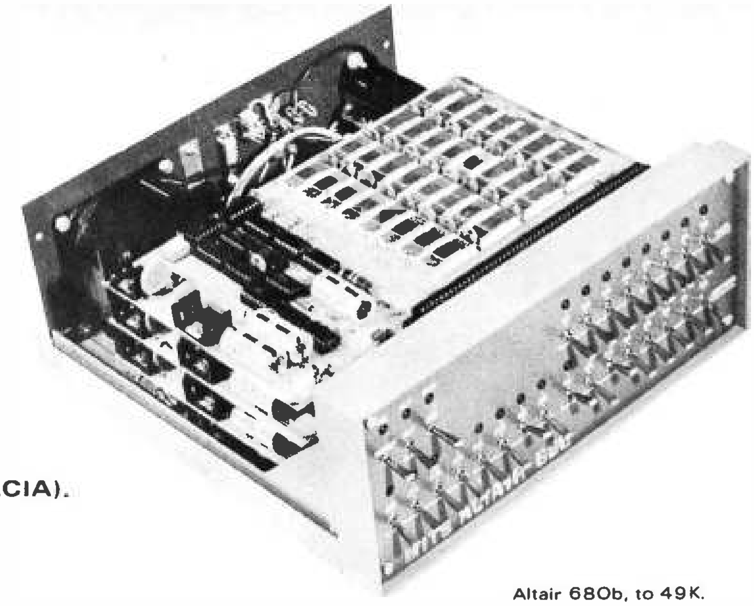
There are several timing and control signals required to operate the MPU. Two clock inputs are required, phase 1 and phase 2. These must be nonoverlapping and run at the Vcc voltage level. In the 680b the clock is a 2-MHz crystal controlled oscillator with logic to provide a 500-KHz two phase clock.

Sixteen active high address outputs are used to specify the sections of memory or I/O to be used. These can drive up to one standard TTL load and 130 pf. There are also eight bi-directional data lines with the same drive capability as the address lines.

NEW MEMORY FEATURES

MITS is pleased to announce the development of a 16K static memory card for the Altair 680b. With an access time of 215 nanoseconds and low power consumption of 5 watts, we feel that this is an excellent addition to the Altair 680b.

The 680b cabinet has room for up to three 16K static memory cards, thereby increasing the memory of the Altair 680b to 49K.



Altair 680b, to 49K.

SPECIAL FEATURES

PROM monitor.

1702A PROM monitor chip programmed so that you can immediately load and run paper tape object programs such as the text editor and assembler (see below).

Asynchronous Communication Interface Adapter (ACIA).

Allows the machine to transmit and receive a character at a time rather than one bit. Minimizes software needed for I/O routines. Contains crystal clock for baud rate synchronization. User-selectable for RS232, Baudot, TTY, 20ma current loop. Baud rates of 50, 75, 110, 134.5, 150, 200, 300, 600, 1200, 1800, 2400, 4800, and 9600.

Two Pass Resident Assembler and Text Editor

A two pass resident assembler and text editor will be available for assembly language programming. This software is compatible with Motorola's format for assembly language programs, text and object files. 8K bytes of memory are required to run this package. The assembler produces a full assembly listing on the second pass, including the hex codes for the location counter and the instruction mnemonics. A symbol table listing is also produced. The text editor has full capabilities for text editing, including line insertion, printing, deletion and modification; as well as commands for changing one string of characters to another and for searching the text buffers for a particular character string.

Basic Interpreter

A BASIC interpreter has been developed which will be comparable to the 8800 8K BASIC interpreter.

Buffered Data Lines

All data lines are buffered to provide fanout capability of over 20 standard TTL loads.

Altair 680b
Specifications

No. of Boards	Up to 3 additional
Microprocessor	
Model	6800
Technology	NMOS
Data Word Size, Bits	8
Instruction Word Size, Bits	8
Clock Frequency,	500KHz
Add Time, Register to Register, Microsec. Per Data Word	2
Number of Instructions	72
Input/Output Control	
I/O Word Size, Bits	8
Number of I/O channels	256 Memory Address Locations Designated
Interrupt Capability	Std.
Type of Interrupt System	Maskable (Interrupt Request) and Non-maskable Interrupt
Software	
Resident Assembler and Editor	Yes
Higher-level language	BASIC
Monitor	Resident System Monitor on PROM
Complete Software Library Separately Priced	Yes

The Altair 680b is also available in this Turnkey Model which has a power indicator light and controls for RESET and RUN/HALT on the front panel. The system PROM monitor, when used in conjunction with a terminal, eliminates the necessity for toggling front panel switches to load bootstraps or to examine and change memory contents.

mits

2450 Alamo S.E./Albuquerque, New Mexico 87106

Programming Quickies:

The Thompson Lister

Noel J Thompson, of the Hawaii Institute of Geophysics, 2525 Correa Rd, Honolulu HW 96822, submits the "Thompson Lister" program for the 6800 using Motorola's MIKBUG program. This program, shown listed at location 0000 in a symbolic assembly language format, is designed to list 6800 programs as an address, an operation code and an optional one or two byte field depending upon the length of the instruction. The sample Teletype output shown at the bottom of the listing was supplied by Noel as part of his listing of the entire program. The program figures out completely whether the first byte of an instruction calls for one, two or three bytes. Quoting from Noel's letter accompanying the program,

"Such a program has two important benefits. First, it provides an adequate listing for your documentation of programs you are working on. You can use it over and over while you are developing a routine,

Second it detects the most common mistake that I find myself making in multi-length-instruction machines — the mistake of forgetting to put in the right number of bytes. If this lister comes up with funny things, probably you left out a byte."

The program is easily relocated by changing constants in the instructions at locations 0000, 0009 and 005D. The data memory used by THMPLSTR is located in the MIKBUG programmable memory region between locations A000 and A07F. Also, the program takes advantage of subroutines PDATA1, BADDR, OUT4HS, OUT2HS and OUTS which are found in Motorola's MIKBUG program.

Addr	Hex Code	Label	Op	Operand	Commentary
0000	CE 00 67	THMPLSTR	LDX	#ANATSGN	Print an '@' sign using
0003	BD E0 7E		JSR	PDATA1	MIKBUG subroutine PDATA1;
0006	BD E0 47		JSR	BADDR	XHI := (address input) [using MIKBUG];
0009	CE 00 64	NEXTBYTE	LDX	#CRLF	Print carriage return and line feed
000C	BD E0 7E		JSR	PDATA1	MIKBUG subroutine PDATA1;
000F	CE A0 0C		LDX	#XHI	X := XHI [XHI as defined by MIKBUG];
0012	BD E0 C8		JSR	OUT4HS	Print current address as 4 hex characters;
0015	FE A0 0C		LDX	XHI	X := XHI [fetch current address pointer];
0018	A6 00		LDAA	0,X	A := @X [fetch current data at X];
001A	B7 A0 0B		STAA	TEMP	TEMP := A;
001D	BD E0 CA		JSR	OUT2HS	Print current data then space;
0020	FF A0 0C		STX	XHI	XHI := X [after OUT2HS has incremented];
0023	BD E0 CC		JSR	OUTS	Print an extra space;
0026	5F		CLRB		B := 0;
0027	B6 A0 0B		LDAA	TEMP	A := TEMP;
002A	81 8C		CMPA	#8C	is op code CPX ?
002C	27 18		BEQ	THREE	if so then go to 3 byte length case;
002E	81 8E		CMPA	#8E	is op code LDS?
0030	27 14		BEQ	THREE	if so then go to 3 byte length case;
0032	81 CE		CMPA	#CE	is op code LDX?
0034	27 10		BEQ	THREE	if so then go to 3 byte length case;
0036	84 F0		ANDA	#F0	A := A & #F0 [mask off low order nybble];
0038	81 20		CMPA	#20	is op code a branch?
003A	27 0B		BEQ	TWO	if so then go to 2 byte length case;
003C	81 60		CMPA	#60	is op code less than #60?
003E	25 08		BCS		if so then go to 1 byte length case;
0040	84 30		ANDA	#30	A := A & #30 [mask to bits 5 and 4 left];
0042	81 30		CMPA	#30	are bits 5 and 4 equal to '1'?
0044	26 01		BNE	TWO	if not then length TWO is indicated;
0046	5C	THREE	INCB		B := B + 1 [here for two increments];
0047	5C	TWO	INCB		B := B + 1 [here for one increment];
0048	F7 A0 0A	ONE	STAB	BOX	BOX := B;
004B	27 10		BEQ	NEXTINST	if zero in BOX then reiterate;
004D	7A A0 0A		DEC	BOX	BOX := BOX − 1;
0050	27 05		BEQ	ONLYONE	if one in BOX then print only one byte;
0052	BD E0 C8		JSR	OUT4HS	print two bytes of 3 byte instruction;
0055	20 03		BRA	SAVEADDR	go save address for next round;
0057	BD E0 CA	ONLYONE	JSR	OUT2HS	print one byte left in 2 byte instruction;
005A	FF A0 0C	SAVEADDR	STX	XHI	XHI := X [save pointer to next address];
005D	7E 00 09		JMP	NEXTBYTE	reiterate program forever till interrupt;

Constants required:

Addr	Hex Code	Label	Description
0064	0D 0A 04	CRLF	Carriage return, line feed, stop code in PDATA1 format;
0067	40 04	ANATSGN	Initialization prompting message in PDATA1 format;

Sample listing on Teletype:

```
0018 A6 00
001A B7 A00B
001D BD E0CA
0020 FF A00C
0023 BD E0CC
0026 5F
0027 B6 A00B
002A 81 8C
```

BOOK REVIEWS

Computer Resource Book – Algebra *by Thomas A Dwyer and Margot Critchfield, Houghton Mifflin, Boston, 1975; 8-1/4 x 11, 178 pages. Paperback $4.20.*

This book contains a collection of computer explorations of algebra that were designed to be carried out in any algebra course where a computer is available. Since the computer programs are written in BASIC, the book begins with a 25 page "whirlwind tour" of computer programming in BASIC. Following this introduction to BASIC are 10 sections on things to do with a computer while learning algebra. In evaluating this excellent resource book, one can conclude that the authors have taught computer programming and high school algebra, and have taught both subjects well. Dwyer and Critchfield have written a book which draws the reader into becoming an interested and active participant in learning algebra. They were able to put themselves in the reader's place by asking (and answering) some of the questions about algebra and computer programming which puzzle students. The book was designed to be read by high school students as well as teachers, and I suspect that many students will want to take it home with them after school.

As the authors point out in the introduction, this resource book is meant to be used in an algebra course as a supplement to a regular algebra textbook. Access to a time sharing system or an on site small computer is necessary when using the book. Teachers and students may want to integrate these computer-related algebra topics in their regular algebra courses or select programs from the book to use as course projects. Many of the BASIC programming skills needed in using the book are presented in the prologue and the remainder are interspersed throughout the algebra topics.

The "Introduction to BASIC" section begins by illustrating the differences among system commands, BASIC statements, computer programs, user input, and computer output. The authors put the computer-programming novice at ease by taking a realistic, relaxed (and at times humorous) approach to using computers. For example, the reader is given numerous reminders at the beginning of the book to press the carriage return key after typing a line and is told "don't be afraid to make mistakes -- you won't hurt anything." To avoid misleading the beginner, each BASIC statement is introduced and explained in the context of a complete, executable program rather than as a non-executable program segment. Sufficient margin notes are used to clarify lines and sections of program listings and sample executions are included with most of the program listings.

The 10 algebra topics included in this book are the language of algebra, operations with real numbers, linear equations in one variable, inequalities, open sentences in two variables, systems of linear equations, quadratic equations and functions, rational expressions and polynomial equations, polynomial functions and complex numbers, and computer-generated animation. Each algebra topic begins with a checklist of necessary computer programming skills and a short quiz on prerequisite knowledge of algebra. Next, several coaching and practice computer programs are listed and explained, followed by a number of application pro-

Now, you can buy an Altair 8800 or Altair 680 computer kit right off the shelf. Most all Altair options, software and manuals are also available. The MITS Dealer List below is just the beginning:

off the shelf

RETAIL COMPUTER STORE, INC.
410 N.E. 72nd
Seattle, WA 98115
(206) 524-4101

COMPUTER KITS
1044 University Ave.
Berkeley, CA 94710
(415) 845-5300

THE COMPUTER STORE
(Arrowhead Computer Co.)
820 Broadway
Santa Monica, CA 90401
(213) 451-0713

THE COMPUTER SHACK
3120 San Mateo NE
Albuquerque, NM 87110
(505) 883-8282

GATEWAY ELECTRONICS
2839 W. 44th Ave.
Denver, CO 80211
(303) 458-5444

GATEWAY ELECTRONICS
8123-25 Page Blvd.
St. Louis, MO 63130
(314) 427-6116

BYTE'TRONICS
Suite 103 – 1600 Hayes St.
Nashville, TN 37203
(615) 329-1979

CHICAGO COMPUTER STORE
517 Talcott Rd.
Park Ridge, IL 60068
(312) 823-2388

MARSH DATA SYSTEMS
5405-B Southern Comfort Blvd.
Tampa, FL 33614
(813) 886-9890

MICROSYSTEMS
6605A Backlick Rd.
Springfield, VA 22150
(Washington DC area)
(703) 569-1110

THE COMPUTER SYSTEMCENTER
3330 Piedmont Road
Atlanta, GA 30305
(404) 231-1691

THE COMPUTER STORE, INC.
120 Cambridge St.
Burlington, MA 01803
(617) 272-8700

THE COMPUTER STORE OF ANN ARBOR
310 East Washington St.
Ann Arbor, MI 48104
(313) 995-7616

COMPUTER PRODUCTS UNLIMITED
4216 West 12th St.
Little Rock, AR 72204
(501) 666-2839

the COMPUTER STORE, INC.
63 South Main St.
Windsor Locks, CT 06096
(203) 871-1783

the COMPUTER STORE of NEW YORK
55 West 39th St.
New York, NY 10018

THE COMPUTER ROOM
3938 Beau D'Rue Drive
Eagan, MN 55122
(612) 452-2567

NOTE: Altair is a trademark of MITS, Inc.

MITS, Inc. 2450 Alamo S.E. Albuquerque, N.M. 87106

grams and simulations. Most of the 125 programs which are listed are interesting enough to elicit an "I want to try it out" response on the part of students, and many of the programs are unavailable in other computer supplemented sources. Activities such as modifying the given programs to make them more efficient, revising programs to make them do different things, and creating new programs are suggested at the end of each algebra topic.

If you want to liven up your algebra classes with computer activities such as building an ultramatic root finder, making and breaking secret codes, writing robot coaching programs, programming Honest Hal the simulated used-car salesman, and many others, this fascinating book is for you and your students. Even if you only want to see how computers can be used to enhance learning mathematics or find some new ideas for mathematical applications of computers, the *Computer Resource Book — Algebra* is still appropriate.

My only complaint about this book is that it ends too soon; and where are the computer resource books for geometry, trigonometry, probability, and statistics?

Frederick H Bell
Associate Professor
School of Education
University of Pittsburgh
Pittsburgh PA 15260

The Art of Computer Programming *by Donald E Knuth.* Volume I: Fundamental Algorithms. *Addison-Wesley, Reading MA, 1975, $20.95.*

There is some doubt in my mind as to how to classify this series. It is undeniable that we have here a standard, perhaps even classic, reference work, which sits handy on many a programmer's desk. But it clearly is also a text, useful for self instruction and used in several university programming courses. The exercises, while very instructive, could also fall in the realm of recreational mathematics (another field the author has contributed to). It even makes entertaining reading at times (no mean feat for a text and reference work). I suppose the best description, however, would be that this is a cookbook of computer algorithms.

More specifically, it deals with "nonnumerical analysis." Numbers occur only "coincidentally," with more stress given to the decision making capabilities of the computer. The numeric computing which is involved consists mainly of addition and subtraction (these qualities are among the stronger points of inexpensive computers, bless 'em). Of course, the techniques presented are also of considerable use in numerical computing, as there is always a non-numerical background in programming.

By numerical computing, I refer to the solving of equations, finding of roots, and other number oriented calculations (the "traditional stuff"). While that aspect of the field is little covered by this series, that doesn't mean there is no mathematics involved. Indeed, Mr Knuth is himself a professional mathematician. This need not scare away those who are not, however, as the author kindly presents theory and algorithm with the minimum math needed to understand and use them, keeping more difficult analysis towards the end of each section, where only those with more interest need tread. (Some knowledge of basic math notation would be helpful, though.) Also optional are the exercises which conclude each section (the answers *are* included, by the way).

If this series is not just for the mathematician, who then? Everyone? Not quite. It is aimed at the person with more than a casual interest in computers, the programmer concerned not only with how to use some one else's subroutines, but with how to use the algorithms behind the subroutines to improve his own creations. The reader's relationship with the computer should be beyond the stage of introductions. For example, Knuth suggests that the reader should have written and debugged at least four programs on at least one machine. (The programmer who has written scores of programs for several machines may get at least as much value from the series. As I say, it is a reference, too.) It's for you if you're getting the hang of your programming units (be they hex machine codes or FORTRAN statements), and you want to start putting them together to *do* something.

To connect the units with the application in mind, you need algorithms. (Roughly, an algorithm is a step by step procedure for doing something.) That's what *The Art of Computer Programming* is all about. The algorithms are chiefly presented in two forms: in a general English-language outline form and, when appropriate, in the assembly language for a hypothetical computer of the author's own concoction, MIX (the "world's first polyunsaturated computer"). Knuth has several good reasons for getting us involved with this mongrel machine (read the book), but for our purposes there is the additional advantage of easier translation of algorithms, ie: from his assembly language to our assembly language. While many algorithms translate well to higher level languages like

BASIC, some things can be better done closer to the assembly language level, eg: systems programs.

While it's not necessary to the use of the series, I plan to have a MIX simulator running on the MOS Technology 6502, and I expect there will be simulators for other micros as well. This will allow very quick checkout and modification of the MIX algorithms given in *The Art of Computer Programming* before further translation, and make possible the exchange of programs with other hobbyists.

In particular, let's consider *Fundamental Algorithms*, which is the first volume of seven in *The Art of Computer Programming* (of which three have been published so far). Chapters 1 and 2 are contained therein. Chapter 1 features a concise tutorial of much of the math that is particularly useful to programming. You may skim this section, but don't be surprised if over the years you keep referring back to it. Then follows a complete presentation of MIX and the MIX language, including the description of a MIX simulator. In fact a well commented listing of the MIX simulator is given and explained (written in MIX, of course). This presentation will greatly help anyone interested in writing their own simulator (which is a very useful exercise).

Chapter 2 is titled "Information Structures," and begins the meat of the series. Major topics covered include stacks, arrays, linked lists, dynamic storage allocation, and trees. (These trees have roots and leaves, but [generally] aren't green and usually grow upside down.) Whether you know all or none of the above terms, this chapter should leave you with a good understanding of what they are and how to use them. No "TKIS" *[page 42, January BYTE]* is complete without some of these structures. Nor is any programmer.

Zhahai Stewart
PO Box 1637
Boulder CO 80302 ■

Humanizing Computer Systems *by Keith R London, McGraw Hill, New York, $15.*

What is the people impact when computers are placed in a working environment? What are associated problems of developing and implementing systems to meet the requirements of an organization and yet satisfy the people — the employees?

This new book, "The people side of systems," is a valuable work which puts aside the technical aspects of computing and looks at the implications for people when a computer comes into their every working day association.

The author is Keith R London who has produced considerable output related to management of computer systems. In this new book, he directs attention to how computers can cause disruptions in employee relationships.

Keith acknowledges the need and use of computers, but he also sees a need to reorient project leaders and systems analysts away from a pure hardware/software aspect to giving some attention to the impact the system has on those it comes into contact with: workers in a company or group. The thrust is to show how to integrate a computer with people.

This book is not directly of import to the average Byter other than giving some insight to better relationship with the rest of the family who may think, with some degree of justification, that the excessive interest hobbyists show towards do it yourself computers has placed them just this side of the funny farm. The book is a must for those who have to make computers work in a large "people" group.

Art Brothers
555 Avenue G
Boulder City NV 89005 ■

Interested in Notes about Electronic Music?

Electronotes is the name of an excellent newsletter which will be of interest to anyone who is into electronic music experimentation. Published by Bernie A Hutchins, it is subtitled "Newsletter of the Musical Engineering Group." The address is 203 Snyder Hill Rd, Ithaca NY 14850 . . . write for information on signing up for a subscription. The price of a typical subscription is $16 for one year.

Bernie sent BYTE several copies of the newsletter, and a reprint book entitled *Musical Engineer's Handbook*. The newsletter has been published by him and his associates for several years, and is circulated to a select group of electronic music people. The beginnings of microprocessor automation in traditional electronic music are found in the pages of *Electronotes* where the November 1975 issue, EN#59, starts a series of tutorials on what a microprocessor can do for the electronic musician, written in terms familiar to the music person. The author of the series, which extends through four issues of *Electronotes*, is Bill Hemsath. The theme is continued in the March 1976 issue where Douglas Kraul discusses some elementary points about "Analog Interfaces for Microprocessor Systems." *Electronotes* is jammed full of information on circuitry and equipment needed to produce electronic music sounds, and will provide an excellent and specialized forum for those who want to find out about progress in this field.

The *Musical Engineer's Handbook* is a publication in the same vein, as one might expect since it is made up of materials based upon past issues of *Electronotes*. Its dedication reads: "This book is dedicated to musicians everywhere and of all times, without whom many of us would have only electronics to do." It is a very thick photo offset publication (354 pages) in a GBC binding, containing fundamental information on the technology of electronics in music applications. The book has a combination of background theory with numerous practical applications circuits. It is an essential sourcebook for anyone seriously interested in the art and practices of electronic music. The book is available from *Electronotes* for $18, in return for which you'll get thousands of dollars worth of ideas.■

Still More BYTE's Books

•**DESIGNING WITH TTL INTEGRATED CIRCUITS** by the Components Group, Texas Instruments Inc. Edited by Robert L Morris and John R Miller.

People often ask questions like "Where do I get basic information on hardware design?" One answer is in **"Designing With TTL Integrated Circuits."**

This book, published by McGraw Hill in 1971, is a fundamental starting point for any person designing peripherals and custom logic employing TTL integrated circuits. While its publication date precludes any reference to the later additions to the TTL 7400 series of components found in the **Data Books,** it is nevertheless the source of a wealth of ideas on TTL integrated circuits and design of logic with this family of circuits.

What is fanout? You may have heard this term mentioned at computer club meetings or in advertisements for circuitry, or in articles in BYTE. You can find out background information on the calculation of fanouts by reading the chapter on **Circuit Analysis and Characteristics of Series 54/74.**

Worried about noise, shielding, grounding, decoupling, cross talk and transmission line effect? (Or, more properly, did you know you should worry about these effects in certain circumstances?) Find out about general precautions and background information by reading the chapter on **Noise Considerations.**

The chapter on **Combinatorial Logic Design** gives 53 pages of background information on Boolean algebra and practical representations of logic in the form of SSI gates. The chapter includes a description of Karnaugh mapping techniques and the minimization of logic. From combinatorial design, the book progresses into **Flip Flops,** including background information on the workings of these devices, and fairly detailed descriptions of the uses and applications of these devices including synchronization of asynchronous signals, shift registers, flip flop one shots, etc. Then the book returns to static combinatorial logic with its description of the **Decoders** available in the 7400 line as it stood in 1970-1971.

A chapter on **Arithmetic Elements** gives fundamental descriptions of binary arithmetic, diagrams of the basic gate configurations for combinatorial logic adders, and a section on number representations for use in computers. Much of the material in this section is dated, due to the fact that the later 74181 series of multiple function arithmetic units had not yet appeared when the book was written. But for a background on arithmetic operations implemented with the simpler 7483 circuits, this chapter is ideal. A chapter on **Counters** and a chapter on **Shift Registers** complete the detail logic sections. The book is closed out by a chapter on miscellaneous **Other Applications** including a simple binary multiplier, a 12 hour digital clock and a modulo-360 adder.

The most important use of this book is its value as an introduction to TTL logic. By reading and studying it, you will begin to understand the ways in which SSI and MSI TTL gates can be utilized in your own experimental logic designs. After studying this text, you should be able to make much more sense out of the technical information summaries typically published as specifications sheets and data catalogs.

Order your copy today from BYTE's Books, $24 postpaid.

•**MICROCOMPUTER DESIGN** by Donald P Martin, Martin Research. Edited and Published by Kerry S Berland, Martin Research.

Purchase your copy of the definitive source for circuitry and hardware design information on the 8008 and 8080 computers today.

Even Intel, the originator of the microprocessor revolution, is hard put to compete with the wealth of information found in Martin Research's new second edition of **Microcomputer Design.** This is the book which was originally published as an expensive (but quite practical) engineering report in loose leaf form, at about the time the microprocessor technology was first catching on in the form of the 8008. This 388 page second edition of the manual is loaded with detailed information on how to build and use computers based on the 8008 and 8080.

But even if you do not intend to use the 8008 or 8080, the practical pointers on digital logic design, peripherals and applications of hardware techniques will more than justify the new low price of $25 for this handbook. **Microcomputer Design** is a must for 8008 owners and 8080 owners who want to truly understand how their processors process.

Microcomputer Design is complete with numerous illustrations, tables and diagrams, plus reprints of the specifications sheets for the Intel processors. There are numerous practical examples of circuitry and many complete computer designs ranging from "minimal microcomputers" to a full blown 8080 processor.

Order your copy today, $25 postpaid from BYTE's Books.

BYTE PETERBOROUGH, NH 03458

___ Designing With TTL Integrated Circuits $24

___ Microcomputer Design $25

Send to: Name __________

Address __________

City ______ State ______ Zip ______

☐ Check enclosed

☐ Bill MC No. ______ Exp. Date ______

☐ Bill BA No. ______ Exp. Date ______

Signature __________

Please allow six weeks for delivery.

Continued from page 14

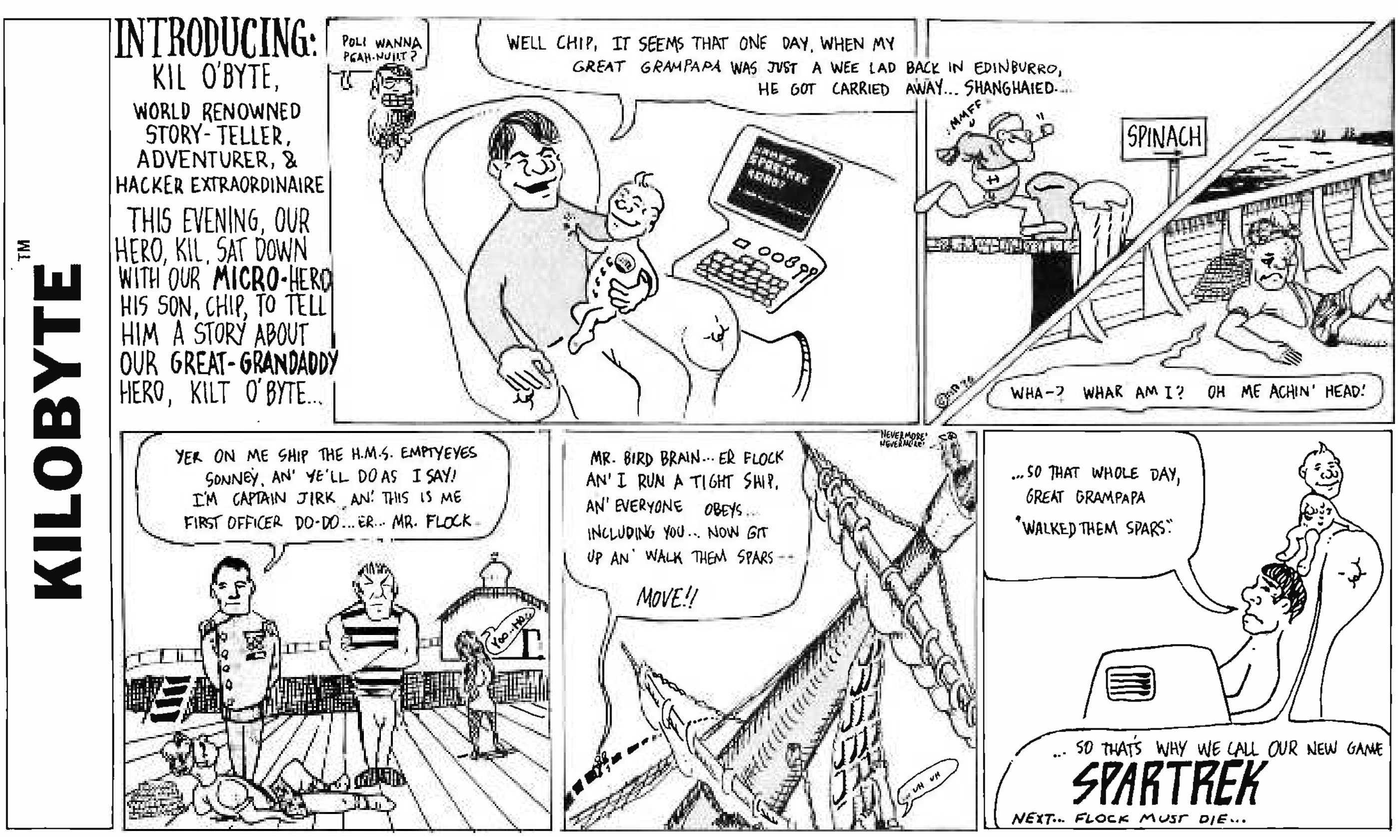
decision boundary. There are innumerable ways to vary this technique.

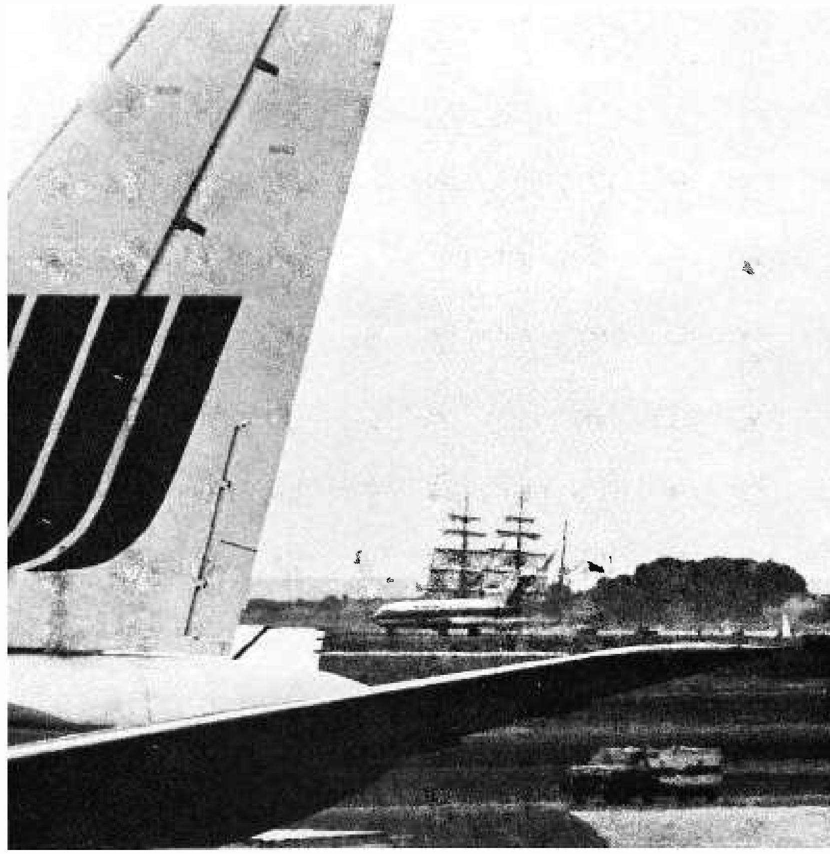
For those who would like to try their luck at hand sent Morse decoding, I have included a brief flow chart outlining a typical process from approach (3). Successful models have been constructed by Pickering Radio Co (to name but one company . . . look at *Ham Radio, QST,* or other amateur radio magazines) and a computer model has been designed using about 4 K of memory on a PDP-12 as described in a master's thesis by J A Guenther. For those interested in the exact PDP-12 algorithm, Guenther's thesis is available for a nominal charge from the National Technical Information Service of the US Department of Commerce [*See bibliography*].

As you can see, the problem of translating hand sent Morse code is not easily solved. To further complicate matters, most of the really good information is classified, or even worse, proprietary to a particular company. So, I wish you all the best of luck, you'll probably need it. As a final note, just remember: When you think you've got the problem licked, Chisholm's Law of Human Interaction will apply (eg: Some joker – radio amateurs call them "lids" – will come along with a "fist" that will tear your algorithm apart).■

BIBLIOGRAPHY

1. Blair, C R, "On Computer Transcription of Manual Morse," *Journal of the Association for Computing Machinery*, pages 429-442, vol 6, no 3, July 1959.
2. Duda, J S, "Noise Reduction for CW Reception," *Ham Radio*, pages 52-55, September 1973.
3. Gold, B, "Machine Recognition of Hand-Sent Morse Code," *Institute of Radio Engineers Transactions on Information Theory*, pages 17-24, March 1959.
4. Gonzales, C and Vogler, R, "Automatic Radiotelegraph Translator and Transcriber," *Ham Radio*, pages 8-23, November 1971.
5. Guenther, J A, *Machine Recognition of Hand Sent Morse Code Using the PDP-12 Computer*, master's thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, December 1973. Available as document number AD-786-492 for a nominal cost from the National Technical Information Service, US Department of Commerce, 5285 Port Royal Rd, Springfield VA 22151.
6. McElwain, C K and Evans, M B, "The Degarbler – A Program for Correcting Machine Read Morse Code," *Information and Control*, vol 5, pages 368-384, May 1962.
7. Petit, R C, "Morse-a-Verter," *QST*, pages 30-35, January 1971.





Waiting to leave at Logan Airport in Boston, Ed Zealy captured this interesting juxtaposition of transportation technologies. The tail of a 707 frames the distant image of a 727 masked against a sailing vessel leaving Boston as a part of "Operation Sail."

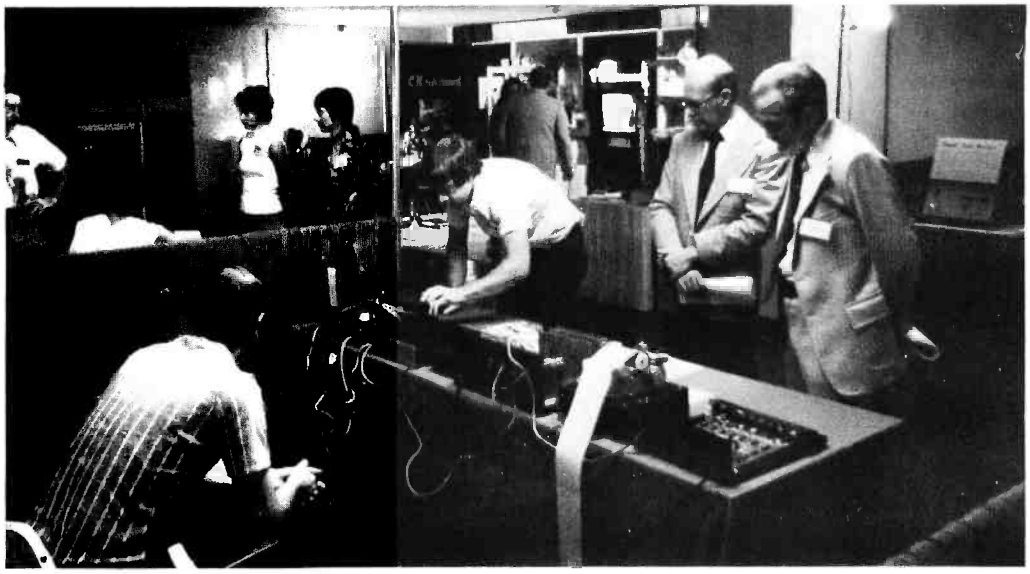
Travelogue

. . . Notes by Carl Helmers, Editor

On July 16, 17 and 18, the American Radio Relay League's 1976 National Convention was held in Denver CO at the Denver Hilton hotel. The ARRL is the amateur radio operator's main organization and representative in matters concerning the legal availability of amateur bands in the radio spectrum. Without the ARRL, amateur radio as it is today and has been practiced since early in this century would not exist. (For information on amateur radio, contact the ARRL at 225 Main St, Newington CT 06111.)

As part of the affair, reflecting the interest of many radio amateurs in computing applications to their field, the Denver Amateur Computer Society was asked to coordinate a series of microprocessor sessions at the convention, and this in turn helped attract a number of personal computing vendors and spectators to the exhibition area. BYTE was present at the con-

A sign in the lobby of the Denver Hilton greets the several thousand visitors to the ARRL convention.



Gary Kay (leaning over table) and Joe Deres (seated) of Southwest Technical Products demonstrate their wares at the ARRL convention. The table contains an SWTPC 6800 system, CT-1024 terminal, GT-6144 graphics display unit, and printer.

Every manufacturer shows off the product in the factory. At MITS, the new Altair 880B machine is quite effectively integrated with three floppy disks, a Lear-Siegler ADM-3 terminal, and a disk operating system in the display room. Here Dave Bunnell and Carl Helmers are engaged in a conversation with the demonstration system in the background.

vention with a booth in the exhibit area staffed by Deena and Ed Zealy, Beth Alpaugh and me. I also participated in one of the microprocessor oriented technical sessions by giving an informal talk at which I shared the forum with Jack Cox WOKMV.

After the convention, we all flew to Albuquerque where we spent some time talking with Ed Roberts and Dave Bunnell of MITS, then Beth and I flew to San Antonio to pay a visit to Dan Meyer, Gary Kay and Joe Deres at Southwest Technical Products. Perhaps the highlight of the whole trip (at least in my mind) was entirely unplanned. While at Southwest, Ron Komatz, the local representative for Motorola Semiconductor Products in Austin TX, walked in. In passing, he suggested something like "How would you like to come back with me to Austin this afternoon and take a quick tour through the Motorola Semiconductor wafer fabrication facility?" With an offer like that, we could hardly refuse . . . .

So, after some hurried reservations changes and Ron's call ahead to the people

The interior of the main assembly area at MITS is illustrated in this shot taken during the tour of the facility.

A candid shot of Ed Roberts, president of MITS, during an informal discussion in his office in Albuquerque.

at the Motorola plant, we drove up to Austin in Ron's car for a quick tour (total time, less than an hour and a half including 45 minutes of some informal discussion with several of the engineers and software people at Motorola).

It is in places such as the Motorola plant in Austin where the space age technology of integrated circuits and the technological leads of the American semiconductor manufacturers are so much in evidence. The silicon wafer fabrication plant is like a science fiction image: clean rooms with highly filtered air, workers dressed in lint- and dust-free smocks undergoing cleaning procedures prior to entering the fabrication area, exotic gases and electricity piped into the work areas, intricate optical instruments for the microscopic photo reproduction of IC mask patterns, air bearing transport slides for the disks of silicon being handled by the facility, red glowing diffusion ovens maintained at carefully controlled temperatures needed to dope the silicon chips with precise amounts of impurities at each stage of production.

The result at the end of multiple stages of the fabrication process, which we viewed through glass partitions, is a three inch wafer of silicon with hundreds to thousands (depending on the particular IC) of tiny circuit patterns, waiting to be ground down to less than 10 mils thick, scribed and separated into individual pieces which can be tested in automatic equipment then shipped overseas for assembly. It is the high technology of such semiconductor facilities which makes possible the personal computer as we know it today, and as it will improve and evolve in the future.■

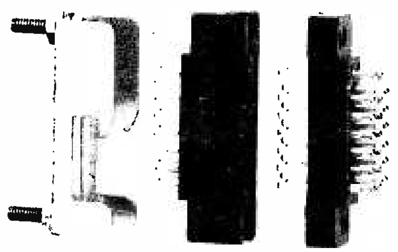
What's New?

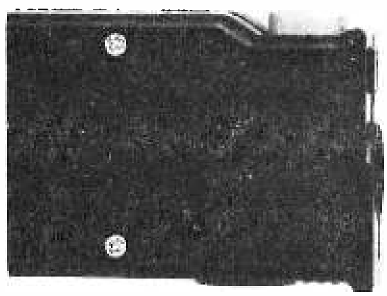
Mating Game

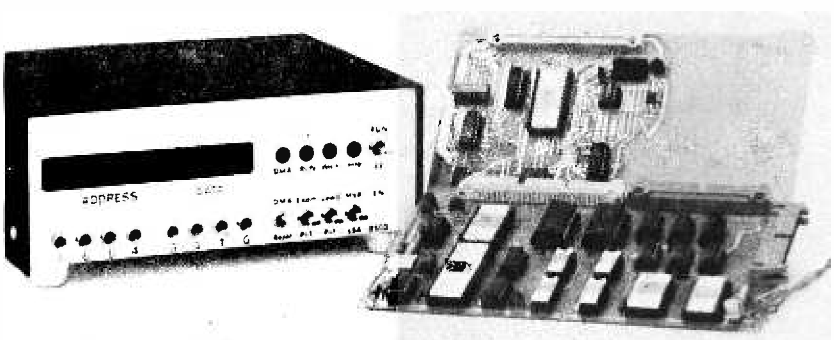
What do you use for a plug which will mate your computer with the real world across a cable? For parallel interconnection, one option is this Hirose 20 pin connector which according to its distributor has been widely used in the video field wherever multiple wires must be used with quick connect and disconnect.

These connectors are heavy duty, and will handle audio, video and machine control signals. They were designed for quick disconnect when the light colored button (see photo) is pressed. The contacts are high quality brass with silver plating and the cable assembly features a heavy duty wire lock.

Physical ratings are 350 VAC at sea level, 3 amperes per contact, insulation resistance 1000 megohm at 500 VDC, contact resistance 7 milliohms maximum. This press release information was provided by the Hirose distributor, John Anthony Television, Childs Park Rd, Dingmans Ferry PA 18238. As a special introductory offer to BYTE readers, the price of a combination male cable connector and female chassis connector is $14.95. The same price applies to the alternate combination of female cable connector and male chassis connector. Normal prices are $12.25 for male cable, $12.50 for female cable, $3.90 for female chassis, $4.50 for male chassis. OEM and quantity discounts are available as well.■





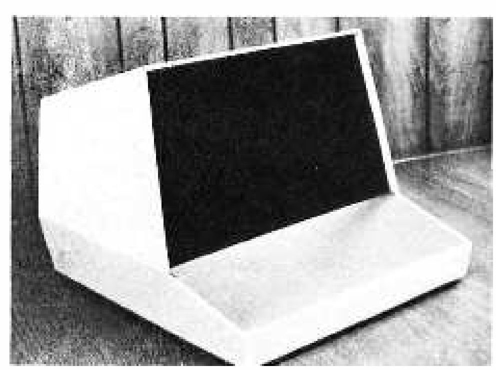


And Yet Another Dedicated Industrial Micro

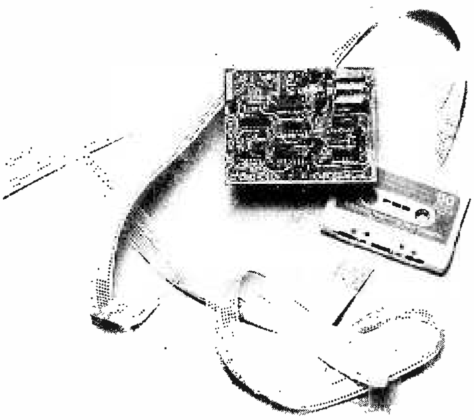
Here is an 8080 processor board intended by its manufacturer, GNAT Computers, as a dedicated computer for industrial applications. The MC80 starts at $189 and should find its way into communications systems, programmable logic for instrumentation and machine tools, and other processes where dedicated intelligence is required. In the experimenter's context, it would make a useful foundation for dedicated applications such as peripheral controllers, mobile computers for land, sea and air navigation, smart terminals, etc. The basic board has 256 bytes of programmable memory, expandable to 512, and space for up to 2 K bytes of custom applications programming in programmable ROMs. Optionally there is a front panel for development and maintenance checkout. Delivery is quoted as 30 days after receipt of order, from GNAT Computers, 8869-C Balboa Av, San Diego CA 92123.■

Building a Homebrew CRT? Using a Commercial Digital Display Chassis?

If your answer is yes, then you'll probably want to think about the problem of creating an enclosure for your design. This new product, the VTE-101 CRT Terminal Enclosure, was shown off at the Trenton Computer Festival in May, and is designed to provide such an enclosure for the do-it-yourself person. The enclosure is made by a structural foam process which uses fire retardant high heat, high impact polystyrene. The enclosure is light weight, but is said to be rigid and tough. It is designed for a complete CRT terminal, and consists of a base section 19 inches (48.3 cm) by 21 inches (53.3 cm) by 4 inches (10.2 cm) and a shroud measuring 11.25 inches high (28.4 cm) featuring a smoke grey screen. It is said that the enclosure will take a monitor up to 14 inches (35.6 cm) diagonal measure. This housing sells for



$69.95, and less expensive models (without monitor shroud) are also available. The products are supplied with hardware, and additional openings can be cut in the material using ordinary woodworking tools. Contact Enclosure Dynamics, PO Box 6276, Bridgewater NJ 08807.■

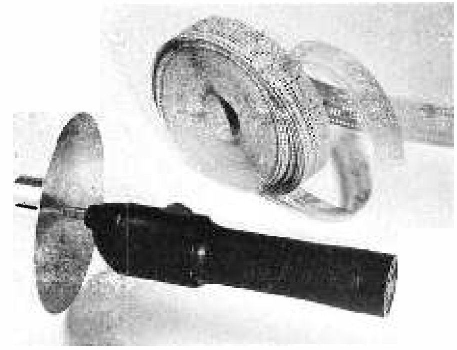


Here's an Interesting Combination of Peripheral Functions!

Electronic Product Associates Inc, 1157 Vega St, San Diego CA 92110, (714) 276-8911, sent along an announcement of this new, low cost audio cassette, Teletype or CRT terminal adapter which allows any serial TTL or MOS output to simultaneously interface a low cost audio cassette recorder via frequency shift keying (BYTE Standard) up to 300 baud and to a standard RS232 CRT and a 20 mA current loop Teletype. The adapter also simultaneously decodes BYTE Standard FSK data from low cost audio cassette players and from 20 mA current loop Teletype and RS232 CRT. Audio cassette information is decoded by a proprietary phase locked loop system developed by EPA which is said to be the most reliable method available for transferring digital data to and from low cost audio cassette players. The model TCC3 is 4.5 X 3.25 inches (11.5 cm X 8 cm) and mounts piggyback on the EPA Micro-68 development computer. The TCC3 price is $129 in singles, completely assembled and tested. Delivery is quoted from stock.■

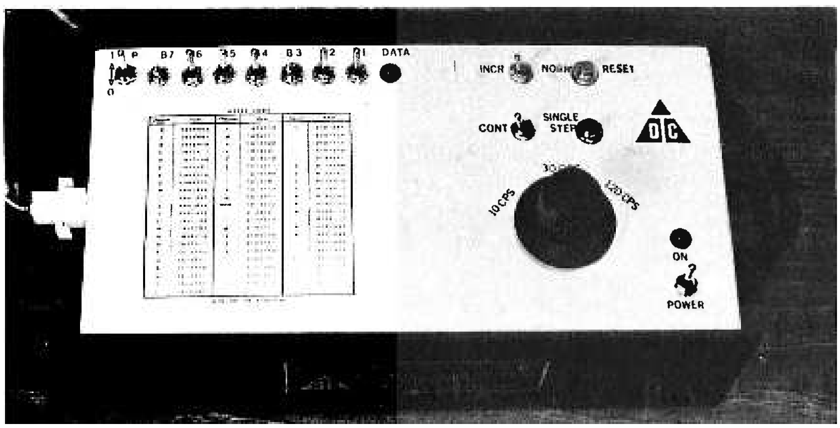
Software New Product

33 Programs and Projects for the Altair 8800 is a new self-published book by Jacques Roth, 543 16th Av, San Francisco CA 94118. This book is a 51 page collection of information printed on loose leaf pages with three hole punching. Programs range from the simple to the devious, and are all designed to run in less than 1 K bytes, with many taking fewer than 256 bytes. Projects include an XY scope plotter using an oscilloscope, two DACs and the PI/O board, a Pong game for the scope interface, a computer telephone dialer interface with several phone dialing programs, a monitor program, etc. Other programming projects include Tic Tac Toe, a 1 byte bubble sort program, and a package of 4 byte arithmetic functions including integer addition, subtraction, negation, move, set zero, left shift, right shift and multiply. The price of this collection if $5.95 postpaid from Mr Roth.■



Wind Up Your (Paper Tape) Affairs

Continuous Expression Processor Inc, 12 Main St, Natick MA 01760, has come out with this cordless paper tape winder as an accessory for use in paper tape systems. (Suppose you have one of the hand powered paper tape readers. You can automate your input processes by using this paper tape winder to pull the tape through the reader.) The unit requires two "C" size flashlight cells and sells for $29.95 postpaid in the continental United States.■



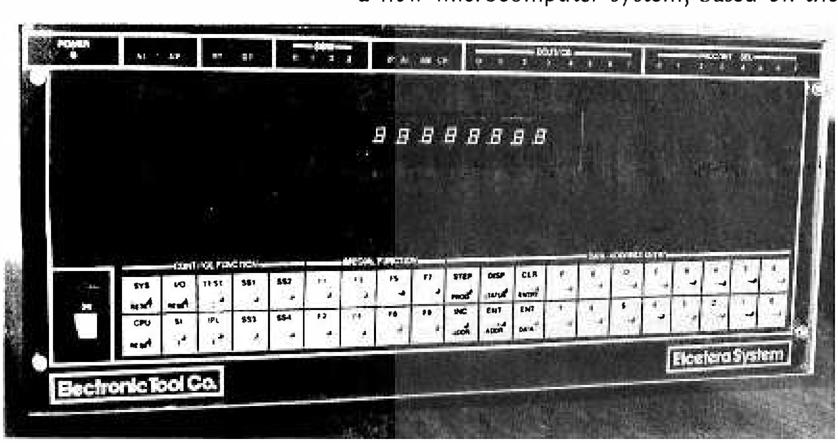
Attention: Computer Service People and Clubs

One way to check out computer terminals and other ASCII serial data and communications equipment is to utilize an ASCII pattern generator such as this new product of the Terminal Data Corporation of Maryland, 11878 Coakley Cir, Rockville MD 20852. This self contained device comes with male and female RS-232 interface plugs, utilizes a 0.01% crystal oscillator for generation of 110, 300 and 1200 baud data rates, and automatically generates ASCII serial patterns: one at a time or continuously, the same character (set by toggle switches) or scanning through the character set. The Model #900 test set is offered as a kit for $249, with a special introductory price of $199 good until September 30 1976. The assembled and tested price is $395.■

Another Completely Packaged Computer

Electronic Tool Co, 4736 W El Segundo Blvd, Hawthorne CA 90250, has introduced a new microcomputer system, based on the MOS Technology 6502 CPU, priced at $675. The ETC-1000 comes with a 40 key keyboard, a programmable 8 digit display, IO interfaces, power supply and memory. All systems are fully assembled, tested and ready to run. According to its manufacturer, the ETC-1000 is intended for system development, control, and small scale data processing applications. As a development system, it provides system support for hardware and software design work. As a control system, it offers an inexpensive high speed computing capability in a sturdy rack mountable package.

The manufacturer describes the ETC-1000 as a "full capability high-performance computer system which you can have running 10 minutes after you open the box." The system needs no external



attachments such as Teletypes, power supplies, or memory expansions to provide basic programming capability and computer operation. Many expansion options are available, including communications, real-time and DMA interfaces, memory expansions, and various add-on CPUs.

A 40 key keyboard for control and data entry is mounted on the front of the ETC-1000. This keyboard includes a full set of hexadecimal keys for data and address entry, system function keys such as load, reset, examine memory, etc. Eight special function keys which may be sensed under program control by the user are available for assignment to user-specified functions.

An 8 digit LED panel display is also contained in the ETC-1000 control unit to display memory contents, system status and user programmed information.

The ETC-1000 contains a direct input output system consisting of eight latched output lines and eight latching input lines which may be used under program control to operate external devices. A hardware interrupt system with two levels is standard; eight additional levels are available. The company says one or two independent full duplex communications streams at speeds between 110 and 1200 bits per second are supported by the ETC-1000 basic system when appropriate PROMs are included. Selection of speed is accomplished automatically by the hardware. 20 mA DC current loop interfaces are provided as standard, with EIA RS-232C capability optional.

The ETC-1000 CPU consists of a MOS Technology 6502 8 bit CPU, plus clocks, control logic, interface buffers, 1024 bytes of high speed RAM, and 256 bytes of ROM containing system control functions.

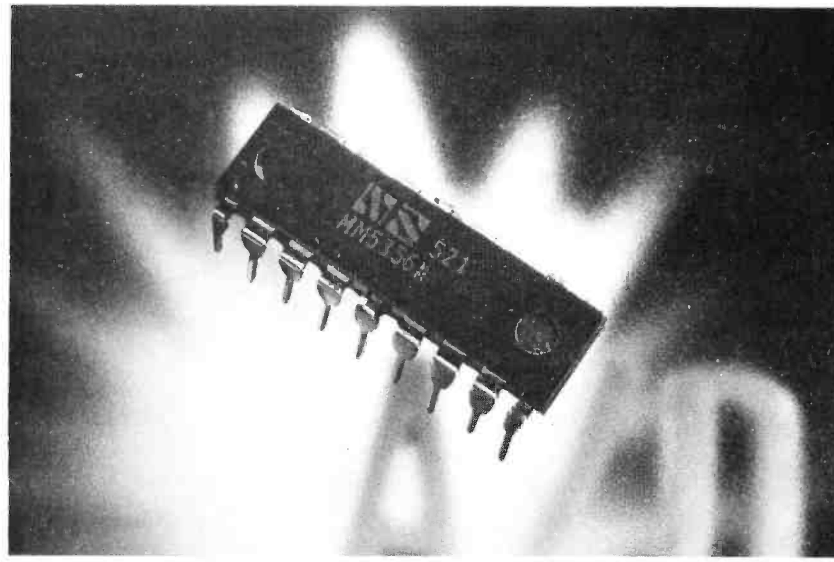
Software currently available includes a resident assembler, IO handlers, diagnostics and other support tools. The manufacturer says that BASIC and PLM support are expected to be available during the third quarter of 1976.

Availability of standard configurations is 30-60 days.■

Attention Analog Interfacers . . . .

National Semiconductor Corporation has just announced a new building block for analog input interfaces. This is the MM5356 8 bit analog to digital converter. The function provided by this chip is converting an input voltage of typically 0 to 10 V or –5 to +5 V into an 8 bit binary word. In order to operate properly, the circuit requires a conversion clock of at least 5 kHz and as much as 2 MHz. The actual conversion time required is specified in the timing diagram of its specification as 40 clock periods. Thus if the clock rate were 40 kHz, the unit could be used to generate a valid digital word 1 millisecond after the beginning of the "start conversion" input command signal. Operating at a typical microprocessor clock frequency of 1 MHz, the conversion time would be 40 μs, corresponding to a maximum sample rate of 25,000 measurements per second.

Where would this item prove useful in the context of personal systems? Well, consider the problem of reading the voltage on a thermocouple used to measure temperature. After amplification and normalization with operational amplifiers, the signal could be read by the analog to digital converter and used in a program written to implement a digital feedback loop used in controlling temperature in some way. Consider the problem of deriving the coefficients used in controlling a digital speech synthesizer: An ADC is an essential input to the process of analyzing such data. Or simply consider the general laboratory problem of using your computer as a voltmeter: Put a voltmeter analog front end on this converter and have your TV display put out the values it reads.



National Semiconductor is located at 2900 Semiconductor Dr, Santa Clara CA 95051. The 100 piece price of this conversion chip is listed as $7.95, so it certainly should be available at a reasonable price through distributors and retailers.■

Excerpts from Future History

The following series of passages was compiled by reader John W Burgeson in a term paper he wrote for his "history of technology" course in the spring term of '25. Note how the attitudes remain the same, but the objects of the attitudes evolve with time.

John W Burgeson
101 Skyline Rd
Georgetown TX 78626

Who Wants an Automobile?

Toward the end of the last century, nobody "wanted" an automobile. Whether the invention of the horseless buggy was due to accident, play, tinkering or rational thinking on the part of people endowed with mechanical abilities is immaterial for our purposes. Surely the invention did not originate with the consumer and was not made with an eye to prevailing consumer wants. Even when the first cars appeared on the road and for many years thereafter, their use for mass transportation was envisaged neither by producers nor by consumers. But today, even small children in America feel the need for a car to take them and their parents shopping, visiting, and later to school. Between the early days of the automobile and the present situation there was a long period of social learning. The learning process was, of course, not spontaneous; first of all, it could not have taken place without the original invention; second, it was a function of numerous stimuli – personal experience, education, and reading, as well as propaganda and advertising. Thus it may be said that wants for automobiles were induced, or to use Galbraith's term, "contrived." But are not most of our wants contrived in this sense? And are not most of our contrived wants, in a certain sense, original with the buyer? It can hardly be said that such want-creation is artificial.

George Katona, The Mass Consumption Society, *New York, McGraw Hill, 1964, page 55.*

Who Wants a Percom?

Toward the middle of the 1970s, nobody "wanted" a percom. Whether the invention of the personal computer was due to accident, play, tinkering or rational thinking on the part of people endowed with electronic abilities is immaterial for our purposes. Surely the invention did not originate with the consumer and was not made with an eye to prevailing consumer wants. Even when the first computers appeared in the offices of large companies and for many years thereafter, their use for mass personal use was envisaged neither by producers nor by consumers. But today, even small children in America feel the need for a percom to help them and their parents manage their affairs, help them with schoolwork, entertain them with electronic games and the like. Between the early days of the percom and the present situation there was a long period of social learning. The learning process was, of course, not spontaneous; first of all, it could not have taken place without the original invention; second, it was a function of numerous stimuli – personal experience, education, and reading, as well as propaganda and advertising. Thus it may be said that wants for percoms were induced, or to use Galbraith's term, "contrived." But are not most of our wants contrived in this sense? And are not most of our contrived wants, in a certain sense, original with the buyer? It can hardly be said that such want-creation is artificial.

George Katona Jr, The Mass Consumption Society *(Second edition), New York, McGraw Hill, 1996, page 55.*

Who Wants a Homer?

Toward the end of the last century, nobody "wanted" a homer. Whether the invention of the home robot-computer was due to accident, play, tinkering or rational thinking on the part of people endowed with cybernetic abilities is immaterial for our purposes. Surely the invention did not originate with the consumer and was not made with an eye to prevailing consumer wants. Even when the first real-time mini-computers appeared and for many years thereafter, their use for personal home management, protection and entertainment was envisaged neither by producers nor by consumers. But today, even small children in America feel the need for a homer to help them and their parents to manage their lives, protect them, entertain them and the like. Between the early days of the homer and the present situation there was a long period of social learning. The learning process was, of course, not spontaneous; first of all, it could not have taken place without the original invention; second, it was a function of numerous stimuli — personal experience, education, and reading, as well as propaganda and advertising. Thus it may be said that wants for homers were induced, or to use Galbraith's term, "contrived." But are not most of our wants contrived in this sense? And are not most of our contrived wants, in a certain sense, original with the buyer? It can hardly be said that such want-creation is artificial.

George Katona III, The Mass Consumption Society *(Third edition), New York, McGraw Hill, 2024, page 55.* ■

The formal organizations used for clubs and societies can range from the totally informal (one person arranging a regular meeting place) to the setting up of all sorts of corporate nonprofit organizational structures, etc. The informal versions usually work best for small groups; formal organization tends to increase with the size of the group. Whatever the case, the less time spent on long drawn out nitpicking at business meetings, the more time there is to devote to more interesting matters . . . .

Meeting Activities for Computer Clubs

Dr Charles F Douds
381 Poplar St
Winnetka IL 60093

A bunch of you got together and started a computer group? Congratulations! That is a good thing for this wonderful and fast growing hobby. You are meeting once every month? Fine, that will help to spread the word about the latest products, glitches, and cures. New people keep showing up at your meetings wondering what it is all about? That's the way it went in Chicago, too. And lots of other places, I'm sure. But now you've gotten to know each other, you're beginning to wonder what you should do at your meetings.

That is an important point. The activities at meetings can make or break such an organization. People don't have to come, and they won't if they don't get something out of them. They want many different kinds of things. They want to learn something. They want to share their ideas. They want to ask questions. They want to socialize. They want to see things. They want help. Of course, not everyone wants all of these at the same time and most people don't want the same thing all the time. Variety and choice are important. If you look around you at other successful special interest organizations you will find many ideas for the kinds of things you might do. Here are some that I found.

Lectures

This is often the first thing you will think of, and quite easy to organize. The lecture may be by one of your own members or somebody invited in from outside. It could be an engineer or analyst from a local firm; a college professor; a sales engineer from one of the big electronics firms or other company utilizing electronics products. The bigger firms might even send somebody in.

Arrangements should be made well in advance. Usually you will do this in person or by phone and follow it up with a letter. Be sure to include detailed instructions on how to get to the meeting location. Of course, you have to get it set up in time to publicize the speaker and his topic. Perhaps he can join you for a meal before or after. Don't forget to send a letter of thanks afterwards. Better yet if you can send it to his boss when that is appropriate.

There are two main problems with lectures. One is finding out how good the speaker is beforehand. A major problem here is that he or she may be good for one kind of audience, but not for yours. It is very likely that your group will be a very mixed bag of hardware types, software types, and enthusiastic types that don't know much

Activities at meetings of computer organizations can make or break the group.

A lecture, with a good speaker on a topic of interest to the group, is one of the easiest types of activity to organize.

about hardware or software. That poses a real problem to a speaker in front of a large group.

The other problem with lectures is having too many of them. They are about the easiest kind of program to arrange, but people get tired of being talked at. Discussion with large groups doesn't work too well. A couple of poor speakers in a row can easily turn off a whole group. Program variety and member choice are important ingredients to a successful group.

Clinics

Clinics are less formal and often involve smaller groups than a lecture. More than one clinic can go on at the same time. People have a choice and different types of topics can be handled. A clinic is usually limited, for example, to one hour. It may also be a lecture, but it can take many different forms. The topics covered can be just about anything. They might include: design of an IO circuit, printed circuit artwork techniques, debugging procedures, an overview of high level languages, etc. The presentation may be in the lecture mode, or the author might simply talk, work on a blackboard, use flip charts, work at his computer keyboard, or use slides or transparencies.

Clinics are greatly enhanced if a handout is provided. It may be just a list of key points or provide details about what is being discussed. Sometimes a marketing minded manufacturer may be willing to provide material to a local member presenting a clinic about the manufacturer's equipment. For reference purposes handouts should include the author's name and address.

If hardware is used or demonstrated, you need to make sure that the group is small so that everyone can see. If you are meeting in a high school or a college and have a crowd, you might be able to use a closed circuit TV so that the people in the back of the room don't go away saying it was lousy, while the ones in the front say it was great. For the same reason, consider the use of PA systems if you are likely to have a large audience.

It is very important that the host check up well beforehand to find out what the author will need and what he or she is going to bring. Assume that the speaker will forget things like extension cords, chalk, and erasers. It is particularly important to check on projectors, screens, and electrical outlets. Make sure that they actually have power in them. Find out where the lighting switches are (especially in motels and hotels!) and where the background music can be turned off. Again, don't forget the thank you letters.

When you get to the point that on occasion several clubs get together for a "meet" that might last a weekend, or you decide to put on a real bash in just your own group, you will probably have several clinics.

This is great because people can then get to the topics that particularly interest them, and still not be trapped in a room for something they care little about. But many people will want to get to all the clinics. This can sometimes be worked out by careful scheduling and persuading the authors to present their clinics twice. This isn't quite as bad for the author as it sounds. An hour clinic is really only 50 minutes and usually at least another 10 minutes should be allowed for questions. Besides, the practice is good for him or her. (I expect to see a lot more women actively involved in this hobby than in other comparable ones such as ham radio or model railroading.)

Demonstration ("Hall") Clinics

If you don't have a lot of rooms for all the clinics you would like to provide, or if you just have one big room where several speakers would interfere with each other, "hall" clinics might solve your problem. They can literally be set up in the halls; but more usually a number of them will be held in a large room – the kind that used to be called a hall.

In the demonstration clinics one person does his thing for an extended period of time. Perhaps he is assembling a kit, laying out the artwork for a printed circuit board,

NOTE:
A brief version of this article originally appeared in the *Micro-8 Newsletter*, vol 2, no 2.

demonstrating his operating system, or whatever. The topics may be similar to the regular clinics, but the format is different. The author does not lecture. He simply talks about what he is doing. He explains and answers questions as he goes along. This gives people the chance to see all the details and exactly how things are done. People are free to move from one demonstration to the next spending as long as they like at each.

It is often important that tables or railings be set up to keep people a few feet back. This makes it possible for a half dozen or so people to see, while still being close enough to observe the details. It is best arranged so that the demonstrator can hand things to the viewers if he wishes to.

Participation Clinics

These clinics would be called "labs" in a school curriculum. The audience gets their heads and hands into the topic and learns by doing. These are good for such things as lessons on programming or introductory circuit design. These clinics require very careful preparation by the author. He or she should fully test out the lesson beforehand. Of course, not a whole lot can be accomplished in an hour or two, but the most important thing is that it gets people started. The author can only do a little bit of teaching, followed by a lot of individual helping. Often these kinds of clinics are best run by two people working together.

A panel discussion, with or without audience participation, can be a good round robin affair which helps clarify and present concepts on a given theme.

If equipment is going to be used – pin boards, voltmeters, etc – it may be necessary to have people sign up beforehand. Sometimes it may be possible to have people work in pairs. As long as the room doesn't get too crowded, it may be possible to let others in as spectators. The host should be prepared to shoo out excess people. The author will appreciate not having to do this for he or she may be busy with the instruction.

Make sure that there will be enough materials available for a reasonable number of people to participate. Don't call it a "participation" clinic if only two or three can do so.

Do It Yourself Clinics

In these kinds of clinic the audience builds something and takes home a working device. They are immensely popular if adequately publicized, but they often require a lot of work to prepare. The item is announced beforehand. It might be a logic probe or a simple power supply. Participants send in their checks for the cost of parts. The announcement includes the list of tools needed, specified *very* exactly. (You would be amazed at the variety of soldering irons and pliers model railroaders will bring to build a printed circuit item!)

At the clinic the author then shows the participants how to build the device step by step. It is very important that there be adequate facilities for checking the devices, too. You are trying to provide people with their first success in a new (for them) endeavor. There is nothing like the feeling of going home with a gadget you know for sure works.

Obviously, it is important that the project be small enough so that it can be built and tested by the neophyte in the available time. It is important that this be checked out beforehand. If there are more than six or eight participants, the author should have one or two helpers – people who know a capacitor from a resistor, can read the color code, and who can recognize a cold-soldered joint.

It might be possible to have "advanced level" clinics of this sort. The problem is that people who are not adequately advanced will still sign up anyway. Not only are they likely to go away dissatisfied, but the author may get trapped into having to rebuild a half dozen units for these people. Keep the projects simple and short. With the complex chips available today, one still might be able to come up with relatively sophisticated projects.

Of course, other variations are possible. If it were clearly advertised as such in advance, it might be possible to start construction at one meeting and complete it at a second meeting. The more advanced builders would probably complete the project and be able to test it themselves, so the second session would involve a smaller group.

The host should make sure that there are suitable tables and adequate power outlets to accommodate all participants.

Show and Tell Sessions

Here we take a page out of the stamp collector's and photographer's book. It is a lot of fun to simply see each other's equipment. So much the better if it is up and working, but projects under construction can be very interesting, too.

I suppose you could even have prizes for the best shaped letters on a TV set – with a separate class for monitors. How about one for the hardest to read Teletype? Or the prettiest set up? Or the worst (or would it be "best"?) job of haywiring that actually works! The possibilities are endless. Why not announce that there will be prizes, but not announce the categories. I would expect that altogether too soon we will be having too

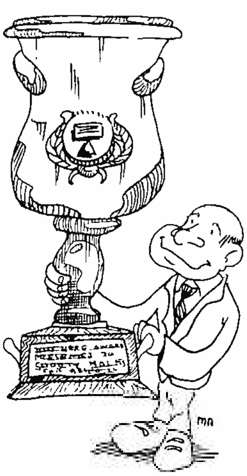
Swap shops and auctions, formal or informal, can be a good addition to a meeting which helps to solve nagging problems of what to do with surplus junk or where to find that critical part or subsystem (which is, of course, someone else's surplus junk).

many committees working on too many prizes, although I must admit that contests do seem to stir up interest and provide real progress among enthusiasts.

I think that the most important aspect of individual displays, apart from the conversations that develop, are the many ideas that one can acquire in such a short time. These may be little details that make life easier or they may be whole new vistas that weren't really meaningful until seen "in the flesh."

Workshops

Workshops can take many forms, cover many subjects, and be conducted in many ways. An expert might work on debugging equipment that participants bring in. Or two or three people might design a special interface. The area is set up in such a way that the audience can watch and listen, but they do not participate. It is important, of course, that the experts do their thinking out loud. Again, a PA system or closed circuit TV may be helpful.



A second type of workshop is essentially a "closed door" session, at least once it starts. Discussion among all the participants is expected. The output of such a workshop is often something that is going to benefit the whole club or others. For instance, the workshop might be devoted to developing a chart comparing the characteristics of certain types of kits, developing the rules and standards for a local computer conferencing network, or other such things. The output of the workshop might become a regular clinic at another meeting. Such workshops require minimal facilities on the part of the host, but the participants certainly have to come adequately prepared and a competent discussion leader is needed.

Panel Discussions

A good panel discussion can fit into many types of programs. The topic area should be definite, but not too narrow, ordinarily, for a hobby group. Sometimes the usual kind of panel discussion, where the discussants speak their pieces then argue back and forth, is appropriate; but it is probably better for a computer group meeting to have a question and answer panel. The audience asks questions, and one or more panel members tell how they handle the problem. Sometimes members of the audience turn out to have good suggestions, too.

The moderator is the key person in setting up a panel. He or she must not hog the stage, must be able to summarize long or confused questions, and have a facility for steering the discussion among the panel members. The biggest difficulty is in getting the audience participating. It is often wise to have several questions planted with a few friends in the audience. It is entirely natural for people to hesitate to speak up in a crowd until they hear one or two questions that they recognize as being as simple, or simpler, than theirs. A dozen people may have the question in mind, but no one wants to be first. The other important point is to stop the session while the questions are still coming. Stop at a high point, not in a valley – and everyone will go away thinking how great it was.

Tape Slide Programs

Many of the activities above can be worked up into a prepackaged recorded program with accompanying slides. The visuals probably should be 35 mm slides because projection equipment is always available. It might be possible to use cassette tapes, but standard reels are probably better. The problem is that it may be awkward to get adequate sound volume for an ordinary size audience from a cassette machine, while there is seldom a problem with an ordinary home tape recorder.

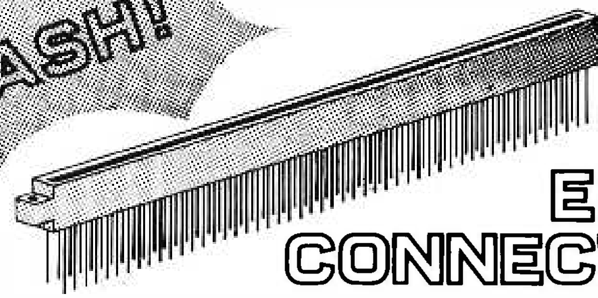
Tape slide programs can be made successfully by amateurs, but they do not work out by simply recording a live clinic even if it uses a lot of slides. A script has to be prepared and worked over. The final taping

A SUPERIOR "301" TYPE OP AMP *TWO FOR $1*

SOMETHING FOR LINEAR FANS...

We bought a batch of 201s, which are the same as LM301s but with extended range and better specs. These are factory prime overruns, and are available in minidip or TO-99 style cases--please specify which case style you want.

FLASH!



100 PIN EDGE CONNECTOR

$5.00 EACH--$22 FOR 5

SOMETHING ALTAIR OWNERS HAVE BEEN WANTING...A 100 PIN EDGE CONNECTOR THAT ACCEPTS ALTAIR PERIPHERALS. PEOPLE HAVE REPORTED A HARD TIME FINDING THESE CONNECTORS, SO WE GOT A BUNCH FOR OFF-THE-SHELF DELIVERY. DESPITE THE LOW COST, THESE CONNECTORS FEATURE GOLD-PLATED 3 LEVEL WIRE WRAPPING POSTS AND A SNUG FIT FOR YOUR PERIPHERAL CARDS. COME AND GET 'EM...

new part, old price:

21021 1K RAM

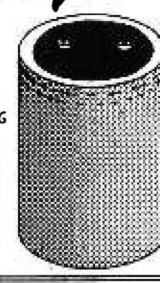
SAME AS 2102 TYPE RAMS, BUT FULL COMMERCIAL TEMP RANGE (0-55°C), LOWER POWER (17 MA TYPICALLY), AND FASTER (450 NS WORST CASE GUARANTEED). A SUPER 2102.

$1.95

150,000 µf at 10 V!

$5.95

+2 LBS SHIPPING



LISTS FOR OVER $14!

WE HAVE A LIMITED SUPPLY OF SPRAGUE POWERLYTIC CAPACITORS. DESPITE THE HUGE AMOUNT OF CAPACITANCE, SIZE IS ONLY 3 1/16 BY 4 3/16. IF YOU HAVE A MICROCOMPUTER WITH A FARAD DEFICIENCY, THIS IS THE TONIC...OR BUILD A BEEFY POWER SUPPLY AROUND IT.

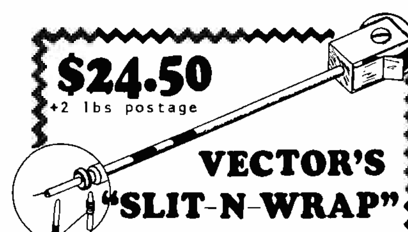
SPECIAL: TINY ROTARY SWITCHES

TINY ROTARY SWITCH. 1 POLE, 7 POSITION PLUS OFF. DANDY FOR PROGRAMMING OR MINIATURE ELECTRONIC PROJECTS.

6/$3

$24.50

+2 lbs postage



VECTOR'S "SLIT-N-WRAP"

This tool is a manual wrapping device, which supplies insulated wire from a spool directly to wrap posts without external prestripping or precutting. More than one level of wrap is rarely required. Speed your breadboarding time tremendously with this tool, and don't bother stocking all that prestripped and precut wire you would normally have to have.

CAVE grafix

LOW PROFILE, LOW COST SOCKETS

16 PIN LOW PROFILE SOCKET, SOLDERTAIL, TIN PLATED: 10 FOR $2.00

14 PIN LOW PROFILE SOCKET, SOLDERTAIL, TIN PLATED: 11 FOR $2.00

LIMITED QUANTITIES ON BOTH

"HOW CAN I PLAY WITH MICROCOMPUTER ICs WITHOUT BREAKING MY BUDGET?"

8008 CHIP SET: 1-8008, 8-2102s, 1-5203 EROM.

$29.98

+2 lbs

HOBBYWRAP TOOL

$41.95

Want the convenience of a power wrapper? Try our Hobbywrap tool. You get the tool, nicads, bit, charger, and instructions. Why solder? Save time and effort.

Play with wire? Request our flyer!

If you're an experimenter looking for some state-of-the-art parts...a computer enthusiast requiring peripherals for your microcomputer system...a tinkerer looking for time-saving tools and gadgets...an audiophile who likes to play around with sound..a ham who needs a digital clock, or a memory board...a CBer who needs a home supply for his mobile rig...or even a mad scientist, you'll find something to tickle your fancy in our flyer.

If you don't have one, ask for one.

TERMS: Californians add tax. You may place Bankamericard® and Mastercharge® orders via our 24 hour answering service (415-357-7007). No CODs, the paperwork is too much. When weight is indicated, please add sufficient postage-excess refunded.

Please note that some items are limited quantity. First come, first served. Thanks for your patronage!

Orders under $10 please add 50¢ service charge.

GODBOUT

BILL GODBOUT ELECTRONICS
BOX 2355, OAKLAND AIRPORT, CA 94614

should not be done by reading from the script unless the person is a professional actor or announcer. A clear, distinct cue signal, described at the very beginning of the tape, must also be provided. It takes a lot of work to put together a good tape slide program, but the results can be well worth it. They provide direct access to top notch information, especially for those groups in remote locations.

This is one way that the manufacturers might be able to make a name for themselves. Even if a program were purely promotional, it still would be interesting to meet the people at the plant, see their facilities, and to watch their products in action. Of course, it would be even better if they would go into the unique features and application possibilities.

As an example of such materials in a specialized field, the National Model Railroad Association has over 30 such programs available to its members for a deposit and return postage (educational material rate). The waiting list is months long.

A Summary of Meeting Activities

- Lectures
- Clinics
- Demonstration ("Hall") clinics
- Participation clinics
- Do it Yourself clinics
- Show and Tell sessions
- Tape slide programs
- Workshops
- Panel discussions
- Tours
- Swap shops
- Auctions
- Mixer sessions
- Contests
- Business meetings

Tours

With a little imagination, some pleading and cajoling, and a bit of persistence, you may be able to come up with some fascinating tours. Many of your group may never have seen a big computer installation. If they have, then they probably haven't seen a manufacturing operation. Or how about an automated security system? It does not matter if there are no electronics manufacturers around; there undoubtedly will be many applications nearby. Then there are the potential applications . . . quarries, turkey farms, mushroom growers, etc.

But don't forget your own members. Maybe some of them would be willing to show how they have things set up at home. Devise a way to get reasonably small groups around to several homes, making sure that only those who are wanted stay on until the wee hours of the night.

Swap Shops and Auctions

Auctions are a lot of fun. If you are having a two-day affair, schedule one for around 10 in the evening. It may last until 2 AM, but everyone has a lot of fun — except, perhaps, for the auctioneer. It usually takes a crew of people to put on a reasonable size auction. They need to be adequately prepared with a good set of forms, display tables, and enough space.

The usual type of swap session, along the lines of an amateur radio "hamfest" is well known, where everyone selling gets some table space and makes his own deals. While auctions can easily collect a percentage of the sale price (often 10%), swap sessions are more easily handled by the host organizations collecting a fixed entrance fee from sellers, buyers, and lookers.

As a service to its members, there could be a swap session bulletin board at every meeting. Just bring a cork board, a bunch of 3 X 5 cards, and some thumb tacks. Even if the program scheduled flops, you might still pick up a bargain.

Mixer Sessions

If your club is small, everyone will soon get to know what each other's interests are; but once it gets over 30 or so people, this will no longer be true. There will be the devoted core who are doing most of the work organizing these programs and doing all the other things that need to be done. If you are one of them and think you know everyone's interests, you will probably discover that there are a lot of people showing up who don't know. At this point it is time for a mixer session.

Try putting signs up on the walls with words indicating topics for discussion. These might be the names of manufacturers, types of equipment, programming projects, etc. At this point a bit of the summer camp counselor is needed to get the people to assemble in these areas and to discuss the topic. Of course, it would be wise to have a few people designated to cover each one and to handle the initial introductions.

Once when I attended a regional model railroad convention, I wanted to find all the people I could who were interested in applying electronics to that hobby. I hung a piece of cardboard on my back with a few key words in large letters, met several interesting people, and was given the names of several others not at the convention. When a speaker has to cancel out at the last minute, you might try that idea. Hang a sign on everyone's back as they come in the door. I bet it will be one of your top-rated meetings.

Contests

Contests in many special interest and hobby groups often seem to wind up taking on a life of their own, leaving most of the membership out. Most contests can be won by the liberal application of not just skill and time, but also money. They certainly can provide an incentive to improve designs and techniques. A major problem is to determine what the goals of the contest are to be. It seems to me that for a bunch of enthusiasts, they should be to encourage participation and to have fun. They shouldn't get out of hand in terms of skill or money demands. When this happens, they just become spectator events. In my opinion, the major goal of contests should be to encourage active participation. We have enough TV watchers now; let's apply our imaginations and have more keyboard button punchers and TVT watchers!

Business Meetings

We hate them, but we can't seem to get along without them. Every organization has to have business meetings. Unfortunately, the kind of people who enjoy business meetings tend to be the ones who run them. Certainly there is little reason why a computer hobbyists group has to have a business meeting involving everyone as a part of every meeting. The amount of time spent on business meetings should be minimal!

The work of the organization should be done outside the whole group sessions by a steering committee. Undoubtedly, you will wind up electing officers (although the Chicago Area Computer Hobbyists Exchange operated for more than a year without any). The really important thing is for all to know who is making what kind of decisions or taking what type of actions. You can have officers and still be very confused about this. The titles don't make the difference. Sometimes they just seem to attract people who like titles for the sake of having the titles.

However, it is vital that procedures be maintained so that actions by the few can be overruled by the whole body, or a clique thrown out, when necessary. *You* are never close minded, secretive, or not doing things in the best interests of the organization, of course. But you can only prove this by making it easy for others to do things differently than what you think is best. Make it easy, and they will probably go along with you.

Do not overorganize! There is very little that needs to be "business like" about a hobby organization. It is not a business. People come voluntarily. They come to learn, for relaxation, for fun. The avocation is computers. It is not setting up committees for everything, or writing rules, regulations, and procedures inappropriate for an organization of volunteers.

Do set up committees, but only as needed to ensure that things get done, or to keep the organizational types out of everyone else's hair.

Keep the business meetings short and to the point. Provide a variety of types of programs. Don't be afraid to experiment. Provide choice and change. You'll have a great club.■

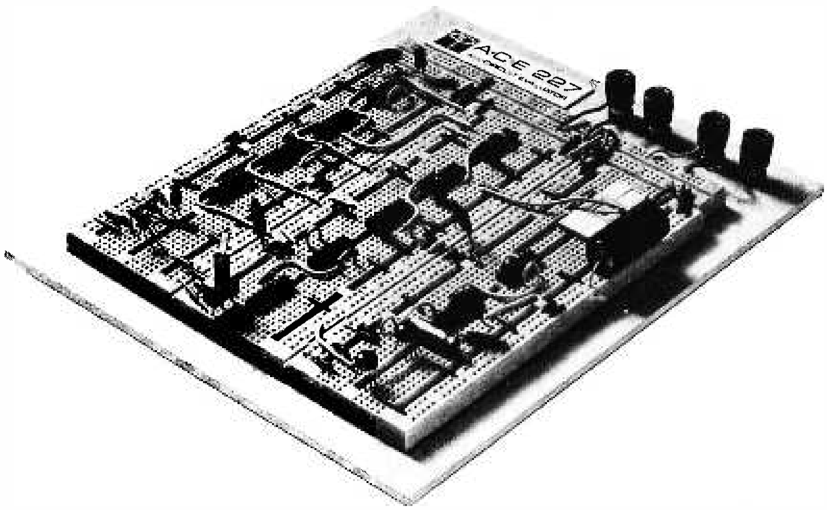
What's New?

The following item is adapted from a press release provided by A P Products, Box 110, 72 Corwin Dr, Painesville OH 44077. It is printed here for its value as background information on the use of solderless prototyping jigs to try out experimental circuits on a temporary basis.

Solderless Breadboards

What is a solderless breadboard? How does it work? What advantages does it offer? Where can it be used? And how?

Before the invention of modern solderless breadboards, designing and testing any given electronic circuit was an aggravating,



tedious, and time consuming task. First a circuit would have to be designed on paper. Then the schematic diagram of the circuit would have to be translated into a circuit board parts layout for either point to point or printed circuit wiring. If a printed circuit were to be used, as was most often the case, the circuit layout would have to be transferred to a copper-clad board, the copper selectively etched, holes drilled, and components soldered in place. Then, if a component proved the wrong value, it would have to be desoldered and a new one soldered in place. If the printed pattern were in error, a whole new board would have to be designed, etched, drilled, filled and soldered.

When A P Products came up with the idea of arranging a breadboard with a matrix of interconnected holes, the process was simplified. The interconnections are made by conductive spring clips that grip each component lead firmly to establish a good electrical connection without soldering. The matrix of holes was placed on a tenth inch (.254 cm) spacing pattern to conform with standard component lead spacing.

The interconnection pattern was designed to provide ample access to each lead of each component, especially with modern transistor and integrated circuitry in mind. And distribution strips were designed to provide power and signal lines where needed.

Circuit design testing now becomes a matter of plugging in components and wires. Integrated circuits and discrete components plug into the solderless breadboard and ordinary 22 gauge solid wire jumpers are used to interconnect them.

A given circuit can now be prototyped in minutes rather than hours or days. Many designers work directly with component specification sheets, many with schematic diagrams. Changes in parts values are as easy as pulling out one part and plugging in another. And the geometry of the modern solderless breadboard translates into a printed circuit layout readily once the circuit is ready to commit to hardware.

In addition, solderless breadboards can serve as a basis for semipermanent circuits in applications where the need for a given circuit requires reliability but does not require longevity.

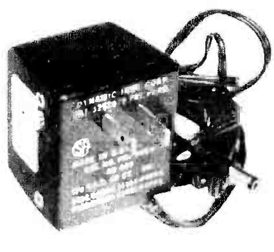
Applications for modern solderless breadboards are as wide as all of electronics. There are professional applications in machine control, data processing, test and measurement, device testing, prototyping and equipment adjunctive aids. There are hobby applications ranging from communications to photography to automotives to biofeedback to music to model railroading and more. And, of course, solderless breadboards are perfect for educational and instructional applications.

Solderless breadboards and breadboarding aids come in many sizes and prices, and have been used in designs as simple as a logic probe or as complicated as a small computer.

A P Products has available a free catalog of their ACE All Circuit Evaluator solderless breadboards, Super Strips™, terminal and distribution strips, integrated circuit test clips and accessories.■

BYTE'S BUGS

On page 68 of the August issue, there is a typographical error in the box "A Note About Construction." The Motorola part number referenced at two places in the box should be "MCM6571L" not "MCM6517L" as printed.■



UNIVERSAL POWER SUPPLY

A unique plug-in supply by Panasonic. Useful for calculators, small radios, charging many & various small NiCad batteries. Adjustment screw plug on the side changes output voltage to 4½, 6, 7½, or 9 volts DC at 100 MA. Output cord with plug, 6 ft long.

No. SP-143C $4.50 3/$12

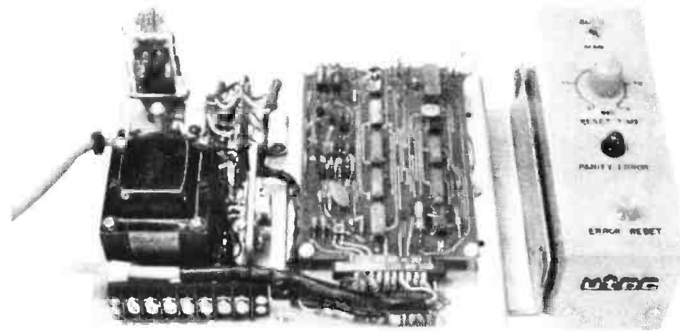
POWER SUPPLY

LAMBDA 5VDC 74 AMP

LV-EE-5-OV **$125.00**

NJE 5/OUP-D5

5 VDC 32 AMP **$75.00**



PARITY DETECTOR

New packaged, made for RCA, detects even or odd parity, baud rate 110, 150 or 134.46. Built-in logic supply for the IC's, operates from standard 115 vac. Control panel allows manual or automatic reset mode of operation. Aluminum enclosure (not shown), covers the electronics. TTY compatible.

Ship wgt. 10 lbs. $16.50

CLOCK KIT $14.00

Includes all parts with MM5316 chip, etched & drilled PC board, transformer, everything except case.

SP-284 $14.00 each 2/$25.00

COMPUTER DISPLAY TUBE

New Sylvania 9 inch CRT, 85 degree deflection, with tinted faceplate. Same as used in Viatron systems (buy a spare). With complete specs.

Ship wgt. 5 lbs. $15.00

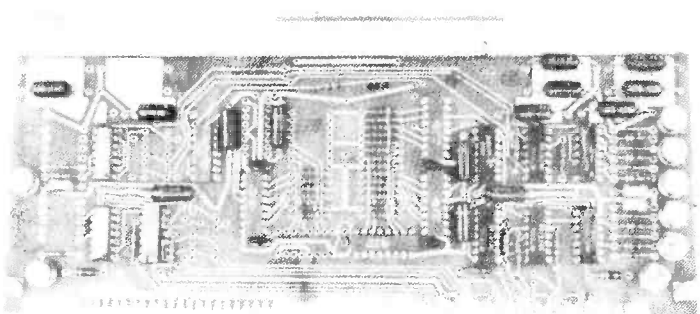
LINEAR by RCA, brand new, gold bond process

301	$ .60	747	$ .82	MM5314	$3.00
307	.52	748	.50	MM5316	3.00
324	1.80	1458	.96	7001	8.00
339A	1.60	3401	.80		
741	.50	555 timer	.60		

MEMORY SYSTEM $125.00

New memory system by Honeywell, small . . . measures only 9x4x1 inches. 1024 core memory, 1024 words with 8,9,10 bits/word. Random access, with all logic, register, timing, control, core select and sense functions in one package. New, booklet of schematics and data. Looks like a good beginning for a mini-computer. Limited supply on hand.

Ship wgt 3 lbs. #SP-79 . . . . $125.00



DATA ENTRY AUDIO TERMINAL

Sends and receives hard copy or audio of touch tone data sent & received. Output 600 ohms for phone lines. ITT touch phone pad with oscillators. Hard copy by strip printer 5 characters per inch, 35 digits visually displayed at once. Prints & receives touch tone codes, digits only. Power supply etc. all in the one case. With two units you can send and receive with monitoring of visual & audio tone at both stations.

Original cost $1065 Ship wgt. 25 lbs. $49.00 each

Meshna

Please add shipping cost on above.

PHONE 617–595-2275 FREE CATALOG SP-8 NOW READY

MESHNA PO Bx 62 E. Lynn Mass. 01904

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
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

CLOCK CHIPS — CALCULATOR CHIPS

MM5309	6 Digit, BCD Outputs, Reset PIN.	$5.95
MM5311	6 Digit, BCD Outputs, 12 or 24 Hour	4.95
MM5312	4 Digit, BCD Outputs, 1 PPS Output	4.95
MM5314	6 Digit, 12 or 24 Hour, 50 or 60 Hz	4.95
MM5316	4 Digit, Alarm, 1PPS Output	6.95
MM5318	Video Clock Chip, For Use With MM5841	9.95
CT7001	6 Digit, Calander, Alarm, 12 or 24 Hour	6.95

CALCULATOR CHIPS

MM5725	6 Digit, Four Function, Less Decimal	$2.95
MM5738	8 Digit, 5 Function, +, =, ×, ÷, %	2.95
MM5739	8 Digit, 4 Function, Floating Decimal	2.95
CT5030	12 Digit, 4 Function and %	7.95

MISC. MOS

MM5320	TV Camera Sync. Generator	$19.95
MM5330	4½ Digit DVM Chip	9.95
MM5369	60 Hz Timebase Circuit From 3.58 MHz	3.95
MM5841	Video Generator for MM5318	18.00
MC1408-L7	7 Bit Digital to Analog Converter	9.95
MK5007	4 Decade Counter with Latches	10.95
LD110-LD111	3½ Digit DVM Chip Set	25.00
95H90	100 MHz ÷ 10 Counter For Prescalers	13.95

THE NEW RCA CA3140
MOST USEFUL OP AMP SINCE THE 741
CA3140 vs. 741 at a glance

	CA3140T,S	CA741CT,S
Input Resistance R_I (MΩ)	1,500,000	2
Input Current I_I (pA)	10	80,000
Slew Rate, SR (closed loop) (V/μs)	9	0.5
Gain-Bandwidth Product, f_T (MHz)	4.5	1.0

$1.25

DIP SWITCH

These switches feature seven SPST slide switches in a molded dip. They are ideally suited for microprocessor applications **$1.95**

VECTOR WIRING PENCIL

SPECIAL SPECIAL

Vector Wiring Pencil P173 consists of a hand held featherweight (under one ounce) tool which is used to guide and wrap insulated wire, fed off a self-contained replaceable bobbin onto component leads or terminals installed on pre-punched "P" Pattern Vectorbord. Connections between the wrapped wire and component leads, pads or terminals are made by soldering. Complete with 250 FT of red wire. **$7.95**

REPLACEMENT WIRE — BOBBINS FOR WIRING PENCIL

W36-3-A-Pkg. 3	250 ft. 36 AWG GREEN	$1.95
W36-3-B-Pkg. 3	250 ft. 36 AWG RED	$1.95
W36-3-C-Pkg. 3	250 ft. 36 AWG CLEAR	$1.95
W36-3-D-Pkg. 3	250 ft. 36 AWG BLUE	$1.95

1/16 VECTOR BOARD

0.1" Hole Spacing — P-Pattern

	Part No.	L	W	Price 1	Price 2-Up
PHENOLIC	64P44 062XXXP	4.50	6.50	1.72	1.54
	169P44 02XXXP	4.50	17.00	3.69	3.32
EPOXY GLASS	64P44 062	4.50	6.50	2.07	1.86
	84P44 062	4.50	8.50	2.56	2.31
	169P44 062	4.50	17.00	5.04	4.53
	169P84 062	8.50	17.00	9.23	8.26
EPOXY GLASS COPPER CLAD	169P44 062C1	4.50	17.00	6.80	6.12

VECTOR TERMINALS

Series	Part No., Finish: Gold	Part No., Finish: Tinned	Post Size	25 pcs Gold	25 pcs Tin	50 pcs Gold	50 pcs Tin
T44	T44-1	T44	.025 sq	1.50 lot	1.00 lot	2.75 lot	1.75 lot
T46	T46-1	T46	.028 sq	1.75 lot	1.25 lot	3.00 lot	2.25 lot
T49	T49A	T49	.025 sq	1.75 lot	1.25 lot	3.10 lot	2.25 lot

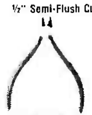
T42-1 PKG. 100 Terminals $3.50 C $13.00M

NIBBLING TOOL

Nibbling Tool $6.95
Replacement Punch $3.75 Each

DIAGONAL CUTTER

½" Semi-Flush Cut Fine Diagonal
14



Light Blue Handle
A97MS $8.50 ea.

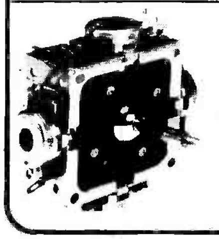
64 KEY KEYBOARD

This keyboard features 64 unencoded SPST keys, unattached to any kind of P.C.B. A very solid molded plastic 13" x 4" base suits most applications

$19.95

HD0165 16 LINE TO FOUR BIT PARALLEL KEYBOARD ENCODER $7.95

JOYSTICK



These joysticks feature four 100K potentiometers, that vary resistance proportional to the angle of the stick. Sturdy metal construction with plastics components only at the movable joint. Perfect for electronic games and instrumentation.

$9.95 ea.

MICROPROCESSOR COMPONENTS

8080 SUPPORT DEVICES

8080A $34.95

8212	8 BIT INPUT/OUTPUT PORT FOR 8080	$ 5.95
8216	NON INTERRUPT BI-DIRECTIONAL BUS DRIVER	7.95
8224	CLOCK GENERATOR AND DRIVER FOR 8080	12.95
8228	SYSTEM CONTROLLER AND BUS DRIVER FOR 8080	12.95

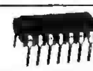
8080 $24.95

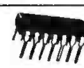
CPU'S			RAM'S			
8008	8 BIT CPU	$19.95	1101	256 x 1	Static	$ 2.25
8080	Super 8008	24.95	1103	102 x 1	Dynamic	2.95
8080A	Super 8008	34.95	2101	256 x 1	Static	6.95
SR'S			2102	1024 x 1	Static	1.95
2504	1024 Dynamic	$ 9.00	2107	4096 x 1	Dynamic	9.95
2518	Hex 32 BIT	7.00	2111	256 x 4	Static	7.95
2519	Hex 40 BIT	4.00	7010	1024 x 1	MNOS	29.95
2524	512 Dynamic	2.95	7489	16 x 4	Static	2.49
2525	1024 Dynamic	6.00	8101	256 x 4	Static	7.95
2527	Dual 256 BIT	3.95	8111	256 x 4	Static	7.95
2529	Dual 512 BIT	4.00	8599	16 x 4	Static	3.49
2532	Quad 80 BIT	3.95	91L02	1024 x 1	Static	2.49
2533	1024 Static	7.95	74200	256 x 1	Static	6.95
3341	Fifo	6.95	93410	256 x 1	Static	2.95
74LS670	16 x 4 Reg.	3.95	MM5262	2K x 1	Dynamic	.99
UART'S			**PROMS**			
AY-5-1013	30K Baud	$5.95	1702A	2048	Famos	$15.95
ROM'S			5203	2048	Famos	14.95
2513	Char. Gen.	$11.00	82S23	32 x 8	Open C.	5.00
2516	Char. Gen.	13.00	82S123	32 x 8	Tristate	5.00
74S387	1024-Bit Programmable	1.95	74S287	1024	Static	7.95
			3601	256 x 4	Fast	5.00

BIPOLAR PROM SPECIAL

6300-1 (70 NS) 256 x 4 OPEN COLLECTOR BIPOLAR PROM (EQUIVALENT TO 82S126) **$2.95**
6301-1 (70 NS) 256 x 4 TRI STATE BIPOLAR PROM (EQUIVALENT TO 82S129) **$2.95**

Special Requested Items





RC4194	Dual Track V Reg	$ 5.95	N8T97	$ 2.00	MK5007	$10.95	MC4044	$ 4.50
RC4195	±15V Track Reg	3.25	4024P	2.25	8263	5.95	LM3909	1.25
F9368	Decoder	3.95	DM8130	3.25	8267	2.75	MM5320	19.95
LD110/111	DVM Chip Set	25.00	CD4520	$2.50	8288	1.15	4072AE	.45
CA3130	Super CMOS Op Amp	1.39	MC14016	.56	8826	3.00	7422	1.50
MC1408L7	D/A	9.95	2525	6.00	8880	.80	7497	4.00
F3341	FIFO	6.95	2527	3.95	4511AE	2.50	74186	5.00
MM5841	Character Gen.	18.00	CD4518	2.50	XR4136	2.00	74279	.90
AY5-9100	Push Button Dialer	17.50	MM5309	5.95	4566AE	3.00	82S90	4.00
CDP1802	CPU Microprocessor	39.95	DM8131	4.00	ICM 7205	29.95	MCT-2E	2.95

Continental Specialties

PROTO POWER!

POWER FOR THE PROFESSIONAL

For the economy-minded experimenter . . .

Made in USA
Patents Pending

$75 Proto Board 203

Breadboard Prototesting with 5 Volt, 1 AMP Regulated Power Supply Included!

A total ready to use power breadboard prototest device with a built in regulated, short proof power supply. Just plug in and start building! 2 extra floating 5-way binding posts for external signals. Self-contained with power switch indicator lamp and power fuse. 24 14 pin DIP capacity. Attractive two-tone quality case. All metal construction 9½" L x 6½" W x 2¾" H. 6 lbs. Order today!

NEW

Proto Board 100

$17.95

Here's a low cost, big 10 IC capacity breadboard kit with all the quality of QT Sockets and the best of the Proto Board series . . . complete down to the last nut, bolt and screw. Includes 2 QT-35S Sockets, 1 QT-35B Bus Strip, 2 5-way binding posts, 4 rubber feet, screws, nuts, bolts and easy assembly instructions.

THE MINI-BREADBOARD BUDGET KIT

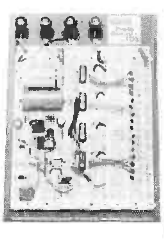
Proto Board 101. Compact, inexpensive 10-14 pin DIP capacity 5.8" long x 4.5" wide **29.95**

Proto Board 102. Compact 17-14 pin DIP capacity 7" x 4½" **39.95**

Proto Board 103. 2,250 solderless tie points. (4) 5-way binding posts. 24-14 pin DIP capacity 9" x 6" **59.95**

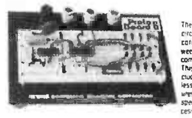
Proto Board 104. Huge 3,060 solderless tie points. (4) 5-way binding posts. 32-14 pin DIP capacity 9½" long x 8" wide **79.95**

COMPLETE KIT . . .



DESIGN MATE 1

CIRCUIT DESIGNER
You asked for it. HERE IT IS! CSC's basic circuit designer to help you build and test ANY electronic circuit. Solid #22 AWG wire interconnects any discreet components without solder. Resistors, Transistors, Linear Digital ICs in TO-5 DIP packages (from 8-40 pins). And more. Components fit directly into Sockets and Bus strips instantly and easily. Plus DESIGN MATE ONE's (DM-1) variable regulated power supply gives 5-15VDC up to 600ma (9 watts). Now that's design flexibility for you!
You can even monitor the DM-1's internal power supply or external circuits thanks to the built-in 0-15V voltmeter. If you're into electronics, DM-1 is a must for your laboratory. It comes to you completely wired, tested and ready to use for only $54.95



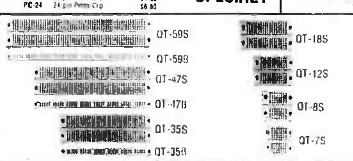
PROTO BOARD 6
The PB-6 lets the user test and build circuits without soldering or patch cords. All interconnections between components are made with common #22 AWG hook up wire. This quality breadboarding kit includes 630 component tie points at less than 2.5 cents each. It measures 6" long by 4" wide. Designed specially to Breadboard Microprocessor Circuits $15.95

Now trace signals or troubleshoot fast. Insert signals or wire unused circuits into existing boards. Scope probes, test leads lock onto Dynagrip head. Plastic construction — no springs or pivots. Thousands of operations. Narrow throat for high density PC boards.

PC-14	14-pin Proto Clip	$3.60 ea
PC-16	16-pin Proto Clip	$3.80
PC-24	24-pin Proto Clip	$6.80

SPECIAL!

LOGIC MONITOR
Simultaneously displays static and dynamic logic states of DTL, TTL, HTL or CMOS DIP ICs.
Pocket size. $84.95.



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

DIGITAL CLOCK KIT — 3½ INCH DIGITS

This clock features big 3½" high digits for viewing in offices, auditoriums, etc. Each digit is formed by 31 bright 0.2" LED's. The clock operates from 117 VAC, has either 12 or 24 hr. operation. The 6 digit version is 27" x 3½" x 1½", and the 4 digit is 18" x 3½" x 1½". Kits come complete with all components, case and transformer.

4 DIGIT KIT $49.95 — 4 DIGIT ASSEMBLED $59.95
6 DIGIT KIT $69.95 — 6 DIGIT ASSEMBLED $79.95

Specify 12 Or 24 Hr. When Ordering

Satisfaction Guaranteed. $5.00 Min. Order. U.S. Funds.
California Residents — Add 6% Sales Tax — Data Sheets 25c each
Send a 13c Stamp (postage) for a FREE 1976 Catalog

JAMES

1021 HOWARD AVE., SAN CARLOS, CA. 94070
PHONE ORDERS — (415) 592-8097

5 FUNCTION ELECTRONIC CALCULATOR RADOFIN MODEL 8P

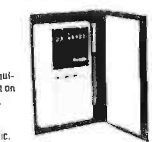
$8.95

FEATURES:

- 8 Digit Display
- 5 Functions consists of addition, subtraction, multiplication, division, percentage; with constant on all functions, with full floating decimal point.
- Power source is 1 piece 9V DC Battery 006P, jack for AC adapter
- Black superfine grained finish plastic cabinet

5 FUNCTION ELECTRONIC CALCULATOR WITH WALLET-NOTEBOOK AND POCKET CHECKBOOK RADOFIN MODEL 1710

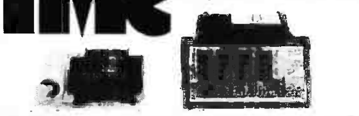
$19.95



FEATURES:

- 8 Digit Display
- 5 Functions consists of addition, subtraction, multiplication, division, percentage; with constant on all functions, with full floating decimal point.
- Power source is 6 AAA cells 9 V DC
- Wallet is 2 tone texon; cabinet is black plastic.

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 $44.95
B. 5V Power Supply $14.95

JE700 CLOCK

12 or 24 Hour

115 VAC

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 1". It utilizes a MAN72 high brightness readout, and the MM5314 clock chip.

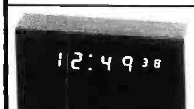
$17.95

Liquid Crystal Temperature Display

Six Digit Light Emitting Diode Display

DL 33 Display

This clock makes an attractive addition to any desk. It has an extruded, black anodized aluminum case. It displays hours, minutes, and seconds with .11 inch displays, and comes complete with a liquid crystal thermometer. It operates off 115 VAC at 50 or 60 Hz. **$24.95**



This large digit clock (.6" hours & minutes, .3" seconds) features the MM5314 clock chip. It operates from 117 VAC, and will operate in either a 12 or 24 hour mode. The clock is complete with a walnut grain case, and has fast set, slow set, and hold time set features.

KIT - ALL COMPONENTS & CASE $34.95
WIRED & ASSEMBLED $39.95

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: (H) - 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**

exelar

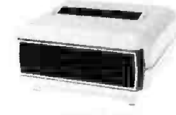
This watch is manufactured by National Semiconductor. It provides 5 functions; hours, minutes, seconds, date, A.M. Indicator dot. Accuracy is assured to 5 seconds per month by precision quartz crystal. If something should go wrong with the watch, repair is assured within 48 hours after it is received. Complete with steel black leather band.

ES4-YS
3 MICRON GOLD PLATE BEZEL
$29.95

DIGITAL ALARM CLOCK

Novus

This 4 digit Novus Alarm Clock is a very reliable and smartly styled unit. It provides such features as an alarm settable to any minute of the day, a 7 minutes snooze alarm, a power failure indicator, and even an A.M., P.M. indicator.



$17.95 NOT A KIT

FAIRCHILD ANNOUNCES THE FAIRCHILD

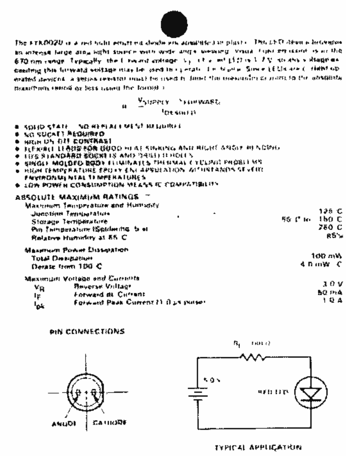
SOLID STATE TECHNOLOGY KIT

- FOR THE EXPERIMENTER WITH TASTE FOR "STATE OF THE ART" PRODUCTS
- COMPLETE SPECIFICATIONS ARE PRINTED ON THE BACK OF EACH TECHNOLOGY KIT

FTK0020

CARD FRONT

FTK0020



CARD BACK

FTK0011

CARD FRONT

DIGITS

FTK0001	0.5" High Common Cathode Digit	$1.00
FTK0002	0.5" High Common Anode Digit	1.00
FTK0003	.357" High Common Cathode Digit	.75
FTK0004	0.8" High Common Cathode Digit	2.00
FTK0005	0.8" High Common Anode Digit	2.00

0.8" HIGH DISPLAY ARRAYS

FTK0010	12 Hour, 3½ Digit Clock Display	7.00
FTK0011	24 Hour, 4 Digit Clock Display	8.00

LED LAMPS

FTK0020	10 Red LED Lamps	1.00
FTK0021	5 Mixed Colored LED Lamps	1.00
FTK0022	10 LED Mounting Clips	1.00
FTK0023	5 Three Piece LED Mounting Adapters	1.00

PHOTO TRANSISTORS

FTK0030	5 Flat Lens Photo Transistors	1.00
FTK0031	5 Round Lens Photo Transistors	1.00
FTK0032	3 Flat Lens Photo Darlingtons	1.00
FTK0033	3 Round Lens Photo Darlingtons	1.00

PHOTO ARRAYS

FTK0040	9-Element Tape Reader Array	16.00
FTK0041	12-Element Card Reader Array	24.00
FTK0042	Reflective Opto Coupler	4.00

COUPLERS

FTK0050	3 General Purpose Opto Couplers	1.00
FTK0051	Darlington Opto Coupler	1.00

MOS CLOCK CIRCUITS

FTK0400	Digital Clock/Calendar Circuit (FCM7001)	7.00
FTK0401	Digital Clock/Calendar with BCD Outputs (FCM7002)	7.00
FTK0402	Direct Drive Digital Clock Circuit with AC Output (FCM3817A)	5.00
FTK0403	Direct Drive Digital Clock Circuit with DC Output (FCM3817D)	5.00
FTK0405	Direct Drive Digital Clock/Calendar Circuit (FCM7015)	6.00

KITS

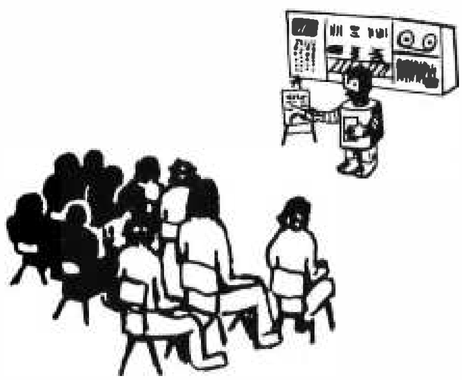
FTK0106	Automobile Clock Kit	40.00

- THESE PRODUCTS ARE PACKAGED FOR OUTSTANDING WALL DISPLAY APPEARANCE
- FULL FAIRCHILD PRODUCT LINE TO FOLLOW
- DEALER'S AND WHOLESALER'S INQUIRIES INVITED—PRICE LIST AVAILABLE.
- BUY WITH PRIDE THE PRODUCTS BUILT BY THE INDUSTRY'S LEADER — FAIRCHILD

Satisfaction Guaranteed. $5.00 Min. Order. U.S. Funds.
California Residents — Add 6% Sales Tax
Send a 13c Stamp (postage) for a FREE 1976 Catalog.

JAMES

1021 HOWARD STREET
SAN CARLOS, CA 94070
PHONE ORDERS — (415) 592-8097



Clubs and Newsletters

LICA of NY

In issue number 5 of STACK, the newsletter of the Long Island Computer Association, is a department heading "LICA, Where Goest Thou?", written by Albert L deGraffenried. It describes the double-barreled reward of exchange of information, and comradery, which is also found in other types of clubs, as well as hidden assets, such as the easy crossing of age and credentials boundaries. There are group benefits too, both given and received, and Al cites what radio hams have contributed to the community in times of disaster. This leads to what a computer club has: "group power" – the unique ability to solve *numerically* large problems which may have substantial social impact and perhaps dollar returns for the club.

Al may have hit upon a really good idea. Maybe your particular community has a need that your club or group could fill. This could be an area to explore. We'll be looking forward to seeing what progress LICA makes in this direction, in future issues.

Morris Balamut is editor of STACK, POB 864, Jamaica NY 11431.

AMRAD Computerfest

The Amateur Radio Research and Development Corp is sponsoring a Computerfest to be held on October 24 1976 at the Vienna Community Center, 120 Cherry St, Vienna VA, near Exit 11S of the Washington DC Beltway. The exposition will be almost entirely devoted to small computers of the type suitable for home use. There will be displays of microcomputer systems by various manufacturers' reps, as well as tables for used or surplus equipment, circuit boards and parts. Admission will be $4 at the door, $3.50 advance registration by mail. Make checks payable to AMRAD. Write: COMPUTERFEST, POB 682, McLean VA 22101.

AMRAD is a nonprofit club. The members for the most part have amateur radio licenses, although two thirds of their activities are now focused on microcomputer subjects.

Denver Amateur Computer Society – DACS

DACS has informed us of the appointment of Jim Clark as the new editor of the DACS newsletter. They would like to set up an exchange of newsletters with other clubs. Send your newsletters to Jim Clark, 538 So Swadley St, Denver CO 80228.

Tektronix 4051 Users Group

Vic Kley, POB 2117, Berkeley CA 94702, is an enthusiastic user of the Tektronix 4051 scientific computer. This is a machine which has a desk top package including keyboard and storage tube vector graphics display, as well as built-in mass storage and BASIC interpreter. Its internal architecture is that of a Motorola 6800, and it interfaces for scientific use to the IEEE 488 parallel bus standard (the Hewlett Packard instrumentation bus).

To date, there has been no organized user's group for this machine. Vic Kley would like to remedy that situation. Any persons interested in the concept of an independent 4051 users group should contact him at the above address.

CENOACA

Lee Lilly of the Central Oklahoma Amateur Computing Association issues a newsletter called *NEWSBITS* and offers to establish a regular exchange of newsletters with other clubs. Write: CENOACA, Box 2213, Norman OK 13069. Anyone interested in joining the club? They meet on the second Saturday of each month at 10 AM in the Oklahoma City Warr Acres

S. D. SALES CO.

P. O. BOX 28810-B
DALLAS, TEXAS 75228

4K LOW POWER RAM BOARD KIT

THE WHOLE WORKS

Imsai and Altair 8080 plug in compatible. Uses low power static 21L02—1 500 ns. RAM's, which are included. Fully buffered, drastically reduced power consumption, on board regulated, all sockets and parts included. Premium quality plated thru PC Board.

$89.95

8 Digit LED "METRIC MASTER" **$19.95**

"RAPID MAN — 12" **$29.95**

$1,000,000 CALCULATOR PURCHASE!

We bought the entire stock of a major manufacturer. New, guaranteed units.

Five functions PLUS complete Metric Conversion functions. Rechargeable batteries. Small, hand held size. With AC charger.

12 Digit - Desk Top Style. Sturdy design. With memory and four complete functions. Big, bright display.

ALARM CLOCK KIT SIX DIGIT LED

Thousands of hobbyists have bought and built our original clock kit and were completely satisfied. But we have received many requests for an alarm clock kit with the same value and quality that you have come to expect from S. D. So, here it is!

THE KIT INCLUDES:
- 1 Mostek 50252 Alarm Clock Chip
- 6 Hewlett Packard .30 in. common cathode readouts
- 15 NPN Driver Transistors
- 2 Switches for time set
- 2 Slide Switches for alarm set and enable
- 1 Filter Cap
- 4 IN4002 Rectifiers
- 1 IN914 Diode
- 1 .01 Disc Cap
- 15 Resistors
- 1 Speaker for alarm
- 1 LED lamp for PM indicator

$9.95

PCB — $3.00
XFMR — $1.50

60 HZ CRYSTAL TIME BASE — FOR DIGITAL CLOCKS S. D. SALES EXCLUSIVE!

$5.95

KIT FEATURES:
A. 60 hz output with accuracy comparable to a digital watch
B. Directly interfaces with all MOS Clock chips
C. Super low power consumption (1.5 Ma typ.)
D. Uses latest MOS 17 stage divider IC
E. Eliminates forever the problem of AC line glitches
F. Perfect for cars, boats, campers, or even for portable clocks at ham field days. BUY TWO FOR $10.00!
G. Small size, can be used in existing enclosures.
KIT INCLUDES CRYSTAL, DIVIDER IC, PC BOARD PLUS ALL OTHER NECESSARY PARTS AND SPECS.

MOTOROLA RTL IC'S

Brand new, factory prime. Hard to find, but still used in a variety of projects. (See the RTL Cookbook by Howard W. Sams.)

MC724P — 59c	MC780P — 89C	MC791P — 69c
MC725P — 59c	MC785P — 49c	MC792P — 59c
MC764P — 49c	MC787P — 89c	MC799P — 59c
MC767P — 69c	MC788P — 49c	MC970P — 89c
MC771P — 49c	MC789P — 59c	MC9709P — 69c
MC775P — 89c	MC790P — 89c	MC9760P — 69c

TTL INTEGRATED CIRCUITS

7400 — 19c	7430 — 19c	7476 — 35c	74153 — 75c
7402 — 19c	7432 — 34c	7480 — 49c	74154 — 1.00
74L04 — 29c	7437 — 39c	7483 — 95c	74157 — 75c
74S04 — 44c	7438 — 39c	7485 — 95c	74161 — 95c
7404 — 19c	7440 — 19c	7586 — 45c	74164 — 1.10
7406 — 29c	7447 — 85c	7490 — 65c	74165 — 1.10
7408 — 19c	7448 — 85c	7492 — 75c	74174 — 95c
7410 — 19c	7451 — 19c	7495 — 75c	74181 — 2.50
7411 — 29c	7453 — 19c	7496 — 89c	74191 — 1.25
7413 — 50c	7473 — 39c	74121 — 38c	74192 — 1.25
7420 — 19c	7474 — 35c	74123 — 65c	74193 — 1.00
	7575 — 35c	74141 — 75c	74195 — 69c

"CUBO" —DIGITAL ALARM CLOCK CUBE
A PERFECT GIFT — NOT A KIT!

The CUBO Alarm Clock mfg. by Corvus, division of MOSTEK CORP. Originally sold for $49.95. We bought out their entire inventory. All new, individually gift boxed. Mini size (2½" cube) with maxi performance.

FEATURES:
A. 4 Digit H.P. Bright Display
B. "Second Hand" LED Activity Indicator
C. 24 Hour Alarm; 12 hour real time format
D. 10 Minute Snooze
E. Auto Display Dimming — Adjusts to Ambient Light
F. AM/PM Indicator
G. Power Failure Indicator
Uses Famous Mostek MK50250 MOS LSI IC

$14.95 EACH (Add 75c P.&H.)

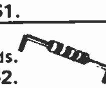
1000 MFD FILTER CAPS

Rated 35 WVDC. Upright style with P. C. leads. Most popular value for hobbyists. Compare at up to $1.19 each from franchise type electronic parts stores.
S.D. SPECIAL 4 for $1.

SLIDE SWITCH ASSORTMENT

Our best seller. Includes miniture and standard sizes, single and multi-position units. All new, first quality, name brand. Try one package and you'll reorder more. SPECIAL — 12/$1.

RESISTOR ASSORTMENT



1/4 W 5% and 10%. PC leads. A good mix of values. 200/$2.

MOTOROLA POWER DARLINGTON

Back in Stock!
Like MJ3001. NPN 80V. 10A. HFE 6000 TYP. TO—3 case. We include a free 723 C volt reg. with schematic for power supply.
SPECIAL — $1.99

WESTERN DIGITAL UART

No. TR1602B. 40 pin DIP.
This is a very powerful and popular part.
NEW — $6.95 with data
LIMITED QUANTITY

74S200

256 Bit High Speed RAM
Same as 82S16
$3.95

FAIRCHILD BIG LED READOUTS

A big .50 inch easy to read character. Now available in either common anode or common cathode. Take your pick. Super low current drain, only 5 MA per segment typical.
YOUR CHOICE: 6 for $7.50

FND — 510	Common Anode	$1.50 ea.
FND — 503	Common Cathode	$1.50 ea.

INTEL 1702A 2K ERASEABLE PROM'S

$6.95

We tell it like it is. We could have said these were factory new, but here is the straight scoop. We bought a load of new computer gear that contained a quantity of 1702A's in sockets. We carefully removed the parts verified their quality, and are offering them on one heck of a deal. First come, first served. Satisfaction guaranteed.
BACK IN STOCK!

1K PROM BACK IN STOCK!

82S129. 256X4. Bipolar, 50 NS. FAST. WITH SPECS.
$3.95

8T97B

Hex Tri-State Buffer. Back in stock.
$1.25

UP YOUR COMPUTER!
21L02—1 1K LOW POWER 500 NS STATIC RAM

TIME IS OF THE ESSENCE
And so is power. Not only are our RAM's faster than a speeding bullet but they are now very low power. We are pleased to offer prime new 21L02 — 1 low power and super fast RAM's, Allows you to STRETCH your power supply farther and at the same time keep the wait light off.
8 FOR $17.50

CALL YOUR BANK AMERICARD OR MASTER CHARGE ORDER IN ON OUR CONTINENTAL UNITED STATES TOLL FREE WATTS: **1—800—527-3460**

Texas Residents Call Collect
214/271-0022

SALE ON CUT LEAD SEMICONDUCTORS

Leads were cut for PCB insertion. Still very useable. All new, unused. Some House no.

1N914/1n4148	100/$2.
1N4002 1 Amp 100 PIV	40/$1.
1N4745A 16V 1W Zener	20/$1.
EN2222 NPN Transistor	25/$1.
EN2907 PNP Transistor	25/$1.
2N3904 NPN Driver Xstr	25/$1.
2N3392 GE Pre-amp Xstr	25/$1.
C103Y SCR. 800MA 60V	10/$1.

C&K MINI TOGGLE SWITCH

No. 7103 SUB MINI SPDT Center OFF.
SPECIAL — 99c

TERMS:

Money Back Guarantee. No COD. Texas Residents add 5% tax. Add 5% of order for postage and handling. Orders under $10. add 75c. Foreign orders: US Funds ONLY!

S. D. SALES
P. O. Box 28810-B
Dallas, Texas 75228

ORDERS OVER $15. CHOOSE $1. FREE MERCHANDISE

Branch Library at NW 63rd and MacArthur, Oklahoma City.

Atlanta Bootstrap Volume

Jim Stratigos of the Atlanta Area Microcomputer Hobbyist Club (AAMHC) has sent us the first issue of their newsletter. In an editorial they say that they will try to print items of interest such as software and hardware articles, tutorial or instruction columns, meeting notices and minutes, club activities, and hope to include a swap/buy/sell column in the near future. Anyone wishing more information concerning the club should write: Jim Stratigos, Editor, Box 33140, Atlanta GA 30332.

VCCS – Change of Address

The Ventura County Computer Society has a new address: The VCCS, POB 525, Port Hueneme CA 93041. The society meets at the Camarillo Library, Ponderosa Dr, Camarillo, the last Saturday of the month. Two exceptions are: in November, on the 20th, and in December on the 18th.

The Shift Register

Eric Rehnke, (216) 888-7531, of the Cleveland Digital Group sent along a copy of the June newsletter, volume 1:7. It included a biorhythm program, impressions of the MOS Technology KIM-1, and a number of language conversions. The CDG hopes to put out a small booklet on the differences between systems so users will not have problems converting from one basic to the next.

CACHE Register

In volume 1:5 of this Chicago area based club's newsletter, CACHE reports that they have 145 members. In attendance at the May meeting were Dr Suding of the Digital Group, as was the president, Dick Bemis. Besides showing Digital products, Dr Suding engaged CACHE members in a discussion of the hobbyist field, and of the new Z-80 from Zilog. Bill Precht continued his popular clinic on programming basics. They say that the response to this discussion was so good, the steering committee is planning a series of talks on the relative merit and drawbacks to the major chips. The club's mailing address is: POB 36, Vernon Hills IL 60061.

Personal Systems of San Diego

In the June issue of the San Diego Computer Society newsletter, a feedback sheet was included. These sheets were to be mailed back, or turned in at the next meeting. It is linked to a column called "Interaction" by Ron Eade. The column surveys the results of the previous forms, or sheets. These sheets are numbered from 0-60. 1-9 are ways you may assist the club as a whole, ie:

"I will type articles for the newsletter,"
"I will volunteer. . . .,"
"I will help. . . .,"
"I will write. . . ."

10-45 deals with SIG (special interest groups). This is divided into four subgroups, hardware, chips and systems; hardware, systems support; software, and applications. The last section deals with regional organizations meetings, ie: Coast, East County, South County. By circling the appropriate numbers, you are informing the steering committee of your willingness to participate in club activities, and you will be informed as to when and where your SIG will meet. This is probably an excellent way to exchange information in large clubs. The club's mailing address is: POB 9988, San Diego CA 92109.

Homebrew Class

The Homebrew Computer Club's newsletter has a new face. According to Robert Reiling and Joel Miller, Laurel Publications will be donating typesetting services on their computerized typesetting/text editing system as well as providing graphics, layout and editorial services. This certainly increases the appeal and readability; looks good. Bob will continue as chief editor, and Tom Pittman is the man to write to regarding the mailing list. POB 626, Mountain View CA 94042.

Triangle of North Carolina

The Triangle Amateur Computer Club meets every fourth Sunday at 2 PM at the Dreyfus Auditorium, Research Triangle Institute, Research Triangle Park, NC. They are currently exploring a club interchange standard, have a monthly newsletter, and are starting software and hardware club projects. For more information on this Raleigh, Durham, Chapel Hill area club, contact: Russell Lyday Jr at (919) 787-4137 or write Triangle Amateur Computer Club, POB 17523, Raleigh NC 27609.

A Bit of Ham from Louisiana

Emil Alline (WA5WUJ) passed a note along informing us of the formation of the "Crescent City Computer Club." The address is now Box 1097, University of New Orleans, New Orleans LA 70122. They meet at 8 PM on the second Friday of every

month in Room 2120 of the science building of the University. He also notes that the FCC and ARRL are taking steps toward making it easier than ever before to become a ham and with more privileges than ever.

Chicago Area F-8 Users

If you live in the Chicago area, Louis Burgyan is interested in meeting other F-8 users. He may be reached at (312) 327-0472, evenings.

The Computer Network of Kansas City

The *KC Thru-Put* is the organ of this Kansas City group. President Earl Day writes a column called "Day Dreams," and Bart Schwartz is editor. The group first met in May with 12 members and as of the July issue, number 2, has increased to 37. At the June meeting the possibility of making an educational program for the local PBS (Public Broadcasting Station) channel was discussed, but was deferred until the club is more established. Anyone wishing to contact this organization may write: *KC Thru-Put*, 968 Kansas Av, Kansas City KS 66105.

Microcomputer Society of Florida

In the *Marsh Data Systems Newsletter* number 5, Don Marsh writes that the Microcomputer Society of Florida has chapters in Vero Beach, Ft Myers, Sarasota, Jacksonville, Gainesville, as well as Tampa. The society will soon be publishing a newsletter of its own. They suggest that their chapters consider getting a free write-up in the local papers. Computer groups are newsworthy. In Tampa they have had a write-up in each of the major papers, and have been on television twice.

For more information contact: Marsh Data Systems, 5405B Southern Comfort Blvd, Tampa FL 33614.

New York Amateur Computer Club

Elections of club officers have been held, but as of the June newsletter, the results were not yet complete. Club member David Ripps is scheduled to speak on the topic of "Systems Monitors and Their Features." The newsletter also included a report from the technical committee, an article by Alan Yorinks, "A Short Course in Digital Logic Troubleshooting," and a humorous essay on the living room workshop by MCS III.

The address is: NYACC, c/o R Schwartz, 1E, 375 Riverside Dr, New York NY 10025.

ACGNJ News

In June at the general meeting, Marty Nichols, Tom Kirk, and Roger Amidon gave a presentation on the "String Language Processor." The high level language was running on 8080 based systems at the meeting. The demonstration was given with video monitors which were set up around the meeting room.

The bylaws were also printed, in this, the July volume 2:7 issue. A "Life for the M6800" reprint, and an article, "Interfacing the Original TV Typewriter to a Computer" by Monty Shulte, were included.

The address is: ACGNJ, UCTI, 1776 Raritan Rd, Scotch Plains NJ 07076.

Computer Hobbyist Group – North Texas

In volume 2:6 of the newsletter, they state that at the May meeting they were honored with the first look at three new products by the Southwest Technical Products Corp (SWTPC). Displayed at the meeting were a dual cassette controller/interface, a graphics controller for input to CRT, and a small but quick printer. TCHG-NT's address is: 2377 Dalworth 157, Grand Prairie TX 75050.

A Roving Computer Show for the Experimenter?

Well, not quite; but a firm called Marketing Ventures Inc, 5012 Herzel Pl, Beltsville MD 20705, is organizing a show called TECHNIHOBBY USA which will be travelling to four cities in the US this fall. The boundaries defined for the show's content are not limited to computers alone, but encompass amateur radio, radio control models, do it yourself electronics, as well. The firm is looking for participation by local clubs in its exhibitions to be held:

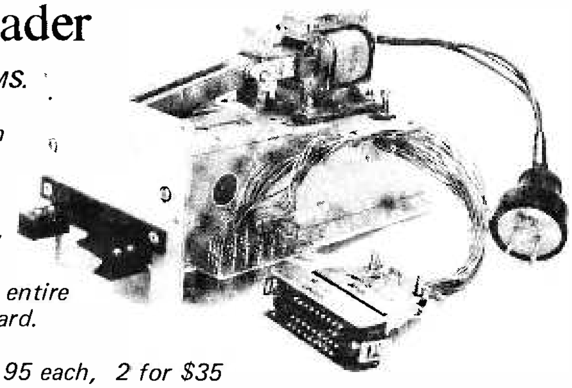
Boston	November 4, 5 and 6
Washington DC	November 12, 13 and 14
Atlanta	November 19, 20 and 21
Los Angeles	December 5, 6 and 7

For further information contact Robert E Harar, at (301) 937-7177.■

DELTA ELECTRONICS

POST OFFICE BOX 2, AMESBURY, MASSACHUSETTS 01913 Phone (617) 388-4705

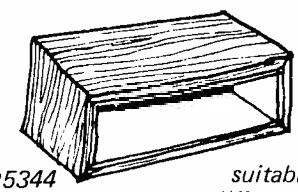
AMP Security System Card Reader



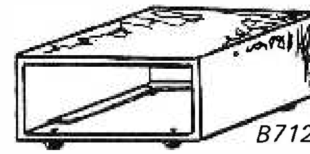
This card reader was made by AMP for use in SECURITY SYSTEMS. A stiff 2 1/8" x 3 3/8" card (credit card size) is inserted, which closes a micro-switch. A 115v AC solenoid is then energized, which pulls down a set of wipers to read through holes in the card. The wipers are arranged in 3 8 bit bytes + 1 bit, for a total of 25 bits. By turning the card over, 48 bits are possible.
This device is ideal for security systems...entry can be controlled by means of a card with an almost infinite number of combinations, rather than using an easily duplicated key. As another example, an entire Social Security number, plus an entry code, could be read from a card. 5" x 5" x 9" deep. Shipping weight 6 lbs.

STOCK NO. B5353 — $19.95 each, 2 for $35

Wood Cabinets



B5344



B7129

One of the problems with home-brew projects is finding a suitable enclosure, one that will do justice to your latest pride & joy. We have 2 wood cabinets that are suitable for small projects. B5344 has real teak veneer on all 4 sides over a ½" particleboard base. Outside dimensions are 13¾"W x 5 1/8"H x 6¾"D, with a ½" deep recess in the front. Mounting holes in the bottom. B7129 has a walnut vinyl covering over a 3/8" plywood base. Outside dimensions are 8¾"W x 4¼"H x 10¾"D. With small vent and rubber feet on the bottom.

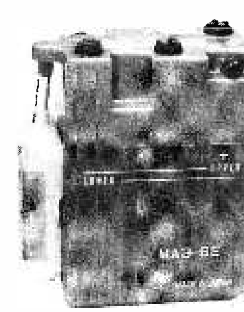
STOCK NO. B5344	*Teak cabinet*	*Shipping weight 6 lbs.*	*$7.95 each, 2/15.00*
STOCK NO. B7129	*Walnut cabinet*	*Shipping weight 3 lbs.*	*$5.95 each, 2/10.00*

Switchcraft Push Button Switch Assembly

Switchcraft series 7000 push button switch assembly, with 6 sections. 5 sections are interlocked in a 1-of-5 pattern, and the 6th button is momentary, which releases the others. Sections 4 & 5 may be depressed together; the others are excluded by means of a lockout bar. The switch comes with 12 sets of contacts: 6 SPST, normally open, 3 SPDT, and 3 DPDT, all rated at 3 amps. The contacts may be arranged in any order, up to 4 in each section. Buttons are ½" square black, on 5/8" centers. Overall size 5½" wide x 3½" deep x 1⅞" high. This versatile switch lists for well over $30 ! Shipping weight 2 lbs.

STOCK NO. B6408 — $2.50 each, 3 for $6.00

Lead-Acid Storage Battery



Lead acid storage batteries are still the cheapest & easiest to work with of all types of batteries. Two of these would be ideal for keeping the memory up in your computer when the power is off. They are rated at 6 ampere—hours, 6 volts. 2¼" x 3¾" x 5½" high. The fill holes have screw-in caps and an overflow container is provided, making this battery spill proof. We provide charging data, you provide the electrolyte. 3 lbs.

STOCK NO. B5074 — $5.95 each, 2/11.00

Power Transformers

B9397....(shown) A versatile power transformer from a line printer manufacturer. The primary is tapped for operation at 115v or 230v. There are 4 secondaries: 34v centertapped @ 3 amps, 17v @ 4 amps, 11v @ 5 amps, and 6.3v @ 1.5 amps. This would make an ideal transformer for a +5 volt and ±15 volt power supply. 3¾" x 4½" x 4¾" high. Shipping weight 10 lbs.

STOCK NO. B9397 — $12.95 each, 2 for $24

B9940....Dual 115v primaries, 3 secondaries: 17v center tapped @ 10 amps, 22v center tapped @ 5 amps, and 25v center tapped @ 7 amps. 3¾" x 4¾" x 4½" high, shipping weight 15 lbs. Another good +5v and ±15v supply transformer. **STOCK NO. B9940 — $14.95 each, 2 for $28**

Send for our latest free catalog. Minimum order $5, phone orders welcome. Include sufficient postage (2 lbs min.), excess will be refunded. BANKAMERICARD & MASTERCHARGE welcome, ALL numbers needed for processing. Minimum charge $15.

Continued from page 22

theless, amateurs and experimenters may find that a perusal of the patent literature is quite fruitful.

Howard L Grams
2616 N Salisbury St
W Lafayette IN 47906

APPLICATIONS DIGEST, MATERIALS NEEDED

I am compiling a "Microcomputer Application Digest" to be published by Howard W Sams & Co. The text is arranged by subject (Biorhythms, Electronic Music, Speech Analysis, RTTY, Terminals, Business Systems, Security, Video Art, Video Games, etc). Each section will cover several real systems, a brief tutorial on the subject, block diagrams, list of components, and names of contributing parties. Sources of additional information will also be provided.

At this point, I am seeking inputs from all interested persons who wish to share their experience to increase the exchange of information in their application area.

To keep it simple, I would like those who want to be in the book to drop me a postcard with your name, address, phone number, and a brief explanation of your application. I will then send you a form to fill out which will put the information into a standard easy to read format. This will then be collated into the book.

Not all applications need to be up and running to qualify for the text. Ideas, well thought out, are as valuable as finished systems.

If you wish our field to expand, I urge you to take time to drop me a postcard and to expound on your efforts with microcomputers—it's by spreading ideas that new ideas grow.

Mitchell Waite
H S Dakin Co
3101 Washington St
San Francisco CA 94115

NEXT OF KIM . . . .

Congratulations on an excellent magazine! I look forward to its arrival every month as each issue contains at least one article (and usually 3 or 4) that I can use immediately.

I would also like to publicly thank one of your advertisers for their fast delivery times, their prompt no-questions-asked warranty service, and their excellent newsletters. This company is MOS Technology. After months of fighting with two other computer manufacturers, I ordered a KIM-1. The KIM-1 is, in itself, a fantastic product, but with MOS Technology backing it, it is, in my opinion, the best buy on the market today!

There are presently three KIM-1 owners who work here at Eastern Washington State College. Also, the chemistry department has an Altair 8080 and the psychology department has a DEC PDP-8/F. We would like to invite anyone in the greater Spokane area (or anywhere else) who is interested in building a system or learning more about microprocessors and/or programming techniques to contact either myself of Dr R Keefer. I am a technician for the psychology department and Dr Keefer is a professor for the mathematics department. Perhaps we may have enough interested people to start a computer club or a KIM-1 users group.

Keep up the good work.

Tony Kjeldsen K7VNT
5315 N Allen Pl
Spokane WA 99208

PS I'm afraid I must agree with B L Donelan (June 1976 BYTE) concerning the December issue cover.

It's all right here

- **Prompt service**
- **In-depth inventories**

BRIGAR electronics

ELECTRONICS for INDUSTRY

LOW, LOW PRICES!

PHONE ORDERS
607 723 3111

Offices & Warehouse: 10 Alice Street, Binghamton, New York 13904 • Tel: 607-723-3111

SAVE **SAVE**

MINIATURE LAMPS

INCANDESCENT LAMPS

CHICAGO MINIATURE LAMPS #387
SAME AS #327 EXCEPT LONGER LIFE

28 volts. .04 amps. Bulb Style T-1-3/4 Mid-Flange Base. BRAND NEW BOXED, 100 per box. 40,000 pcs in stock.

PRICE SCHEDULE

1 - 99 .. 40¢ each   100 - up .. 35¢ ea.

NEON INDICATING GLOW LAMPS

NE-2 neon lamps, with dropping resistor to operate directly from 115 VAC. Nominal Watts - 1/17. T-2 Clear Bulb, wire terminals.
25,000 pcs in stock. Mfd by G.E. Co.
10¢ each up to 1000 pcs.
1000 pcs & up 7¢ each.

INDICATOR LIGHTS

BRITE GLO INCANDESCENT LAMPS

BRITE-GLO Incandescent Lamps
Ultraminiaturized Series, 100,000 hr.life Versatile, low cost indicator lamp. Ideal for Stero Amplifiers, Mobile Equipment, Instrumentation, Computer and Display Panels, Etc.
High temperature, T 1-3/4 lamps - Metal Base - Wire Terminals. Metal base increases light output and protects lamp from damage.
41, 685 pcs - 6.3 volts, 75 ma, 8" wire leads.
40,263 pcs - 6.3 volts, 200 ma, 13" wire leads.

DISCOUNT SCHEDULE

SALE   SALE

1 - 99	25¢ ea.
100-499	20¢ ea.
500-999	15¢ ea.
1000-UP	12¢ ea.

BARGAINS PRECISION RESISTORS

Type RS These should be specified when space is at a premium and when precision and power are needed. They are of the conventional type with pigtail mounting leads. They are completely protected and silicone sealed. Surpasses MIL R 26

BRAND NEW — FINEST QUALITY

QUAN	MFR	TOL. (%)	OHMS	WATTS	PRICE Each
10,686	T.I.	1	191	1/8	$.04
3,000	Electra	1	191	1/8	.04
29,000	I R C	1	243	1/8	.04
75,000	T.I.	1	2.43K	1/8	.04
35,000	T.I.	1	909	1/4	.06
7000	T.I.	1	10	1/2	.07
3000	Electra	1	15.4	1/2	.07
5700	Electra	1	28	1/2	.07
7900	I R C	1	53.6	1/2	.07
7200	I R C	1	90.9	1/8	.04
6600	I R C	1	1650	1/8	.04
1800	I R C	1	6190	1/8	.04
1000	I R C	1	2670	1/8	.04

Computer Grade Capacitors

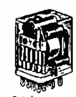
MALLORY PRICE SLASHED

CG (STANDARD) 85°C Supplied with PVC Insulating Sleeve

Quan.	Mfd.	Vdc	Size	Each
10000	1100	35	1-25/64 x 2-11/64	$.75
25000	100	350	1-25/64 x 2-11/64	.50
10000	80	400	1-25/64 x 2-11/64	.60

POTTER & BRUMFIELD RELAYS

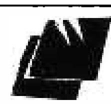
SUB-MINIATURE GENERAL-PURPOSE



TYPE KH 5505 (Identical to KHU17D11)
COIL: 24vdc, 650 ohms.
CONTACTS: 4PDT, 3amps at 30vdc or 120 vac

Small rugged DC relay with clear polycarbonate dust cover. Contacts are gold flashed silver.
Mounts by solder terminals or 3-48 stud 5/16" long. Also mounts as plug-in.
6500 pc in stock, BRAND NEW, while they last at this price. $1.25 each

LAMINATED PHENOLIC PLASTIC PANELS



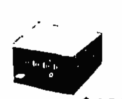
BRAND NEW PHENOLIC SHEETS
Grade LE- Natural
Type - FBE, Mil Spec: 15035, .062" thick
Needed by all experimenters, amateurs and radio men for insulating parts - terminal strips - building sets or for research & development, etc.

Stock # 560 - 6" x 6" x .062"	42¢ each
" # 561 - 6" x 12" x .062	82¢ each
" # 563 - 12" x 12" x .062	$1.58 ea
" # 564 - Full sheets 38" x 47" x .062"	$13.00 each.

BRAND NEW QUALITY SHEETS

High-Speed Reset Counter Veeder-Root

4 FIGURE COUNTER MANUAL RESET
115VDC/115VAC
VEEDER ROOT 177604

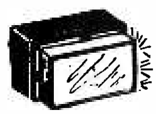


115VAC (Rectifier supplied). Basic counter is 115VDC, 6 watts 4 figure, black on white background. Push button reset. Dim.: 1-7/8" width 1-1/8" high x 2-1/2" deep.

$12.95

LIGHTED PUSHBUTTON SWITCHES

$2.95 Each



Unit can be used on 6 v or in series for 12 volt SPST, mom, PC board gold plated pins for wiring or easy installation.
10,000 pcs in stock

Mfr: Grayhill Switch Co.
2" x 1½" x 1½"

Widely used on computers, machine tools, office machines, guidance control and a wide variety of applications.

Computer Grade Capacitors

PHONE ORDERS
607 723 3111

Instant off-the-shelf delivery.

SAVE

QUAN.	MFD.	VOLTS	SIZE	PRICE EA
350	30000	5	1-3/8 x 4-3/16	$1.50
494	60000	5	3" x 4-1/2"	1.50
1016	25000	6	2-1/16" x 4-9/16"	1.50
527	40000	7	2" x 5"	1.50
353	15000	10	2" x 4-1/8"	1.25
869	15500	10	2" x 4-7/16"	1.25
277	40000	10	3" x 4-1/2"	1.70
316	74000	10	3" x 4-3/4"	2.00
375	15000	12	2" x 4-1/2"	1.25
267	14000	13	2" x 4-3/8"	1.25
600	10000	15	2-1/16" x 4-1/2"	1.25
529	12500	16	2" x 4-1/8"	1.25
252	11500	18	2" x 4-1/2"	$1.25
628	11000	19	2" x 4-1/2"	1.25
1014	8200	25	1-3/8" x 3-1/8"	1.70
10706	1100	35	1-25/64" x 2-11/64"	1.00
4200	750	60	1" x 3-1/2"	.75
792	1500	80	2" x 4-1/2"	1.30
478	450	100	1-7/16" x 2-9/16"	1.00
337	400	100	1-7/16" x 2-1/2"	1.00
200	500	150	2-1/16" x 2-3/8"	1.00
500	750	200	1-1/2" x 4"	1.00
23436	100	350	1-25/64" x 2-11/64"	.75
11800	80	400	1-25/64" x 2-11/64"	.75

Electrolytic Capacitors

TUBULAR, HIGH RELIABILITY

QUAN.	MFD	VDC	EACH
600	100	3	.30
375	250	3	.35
298	500	6	$1.00
299	1000	6	1.00
571	1200	6	1.02
375	1500	6	1.10
313	2000	6	1.11
597	100	12	.60
4000	50	15	.30
26,000	200	15	.30
416	1500	15	1.10
350	2000	15	1.10
2500	10	25	.25
1161	2000	25	1.40
2600	10	50	.40
1000	500	50	.80
825	100	50	.95
1099	150	50	.90
6549	250	50	.98
1942	300	50	1.07
10,500	60	150	.70
520	80	150	.90
290	300	150	1.00
476	40	200	1.00
409	100	250	1.39
284	200	250	1.50
255	5	350	.90
281	10	350	.90
20,000	4	250	.35
339	8	250	.70
400	16	250	.60
1055	30	350	$1.00
246	40	350	1.14
967	4	450	.98
400	10	450	1.07
529	15	450	1.07
453	20	450	1.14

METALLIZED PAPER FILM

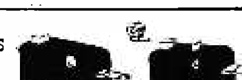
QUAN	MFD	VDC	EACH
899	.1	100	$ .35
437	.33	100	.40
400	1.0	100	.45
1360	.22	150	.40
820	.022	200	.35
4200	.033	200	.35
400	.1	200	.40
771	1.0	200	.40
417	1.5	200	.40
1100	.001	400	.35
1100	.0022	400	.40
1600	.022	400	.45
565	.047	400	.50
4000	.022	600	.25
389	.05	600	.30
500	.1	600	.30
600	.2	600	.30
717	.25	600	.30
30,000	.01	100	.15
1000	.047	100	.20
291	.22	400	.30

HIGH POWER DUMMY LOAD

NEW!

Mfg. Eastern Microwave. #10-1254...RF Coaxial Load Resistor utilizing water flow to cool and provide a load resistance of 50 ohms for inputs from DC to 4000 MHz. Maximum power input is 1000 watts. Requires a flow of .218 GPM min. per 100 watts of input power; 100 psi pressure. Compression fittings on water line inlet and outlet for copper tubing supply lines. Has a Flange type coaxial fitting, friction type, 13/16" I.D. with 2-1/4" flange. Completely shielded unit of machined aluminum. Size: 1-5/8" diameter by 5-3/8" long. Shipping weight, 2 lbs.
5,000 pcs in stock, NEW BOXED........$14.95 each......

Cherry Electrical Products



These switches are postage stamp sized precision switches which feature long operating life & high electrical capacity for their size
5 amp, 1/4 HP, 125-250 vac, SPDT, quick dis-connect terminals. 2,000 pcs in stock without lever..75¢ each
2,400 pcs in stock with extended roller lever. 95¢ ea

TANTALUM CAPACITORS

SPRAGUE

TYPE 109D

TUBULAR SINTERED-ANODE TANTALEX CAPACITOR
#109D147X9060T2
1620 pcs - 140 uf ±10% - 60vdc $2.00 ea.

GENERAL ELECTRIC PLAIN FOIL TANTALUMS +125° C (Use CL31 Instead of CL30; or, CL33 Instead of CL32 for Mylar sleeve) CL30/31—POLARIZED

3,000 pcs CL31BE180MPE, 18uf - 15vdc   65¢ each
300 pcs CL31BE550MPE, 55uf - 15vdc   85¢ each
non-polarized CL32/33
900 pcs CL33BL020MNE, 2.0uf - 75vdc $1.00 each

MALLORY MTP/MTPH ULTRA-MINIATURE TANTALUM CAPACITORS
MTP TYPES

10,550 pcs MTP685MO20P1D, 6.8uf - 20vdc   55¢ each
2,100 pcs MTP106MO30P1A, 10uf - 30vdc   75¢ each
1,028 pcs MTP336MO10P1A, 33uf - 10vdc   65¢ each

MALLORY TAS SOLID ELECTROLYTE TANTALUM CAPACITORS

4996 pcs, 4.7uf- ±10%, 6vdc, TAS475K006P1A   35¢ ea
394 pcs, .47uf ± 10%, 6vdc, TAS474K006P1A   38¢ ea
1430 pcs, 47uf ± 10%, 6vdc, TAS476K006P1C   40¢ ea
326 pcs, 220uf ± 20%, 6vdc, TAS227M006P1G $1.20 ea
285 pcs, 5.6uf ± 10%, 10vdc, TAS565K010P1C   40¢ ea
4574 pcs, 10uf ±10%, 10vdc, TAS106K010P1C   40¢ ea
900 pcs, 15uf ±20%, 10vdc, TAS156M010P1C   40¢ ea
887 pcs, .068uf ±10%, 20vdc, TAS683K020P1A   51¢ ea
152 pcs, 2.2uf ± 20%, 20vdc, Sprague   40¢ ea
1457 pcs, .47uf ±10%, 35vdc, TAS474K035P1A   51¢ ea

GENERAL ELECTRIC PLAIN FOIL TANTALUMS +125° C

365 pcs G.E. 40uf,±20%, 50vdc, 125°C
Type: 2K105AA6M. $1.75 each
1168 pcs SPRAGUE, 40uf, ±20%, 50vdc, 125°C
Type: 220D005A3M. $1.75 each   C

All Orders F.O.B. Binghamton, NY.
Minimum Order $5.00
NY State Residents please add sales tax

BRIGAR ELECTRONICS publishes a monthly flyer
If you want to be placed on our active mail list, send us your
Name ____________
Address ____________

ATTENTION PDP-8 LOVERS!

Thought I would drop a line to BYTE and say how much I enjoy the magazine. I first got into home computing in 1973. A friend down in Texas found a used PDP-8 and called me. We went down on business and at the same time stopped over to see the machine. We bought it and back home made a crate from dimensions in the manual. Next trip down we carried the crate knocked down on the plane, crated it and shipped it back. It went on the air with very little trouble and we immediately had a vast amount of software from DEC and the users group DECUS. We now have six PDP-8s in the Washington DC area with one fellow just finishing a homebrew machine with the Intersil IM6100 chip. Out of this group all the fellows are electrical engineers, none of which has a computer science background. Two of us are interfacing surplus speech synthesizers and are interested in using the 8s for signal processing and speech synthesis. One other is interested in motion picture photography and is interfacing a high resolution CRT system to his 8 and an animation camera. The PDP-8s are the straight 8 version first built about 1966 being implemented with discrete components throughout. We have spare cards and the maintenance is done by each person, although the more experienced of us give help to the others. We are looking forward to articles in BYTE on PDP-8 and IM6100 type systems; no doubt they will start with the introduction of the IM6100 kit by PCM. We are of course always glad to hear from other PDP-8 people.

Frank Gentges
3512 Orme Dr
Temple Hills MD 20031
(301) 894-2613

A SALUTE TO THE PDP-8 AND FAREWELL

This letter is an opinion in response to the article in BYTE #9, "Chip Off the Olde PDP 8/E" by Robert Nelson [*page 60*] in which he made the incredible statement, "The PDP-8 at this point may truly be *the* universal computer" and which concluded with the statement that "Many of the microprocessor chips available today were not designed to be the heart of a general purpose minicomputer . . . (but) . . . were primarily designed . . . (for industrial control applications) . . . . "

Mr Nelson either overlooks or never knew that the PDP-8 was frequently sold not as a computer but as a controller. In any case,

"The PDP-8 is an old warhorse which should be put out to pasture."

sophistication is where you find it. The 8080 is far more sophisticated than the PDP-8 so if one needs a capable computer, regardless of the imputed applications intent of the designers, the 8080 is better than the PDP-8 hands down.

The PDP-8 was truly a marvel for its time and for this DEC deserves great praise. However, the PDP-8 is an old warhorse which should be put out to pasture. A group within DEC's environs realized the PDP-8 was over the hill so they broke away from Data General and made the NOVA which was a vast improvement, conceptually, over the PDP-8. But now the original NOVA and its replacement the SUPERNOVA are both also over the hill, eclipsed if you please, by later innovations. The DEC people felt the hot winds of change blowing on their collective necks so they developed the PDP-11 (no doubt partly in reaction to the success of the NOVA which is a mighty fine little computer except for the possibility it is somewhat wasteful in its utilization of memory address space). But it is not my purpose to discuss the PDP-11.

It is fantastic that Mr Nelson claims the PDP-8 is easy to understand from a software standpoint. It is not all that difficult to understand, but he must have meant that the PDP-8 is easy to understand when compared with other computers. The PDP-8 is not a "clean" machine — it has some complications that are needlessly tedious when compared with a classic von Neuman computer such as the IBM 7090 (the 7090 was somewhat larger than the PDP-8). The so-called microinstructions are enough to drive a beginner right up the wall. The 8080 has its little complications; but, if desired, one could select a subset of the 8080 instructions [*Such as Charles Howerton used in his July 1976 BYTE article on page 22*] which if used for comparison with the PDP-8 would be both simpler and more powerful than the PDP-8. Many complications in programming have to do with addressing. Compared to the pain of addressing on a PDP-8, addressing in the 8080 could not be easier. Addressing only 4 K on the PDP-8 requires 24 bits: 12 bits for the instruction and another 12 bits for an indirect address vector. The poor little 8080 "industrial controller" directly addresses 64 K in the same 24 bits.

Mr Nelson says the PDP-8 has a "convenient parallel word length of 12 bits." Convenient for what? The only thing I can think of is a 12 bit A to D converter or reading all 12 levels of a punched data card. One hardly needs 12 bits for an operation code (8 bits allows 256 possible operations) and 12 bits is a short and limited address field. In the "olden" days, before ASCII, the 12 bit word could hold two characters at 6 bits each, but that is ancient history.

The autoindexing locations in octal 0010 to 0017 are kind of cute. The Data General NOVA extended this concept to also allow auto decrement locations, but neither the auto increment or auto decrement locations are used very much these days because no program is really quite sure they are not used by some other program which leads to the consequence that they are not used by anyone. In a sense, the NOVA was an extension of the PDP-8 in concept, but the NOVA people should have looked elsewhere for inspiration.

Now for some fundamental theory. Mr Nelson points out that instructions can be manipulated and data words can be executed, concerning which he says, "Software people will recognize this convenience." Convenience my foot — this is a bucket of worms. Before the invention of index registers and indirect addressing, this was the only method available for instruction modification. Very few modern programmers play such games with instructions because it unduly complicates on line debugging and prevents a program being written as "pure procedure." A program consists of pure procedure when it *never* changes any location that is peculiar to (is assigned to) the procedure block itself. Such a program has the virtue that it can be placed into read only memory (ROM, PROM or EROM) or protected programmable memory. A conventional PDP-8 program cannot be written as pure procedure because the return vector from a subroutine is always stored into the first location of the subroutine. This is the classic subroutine linkage and is how it was done on most computers for many years. Thus, it follows that the PDP-8 and read only memory do not get along together.

The interrupt structure of the PDP-8 (and the NOVA) is quite crude when compared with most modern computers (maxi, mini, and micro). The PDP-11 has a particularly nice vectored interrupt structure.

I think that the article by Mr Nelson is an excellent example of the smoke screen that is thrown up when discussing a machine that is quite deficient in terms of registers. He mentions the multiplexer (DX) register and the temporary register (TEMP) that are absolutely unavailable to you in any sense; but, when thrown in with the "real" registers, it seems to up the count. He also includes features common to all CPUs such as the PC, MAR, IR and the ALU which is not a register anyway.

A virtue of the PDP-8 which must be admitted is the large amount of software that has been written for it. Considering that practically none of it may be placed into ROM, this treasure trove is of little value unless you have it on disk or other fast mass storage device. Now you are beginning to talk the kind of bucks that would be better spent for a more effective computer. It is not nice for me to say it, but this vast store of freely available software contains a heck of a lot of junk that would be more trouble to understand and get working than it is worth. You obviously need systems programs like an editor, assembler, and a debug executive, but beyond these you need little more. These should properly be supplied by the manufacturer anyway!

Contrary to Mr Nelson, I say the PDP-8 or its near equal the IM6100 are not at all the "ideal machine for the computer hobbyist." If you could "fall in" and get one for nothing, then take it, but for your hard earned bucks — no way man.

Webb Simmons
1559 Alcala Pl
San Diego CA 92111

There are two sides of every coin, and we print your letter to give an example of a critique which might be made. Does anyone else have some information or opinions to contribute on the basis of personal experience with PDP-8s?

Digital Equipment Corporation is the source of a whole industry, the minicomputer industry, and the PDP-8 is historically the first successfully marketed minicomputer. (Successfully is defined here as widely sold and installed.) All present microcomputer work owes its heritage to the earlier minicomputer industry and indirectly to Digital Equipment Corporation and the PDP-8.

We caution readers that just as Robert Nelson may be a bit pro-PDP-8 in his recent two part article, the above letter represents and emphasizes the other side of the coin. An important item to remember in forming your own opinion about the matter is that virtually any machine with the characteristics of programmability can be used in the small computer systems context. If you are having a race to see who is the fastest, who uses the least memory, who has the best software development systems, etc, then differences in the design and history of a computer architecture will enter into the decision. ■

BYTE

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 inquiries not solicited. Correspond directly with company.**

BOMB

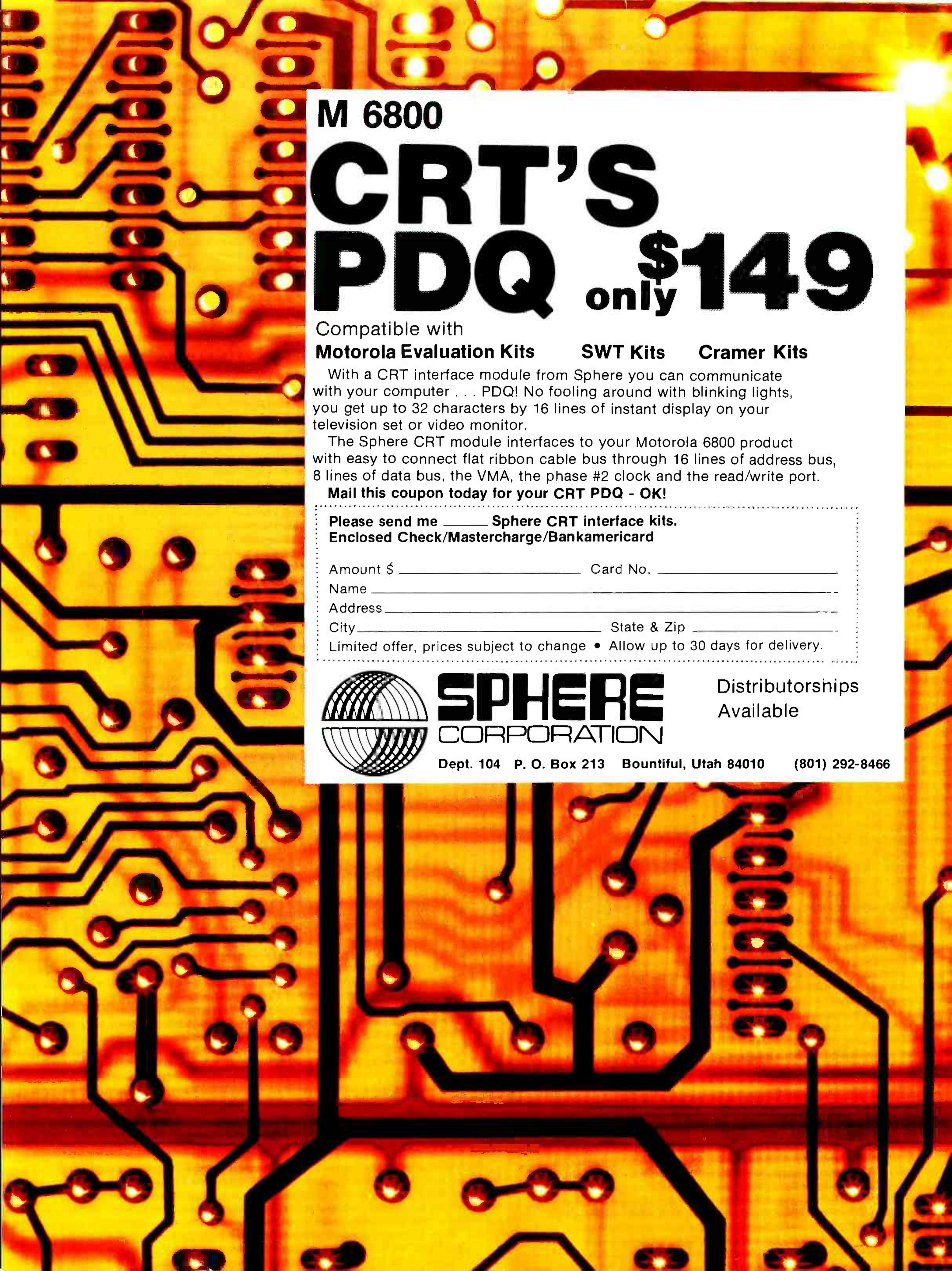
BYTE's Ongoing Monitor Box

Feedback is what keeps a linear amplifier in line. Like a linear amplifier, BYTE can use a bit of feedback. The BOMB analysis is done once a month to provide encouragement to authors and some formal feedback on how readers appreciate articles. BYTE pays the winning author a $50 bonus, so you can encourage the authors you like by voting your preferences. Remember that with few exceptions BYTE authors are just readers who have sat down at their typewriters to tell a story about what they've done or what they know about some aspect of this technology.

BOMB Results for July

Results of the BOMB survey for the July BYTE were as follows: First place winner was James R Jones, for his article "Coincident Current Ferrite Core Memories." The runners-up were tied: Bob Baker, for "Put the 'Do Everything' Chip in Your Next Design" and Richard J Lerseth for "A Plot Is Incomplete Without Characters." Who'll win the August BOMB? Find out next month. Who'll win the October BOMB? You can affect the course of events by supplying a personal evaluation of this issue's articles. Watch for the tally in January's BYTE.■

PAGE	ARTICLE	LIKED: LEAST … BEST
12	Hickey: Computer. . .vs. . .Hand Sent Morse Code	0 1 2 3 4 5 6 7 8 9 10
26	Hosking: A Ham's Application Dreams	0 1 2 3 4 5 6 7 8 9 10
30	Grappel-Hemenway: Add This 6800 Morser. . .	0 1 2 3 4 5 6 7 8 9 10
36	Krakauer: Efficient Storage. . .	0 1 2 3 4 5 6 7 8 9 10
42	Sewell: If Only Sam Morse Could See Us Now	0 1 2 3 4 5 6 7 8 9 10
52	Filgate: Morse Code Station Data Handler	0 1 2 3 4 5 6 7 8 9 10
74	Guthrie: Mathematical Function Unit–Part 2	0 1 2 3 4 5 6 7 8 9 10
82	Baker: Keep PACE with the Times	0 1 2 3 4 5 6 7 8 9 10
90	Fylstra-Wilber: Homebrewery vs Priesthood	0 1 2 3 4 5 6 7 8 9 10
118	Douds: Meeting Activities. . .	0 1 2 3 4 5 6 7 8 9 10


M 6800
CRT'S
PDQ only $149
Compatible with
Motorola Evaluation Kits SWT Kits Cramer Kits
With a CRT interface module from Sphere you can communicate with your computer . . . PDQ! No fooling around with blinking lights, you get up to 32 characters by 16 lines of instant display on your television set or video monitor.
The Sphere CRT module interfaces to your Motorola 6800 product with easy to connect flat ribbon cable bus through 16 lines of address bus, 8 lines of data bus, the VMA, the phase #2 clock and the read/write port.
Mail this coupon today for your CRT PDQ - OK!
Please send me ______ Sphere CRT interface kits.
Enclosed Check/Mastercharge/Bankamericard
Amount $ ____________ Card No. ____________
Name ____________
Address ____________
City ____________ State & Zip ____________
Limited offer, prices subject to change • Allow up to 30 days for delivery.
SPHERE CORPORATION
Distributorships Available
Dept. 104 P. O. Box 213 Bountiful, Utah 84010 (801) 292-8466

Measuring just 11" wide x 11" deep x 5" high, and weighing a mere 7 pounds, the Altair ™ 680b is a complete, general-purpose computer.

The secret to this revolutionary, small computer is its CPU board. This double-sided board fits along the bottom of the Altair case and plugs directly into the front panel board. It contains the new 6800 microprocessor, 1,024 bytes of RAM memory, a 256 byte PROM monitor, provisions for 768 bytes of additional PROM or ROM, and a single Interface port with a Motorola ACIA serial interface adapter which can be configured either RS-232 or TTY. A five level Baudot interface option is also available.

The Altair 680b can be programmed from front panel switches, or it can be interfaced to a video display terminal, or teletypewriter. Three additional circuit boards can be plugged inside the Altair 680b for further memory and interface expansion. The first of these boards is a 16K static RAM memory board.

Software already developed includes Altair 680 BASIC with all the features of the 8K BASIC previously developed for the Altair 8800. These include Boolean operators, the ability to read or write a byte from any I/O port or memory location, multiple statements per line, and the ability to interrupt program execution and then continue after the examination of variable values. This software takes only 6.8K bytes of memory space and a copy is **included free with the purchase of the Altair 680 16K memory board.**

Other software includes a resident two pass assembler. The Altair 680b is also compatible with Motorola 6800 software.

The Altair 680b is ideal for hobbyists who want a powerful computer system at an economic price. Altair 680b owners qualify for membership in the Altair Users Group, and like other Altair owners, they receive a complimentary subscription to **Computer Notes** and complete factory support.

PRICES:

Altair 680b kit with complete, easy-to-understand assembly manual, operator's manual, and programming manual $466
Assembled Altair 680b $625
Altair 680b Turnkey model kit 395
Expander Card 680MB (required to expand 680) .... $ 24
Altair 680BSM 16K static RAM board kit with 680 BASIC $685
Altair 680 BASIC when purchased separately $200
Baudot option .... $ 42

MAIL THIS COUPON TODAY

☐ Enclosed is a check for $__________
☐ BankAmericard # __________ or Master Charge # __________
☐ Altair 680b ☐ Kit ☐ Assembled ☐ *Other* (specify)
enclose $8 for postage and handling
☐ Please send free information package

NAME __________
ADDRESS __________
CITY __________ STATE & ZIP __________

mits

2450 Alamo SE/Albuquerque, NM 87106, 505-243-7821

NOTE: Altair is a trademark of MITS, Inc.

Price, specifications subject to change. Please allow up to 60 days for delivery.