'68' MICRO JOURNAL

2.00

VOLUME I ISSUE I · Devoted to the 6800 User · February 1979

SERVING THE 6800 USERS WORLDWIDE

The Software Company

Technical Systems Consultants, Inc.

TSC, Technical Systems Consultants, is **the software company** for all the newest, most innovative ideas in computer software. TSC builds a variety of programs, packages and games so you can get down to business or just some fun.

Text Editing System

The most complete and versatile editor available for the 6800 and 8080 micro. The system is line and content oriented for speed and efficiency and features such commands as block move and copy, append and overlay, as well as string manipulators. The 6800 version requires 5K beginning at 0 hex, the 8080 needs 6K starting at 1000 hex. Both should have additional file space as required.

SL68-24	6600 Text Editing System	$23.50
SL68-24C	w/cassette	$30.45
SL68-24P	w/paper tape	$31.50
SL68-24D	w/mini flex disc	$31.50
SL66-24F	w/flex disc	$50.00
SL80-10	8080 Text Editing System	$28.50
SL80-10P	w/paper tape	$37.50
SL60-10F	w/ CP/M disc	$40.00

Space Voyage

SL68-5C	w/cassette	$18.95
SL66-5P	w/paper tape	$19.00
SL80-9P	w/paper tape	$19.00

6800 Disassembler

SL68-27C	w/cassette	$15.95
SL68-27P	w/paper tape	$13.00

Micro BASIC Plus for 6800

SL68-19C	w/cassette	$22.90
SL68-19P	w/paper tape	$21.95

6800 Floating Point Package

SL-66-4	Floating Point Package	$6.50
SL68-4P	w/paper tape	$9.50

Text Processing System

As a complement to the Editor, the Processor supports over 50 commands for left, right or center justification, titling, paging and general text output formatting. A loop command is available for repeated formatting jobs such as form letters. Also included are capabilities for macro definition to build special formatting commands. The program requires about 8K of RAM and previously edited text.

SL68-29	6800 Text Processor	$32.00
SL68-29C	w/cassette	$36.95
SL68-29P	w/paper tape	$40.00
SL68-29D	w/mini flex disc	$40.00
SL66-29F	w/flex disc	$75.00
SL80-11	8080 Text Processor	$32.00
SL80-11P	w/paper tape	$41.00
SL80-11F	w/ CP/M disc	$50.00

Relocator

This self-prompting, easy to use program relocates object code in RAM or from tape. Complete instructions included for making the TSC Editor and Assembler or Editor and Text Processor co-resident. (As sold they reside in the same area.) Just over 1K in length.

SL68-28	6800 Relocator	$ 8.00
SL66-28C	w/cassette	$14.95
SL80-13	8080 Relocator	$ 8.00
SL80-13P	w/paper tape	$13.00

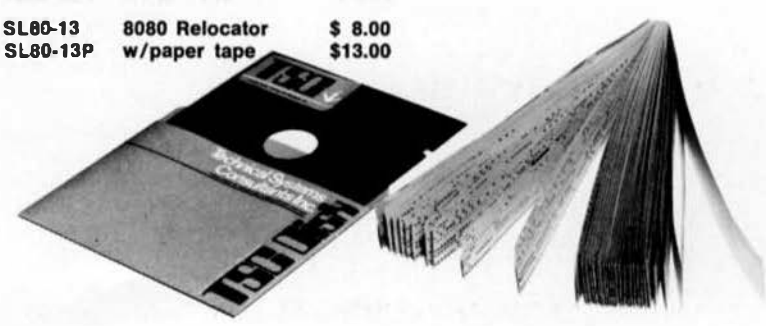
The Mnemonic Assembler

The ideal addition to the Text Editing System, together they form a complete program development center. The Assembler is one of the most versatile available and allows for easy adaptation to most systems. The Assembler is many times faster than other resident assemblers. Requires approximately 5.5K plus file and symbol table space.

SL68-26	6800 Mnemonic Assembler	$23.50
SL68-26C	w/cassette	$30.45
SL68-26P	w/paper tape	$31.50
SL68-26D	w/mini flex disc	$31.50
SL68-26F	w/flex disc	$50.00
SL80-12	8080 Mnemonic Assembler	$25.00
SL80-12P	w/paper tape	$34.00
SL80-12F	w/ CP/M disc	$40.00

All programs include complete source listing. Cassettes are in the Kansas City standard format.

All orders should include check or money order. Add 3% for postage and for orders under $10, please add $1 for handling. Send 25¢ for a complete software catalog.



TSC™ Technical Systems Consultants, Inc.
Box 2574 W. Lafayette, IN 47906
Specialists in Software & Hardware for Industry & the Hobbyist

'68' MICRO

Portions of the text of '68' Micro Journal set using the following:
6800/2, DMAF1 and CT-82
Southwest Technical Products Corp.
219 W. Rhapsody
San Antonio, TX 78216

Editor, Word Processor and Sort/Merge
Technical Systems Consultants, Inc.
Box 2574
W. LaFayette, IN 47906

Selectric I/O
World Wide Electronics, Inc.
130 Northwestern Blvd.
Nashua, NH 03060

Publisher/Editor
Don Williams Sr.

Assistant Editor — Software
Mickey E. Ferguson

Assistant Editor — Hardware
Dennis Womack

Associate Editor — Southwest
Dr. Jack Bryant

Associate Editor — Midwest
Howard Berenbon

Subscriptions and Office Manager
Joyce Williams

Typography and Color Separations
Williams Company, Inc.
Chattanooga, TN 37421

CONTENT

Publisher:
'68' Micro Journal
3018 Hamill Rd.
PO Box 849
Hixson, Tennessee 37343

'68' Micro Journal is published 12 times per year. Second class postage pending at Chattanooga, Tennessee. Address corrections are requested. Return postage guaranteed. Postmaster: Send form 3579 to '68' Micro Journal, PO Box 849, Hixson, Tn. 37343.

Subscription Rates U.S.A.:
(Special Charter Rate)
1 year $10.50
2 years $18.50
3 years $26.50
Lifetime $125.00 (one-time payment) Twice rate shown above for Air Mail/First Class anywhere in the U.S.A.

—ITEMS SUBMITTED FOR PUBLICATION—

(Letters to the Editor for Publication) All 'letters to the Editor' should be substantiated by facts. Opinions should be indicated as such. All letters must be signed. We are interested in receiving letters that will benefit or alert our readers. Praise as well as gripes is always good subject matter. Your name may be withheld upon request. If you have had a good experience with a 6800 vendor please put it in a letter. If the experience was bad put that in a letter also. Remember, if you tell us who they are then it is only fair that your name 'not' be withheld. This means that all letters published, of a critical nature, cannot have a name withheld. We will attempt to publish 'verbatim' letters that are composed using 'good taste.' We reserve the right to define (for '68' Micro) what constitutes 'good taste.'

(Articles and items submitted for publication) Please, always include your full name, address, and telephone number. Date and number all sheets. TYPE them if you can, poorly handwritten copy is sometimes the difference between go, no-go. All items should be on 8X11 inch, white paper. Most all art work will be reproduced photographically, this includes all listings, diagrams and other non-text material. All typewritten copy should be done with a NEW RIBBON. All hand drawn art should be black on white paper. Please no hand written code items over 50 bytes. Neatly typed copy will be directly reproduced. Column width should be 3¼ inches.

(Advertising) Any Classified: Maximum 20 words. All single letters and/or numbers will be considered one (1) word. No Commercial or Business Type Classified advertising. Classified ads will be published in our standard format. Classified ads $7.50 one time run, paid in advance.

Commercial and/or Business advertisers please write or phone for current rate sheet and publication lag time.

LETTERS

. . . a quick note from:

Wayne Green WUSD!

PETERBOROUGH NH 03458

Editor-Publisher

73 Magazine Radio Amateurs
Kilobaud MICROCOMPUTING
MICROCOMPUTING Industry
Instant Software

603-924-3873 6532 3883 3737

January 22, 1979

Don Williams
3018 Hamill Road
Hixson TN 37343

Dear Don,

Your letter announcing the 68 Micro was interesting and I'm anxious to know when publication will be starting. The demise of Sphere was quite a setback for the 6800 and the slowness of the SWT has not helped promote the chip as much as it has deserved. The problems with the Altair 680 also contributed, I suspect, to disfavor among hobbyists with the 6800...again unfairly.

If you run into any 6800 oriented programs which might be publishable, please keep in mind that Instant Software is anxious to support the 6800. So far, out of several hundred programs submitted for publication, only a very few have been for 6800 systems...perhaps three. As we pour forth our programs into computer stores and sell them via mail order, this means that 6800 systems will be quickly forgotten due to the preponderance of 6502, Z80 and 8080 based programs.

We're anxious, also, to find hobbyists with the equipment and time to translate TRS-80 and PET based programs for use on 6800 systems. If we can get them to publish, I'll be glad to support the 6800.

Regards....(73)...Wayne

Devoted entirely to the 6800 user

'68'

'68' Micro Journal
POB 849
3018 Hamill Road
Hixson, Tennessee 37343

February 5, 1979

Kilobaud Microcomputing

Peterborough, NH 03458

Attention: Mr. Wayne Green

Dear Wayne,

Thank you for your letter of the 22d of January. '68' Micro Journal should be out by the latter part of this month.

I can not agree with you, concerning the setback to the 6800, by the demise of Sphere. Even more strongly do I disagree, concerning your alleged slowness of the SWTPC. Not only is the SWTPC not slow, it is very dependable, more than I can say for most non-6800 machines. Especially those using the S100 bus. Down here at the 'fixer, user level', this and much more is readily apparent. It boils down to a difference between tools and toys.

The problems with the Altair 680 were in no way related to the 6800. It was poorly designed and very limited in it's utilization of the 6800. It would have failed with any microprocessor chip.

The Sphere situation was somewhat more complex. Basically the lack of software, factory support and again the poor utilization of the 6800, all contributed to the Sphere failure. As a matter of information, we have locally, two healthy Sphere systems. Both running all 6800 software and driving SWTPC MF-68 dual floppy disk systems. We have a series of articles coming on the necessary conversion, which is very simple to implement. I believe that the 6800 is so honest it could nearly run connected to a bowl of chicken noodle soup.

The SWTPC is not slow. The BASIC they furnish (at practically no cost) is slower in operation than some others. That is the major disadvantage. The reason being is that it operates in BCD and carries the precision factor to a higher degree than most of the 'flash' versions of BASIC. For me dependability, ease of programming, precision and peaceful nights of sleep are far more important. There are now and coming soon, other versions of BASIC (interpreter or compiler), written to run on 6800 machines, that equal anything available for microcomputers.

As you can tell I like the 6800. I do not believe that any scheme of selling software is about to cause the 6800 to be quickly forgotten. The level of software presently available from most software vendors is on about a par with what most 6800 users had two years ago, trivial games and so-called business software loaded from tape. Bio's and Lunar Landers are far from what is needed. Maybe someday the 'appliance machines' can do the job, but I don't know of any that can now.

In my view the thing that most hurts the 6800, and anything else, is some of the uninformed 'Doomsday Prophets', who spread their dire predictions without any basis of fact or knowledge.

Thanks again for your letter. If you are ever down this way, please drop by. The coffee is always hot and we are all in this great hobby of small(?) computers together. Maybe we could kick it around some more.

With best regards,

Don Williams Sr.
Publisher/Editor

1157 Warwick Road,
Solihull,
West Midlands B91 3HW.

'68' Journal
Williams Data-Comp Division
Hamilton Publishing Inc.
3018 Hamill Road
Hixson Tn 37343

31 October 1978.

Dear Mr. Williams,

Many thanks for your letter which I received this morning. I can only say that I am surprised that a magazine such as yours has not been started sooner. I am truly convinced that with the possible exception of the 6809, the 6800 is the best all-round processor on the market.

You may be interested to know that the 6800 is probably the most popular processor on this side of the Atlantic (at least in terms of systems where the choice of processor is of interest to the purchaser, ie. not the TRS-80)

Whilst I can only commiserate with you on the state of the dollar, it does mean that your journal is very attractively priced over here. I suggest that you approach the following who might give you some publicity:

The Amateur Computer Club,
7 Dordells,
Essex, England.

Personal Computer World Magazine,
62A Westbourne Grove,
London W2, England.

Practical Computing,
2 Duncan Terrace,
London N1, England.

I will certainly give more than a little thought to the question of an article for you, however, time is a little tight here, to the extent that my 6800 system which I started a year ago is not yet finished, let alone working. Nevertheless, I will do my best.

Best Wishes for your enterprise,

Yours sincerely,

Paul M. Jessop.

Ps. Why not have a system where contributing authors get a free subscription, to a max. of one per year per author, the results might be quite interesting and you could reduce the author's fees slightly to pay for it.

FROM NOEL J THOMPSON
HAWAII INSTITUTE OF GEOPHYSICS
2525 CORREA ROAD
HONOLULU HAWAII 96822
OCT 31 78

* TO DON WILLIAMS, PUBLISHER '68' JOURNAL

* CONGRATULATIONS ON YOUR DECISION TO PUBLISH A JOURNAL FOR US 'FORMER OUTCASTS' IN THE 6800 WORLD. WE HAVE NEEDED THIS FOR SOME TIME. SEEMS LIKE MOST EVERYBODY ELSE JUMPS ON THE 'OBSOLETE 8080' BANDWAGON.

* ENCLOSED IS 3 YEARS WORTH OF TARIF. MAY YOU LAST AT LEAST THAT LONG!

* WHEN NOT BABYSITING A HARRIS /6 OR AN HP 2000, I WORK WITH MY OWN 6800 SYSTEM, CENTERED AROUND THE AMI PROTO SERIES. THIS AMI EQUIPMENT IS MOST VERSATILE, MOST DEPENDABLE, AND VERY DELIGHTFUL.

THE BUILT-IN PROM-BURNER FACILITIES MAKE ME A BELIEVER IN EPROMS. MANY ROUTINES BECOME FIRMWARE, ALWAYS PRESENT, ALWAYS INSTANTLY AVAILABLE.

THE SOFTWARE SUPPLIED ON THE AMI PROTO ROM INCLUDES 25 FORMALIZED SUBROUTINES, FROM PRINT ROUTINES TO MULTIPLIES TO PUSH-ALL-PULL-ALL. THE PERMANENCE AND DEPENDABILITY OF THESE ROUTINES MAKE THEM A PART OF ALL PROGRAMING.

THE AMI PROTO ROM OPERATING SYSTEM IS SO FAR AHEAD OF MANY OTHERS THAT ITS A SHAME IT ISNT ADVERTIZED. THE ROM CONTAINS MOVE-BLOCK, BURN-EPROM, LOAD

FROM EPROM, BETTER MEMORY CHANGE - RUN - BREAK - DISPLAY FACILITIES THAN MOST.

* THEN THERE IS THE AMI PROTO ASSEMBLER ROM, WHICH YOU JUST PLUG IN AFTER SPENDING $30. IT'S ALWAYS THERE, IT ALWAYS WORKS. IT DOESN'T HAVE LABELS, BUT IT UNDERSTANDS THE CORRECT ASSEMBLER LANGUAGE, AND IT ACCEPTS STRAIGHT HEX, ANY NUMBER OF HEX PAIRS PER LINE, WITHOUT WARNING, INTERMIXED WITH THE ASSEMBLER LANGUAGE.

* I AM SURE THE AMI SERIES OF 6800 EQUIPMENT IS NOT AS WIDELY SOLD AS IT SHOULD BE, BECAUSE THE ONLY ADS YOU SEE FOR IT ARE ONE-LINERS. IT DESERVES MUCH MORE ATTENTION. PERHAPS YOUR JOURNAL WILL BE THE PLACE.

* THE VERY AVAILABILITY OF AN INSTANT-EPROM BURNED ON MY AMI SYSTEM HAS CAUSED IT TO BE THE SYSTEM OF CHOICE FOR A FEW SCIENCE PROJECTS HERE IN OUR GEOPHYSICS RESEARCH OPERATION. WHEN SOMEONE WANTS A HALF-K OF PROGRAM FOR HIS RCA OR ANY OTHER BOX, I CAN MAKE HIM A ROM IN THE TIME IT TAKES TO TRANSFER THE DATA FROM A HOST TO MY MEMORY, PLUS ABOUT TWO MINUTES TO SAY 'BURN'. THIS HAS BROUGHT PEOPLE MY WAY, EVEN THOUGH THE 6834 EPROMS ARE SORT OF OUT IN LEFT FIELD REGARDING COMPATIBILITY WITH EVERYBODY ELSE'S BURNERS.

* THE SYSTEM HERE LACKS A DISC SO FAR. I'LL SPEND THE MONEY WHEN THE FIELD SETTLES DOWN. MENAWHILE, I USE TOM PITTMANS TINY BASIC, WHICH COMES IN VARIOUS FORMS, INCLUDING A SET OF EPROMS FOR THE AMI SYSTEM. THIS BASIC DOES ALL IT SHOULD. TOM HAS EVEN WRITTEN AN ENTIRELY WORKING ASSEMBLER FOR THE INTERMEDIATE LANGUAGE OF THE TINY BASIC, WHICH RUNS IN TINY BASIC. THIS IS INCREDIBLE WHEN YOU REALIZE THAT THERE ARE NO STRING CAPABILITIES IN TINY BASIC, YET THE INPUT TO THE ASSEMBLER IS ALL STRINGS. HE PEEKS AND POKES A LOT.

* WHAT I WANT NEXT IS A BIGGER BASIC. BUT I HAVE MY LIST OF REQUIRENENTS. IT MUST BE EPROM-ABLE. IT MUST HANDLE STRINGS AND SUBSTRINGS. IT MUST BE WORKABLE IN HIGH MEMORY, BECAUSE I DON'T WANT THE EPROMS IN LOW MEMORY.

PLEASE DISCUSS IN YOUR MAGAZINE WHAT CAN BE DONE WITH EACH OF THE 6800 BASICS IN THIS REGARD, ESPECIALLY THE NEED TO BE EPROM-ABLE. YOU SURE CAN'T TELL FROM THE TINY ADVERTISEMENTS.

JUST ANNOUNCING THAT A SOFTWARE PRODUCT IS 'FOR YOUR SWTPC' OR THE LIKE MISSES A GOOD PART OF THE MARKET.

* YOU INDICATE AN INCLINATION TO THE 50 PIN BUSS. THAT'S FINE, BUT PLEASE START BY PUBLISHING THE 50 PIN BUSS. I HAVE NEVER SEEN IT, HAVING MISSED THAT ONE ISSUE OF AN EARLY MAGAZINE WHERE I HEAR IT WAS DISCUSSED IN DETAIL.

* THEN FOLLOW THAT WITH A DISCUSSION OF ITS COMPATABILITY WITH THE AMI BUSS, WHICH SHOULDNT BE VERY DIFFERENT. IF WE MUST ALL HEW TO ANOTHER 'DEFACTO STANDARD' 50 PIN BUSS, LETS HEAR ABOUT HOW TO GET THERE. HAVING THIS AT HAND, A WHOLE RAFT OF PRODUCTS ALREADY ON THE MARKET MIGHT BE PLUGABLE INTO AN EXTENSION OF AMI EQUIPMENT.

* THE MAGAZINE TRADE HAS WANDERED OFF ON TWO SEPARATE WAYS, LEAVING A GREAT GAP IN THE MIDDLE.

ON THE ONE HAND, WE HAVE A CONTINUING DIALOG ON NEW LANGUAGES, SUCH AS LISP, DISCUSSED BY GUYS WHO TOOK COMPILER THEORY IN COLLEGE.

ON THE OTHER HAND, GREAT AMOUNTS OF PAPER DISCUSS SPECIFIC SYSTEMS, AND SPECIFIC PRODUCTS TO PLUG INTO THOSE SYSTEMS, AS THOUGH NONE OF THEIR READERS YET OWN ANY HARDWARE, AND THEY ARE ALL GOING OUT TO BUY A NAME BRAND.

THE GAP IN THE MIDDLE SURROUNDS THOSE OF US WHO ALREADY HAVE SOMETHING, NOT THAT BRAND, OR WHO BUILD OUR OWN, NOT EXACTLY LIKE SOME BRAND. THAT IS, THE DETAILS ARE BEING NEGLECTED AT THE HARDWARE LEVEL.

* A SIMILAR SITUATION WOULD EXIST IF THERE WERE PUBLICATIONS ON AUTOMOBILES FOR SALE, AND PUBLICATIONS ON WHERE TO GO ON YOUR VACATION, BUT NONE ON KEEPING YOUR CAR WORKING. YET MAGAZINES LIKE POPULAR MECHANICS GO ON AND ON VERY SUCCESFULLY, BOUGHT BY THOSE WHO WANT TO KNOW THE IN-BETWEEN.

* BEST OF LUCK IN YOUR VENTURE.

NOEL J. THOMPSON
UNIV HAWAII GEOPHYSICS
2525 CORREA ROAD
HONOLULU HAWAII 96822

* * FLASH * *

At 10:20 AM on February 2, 1979, the first 6809 to be installed in a complete system, was reported from San Antonio, Texas.

SWTPC announced that it worked as anticipated.

Watch for an early review of the new SWTPC MP-09 system board, running on one of our machines.

EDITORS REMARKS

This, the first issue of '68' Micro Journal, is the beginning of a long and exciting experience for myself and our worldwide associate editorial staff. Commencing a few months back we began to appoint domestic and foreign 'Associate Editors'. They bring a background of experience and knowledge to our ranks that will enrich us all, as the months pass. To them I want to express my heartfelt thanks, for undertaking this task. By their experience and guidance we all shall profit.

From the start we know that without good material no magazine can 'RUN'. It is our dedicated promise to you that we will bring you more solid 6800 material, each month, than previously available.

First I want to personally thank each of you for your support. Without the advance subscriptions (far more than we ever hoped for) and the support of the advertisers, we could not have 'booted up'. Let no one doubt, WE ARE FOR REAL.

It is my intention to send you the spectrum of 6800 material. This magazine is for YOU. If there are things you do or do not like about '68' Micro Journal, let me know. I need your input. Our advertisers need your support. Let them know you read '68' Micro. That way we all profit.

As you can plainly see most of the other small computer magazines are tending to cater more and more to the S100 bus or the burgeoning 'appliance machine' market. Hard dollar facts dictate this. This could have left the 6800 users 'out in the cold'. As time goes by, faster than you might imagine, it will be more apparent that '68' Micro Journal came at the right time.

The myth that the S100 bus and 80 type CPU are the standard is fast crumbling. Those who put the 6800 to critical usage will verify that we have the best combo going. Who needs 413 different breeds and kinds of memory boards? I much prefer one or two that work the first time, cost less and hang in there year after year. They talk standard, we have a standard. Ours all fit the same bus, know all the same signals, give the same dependable results and can be purchased knowing that they will perform as expected. The 6800 user now has a better choice of hardware and software than any other group of micro computer end users. Check with some of your buddies who run (sometimes) the other brand. Kinda makes you feel good, doesn't it.

We need good material. Only you can supply that. We need software and hardware type articles. Hints and kinks type material helps us all. If you have a 'fix' of any type, let us know. This can save a lot of hours pouring over technical data by those who are just beginning or are not technically bent. If we all cooperate '68' Micro Journal can be the focal point for all who enjoy this great hobby. We will award subscription extensions, for material accepted, up to the 'Life Subscription'. You not only help yourself but others when you contribute. No material should be considered too trivial or insignificant to be considered. Sure, we need the heavy stuff (for some) but remember the beginner. One persons 'cake' may be another persons 'brick'. A great need exists for material for all levels of skill.

As time progresses we will go to four color advertisements and all the other niceties of normal magazines. I have tried to keep cost down to a reasonable level. For example; since we first announced the formation of '68' Micro Journal, paper prices have increased thirteen percent. Labor six percent and more coming. I need your input, reader and advertiser, concerning the quality of our effort. Do you want slicker paper (higher cost to everyone)? Do you want lots of color (higher cost to everyone)? Do you want cartoons and jokes or 6800 tidbits, as space fillers? Do you want abstract material or proven applications? Do you want an expanded classified section (for non-business) swaps or un-loads? And so the list goes on. I need to know what you want. You are going to make the judgement concerning the format and content of '68' Micro Journal. So let me know and now.

DMW

SSI Microcomputer Software Guide
published by:

S S I
4327 East Grove Street
Phoenix, Arizona 85040

Did you ever have this problem? You are working on a program and you encounter a problem that totally stumps you, but you remember having read an article somewhere that contains the solution to your problem. But you can't remember where you saw the article. Or perhaps several of the computer magazines are asking you to subscribe (or re-subscribe) but you only want to take those who best cover your areas of interest. I used to have one or the other of these problems at regular intervals, before I got a copy of the SSI Microcomputer Software Guide.

The SSI Microcomputer Software Guide claims to be the most comprehensive reference guide to microcomputer software ever published. That is a statement with which I cannot argue, as it is an index of over 2000 programs from 130 different sources with software listed by 236 classifications. This 124 page book is really packed with information; for example, there are sixteen entries under the heading REAL TIME, twenty six listings for PLOTTING, and over thirty listings for LANGUAGES.

The SSI Microcomputer Software Guide contains just about everything that you could hope to know about where to find microcomputer software and software articles up to the date of its publication. And that brings me to its biggest fault; it does not say, "Volume 1 of a series". I do hope that SSI will update this helpful text on an annual basis. If they do, I'll certainly subscribe because it is a super bargain for $7.95.

MEF

MP-09 CPU Board (6809)

Coming soon for the serious user of the SWTPC 6800 Computer are two welcome additions. One, although not an addition, actually an upgrade, is the new 6809 board. For the user who would like a real 'weapon', this will be it. The new board, which will plug onto the existing motherboard, of your SWTPC, will probably be called the MP-09. The MP-09 will sell for about $195.00 assembled and $175.00 in kit form. Like the MP-A2 CPU board it will have provisions for 8K of 2716 EPROM, 2K ROM monitor and new control and address lines. It should be no hassle to convert most existing systems to the new board. Some foil cuts and jumps will be required, mostly on the baud rate lines of the 50 pin portion of the motherboard. Delivery is expected in the next month or so, depending upon delivery of the 6809 chips.

Some of the more important features include:

- SS50 Bus compatibality
- Paged memory addressing
- Extended addressing to 256K bytes
- Two 8 bit accumulators which can be concatenated to form a 16 bit accumulator
- Two 16 bit indexable stack pointers
- Two 16 bit index registers
- Direct page addressing throughout memory
- Full 65K range branches
- 8X8 unsigned multiply
- Transfer and/or exchange of any equal size registers
- Push or pull any registers from either stack
- Enhanced PC pointer control
- Index mode includes accumulator or up to 16 bit offset
- Index mode with auto increment and/or decrement with all of the pointer (4) registers

When we receive our new 6809 board from Southwest Technical Products we will have one of the first and most comprehensive reports available. Watch for it.

SWTPC 'Winchester' Hard Disk

Not to be slighted is the new 'Winchester' hard-pak drive coming. Twenty megabytes raw and 16 megabytes formatted, Cal-Comp drive and SWTPC controller.

On last report it was up and running, so watch for additional information, as it becomes available. We hope to have a complete report on it soon, just as soon as we can get our hands on one.

The speed with which our mailing list is growing, we need it. At the present we have one complete SWTPC (6800/B, CT-82 Terminal, DMAF1 dual disk and two form printers) doing most of our subscriber data handling.

For additional information contact Southwest Technical Products, 219 W. Rhapsody, San Antonio, Texas 78216. Don't wait too long; the waiting line can get really streached.

DMW

'CACHE'
Chicago Area Computer Hobbyist Exchange

CACHE a not-for-profit organization announced the following information concerning their new 'Computerized Bulletin Board System'. This system was designed and programmed by Ward Christensen and Randy Suess. The system requires a 110 or 300 baud terminal with a type 103 modem. After the connection is made, type C/R a few times to set the baud rate compatibility. The system prompts from then on. You may just get a busy signal (we did), but keep trying (we did) and you will get thru (we did).

BOSTON	617 963-8310
CHICAGO	312 528-7141
DALLAS	214 641-8759
SAN DIEGO	714 565-0961

We placed a request for a good chess game for the 6800. Got no replies but found a very active computer message and swap-shop going, 24 hours a day.

Thanks, Phil Schuman for the info, keep it coming. Anyone got info for a good duplex modem for the SWTPC?

DMW

Conducted by Jack Bryant

CRUNCHERS CORNER

This monthly column is intended to provide a place for the exchange of ideas on microcomputer arithmetic. A systematic exposition of fixed and floating point arithmetic, hardware and software, algorithms for approximation and so on is planned. Questions and comments submitted to this column can be on any subject relevant to "number crunching," and should be addressed to:

Jack Bryant
Department of Mathematics
Texas A&M University
College Station, Texas 77843

We ask that all correspondents supply their names and addresses.

YOUR COMPUTER AND THE REAL WORLD

Consider a large field sparsely populated by rabbits (known, of course, for their reproductive potential). On the average, each female may be expected to produce six offspring every 60 days, three of which will be female. The daughter rabbits may be assumed (in this over-simplified model) to produce their first litter at age 60 days. Given that one mated pair is present on day 1, what is the population of rabbits after two years?

Pencil-and-paper analysis of this question may go roughly as follows. First of all, we notice we need not count the males. We simply assume there are as many males as females. On day 60, the first litter will be born, containing three females. On day 120, each daughter plus the original parent will produce three daughter offspring, bringing the total to 16 females (and 32 rabbits). In Table 1 this process is extended to day 720.

TABLE 1. EXPONENTIAL GROWTH

Day	Number
1	2
61	8
121	32
181	128
241	512
301	2,048
361	8,192
421	32,768
481	131,072
541	524,288
601	2,097,152
661	8,388,608
721	33,554,432

While this example may seem a little absurd, the model we assume is applicable in many other situations: instead of rabbits, suppose we had a bacterium which divides asexually every 30 hours. After 60 hours the population will be four times what it was initially. Table 1 can be used to find the population after 720 hours (= 1 month). Another example of the same general problem arrises in business: namely the compound interest problem. Money at 6 per cent interest doubles about every 12 years. In these times, of course, 6 per cent interest is unusual. At 26 per cent interest, whatever is borrowed (or loaned) doubles every 3 years. Using the same ideas as we used to build Table 1, we can see that, for example, $2,000 becomes (about) $128,000 after only 18 years.

Although these three problems have a similar underlying assumption, they have features which are quite different. The rabbit overpopulation problem, for example, ignores predators and a limited food supply. (Some rabbits will be eaten by foxes; others will starve.) Also, it is not true that exactly three females are found in each litter. This means the numbers in Table 1 are only approximately correct. The model seems much more believable for the bacterium. Still, the accuracy of the answer is questionable. For the business problem, $2,000 at 26 per cent interest compounded annually for 18 years is exactly $128,144.18. (Business arithmetic is performed with nearest-cent rounding. With truncation (rather than rounding), the amount is $128,142.66. It is easy to imagine that this roughly .001% difference could amount to many dollars. Without rounding the amount is approximately $128,144.43.)

In each of these problems, part of the real world is modelled. Certain assumptions are made about the real world, and these are translated into mathematical terms. Then the mathematical model is "explained" to a computer which then computes. Finally, the result of the computation is compared with what is known about the real problem: this can sometimes result in additional assumptions (for example, predators) or the choice of different computational techniques (for example, nearest cent rounding).

The computer is usually essential in today's mathematical modelling problems. Usually, exact (pure) mathematical techniques for obtaining a solution are not known. In addition, the accuracy of the data or assumptions can be questioned, so that even an exact mathematical analysis may not be strictly accurate when translated back to the real world. In some problems, an approximate analysis is appropriate, and can lead to an improved model (better assumptions) or better data collection techniques. An example of approximate analysis which we will study in detail in the future is floating point computation. Table 1, for example, could have been computed with 7 digit accuracy: If this is done, the last entry becomes 3.35544×10^7, which is close enough for a rabbit but no good at the bank.

One point of this example is clear: before your computer can be used to solve many of the problems of the real world, it must be taught (programmed) to do a variety of arithmetic calculations. Also, it should be able to understand (and give results in) decimal format numbers. In this column, we intend to begin (at the beginning) with a microprocessor instruction set and show how the extended arithmetic operations needed can be coded.

At this point there is a problem of what approach we should take. Should we define a hypothetical microprocessor and code everything in this language or should we take an actual existing microprocessor and use its instruction set? The advantage of the first approach is that the ideas are likely to be presented with more generality. On the other hand, a reader following the second approach will actually learn something of a real microprocessor and can immediately test the techniques developed. We have elected the second route, and have chosen the Motorola M6800 as the microprocessor.

THE M6800 MICROPROCESSOR

The M6800 has two 8 bit accumulators, one 16 bit index register, a 16 bit stack pointer, a 16 bit program counter and a 6 bit condition code register. Data is 8 bits wide and addresses are 16 bits wide. The condition code register contains five bits which may be changed by an arithmetic operation:

C Carry-Borrow

V Two's Complement Overflow

Z Zero

N Negative

H Half Carry

The first four of these are used for conditional branching. Also, the Carry-Borrow bit directly supports multiple precision arithmetic. The Half Carry bit is used for performing decimal arithmetic.

The accumulators are named A and B, the index register X and the stack pointer S. Since we are approaching the problem from an assembly language programmer's viewpoint, the program counter register is of less concern. The program counter points to the next instruction to be executed, and is updated automatically after execution. Machine-language instructions are one, two or three bytes long. The first byte is always the operation; the others, if present, constitute either an operand, a pointer to an operand or an address from which the next instruction is to be taken.

Assembly language statements are coded

[label] nmonic [operand [operand]] [comments]

Brackets ([,]) are used here to indicate something which may be optional. Nmonics are three letter codes which are easily learned. An assembly language program must be translated into machine language and stored in the computer memory before it can be run. A program to perform this task is called an Assembler. Of course, the translation can be done by hand.

One important program which is usually constantly present is called the monitor. Its function is to provide a means for setting the contents of memory as desired, to start the execution of a program, and so on. While a number of monitor programs are available for the M6800 on read only memory, most are compatible with the Motorola program MIKBUG in their main input/output features. MIKBUG contains perhaps 15 useful subroutines that may be called by a user program to perform character, byte and block oriented input or output through the system console monitor. A few of these will be introduced as we proceed to write programs.

Before we introduce any M6800 assembly language programs, we pause to develop some essential background on representation of numbers as we know them andas the computer knows them.

NUMBER SYSTEMS

Everyone knows the usual decimal notation for numbers--at least for integers. The common representation is called sign-magnitude. The magnitude is given a positional representation in decimal. For example,

$$2743 = 2 \times 10^3 + 7 \times 10^2 + 4 \times 10 + 4 .$$

A negative number is indicated by simply prefixing the string of digits with a minus sign. The base of the decimal system is the number 10. The system uses eleven symbols to denote an integer: the digits 0 - 9 and the minus sign.

Although computers could be manufactured to directly handle numbers in this representation, this practice is not common. Instead, modern computers use binary, or base 2, arithmetic. In binary arithmetic, only two symbols, namely 0 ("off") and 1 ("on") are used. The usual notation allows an unrestricted "width" (number of digits) of a number. In contrast, in computer arithmetic a fixed width is usually assumed. For example, the data bus in the M6800 is 8 bits wide. The fact that the width is known allows negative numbers to be represented by turning on a particular bit, usually the first (= most significant) bit. We will return to this presently.

Sometimes only positive numbers are required. This is the case in M6800 indexed addressing: the offset is an 8 bit unsigned integer, which is added to the value of the index register to construct an address. The index register is a base address. When only positive numbers are needed, numbers 0 - 255 can be represented in 8 bits. The number 131, for example, can be written in binary (as an unsigned number) as follows: First, divide 131 by 2, obtaining the quotient 65 and remainder 1. The least significant bit is 1. Now divide 65 by 2, obtaining quotient 32 and remainder 1. The next bit is 1. Divide 32 by 2 obtaining 16 with remainder 0. The next bit is 0. Continue until 1 is divided by 2 giving quotient 0 with remainder 1. The representation which evolves is

$$131_{10} = 10000011_2 .$$

That is, the unsigned binary representation for 131 is the bit pattern 10000011.

One thing we notice is that binary representations are longer than decimal. In fact, they are much too long. (The last number in Table 1 is 26 bits long in binary.) For this reason, other representations from which the binary representation can be obtained immediately are used. If the base is a power of two, say 2^n, then the binary representation can be blocked in groups of n and each group converted independently. (Because 10 is not a power of 2, this does not work for decimal to binary conversions.) The system with base $8 = 2^3$ is called the octal system. Starting at the right, the conversion to octal proceeds as follows:

$$\underset{2}{\underline{10}}\ \underset{0}{\underline{000}}\ \underset{3}{\underline{011}}$$

That is, $131_{10} = 203_8$. Another system, clearly superior for an 8 bit machine, is the hexadecimal system with base $16 = 2^4$. In this system, the numbers ("digits") ten through fifteen are denoted by A through F. To convert to hexadecimal, group the number in two sets of four bits:

TABLE 3. INTERPRETATION OF BIT PATTERNS AS FOUR BIT NUMBERS

Bit Pattern	Unsigned	Sign-Magnitude	One's Complement	Two's Complement
0000	0	0	0	0
0001	1	1	1	1
0010	2	2	2	2
0011	3	3	3	3
0100	4	4	4	4
0101	5	5	5	5
0110	6	6	6	6
0111	7	7	7	7
1000	8	-0	-7	-8
1001	9	-1	-6	-7
1010	10	-2	-5	-6
1011	11	-3	-4	-5
1100	12	-4	-3	-4
1101	13	-5	-2	-3
1110	14	-6	-1	-2
1111	15	-7	-0	-1

four bit examples. All of the features we wish to illustrate can be found here.)

Note that all four systems give the same meaning to positive numbers. In the sign-magnitude and one's complement system, each number has a "negative," obtained by complementing the sign bit (in the sign-magnitude system) or the entire bit pattern (in the one's complement system). (The complement of a bit is 1 if the bit is 0 and 0 if the bit is 1.) To find the negative of a two's complement number, we complement the bit pattern

$$\underset{8}{\underline{1000}}\ \underset{3}{\underline{0011}}$$

We use the convention that a prefix of '$' means hexadecimal. Thus, $131_{10} = 203_8 = \$83$. Table 2 shows bit patterns corresponding to each of the base 16 digits. For fun, try converting some others: 94 (answer 01011110_2 = $5E), 240 (answer 11110000_2 = $F0), 255 (answer $1111\ 1111_2$ = $FF), 271 (10000111 ?).

TABLE 2. BINARY EQUIVALENTS OF HEXADECIMAL DIGITS

0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111

What happened? Well, 271 exceeds the capacity of an 8 bit unsigned integer data type. Of course we may have suspected that when 255 turned out to be $FF. There are only 256 possible bit patterns in all with 8 bit data, and so only 256 possible non-negative numbers.

There are three principle methods to represent negative numbers in binary: they are sign-magnitude, one's complement and two's complement. The meaning of each bit pattern in each of the three systems is shown in Table 3. To keep the table short, four rather than 8 bit numbers are given; the most significant bit is the sign bit. (We will contioue to use occurs, and the V (overflow) bit is set on a two's complement overflow.

Listing 1 is a M6800 program which demonstrates hex addition. The program is less than 40 bytes long, and contains only one arithmetic instruction: ABA (Add Accumulator B to Accumulator A). To someone unfamiliar with assembly language programming, the program may seem brutally incomprehensible. So be it. The program is probably as simple as one which actually does something can be. The program begins execution at $0000; expect a new line to begin with execution. Enter two 8 bit numbers in hex (with no spaces or carriage returns). The sum of the numbers (in hex) and the value of the processor condition code should be displayed.

In this program, the stack is used to save A after the instruction ABA because other ways of storing A change the condition code. Only accumulator A can be used to view the condition code register. Another way to accomplish the same thing would be to store the first operand somewhere and perform the addition in accumulator B. Most of the program in Listing 1 is unchanged.

```
        ...
        JSR   BYTE    FETCH OPER NO.
        STA A TEMPA   STORE IT
        JSR   OUTS
        JSR   BYTE    FETCH OTHER.
        TAB           STORE IT IN B.
        ADD B TEMPA   ADD TO FIRST.
        TPA           FETCH CC.
        PSH A         STORE ON
        PSH B         STACK.
        TSX           POINT TO  UM AND CC.
        ...
TEMPA   RMB   1       SPACE FOR TEMP ACCB VALUE
```

and add 1. For example, -5 = 1011 → 0100 + 1 = 0101 = 5. Something strange for -8 happens: -8 = 1000 → 0111 + 1 = 1000 = -8. This situation is flagged in a microprocessor by the overflow condition code.

The main advantage of the two's complement system is that the hardware (internal adder) can perform exactly the same function when adding or subtracting both unsigned and two's complement numbers. That is, the hardware will not need to know how the user is interpreting the number. This means no special instructions are needed for handling signed numbers separately from unsigned ones. As long as the result of a computation does not exceed the capacity of the internal adder, there is no difference. The condition code register contains two bits, named C and V, to flag when an overflow has occurred. Consider the following four examples of binary additions (see Table 4). The carry bit is set provided an unsigned overflow

T BLE 4. EXAMPLES OF BINARY ADDITIONS

Bit Patterns	Unsigned	C	Two's Complement	V
1000 0111 1111	8 7 15	clear	-8 7 -1	clear
0111 1111 0110	7 15 6	set	7 -1 6	clear
0100 0100 1000	4 4 8	clear	4 4 -8	set
1000 1000 0000	8 8 0	set	-8 -8 0	set

```
                        NAM   ADDTST  PROGRAM TO TEST ADDITION
               *
               *        MI  NG ENTRY POINTS:
E055           BYTE     EQU   $E055   BYTE OUTPUT SUBROUTINE
E07E           PDATA1   EQU   $E07E   BLOCK OUTPUT
E0CA           OUT2HS   EQU   $E0CA   OUTPUT 2 HEX + SPACE
E0CC           OUTS     EQU   $E0CC   OUTPUT A SPACE
               *
0000 CE 00 23  LOOP     LDX   #CRLFD  PREPARE FOR NEW LINE.
0003 BD E0 7E           JSR   PDATA1  CR/LF
0006 BD E0 55           JSR   BYTE    FETCH FIRST OPERAND.
0009 36                 PSH A         STASH IT.
000A BD E0 CC           JSR   OUTS    SPACE ONE.
000D BD E0 55           JSR   BYTE    FETCH SECOND OPERAND.
0010 33                 PUL B         FIRST ONE AGAIN.
0011 1B                 ABA           ADD THE TWO.
0012 36                 PSH A         SAVE FOR LATER.
0013 07                 TPA           FETCH CONDITION CODE.
0014 33                 PUL B
0015 36                 PSH A         PLACE CC THEN
0016 37                 PSH B         SUM ON STACK.
0017 30                 TSX           POINT TO STACK.
0018 BD E0 CC           JSR   OUTS
001B BD E0 CA           JSR   OUT2HS  PRINT THE SUM.
001E BD E0 CA           JSR   O T2HS  PRINT THE CONDITION CODE.
0021 35                 TXS           RESTORE THE STACK POINTER.
0022 20 DC              BRA   LOOP    DO IT AGAIN.
               *
0024 0D 0A 04  CRLFD    FCB   $0D,$0A,$04   DATA FOR CR/LF PRINT.
                        END

               SYM OL T  LE

BYTE    E055    CRLFD   0024    LOOP    0000     T2HS  E0CA
OUTS    E0CC    PDATA1  E07E
```

Listing 1. Addition Test Program. Execution begins at address 0000.

One subtle point which this program demonstrates has nothing to do with arithmetic: it is that MIKBUG program OUT2HS includes an INX instruction. This fits nicely with the PSH-TSX-TXS instructions. This is illustrated in the program. The disadvantage of selecting a particular processor (such as the M6800) rather than an abstract processor is also revealed. Each actual processor will have its own programming tricks. Tricks are like fog to a beginner. An abstract untricky processor may be better to communicate abstract ideas. On the other hand, the tricks are pretty good too. We will not always be beginners.

Next month we discuss multiple precision addition and subtraction and conversion between decimal (ASCII) and binary.

```
 ┌──────────── First Operand
 │   ┌──────── Second
 │   │   ┌──── Sum
 │   │   │   ┌ Condition Code
7F  80  FF  D8
7F  FF  7E  F1
40  40  80  DA
80  80  00  07
```

Figure 1. Sample Run of the Program of Listing 1.

Note that the first two bits of the condition code are always on; the bits are 1 1 H I N Z V C . In this example, the interrupt mask I was set, so the first hex character is F or D depending on the half-carry bit. The second nybble (hex character) is NZVC (in binary). As unsigned problems, the four examples shown here are 127 +128 = 255, 127 + 255 = 126, 64 + 64 = 128 and 128 + 128 = 0 , respectively; they are exactly analogous to the four problems in Table 4.

Howard Berenbon
Software Exchange
2681 Peterboro
W. Bloomfield, MI 48033

AND NOW A TELEPHONE WITH BYTE

Home computing is a term I have been using frequently to describe my activity as a computer hobbyist. I am operating a homebrew Motorola 6800D1 microcomputer system, expanded to 8K. But how many of us are actually using our systems for home applications? Oh, it is interesting to write and use programs for playing games and use other programs which assist us in our programming. But it would become even more interesting to have programs for our systems which do other things, beside number manipulation. Up to now, demonstrating a computer game or two to friends doesn't really show them what the system can do. Games are fun for a while, but then I always hear the comment, "but what is the computer good for...what can it be used for". So I decided to write two telephone applications programs, the 6800 AUTOMATIC TELEPHONE DIALER and the 6800 TELEPHONE ANSWERING DEVICE PROGRAM.

The 6800 AUTOMATIC TELEPHONE DIALER PROGRAM is an excellent program to demonstrate what a home computer can really do. It is written in 6800 machine language and is available for $9.95 postpaid from my address shown above. I offer a complete documentation package, including circuit diagrams and instructions. Included is a punched paper tape in Mikbug* format and an object code listing. Instructions are provided for modifying the dialer to operate on other 6800 systems. You do not have to be an engineer or scientist to appreciate the home computer. Having your 6800 system dial the telephone demonstrates that hidden mathematical magic of this new technology.

The dialer program is written to operate in 6800 systems using Mikbug* but may be converted to operate on most any 6800 system. It will run in less than 500 bytes of RAM memory. Up to 650 variable length telephone numbers can be stored in memory, to be accessed later for dialing. This of course, is limited to the size of your system memory. A buffer area is used to store the phone numbers. Any length phone number may be stored, including area codes to make long distance calls. A character code representing the particular phone number is used to access the number from memory. When you are ready to dial, just enter the code. A subroutine runs thorurgh the data buffer searching for the code entered. Once the code is found the number is processed for dialing through the telephone interface circuit.

The interface circuit is used to interface the computer to the telephone. using only 5 common electronic components. A PIA 6820 is connected to the interface circuit. Dialing is performed with a series of pulses outputted to the telephone interface through the PIA. The PIA is initalized for I/O before entering into the dialers mini-operating system. Telephone numbers are loaded using the desired code, then accessed later with the 'Search' command.

The telephone dialer interfaces easily to any standard telephone, dial or pushbutton. It dials a number in approximately 4 seconds, or about twice as fast as it takes to dial the telephone normally.

The interface may be left connected to the telephone all the time.

Other applications include intergrating the dialer in a home or office burglar alarm system for dialing the local police department to report a breakin; using a pre-recorded message activated by your system. It could also report a fire and the address, as well. Business dialing becomes much more efficient and faster. I would be interested in hearing from others with ideas for these programs.

A second applications program offered is the 6800 TELEPHONE ANSWERING DEVICE PROGRAM, available for only $4.95 postpaid. The program allows the computer to answer the telephone, play a pre-recorded message and record the callers message. It can be used with any 6800 system, having as little as 500 bytes of RAM. Complete documentation, assembly listing and circuit diagrams are included.

Several other programs are available as described in my free flyer. Please address all request to the address shown at the top of this article.

* Mikbug is a registered TM of Motorola, Inc.

FLEX TO BFD

Coast Guard Photo Journalist
Dale L. Puckett
183 Farm Acre Rd.
Syracuse, NY 13210

One of the best disk hardware systems on the market is the Smoke Signal Broadcasting BFD-68. The unit comes in a sturdy cabinet with power supply and provisions for from one to three disk drives. A controller board plugs into the SS-50 bus.

On the other hand, one of the best disk software systems is FLEX which was written by Technical Systems Consultants (TSC) and is delivered with the SWTPC MF-68 Disk system. This article will tell you how to use the two as a team.

I started out with the BFD-68 in January of 1978. At the time, I was told by the Computer Mart in New York City that the Smoke Signal System was the only one to consider because of the poor software delivered with the MF-68.

Why then did I want to make the switch to FLEX? Why didn't I just stick with DOS-68, the operating system which comes with the BFD-68. There were several reasons.

Compatibility was probably the most important. I wanted to exchange a few programs with my brother in Oklahoma City. He uses FLEX and didn't have the time to make tapes.

Recently I was transferred to Syracuse University for a year of special training. I went to a local CHIPS meeting and met a couple of 6800 users. They used FLEX. And, we wanted to work out a few problems together. It was time to look into FLEX.

Since I have made the switch, I have been amazed at the ease of operation. I think FLEX is great because it is people oriented.

A good example is the method FLEX uses to assign extensions to files. Depending on the source and type of material it defaults to a particular extension, usually .TXT for text and .BIN for binary. The assembler automatically assigns a transfer address to the file, making it easy to create command files.

To execute a command, you simply type the command name along with any paramaters. You may use a comma or a space as a delimiter when typing in a command line.

FLEX ignores all but a few special control characters. This allows you to hit control p, control v, to erase the TVT screen without affecting the operating system. If you try that with DOS-68, you get an obnoxious, unhuman error code. FLEX gives you an error message in plain English, ie, NO SUCH FILE.

Allow me one more sales point and then we'll move on to the technical details involved in the conversion. The TTYSET utility command alone makes FLEX far superior to the competition.

With TTYSET, the user can define the Backspace Character, Delete Character, End of Line Character, Depth Count, Width, Null Count, Tab Character, Full or Half Duplex, Eject Count, Pause On or Off, and the Escape Character.

All of these parameters can be changed at once, or they may be changed separately. You can even build a file which changes the parameters for various input-output devices and then execute that file when you change terminals. If you use FLEX's GETCHR and PUTCHR routines with all of your software, you have complete control of all these parameters automatically at all times.

For a more complete description of FLEX's capability take a look at Mickey Ferguson's article, on Page 72 of the October, 1978, KILOBAUD.

Down to business. TSC writes outstanding software. Everything is structured and easy to follow.

The conversion is made easier because TSC has gathered all of their disc driver routines and placed them on one page of memory. You will find them at $7F00.

FLEX uses five routines for all communication with the disk hardware. The table below gives you the location of each routine in the jump table and the location of each routine.

```
ADR    NAME     TARGET
$7F00  Read     $7F17
$7F03  Write    $7F67
$7F06  Verify   $7F92
$7F09  Restore  $7FA2
$7F0C  Drive    $7FD2
```

To use FLEX with the Smoke Signal Broadcasting BFD-68 unit you must also know the locations of the driver routines in the ROM on the controller board. It was a lot of work but we managed to trace these locations down and label them. Here is the jump table.

```
ADR    Name or Function
$8020  Cold Start
$8023  Warm Start
$8026  Initialize PIA
$8029  Read Sector
$802C  Write Sector
$802F  Read Track
$8032  Write Track
$8035  Seek
$8038  Restore

*  Note table is in
*  two parts.

$8060  Send Command
$8063  Operation Complete
$8066  Clean up & Return
$8069  Write Sector Register
$806C  Write Track Register
$806F  Write Track Number
$8072  Read Track Register
$8075  Step In
$8078  Step Out
$807B  Step
```

Let's start with the READ routine. FLEX calls this routine with the FCB Sector Buffer Address in X, the track number in A and the Sector Number in B.

The BFD-68 ROM expects to see the track number in $A07C, the sector number in $A07D and the FCB Sector Buffer in $A07E. The following code does the job nicely.

```
ORG $7F17
STA A $A07C  track
STA B $A07D  sector
STX   $A07E  sector buffer
JMP   $8029  call BFD-68 read
```

The write routine is identical. The A-register goes to $A07C, the B-register goes to $A07D and the X-register goes to$A07E. This time you jump to $802C however.

You should ORG your write routine at $7F67, the location pointed to by FLEX's jump table. Also notice that there is an implied RTS at the end of these jumps, so you are actually just pointing to a subroutine when you jump.

The Verify routine is next. All you need to do is read the sector you just stored. Use this code.

```
ORG $7F92
JSR $8029  BFD-68's read sector
BIT B $18
RTS
```

The Restore function is next. It instructs the drive to do a Seek to Track 0. Because it is a little more complicated I will ask you to refer to the assembled source listing accompanying this article.

The final disk driver routine is located at $7FD2. It is the Drive Select routine. I own a single drive system and used the simple routine listed here.

```
PSH A
LDA A $08
STA A $A07B
PUL A
RTS
```

If you have more drives I think the routines given in the assembly listing will work. I have not been able to check them out completely however since my hardware is strapped to indicate one drive and the controller chip can not recognize any additional drives.

There are several problems that are worth mentioning. First. The stack pointer is set to $A07F by FLEX. BFD, however, uses $A07A to $A07F for a table.

To fix this problem use the code given under the labels INTPI1 ad INTPI2 in the listing. These routines set the stack to the proper location and then call a subroutine which initializes the PIA on the controller board. They then jump back to FLEX.

The MF-68, on the other hand, simply defines addresses in the I/O range for the registers used by the controller chip. The MF-68 does not require any PIA initialization.

I should mention here that if you have any additional software that sets the stack pointer at $A07F you should change it to $$A079. You must do this in both BASIC and in the TSC Assembler which you buy to run in FLEX.

There is only one other change that is required to make you the proud owner of a BFD/FLEX system. You must make a change in the NEWDISK utility or you won't be able to initialize a new disk.

To make these changes use the command, GET NEWDISK.CMD. Then, at location $0460 install the code:

```
STX $A07E  point to buffer
LDAA $08  and drive number
STAA $A07B
JMP $8032  the BFD write single track
           command.

ORG $0486
STAA $A07C  store the track number
JMP  $8035  call BFD Seek routine
```

After you have made the changes above, save a new copy as NEWDISK1.CMD on your FLEX system disk. The user should note here that I have made no other changes to the NEWDISK utility. The disk that you NEWDISK1 can be used and booted on anyones MF-68 system. Please note, however, that the code above makes it necessary for you to place the disk you want to intialize in drive number 0.

It is quite possible that you might want to write a new bootstrap loader routine, assemble it at 0534, then save it with the previous changes as NEWDISK2.CMD. You would then be able to boot directly into FLEX from a cold start. Since I considered that project a little over my head I decided to do it the easy way.

I use BFD's boot to load FLEX into low memory along with a move routine. DOS-68 is linked to the move routine, which automatically executes FLEX after the move is complete. See the assembly listing for details.

The only other thing you will need to get started is a copy of FLEX from SWTPC. You'll need to have a friend with an MF-68 system make you a tape copy of FLEX, $7000-$7FFF, or take your controller board and drive to his computer.

Using one method or the other get a copy of FLEX loaded at $7000. Then, move it into low memory and save it along with the move routine on a BFD-68 disk.

Finally, link the move routine to the DOS-68 boot. Then, when you type, J 8020, for a cold start you will wind up in FLEX if this disk is in drive number 0.

I hope you have been able to follow this conversion and will enjoy the benefits of using FLEX with the BFD-68. If you have any questions send me a SASE and I'll try to help. If you would rather not type in the new code I can supply the code shown in the assembly listing along with the single drive copy command on a cassette tape, KC standard format at 300 baud for $10. I will put both the source and the object on the tape.

I would be especially interested in hearing from anyone who writes a routine to boot FLEX directly.

```
                        NAM     SDC

                * by Dale L. Puckett
                  Chief Photojournalist
                * U. S. Coast Guard

                * 163 Farm Acre Road
                * Syracuse, New York  13210

                * November 16, 1978

                * EQUATES
7103            WARMS   EQU     $7103
710F            GETCHR  EQU     $710F
7112            PUTCHR  EQU     $7112
7118            PSTRNG  EQU     $7118
7127            GETFIL  EQU     $7127
713C            RPTERR  EQU     $713C
7803            FMSCLO  EQU     $7803
7806            FMS     EQU     $7806
7740            FCB     EQU     $7740

7600                    ORG     $7600
7600 20 05      SDC     BRA     SDC1
7602 01         VN      FCB     1
7603 00 00      SAVEX   FDB     0
7605 00 00      TWICE   FDB     0

7607 CE 77 40   SDC1    LDX     #FCB
760A BD 71 27           JSR     GETFIL
760D 25 1C              BCS     ERROR
760F CE 77 40           LDX     #FCB
7612 6D 0C              TST     12,X
7614 26 09              BNE     SDC2
7616 CE 76 DD           LDX     #EXTSTR
7619 BD 71 18           JSR     PSTRNG
761C 7E 71 03           JMP     WARMS

761F CE 77 40   SDC2    LDX     #FCB
7622 86 01              LDA A   #1
7624 A7 00              STA A   0,X
7626 BD 78 06           JSR     FMS
7629 27 09              BEQ     SDC3

762B BD 71 3C   ERROR   JSR     RPTERR
762E BD 78 03           JSR     FMSCLO
7631 7E 71 03           JMP     WARMS
```

```
7634 86 FF      SDC3    LDA A  #$FF       set for binary file
7636 A7 3B              STA A  59,X
7638 CE 01 00           LDX    #$100      point to memory
763B FF 76 03           STX    SAVEX
763E FF 76 05           STX    TWICE
7641 CE 77 40   LOOP    LDX    #FCB       read to memory
7644 BD 78 06           JSR    FMS
7647 26 18              BNE    ERROR1
7649 FE 76 03           LDX    SAVEX
764C 8C 3F FF           CPX    #$3FFF     check for end of memory, your
764F 27 08              BEQ    RPTMEM     address may be different here
7651 A7 00              STA A  0,X
7653 08                 INX
7654 FF 76 03           STX    SAVEX
7657 20 E8              BRA    LOOP

7659 CE 76 F0   RPTMEM  LDX    #MEMSTR
765C BD 71 18           JSR    PSTRNG
765F 20 CA              BRA    ERROR

7661 A6 01      ERROR1  LDA A  1,X        look for EOF error
7663 81 08              CMP A  #8
7665 26 C4              BNE    ERROR
7667 86 04              LDA A  #4         If it is close file
7669 A7 00              STA A  0,X
766B BD 78 06           JSR    FMS
766E 26 BB              BNE    ERROR
7670 CE 76 B6           LDX    #PROMPT    tell user to change disk
7673 BD 71 18           JSR    PSTRNG
7676 BD 71 0F           JSR    GETCHR     get go ahead from user
7679 CE 77 40           LDX    #FCB       open for write
767C 86 02              LDA A  #2
767E A7 00              STA A  0,X
7680 BD 78 06           JSR    FMS
7683 26 A6      ERROR5  BNE    ERROR
7685 86 FF              LDA A  #$FF       set to binary
7687 A7 3B              STA A  59,X

7689 FE 76 05   LOOP1   LDX    TWICE      write memory onto disk
768C BC 76 03           CPX    SAVEX
768F 27 10              BEQ    DONE
7691 A6 00              LDA A  0,X
7693 08                 INX
7694 FF 76 05           STX    TWICE
7697 CE 77 40           LDX    #FCB
769A BD 78 06           JSR    FMS
769D 26 E4              BNE    ERROR5
769F 20 E8              BRA    LOOP1

76A1 86 04      DONE    LDA A  #4         close file
76A3 CE 77 40           LDX    #FCB
76A6 A7 00              STA A  0,X
76A8 BD 78 06           JSR    FMS
76AB 26 D6              BNE    ERROR5
76AD CE 76 D1           LDX    #DONSTR
76B0 BD 71 18           JSR    PSTRNG
76B3 7E 71 03           JMP    WARMS      all done

                * Strings
```

```
76B6 43          PROMPT  FCC     'CHANGE DISK THEN HIT A KEY'
76D0 04                  FCB     4
76D1 46          DONSTR  FCC     'FILE COPIED'
76DC 04                  FCB     4
76DD 45          EXTSTR  FCC     'EXTENSION REQUIRED'
76EF 04                  FCB     4
76F0 4E          MEMSTR  FCC     'NOT ENOUGH MEMORY'
7701 04                  FCB     4

                         END     SDC

NO ERROR(S) DETECTED


 SYMBOL TABLE:
DONE   76A1  DONSTR 76D1  ERROR  762B  ERROR1 7661  ERROR5 7683
EXTSTR 76DD  FCB    7740  FMS    7806  FMSCLO 7803  GETCHR 710F
GETFIL 7127  LOOP   7641  LOOP1  7689  MEMSTR 76F0  PROMPT 76B6
PSTRNG 7118  PUTCHR 7112  RPTERR 713C  RPTMEM 7659  SAVEX  7603
SDC    7600  SDC1   7607  SDC2   761F  SDC3   7634  TWICE  7605
VN     7602  WARMS  7103

                     NAM    BFDFLEX

                 * These routines allow FLEX, written by
                 * Technical Systems Consultants, to run in the
                 * BFD-68 environment.

                 * Equates

A07C             BFDTRG  EQU   $A07C     Smoke Signal Track Register
A07D             BFDSRG  EQU   $A07D     Smoke Signal Sector Register
A07E             BFDSBU  EQU   $A07E     Smoke Signal Sector Buffer Pointer
8029             BFDRED  EQU   $8029     SSB Read Sector routine
8026             BFDINT  EQU   $8026     SSB Initialize PIA routine
A07B             BFDDRG  EQU   $A07B     SSB Drive Register
7145             FLXCOS  EQU   $7145     SSB Cold Start Address
715D             FLXWST  EQU   $715D     SSB Warm Start Address
802C             BFDWRT  EQU   $802C     SSB Write Sector Routine
7F11             XTEMP   EQU   $7F11
7F0F             FLXDRG  EQU   $7F0F     Flex Drive Register
8072             BFDRTR  EQU   $8072     SSB routine to read track register
7F13             TRKPTR  EQU   $7F13     Flex track holder pointer
8038             BFDRST  EQU   $8038     SSB Restore routine
7100             FLEX    EQU   $7100     Cold Start jump
A048             PC      EQU   $A048     Program Counter
2800             FLEXLO  EQU   $2800
A002             BEGADR  EQU   $A002
37FF             FLXLOE  EQU   $37FF
A004             ENDADR  EQU   $A004
7000             FLEXHI  EQU   $7000
A020             TARADR  EQU   $A020

                 * Read routine

7F17                     ORG   $7F17
7F17 B7 A0 7C    READ    STA A BFDTRG
7F1A F7 A0 7D            STA B BFDSRG
7F1D FF A0 7E            STX   BFDSBU
7F20 7E 80 29            JMP   BFDRED
```

```
                 * Initialization routine for PIA and
                   Stack Pointer

  7F23 BE A0 79  INTPI1  LDS    #$A079     Set stack below BFD registers
  7F26 8D B0 26          JSR    BFDINT     Go initialize PIA
  7F29 7E 71 4B          JMP    FLXCOS+3   Return to FLEX

  7F2C BE A0 79  INTPI2  LDS    #$A079
  7F2F 8D 80 26          JSR    BFDINT
  7F32 7E 71 60          JMP    FLXWST+3

                 * Routine which sets proper code for
                 * BFD Drive Select Register and
                 * places it in BFDDRG.

7F35 81 00     CMPDRN  CMP A  #0
7F37 26 06             BNE    CHK1
7F39 86 08             LDA A  #$0B
7F3B B7 A0 7B  RET10   STA A  BFDDRG
7F3E 39                RTS
7F3F 81 01     CHK1    CMP A  #1
7F41 26 04             BNE    CHK2
7F43 86 10             LDA A  #$10
7F45 20 F4             BRA    RET10
7F47 81 02     CHK2    CMP A  #2
7F49 26 04             BNE    DEFAUL
7F4B 86 20             LDA A  #$20
7F4D 20 EC             BRA    RET10
7F4F 20 EB     DEFAUL  BRA    RET10-2   Default to Drive #0

               * Patch within FLEX

7145                   ORG    FLXCOS
7145 7E 7F 23          JMP    INTPI1

715D                   ORG    FLXWST
715D 7E 7F 2C          JMP    INTPI2

               * Write routine

7F67                   ORG    $7F67
7F67 B7 A0 7C  WRITE   STA A  BFDTRG
7F6A F7 A0 7D          STA B  BFDSRG
7F6D FF A0 7E          STX    BFDSBU
7F70 7E 80 2C          JMP    BFDWRT

               * Verify routine

7F92                   ORG    $7F92
7F92 BD B0 29  VERIFY  JSR    BFDRED
7F95 C5 1B             BIT B  #$18
7F97 39                RTS

               * Restore Routine

7FA2                   ORG    $7FA2
7FA2 FF 7F 11  RESTOR  STX    XTEMP
7FA5 BD 7F D2          JSR    DRVSEL    Go select drive
7FA8 BD 80 38          JSR    BFDRST
7FAB FE 7F 11          LDX    XTEMP
7FAE 26 01             BNE    RES1
```

```
7FB0 39                 RTS
7FB1 17        RES1     TBA
7FB2 85 40              BIT A  #$40
7FB4 26 02              BNE    RES2
7FB6 0C                 CLC
7FB7 39                 RTS
7FB8 C6 0B     RES2     LDA B  #$0B      Write protected signal
7FBA 0C                 CLC
7FBB 39                 RTS

               * Drive Select
               * Note: Author has a single drive system
               * and has been unable to
                 test this routine.

7FD2                    ORG    $7FD2
7FD2 A6 03     DRVSEL   LDA A  03,X      Get Drive Number
7FD4 84 03              AND A  #$03      from FCB
7FD6 8D 15     DRV1     BSR    DRV2      Get Track holder pointer
7FD8 36                 PSH A
7FD9 BD 80 72           JSR    BFDRTR
7FDC 32                 PUL A
7FDD E7 00              STA B  00,X      Save current track
7FDF B7 7F 0F           STA A  FLXDRG    tell FLEX which drive
7FE2 BD 7F 35           JSR    CMPDRN    put drive number in BFDDRG
7FE5 8D 06              BSR    DRV2
7FE7 A6 00              LDA A  0,X       Restore track number
7FE9 B7 A0 7C           STA A  BFDTRG    and let BFD know location
7FEC 39                 RTS
7FED FE 7F 13  DRV2     LDX    TRKPTR    Point to track holder pointer
7FF0 F6 7F 0F           LDA B  FLXDRG    Get drive number in B-reg
7FF3 27 04              BEQ    RET1
7FF5 08        DRIV21   INX
7FF6 5A                 DEC B
7FF7 26 FC              BNE    DRIV21
7FF9 39        RET1     RTS

               * Routine to move FLEX from $2800 to $7000.
               * This routine is loaded as part of FLEX file
               * on BFD Disc.

               * It is linked to the DOS-68 boot routine
               * and will execute when the user
               * turns on the computer and types
               * J 8020.

2700                    ORG    $2700
2700 CE 71 00  MOVFLEX  LDX    #FLEX
2703 FF A0 48           STX    PC
2706 BF A0 42           STS    $A042
2709 CE 28 00           LDX    #FLEXLO
270C FF A0 02           STX    BEGADR
270F CE 37 FF           LDX    #FLXLOE
2712 FF A0 04           STX    ENDADR
2715 CE 70 00           LDX    #FLEXHI
2718 FF A0 20           STX    TARADR
271B FE A0 02           LDX    BEGADR
271E A6 00     MOV1     LDA A  0,X
2720 FE A0 20           LDX    TARADR
2723 A7 00              STA A  0,X
```

```
2725 08                  INX
2726 FF A0 20            STX     TARADR

2729 FE A0 02            LDX     BEGADR
272C BC A0 04            CPX     ENDADR
272F 27 06               BEQ     DONEMV
2731 08                  INX
2732 FF A0 02            STX     BEGADR
2735 20 E7               BRA     MOV1
2737 7E 71 00   DONEMV   JMP     FLEX

                         END

NO ERROR(S) DETECTED
```

SYMBOL TABLE:

BEGADR	A002	BFDDRG	A07B	BFDINT	8026	BFDRED	8029	BFDRST	8038
BFDRTR	8072	BFDSBU	A07E	BFDSRG	A07D	BFDTRG	A07C	BFDWRT	802C
CHK1	7F3F	CHK2	7F47	CMPDRN	7F35	DEFAUL	7F4F	DONEMV	2737
DRIV21	7FF5	DRV1	7FD6	DRV2	7FED	DRVSEL	7FD2	ENDADR	A004
FLEX	7100	FLEXHI	7000	FLEXLO	2800	FLXCOS	7145	FLXDRG	7F0F
FLXLOE	37FF	FLXWST	715D	INTPI1	7F23	INTPI2	7F2C	MOV1	271E
MOVFLE	2700	PC	A048	READ	7F17	RES1	7FB1	RES2	7FB8
RESTOR	7FA2	RET1	7FF9	RET10	7F3B	TARADR	A020	TRKPTR	7F13
VERIFY	7F92	WRITE	7F67	XTEMP	7F11				

TINY MUSIC

Noel J. Thompson
Hawaii Institute of Geophysics
2525 Correa Rd.
Honolulu, Hawaii 96822

* * * WOULD YOU LIKE TO MAKE YOUR 6800 GENERATE MUSIC? WOULD YOU LIKE TO DO IT CHEAPLY, GET STARTED QUICKLY, AND BE ABLE TO EXPAND TO BIGGER AND BETTER SOUNDS AS YOU SEE WHAT CAN BE DONE?

THEN PERHAPS YOU WOULD BE INTERESTED IN WHAT HAS BEEN DONE ON MY SYSTEM. HERE IS A WAY TO GO AT IT.

SINCE I'M AN ENGINEER, I'VE ALWAYS BUILT THE HARDWARE FIRST AND THE SOFTWARE LAST. THIS FLIES COUNTER TO THE TRADITIONS OF THE TOP - DOWN PROGRAMMER, BUT THATS THE WAY I AM.

HERE, BOTTOM - UP, IS THE WAY YOU CAN GO AT IT.

* * * THE HARDWARE * * *

THE GENERATION OF MUSICAL TONES WITH A MICROCOMPUTER CAN BE DONE WITH A DIGITAL - TO - ANALOG CONVERTOR, WHICH YOU CAN BUILD YOURSELF. IT IS NOT NECESSARY TO PUT MONEY INTO A D-A CONVERTOR BUG, OR TO SUPPLY THE MINUS POWER SUPPLIES THAT MAY GET INVOLVED IN DOING IT 'RIGHT'.

FIGURE I SHOWS A SIMPLE DIGITAL-TO-ANALOG (D-A) CONVERTOR. THIS SCHEMATIC ASSUMES YOU HAVE ALREADY ADDRESSED A MEMORY LOCATION FOR THE D-A FUNCTION, WITH A DECODING OF THE ADDRESS LINES. IN THIS CASE THE LOCATION IS BE19, AND EXECUTING A STORE-A-AT-BE18 (B7 F819) INSTRUCTION RAISES WIRE DE19 MOMENTARILY.

THE 8 BIT COMPUTER DATA DUSS IS SUPPLIED DIRECTLY TO THE EIGHT INPUT PORTS OF A 74100 3-BIT LATCH, WHICH GRABS EIGHT DATA BITS WHEN PINS 12 AND 23 ARE HIGH, AND RETAINS THEM INDEFINITELY WHEN PINS 12 AND 23 ARE LOW. THESE BITS ARE PRESENTED TO THE EIGHT OUTPUT PINS.

IF YOU ALREADY HAVE AN 8-BIT PORT AVAILABLE, YOU ALREADY HAVE EVERYTHING BUT THE RESISTORS. CONFIGURE THE PORT FOR OUTPUT, AND TIE THE RESISTOR NETWORK TO THE EIGHT WIRES.

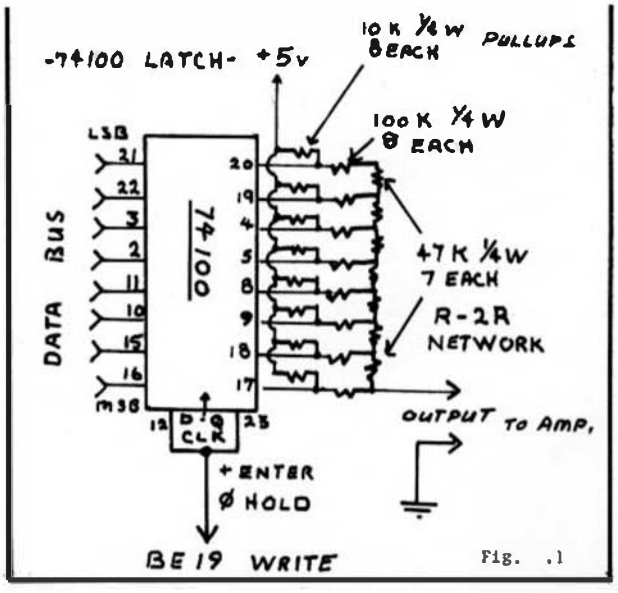


Fig. .1

THE GROUP OF 47K AND 100K RESISTORS CONSTITUTE WHAT IS KNOWN AS AN 'R - 2R LADDER NETWORK'. A VOLTAGE APPLIED TO ANY INPUT TERMINAL HAS HALF THE EFFECT ON THE OUTPUT VOLTAGE AS DOES A VOLTAGE APPLIED TO THE NEXT INPUT TERMINAL TOWARD THE OUTPUT. THUS WHEN A BINARY NUMBER IS APPLIED TO THE EIGHT INPUT TERMINALS, FROM THE 74100, A VOLTAGE APPEARS AT THE OUTPUT WHICH IS PROPORTIONAL TO THE BINARY NUMBER. FULL SCALE OUTPUT WITH 5 VOLT INPUTS IS ABOUT 2 1/2 VOLTS. LOADING ON THE OUTPUT DOESN'T CHANGE THE PROPORTIONS OF THE NETWORK, SO MOST ANY HOME MUSIC AMPLIFIER CAN BE FED FROM THE OUTPUT WITH PLENTY OF VOLUME, REGARDLESS OF THE AMPLIFIER CHARACTERISTICS.

TTL EQUIPMENT SUCH AS THE 74100 DOES NOT PUT OUT A FULL FIVE VOLTS WHEN HIGH, DUE TO THE INTERNAL CONFIGURATION OF TRANSISTORS. TTL DOES, HOWEVER, SUPPLY A HIGH IMPEDANCE OUTPUT WHEN THE OUTPUT IS PULLED HIGHER THAN THE NORMAL 4.3 VOLT OUTPUT. THUS THE 1K RESISTORS CAN PULL THE 'HIGH' OUTPUT UP TO 5 VOLTS. THIS MAKES ALL 'HIGH' OUTPUTS EQUAL, IN SPITE OF DIFFERENCES BETWEEN THE VARIOUS TTL OUTPUT STAGES.

THESE PULL-UP RESISTORS ARE NOT NECESSARY, HOWEVER. LEAVE SPACE FOR THEM AND TRY THE THING WITHOUT. YOU WON'T NEED PULL-UPS UNLESS YOU DECIDE YOU WANT PRECISE VOLTAGES TO COME OUT. THIS ISN'T NECESSARY FOR MUSIC.

A REQUIREMENT OF SUCH A D-A CONVERTOR IS THAT THE RESISTOR VALUES IN THE 'R - 2R NETWORK' BE ENOUGH ALIKE SO THAT THE OUTPUT FROM A BINARY 10000000 WILL BE THE PROPER ONE-BIT GREATER THAN THE OUTPUT FROM A BINARY 01111111. WITH A RESOLUTION OF 8 BITS, THE RESISTORS CLOSE TO THE OUTPUT SHOULD BE WITHIN A PART IN 256.

HOWEVER, FOR THIS PROJECT, PICK A BUNCH OF RESISTORS FROM THE SAME BATCH AND PUT THEM IN. DONT WORRY ABOUT TRIMMING THEM. YOU'LL NEVER HEAR THE ERRORS, AS THEY WILL HAVE AN EFFECT ONLY ON THE OUTPUT WAVESHAPE, NOT ON ITS FREQUENCY.

THINGS WOULD BE DIFFERENT IF YOU WERE TALKING ABOUT TWELVE BIT OUTPUT. THEN MUCH MORE ACCURACY WOULD BE REQUIRED.

A CAPACITOR, ANYTHING OVER ABOUT ONE MICROFARAD, CAN BE USED TO COUPLE THE OUTPUT TO AN AUDIO AMPLFIER, IN CASE THE AMPLIFIER DOESNT LIKE THE OFFSET IN CENTER VOLTAGE PRESENTED BY A D-A CONVERTOR WHOSE OUTPUT IS ALWAYS POSITIVE VOLTAGES.

* * * SINE GENERATOR SOFTWARE * * *

SO FAR WE ONLY HAVE A DC VOLTAGE WHICH CAN BE CHANGED UNDER PROGRAM CONTROL. WE DO NOT HAVE AN OSCILLATOR, SO OUR D-A CAN'T YET MAKE MUSIC. BUT THE REST IS ALL SOFTWARE.

CHANGING THE VOLTAGE FROM THE D-A CONVERTOR AT A RAPID RATE CAN MAKE AN AUDIO TONE. IT WOULD BE NICE TO MAKE THE TONE A SINE - WAVE BY CHANGING THE VOLTAGE IN THE PROPER MANNER.

AFTER YOU MAKE THE D-A CONVERTOR WORK, AND WRITE A FEW TEST ROUTINES TO OUTPUT VOLTAGES, THEN SQUEAKS AND BUZZES, TO YOUR AMPLIFIER, YOU ARE READY FOR A SINE GENERATOR, TO MAKE NICE PURE SINE WAVE TONES.

EVEN IF YOUR HIGHER-LEVEL LANGUAGE HAS A SINE FUNCTION, THIS FUNCTION IS NOT APPROPRIATE TO GENERATING AUDIO TONES. IT IS FAR TOO SLOW. YOU NEED TO GO OUT WITH A CALL OR USR FUNCTION AND RUN A LITTLE MACHINE-LANGUAGE ROUTINE FOR THIS.

BELOW IS A SINE TABLE. ACTUALLY IT'S HALF A COSINE, STARTING AT THE LOWEST NUMBER AND ENDING AT THE HIGHEST NUMBER. THE ADDRESS INTO THE TABLE IS THE 'PHASE' NUMBER, INDICATING HOW FAR ALONG THE CURVE WE WANT TO PICK A VOLTAGE. THE DATA REPRESENTS THE VOLTAGE TO BE GENERATED, IN HEX. NOTE THAT HALF WAY THRU THE TABLE APPEARS HEX 80, OR MID-SCALE, WITH ADJACENT NUMBERS CHANGING AT A RATHER RAPID RATE, WHILE AT THE BEGINNING AND END OF THE TABLE, THE CHANGE FROM NUMBER TO NUMBER IS MORE GRADUAL.

```
2000   01 01 01 02 03 04 06 08
2008   0A 0D 10 13 16 1A 1E 22
2010   26 2A 2F 34 39 3F 44 4A
2018   4F 55 5B 61 67 6E 74 7A
2020   80 86 8C 93 99 9F A5 AB
2028   B1 B6 BC C1 C7 CC D1 D6
2030   DA DE E2 E6 EA ED F0 F3
2038   F6 F8 FA FC FD FE FF FF
```

* * * THE SINE PROGRAM * * *

THE CODE SHOWN BELOW IS ALL THAT IS REQUIRED TO GENERATE TONES. OUR SYSTEM USES NICELY DEDICATED MEMORY LOCATIONS FOR THE VARIABLES A THRU Z IN TOM PITTMAN'S TINY BASIC. TINY-A IS LOCATED AT TWICE ITS BINARY VALUE, AT HEX 82 AND 83. TINY-B IS LOCATED AT HEX 84 AND 85. THUS WHEN CALLING THIS ROUTINE, WE PUT THE DESIRED MUSICAL PITCH IN TINY-A, AND THE NOTE LENGTH IN TINY B, AND RUN THIS SUBROUTINE.

```
2040 8D 00    !GOSUB.  ! FINDING WHERE
2042 30       !TSX     ! THE P REGISTER
2043 31       !INS     ! IS, TO FIND
2044 31       !INS     ! THE SINE TABLE.
2045 EE 00    !LDX,00
2047 FF 0004  !STX 0004, TABLE BASE

204A 96 83    !LDAA 83, PITCH VALUE
204C 9B 03    !ADD PREVIOUS PHASE ANGLE
204E 97 03    !STORE NEW PHASE
2050 2A 01    !BRANCH PLUS
2052 43       !IF NEGATIVE, COMPLEMENT A
2053 44       !SHIFT RIGHT,TABLE IS
              !  1/4 OF 256 BYTES.
2054 77 05    !STORE TABLE INDEX
2056 FE 0004  !LDX TABLE + INDEX
2059 A6 00    !LOAD SINE FROM TABLE
205B B7 BE19  !SEND TO D/A CONVERTOR
205E DE 84    !LDX LENGTH OF TONE
2060 27 0     !JUMP OUT IF ZERO
2062 09       !DECREMENT X
2063 DF 84    !STORE LENGTH
```

```
2065 20 E3    !AGAIN AT 204A
2067 39       !RETURN
```

THIS SINE GENERATOR ROUTINE IS FULLY RELOCATABLE IF IT IS CONTIGUOUS TO THE SINE TABLE, AND IF THE SINE TABLE BEGINS ON AN EVEN MULTIPLE OF 100 HEX. THE FIRST SIX COMMANDS IN THE ROUTINE FIND THE LOCATION OF THE PROGRAM, AND STORE THE HIGH BITS IN 0004. LOCATIONS 0003, 0004 AND 0005 ARE USED FOR TEMPORARIES.

ONLY THE LOW BYTE OF TINY-A IS USED FOR PITCH. THE ROUTINE ADDS THIS BYTE (LOCATION 83) TO THE PREVIOUS PHASE (LOCATION 0003) TO DECIDE HOW FAR ALONG THE SINE TABLE TO MOVE FOR EACH INCREMENT. THIS NUMBER IS SAVED FOR THE NEXT PASS.

IF THE RESULTING PHASE IS IN THE SECOND HALF OF THE RANGE OF 0 TO 255, IT IS NEGATED. THIS FOLDS THE SINE TABLE BACK ON ITSELF, CREATING THE ENTIRE WAVEFORM WITH ONLY HALF A TABLE.

THE PHASE IS THEN SHIFTED RIGHT (DIVIDED BY TWO) TO REDUCE THE TABLE EVEN FURTHER TO 64 BYTES. THE ACCURACY OF WAVEFORM IS DEGRADED A NEGLIGIBLE AMOUNT BY THIS. THE PITCH RESOLUTION IS NOT DEGRADED, SINCE ALL EIGHT BITS WERE SAVED FOR THE NEXT PASS.

THE TABLE POINTER IS STORED AND RETRIEVED ALONG WITH THE TABLE INDEX BYTE, AND THE VALUE FROM THE SINE TABLE IS RETRIEVED. THE SINE IS SIMPLY SENT TO THE D/A CONVERTOR.

THE ROUTINE CYCLES FOR THE NUMBER OF COUNTS IN TINY-B, GIVING A CONSTANT LENGTH OF TONE FOR A GIVEN TINY-B, REGARDLESS OF THE PITCH. EACH PASS THRU THE BASIC ROUTINE TAKES ABOUT 50 MICROSECONDS, GIVING A 20 KILOCYCLE SAMPLING RATE. THIS IS TOO FAST FOR SOME PURPOSES, SO A DELAY CAN BE ADDED TO LOWER THE PITCH RANGE. IF YOU GO TOO FAR WITH THIS, THE SAMPLING RATE ITSELF WILL BE HEARD, AS WILL BEAT NOTES BETWEEN THE SAMPLING RATE AND THE DESIRED FREQUENCY, SO PERHAPS YOU DON'T WANT TO SLOW THE SAMPLING RATE BELOW 10 KILOCYCLES OR SO (100 MICRO- SECONDS PER LOOP).

* * * THE SPECTRUM OF PITCHES * * *

A DIGITAL DESIGNER WORKING WITH RADIO RUNS UP AGAINST A FREQUENCY PROBLEM, BECAUSE THE EASY THING TO DO WITH DIGITAL CIRCUITS IS TO DIVIDE, NOT MULTIPLY.

WHEN SYNTHESIZING A FREQUENCY BY DIVIDING FROM A MUCH HIGHER FREQUENCY CRYSTAL, SUCH AS TEN MEGACYCLES, THE NUMBERS AVAILABLE FROM A DIVIDE CHAIN ARE FOUND TO BE FRACTIONS, NOT NICE ROUND INTEGERS. MUSIC IS BUILT OF HARMONICS OF A FUNDAMENTAL NOTE, NOT SUBHARMONICS OF A HIGH NOTE.

IN THIS CASE, THE PROBLEM IS SOLVED BY MULTIPLYING INSTEAD OF DIVIDING. ALL FREQUENCIES PRODUCIBLE BY THE SOFTWARE ARE MULTIPLES OF, RATHER THAN FRACTIONS OF, A FUNDAMENTAL FREQUENCY. THAT FUNDAMENTAL IS THE FREQUENCY PRODUCED WHEN TINY-A = 1, AND THE LOOP MUST STEP THRU 256 STEPS TO GENERATE A CYCLE.

WHEN TINY-A = 3, THE PROGRAM STEPS THRU THE SINE TABLE THREE STEPS AT A TIME, GOES THRU THE WHOLE SINEWAVE THREE TIMES AS FAST, AND GENERATES A PITCH THREE TIMES AS HIGH, OR THE MUSICAL DOMINANT, ONE AND ONE HALF OCTAVE ABOVE THE FUNDAMENTAL.

AS TINY-A BECOMES HIGHER, ALMOST ALL NOTES IN THE MUSICAL SCALE APPEAR.

WITH A 20,000 CYCLE PER SECOND LOOP RATE, 20,000 DIVIDED BY 256 GIVES 78 CYCLES PER SECOND, APPROXIMATELY A LOW MUSICAL 'E'. ALL PITCHES PRODUCABLE ARE HARMONICS OF THIS 'E', AND THUS MUSIC CAN BE PRODUCED IN THE KEY OF 'E' BY SELECTION OF APPROPRIATE HARMONICS.

FOR EXAMPLE, IF TINY-A IS SUCCESSIVELY 4, 5, AND 6, A MAJOR TRIAD TWO OCTAVES ABOVE THE FUNDAMENTAL 'E' WILL BE PRODUCED, THE NOTES KNOWN AS E, G-SHARP AND D.

AHA, YOU SAY, 'A = RND(3)+4' WILL PRODUCE NOTES IN THIS CHORD.

* * * NOW MAKE SOME NOTES * * *

TRY SETTING A = NUMBERS FROM 1 TO 16, WITH B ABOUT 2000 DECIMAL, FOR A REASONABLE LENGTH OF NOTE.

```
10  B=2000      ! LENGTH OF TONE
20  A=1         ! STARTING PITCH, LOW
30  X = USR(H256)  ! OR HOWEVER
                   !YOU CAN GOSUB HEX 2040
40  A=A + 1
50  IF A = 17 THEN A = 1
60  GOTO 30
```

THE RESULT SHOULD BE A NICE LITTLE RUN OF NOTES UP THE SCALE, ABOUT FOUR NOTES PER SECOND, REPEATING AFTER SIXTEEN NOTES.

IF SO, YOU ARE OFF AND RUNNING. PLAY WITH A AND B AND FIND OUT WHAT YOU CAN DO WITH IT.

* * * ITS TIME FOR MUSIC * * *

FOR A FIRST ATTEMPT, LETS MAKE SOME RANDOM MUSIC ON JUST FIVE NOTES. THE FOURTH THRU EIGHTH HARMONICS OF THE FUNDAMENTAL FORM A CHORD KNOWN AS A MAJOR 7TH CHORD, SO PLAY WITH THEM, WITH THE HELP OF A RANDOM NUMBER GENERATOR.

```
 90  REM  MUSIC!
100  B=3000           ! LONGER NOTE
110  A = RND(5) + 4   ! CHOOSE NOTE
120  X = USR(H256)    ! PLAY NOTE
130  GOTO 110
```

THIS WILL PLAY A CONTINUOUS MELODY ON THE CHORD.

YOU MAY NOTICE THAT THE 7TH IN THE CHORD SOUNDS A BIT OFF PITCH. THE FACT IS, HOWEVER, THAT YOU ARE BEING EXPOSED TO THE TRUE 7TH CHORD, PERHAPS FOR THE FIRST TIME. WHEN A = 7, THE PITCH IS TRULY SEVEN TIMES THAT AT THE FUNDAMENTAL. WE ARE SO USED TO HEARING THE 'EVEN - TEMPERED' SOUND, TO WHICH OUR KEYBOARD INSTRUMENTS ARE TUNED AS A

COMPROMISE, THAT THE TRUE MAJOR 7TH CHORD SOUNDS A BIT OFF WHEN WE FIRST HEAR IT.

* * * LETS IMPROVE THE MUSIC * * *

MUSIC IS SUPPOSED TO HAVE MELODY, HARMONY, RHYTHM AND FORM. WE ARE WELL ON THE WAY TO HAVING MELODY WITH THE LITTLE PROGRAM ABOVE. THE NEXT PROGRAM ADDS RHYTHM AND FORM, AND SINCE IT STILL PLAYS ONLY ONE NOTE AT A TIME, IT CAN NOT HAVE HARMONY IN THE FORM SUPPLIED BY MULTIPLE INSTRUMENTS, BUT TO THE EXTENT WE PLAY A GROUP OF NOTES WHICH ARE ALL IN THE SAME CHORD, IT DOES HAVE HARMONY.

HARMONY IS IMPLIED IN MUSIC BY CHOOSING NOTES WITHIN A CHORD FOR A PERIOD OF TIME, SUCH AS A MEASURE. HERE, IN DO-RE-MI NOTATION, ARE THE THREE PRIMARY CHORDS USED IN MUSIC:

```
SCALE          DO RE MI FA SO LA TI DO
TONIC          DO    MI    SO
SUB-DOMINANT            FA    LA    DO
DOMINANT                   SO    TI    RE
```

NOTE THAT ALL EIGHT NOTES (THE WHITE KEYS) APPEAR WITH THE USE OF ONLY THREE CHORDS.

FORM IS SUPPLIED BY BREAKING THE SEQUENCE OF NOTES INTO GROUPS OF FOUR. THE VARIABLE M INDICATES NOTES REMAINING IN A MEASURE. WHEN IT REACHES ZERO, IT IS RESET TO FOUR.

FORM IS FURTHER SUPPLIED BY STOPPING AFTER 32 MEASURES OF FOUR NOTES EACH, COUNTED BY THE VARIABLE T.

TO MAKE THE FORM INTERESTING, WE CHANGE THE CHORD FROM WHICH TO CHOOSE OUR NEXT FOUR NOTES, AT THE BEGINNING OF EACH MEASURE OF FOUR NOTES.

RHYTHM IS IMPROVED BY PERMITTING A RANDOM CHOICE OF THE LENGTH OF THE NOTE, WITH THE CONSTRAINT THAT NO NOTE IS ALLOWED TO EXTEND ACROSS A BOUNDARY INTO THE NEXT MEASURE. THIS IS DONE WITH:

```
B = RND(M) + 1
```

WHERE B IS THE LENGTH OF THE NEXT NOTE, AND M IS THE NUMBER OF BEATS LEFT IN THE MEASURE.

SINCE B VALUES OF 1 THRU 4 ARE WAY TOO SMALL FOR OUR TONE-GENERATOR, (WE USED VALUES OF 2000 AND 3000 IN THE FIRST PROGRAMS ABOVE) B IS THEN MULTIPLIED BY 4000. THEN 2800 IS SUBTRACTED FROM B. THIS IS AN ARTIFACT OF THE PARTICULAR DOUBLE - INTERPRETER BASIC WE USE. THE BASIC IS SO SLOW, THAT A PAUSE IS CLEARLY AUDIBLE BETWEEN NOTES. SUBTRACTING 2800 MAKES A SERIES OF ONE-BEAT NOTES TAKE THE SAME AMOUNT OF TIME AS A SINGLE FOUR-BEAT NOTE. THIS NUMBER WOULD BE DIFFERENT ON A DIFFERENT BASIC. PERHAPS IT MAY BE ZERO IF YOU HAVE A FAST INTERPRETER.

HERE IS THE PROGRAM MUSIC2.

```
200 REM MUSIC2
210 T = 0    ! BEGINNING OF TUNE
220 K = 1 ! FIRST OF 4 MEASURES
230 M = 4 ! FOUR NOTES IN THIS MEASURE
300 REM WHICH CHORD FOR THIS MEASURE
310 IF K=1 THEN J=12 !TONIC CHORD
320 IF K=2 THEN J=16 !SUBDOMINANT CHORD
330 IF K=3 THEN J=18 !HIGH DOMINANT
340 IF K=4 THEN J=9  !LOW DOMINANT
350 REM CHOOSE NOTE IN THAT CHORD
360 D = RND(3) + 4
370 A = D * J / 2
380 IF A>40 THEN A=A/2 !TOO HIGH
390 IF A=22 A=23 ! ILLEGAL PITCH
400 REM CHOOSE NOTE LENGTH
410 B=RND(M) + 1
420 M = M - B !REMAINING BEATS
430 IF M=1 IF B=1 THEN A=0 !THROW IN
     OCCASONAL REST AT END OF MEASURE
440 B = B*4000 - 2800 !REAL NOTE LENGTH
600 X = USR(8256) !SEND THE NOTE
610 IF M=0 THEN M=4 ! NEXT MEASURE
620 IF M=4 THEN K=K+1 !NEXT HARMONY
630 IF K=5 THEN K=1   !RECYCLE HARMONY
640 IF M=4 THEN T=T+1 !COUNT MEASURES
700 IF T < 32 GOTO 300 !CONTINUE
710 STOP  ! THE TUNE IS OVER
720 B=16000 ! PLAY A FINAL NOTE
730 A=24    ! ON THE DOMINANT
740 X=USR(8256)
750 REM THE TUNE IS FINISHED.
760 B=20000 ! PAUSE A BIT
770 A=0     ! WITH NO TONE
780 X=USR(8256)
800 GOTO 200 !START A NEW TUNE
```

MUSIC2 PLAYS A 32 MEASURE TUNE, THEN PAUSES AND STARTS A NEW ONE.

THE RULES IN THIS TUNE ARE THAT THERE ARE 32 MEASURES, IN GROUPS OF FOUR. THE FIRST MEASURE OF FOUR USES NOTES RANDOMLY CHOSEN FROM THE TONIC CHORD (J=12). THE SECOND MEASURE USES THE MUSICAL FOURTH CHORD, OR SUB-DOMINANT. THE THIRD AND LAST USE THE MUSICAL FIFTH CHORD, OR DOMINANT.

NOTICE THE VALUES OF J, 12 FOR THE TONIC, 16 FOR THE SUBDOMINANT, AND 18 OR 9 FOR THE DOMINANT. THESE THREE CHORDS, AROUND WHICH MOST MUSIC IS WRITTEN, HAVE SIMPLE PITCH RATIOS TO EACH OTHER, WITH THE DOMINANT AT 3/4 OF THE TONIC, AND THE SUBDOMINANT AT 4/3 OF THE TONIC. THE INTEGERS 9, 12 AND 16 ARE THE SMALLEST INTEGERS SATISFYING THIS RELATIONSHIP.

THE ACTUAL NOTE PLAYED IS EITHER THE 4TH, 5TH OR 6TH HARMONIC OF THE BASE J FOR THE GIVEN MEASURE, AS CALCULATED BY:

```
360 D = RND(3) + 4
370 A = D * J / 2
```

THE OCCASIONAL DIVISION BY TWO IS TO PREVENT THE NOTES FROM BEING TOO HIGH PITCHED TO BE PLEASING.

* * * THAT'S A FUNNY PROGRAM * * *

WELL YES. IT IS WRITTEN IN A TINY BASIC, HAVING ONLY INTEGERS, NO ARRAYS, NO MULTIPLE STATEMENTS PER LINE. IF YOU HAVE A FANCIER BASIC, YOU CAN SHAPE UP THE PROGRAM. PLEASE DO.

* * * A FANCIER VERSION * * *

MY DAUGHTER, SHARON, HEARD MUSIC2 AT WORK AND DECIDED TO PUSH MORE MUSIC THEORY INTO IT. ONE OF HER PROGRAMS CHOOSES AMONG FIVE SEQUENCES OF HARMONIES, EACH FOUR MEASURES LONG. SINCE SHE DIDN'T KNOW COMPUTERS, SHE WASN'T BOTHERED BY THE LACK OF ARRAYS. SHE JUST SLUGGED IT OUT WITH WHAT TINY

THE EDITOR-

The only microprocessor editor with all the features and ease of use normally found only on large machines. "THE EDITOR" lets you fully use the CT-82's capabilities.

LINE POINTER —Now you understand why the CT-82 has 82 columns. The left two columns are used for a line pointer, which indicates the line of text being edited.

FILE WRAPAROUND—"THE EDITOR" may make multiple passes over the file being edited without restarting the editor.

AUTOMATIC CARRIAGE RETURN—The last word in a line will automatically be started on the next line if it will not fit in the space remaining on the line.

SIMPLE COMMANDS—Commands consists of a single letter, or a key press on the cursor pad. No complicated format to be learned and remembered.

MULTIPLE COMMANDS and REPEATS—Command line may have more than one command. "THE EDITOR" will execute command strings sequentially. Repeat function allows changes in a string through the text file.

SOURCE TEXT TABS—Tab stops appropriate for source text input may be set to operate from the space bar, or any other key.

SHIFT INVERSION—The keyboard may be set to produce either capital, or lower case letters when shift is used.

SCREEN POSITIONING—Scroll up, scroll down, line pointer up, line pointer down, home file, top of memory, bottom of memory, move relative to pointed line and form feed are provided.

"THE EDITOR" is available only for Southwest Technical Products computer systems using the CT-82 and running under FLEX-5®, or FLEX-8® operating systems. It may be used to edit any files, or programs compatible with the DOS, except binary files. Edited files are compatible with the TSC Text Processing program. The combination makes a powerful and inexpensive word processing system.

Editor FLEX-5® or FLEX-8®. . . . . $25.00 ppd. in Continental USA

®FLEX is a registered trademark of TSC Inc.

SOUTHWEST TECHNICAL PRODUCTS CORPORATION
219 W. Rhapsody
San Antonio, Texas 78216 (512) 344-0241

COULD DO, AND CAME UP WITH THIS:

* * * MUSIC3 * * *

```
100 N = 1 ! NOTE
150 P = 0 ! PHRASE
160 C=8256 !CONVERTOR LOCATION
180 IF N=2 B=4
200 K=1 ! MEASURE

250 REM START 4 MEASURE PHRASE
255 P=P+1
260 REM ONLY 8 PHRASES PER TUNE
265 IF P=9 GOTO 2000
300 S=RND(5) + 1 !CHOOSE CHORDS
301 IF S=1 PRINT"DO FA SO DO"
302 IF S=2 PRINT"DO LA FA SO"
303 IF S=3 PRINT"DO DO*FA SO"
304 IF S=4 PRINT"DO LA*RE TI"
305 IF S=5 PRINT"DO LA*RE*SO"
307 ! CHOOSE RHYTHM
308 R=RND(5) + 1
310 A=48

500 D=RND(3) !NEW NOTE PITCH
502 IF K=1 IF N=1 GOTO 510
505 A=48+12*D
510 IF K=3 GOTO R*10+1800
512 B=1
513 IF N=R B=2

516 IF B=0 GOTO 530
520 LET B=B*4000-3000 !NOTE LENGTH
530 IF K>1 GOTO S*100+1000

1000 REM SEND NOTE
1005 X=USR( C )
1010 N=N+1
1020 IF N=6 N=1
1030 IF N>1 GOTO 500
1040 K=K+1 ! ANOTHER MEASURE
1050 IF K=5 K=1
1060 IF K=1 GOTO 250 !NEW PHRASE
1070 GOTO 500

1100 IF K=2 A=64+16*D
1110 IF K=3 A=36+9*D
1120 IF K=4 A=48+12*D
1130 GOTO 1000

1200 IF K=2 A=40+8*D+4*(D/2)
1210 IF K=3 A=64+16*D
1220 IF K=4 A=36+9*D
1230 GOTO 1000

1300 IF K=2 A=48+12*D-29*(D/2)
1310 IF K=3 A=64+16*D
1320 IF K=4 A=36+9*D
1330 GOTO 1000

1400 IF K=2 A=80-29*D+50*(D/2)
1410 IF K=3 A=54+10*D+7*(D/2)
1420 IF K=4 A=45+9*D
1430 GOTO 1000

1500 IF K=2 A=80-29*D+50*(D/2)
1510 IF K=3 A=54+14*D+34*(D/2)
1520 IF K=4 A=36+9*D
1530 GOTO 1000

1800 ! CHOOSE 3RD MEASURE RHYTHM
     !DEPENDING ON R
1801 ! EITHER 4-2, OR 4-1-1, OR 2-4,
1802 ! OR 1-1-4, OR 2-2-2

1810 IF N=1 B=4    ! 4 2 RHYTHM
1814 IF N=1 GOTO 516
1815 B=2
1816 N=5
1818 GOTO 516

1820 IF N=1 B=4    ! 4 1 1 RHYTHM
1821 IF N=1 GOTO 516
1822 IF N=2 B=1
1823 IF N=2 GOTO 516
1824 B=1
1825 N=5
1826 GOTO 516

1830 IF N=1 B=2    ! 2 4 RHYTHM
1831 IF N=1 GOTO 516
1832 B=4
1833 N=5
1834 GOTO 516

1840 IF N=1 B=1    ! 1 1 4 RHYTHM
1841 IF N=1 GOTO 516
1842 IF N=2 B=1
1843 IF N=2 GOTO 516
1844 B=4
1845 N=5
1846 GOTO 516

1850 B=2      ! 2 2 2 RHYTHM
1851 IF N<3 GOTO 516
1853 N=5
1854 GOTO 516

2000 A=48
2010 B=15200
2020 X=USR( C )
2030 A=0 ! NO TONE
2040 B=30000 ! FOR A WHILE
2050 X=USR( C )
2060 PRINT
2070 GO TO 100 ! AND START A NEW TUNE.
```

IN THIS PROGRAM, MUSIC3, THERE ARE 6 BEATS (N) IN A MEASURE, FOUR MEASURES (K) IN A PHRASE, AND EIGHT PHRASES (P) IN A TUNE.

THE CHORD SEQUENCE FOR A PHRASE IS CHOSEN FIRST (LINES 300 - 305). IT IS ACTED UPON IN THE ROUTINES IN LINES 530 AND LINES 1000 - 1530.

THE RHYTHM FOR MEASURES 1, 2 AND 4 ALWAYS CONTAINS FIVE NOTES. FOUR OF LENGTH ONE, AND ONE OF LENGTH TWO, AS CHOSEN BY LINES 308 AND LINES 512 - 513.

THE RHYTHM FOR MEASURE 3 ALWAYS CONTAINS FOUR NOTES, CHOSEN BY LINES 308, 510 AND 1800 - 1854. AT THE END OF SUCH A MEASURE, N IS SET TO FIVE TO CONVINCE LINE 1050 THAT THE MEASURE IS FINISHED.

MUSIC3 PLAYS INTERESTING MUSIC. NEXT TIME, I'LL SHOW SHARON ABOUT SUBROUTINES AND ARRAYS AND THE SAME THING WILL BE DONE IN HALF THE SPACE.

IF YOU WOULD LIKE TO PURSUE THIS WITH SHARON, WRITE:

SHARON THOMPSON
2306 HWY. AB,
MCFARLAND WISCONSIN 53558

* * * CONCLUSIONS * * *

MUSIC CAN BE INVENTED BY COMPUTER, AND PLAYED AS IT IS GENERATED. THE RULES OF MELODY LINES ARE KNOWN, AND CAN BE INCORPORATED INTO A COMPUTER PROGRAM.

AS THIS IS BEING WRITTEN, I AM LISTENING TO THE PROGRAM MUSIC2. IT IS INTERESTING ENOUGH TO RUN FOR HOURS WITHOUT DRIVING ME UP THE WALL.

* * * WHAT NEXT? * * *

ONE CONSTRAINT NOT MENTIONED COMES FROM THE HUMAN VOICE MECHANISM. DUE TO THE DIFFICULTY OF TIGHTENING OR LOOSENING THE VOICE MECHANISM RAPIDLY, MOST MUSIC DOES NOT MAKE JUMPS FROM NOTE TO NOTE OF MORE THAN AN OCTAVE. SUCH A CONSTRAINT CAN BE ADDED BY SAVING THE PREVIOUS PITCH, COMPARING IT WITH THE NEXT PROPOSED PITCH, AND MOVING THE NEW PITCH AN OCTAVE (DIVIDE OR MULTIPLY A BY 2) IF THE JUMP IS TOO GREAT.

THE SINE GENERATOR PROGRAM COULD BE EXPANDED TO HANDLE TWO NOTES AT ONCE, OR THREE. THEN IT COULD PLAY CHORDS. FOR TWO NOTES AT ONCE, I WOULD SUPPLY TWO PITCHES, CALCULATE TWO SINES, DIVIDE EACH BY TWO, AND ADD THEM TOGETHER. A COMPOSITE WAVE WOULD BE GENERATED WITH THE PROPER TWO-PITCH SOUND.

SMOKE SIGNAL BROADCASTING INTRODUCES

NEW 6800-BASED MICROCOMPUTER

HOLLYWOOD, CA ... A new high performance dual-floppy microcomputer, featuring SS-50 BUS compatibility and a new controller design, has been introduced by Smoke Signal Broadcasting, designers and manufacturers of small business computer systems.

The new microcomputer allows up to 60K of usable memory by adding two more slots. Disk storage can also be increased to four mini-floppies or four 8-inch floppies.

Price for the "CHIEFTAIN" microcomputer is $2,595 retail.

For more information contact Ed Martin, Smoke Signal Broadcasting, 6304 Yucca Street, Hollywood, CA 90028. (213) 462-5652

SEMICONDUCTOR Memory Primer

A Two Part Series
Part I

Don Kinzer
3885 NW Columbia Ave.
Portland, OR 97229

Read-write memory is one of the most important components of your computer system. Whether you are choosing which type and manufacturer of memory system to buy or are preparing to design your own, a better understanding of principles of operation and system design techniques will enable you to make the right decisions.

Before proceeding we must deal with the problem of semantics. The commonly used jargon associated with memory systems unfortunately is not precise and therefore leads to some misunderstanding of basic principles.

The manner in which a particular memory cell is accessed is described by the terms random access and sequential access. Random access implies that the latentcy time, or the time required to access any particular cell, is nearly constant with respect to address. Sequential access means that one must wait a varying amount of time to access a particular cell dependent upon the location being accessed. Figure 1 gives examples of each of these classes.

Volatility describes the lack of ability of a storage device to retain information with power removed. A non-volatile device will retain information with power removed. Figure 2 gives examples of these classifications.

The terms static and dynamic refer to the necessity to perform periodic refresh to retain information, sometimes called AC volatility. Static memories require no refresh cycles while dynamic memories do. The structure of the memory cell dictates into which class it falls. We will discuss this in detail later. Figure 3 gives some examples of static and dynamic devices.

As stated before access time is the time required from assertion of required signals until a read or write is completed. Cycle time is the elapsed time from the beginning of one cycle until the next cycle may begin. In general, static memory devices will have access and cycle times which are equal.

The reason cycle times are sometimes longer is related to the way in which cells are read. A non-destructive read out (NDRO) leaves a cell unmodified after a read thereby allowing equal cycle and access times. However, under destructive readout (DRO) the contents of a cell are modified by the act of reading the cell and therefore must be re-written after the read is complete. The re-write time contributes to the increased cycle time. Figure 4 gives examples of the readout classifications.

Now that most of the necessary terms have been defined we may continue with the main subject of this article, namely semiconductor read-write memories.

The most commonly used read-write semiconductor memory in current micro computer systems is of the static type. As will be seen, this fact is attributable to the inherent simplicity of the static memory circuitry. Figure 5 schematically illustrates a typical static memory cell. The circuit is essentially a bistable latch composed of cross connected transistors Q_3 and Q_4 with load transisitors Q_5 and Q_6. Transistors Q_1 and Q_2 act as switches to connect the cell to the bit lines when that particular cell is selected by the word line. The cell is written to by driving the bit lines with the appropriate logic levels. The cell is read by sensing the logic levels of the bit lines which are driven by the selected cells.

System design using static devices is relatively simple and straight forward. Standard design practices regarding layout, interconnection, and supply bypassing will yield good, reliable designs. Many examples of static memory system design may be found in the literature.

In the quest for greater bit densities, semiconductor manufacturers developed the dynamic memory cell, a typical example of which is shown in Figure 6. Notice that there are half as many transistors as the static memory cell example.
The dynamic memory cell relies upon enhanced parasitic capacitance as the energy storage element, shown in dotted lines in Figure 6. The cell is written to by turning Q_3 off using READ ENABLE, turning Q_1 on using the WRITE ENABLE, and applying the desired logic level to the BIT LINE which charges the storage capacitor to the proper level. The cell is read by turning off Q_1, turning on Q_3, and sensing the level of the BIT LINE. Even though the input impedance of Q_2 is high, the charge on C_1 gradually dissipates. This unfortunate fact leads to one of the complexities of dynamic memories. The charge on C must be periodically restored, or refreshed, to maintain the stored information. The refresh is accomplished by turning on both Q_1 and Q_3 of Figure 6 which implies a simultaneous read and write. Typically, each cell must be refreshed every 2 ms.

Most manufacturers have now gone to a single transistor dynamic memory cell yielding even greater densities. A typical single transistor cell is shown in a typical matrix organization in Figure 7. A single cell is shown as Q_1 and C_1. For a write operation the proper logic level is applied to the DATA IN line, the proper ROW ENABLE and COLUMN ENABLE are turned on to select the cell, and

Continued on page 36

new product

Technical Systems Consultants Inc. is making the FLEX™ disk operating system available for general OEM licensing. The operating system, written for the 6800 microprocessor, is very versatile and extremely flexible. It provides the user with a powerful set of system commands to control all disk operations directly from the user's terminal. Some important features include fully dynamic file space allocation, automatic removal of defective sectors from a disk, automatic space compression and expansion on text files, complete user environment control using the TTYSET utility command, printer spooling with queue management, random and sequential files, and batch job type program entry.

The Utility Command Set included with FLEX™ resides on the system disk. The individual commands are only loaded into memory when needed. The set of commands may be modified or expanded at any time without the necessity of replacing the entire operating system. These utilities perform such tasks as saving, loading, copying, renaming, deleting, appending, and listing disk files. All of the necessary tools are provided for complete user interaction with the disk.

TSC is currently offering a large variety of support software which runs under FLEX™. These programs include an assembler, text editing system, text output formatter, assembler language debug package, sort/merge package, ISAM file structures, over 36 additional utility commands, and a soon to be released BASIC compiler, all available for licensing.

Information on the non-exclusive license is available from Technical Systems Consultants, Inc., Box 2574, West Lafayette, Indiana 47906. (317) 423-5465

The Home Inventory System is a series of programs designed for creating, updating, and reporting an inventory file that runs under the Smoke Signal Broadcasting RANDOM Disk Operating System on a 6800 MPU system. The system will operate on either a one, two, or three drive system. The random disk file access allows on-line updating and inquiry of any item in the file (maximum of 511 items). All functions of the system are available through the "Primary Menu". The reports can be directed to any port (hard-copy or CRT) in either 64 characters/line or 80 characters/line. Reports are sequenced by item number, part number, part name, category code, or location code. Reports of requested location or category codes are available. Some possible uses for the system include: collections (such as antiques, coins, stamps, etc.), wine cellar, food items, or home furnishings for inventory purposes. The Home Inventory System is available from COMPUTERWARE SOFTWARE SERVICES 830 First Street Encinitas, CA. 92024 for $49.95.

SOUP-UP YOUR TVT

Dr. Edgar M. Pass
Computer Systems Consultants, Inc.
1454 Latta Lane NW
Conyers, GA 30207

Since its first appearance in the Spring of 1975 in Radio Electronics, the TVT-II has been built by more hobbyists than any other similar construction project. For years, thousands of frustrated computer programmers had dreamed of an affordable home terminal throught which they could communciate with the computer at their business or university. With the appearance of the TVT-II, the cost of such a home terminal dropped from over a thousand dollars to the $300 price range. It would be hard to estimate how many TVT-II kits SouthWest Technical Products and Mini-Micro-Mart have sold, or how many people have built the project from scratch or with circuit boards available from several sources, but the total would be well into the thousands.

The modifications to the basic TVT-II circuit as described here materially increase the usefulness of the device. All of the modifications are compatible with each other, and each is independent of all others. Every modification has been installed on at least one TVT-II, and at least one TVT-II has every enhancement described below. All of the modifications described below were developed and tested by members of the Atlanta Area Microcomputer Hobbyist Club.

64 - Characters - Per-Line

This modification was designed and implemented by Dave Kemp, a member of AAMHC, and was further advanced by other members of AAMHC. It has appeared in a number of places since and, though credit for the original idea has been disputed, it should go to Dave Kemp. The change converts the 32 characters per line to a single page of 16 lines with 64 characters per line. After the modification, the TVT-II will will function normally with the cursor returning to home after being moved past character 64 on line 16. Operation of computer-controlled cursor board, screen-read board, and serial or parallel interface boards will not be affected. This modification greatly enhances the usefulness of the TVT-II as a computer terminal, and maintains the capability to use the TVT-II with a standard black-and-white TV set.

The modification requires the addition of one IC, a 7486 quad exor, to the main board. A number of jumpers are added and PC foils are cut. Read each step carefully and double-check the work before proceeding to the next step. A sharp razor blade or exacto knife should be used to cut plating. First heat the spot to be cut with a soldering iron to break down the glue under the foil. Only a small (1/16 in.) gap is necessary. Use small solid conductor wire with plastic insulation for jumpers (30-gauge wire-wrap wire is ideal). It is assumed that you have a component layout and schematic available. Ensure that the TVT-II is working normally before making this modification. For those who need to know such things, address line A9 is changed from high-order to low-order in addressing logic.

Review figures 1 and 2 before starting. Figure 1 shows the revisions to the character counters and other circuitry. Figure 2 shows the major modifications to be made to the board itself. Then perform the following changes, in order:

1. Replace R7 (4.7K between IC-17 and 18) with a 1.2K unit. This increases the dot-clock frequency by a factor of

three.

2. Cut the foil from IC-40 pin 6 to IC-40 pin 11 under IC-40 (see figure 2). To do this you will have to remove IC-40. If you used IC sockets with holes in the bottom you may not have to remove the socket. If you do remove IC-40 or a socket be careful! Desoldering an IC from a plated-thru hole is not easy. Use a small, hot iron and solder wick or solder sucker. After you cut the foil, replace IC-40, being careful to rework the plated-thru holes, if necessary.

3. Cut the foil at 40-5 (IC-40 pin 5) and 40-13 on the bottom side of the circuit board (see figure 2). Cut foil at 40-11 and 40-12 on the top of the board. Cut foil from 40-4 to 33-8 at the feed-thru near 40-14 (see figure 2).

4. Piggyback the 7480 on top of IC-40 by bending all pins except 7 and 14 out horizontally and soldering pins 7 and 14 of the 7480 to pins 7 and 14 of IC-40. If you wish, the 7486 may be socketed and placed adjacent to IC-40, standing vertically, with heavy power supply wires providing support. The 7486 will be labeled IC-43.

5. Cut the foil on the bottom of the PC board at 23-10 (see figure 2). Connect 43-3 to 17-12. Connect 43-2 to 23-10. This adds an exor gate in series with the video out from 23-10. With 43-1 at logic 1 (floating or connected to +5), the video polarity is inverted from the normal TVT-II display. With 43-1 grounded, the polarity is normal. Connect 43-1 to a switch or other logic for multipolarity displays. When using an RF modulator, black characters on a white background provides a better display.

6. Cut the foil at 35-5 on the top side of the board at the feed-thru (see figure 2). Connect 40-11 to 35-5. Connect 40-12 to 33-8. Cut the foil at 27-13 and at 27-9 on the bottom of the board (see figure 2). Connect 40-13 to 27-9, 27-1 to 27-13, and 27-8 to 40-4.

7. Connect 40-6 to 35-4, 40-5 to 43-5 and 40-4 to 43-10. Cut the foil between 35-4 and R23 on the bottom of the board (see figure 2). Cut the foils at 28-4, 28-5 and 33-9 on the top of the board (see figure 2). Connect 28-5 and 43-4 to R23 (pad closest to edge of board). Connect 28-4 to R46 by soldering a jumper from 28-4 to the foil immediately below it on the bottom of the board. Connect 33-9 to 28-4.

8. Cut the foil from 12-4 to 21-1 at 21-1 on the bottom of the board. Connect 12-4 (the foil that was connected to 21-1) to 21-14. Connect 21-1 to 21-12 and 21-12 to address line A9 which is the foil that was cut from 27-9 (see figure 2). Connect 43-9 to A9.

9. Connect 43-8 to 43-11. Connect 42-3 (the foil that was cut from 40-11) to 43-11. Connect 33-6 (the foil that was cut from 40-12) to 43-12. These two jumpers are short wires on the top of the board (see figure 2).

10. Connect 33-3 (the foil that was cut from 33-9) to 43-13.

11. Double-check your work, then apply power.

12. Adjust R4 and R6 for a centered display. If adjusting R6 will not reduce the width of the display enough, decrease the value of R7. The exact value of R7 may be anywhere from zero to 2.2K, depending on your other component values.

13. If, after obtaining a properly-centered display, you notice that some characters lose dots occasionally, you may need to replace IC-22. Some surplus 2513's are not fast enough for 64 characters per line, which requires access times of 63/64 microseconds per character.

14. If you cannot get a stable 64-character display or characters cannot be entered properly from the keyboard, re-check all steps carefully.

An excellent extension (not provided here) of this modification would be to use the video polarity inverter at IC-43 pin 1 to change the cursor indicator from a blinking solid block to an alternating polarity block. The character at that position would then be shown in alternate white on black and black on white, as is used on several more-expensive terminals. The inputs to this circuit would involve the cursor addressing comparator, the serialized outputs of the character generator, and other control levels.

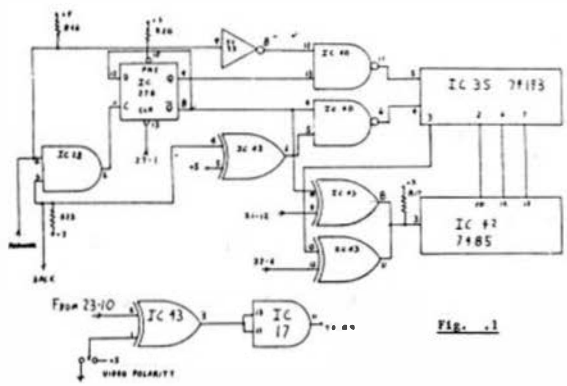


Fig. 1

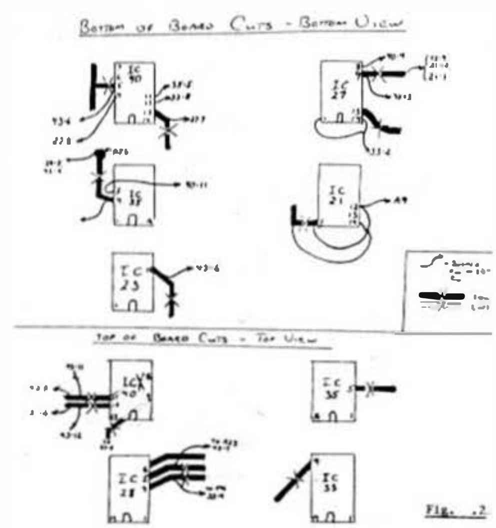


Fig. 2

REPEAT KEY

Many keyboards provide a repeat-key position which is only a momentary closure to ground, rather than providing the circuitry for actually generating repeated character transmission to the TVT-II. Figure 3 provides two diagrams of minimum circuits which will interrupt the strobe line and simulate a character being struck multiple times. The proper circuit is chosen dependent upon whether the keyboard strobe line is normally-high or normally-low. This modification is installed on the serial interface board. Run the line to the main board thru an unused pin in the serial interface board socket. For faster repeat operation, replace C17 on the main board with a 0.5 MF (approximately) unit or construct a separate oscillator. With the circuit in figure 3, one character is generated each time the cursor blinks while the repeat key is depressed. In practice, this has not been a problem; however, figure 4 provides a Schmitt-trigger circuit which may be used if a precise number of characters is essential. A circuit similar to this appears in Don Lancaster's TTL Cookbook.

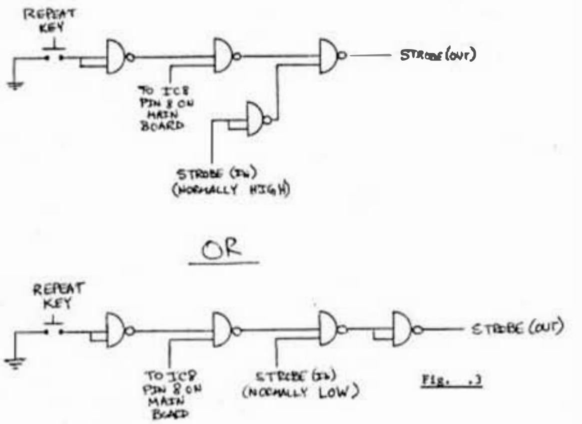


Fig. .3

BREAK KEY

The solution to the implementation of the break key as suggested in the SouthWest Technical Products Serial Interface Board instructions is to attach a 100-ohm, ½-watt resistor between + 5 volts and one side of a normally-open momentary-contact pushbutton, and to attach the other side of the pushbutton to JS-1 pin 6. The effect is to pull the RS-232 output of the serial interface up near + 5 volts which is then recognized by the RS-232 receiver on the other end of the line as a break signal. This solution works, but often requires a separate pushbutton since many keyboards have a break-key position which provides a momentary closure to ground. The solution in this case is to attach a 5K resistor (approximate) between the pushbutton contact and the base of Q2 on the serial interface board. When the base of Q2 is grounded, its output is close to +5 volts, which is recognized as the break condition.

DEBUGGING TIP

If no keyboard is available, the 2513 is possibly defective, or the data being inserted into or retrieved from the TVT-II memory is suspect, there is a quick way to generate a test pattern on the screen. Disconnect the memory board and connect the memory address lines, A1-A5, to the data inputs of the character generator chip. This can be done by making the following connections:

IC21-9	to	IC22-17
IC21-8	to	IC22-18
IC21-11	to	IC22-19
IC14-9	to	IC22-20
IC14-8	to	IC22-21

The display on the TV should then be characters and numbers, in ASCII sequence. If it is not, start checking signals from the 2513 forward and backward.

SCROLLING OF DISPLAY

This modification is installed on the memory and main boards. It modifies the upper four bits of the memory addressing logic so that the display appears to scroll up each time the terminal attempts to perform a cursor down operation from the last line of the screen. To implement this change, perform the following operations, while referring to a TVT-II schematic diagram and figure 5:

1. Pull IC-34, straighten pins 5 and 11, and replace.
2. Cut foils A5-A8 on the memory board adjacent to the socket.
3. Connect lines A5-A8 (IN) to the socket side of the cut foils and lines A5-A8 (OUT) to the other side at eyelets. These wires must be very short.
4. Connect IC-34 pin 11 to pin 12 and to 74191 pin 14.
5. Connect IC-34 pin 5 to ERASE-EOL and CURSOR-DOWN on the main board at convenient plated-thru eyelets.
6. Complete wiring of the 74191 and 7483.

NOTE: If you have a spare 74193, it may be used in place of the 74191 by wiring the appropriate corresponding pins. The two units are fairly similar in function, but different in pinout diagrams.

If the computer-controlled-cursor board is wired for home-up operation upon reception of a certain character, that option should be run thru a switch controlling the computer-controlled character decoding only, not the manual (switch-debouncing) section, in order to selectively disable it when scrolling only is desired.

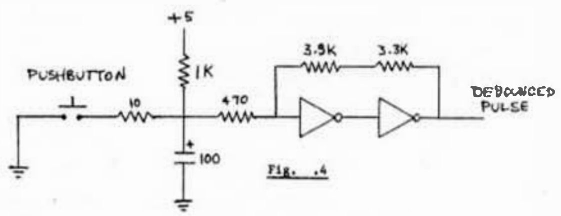


Fig. .4

When power is first applied to the TVT-II, the position of the cursor on the screen and the contents of the screen display will be random. Debounced switches providing home-up and clear-screen may then be actuated. The two functions may be diode-or'ed together to one switch. The results of these operations are to leave the screen cleared with the cursor positioned to the first location of the first line of the screen. When the cursor attempts to move down on the screen, it will do so until it reaches the last line on the screen, in which case it will remain on the last line and the screen display will scroll up.

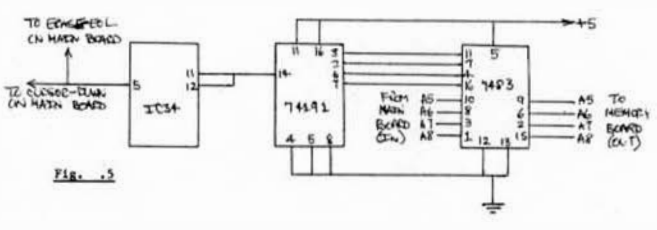


Fig. .5

DISPLAY LOWER-CASE DATA AS UPPER-CASE

Since the memory of the TVT-II can store only the low-order six bits of an ASCII character and the 2513 character generator can display only the same low-order six bits, the TVT-II cannot display lower-case data properly. For example, lower-case 'P' is displayed as a zero, lower-case 'A' is displayed as an exclamation point, etc., as may be seen from a review of an ASCII character set table. It would be advantageous to display a lower-case letter as the corresponding upper-case letter. The modification provided in figure 6 does exactly this. If the two most significant bits of an ASCII character are both at level 1, the next-to-the-most-significant bit is changed to level 0. However, the rubout or delete character (all bits at level 1) is left as-is, rather than converting it to the underline character, since many systems send rubout characters to cause a time delay on transmission and retransmission of data. The modification is installed on the serial interface board. Cut the bit 6 foil just above the socket to the main board and attach the bit 6 (OUT) line there (on the socket side). All input lines come from the UART.

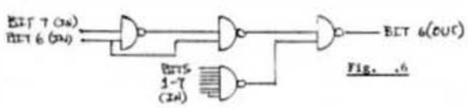


Fig. 6

BACKSPACE AND HORIZONTAL-TAB

Many systems use the backspace character (CTRL-H) as a character correction device and the horizontal-tab character (CTRL-I) as a tabulation device. Unfortunately, in many cases, such as in using the South west Technical Products AC-30 cassette interface, the computer-controlled-cursor board must be programmed for a different set of characters. In this case, there is no indication that the backspace or horizontal-tab has been accepted. This modification, shown in figure 7, provides a cursor-left for a bakcspece and a cursor-right for a horizontal-tab, with only three IC's and two diodes, and maintains the other features of the computer-controlled-cursor board. Obviously, other solutions are possible, such as a distributor, but this solution is simple and inexpensive. The inputs to the 8-input NAND gate provide decoding for all except the last bit of both backspace and horizontal-tab. The last bit is then used to select between the two characters. The two diodes provide for the equivalent of open-collector outputs, since pullup resistors are provided on the computer-controlled-cursor board. Dependent upon the exact components used, it may be necessary to use Schottky, rather than silicon, diodes; however, try silicon diodes first.

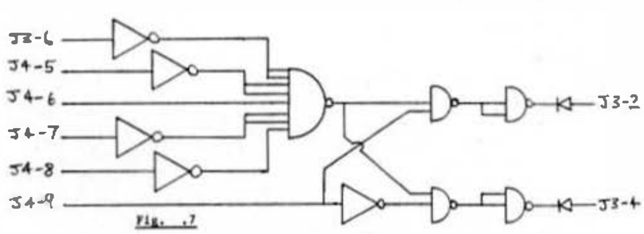


Fig. 7

KEYPRESSED BEEP

It is often convenient to be able to hear an indication of characters being placed upon the screen. While entering data on the keyboard, the sound of a beep for each key depression can easily increase the tactle feel of the terminal. When loading data thru the control interface from a cassette interface or modem, the lack of beeps from the terminal could signify the end of the load. This modification, shown in figure 8, adds a circuit based on two 555 timers which will provide a beep in a speaker when its input line is briefly brought to zero volts from +5 volts. This input line may be connected to the keypressed strobe from the keyboard, to the keypressed strobe on the main TVT-II board, to cursor-down on the main board, to any other desired level in the terminal, or to some combination thru a one-pole, multi-throw switch. The first stage of the circuit serves as a one-shot to lengthen the very-brief keypressed strobe. The second stage serves as an oscillator which is continuously reset except when it is briefly released by the one-shot. None of the components are critical, and the frequency of the oscillator may be easily changed by replacing the .1 MF capacitor from pin 2 to ground.

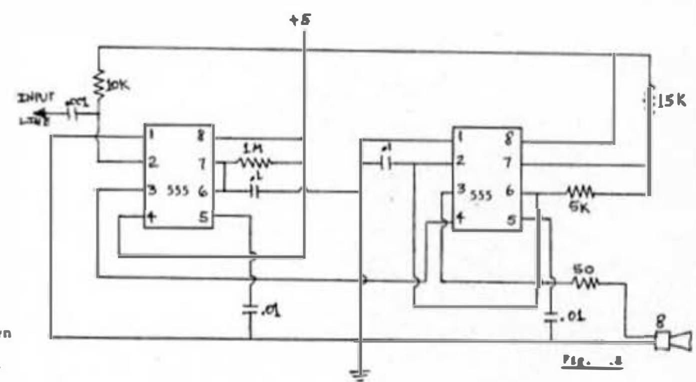


Fig. 8

SUMMARY

The basic TVT-II may be extended quite inexpensively through the enhancements described above, making it an eminently effective device in its price range and extending its useful life-span. As some hobbyists move to more expensive CRT terminals to increase the speed of operation, other hobbyists will be able to purchase the assembled units at reduced prices, so the life-span of the enhanced TVT-II device may be even further lengthened. Obviously, other modifications to the basic TVT-II circuit are possible and desirable. These would include adding other bits to the memory to support upper-and-lower-case character generators, multiple cursors, color displays, blinking characters, etc. Graphic displays are probably beyond the capability of the basic circuit, but the SouthWest Technical Products GR-61 may be used to add this capability for about one hundred dollars. If anyone develops or has modifications which would be of general interest, please send a copy to me for use by the members of the local computer club.

NEW PRODUCTS

Technical Systems Consultants Inc. is making the FLEX™ disk operating system available for general OEM licensing. The operating system, written for the 6800 microprocessor, is very versatile and extremely flexible. It provides the user with a powerful set of system commands to control all disk operations directly from the user's terminal. Some important features include fully dynamic file space allocation, automatic removal of defective sectors from a disk, automatic space compression and expansion on text files, complete user environment control using the TTYSET utility command, printer spooling with queue management, random and sequential files, and batch job type program entry.

The Utility Command Set included with FLEX™ resides on the system disk. The individual commands are only loaded into memory when needed. The set of commands may be modified or expanded at any time without the necessity of replacing the entire operating system. These utilities perform such tasks as saving, loading, copying, renaming, deleting, appending, and listing disk files. All of the necessary tools are provided for complete user interaction with the disk.

TSC is currently offering a large variety of support software which runs under FLEX™. These programs include an assembler, text editing system, text output formatter, assembler language debug package, sort/merge package, ISAM file structures, over 36 additional utility commands, and a soon to be released BASIC compiler, all available for licensing.

Information on the non-exclusive license is available from Technical Systems Consultants, Inc., Box 2574, West Lafayette, Indiana 47906. (317) 423-5465

Technical Systems Consultants, Inc. is pleased to announce the availability of the TSC 6800 Debug Package. It is an extremely powerful and complete assembler language program debugging tool which is capable of simulating all functions of the 6800 microprocessor, including interrupts and I/O operations. It is an ideal substitute for hardware logic analyzers or CPU emulators at only a fraction of the cost.

Any number of breakpoints may be user defined. Each breakpoint may invoke any one or combination of eight different actions. These actions may be dependent on a user defined condition such as register A=$FF or memory location $1855=0. The actions may also be delayed or limited by a pass count. Histogram breakpoints may be set to enable profiling of the executed program. Breakpoints may be set in RAM or ROM!

Complete simulation control allows trace mode to be enabled at anytime. During trace, registers and opcode mnemonics are displayed after each instruction is executed. Single or multiple instruction stepping is permitted as well as simulation speed control. The trace back feature allows the past 256 executed instructions to be viewed. Program execution may be halted at anytime by operator command.

Memory protection and traps are another key feature. Any section(s) of memory may be write, execute, memory, or simulate protected. Execution traps allow program exit on general conditions such as interrupt instruction, transfer instruction, subroutine nest count, and instruction count timeout.

General features include a line at a time assembler, disassembler, memory interrogation commands, hex calculator, machine states counter, stack protection, register modifier, and mode control. In all, there are over 50 commands available. The manual includes detailed operating instructions as well as the complete commented source listing. Requires 9K at $3C00.

SL68-30	Manual and source listing	$35.00
SL68-30C	with KCS Cassette	$41.95
SL68-30D	with mini FLEX™ diskette	$43.00
SL68-30F	with 8" FLEX™ diskette	$55.00

All inquiries should be made to Technical Systems Consultants, Inc., Box 2574, West Lafayette, Indiana 47906.

6800 SORT/MERGE PACKAGE

TSC has announced a full-disk sort/merge package for the 6800 microprocessor. The package was reportedly designed for high speed and convenient operator interface. Written in 6800 assembly language, it is directly compatible with the standard 8 inch FLEX™ disk operating system as found on SWTPC's DMAF-1 floppy disk system. Any type and size file may be sorted. Parameters for the sort may be supplied in any of three ways: as part of the command line, thru use of a "parameter editor" or by specifying an existing parameter file. The package is a full disk sort/merge meaning that files too large to fit in memory will be broken into multiple, temporary work files which are individually sorted and then merged into one. At the end of the merge process, all temporary work files are deleted. It is also possible to merge previously sorted files. The final output file may be routed to disk, CRT terminal, or printer. Features of the TSC Sort Package include:

- Any size, fixed or variable length input records
- Fixed or variable length fields
- User definable record and field terminators
- Accepts multiple input files
- User-definable drives for input, output and work files
- Up to 20 input or output keys
- Total sort key length of up to 250 characters
- Each key may be specified as ascending or descending
- Keys may be reverse order of input
- Keys may be right or left justified
- Handles non-ASCII collating sequences
- Ability to sort upper and lower case as equivalent
- Two levels of run-time messages
- Uses fast modified Quicksort on internal sort
- Merge process utilizes K-way merge selection tree

Included in the purchase price of $75.00 (order part no. AP68-10) is a complete users manual and 8 inch FLEX™ disk containing the object code. No source listing is included with this package. The manual may be purchased separately for $15.00. Contact Technical Systems Consultants, P.O. Box 2574, West Lafayette, Indiana 47906.

THE MICRO WORKS DS-68 DIGISECTOR

The Micro Works Digisector[R] is a random access video digitizer; its resolution and speed are unmatched in industry and the price is unbeatable anywhere. The Digisector and an inexpensive TV camera are all you'll need to see eye to eye with your 6800. Since operation is straightforward, you don't have to be a software wizard to utilize the Digisector's extensive capabilities. The Micro Works Digisector board provides the following exclusive features:

- High Resolution--a 256 X 256 picture element scan
- Precision--64 levels of grey scale
- Speed--conversion times as low as 3 microseconds per pixel
- Versatility--accepts either interlaced (NTSC) or non-interlaced (industrial) video input
- Compactness--utilizes 1 I/O slot in your SWTPC 6800 or equivalent
- Economy--a professional tool priced for the hobbyist

Operation is simple: the computer sends the Digisector two 8 bit addresses (X and Y coordinates), and the Digisector returns the digitized brightness of the image at the specified location. For set-up and monitoring purposes the Digisector also produces an output, comprised of the camera's video signal plus a superimposed intensified cursor, showing exactly where the Digisector is looking.

The software supplied will digitize one pixel every other horizontal scan line, filling 16K of memory with a 6 bit grey scale value per byte in a little less than four seconds. This provides a spatial resolution of 128 by 128 elements, optimum for computer portraiture and slow scan TV. The software drives a Malibu Design Group Model 160 graphics line printer, and is fully commented to ease interfacing to other printers.

Applications include precision security systems, moving target indicators, computer portraiture, fast to slow scan conversion for ham radio operators, and salvation for a Droid in dire need of a wall socket. With clever software, the Digisector can read paper tape, punched cards, strip charts, bar codes, and musical scores.

The Digisector, like all Micro Works products, comes fully assembled, tested and burned in. Give your computer the gift of sight!

Price: DS-68 $169.95
DS-68R (Regulated +12) $179.95

ED MARTIN APPOINTED DIRECTOR OF MARKETING AT SMOKE SIGNAL BROADCASTING

HOLLYWOOD, CA ... Ric Hammond, president of Smoke Signal Broadcasting, manufacturers of computer peripherals for M6800 microprocessor-based computers, announced the appointment of Ed Martin as Director of Marketing.

Martin will be responsible for all marketing, sales and support activities for the complete Smoke Signal product line. In addition, he will be instrumental in the development of new products, new markets and the establishment of an international network of retail dealers.

He brings to his newly created post an extensive background in sales and marketing. Most recently he was National Sales Manager at Micro Peripherals Inc. and served in a similar capacity at Data Measurement Corporation. His technical background includes the Manager of Application Engineering at Cal Comp and Engineering Program Manager at Singer Business Systems.

Aimed at the hobby and personal computing market, Martin explained, "Smoke Signal Broadcasting's line of 8-inch and 5¼-inch floppy disk systems, 16K Static Ram boards, 8K EPROMs board and extensive software library is the smart alternative for the M6800-based microcomputer user."

For additional information, contact Ed Martin at Smoke Signal Broadcasting, 6304 Yucca Street, Hollywood, CA 90028 (213) 462-5652.

the REFRESH LINE is turned on. This action charges or discharges the capacitor C. To read the information back the cell is selected by the appropriate row and column enables. The read amplifier drives the data out line which is sensed by the output amplifier. A cell is refreshed by deselecting all column enables and selecting the proper row enable. The cell's level is sensed and amplified by the read amp. The REFRESH line is then turned on allowing the read amp to charge the cell back to the proper level. In this fashion an entire row is refreshed simultaneously.

Figure 8 schematically illustrates a typical circuit for interfacing 64K x 8 bits of MK4116 to the SS-50 (SWTPC) bus. Many of the details of dynamic memory system design are illustrated there.

The MK4116 is a 16K x 1 N-MOS silicon gate single transistor cell dynamic memory array in a 16-pin DIP. Excerpts from the data sheet contained in the Mostek 1977 Memory Products Catalog are shown in Figure 9.

Fourteen-bit Addresses are presented to the MK4116 in two halves and are clocked in by the two clocks RAS (row address strobe) and CAS (column address stobe) which must be properly sequenced as depicted in Figure 9C. The MK4116 has other allowable timing sequences but this one was selected because of its relative simplicity.

The timing diagrams of Figure 10 shows the timing of the memory interface. Referring to the schematic of Figure 8 the delay line DL1 generates the system timing. This function could be accomplished with one shots but the delay line is both more accurate and more reliable.

For the time being we will ignore the refresh timing logic and explain the read-write logic. Before an access the input to and all the taps of the delay line are high. Flip-flops U5A and U12B are set while U12A and U13A are reset. This is the idle state.

An access is initiated on the rising edge of $\emptyset_2$ (rising edge of $\overline{\emptyset}_1$) if $\overline{VMA}$ is true. This is accomplished by U5A which starts a pulse down the delay line via U8A, U8B and U9A-B. Note also that the address present on the address bus is latched on $\emptyset_2$ as is R/$\overline{W}$. This insures that these signals will not change midway through a cycle.

One of the four signals $\overline{RAS}_0$ - $\overline{RAS}_3$ is fired shortly after the access has begun depending on which bank of 8 RAMs are selected by A_{14} and A_{15}. The RAS clock strobes the row address into the memory chip. The row address is supplied by the INTEL 3242 Dynamic RAM refresh controller chip U3 which acts as a 7 wide, 2 to 1 multiplexor. Sixty nanoseconds after the cycle pulse starts down the delay line the 60 ns tap will go low causing the 3242 to output the column address. At 100 ns the $\overline{CAS}$ clock will fire strobing in the column address.

Up to this point there is no difference between read and write cycles. The read cycle will be explained first.

A short time after $\overline{CAS}$ goes active the contents of the selected address appears at the D_{out} pin of the RAM chip. The data is buffered onto the system bus through the tri-state latch U17.

When the pulse reaches the 200 ns tap of the delay line a positive going edge is injected into the delay line by U7G, U8A, U8B, and U9A and B. When this edge reaches the 80 ns tap $\overline{CAS}$ is terminated via U11C and U12A. Similarly $\overline{RAS}$ is killed by U11C, and U8D. This same edge generates the LATCH signal which latches the read data into U17 allowing the RAM to go idle (i.e. no RAS and CAS) while still retaining data on the bus.

A write cycle is nearly identical to a read cycle. On a write cycle R/$\overline{W}$ will be low at $\emptyset_2$ thus causing U5B to clear. The READ signal which enables read data onto the bus will not be active during the write thus leaving the data bus free for data from the CPU.

A write operation is accomplished when the falling edge reaches the 200 ns tap (the first pass through the delay line). This clocks the Q output of U12B to a low thereby applying a $\overline{WRITE}$ signal to the RAM chips. By this time, write data from the CPU is stable. WRITE is removed by the same mechanism which kills CAS.

The 4116 RAM has several operating modes. The one chosen for this application is the READ/WRITE cycle shown in the figure 9c. For a read operation the "write" part of the cycle is omitted. For a write operation the "read" part of the cycle is performed but the resulting data is ignored.

The resistors marked R_d in the schematic are for damping the ringing on the lines, keeping damaging overshoot to a minimum. The ringing results because of the almost purely capacitive MOS inputs. This, when combined with the parasitic inductance of the PC board traces forms a resonant circuit. The resistors, which usually have values in the 10-60 Ω range, are chosen empirically, a typical value being 47Ω.

List of Captions

Random Access	Sequential Access
Bipolar memory (82S116)	Magnetic tape, discs, drums
MOS memory (2102, 4116, 2708)	CCD's
Magnetic core	Bubble memory

Figure 1

Examples of access types

Volatile	Non Volatile
read-write MOS memory (2102, 4116)	Bipolar ROM (82S115)
read-write bipolar memory (82S116)	MOS ROM (2708)
CCD's	Magnetic tape, discs, drums
	Magnetic bubbles
	Magnetic core

Figure 2

Examples of Volatility classes

Static	Dynamic
Some MOS memory (2102, 2114)	Some MOS memories (4027, 4116)
Most bipolar memory (82S116, 2147)	CCD's
Magnetic tape, disc, drums	

Figure 3

Some Examples of Static and Dynamic Memories

Non-destructive Readout	Destructive Readout
Static MOS Memories	Magnetic core
Bipolar memories	Dynamic MOS memories
Magnetic disk, tape, drum	

Figure 4

Examples of Readout Classifications

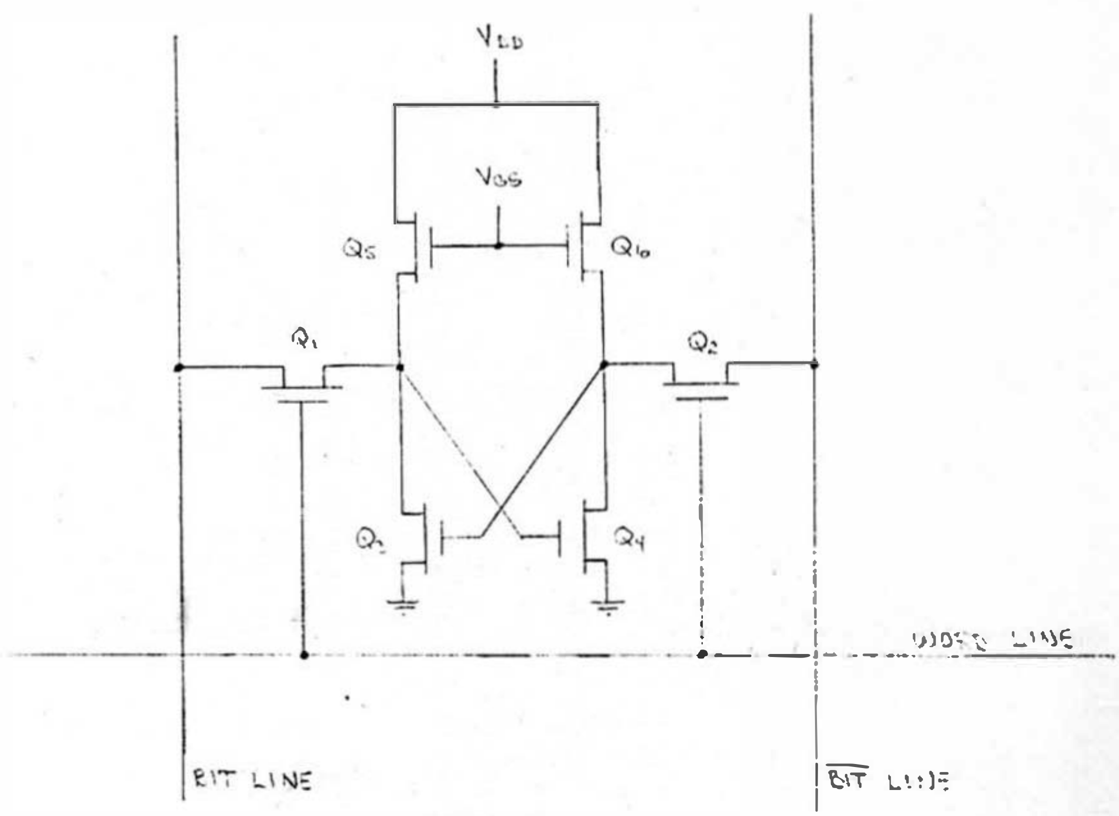


Figure 5

Typical Static Semiconductor Memory Cell
Utilizing 6 Transistors

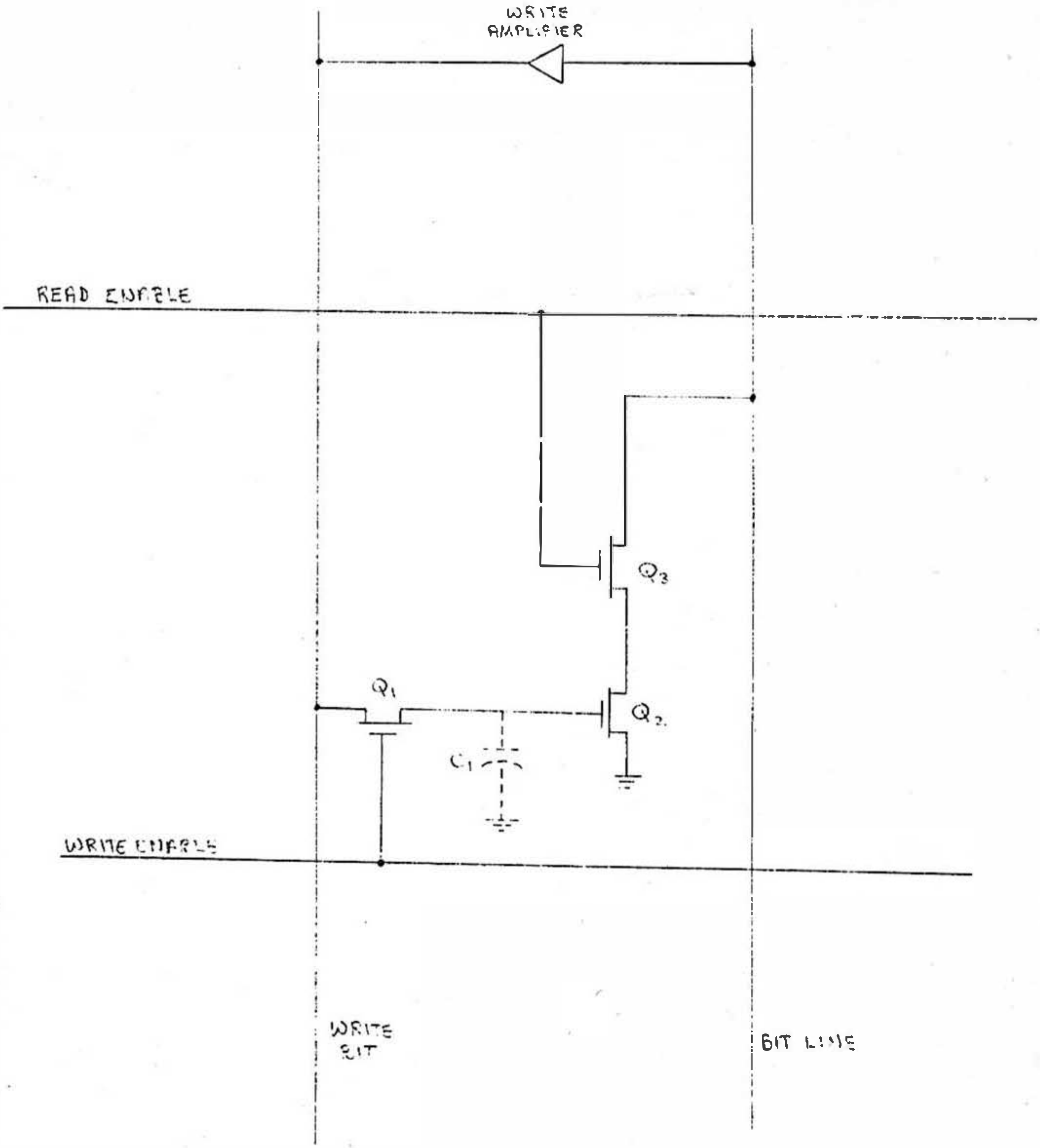


Figure 6
A 3-transistor dynamic memory cell

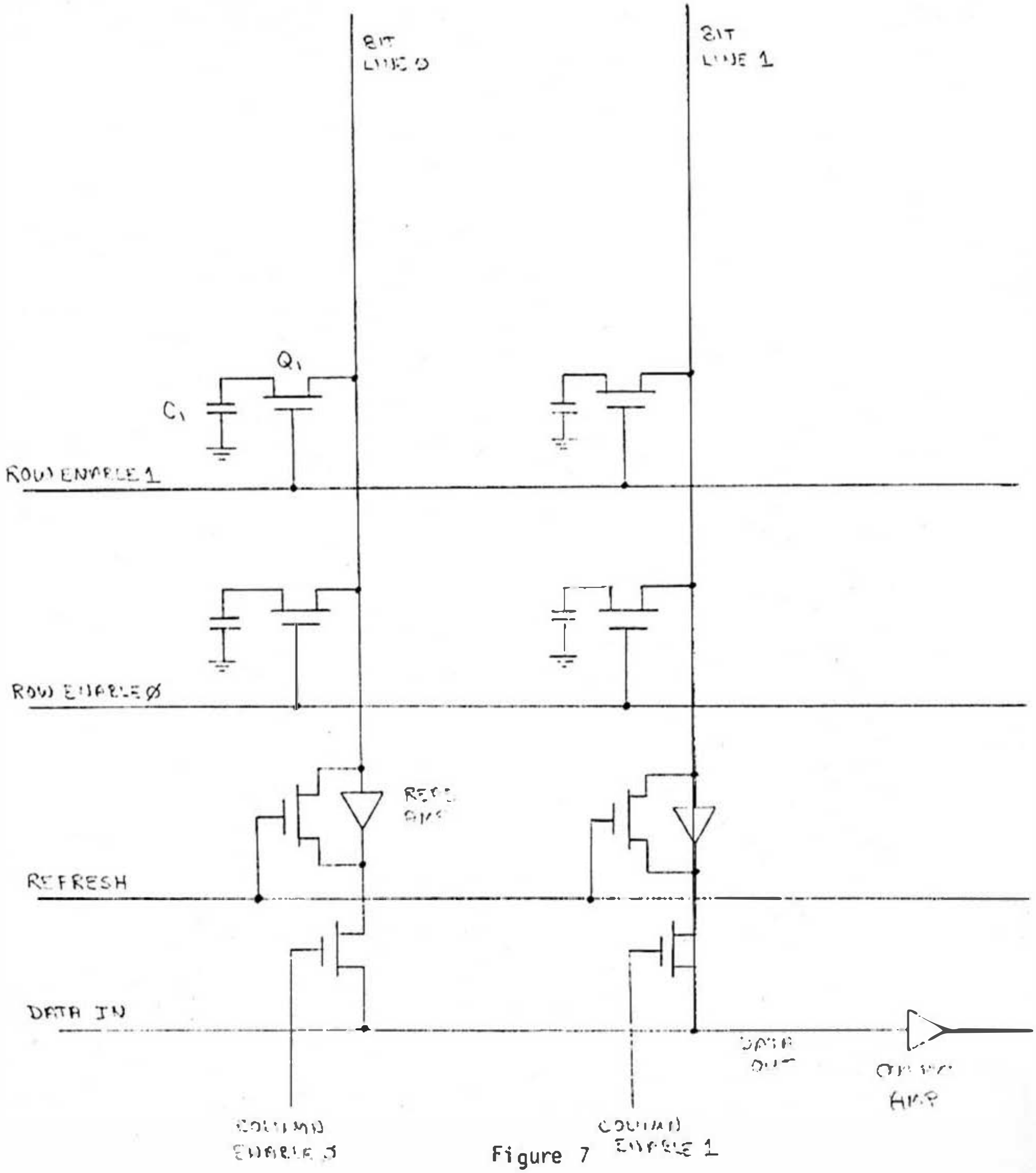
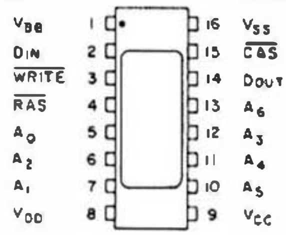


Figure 7

A simplified typical single transistor dynamic memory cell



PIN NAMES

A_0-A_6	ADDRESS INPUTS
$\overline{CAS}$	COLUMN ADDRESS STROBE
D_{IN}	DATA IN
D_{OUT}	DATA OUT
$\overline{RAS}$	ROW ADDRESS STROBE
$\overline{WRITE}$	READ WRITE INPUT
V_{BB}	POWER (-5V)
V_{CC}	POWER (+5V)
V_{DD}	POWER (+12V)
V_{SS}	GROUND

Figure 9a
Pinout of the MK 4116 16K X1 Dynamic RAM

PARAMETER	SYMBOL	MK 4116-2 MIN	MK 4116-2 MAX	MK 4116-3 MIN	MK 4116-3 MAX	UNITS
Random read or write cycle time	tRC	375		375		ns
Read-write cycle time	tRWC	375		375		ns
Page mode cycle time	tPC	170		225		ns
Access time from $\overline{RAS}$	tRAC		150		200	ns
Access time from $\overline{CAS}$	tCAC		100		135	ns
Output buffer turn-off delay	tOFF	0	40	0	50	ns
Transition time (rise and fall)	tT	3	35	3	50	ns
$\overline{RAS}$ precharge time	tRP	100		120		ns
$\overline{RAS}$ pulse width	tRAS	150	10,000	200	10,000	ns
$\overline{RAS}$ hold time	tRSH	100		135		ns
$\overline{CAS}$ hold time	tCSH	150		200		
$\overline{CAS}$ pulse width	tCAS	100	10,000	135	10,000	ns
$\overline{RAS}$ to $\overline{CAS}$ delay time	tRCD	20	50	25	65	ns
$\overline{CAS}$ to $\overline{RAS}$ precharge time	tCRP	-20		-20		ns
Row Address set-up time	tASR	0		0		ns
Row Address hold time	tRAH	20		25		ns
Column Address set-up time	tASC	-10		-10		ns
Column Address hold time	tCAH	45		55		ns
Column Address hold time referenced to $\overline{RAS}$	tAR	95		120		ns
Read command set-up time	tRCS	0		0		ns
Read command hold time	tRCH	0		0		ns
Write command hold time	tWCH	45		55		ns
Write command hold time referenced to $\overline{RAS}$	tWCR	95		120		ns
Write command pulse width	tWP	45		55		ns
Write command to $\overline{RAS}$ lead time	tRWL	60		80		ns
Write command to $\overline{CAS}$ lead time	tCWL	60		80		ns
Data-in set-up time	tDS	0		0		ns
Data-in hold time	tDH	45		55		ns
Data-in hold time referenced to $\overline{RAS}$	tDHR	95		120		ns
$\overline{CAS}$ precharge time (for page-mode cycle only)	tCP	60		80		ns
Refresh period	tREF		2		2	ms
$\overline{WRITE}$ command set-up time	tWCS	-20		-20		ns
$\overline{CAS}$ to $\overline{WRITE}$ delay	tCWD	70		95		ns
$\overline{RAS}$ to $\overline{WRITE}$ delay	tRWD	120		160		ns

Figure 9b

AC Characteristics of the 4116

READ–WRITE/READ–MODIFY–WRITE CYCLE

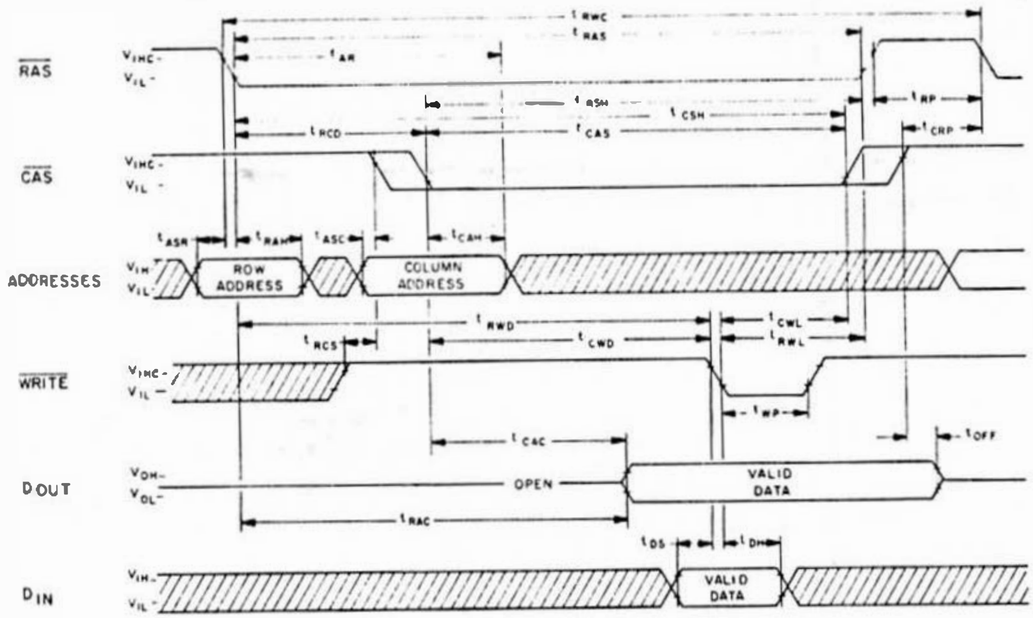


Figure 9c

The 4116 Read/Write -- Read/Modify/Write Cycle Timing

"$\overline{RAS}$–ONLY" REFRESH CYCLE
NOTE: $\overline{CAS}$ = V_{IHC}, WRITE = Don't Care

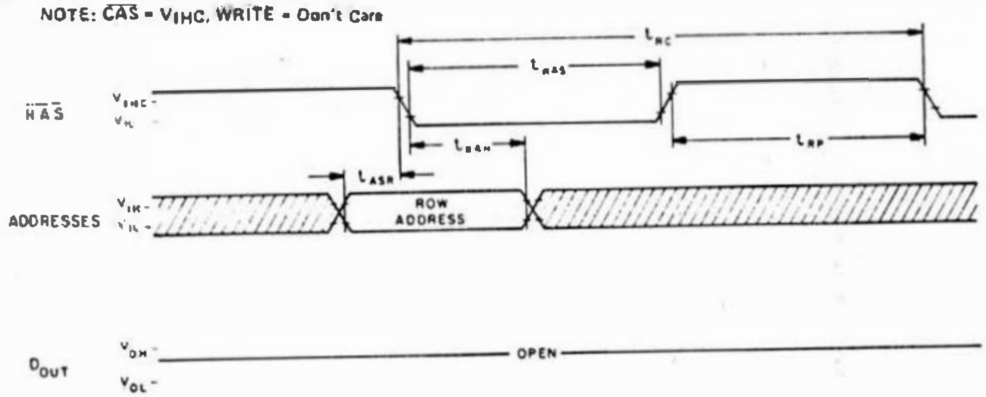


Figure 9d

The 4116 RAS only Refresh Cycle

Continued

ADVANCED COMPUTER PRODUCTS

THE FIRST TO OFFER PRIME PRODUCTS TO THE HOBBYIST AT FAIR PRICES NOW LOWERS PRICES EVEN FURTHER!

1. Proven Quality Factory tested products only, no re-tests or fallouts. Guaranteed money back. We stand behind our products.

2. Same Day Shipment All prepaid orders with cashiers check, money order or charge card will be shipped same day as received.

STATIC RAM BOARDS

Now, just more than 1/2¢ per bit!

S-100 32K (uses 2114)

ASSEMBLED		Kit	
450ns.	639.95	450ns.	539.95
250ns.	699.95	250ns.	599.95

Bare Board 49.95
Bare Board w/all parts less mem. 99.95

WMC 16K (uses 2114)

ASSEMBLED		KIT	
450ns.	325.00	450ns.	279.00
250ns.	375.00	250ns.	299.00

Bare Board 29.95

LOGOS I 8K

ASSEMBLED		KIT	
450 ns.	149.95	450ns.	125.95
250ns.	169.95	250ns.	149.95

Bare PC Board w/Data $21.95
Now over 1 year successful field experience
"Special Offer" Buy (4) 8K 450ns. Kits $117.00

EXIDY SORCERER

Complete expandable Z-80 based computer.
S-100 Expansion module $299.00
SPECIAL OFFER! Buy 32K Exidy and receive FREE Sanyo or equiv. 9" Monitor.
Offer ends March 31

SPECIAL KEYBOARD BUY WHILE THEY LAST

"Clare Pender 63 Key ASCII w/26 Pin & 34 Pin Output Conn." $54.95

IMS STATIC RAM BOARDS

★ Memory Mapping ★ Low Power
★ Phantom ★ 250 ns. or 450 ns.
Only available assembled & tested

	250 ns	450 ns
8K Static	$209.00	$189.00
16K Static	$449.00	$399.00
32K Static	$889.00	$819.00

EXPANDORAM MEMORY KITS

★ Bank Selectable ★ Uses 4115 or 4116 200 ns.
★ Write Protect ★ Power 8VDC, ±16VDC
★ Phantom ★ Lowest Cost/Bit

Expando 32 Kit (4115)		Expando 64 Kit (4116)	
8K	$189.95	16K	$275.95
16K	$249.95	32K	$429.95
24K	$325.00	48K	$599.95
32K	$399.95	64K	$719.95

FLOPPY DISK DRIVES

SA 400 5¼" Single Density $295.00
SA 800/801 Single-Sided* $495.00
*Dual Drive Add on Mainframe inc. box, 2 drives, power, cable, fan $1295.00
Persci Model 277 Dual $1195.00
Persei Box & Power Supply $325.00
Siemens FDD108 .... $399.00*

ANADEX PRINTER

Model DP-8000 compact, impact, parallel or serial. Sprocket feed, 80 columns, 84 lines/min., bi-directional. New only $975.00

Z-80/Z-80A/8080 CPU BOARD

★ On board 2708 ★ 2708 included (450ns.)
★ Power on jump ★ completely socketed
Assembled and tested $185.00
Kit $129.95
Bare PC Board .... $ 34.95
★ For 4MHz Speed Add $15.00
8080A Kit .... $ 99.95
8080A Assembled $149.95

ACOUSTIC COUPLERS • MODEMS

We now stock complete AJ line.
A242A 300 Baud Coupler $325.00
AJ243 LA 36 Coupler $295.00
ADAC242 Refurb. Coupler $275.00
AD342 Orig./Ans. Coupler $398.00
AJ1245 ½ Duplex 300/1200 Bd. ... $795.00
AJ1234 Vadic Comp 1200 Bd Mod ... $895.00
AJ1255 1200 Bd. Vadic Orig/Ans. ... $895.00
Plus more Anderson Jacobson Products

Hi PLOT LOW COST DIGITAL PLOTTER

IN STOCK

★ RS 232
★ Plot Size 7" x 10"
★ Digitizer Avail. Soon
★ High Resolution
★ 2.4"/sec Plot Speed

NEW
List: $1085.00
OUR PRICE $999.00

BYTE USER 8K EPROM BOARD

★ Power on Jump ★ Reset Jump
Assembled & Tested $94.95
Kit $64.95
Bare PC Board $21.95
Special Offer: Buy 4 kits only $59.95 each
MR-8 8K w/1K Ram $99.50
MR-16 16K w/1K Ram $99.50
EPM-1 4K 1702 .... $59.95
EPM-2 2708 or 2716 Eprom $69.95

TARBELL FLOPPY INTERFACE

★ S100 Compatible ★ Uses CPM
★ Jumper Selectable ★ Persci, Shugart, etc.
Assembled and tested $269.95
Kit $179.95
Bare PC Board .... $ 39.95
SD Versafloppy Kit $159.95
SD Versafloppy Assembled $189.95
★ Cassette I/O Kit only $115.00
★ 1771-01 Floppy Chip only .... $ 27.95

TRS 80 16K-UPGRADE KIT

★ 16K with Jumpers & Instructions for either Level I or Level II $89.95
★ 16K for Apple II Upgrade $89.95
Special: TRS80 Schematic .... $ 4.95

TRS 80 TO S-100 PET TO S-100 ADAPTER

Allows Pet/TRS 80 to be interfaced to popular S-100 Bus.

Pet to S-100 Kit $189.95
Assembled $269.95
TRS 80 to S-100 HUH 8100 Kit $275.00
Assembled $355.00
For Low Cost 6 Slot Bus to expand Pet/TRS 80 .... $149.95 Kit

DATABOOKS & MANUALS

DISKETTES

S-100 MOTHERBOARD SPECIAL

8 slot expandable w/9 conn.
reg $69.95, NOW $52.95

PCG PROGRAMMABLE CHARACTER GENERATOR

UV "Eprom" Eraser

PROMS

SUPPORT DEVICES

MICROPROCESSORS

SOCKETS

CONNECTORS

CTS DIPSWITCHES

LIVERMORE BASIC

KEYBOARD ENCODERS

NAKED PC BOARD SALE

ATTENTION KIM USERS

FLOPPY DISK I/O

ATTENTION PET USERS

RAMS DYN.

CHARACTER GEN

UARTS/USRTS

ATTENTION TRS-80 USERS

ATTENTION APPLE II USERS

BAUD RATE GEN

TV CHIPS

A/D CONVERTERS

TV INTERFACES

STATIC RAM HEADQUARTERS

WAVEFORM GENERATORS

CHARGE COUPLED DEVICES

$18.95 each (reg. 43.00)

CRYSTALS

DISPLAYS/OPTO

MONTHLY SPECIALS

DISCOUNT COMPUTER CORNER

P. O. BOX 17329 Irvine, California 92713 New Phone (714) 558-8813
TWX: 910-595-1565

Retail Store Open Mon. – Sat.
Located at 1310 "B" E. Edinger, Santa Ana, CA 92705

COMPUTERWARE . . .

PRODUCT DESCRIPTION		PRICE
RANDOM ACCESS DISK FILE BASIC w/EDIT, line input, & more new features		89.95
SUPER DISK FILE HANDLING BASIC w/EDIT, line input, & more new features	FLEX ONLY	49.95
SUPER CASSETTE FILE HANDLING BASIC	cassette	29.95
PROM RESIDENT CASSETTE FILE BASIC (SUPER features)	cassette 2716	100.00 250.00
PILOT - OUR VERY OWN FOR THE 6800 (with COMMENTED SOURCE LISTING) (also with SOURCE CODE ON DISK)	(SSB or FLEX)	24.95 37.95 49.95
DISK CHECK FILE MAINT. SYSTEM • (Complete checking system)	(SSB or FLEX)	49.95
DISK NAME AND ADDRESS SYSTEM • (Selective printing - labels) (Sorting - Merging - Good manual)	(SSB or FLEX)	49.95
DISK HOME INVENTORY SYSTEM • (Random access files - online inquiry) (6 Reports either hardcopy or display)	SSB ONLY	49.95
AMERICANA PLUS (14 NEWTECH Songs) (Dixie, Noel, Eyes of Texas, & more)	cassette (SSB or FLEX)	15.95 19.95
CSS FOUR PART #1 (8 NEWTECH Songs) (O Holy Night, Moonlight Sonata, Rhapsody in Blue, & more)	cassette (SSB or FLEX)	24.95 24.95
BASIC PGMS #1 (Bluff, Chase, Animal) (Hamurabi, Biorythm, Horse race) • (Mastermind, State Caps, Skydiver)	cassette (SSB or FLEX)	19.95 19.95
BASIC PGMS #2 (Decision, Black Jack) (Math Lesson, Lunar, Keno) • (Stockmarket, Furs, Crash, Wumpus)	cassette (SSB or FLEX)	19.95 19.95
BASIC PGMS #3 (Doall, Maze, Roadrace) (Depth charge, Lifetime, Baseball) • (Interest & Income prop calc)	cassette (SSB or FLEX)	19.95 19.95
WHIZ (TM) - High Speed Binary Cassette Punch and Load System	cassette	15.95
DOODLEBUG - EXTENDED SMARTBUG MONITOR • (with SMARTBUG in EPROM)	cassette 2716	19.95 69.95

• indicates that source listing is included

•••••• Cassette format: AC-30 --- 8" disk add $2.00 ••••••
•••••• ALL SSB Software is available on 8" diskettes ••••••

C O M P U T E R W A R E   S O F T W A R E   S E R V I C E S
830 First Street Encinitas, California 92024

PRODUCT DESCRIPTION		PRICE
LEARN ASSEMBLER - PART I (5 Lessons - Requires 16K)	cassette (SSB or FLEX)	19.95 19.95
LEARN BASIC PACKAGE (I II & III) (All 12 lessons - need 16K)	cassette (SSB or FLEX)	39.95 39.95
RENBAS - RENUMBER BASIC Programs Same features as below	cassette w/source list	24.95
RENBAS - RENUMBER BASIC Programs Selective starting number and increment value	(SSB or FLEX) w/source list w/source disk	 24.95 34.95
XREF - Assembly Program Label Cross-Reference - Complete Commented Source Listing	(SSB or FLEX) w/source list w/source disk	 24.95 34.95
SUBMIT - DOS Batch Processor Macro capabilities Dos Error Recovery	BFD-68 ONLY w/source list w/source disk	 24.95 34.95
BFD-68 DISK TRANSIENTS #1 Includes: Listp, Init, Mem Lodhex, Savlod, Printp Cordmp, Disasm, Convrt, Epend	BFD-68 ONLY w/source list w/source disk	 24.95 34.95
BFD-68 DISK TRANSIENTS #2 Includes: Print, Dir, Pip Cksm, Search, Fill, Filcom Delete - Wild Card Options!	BFD-68 ONLY w/source list w/source disk	 24.95 34.95
BFD-68 DISK TRANSIENTS #3 Includes: Fdump, Sdump, Map, Scout, Memdump, Compare, Hardcopy for other transients	BFD-68 ONLY w/source list w/source disk	 24.95 34.95
FLEX UTILITY COMMANDS #1 Includes: Examine, Findhex1,2,3 Dir, Kill, Files, Frags, Repeat	FLEX ONLY w/source disk	19.95 24.95

SMOKE SIGNAL BROADCASTING

SMARTBUG If documentation purchased		documentation (2708) (2716) 5v	19.50 39.95 49.95
EDITOR or ASSEMBLER EDITOR and ASSEMBLER	SE-1 OR SA-1 SE-1 & SA-1	cassette/disk cassette/disk	29.00 53.00
TEXT PROCESSOR		disk	39.95
SSB BASIC AND RANDOM DOS		disk	39.95
TRACE - DISASSEMBLER	TD-1	disk cassette	25.90 19.95
SOURCE GENERATOR	SG-1	disk cassette	29.90 24.95

COMPUTERWARE...

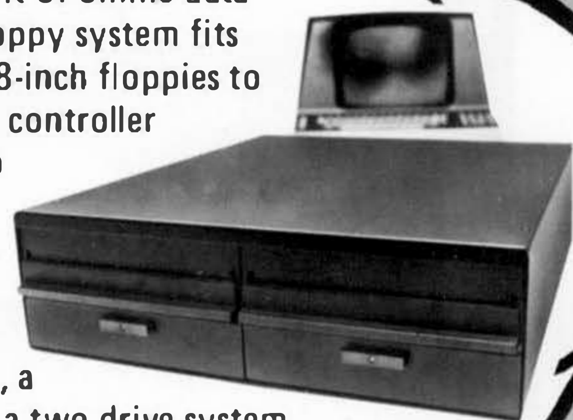
We're not just blowing smoke

SMOKE SIGNAL BROADCASTING PRESENTS IT'S NEW...

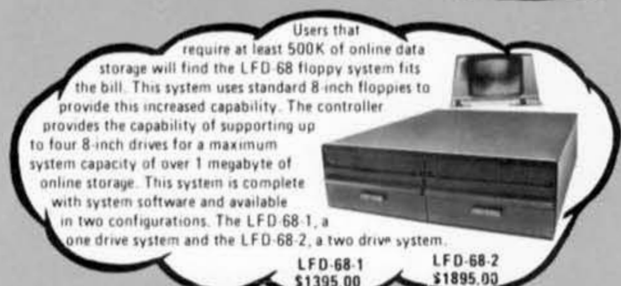
8-INCH FLOPPY DISK SYSTEM

- SS-50 Bus Compatible
- Expandable to 1 Megabyte
- 500K Bytes of Online Storage
- Completely Software Compatible with existing BFD-68 Mini-Disk Systems

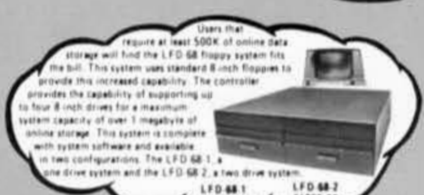
Users that require at least 500K of online data storage will find the LFD-68 floppy system fits the bill. This system uses standard 8-inch floppies to provide this increased capability. The controller provides the capability of supporting up to four 8-inch drives for a maximum system capacity of over 1 megabyte of online storage. This system is complete with system software and available in two configurations. The LFD-68-1, a one drive system and the LFD-68-2, a two drive system.



LFD-68-1 $1395.00 **LFD-68-2 $1895.00**



Ask about our new DFD-68-2 with two 8-inch double sided floppy disk drives with 512K bytes of storage per drive and capability of adding additional drives to provide over 2 megabytes of online storage



SMOKE SIGNAL BROADCASTING®

31336 Via Colinas, Westlake Village, CA 91361, (213) 889-9340

SMOKE SIGNAL BROADCASTING®

31336 Via Colinas, Westlake Village, CA 91361
(213)889-9340

☐ Send information on your LFD-68 Systems
☐ Send name of nearest dealer

Name ____________________
Address ____________________
Company ____________________
City ____________________
State/Zip ____________________

'68' Micro Journal
3018 Hamill Rd.
Hixson, TN 37343

Application to Mail at Second Class Postage Rate is Pending at Chattanooga, TN.

TIME VALUE
PLEASE DO NOT DELAY

5-1/4" Minidisk — Soft or Hard Sector

SAVE

DISK

SOUTH EAST MEDIA SUPPLY

6131 Airways Blvd. 615-892-1328
Chattanooga, TN 37421