Micro-8 Computer User Group Newsletter
Hal Singer & John Craig - Editors
Cabrillo Computer Center
4350 Constellation Road
Lompoc, CA 93436

May 20, 1975
Volume 1, Number 7

Phone (805) 733-3501 (1:40-4:30 PST)
(805) 735-1596 (evenings)

We finally got another issue of the MICRO-8 Newsletter thru the printer and mailed out. Hope your impressed. Note: 72 pages! Subscriptions have made it economically possible to print and mail out this size newsletter. At the rate subscriptions are coming in, future NL's can be this big if we receive enough camera ready material.

A good many participants have running 8008 and 8080 computers and are concerned with trying to make them do something. An effort was made to try to make this issue "heavy" on software. The only way you can get good at programming is to read code so even if you aren't interested in all of these programs, figuring out the code will make it easier to write your own programs.

Printing Format

One wouldn't think that page arrangement would be a big problem in preparing a newsletter but it is. Printing is expensive and first class postage "eats you alive" and is necessary because 3rd class is unreasonably slow in most cases so some kind of reduced format is necessary if a lot of material is to be included. By printing sideways on $8\frac{1}{2}$ x 11 pages, four sheets can be printed per page with only a 2/3 type reduction making it acceptable for reading (although it doesn't do anything for the printing quality when using marginally acceptable camera ready copy).

These sheets can be folded into a booklet as in NL# 5 which is nice except we keep getting complaints that they cannot be punched and placed in 3 hole notebooks. Along comes Eric Schott, 208 14th Ave., Juanita, Altoona, PA 16601 to the rescue. He sent in some sample booklets showing possible printing formats. His unreduced format wasn't acceptable because getting twice as much material printed at only a 2/3 type size reduction is too good a deal to turn down.

His reduced format seemed ideal. It's main feature was that the pages could be placed in a 3 ring notebook and when turned sideways, 4 full pages were arranged in front of you allowing easy page-to-page reference and two-page schematics where necessary. Computer hobbyists are obviously clever guys at getting electronic stuff working but they are sure slow at figuring this format out. All copies are being sent out prepunched to make it a little easier to figure out. Eric and I are sold on this format so you are going to get it whether you like it or not. Try it, you'll like it!

What Is A Newsletter?

There seems to be a little confusion as to what we are trying to do. This is a newsletter service, not a magazine. Our intention is to reprint and distribute material sent in by participants that you might be interested in and get it into your hands as fast as possible with the names and addresses of the contributors so you can contact them if you want more information. Also names, addresses and telephone numbers of participants are included to encourage communication and formation of local groups. Most of the material that comes in is a randomly organized letter containing a whole bunch of items. It is a non-trivial

task to extract and condense this material and include it with the participants name and address. So far, I think we have done pretty well, and we'll continue to try to do the same.

We receive occasionalàl complaints about lack of subject titles and grouping. I think the important title is the submitter's name, address and telephone number. An excellent project for some participant would be a subject matter index. I'd rather spend my time extracting and condensing more material from participant's letters.

John Craig and I have vowed that we will not be the ones that let the newsletter die as long as there is a need to disseminate information. As long as you will send in material, we will continue to try to get it out to you in as readable form as possible. You can help tremendously by submitting material typed with a fairly new "black" typewriter ribbon, schematics drawn with a black flow tip pen, and sharp clear Xerox copies. As soon as material submitted drops off, there will no longer be a need for the NL service and we will cease publishing and return unused funds.

Regarding MITS and 8080 Systems

In NL's 5 and 6, some harsh comments from participants and the editors regarding MITS's sales policies have been interpreted by some as disinterest in 8080 systems (ALTAIR 8800). Nothing could be farther from the truth. At this time, one should seriously consider whether starting an 8008 system that is not easily expandable to an 8080 is a wise move. Most of us started with 8008's and the information is heavily directed that way. Remember though, all programs and interfaces for an 8008 can easily be modified for 8080's but it doesn't work quite so well the other way around. Letters coming in in response to the PE mention confirm that there are an enormous number of ALTAIR 8800 owners desperate for memory expansion and peripheral interface information as well as software. Please contribute any technical information or programs you have for 8080 systems.

Popular Electronics Mention in Computer Notes Column

You probably noticed that Jerry Ogden printed an excerpt from a 5 month old letter in his Computer Notes column in the June PE. When that letter was written, we were still doing spirit master duplication and offered to supply NL's 1-4 to prospective ALTAIR 8800 builders for a 50¢ SASE. Since PE did publish the 5 month old offer, we feel we must honor our agreement so NL's 1-4 are being sent out. The June issue has only been out for two weeks and over 250 requests have been received.

To remain fair to the participants that ordered the printed back issues at $3.50, the fee schedule for these new participants is: NL's 1-4 50¢ SASE, NL's 5 & 6 $3.00, NL's 7-12 $6.00. All participants that renewed for $6.00 will receive issues 7-12.

Flash Announcements

The 1st meeting of the Amateur Computer Society of New Jersey will be held Friday, June 13, 1975 from 7:00 to 10:00 pm at the Union County Technical Institute, 1776 Raritan Road, Scotch Plains, NJ. Contact Sol Libes at 889-2000 Ext. 282 or 291 days or 277-2063 evenings.

See the announcement inside on the LA Area meeting June 15.

8080 boards are now available for the Mod-8 system from Space Circuits. Thanks to Cabell A. Pearse, 3523 Tilden St. NW, Wash. DC 20008 for this information.

A System For The Novice Computer Hobbyist

I'm reluctant to recommend anything because the smallest hint of endorsement is interpreted by some as unconditional recommendation. However we do have some very novice participants that want to get into hobby computer construction without getting in over their head. Everything that I have been able to find out indicates that the Martin Research MIKE-2 system (see the 6 page ad later in the newsletter) is the one for them. It's reasonably priced, very well designed, and supported by a reliable company. Contact Mark Condic, 410 Woods Lane, Apt. 6A, Downer's Grove, IL 60515 or Martin Research for additional information. Mark's college class built up 14 of these systems and they were very favorably impressed. Be cautious on ordering boards only because the IC's to stuff them are the very latest releases and a little difficult to find.

Miscellaneous Comments On 8008's and 8080's

IC Electronic Supply, 15723 Roscoe Blvd., Van Nuys, CA 91406 (213)894-0171 or 619 W. Katella Ave., Orange, CA 92667 has 8008's for $24.95. 8080's are being sold in the L.A. area for $110 but I'm not sure where yet. Rumor has it that the TI 8080 will sell off the shelf at $100.

Intel has announced their 8080-A and all future deliveries will be that part. It differs from the regular 8080 by having full TTL drive capability, improved synch handling, and ability to jam in multi-byte instructions. It was probably introduced to complicate the lives of the second source suppliers. It will work in the same socket as the regular 8080.

Mr. Bill Roberts of MITS clarified the question of what kind of 8080's they are using in ALTAIR 8800's. It seems that Intel was a little upset when MITS was selling a board set including the 8080 for $294 at the same time they were trying to sell 8080's at $350 in single quantity. Those shipped to MITS were given a 8080-8 marking but were otherwise identical. When the fallout rumor started, MITS started accepting and shipping only those with the regular marking.

Intel will produce the 8008 companion chip that contains all the multiplex logic in the Fall and said that pricing of the pair will make it very cost effective compared to the 8080 so maybe the 8008 is not dead yet.

Programming Manuals For The Beginner

Nat Wadsworth of Scelbi Computer Consulting, 1322 Rear, Boston Post Road, Milford, CN 06460 has done it again. His first programming manual was higly recommended. Now he has written "Machine Language Programming For The 8008 (And other similar microcomputers)". You just can't believe the rave revues that keep coming in. It is an absolute must for your bookshelf. Coverage includes: 8008 instructions, flow charting, mapping, editing and assembling, debugging tips, fund. prog. techniques, loops, counters, pointers, masks, organizing tables, search and sort routines, math operations, muliple prec. arith., a floating point package, i/o programming, real time programming, programming for proms, etc. Its available for $19.95 book rate or $22.95 first class. The floating point package is worth that.

MIL Mod-8 Documentation Package

Anyone interested in or actually building the MIL Mod-8 computer system will want to order the $10.00 documentation package being offered by Robert Swartz, 195 Ivy Lane, Highland Park, IL 60035 472-6660 days and 432-6423 evenings, has prepared. Robert is our Mod-8 expert with his own unit having operated perfectly for the last few months. He has debugged another unit and is thoroughly familiar with the system, errors in the PC boards, and the available documentation. The package will include:

1) The MF8008 Data Book containing Mod-8 circuits, foil diagrams and Monitor-8 listings (these are scarce as hen's teeth because MIL ordered them all destroyed) as long as the supply lasts.
2) Mod 8-8 Prom Programmer documentation including circuity, parts list, and component placement.
3) Audio cassette/Mod-8 interface documentation.
4) Notes on the MOD-8 system inluding manual errors and parts lists.
5) Mod-8 self-test programs (getting this system up isn't trivial since not front panel is available)
6) Notes on I/O Port expansion
7) Hints on getting the system running.

I've seen Robert's package and no way are you going to get a Mod-8 up and running without it. (At the present time the only known source of the MF8008 book is duplicating it and selling it for $5.00)

Scope Graphics Terminal

Carl Hellmers of MP Publishing Co., Box 378, Belmont, MA 02178 has a Digital Graphics Oscilloscope Display interface by James Hogenson in Vol 1, #5, that will really turn you on. It drives any cheapie Heathkit or Eico oscilloscope, uses 4 2102's to store a raster of 64 x 64 dots that can be either on or off. Graphics game programs using 8008's or 8080's should be a cinch with this interface. Double sided plated thru PC boards are available at $25 from MP Publishing. We have a copy of the article in our literature loan file to send out on short term loan if you are not an MP Publishing subscriber. You may become one after you get a close look at this interface.

Information Regarding Suppliers

Let us repeat! When you send off a check to a supplier, kiss it goodby, because you may never see that money again. What's almost as bad is if the check gets cashed and you wait months for delivery on items you need now. If a guy's advertising looks too good to be true, it probably is. If it looks like he is offering too many hard to get items, beware. If he can supply them, other people could also, and they wouldn't be hard to get. If the prices are much lower than other outfits, he's probably selling junk and its hard enough to get these complicated computer systems running without having to find defective parts as well. What do you do when a guy offers something you have to have and he is the only one that offers it? I don't know. But you'd better clarify everything by telephone and or letter before you send your money.

If they accept Mastercharge or BankAmericards, you've got several things in your favor. At least some bank recognizes them and part of the agreement they sign is that they will submit information on when and how items were shipped. If you don't get them, you just stop payment on the bill at your bank.

William E. Shawcross, 1105 Massachusetts Avenue, Cambridge, MA 02138 (617) 547-7652 is managing editor of "Sky And Telescope" Magazine with a background in physics and astronomy. He took the Control Data Institute course in computer programming and done work with a CDC 3150 and a tape-oriented Honeywell-200 (tape oriented). He purchased a 256 word ALTAIR 8800 that ran the first time he turned it on after a week of spare time construction. He has a SWTP TVT-II with all the bells and whistles on order is looking forward to putting it together. He'll add more memory and a Suding cassette and should have a system that will keep him amused and occupied as well as helping him with commercial programming--mailing lists and files. He supplied the following program that is a 14 byte program to zero memory, starting just below the program and progressing down to zero and then repeating the whole business until stopped. It is listed here with octal 377 as the high address.

8080 Program To Zero Memory

Address (octal)			Instruction	Purpose
362	001	LXI	00 000 001	Load B&C registers with
363	362		11 110 010	starting address plus 1
364	000		00 000 000	
365	257	XRA	10 101 111	Zero accumulator
366	013	DCX	00 001 011	Decrement B&C registers
367	002	STAX	00 000 010	Store acc. in address in B&C
370	200	ADD	10 000 000	Effectively put B in accum.
371	261	ORA	10 110 001	Logical OR accum. with C
372	312	JZ	11 001 010	Jump to beginning if
373	362		11 110 010	zero (preceding 2 steps check
374	000		00 000 000	to see if address has reached 0)
375	303	JMP	11 000 011	If not zero, return to XRA
376	365		11 110 101	instruction
377	000		00 000 000	

Sincerely yours,

William Shawcross

R. E. SMALLWOOD, 20-12 ST. N.W., CALGARY ALBERTA, CANADA T2N 1Y3 SENT IN A BROCHURE ON THE PHI-DECK CASSETTE UNIT, SOME VERY INTERESTING CLIPPINGS ON THE MIL THING (SEEMS THE CANADIAN GOVERTMENT GOT TAKEN PRETTY BADLY), AND INFORMATION ON A MOTOROLA 6800 MICROPROCESSOR EVALUATION KIT CONSISTING OF 1 MC6800L (MPU), 2 MCM6810L1 (ROM), 1 MC6850L (ACIA) AND EXTENSIVE DOCUMENTATION FOR $300.00. THIS SHOULD BE AVAILABLE THRU ANY MOTOROLA DISTRIBUTOR HERE. HE SAYS THAT HE HAS HAD HIS DOUBTS ABOUT THE ALTAIR 8800 SINCE RECEIVING THE DATA PACKAGE WITH POOR QUALITY LAYOUTS, NO INFORMATION AND LOTS OF ADVERTISING LITERATURE. HE THINKS THE MAGAZINES HAVE BECOME COME-ONS FOR KIT MANUFACTURERS IN AN EFFOR TO PRESENT COMPLEX PROJECTS AND THE ALTAIR 8800 IS THE MOST BLATANT EXAMPLE.

Angel Bravo, 10333 Felson Street, Bellflower, CA 90706 announces that his computer, nicknamed the FRUSTRATION-8 is now running perfectly. He has had some very frustrating experiences. First he gambled and lost with the Electronic Component Sales swindle. He had something wrong with his original MF8008 which Marty Spergle of M and R replaced. He had to dig out some bad IC's with the help of Paul Farr. He can now see that a standardization scheme will have rough going, but hopes we can arrive at some standard, however limited it may be. At present he is learning toward Dr. Suding's approach, that is the TVT, cassette, calculator, and maybe Bob Cook's Baudot TTY. He says each one of us is going to have to set his own goals and not try to shoot for the moon, especially with the cross fire we find ourselves in with announcement of the LSI-11, PACE, M6800, MITS, etc.

Peter Wolfe, 4209 Highland Drive, Yarrow, BC Canada V0X 2A0 decided there was no difference in cost between the $397 ALTAIR 8800 and the MARK-8 after he got thru paying duty. He has ordered the 8800 and a SWTP keyboard. He appreciates Jim Fry's 2102 group buying effort. He hopes that we will soon agree on a cassette standard so he can decide which one to buy. (Maybe you'll have to settle for one of each, there cheap.) He is also trying to decide what kind of output device to use and is presently impressed with TVT-II.

Kendall Stambaugh, 5009 Guide Meridian, Bellingham, WA 98225 734-9424 has offered to redraw schematic diagrams for us. Thanks Ken, we'll get some to you soon.

Richard A. Peterson, 9004 184th Avenue E, Sumner, WA 98390 appreciates the information in the NL and says we do need a central clearing house for the start of this giant new hobby effort. It takes time and money to have a good organization and there is no way this thing can be sustained by depending solely on donated time. Popular Electronics is a joke (seems like a front for MITS), so all we have is Radio-Electronics (I'm not so sure they aren't a joke also) and the efforts of the various newsletter and publishing groups.

Joe D. Ringland, po Box 4784, Rolling Bay, Wa 98061 has constructed an ALTAIR 8800 and is awaiting a keyboard from Mini Micro Mart and a TVT-2 from SWTP. He has had great success translating 8008 programs to 8080 format. The only adjustment that need to be made are in timing loops and it would help if people would specify the time for their loops. Both 8080 and 8008 people would be helped since people use different clock frequencies and memory cycle times.

Gary L. Dickman, 707 Elm Street, Hays, KS 67601 is building a Mark-8 and will use a 4k 2102 board for memory. He is extending the input and output ports and is redesigning the LED Output register board to accept 7-segement readouts. He will soon have a PC layout ready for a double sided board but can't find anyone to etch a double sided board. Any suggestions? These layouts and/or boards are available to any user group participant. He will add the Suding cassette interface, Suding TVT, and Suding calculator and is contemplating a computer controlled cassette deck or a floppy disk for the future and would appreciate any information in this direction that participants can supply. He will use his Mark-8 as a sophisticated CCTV Video Titler and Animator (graphics terminal?) and for a music machine courtesy of Phil Mork's music program.

Mr. G. Penner, Bio Medical Electronics, Boca Raton Community Hospital, 800 Meadows Road, Boca Raton, FL 33432 (305) 395-7100

Albert Sardo, PO Box 6678, San Jose, CA 95150 sent a flyer to some participants advertising ICs. I never received one but I heard that he had slow 1702As at $13.

Robert L. Hatfield, 228 - 22nd Street, Ashland, KY 41101 should have a Mark-8 running by now. He is using a Precision Systems power supply. He has received a copy of Scelbi"s "Machine Language Programming For The 8008" and says that this is the best programming book he has ever seen.

LTC Donald R. Kelley, Facility Engineer, Fort Lesley J. McNair, Washington, DC 20319 is working on a Mark-8 but also ordered, received, and assembled the ALTAIR 8800. It is up and running but he is still having trouble with the MARK-8 memory board and shorts.

Geoffrey D. Lowe, 3615 North Pine Grove, Chicago, IL 60613 was not happy with the way NL#6 pages were arranged. He is a programmer (languages: PL/1, FORTRAN, Assembler) on fourth generation IBM equipment (S/370-145, DOS-VS). He ordered parts for the TVT-1 and then found out the PC boards were not obtainable (try Semtronics). When he finally gets a computer built, he plans to use it for financial files, book and record file lists and data retrieval and eventually, graphics. Peripherals will include two cassette drives, TVT type terminal for program entry and monitoring and an off-line printing unit for hard copy and perhaps a floppy disk. He has had excellent luck with Godbout and James as suppliers and received the typical rude response from SWTP when questioning them about TVT-1 boards and parts. He recommends the SCELBI manual and has ordered the new programming manual from them. He would like denser memory boards and wishes someone would put together a glossary of terms especially relating to microprocessors that would help the newcomers to the field understand such terms as scrolling, burned-in IC's, crowbard power regulators, PROM programming, clock rates, etc.

Beardsley Ruml, II, 3306 Cathedral Avenue NW, Washington, DC (202)333-0173 says that now that the computer is properly seen as simply another tool, folks like him who have no training in electronics, will want to learn the necessary background info. We would do him and others a great favor if a small section was devoted to the "beginners" by citing some of the best materials for "starting at the ground floor". (It seems we keep mentioning this over and over but its true. Start with the SCELBI manual, work up to the new SCELBI manual and go from there. We will try to encourage some readers to write some articles for the beginner that will get him started.)

R. D. "Slim" Cummings, 510 West Fifth St., Pittsburg, KS 66762 is WA0EDA and just came aboard.

Jim Farschon, 3949 Mt. Everest Blvd., San Diego, CA 92111 (714) 459-0211 (day) (714) 292-9180 (night) has ordered a MIKE 201 kit with an 8008 and 1k of RAM from Martin Research. He will soon build a scientific calculator interface and is working closely with Mr. George Enos who assembled and checked out his ALTAIR 8800 in two days. George is now impatiently awaiting delivery of a keyboard so he can use it. Meanwhile he is designing a paper tape reader which they hope to build for under five dollars. He is interested in hearing from others in the San Diego area and is willing to serve as the focal point to see if there is sufficient interest to start a local user's group.

Terry G. Harris, 417 Northwestern Drive, Grand Forks, ND 58201 (701) 772-8453 has built the Scelbi 8-H minicomputer -- a very simple construction project from bare bones and an excellent set of instruction manuals, both software and hardware, RE TVT-1 -- just finished, what a mind-bender, wishes he had been a few months slower to take advantage of Dr. Suding's TVT, Scelbi Mag Tape Interface -- works great but he feels awful about spending so much when the Suding unit works so well, SWTP keyboard -- it works. He has ordered Scelbi's new 8008 programming manual and has been toying with the idea of writing a mini ALGOL compiler. The subset would be about 4 to 6 K. He would like info on a good line printer and a floppy disk.

John Bottoms, Box 158 CTS, 1000 West 42nd, Indianapolis, IN 46208 says he has TVT-II boards available for $18.00.

Ron Durnin, 2310 Munroe Avenue, Saskatoon, Sask. S7J 1S5

Per Biorn, PO Box 309, Quakertown, NJ 08868 works for Bell Labs, 2B-310, Murray Hill, NJ 07974 and is presently building a Mark-8.

Lawrence Miller, 826 Halstead Blvd., Jackson, MI 49203 is a graduating senior at Jackson High School and will be going to Michigan Technical University in the Fall. He is building a Mark-8 and a TVT-1 and wants to know if there is a compiler or interpreter for BASIC or FORTRAN for the Mark-8.

Peter Middleton, Box 714, De Kalb, IL 60115 is connected to the Dept. Of Music, Northern Illinois University, DeKalb, IL 60115.

Jim Squires, PO Box 1124, Santa Maria, CA 93454 (805) 925-1402 is employed by Alan Hancock College as a programmer and is working on his Masters Degree at Cal Poly, San Luis Obispo. He is looking for a Masters Degree thesis project and favors an 8080 operating system. I'm trying to get him to write an 8008/8080 operating system built around a pair of Phi-decks. Let's flood him with encouragement and maybe he will.

M. P. Squires, Rt. 2, Box 326, Nokesville, VA 22123 is Jim's brother and an IBM logic designer in Manassas, VA.

S. Joseph Toy, Route 3, Box 73, Chico, CA 95926 has a Scelbi 8-H, the Scelbi cassette tape interface, and their TTY interface. Input is via a Model 15 TTY keyboard and output is on the TTY printer. The system is being used to handle data in a plant breeding program. He is a ham, WB6KAI, and currently operates on 160 meters.

J. W. Ward, 7236 Cirrus Way, Canoga Park, CA 91304 also noticed the same ouput instruction bug described by G. Hart in Vol. 1, NL #5. He has had his Mark-8 working (?) since late December but is still trying to track down an intermittent bug and wants to know if there are any timing diagrams available for the CPU board. He also has trouble with the clock sometimes oscillating at 2 times the correct frequency and will try the fix suggested in NL#5.

Tom Newman, 2230 Sweetwater Drive, San Leandro, CA 94578 (415)352-2315 says that if we need any info on Diablo disks or HYTYPE I to let him know because he works for Diablo-Hayward.

C. Richard Corner, PO Box 2017, Fargo, ND 58102 must have a Precision Systems PS since he wants Dave Chapman's redrawn schematic.

Raymond L. Heinrich, 1726 N. 16th Street Apt. 1, Arlington, VA 22209 (If all the guys in the Washington D.C. area ever get together, they'll take over the whole hobby computer movement.)

Robert Pearce, 28 Hakin Street, Danbury, CT 06810 has the Mark-8 PC boards and plans to use plug-in boards. The back plane will be Radio Shack copper clad 6" x 9" sheet etched for 8 board connectors. He intends to have 8K of 2102 memory.

Mark Spohr, 527 Lafayette Avenue, Cincinnati, OH 45220 says that a National Semiconductor engineer told him that the UV erasable PROMs cannot be indefinitely erased and reprogrammed. After a certain number of cycles (he said 52) the device will not work properly unless it is reconditioned by baking in an oven at 400° for 45 minutes. After this reconditioning, the device can again be erased and reprogrammed 52 times (NS recommends only 35 cycles before reconditioning).

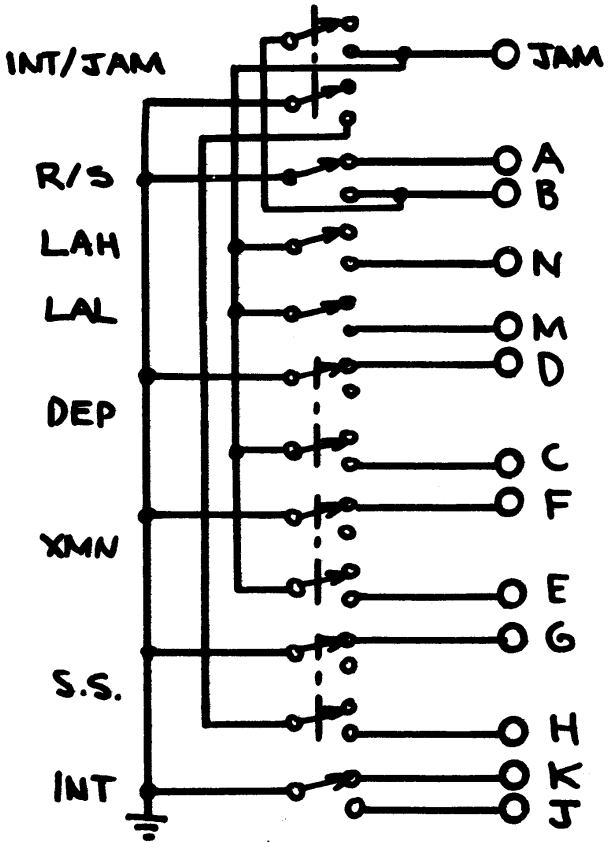
Joseph Weintraub, Lightworks Productions, Inc., 46-16 65th Place, Woodside, NY 11377 is interested in a computer interface with a light pen for video graphics. He says he has done some work with a color set. Where can he buy a light pen? Joe is very interested in joining a local group in the New York Area and if there isn't one, he like to start one. Write him if your interested.

Stephen B. Gray, Amateur Computer Society, 260 Noroton Avenue, Darien, CN 06820 has been swamped by inquiries about the ACS. He says that back issues are available of his newsletters as follows; Volume I consisting of the first 11 issues (1966-1968) at $3.00; Volume II, 12 issues (1969-1972), also $3.00; Volume III, 11 issues so far and there will be several more, $5.00. He's making noises about not continuing his newsletter beyond the few issues left in Volume III. The hobbyist needs all the help he can get. Write him and encourage him to keep publishing.

Mike Anastasion, 6211 Quincewood Circle, Citrus Heights, CA 95610 is a systems analyst programmer III with about seven years DP experience. He has written I/O subroutines, interrupt subroutines, and load business application programs. He has dreamed of having his own home computer for fun and profit since he started in DP and is now waiting for the "dust to settle" before determining the best way to go. So far the 8080 looks promising, especially the ALTAIR 8800, but MITS' I/O device price list has him scared. (Don't panic Mike, anything that works with an 8008 system can be easily modified to interface with an 8080 system and several companies are already cranking out compatible peripherals and ALTAIR owners should have no trouble trading information that they acquire.) He would like to see an article in the NL that would evaluate all available systems so that the beginner could more easily decide which system to build. He has a set of SWTP TVT-I printed circuit boards he is willing to sell and he will trade programming time for electronic goodies.

Richard C. Creighton Jr., 1053 Princewood Drive, Orlando, FL 32810 is starting assembly of an 8080 CPU based computer system. In is currently working in the Medical Electronics (x-ray) service field and formerly was an instructor in the USAF teaching radar repair. He is currently teaching programming to a friend and is trying to complete the computer to make the learning situation more meaningful. He would like to organize a local group in the Orlando or central Florida area. Contact him if you are interested. He is interested in any information on an 8080 operating system and a BASIC compiler. (See what MITS has done.) He also hopes to develop a short form assembler and eventually programs for games and data handling.

Harold L. Novick, Attorney, 2810 Henderson Court, Wheaton, MD 20902 (301) 933-7453 has had his Mark-8 up and running since mid-January. He has two non-electronic problems: 1) Placating one's spouse 2) Getting one's wife at least half interested so it is easier to explain why $50 for more memory is necessary when the ½K installed is not being used because of lack of peripherals. Even his friends are offering to testify against him at sanity hearings. He and Steve Winick may have found a tremendous buy. They are tracking down and hoping to bid on some surplus terminals. With luck, they may be able to offer Dura Mach 10's at $200-$250, Itel 1041 at $300-$350, Dura 1041 at $300-$350, Friden Flexowriter's at $50 to $100, Itel 941's at $200-$250. The catch is that they are government surplus and cannot be assumed to be in operating condition, but at those prices, one could afford to spend another $100 to put them in working shape. Anyone interested should send a SASE. Firm details will be printed in the NL as soon as the details get worked out.
He made the following mods to Phil Mork's front panel switch mods (NL #4). He offers the following program to zero all of the 8008 registers in approximately ½ the usual instructions. In the process of working on our hobby its possible that a patentable development may come up. Hal provided this information:
Patents: A patent is good for 17 years from the date of issue and prevents others from making, using, or selling the patented invention without permission. A patent doe NOT give the owner the right to make his own invention since since someone else may also have a similar patent which would prevent this. If anyone publishes a description of the invention or if anyone publicly uses or sells an embodiment of the invention (including the inventor) more then a year before a patent application is filed, the invention becomes public property and a patent of it is forever barred. I asked if he would be willing to answer preliminary questions participants might have regarding patent problems and he graciously agreed.




```
250 - Exclusive OR A register with its self
310 - Xfer A register to the B register
320 -        "                 L    "
330 -        "                 D    "
340 -        "                 E    "
350 -        "                 H    "
360 -        "                 L    "
```

Program To Zero 8008 Registers.

Switches shown In The RUN Mode

Hal Novick's Mods To Phil Mork's Front Panel Mods.

Bill Fuller, 2377 Dalworth 157, Grand Prarie, TX 75050 has this to say:

I'm one of those Altair owners, so maybe my comments should be heard. I get a little perturbed about some of the bad-mouthing, especially when it is related to emotional reaction. My reasons for going Altair and 8080 was because:

1) At the time I became committed to a home computer (Nov./Dec. 74) differences in price between Altair and others available was not great.
2) ordering from multiple sources and hoping I got everything and it was good left me uneasy.
3) the 8080 IC was priced at $360.
4) I had no source that said I would be able to convert from the 8008 to 8080 within a reasonable price range.
5) I did not know that the Micro-8 NL had already covered so much territory. My only knowledge of "you-all" was a letter to the editor in R-E, until you cropped up in MicroMini Mart NL.
6) I sent my $5 and $10 everywhere info was advertised. Some top-notch, some garbage.
7) I figured I would do all my own design around the basic Altair if I had to, since I was "all alone'.
8) Uncle Sam owed me money.
9) I was interested in limited use--teaching machine and game machine.
10) I was frustrated by reading 1000 pages of claims and counterclaims by CPU manfacturers and writers.
11) The Intel was best supported and Altair was using it.
12) I'm not an expert in programming or hardware.
13) No one knew that parts I couldn't afford one week would be cheap the next week.
14)Ad infinitum.

How do I feel about the Altair? I partially knew what I was getting into, only 250 words when I knew I would need a basic 1000.for what I wanted to do at the time. Since my uses have grown, I see that I got my work cut out for me if some of the attitudes in the NL become prevalent. That attitude being the hell with the Altair and in some cases the 8080. I think there is a place for all models and configurations whether it be 8008, 8080, Altair, PDP, F-8, MC6800, 4040 or whatever.

If we sit around criticising what each other has, instead of trying to work out the mutual problems, then we'll all end up like a guy I met in Dallas. He just got his computer working after 5 years of effort. Think of what went through his mind when he realized his 100+ IC CPU had just been replaced by a cheap multipin IC. I think the desire to standardize is in the right direction. I agree with Larry Pleskac comments--most of us are not far enough along to provide comments. I'm still trying to dig through the M & P ECS Series. Somewhere between ECS 5 and 7 I became lost. Mainly because I need the "hands-on" along with the theory.

Sure I'd like a PDP-11 or something built around the 6800, but I just came out of the DO Loop of frustration trying to determine who was talking and who was acting. So maybe some year Uncle Sam will come across again, then my kids will have the first generation micro.

Robert A. Van Winkle, 12717 Folsom Blvd., #32, Rancho Cordova, CA 95670 has collected the parts for his Mark-8 and plans to etch his own boards and include edge connectors on all boards. He is currently attending classes at Sacremento State and will really get moving after finals. He will make his PC board layouts available to any participant interested. He'd like to acquire an MF8008 manual.

Roger Smith, Smith Enterprises, 4502 E. Nancy Lane, Phoenix, AZ 85040 gives a red-faced thanks to R. D. Hogg who caught his "boo-boo" on the UART I/O board. When he built his TVT, he didn't install diodes D10 thru D14 since he held these bits low on the keyboard where his CLEAR key was. In the intervening months, he forgot his TVT was different in that area. He received an ALTAIR 8800 last week and had it running in two days but his Mark-8 is still not up. He doesn't think he will be buying any other "inexpensive add-ons" from MITS but couldn't resist the "loss leader" initial price. He enclosed a copy of a cross-listing of the 8008 and 8080 instruction sets included elsewhere in the NL.

Jim Lefwich, MD, Geophysical Communications Systems, 410 South Glendora Avenue, Suite 140, Glendora,CA 91740 (213) 335-1241 says he can't remember ever reading a piece six times and enjoying it more (NL #5). Having fought his way thru a full blown processor in 1965-1967 with discretes (transistors, not IC's) and no one around with similar interests, he can assure us that he will be a staunch supporter of our hobby effort. He wears two hats - a general practice of medicine with an adjoining electronics lab for the remaining. It looks like its time to build again so he is trying to decide which way to go. A suggestion for Jay Bowden's power supply circuit in NL#5, if you move one of the diodes so that it is in series with the 150 ohm charging resistor and supply, if the supply zaps shorted, it will isolate the short from the battery and the gear will not load the battery.

Peter Vickers, 162 Nehoiden Street, Needham, MA 02192 (617)444-1410 has had his Mark-8 running for about three months. Consturction on the Mark-8 and TVT begain in mid-October. By the end of November trouble-shooting most of the boards was under way. By the end of January, his room was a junk pile, savings gone, bills unpaid, study habits destroyed. As soon as the units were working exactly right with no more bugs, they were shoved into the corner and not touched again until normalcy finally returns.

Tom Boyko, Varian Data Machines, 12062 Valley View, Suite 204, Garden Grove, California 92645, (remember him from NL #5?). Tom did it the hard way, but probably faster than all of us put together. His wire-wrapped version of the Mark-8 is up and running with his music program playing "A Bicycle Built for Two." (Tom has a subscription to Radio-Electronics but somehow missed last July's issue which described the Mark-8. It's hard to imagine where he would be now if he had got started then.)

==========

Several people have mentioned that the information in the NL is rather hard to digest because it is all so scattered around without any organization. We agree, but this is the nature of the beast. If anyone would care to put together an Index/Cross-reference of any kind we'll be happy to publish it.

==========

Phil Glazer, Ace Electric Service, 40 Broadway Ave., Trenton, New Jersey 08618, says that he can't work, take care of his family, cut the grass, paint, etc. Between our group, the Digital Group & Dr. Suding, the Computer Hobbyist, and the People's Computer Company he is going bananas and beserk! But, he says he is certainly learning plenty & that is what it's all about. He recently finished his TVT I.

NEW ADDITIONS TO THE ROSTER

Gary Alevy, Emory University, Box 21393, Atlanta GA 30322, (ph: 377-4907), is interested in building a Mark-8 and would appreciate any tips on purchasing the various components.

Alan S. Bailey, McKee Vocational & Technical High School, 290 St. Marks Place, Staten Island, N.Y. 10301

Meir Baran, Department of the Army - Picatinny Arsenal, Dover, N.J. 07801 (SARPA-AD-F-D)

Robert A. Barber, 7 Johnson Place, Ardsley N.Y. 10502, is currently building a Mark-8.

Elmer T. Beachley, 5601 Penn Ave. Apt. C-79, Pittsburgh PA 15206, heard about us thru the People's Computer Company.

Neil A. Benson, 10040 Nicollet, Bloomington Minn 55420, has about 3/4 of the parts for a Mark-8. Hal Chamberlain of the "Computer Hobbyist" suggested he get in touch with us before starting construction.

E.L. Bethel, 418 E. School St., Kent OH 44240

Gary K. Berkheiser, 601 Bath St., Bristol PA 19007, is a technician and hobbyist with an interest in micro-computers for both hobby and commercial purposes.

John Bird, Associate Prof, Community College of Baltimore, 2901 Liberty Heights Ave., Baltimore MD 21215

W.A. Bobisud, Route 1 Box 559-D, Grass Valley CA 95945, is a retired Electronics Engineer who has been recently afflicted with 'computeritis'. He has just completed assembly of an Altair 8800 and is presently trying to develope a less expensive I/O card than that provided by Altair. He heard about us thru the Digital Group Clearinghouse.

Robert K. Burant, HMX-1 Exec Flt MCAS, Quantico VA 22134, hasn't made up his mind which way he'll be going (choice of system) but he's getting ready to make the plunge.

John Burger, 1440 Leopold St., Jasper INDIANA 47546, has a Mark-8 up and running with a TVT attached. He's building a cassette interface and is planning to buy the Monitor & 16K Rom thru Mini Micro Mart.

T.F. Caldwell, P.O. Box 116, Burgess VA 22432, is just completing a TVT and is in the process of accumulating parts for the Mark-8.

H.N. Campbell, R.D. 3 - Brockway Road, Moravia N.Y. 13118, has a completed Mark-8 micro-computer which does not compute. As a matter of fact, it doesn't do anything! Upon re-reading some back issues of RE he ran across Hal's letter in the Oct issue. He is now one of us and the happy recipient of all those de-bugging tips in NL's 1 thru 5.

Douglas Faunt Jr., 310 Holly St., Columbia SC 29205, is going in the direction of the DEC LSI-11 (see NL #6), and is building a terminal with Mini-Micro Mart's Univac printer mechanism.

Michael Christoffer, 4139 12th N.E., Suite 400, Seattle WA 98105, is a graduate engineering student at the University of Washington. He just completed construction of an 8008 based microcomputer and is looking forward to some heavy software development.

Lynn E. Cochran, 377 Laurie Meadows Dr., Apt #327, San Mateo CA 94403, is waiting for his Altair 8800 to arrive and is planning to go the Solid State Music memory board route. He, and some of his friends (who have also ordered 8800's) will probably work together on developing I/O boards and a priority interrupt system.

-- And, Lynn, don't worry about providing "camera ready" copy for the newsletter. We're very happy to hear what you newcomers are interested in and planning to do. Those of you that have only sent us your name and address.....we, and everyone else, would like to hear of your plans and/or thoughts. --

C. R. Corner, 514 South 9th St., Moorhead MN 56560

Robert G. Confrey, PSC Box 4636, Beale AFB CA 95903

Richard C. Creighton, 1053 Princewood Dr., Orlando FL 32810

Mark G. Crook, 3 Bel Air Rd., Delmont PA 15626, would like to get the plans for the Mark-8. (They can be ordered from Radio-Electronics, Micro-computer, P.O. Box 1307, Radio City Station, New York N.Y. 10019, for $5.00)

L. Dabrowski, 1815 N. Karlov Ave., Chicago ILL 60639

Gary L. Dickman, 707 Elm St., Hays KA 67601, Ph: (913) 625-9693 or 462-3439

Steven Dompier, 2136 Essex St., Berkeley CA 94705, is currently building an Altair 8800 with disc drives.

James Dunion, 421 Ridgecrest Rd., Atlanta GA 30307, is a graduate student in Information and Computer Science at Georgia Tech. He has built a Mark-8 and the Altair 8800 and is looking for suggestions as to how he could make contributions to the User's Group. If you've got them things built, Jim, then it's software development time....and we'd love to share your efforts with the others.

Robert H. Erbe Jr., #263, 614 W. 36th Pl., Los Angeles CA 90007

David J. Evans, 754 S.E. Daytona Drive, Palm Bay FL 32905, Ph: (305) 724-2613, is an instructor in Computer Science and is currently working toward a Ph.D. in Electrical Engineering. He has built a microcomputer around the 8008 and is very interested in software and hardware developments.

Edward M. Evans, 46 Knoll Crest Ct., West LaFayette IND 47906, is an Instrument Specialist at Purdue University. His recently completed Mark-8 is currently in the de-bugging stage, and he has also built a TVT which works great.

Charles R. Goetowski M.D., 2506 Victor Ave., Glenview ILL 60025, Ph: 729-8886

John Ford, 5561 Esplanada Ave., Santa Maria CA 93454, has a noble project in mind. He is going to develope a modular CHESS program which hopefully can be reduced to 4K, using an overlay technique to roll program segments in and out of memory. (There, John, now you're committed.)

W.C. Fuller, 2377 Dalworth 157, Grand Prarie Texas 75050

Jeff Goldberg, 826 Cambria Ave., Santa Maria California 93454, Ph: (805) 937-5558

Jack Gray, 264 East 21st Street, Paterson NJ 07513

John Hardesty, 909 E. Grovemont, Santa Ana California 92706, is a "four-bit people". He is building a microcomputer using the Intel MCS-4040 and is, of course, interested in the same type of peripherals being built and acquired by us "8-bit people".

Robert L. Hatfield, 228 - 22nd St., Ashland Kentucky 41101

Gray Hoffman, 1502 Spillers Lane, Houston Texas 77043

Donald E. Houghton, 1864 Ortonville Road, Ortonville Missouri 48462

J.P. Hunter, 1207 Malgren Ave., San Pedro, California 90732

Doug Inglis, 2711 So. Eby St., Terrace, B.C. Canada V8G 2X4, is a Computer Science teacher with an interest in micro & mini computer development.

H.J. Kearns M.D., 14421 Wilson Road, Edmond, Oklahoma 73034

Carl R. Kelb, 3775 Nathan Way, Palo Alto, California 94306, has built an 8008 based microcomputer along with Don Lancaster's TVT modified to 80 characters per line.

. . . . . .

Speaking of Mr. Don Lancaster...he sent in a check for membership in the User's Group and Hal Singer promptly returned it with a letter expressing his (and our) gratitude for the contributions he has made to the microcomputer hobbyist field. Thanks again, Don, and welcome aboard.

. . . . . .

Robert A. Kern, Chief - Computer Services Unit, A-306 East Fee Hall, Michigan State University, East Lansing, Michigan 48824, indicates that his department is seriously considering construction of a microcomputer system.

Jack W. Klincher, 15448 Meyers Road, Detroit, Michigan 48227, Ph: (313) 345-4974, is "investigating the field" and trying to decide which microcomputer system he is going to build.

Dick Kopitzke, 200 W. Midway Drive #177, Anaheim, California 92805, is quite enthused about getting started on the Mark-8 and plans to use it as a "brush-up" learning tool (he has previously worked on some of Autonetic's 2nd & 3rd generation nav system computers). He is also looking for employment in "electronics test, computer/digital test, engineering assistant or a Quality assurance function related to those postitions." Phone: (714) 535-7001.

Richard Kulka, 283 Poplar St., Mansfield, Ohio 44903, is in the process of making the PC boards for the Mark-8 and the TVT. He's been playing with electronics for the past 10 years and is looking forward to getting into this latest thing.

Charles (Chuck) W. Leroy, 116 Somerset Pl., Lompoc, California 93436, Ph: (805) RE 6-5475, is currently checking out an Altair 8800 and will be interfacing it with a surplus keyboard, TVT II (modified for direct mem access, Read/Write, & without keyboard), cassette, and graphic CRT display. He's very happy about the fact that PCC is developing a mini-BASIC. (I think we'll all be happy to see a BASIC which we can all use.)

Charles A. Lewis, 3435 Woodmar Courts, West Lafayette, Indiana 47906, has completed the Mark-8 but is having trouble getting it running (perhaps the past NL's will help solve the problem, Charles. If not, let us know.) He says it will address, load, & examine....but not run. He has plans for interfacing a mag tape when it becomes operational.

Ric Martin, WA5TML, 13709 Preston Rd. #232, Dallas, Texas 75240, is building the Mark-8 and TVT I. A Klienschmidt 311 RO will be used for hard-copy (at 300 baud).

Harrison R. McCray Jr., 6716 Rugby Lane, Charlotte, North Carolina 28211

W.H. McCarter, PO Box 864, Slidell, Louisiana 70458

Thomas E. Morgan, 3904 King Arthur Road, Annandale, Virginia 22003

Christopher S. Ott, 47 Mueller Dr., Charleston, South Carolina 29407, is looking for a de-bugged & running Mark-8 for purchase.

Richard A. Peterson, 9004 184th Ave. E., Sumner, Washington 98390, is in the final stages of assembling his Mark-8 & TVT. He heard about us thru Maury Goldberg's "Mini-Micro Mart" and would like to know if anyone has tried driving the UNIVAC 0769 printer that is advertised? He's going to order the MIL Monitor 8 ROM (correction....has ordered). (Most of the questions in your letter, Richard, will be answered in the previous NL's.)

Richard W. Raver, 2842 N. Richmond St., Chicago, Illinois 60618

David P. Rennie, P.O. Box 33388, Houston, Texas 77033, heard about us thru the Digital Group Clearinghouse.

Richard Rhinevault, 79 Sprucewood Circle, Geneva, New York 14456

Joe D. Ringland, P.O. Box 4784, Rolling Bay, Washington 98061

William (Bill) J. Risch, 471 Geneva Apt 212, Aurora, Colorado 80010

J.B. Ross, Asst. Prof. of Physics, Dept. of Physics, Park College, Kansas City, Missouri 64152

Jeffrey P. Royer, 1568 Mariposa Ave., Palo Alto, California 94306, states that "we are in the process of building a Mark-8", and they're planning to use it as an educational exercise and later for computer-controlled games & musical applications. (Who are "we", Jeff?)

Frederick L. Kahl, T&F Electronics, 704 Courtland Circle, Western Springs, Illinois 60558

John D. Turner, NAVSEEACT Japan, Box Five, FPO Seattle, Washington 98762, is awaiting an Altair 8800. He is working near Tokyo at the present and expects to return to the states next year.

Duane L. Gustavus, 818 W. Hickory, Denton, Texas 76201, is completing construction of an Altair 8800 w/1K of static memory. He expressed slight reservations for not going the 8008 route because of the support from groups such as ours. (Don't worry, Duane, it looks like this group is going to be just as active in 8080 software & hardware development as the 8008.)

William H. Freeman, 816 Meadowlark Lane, Glenview, Illinois 60025, is a registered Professional Engineer who recently attended a course on microprocessors at Northwestern University. One of the instructors (Prof. W.J. Lennon) suggested the NL would be of interest. (We hope so.)

Ernst J. Schubert Ph.D., Computer Institute for Continuing Education, POB 851, Los Alamitos, California 90720, is one of the founders of the Institute, and has started a project to build a microcomputer and interface it with an ASR 33 and cassette. He plans to develope an Operating System using BASIC and will be happy to share the results with the User's Group.

William E. Shawcross, 1105 Massachusetts Ave., Cambridge, Massachusetts 02138 Ph: (617) 547-7652

Neal Sheffield Jr., D.D.S., W4ZPZ, 2601 Oakcrest Avenue - Suite E, Greensboro, North Carolina 27408

Robert Sherman, 1010 North Palm Ave. #201, Los Angeles, California 90069

Randy Soderstrom, 4601 Goldfinch Dr., Madison, Wisconsin 53714

Charles D. Soucek M.D., 2807 Ohio, Kansas City, Kansas 66102

Capt. John R. Stanton, 2930 Currie St., Biloxi, Mississippi 39531

Donald E. Tarbell, 144 Miraleste Dr. #106, Miraleste, California 90732, is a busy man indeed. He designed and built his own computer system and has been operating it for the last four years. The CPU is an 8-bit, byte-oriented machine with 16-bit Accumulator, Prog Cntr, & Index Registers (5). It has 60 instructions w/8 addressing modes (both to be expanded). The peripherals include a Model 32 TTY, CRT & keyboard, 500 Kbyte disc, modem, X-Y Plotter & audio cassette. He intends to install 2 more CPU's - - an 8080 and a microprogrammable processor with Writable Control Store. The three processors will share memory (36 Kbytes) and peripherals. He has also developed extensive software which includes a BASIC interpreter and a Sentence Processing Language. (Ya gotta lotta guys boilin' over with envy, Don. I hope yur happy!)

Gary D. Thurmond, 1040 Meadows End Dr., Calabasas, California 91302, W6STR, Ph: (213) 340-1664, just completed the TVT and is waiting for a MOD-8 from Mini-Micro Mart (and the MIL Monitor-8 Rom). He's quite enthused about this new hobby & has really been bitten by the bug.

S. Joseph Toy, Route 3 Box 73, Chico, California 95926

Laurence E. Turner, #102 3023 Blakiston Dr. N.W., Calgary, Alberta, Canada, is interested in the MOD-8 and is wondering if layouts (or negatives) are available for making his own boards? He is presently building some terminals and cautions that the General Instruments keyboard encoder sold by B&F Enterprises is likely not the standard 2376 ASCII encoder as claimed. (Out of two completely separate orders both encoders he recv'd were custom encoders - & B&F would not accept them for return.)

Robert A. Van Winkle, 288 Woodbridge St., Yuba City, California 95991, has two Micro-8's (Mark-8's ?) under construction and "is anxious to meet and/or correspond with anyone working with these items." (We might be able to help you out there, Bob.)

Andrew Vics, Media Co-ordinator, Pequannock Township High School, Sunset Road, Pompton Plains, New Jersey 07444, indicates his school is currently involved in microcomputer experimenting and is interested in the TVT.

L.G. Walker, Rt 1 Box 272, Aledo, Texas 76008, is an analyst and scientific programmer with 16 years of experience in the field. He is currently working as a member of a software team designing and building mobile computerized systems. (And, we sent him a requested copy of the NL so that he can get bit by da bug, too.)

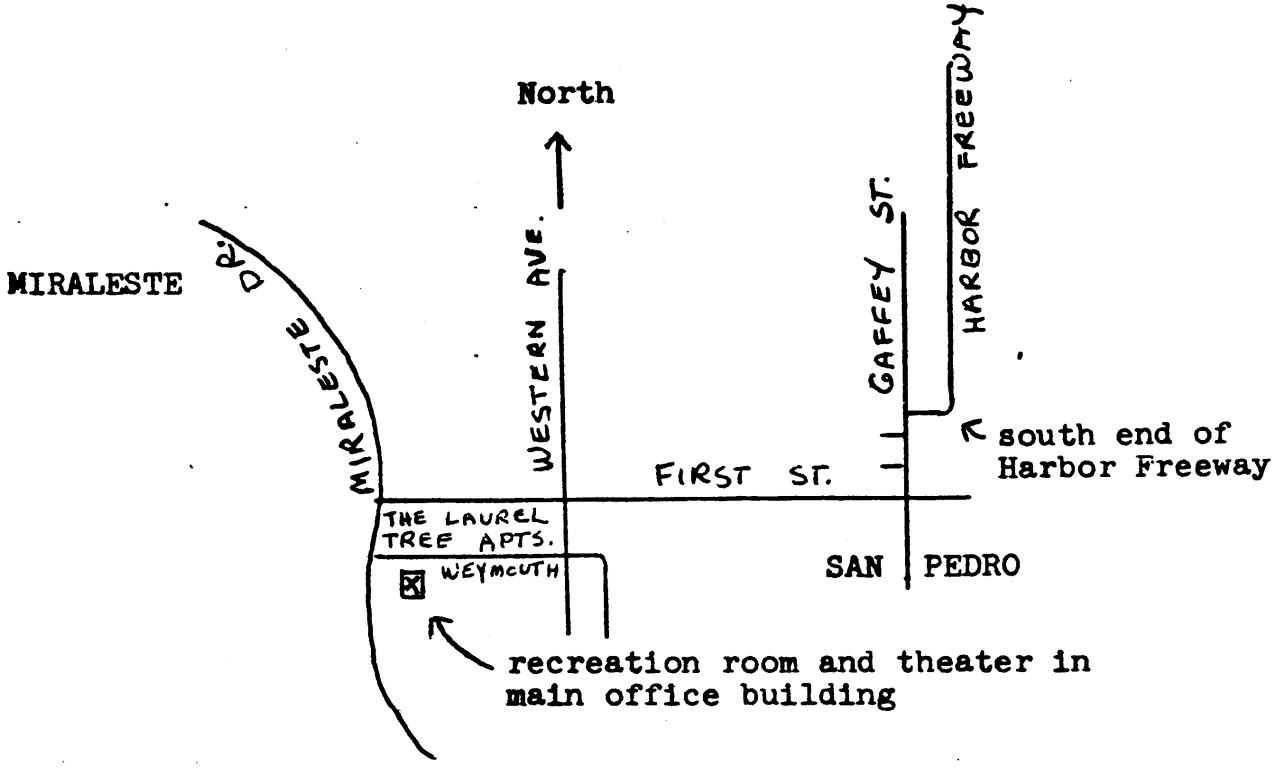
Tate Yoshida, 2951 S. Martin Luthur King Jr. Dr., Chicago, Illinois 60616, heard about us thru the Digital Group.

Bruce Robinson, R[illegible] Gananoque, Ontario, Canada, is an instrumentation design engineer doing analog-to-digital conversion using an 8008 based PRO-LOG microcomputer (for a living). He's very interested in microcomputer applications for model railroading and has some things working in that area. He has also come up with some inventions in the Ham radio field.

Robert Zahorcak, 40 Maple Lane, Lake Hiawatha, New Jersey 07034

Pearce Young, 9478 Cherokee Lane, Beverly Hills, California 90210, is a Los Angeles County Superior Court judge with an interest in electronics which was previously concerned with ham radio and is now actively involved in the computer hobbyist field (by way of a MITS Altair 8800). He, together with Don Tarbell (see page) and a few others, are organizing a meeting of enthusiasts in the Los Angeles area for June 15th. The meeting will be at 2:00 P.M. in the recreation room at Don Tarbell's apartment building (Laurel Tree Apts.). Please drop a note to Don and let him know if you will be attending and how many in your party (Donald E. Tarbell, 144 Miraleste Drive, Apt. 106, Miraleste, Calif. 90732). If you have any items to sell or trade, bring them along. Also, if it is not too large, feel free to bring your own computer. Following is a map showing how to get to Don's apartment:

MAP TO MEETING PLACE



NOTE:

It would be best to park along Miraleste Drive, since most of the apartment spaces are numbered, and subject to towaway. There may be a few "guest" parking spaces available if you arrive early.

Introduction to the MIKE 2

Martin Research has received hundreds of inquiries about our microcomputer modules--both from readers of our book, *MICROCOMPUTER DESIGN*, and from computer hobbyists. Demand for the *MIKE 201* exceeded our initial supply of printed circuit boards. Volume production of PC boards is now underway, and we expect to begin shipping again on June 16, 1975.

Before you decide you can't wait, let us describe what we believe is the best microcomputer yet available.

FIRST: All of our parts are new and of top quality. No factory seconds or temperature rejects.

SECOND: All printed circuit boards are commercial quality, made by professional PC houses to our specifications. Component holes are plated through for easy solderability. Each board comes with its own connector for rapid insertion and removal.

THIRD: The system architecture has been designed for optimum flexibility. It uses a carefully-designed bus structure which permits *any* board to be inserted in any position on the bus. To expand the system, you simply plug more boards onto the bus.

FOURTH: All critical control signals needed throughout the system are decoded right on the CPU board and piped around the system on the bus. Hardware interfaces to the microprocessor generally need only two chips--one strobe decoder, and one latching device (for output ports) or three-state driving device (for inputs).

FIFTH: There are no critical adjustments during assembly. The main timing is controlled by a reliable crystal on the CPU board.

The Boards

The *MIKE 203* system contains four circuit boards, described below.

MIKE 2-1 CPU BOARD An 8008, crystal-controlled oscillator, and all the timing for the system. The bidirectional bus drivers on the CPU board allow many accessory boards to be added to the microcomputer.

MIKE 2-20 CONSOLE BOARD The *MIKE 2-20* has six large, 0.3-inch seven-segment display digits and a twenty-key calculator-type keyboard. Unlike systems with banks of toggle switches and lights, this micro is easy to program, since codes are easily visualized. The six digits are driven by decoding latches, and can be read in octal, decimal, or hex; they are user-accessible, constituting three output ports.

MIKE 2-3 PROM/RAM BOARD This board has room for up to 1K of RAM (1024 eight-bit bytes, or 8192 bits) and 2K of PROM (2048 bytes). Handles up to eight 2112s and up to 8 1702As. The basic system, the *MIKE 203*, uses 256 words of RAM (two 2112s) and 256 words of PROM (one 1702A). The PROM is preprogrammed with our monitor program, described below.

MIKE 2-15 BREAD-BOARD

A standard-sized *MIKE 2* board, with a connector for interfacing to the system bus. All power and bus signals are accessible. Pre-drilled with over 700 holes, with half the board configured for standard 14- and 16-pin DIPs (+5 and ground at the corners), and the other half with universal spacing, for any size sockets.

MIKE 2 Monitor

Just as important as the hardware is the *MIKE 2 MONITOR* that comes with the *MIKE 203*. Consisting of 256 words in a single 1702A PROM, *M2M* allows you to write instructions into memory at any location, and read instructions, simply by punching the keys on the keyboard.

DATA ENTRY

Instructions and data are entered into the *MIKE 2* simply by punching the keyboard. With a little practice, it's just as fast as your pocket calculator. As a key is pressed, the digit appears on the LED display. When a second digit is entered, the first automatically moves left and the second takes its place. For example, say you are entering the number *135*, which is the octal number for an 8008 *OUTPUT 16* instruction. You type in the *1*, and the display reads *000001*. Then you press the *3*, and the digits say *000013*. Finally, you press the *5*, and the display reads *000135*. (Incidentally, you do not have to punch in leading zeros.) As the data appears on the displays, it is also loaded into the 8008's E register, where it is available for further use.

A number of special-purpose keys on the keyboard are used to program the microcomputer.

LOAD H KEY

This key is used to address memory. It transfers the contents of the 8008's E register to its H register, then clears the E register (to 000). For example, you have just entered *010* onto the display; now you press this key. The 8008 H register address is now *010*, and this is the new high-order memory address for writing into and reading from memory.

LOAD L KEY

This key also references memory, at its *low-order* address. Together, the LOAD H and LOAD L keys can address any location in the 8008's full complement of 16K of memory. After the LOAD L key loads the L register from the E (clearing the E), the display blinks back and forth between the *address* of memory, and the *contents* at that address.

WRITE KEY

This key writes into RAM memory. It takes the value previously entered into the E register, writes into RAM at the H/L memory address, and clears the E register. Then it increments the L register (and the H if the L overflows), preparing the user for the next instruction. Now the display begins to alternate between the new address in memory and its contents.

READ KEY

This key reads the contents of RAM or PROM at any address, as selected by the LOAD H and LOAD L keys. The display alternates

between memory address and contents. Each time the READ key is pressed, the program automatically steps to the next-higher location in memory, so that you can check the contents of a whole program quite rapidly.

EXECUTE KEY This key jumps to location 010 000 in RAM memory and executes whatever program you have stored there.

INTERRUPT KEYS One key sets the interrupt flip-flop on the CPU board *low*. (Now no peripheral device on the *MIKE 2* bus can cause an interrupt.) The second key resets the flop, causing an interrupt to the 8008. The *M2M* program takes over, saving the status of the 8008's internal registers automatically. (The A register contents are at 013 370 in RAM; B, at 013 371; C, 013 372; D, 013 372; E, 013 374; flags, i.e, carry, parity, sign, zero, 013 375; H, 013 376; and L 013 377.) The digital display automatically flashes back and forth between location 013 370 and the contents of the A register. The other saved registers and flags can be read out in succession by pressing the READ key. This feature allows the user to use the 8008's interrupt capabilities without including status-saving software in each program.

CONTINUE KEY This key continues execution of the main program at the point where the CPU left off when it was interrupted.

Programming

Here is an example of a program for testing operation of the *MIKE 203*. First, the program; then, how you load it in and execute it.

THE PROGRAM

010000	304	LAE	LOAD A REGISTER FROM E REGISTER
010001	004 021	ADI 021	ADD 021, I.E., 00 010 001 (OR, 0001 0001)
010003	100 007 010	JFC 010007	IF THE A REG. HAS NOT YET OVER-FLOWED, FALL THRU NEXT INSTRUCT.
010006	250	XRA	OVERFLOW; CLEAR A REGISTER
010007	340	LEA	STORE THE COUNT IN THE E REG.
010010	133	OUT 15	OUTPUT TO DIGITS 1 & 2, 4 BITS FOR EACH DIGIT
010011	135	OUT 16	OUTPUT TO DIGITS 3 & 4
010012	137	OUT 17	OUTPUT TO DIGITS 5 & 6
010013	045	RST 040	JUMP TO LOCATION IN M2M PROGRAM WHICH DEVELOPS 1.2 SEC. DELAY
010014	104 000 010	JMP 010000	START AGAIN

ENTERING THE PROGRAM

1. *1, 0.* (Display reads *010.*)
2. *LOAD H.*
3. *LOAD L.* (This loads L register with *000*. Now memory location is *010000*. Digits flash this address and its current contents.)
4. *3, 0, 4.* (*304* on display.)
5. *WRITE.* (Display moves to next location.)
6. *4, WRITE.*
7. *21, WRITE.*
8. *100, WRITE.*
9. *7, WRITE.*

11. *16, WRITE.*
12. *256, WRITE.*
13. *340, WRITE.*
14. *133, WRITE.*
15. *135, WRITE.*
16. *137, WRITE.*
17. *45, WRITE.*
18. *104, WRITE.*
19. *WRITE.* (Unnecessary to punch *000.*)
20. *10, WRITE.*
21. *EXECUTE.*

The display will now count as follows, changing once every 1.2 sec.:
111111 222222 333333 444444 555555 666666 777777 888888
999999 AAAAAA bbbbbb CCCCCC dddddd EEEEEE FFFFFF 000000
111111 ETC.

The System

MECHANICAL The *MIKE 203* is complete except for a cabinet of your own design, and a power supply. The four boards provided each measure 5.5 by 7.0 inches (140 by 178 mm), and include a fifty-pin connector at the rear which connects to a fifty-wire flexible cable. The boards stack up, one on top of the other, and are separated by rigid metal spacers (provided). Eighteen inches (457 mm) of cable are supplied with the *MIKE 203*, more than enough for foreseeable expansion.

POWER Power requirements for the basic *MIKE 203*, including only those basic memory chips detailed above: +5 V, 1.4 A; -9 V, 70 MA. A power supply providing these voltages and ±12 volts as well is under development and will be announced shortly.

Extras

A number of circuit boards are under development:

MIKE 2-5 4K RAM BOARD Available as this sheet goes to press, our 4 K RAM board has room for up to thirty-two 2102 static RAM chips. It includes decoders and bus drivers for complete interfacing with the *MIKE 2* system. Adds up to 4,096 bytes (32,768 bits) of memory. Requires a single +5-volt supply.

MIKE 2-6 CRT DISPLAY INTERFACE.

MIKE 2-9 CASSETTE RECORDER INTERFACE, for use with audio machines.

MIKE 2-10 TELETYPE INTERFACE, for hard copy. Includes control ROM.

MIKE 2-14 PROM PROGRAMMER, for 1702A PROM memories.

MIKE 2-? YOUR SUGGESTIONS APPRECIATED.

For further details, stay in contact with Martin Research. To be added to our mailing list, just drop us a postcard or call us up; we'll send you bulletins as new developments occur.

5/75

MR RESEARCH
1825 S. Halsted St.
Chicago IL 60608
(312) 829 6932

Pagezz

mike 2

PART	DESCRIPTION	BOARD & CONNECTOR ONLY	KIT	ASSEMBLED & TESTED
MIKE 2-1	CPU board, without 8008	19.95	54.95	74.95
MIKE 2-1A	CPU board with 8008	-----	94.95	114.95
MIKE 2-1B	CPU board with 8008-1, fast crystal (60% faster than standard 8008)	-----	99.95	119.95
MIKE 2-20	Console board	19.95	68.95	83.95
MIKE 2-3	PROM/RAM board with 1-1702A (*M2M* program), 2-2112s	19.95	78.95	93.95
MIKE 2-15	Breadboard	19.95	19.95	-----
MIKE 2-030	18" cable and spacers	-----	4.50	6.00
MIKE 2-100	*MIKE 2* MANUAL	-----	10.00	10.00
MIKE 203	*MIKE 2* microcomputer system. One each: 2-1, 2-20, 2-3, 2-15, 2-030, 2-100. Without 8008.	-----	229.95	279.95
MIKE 203A	*MIKE 203*, with 8008 microprocessor	-----	269.95	319.95
MIKE 203B	*MIKE 203*, with 8008-1, fast XTAL	-----	275.95	325.95
MIKE 2-5	4K RAM board. No 2102s, decoders	28.95	-----	-----
MIKE 2-5A	With decoders, 1K of RAM, 450 ns speed	-----	77.95	99.95
MIKE 2-5B	With 2K of RAM, 450 ns	-----	107.95	130.95
MIKE 2-5C	With 3K of RAM, 450 ns	-----	136.95	160.95
MIKE 2-5D	With 4K of RAM, 450 ns	-----	164.95	189.95
2112	Extra 2112 256 x 4 RAM ICs for *MIKE 2-3*	-----	4.75	-----
1702A	Extra 1702A PROMs for *MIKE 2-3*	-----	24.95	-----
MOD-8	7-1702A PROMs, programmed and tested with Microsystems' *MOD-8* monitor program	-----	-----	159.95
MOD-8/CREED	8-1702A PROMs, programmed and tested with modified *MOD-8* for use with 5-bit *CREED* teletypewriters	-----	-----	179.95
MIKE 2-151	Universal I/O breadboard kit: breadboard, chips for interfacing any eight-bit parallel input/output source. With *MIKE 2-15*, connectors, instructions, 1-74LS138/3205, 1-74273, 2-74125s.	-----	27.95	-----
MIKE 2-16	*CREED* interface board. Connects 5-bit *CREED* machine with *MIKE 2*. Available as kit only (*MIKE 2-15* plus kit of parts).	-----	34.95	-----
MICRO-COMPUTER DESIGN	A 300+-page book on hardware design around the 8008. Write for details, pricing. Price with *MIKE 2* purchase of $200 or more:		-----	50.00

mike 2

HANDLING CHARGE Our minimum order is $25.00. Orders for less than this will be charged an additional $1.00 for handling.

TAX Illinois residents, please add 5% state sales tax.

SHIPPING *Within the United States:* shipping is free on *prepaid* orders. Orders requiring billing will be charged for shipping costs. We ship via United Parcel Service (by mail in non-UPS areas); be sure to give a street address suitable for UPS delivery. Air shipment (air mail or UPS blue label) is available at cost.

TERMS Recognized corporate and institutional purchasers, net 30 days. Individuals: for fastest service, use a cashier's check, money order, or *MASTER CHARGE*. We reserve the right to delay shipments while personal checks clear through our bank. When using *MASTER CHARGE*, please supply your account number and its expiration date.

QUANTITY PRICES Please contact Martin Research for quotations on quantity prices on *MIKE 2* parts and systems. Educational rates have been established for the book *MICROCOMPUTER DESIGN*; educators, please write on your letterhead for price lists.

OUTSIDE UNITED STATES Our export agent reserves the right to request prepayment in advance of shipment. Shipping is charged at cost; please specify surface mail or air shipment.

WARRANTY POLICY Parts and circuitry guaranteed to operate as advertised. Parts on warranty for thirty days. However, we cannot be held responsible for damage to parts determined to result from carelessness or misuse. In case of problems, please write Martin Research with an explanation. In order to ensure credit, please do not return goods until you have received written permission.

SHORT FORM ORDER BLANK

Name _____ *Phone* _____

Company/Institution _____

Address _____

_____ *Zip* _____

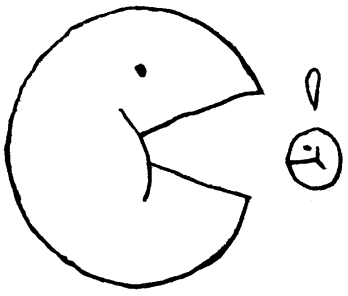
ITEM	*PRICE*	*ITEM*	*PRICE*
		Subtotal (for goods)	
		Ill. residents: 5% tax	
		Handling charge; omit for order of $25.00+	*$1.00*
Subtotal		*TOTAL ENCLOSED*	

1825 S. Halsted St.
Chicago, IL 60608
(312) 829-6932

master charge

Page 24

mike 2



CHOMP!

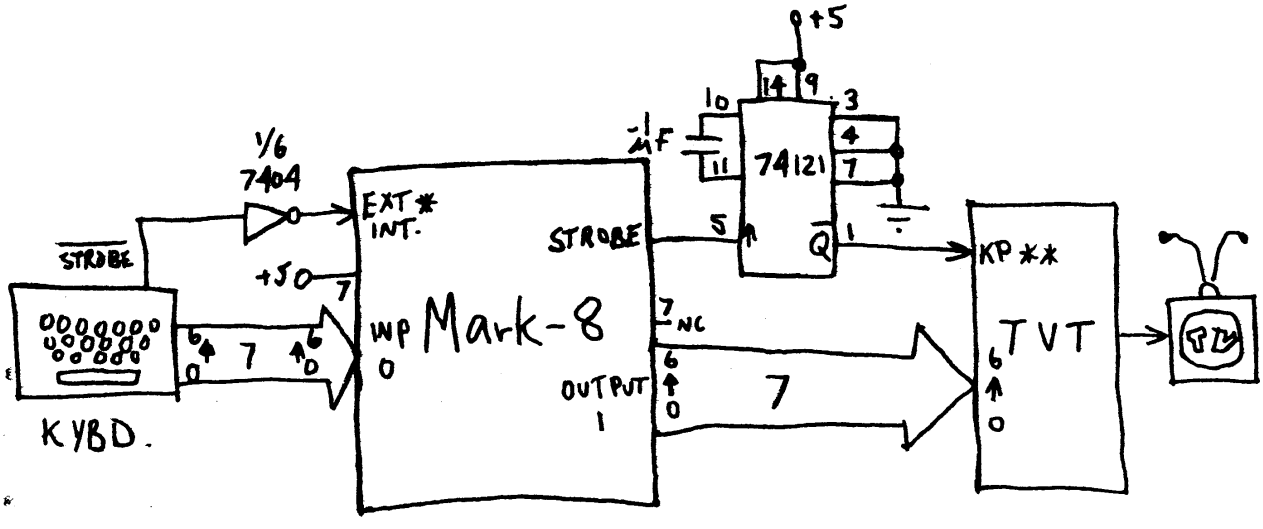
By

Phil Mork
12 Woodland Road
Weston, MA 02193

Any number of players can play Chomp. It is played (in this version) on an 8 by 8 board of +'s with a * in the upper right hand corner. Players take turns chomping until one player is forced to take the poison *. To take a chomp, you type in the letter and number of one of the +'s. That +, and all of the +'s to the left and below disappear. No fair chomping empty space.

This version of Chomp uses the Mark-8 with the TV Typewriter I and an ASCII keyboard. The TVT is on output port 1, and the keyboard is on input port 0. Strobe lines are appropriately connected(see below).
First load the " Load Comments" program then interrupt with a Restart 005. Now set the switch register to 300 (NOP) and type in the comments using returns and line feeds as shown. The output 0 LED's will count characters as they are loaded. LED's should read 365 when done.

Now load the program from 0 000 to 0 164. Interrupt with 005, then set switch register to 300, home TVT and press any key. When the game is over, clear the screen and hit any key to play again. Have fun!



* Remove jumper!
** Remove cursor C16.

Phil Mork

CHOMP by Phil Mork April 21, 1975

Load Comments

```
1 000 066  LLI
  001 000
  002 056  LHI
  003 001
  004 306  LAL
  005 121  OUT 010
  006 000  HLT
  007 101  INP 000
  010 123  OUT 011
  011 370  LMA
  012 060  INL
  013 104  JMP
  014 004
  015 000
```

Comments

```
1 000   TYPE A LETTER AND A NUMBER(ret)
        TO TAKE A BITE. NO FAIR BITING(ret)
        AIR. BITE THE * AND YOU LOSE(3 ret)
        ABCDEFGH(ret)
        +++++++*1(ret)
        ++++++++2(ret)
        ++++++++3(ret)
        ++++++++4(ret)
        ++++++++5(ret)
        ++++++++6(ret)
        ++++++++7(ret)
        ++++++++8(7 ret)
        GO!(2 space) (LF) (space)

1 300   (6 ret)NEXT?(LF)

1 314   (ret)CHOMP! YOU LOSE. PLAY AGAIN?(11 ret) (LF)
```

-Program-

```
000 000 HLT
    066 LLI
    000
    106 CAL
    150
    000
    106 CAL
    117
010 000
    340 LEA
    074 CPI
    010 010
    150 JTZ
    102
    000
    106 CAL
020 117
    000
    320 LCA
    334 LDE
    016 LBI
    010 011
    006 LAI
    015 015
030 106 CAL
    121
    000
    006 LAI
    015 015
    106 CAL
    121
    000
040 011 DCB
    021 DCC
    110 JFZ
    033
    000
    006 LAI
    040 040
    106 CAL
050 121
    000
    041 DCE
    110 JFZ
    045
    000
    006 LAI
    015 015
060 106 CAL
    121
    000
    343 LED
    011 DCB
    110 JFZ
    045
    000
070 066 LLI
    300 300
    106 CAL
    150
    000
    104 JMP
    006
    000
100 -
    -
    106 CAL
    117
    000
    074 CPI
    001 001
    150 JTZ
110 140
    000
    104 JMP
    022
    000
    -
    -
    000 HLT
120 101 INP 001
    123 OUT 011
    044 NDI
    017 017
    056 LHI
    000 000
    051 DCH
    300 LAA
130 300 LAA
    300 LAA
    053 RTZ
    104 JMP
    126
    000
    -
    -
140 066 LLI
    314 314
    106 CAL
    150
    000
    005 RST
    -
    -
150 056 LHI
    001 001
    307 LAM
    074 CPI
    212 212
    053 RTZ
    106 CAL
    121
160 000
    060 INL
    104 JMP
    150
    000
```

Phil Mork's Music Program Modified For 8080's

By Chuck Leroy, 116 Somerset Pl
Lompoc, CA, 93436
(805)RE6-5975
(805)866-8995

MUSIC PROGRAM FOR ALTAIR 8800

Uses Sense Switches for Tempo

```
START   LXI H    MUSIC    SET ADDR.
TEMPO   IN       377      GET TEMPO FROM
                          SENSE SW.
        MOV D,A           PUT IN D
THERE   MOV A,M           GET NOTE
        CPI      377      377 ?
        JZ       START    YES, REPEAT
LOOP    DCR A             NO, DEC. FREQ.
        JZ       CLICK    0 ?
        DCR D             NO, DEC. TEMPO
        JNZ      LOOP     0 ?
        MOV B,A           YES, SAVE NOTE
        IN       377      GET TEMPO
        MOV D,A           PUT IN D
        MOV A,B           RESTORE NOTE
        DCR E             TIME TO GET NEW
                          NOTE ?
        JNZ      LOOP     NO, GO BACK
        INX H             YES, BUMP ADDR.
        JMP      THERE    GO BACK
CLICK   OUT      PORT     OUTPUT
        JMP      THERE    GO BACK
```

NOTE	VALUE
A	343
A#	326
B	312
MIDDLE C	277
C#	264
D	252
D#	240
E	230
F	217
F#	207
G	200
G#	174
A	162
A#	153
B	145
HIGH C	140
C#	132
D	125
D#	120
E	114
F	110
REPEAT	377

```
MUSIC   200   162   140   174
        230   162   162   174
        217   200   174   174
        207   200   162   174
        200   207   153   230
        114   200   145   230
        140   230   153   140
        200   217   145   145
        200   200   217   140
        217   162   217   145
        217   162   140   145
        230   174   140   145
        217   162   140   140
        217   125   140   145
        217   140   140   145
        217   140   230   162
        217   162   230   162
        264   200   140   207
        252   162   140   207
        230   145   140   252
        217   140   140   252
        125   140   145   377
        145   140   145
```

Try Sense Switch 11,12 for Tempo.

Handy Special Register Subroutines

By Dr. Robert Suding, C/O Digital Group, PO Box 6528, Denver, CO 80206

Register operations are a general nuisance on the 8008, especially when using the H&L registers. The following subroutines are part of a new 2K operating system to be available through the Digital Group.

The first thing to do is to make up two jumper plugs to connect the eight Port 5 inputs to the eight Port 5 outputs, and the eight Port 4 inputs to the eight Port 4 outputs.

The next item is to reserve storage addresses 000006, 000007, and 000050 - 000056 as temporary storage locations for registers A through L.

Now the subroutines (addresses may be moved as desired):

Notes

"H&L Register Save" places the current H&L into storage positions 000055 and 000056. The A register is temporarily saved and later restored to permit its utilization without loss of its original contents.

"H&L Register Restore" retrieves the values of H&L registers previously stored in storage positions 000055 and 000056.

"H&L Swap with D&E" exchanges these register sets to double the memory accessing capabilities of the 8008.

"Register A - L Save" places the current A - L registers into storage positions 000050 - 000056. No registers are lost in the operation. By placing this subroutine at the front of your storage dump routine, you can list off all 6 registers by simply sampling storage positions 000050 - 000056.

"Restore A - L Registers" loads registers A - L with the contents of storage positions 000050 - 000056. The former data in registers A - L is lost.

"Swap Current A - L Registers with Stored Ones" exchanges the current A - L Registers with those stored in storage positions 000050 - 000056. No data is lost in the exchange. The result of this operation is a 12 register microprocessor instead of a 6 register microprocessor.

Sincerely,

Robert

Dr. Robert Suding WØLMD

PROGRAM: SPECIAL REGISTER SUBROUTINES

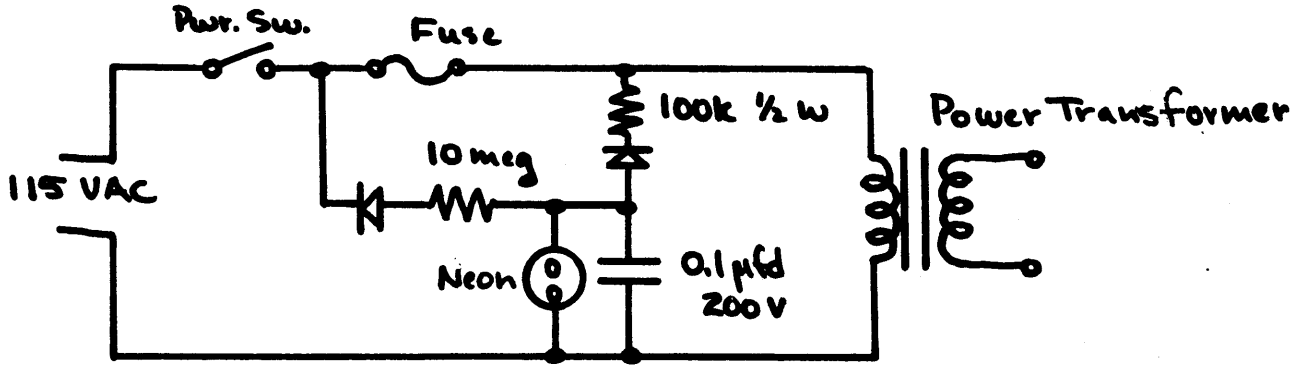
OCTAL ADDRESS	OCTAL CODE	OPERATION
001 000	***	H&L Register Save
001 001	133	Out 5 (Save A)
001 002	305	Load A with H
001 003	131	Out 4 (Save H)
001 004	306	Load A with L
001 005	056	Load H with 000
001 006	000	
001 007	066	Load L with 056
001 010	056	
001 011	370	Load Mem with A
001 012	061	Decrement L
001 013	111	In 4 (Retrieve H)
001 014	370	Load Mem with A
001 015	113	In 5 (Restore A)
001 016	007	Return Uncond.
001 017		
001 020		
001 021	***	H&L Register Restore
001 022	133	Out 5 (Save A)
001 023	056	Load H with 000
001 024	000	
001 025	066	Load L with 055
001 026	055	
001 027	307	Load A with Mem
001 030	131	Out 4 (Save H)
001 031	060	Increment L
001 032	367	Load L with Mem
001 033	111	In 4 (Retrieve H)
001 034	350	Load H with A
001 035	113	In 5 (Restore A)
001 036	007	Return
001 037		
001 040	***	H&L Swap with D&E
001 041	133	Out 5 (Save A)
001 042	305	Load A with H
001 043	353	Load H with D
001 044	330	Load D with A
001 045	306	Load A with L
001 046	364	Load L with E
001 047	340	Load E with A
001 050	113	In 5 (Restore A)
001 051	007	Return
001 052		
001 053		
001 054	***	Register A-L Save
001 055	133	Out 5 (Save A)
001 056	305	Load A with H
001 057	131	Out 4 (Save H)
001 060	306	Load A with L
001 061	056	Load H with 000
001 062	000	
001 063	066	Load L with 056
001 064	056	
001 065	370	Load Mem with A
001 066	061	Decrement L
001 067	111	Input 4 (Retrieve H)
001 070	370	Load Mem with A
001 071	061	Decrement L
001 072	374	Load Mem with E
001 073	061	Decrement L
001 074	373	Load Mem with D
001 075	061	Decrement L
001 076	372	Load Mem with C
001 077	061	Decrement L
001 100	371	Load Mem with B
001 101	061	Decrement L
001 102	113	Input 5 (Retrieve A)
001 103	370	Load Mem with A
001 104	007	Return
001 105		
001 106		
001 107	***	Restore A-L Registers
001 110	056	Load H with 000
001 111	000	
001 112	066	Load L with 050
001 113	050	
001 114	307	Load A with Mem
001 115	133	Out 5
001 116	060	Increment L
001 117	317	Load B with Mem
001 120	060	Increment L
001 121	327	Load C with Mem
001 122	060	Increment L
001 123	337	Load D with Mem
001 124	060	Increment L
001 125	347	Load E with Mem
001 126	060	Increment L
001 127	307	Load A with Mem
001 130	131	Out 4 (Save H)
001 131	060	Increment L
001 132	367	Load L with Mem
001 133	111	In 4 (Retrieve H)
001 134	350	Load H with A
001 135	113	In 5
001 136	007	Return
001 137		

Continued

PROGRAM: SPECIAL REGISTER SUBROUTINES

OCTAL ADDRESS	OCTAL CODE	OPERATION
001 140	***	Swap Current A-L
001 141		Regs with Saved Ones
001 142	133	Out 5 (Save A)
001 143	305	Load A with H
001 144	131	Out 4 (Save H)
001 145	056	Load H with 000
001 146	000	
001 147	306	Load A with L
001 150	066	Load L with 007
001 151	007	
001 152	370	Store A
001 153	066	Load L with 050
001 154	050	
001 155	307	Load A with Mem
001 156	066	Load L with 006
001 157	006	
001 160	370	Store A
001 161	066	Load L with 050
001 162	050	
001 163	113	In 5 (Retrieve A)
001 164	370	Store A
001 165	066	Load L with 006
001 166	006	
001 167	307	Load A with Mem
001 170	133	Out 5
001 171	301	Load A with B
001 172	066	Load L with 051
001 173	051	
001 174	317	Load B with Mem
001 175	370	Store A
001 176	302	Load A with C
001 177	060	Increment L
001 200	327	Load C with Mem
001 201	370	Store A
001 202	303	Load A with D
001 203	060	Increment L
001 204	337	Load D with Mem
001 205	370	Store A
001 206	304	Load A with E
001 207	060	Increment L
001 210	347	Load E with Mem
001 211	370	Store A
001 212	060	Increment L
001 213	307	Load A with Mem
001 214	066	Load L with 006
001 215	006	
001 216	370	Store A
001 217	111	In 4 (Retrieve H)
001 220	066	Load L with 055
001 221	055	
001 222	370	Store A
001 223	060	Increment L
001 224	307	Load A with Mem
001 225	131	Out 4 (Save L)
001 226	066	Load L with 007
001 227	007	
001 230	307	Load A with Mem
001 231	066	Load L with 056
001 232	056	
001 233	370	Store A
001 234	066	Load L with 006
001 235	006	
001 236	357	Load H with Mem
001 237	111	In 4 (Retrieve L)
001 240	360	Load L with A
001 241	113	In 5 (Retrieve A)
001 242	007	Return

PILOT LIGHT & BLOWN FUSE INDICATOR CIRCUIT 5/4/75



If fuse is good and power is one, neon lights all the time. If fuse blows, lamp flashes.

By Ted Lincoln WA6HWJ
410 Bell Avenue, Santa Ana, CA 92707

Memory Diagnostic

James R. Boddie
Apt 216
920 N. Dean Rd.
Auburn, AL 36830

```
0    056   LHI
1    PPP   PPP     PAGE UNDER TEST (NOT SAME PAGE THAT
2    066   LCI     PROG. IS LOADED)
3    000   000
4    016   LBI
5    TTT   TTT     TEST PATTERN
6    376   LML     WRITE IN MEM.
7    060   INL
10   110   JFØ     CONTINUE IF PAGE
11   006   006     NOT FILLED
12   000   000
13   307   LAM     READ FROM MEMORY
14   276   CML     COMPARE WITH PATTERN
15   110   JFØ     JUMP IF DIFFERENT
16   027   027
17   000   000
20   060   INL
21   110   JFØ     CONTINUE IF PAGE
22   013   013     NOT COMPLETELY
23   000   000     CHECKED
24   104   JMP     DO IT AGAIN
25   000   000
26   000   000
27   121   OUTØ    OUTPUT WRONG PATTERN
30   000   HLT     READ FROM MEM.
31   306   LAL
32   121   OUTØ    OUTPUT ADDRESS OF ERROR
33   000   HLT
34   104   JMP     CONTINUE
35   020   020
36   000   000
```

- PROG. LOADS MEMORY ADD. INTO MEM. LOCATION THEN READS FOR ERROR.
- FOR AN ARBITRARY TEST PATTERN
  1. PUT PATTERN IN (5)
  2. CHANGE (6) TO 371
  3. CHANGE (14) TO 271

THE IDEA OF CHECKING THE MEMORY ADDRESS CATCHES BEFORE PLUGGING THE 8008 IN IS GOOD, BUT I FOUND IT NECESSARY TO CLEAR THE R/W FF (½ IC 14) BY MOMENTARILY GROUNDING PIN 14 ON THE EMPTY 8008 SOCKET.

SINCERELY, James R. Boddie

PROGRAM TITLE MEMTEST 1101 NO. 1 A, B, C, D Page 1 of 2

No.		Code	Key
000	056	LHI	Load H Immediate
1	001	DATA,	00 000 001
2	066	LLI	Load L Immediate
3	000	DATA,	00 000 000
4	016	LBI	Load B Immediate
5	377	DATA	11 111 111
6	026	LCI	Load C Immediate
7	000	DATA	00 000 000
010	036	LDI	Load D Immediate
11	377	DATA	11 111 111
12	104	JMP	Uncond. Jump to
13	016	ADDRESS	Start AC, 00 001 110
14	000		—
15	060	Loop Ø, INL	Increment L = L+1
16	302	Start AC, LAC	Load A ≤ C
17	370	LMA	Load M ≤ A
020	301	LAB	Load A ≤ B
21	276	CPL	Compare A ≤ L
22	110	JFZ	Jump False Zero to
23	015	ADDRESS	Loop Ø INL
24	000		—
25	056	LHI	Load H Immediate
26	001	DATA	00 000 001
27	066	LLI	Load L Immediate
030	000	DATA	00 000 000
31	104	JMP	Uncond. Jump to
32	035	ADDRESS,	Read A, C, 00 011 101
33	000		—
34	060	Loop 1, INL	Increment L = L+1
35	306	Read A, C, LAL	Load A ≤ L
36	121	OUT/	Output A to Port Ø
037	307	LAM	Load A ≤ M

No.		Code	Key
040	272	CPC	Compare A ≤ C
41	110	JFZ	Jump on False Zero
42	117	ADDRESS	to- BAD 01 001 111
43	000		00 000 000
44	301	LAB	Load A ≤ B
45	276	CPL	Compare A ≤ L
46	110	JFZ	Jump on False Zero to
47	034	ADDRESS	Loop 1, INL
050	000		—
51	006	LAI	Load A Immediate
52	000	DATA	00 000 000
53	121	OUT/	Output A to Port Ø
54	000	Test Passed HLT	Halt - 300/INT to Cont.
55	056	LHI	Load H Immediate
56	001	DATA	00 000 001
57	066	LLI	Load L Immediate
060	000	DATA	00 000 000
61	104	JMP	Uncond. Jump to
62	065	ADDRESS	Start B, D, 00 110 101
63	000		—
64	060	Loop 2, INL	Increment L = L+1
65	306	Start B, D LAL	Load A ≤ L
66	121	OUT/	Output A to Port Ø
67	373	LMD	Load M ≤ D
070	307	LAM	Load A ≤ M
71	273	CPD	Compare A ≤ D
72	110	JFZ	Jump on False Zero to
73	117	ADDRESS,	Bad 01 001 111
74	000		—
75	372	LMC	Load M ≤ C
76	301	LAB	Load A ≤ B
077	276	CPL	Compare A ≤ L

Remarks:

START WITH A RST-0 (005/INT)
END OF 0's CHECK INDICATED BY ALL 0's DISPLAYED AT PORT Ø
CONTINUE WITH A CONT (300/INT)
END OF 1's WALK THROUGH INDICATED BY ALL 1's AT PORT Ø - INT TO CONTINUE
END OF 1's CHECK INDICATED BY ALL 0's AT PORT Ø - INT (300) TO CONT.
END OF 0's WALK THROUGH INDICATED BY ALL 1's AT PORT Ø

(Use reverse side for formulas, flow charts, notes, etc.)

M. PAUL FARR
Printed in U.S.A.
700-0715
1-73-2M

PROGRAM TITLE MEMTEST 1101 NO. 1 A, [redacted], D Page 2 of 3

No.		Code	Key
100	110	JFZ	JUMP ON FALSE ZERO TO
101	069	ADDRESS	LOOP 2 00 110 100
02	000		—
03	000	TEST PASSED, HLT	HALT - 300 INT TO CONT.
04	056	LHI	LOAD H IMMEDIATE
05	001	DATA	00 000 001
06	066	LLI	LOAD L IMMEDIATE
07	000	DATA	00 000 001
110	026	LCI	LOAD C IMMEDIATE
11	377	DATA	11 111 111
12	036	LDI	LOAD D IMMEDIATE
13	000	DATA	00 000 000
14	104	JMP	UNCONDIT. JUMP TO
15	016	ADDRESS	START A, C, 00 001 110
16	000		—
0_{10}	117_8	000 HLT	BAD - DEFECTIVE
		1101 AT ADDRESS DISPLAYED PORT 0	

No.		Code	Key

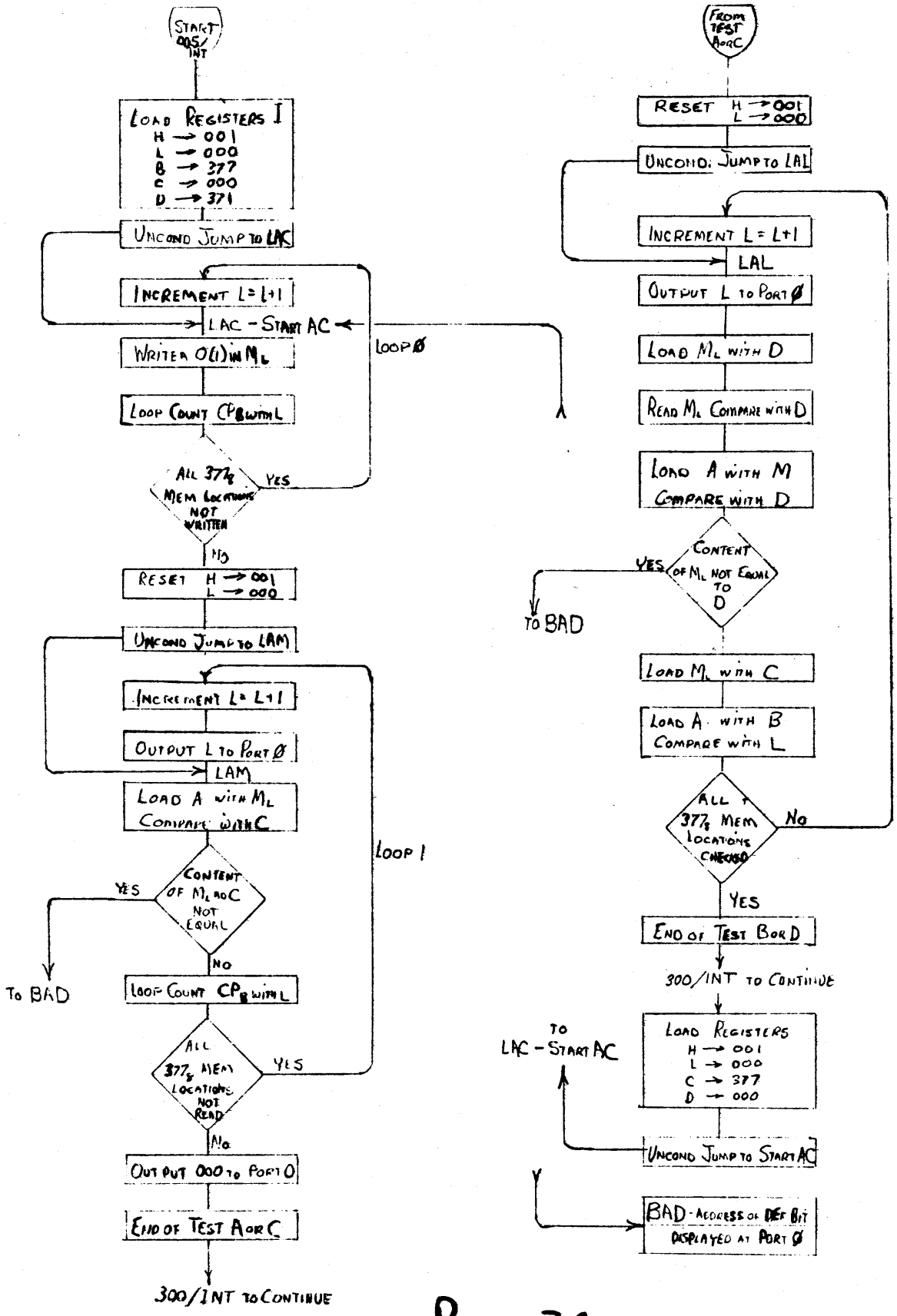
Remarks: IF AT ANY POINT IN THE TEST A ERROR IS FOUND IN THE CONTENTS OF A MEMORY LOCATION, THE PROGRAM WILL EXIT TO "BAD" INDICATED BY THE LOW ADDRESS DISPLAY OF 117_8 (01 001 111). THE ADDRESS OF THE "DEFECTIVE BIT" WILL BE DISPLAYED AT OUTPUT PORT 0 ON CONTROL PANEL.

(Use reverse side for formulas, flow charts, notes, etc.)

M. PAUL FARR
Printed in U.S.A.
700-0715
1-73-2M

MEMTEST 11C1 #1 A, B, C, D

FLOW CHART



M PAUL FARR

A SHORT PROGRAM TO ENTER, DISPLAY, AND STORE ASCII TEXT

By M. Paul Farr, 3723 Jackstadt, San Pedro, CA 90731

Note: See TV Typewriter interface in NL #5.

Address	OP Code	Mnemonic	Descrip.
000	101	INP-0	Input Char.
1	123	OUT-A	Echo To TVT
2	370	LMA	Load To Mem
3	060	INL	Inc Add Low
4	000	HLT	Halt
5	056	LHI	Load Addr High
6	001	001	Page 001
7	066	LLI	Load Addr Low
010	000	000	Loc. 000
1	307	LAM	Load A with M
2	106	CAL	Call Out Subr.
3	040	Output	
4	000	Subr.	
5	060	INL	Incr. Add Low
6	006	LAI	Load A Immed.
7	377	377	11111111
020	276	CPL	Compare A With L
1	150	JTZ	Jmp True Zero
2	004	Start	To Start
3	000		
4	104	JMP	Jump False Zero
5	012	Next	To Next Char.
6	000	Char.	

Output Subroutine
(25 Char./Sec. Delay Loop)

040	123	OUT-A
1	026	LCI
2	003	003
3	031	DCD
4	220	JFZ
5	043	Loop 1
6	000	
7	031	DCC
050	110	JFZ
1	043	Loop 2
2	000	
3	007	RET

Keyboard Octal Loader -- By Gregg Weber, 1000 Plaza Dr. Apt 311-C State College, PA 16801

This is a copy of the loader program he has been using. It loads programs in octal from the keyboard, dumps programs in octal on the TVT, and executes programs in memory. He will soon add cassette load and dump routines.

Directions For Octal Loader

1. TVT connected to output port B (125)
2. Keyboard connected to Input port 0
3. Start with 005
4. Change interrupt instruction to 300 continue
5. Operating instructions.

Program should type *
Example- to load a program starting at 02 000
Type L.
Program should respond with H=
Type 002
Program should respond with L=
Type 000
Program should respond with 02 000
Type contents of 02 000
Program should respond with 02 001
When finished type F
Program should respond with *
To dump memory ,type D and then enter address as before.
Program will dump 60 locations in octal.
To execute a program, type E, and enter address as before.
An incorrect command will result in a ? being typed.

OCTAL LOADER

Label	Address	Code	Mne.
	00000	104	JMP START
	00001	313	
	00002	000	
KBD	00003	000	HET
	00004	101	INP0
	00005	310	LBA
OUT	00006	301	LAB
	00007	125	OUT 3
	00010	036	LDI
	00011	003	
	00012	046	LEI
	00013	163	
LOOP	00014	041	DCE
	00015	110	JFZ LOOP
	00016	014	
	00017	000	
	00020	031	DCD
	00021	110	JFZ LOOP
	00022	014	
	00023	000	
	00024	007	RTN
BINOCT	00025	066	LLI
	00026	122	
	00027	301	LAB
	00030	044	NDI
	00031	007	
	00032	004	ADI
	00033	060	
	00034	370	LMA
	00035	061	DCL
	00036	301	LAB
	00037	044	DDI
	00040	070	
	00041	012	RRC
	00042	012	RRC
	00043	012	RRC
	00044	004	ADI
	00045	060	
	00046	370	LMA
	00047	061	DCL
	00050	301	LAB
	00051	002	RLC
	00052	002	RLC
	00053	044	NDI
	00054	003	
	00055	004	ADI
	00056	060	
	00057	370	LMA
	00060	007	RTN

Gregg Weber's Keyboard Loader Continued

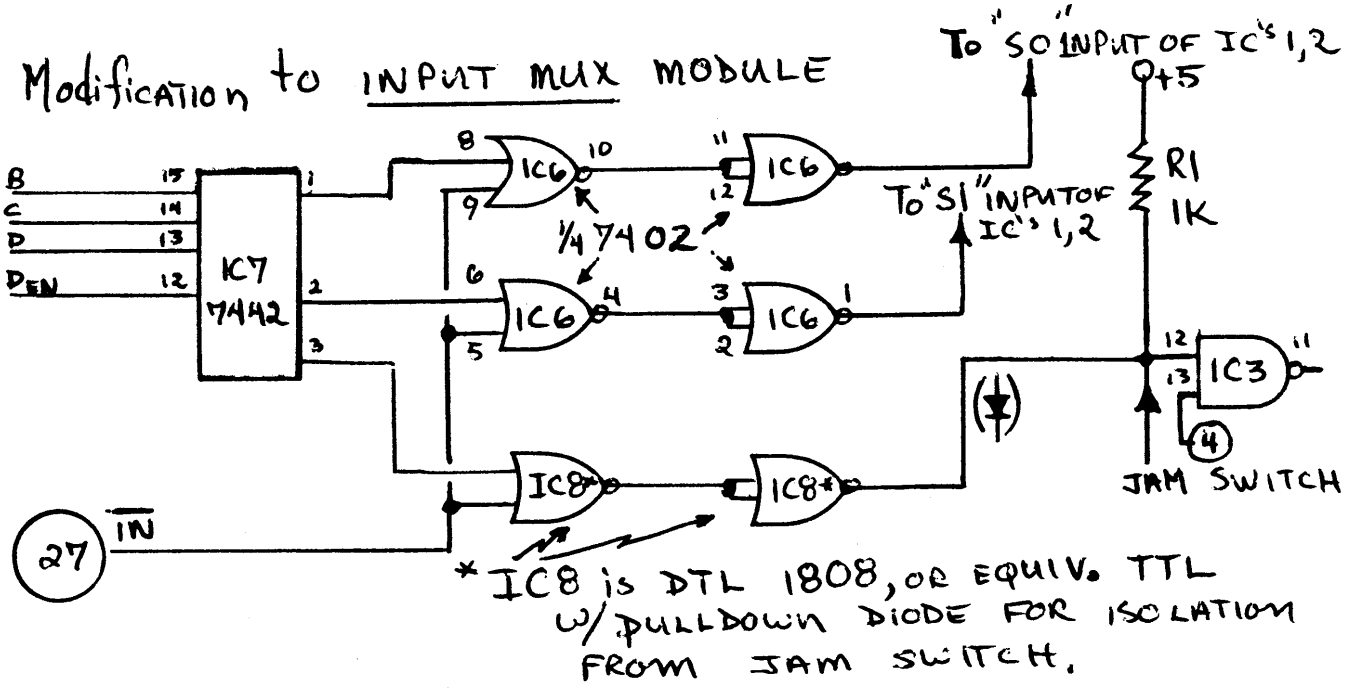
Label	Address	Code	Mne.
OCTBIN	00061	066	LLI
	00062	120	
	00063	307	LAM
	00064	024	SUI
	00065	060	
	00066	002	RLC
	00067	002	RLC
	00070	002	RLC
	00071	310	LBA
	00072	060	INL
	00073	307	LAM
	00074	024	SUI
	00075	060	
	00076	201	ADB
	00077	002	RLC
	00100	002	RLC
	00101	002	RLC
	00102	310	LBA
	00103	060	INL
	00104	307	LAM
	00105	024	SUI
	00106	060	
	00107	201	ADB
	00110	310	LBA
	00111	007	RTN
DATAIN	00112	026	LCI
	00113	375	
	00114	066	LLI
	00115	120	
IN	00116	106	CAL KBD
	00117	003	
	00120	000	
	00121	371	LMB
	00122	006	LAI
	00123	106	'F'
	00124	271	CPB
	00125	150	JTZ START
	00126	313	
	00127	000	
	00130	060	INL
	00131	020	INC
	00132	110	JFZ IN
	00133	116	
	00134	000	
	00135	106	CAL OCTBIN
	00136	061	
	00137	000	
	00140	007	RTN
ENTERA	00141	106	CAL CR

Label	Address	Code	Mne.
	00142	070	
	00143	001	
	00144	016	LBI
	00145	110	'H'
	00146	106	CAL OUT
	00147	005	
	00150	000	
	00151	016	LBI
	00152	075	'='
	00153	106	CAL OUT
	00154	006	
	00155	000	
	00156	106	CAL DATAIN
	00157	112	
	00160	000	
	00161	066	LLI
	00162	124	
	00163	371	LMB
	00164	016	LBI
	00165	114	'L'
	00166	106	CAL OUT
	00167	006	
	00170	000	
	00171	016	LBI
	00172	075	'='
	00173	106	CAL OUT
	00174	006	
	00175	000	
	00176	106	CAL DATAIN
	00177	112	
	00200	000	
	00201	066	LLI
	00202	123	
	00203	371	LMB
	00204	007	RTN
	00205		NOP
	00206		NOP
ADDOUT	00207	106	CAL CR
	00210	070	
	00211	001	
	00212	066	LLI
	00213	124	
	00214	317	LBM
	00215	106	CAL BINOCT
	00216	025	
	00217	000	
	00220	066	LLI
	00221	121	
	00222	317	LBM
	00223	106	CAL OUT
	00224	006	
	00225	000	
	00226	060	INL
	00227	317	LBM
	00230	106	CAL OUT
	00231	006	
	00232	000	
	00233	066	LLI
	00234	123	
	00235	317	LBM
	00236	106	CAL BINOCT
	00237	025	
	00240	000	
	00241	106	CAL DATOUT
	00242	267	
	00243	000	
	00244	106	CAL SP
	00245	076	
	00246	001	
	00247	007	RTN
	00250		NOP
	00251		NOP
	00252		NOP
	00253		NOP
	00254		NOP
	00255		NOP
WDPRIN	00256	106	CALL SETMA
	00257	002	
	00260	001	
	00261	317	LBM
	00262	056	LHI
	00263	001	
	00264	106	CAL BINOCT
	00265	025	
	00266	000	
DATOUT	00267	106	CAL SP
	00270	076	
	00271	001	
	00272	026	LCI
	00273	375	
	00274	066	LLI
	00275	120	
BOUT	00276	317	LBM
	00277	106	CAL OUT
	00300	006	
	00301	000	
	00302	060	INL
	00303	020	INC
	00304	110	JFZ BOUT
	00305	276	
	00306	000	
	00307	007	RTN
	00310		NOP
	00311		NOP
	00312		NOP
START	00313	106	CAL CR
	00314	070	
	00315	001	
	00316	016	LBI
	00317	052	'*'
	00320	106	CAL OUT
	00321	006	

Label	Address	Code	Mne.
	00322	000	
	00323	106	CAL KBD
	00324	003	
	00325	000	
	00326	056	LHI
	00327	001	
	00330	301	LAB
	00331	074	CPI
	00332	114	'L'
	00333	150	JTZ DATA
	00334	360	
	00335	000	
	00336	074	CPI
	00337	104	'D'
	00340	150	JTZ DUMP
	00341	024	
	00342	001	
	00343	074	CPI
	00344	105	'E'
	00345	150	JTZ EXEC
	00346	104	
	00347	001	
ERROR	00350	016	LBI
	00351	077	'?'
	00352	106	CAL OUT
	00353	006	
	00354	000	
	00355	104	JMP START
	00356	313	
	00357	000	
DATA	00360	106	CAL ENTERA
	00361	141	
	00362	000	
READIN	00363	106	CAL ADDOUT
	00364	207	
	00365	000	
	00366	106	CAL DATAIN
	00367	112	
	00370	000	
	00371	106	CAL SETMA
	00372	002	
	00373	001	
	00374	371	LMB
	00375	056	LHI
	00376	001	
	00377	104	JMP READIN
	01000	363	
	01001	000	
SETMA	01002	066	LLI
	01003	124	
	01004	337	LDM
	01005	061	DCL
	01006	347	LEM
	01007	324	LCE
	01010	020	INC
	01011	372	LMC
	01012	110	JFZ HIGH
	01013	021	
	01014	001	
	01015	323	LCD
	01016	020	INC
	01017	060	INL
	01020	372	LMC
HIGH	01021	353	LHD
	01022	364	LLE
	01023	007	RTN
DUMP	01024	106	CAL ENTERA
	01025	141	
	01026	000	
	01027	066	LLI
	01030	116	
	01031	076	LMI
	01032	012	
NEWLIN	01033	106	CAL ADDOUT
	01034	207	
	01035	000	
	01036	066	LLI
	01037	117	
	01040	076	LMI
	01041	006	
LINE	01042	106	CAL WDOUT
	01043	256	
	01044	000	
	01045	066	LLI
	01046	117	
	01047	327	LCM
	01050	021	DCC
	01051	372	LMC
	01052	110	JFZ LINE
	01053	042	
	01054	001	
	01055	066	LLI
	01056	116	
	01057	327	LCM
	01060	021	DCC
	01061	372	LMC
	01062	110	JFZ NEWLIN
	01063	033	
	01064	001	
	01065	104	JMP START
	01066	313	
	01067	000	
CR	01070	016	LBI
	01071	015	'CR'
	01072	106	CAL OUT
	01073	006	
	01074	000	
	01075	007	RTN
SP	01076	016	LBI
	01077	040	'Ø'
	01100	106	CAL OUT
	01101	006	
	01102	000	

Gregg Weber's Keyboard Loader
Continued

Label	Address	Code	Mne.
	01103	007	RTN
KEC	01104	106	CAL ENTERA
	01105	141	
	01106	000	
	01107	066	LLI
	01110	122	
	01111	076	LMI
	01112	104	
	01113	104	JMP DATA3
	01114	122	
	01115	003	
	01116		USED BY LOADER
	01117		USED BY LOADER
ATA1	01120		DATA1 USED BY LOADER
ATA2	01121		DATA2 USED BY LOADER
ATA3	01122		DATA3 USED BY LOADER
	01123		USED BY LOADER
	01124		USED BY LOADER



THIS MOD. CAUSES INPUT INSTRUCTION "105" TO INPUT DATA FROM THE INTERRUPT INSTRUCTION PORT. THE ADDITION OF IC 8 (AND POSSIBLY A DIODE) ARE THE ONLY MODIFICATIONS NEEDED. (TO THE MARK 8)

By Dave Bowles

TERRY RITTER'S EXECUTIVE-LOADER TTY DUMP SYSTEM
(Note: See Terry's writeup on this in NL #6)

By - Terry Ritter, 2524 Glen Springs Way, Austin, TX 78741
(512) 441-0036

Only One Restart Location Is Used

```
00 000 104   Restart - Jump To Executive
     1 000   Loader
     2 004
     3 106   Executive - Modified
     4 000   Instructions
     5 007
     6 300
     7 007
```

Notes:
Uses Keyboard-Strobe
No-Op Interrupt

Any input which is not a command is considered data; is masked and three bits shifted into display from rt.

```
04 000 250
     1 320
     2 121
     3 000
     4 101   (KEYBOARD PORT)
     5 054
     6 377
     7 310
    10 056
    11 004   OP TABLE LOCATION
    12 066
    13 045
    14 301
    15 277
    16 150
    17 075
    20 004
    21 060
    22 060
    23 306
    24 074
    25 074
    26 140
    27 014
    30 004   OP TABLE LOOKUP
    31 301
    32 044
    33 007
    34 310
    35 302
    36 200
    37 100
    40 200
    41 261
    42 104
    43 001
    44 004
    45 050   ASCII H = DISPLAY H (On Port of Display)
    46 305
    47 054   ASCII L = DISPLAY L
```

Op Match - Found

Op Match - Found

```
04 050 306
    51 055   ASCII M = DISPLAY M
    52 307
    53 030   ASCII 8 = LOAD H FROM DISPLAY
    54 352
    55 031   ASCII 9 = LOAD L FROM DISPLAY
    56 362
    57 016   ASCII - = LOAD M "
    60 372
    61 010   ASCII Space—Examine, Increment H,L & Display new M
    62 164
    63 007   ASCII Backspace = Examine Backwards
    64 170
    65 222   PRINT BUTTON = Jump To TTY Dump (Beg At H,L)
    66 200
    67 036
    70 036   ASCII ^ = Call Sub (at H,L)
    71 015
    72 175   Non Implemented Commands
    73 013
    74 000
    75 060
    76 317
    77 306
   100 056
     1 000
     2 066
     3 004
     4 074
     5 061
     6 100
     7 124   Non Single Byte
    10 004
    11 076
    12 007
    13 060
    14 076
    15 000
    16 060
    17 371
```

Single Byte Instr.

```
04 120 302   Single Byte Instr.
   21 104
   22 146
   23 004
   24 074
   25 070
   26 150
   27 136
   30 004
   31 006
   32 004   Rom-Subr. Instr.
   33 104
   34 140
   35 004
   36 303   Any Sub Instr.
   37 314
   40 371
   41 060
   42 370
   43 060   Prepare (Modify) RAM
   44 076
   45 307
   46 353
   47 364
   50 106
   51 003 ⇒ To RAM Subr.
   52 000
   53 335
   54 346
   55 104
   56 002   Start Again
   57 004
   60 104
   61 000
   62 000
   63 007
   64 060
   65 013   Incr. H,L. Subr.
   66 050
   67 007
   70 060
   71 061
   72 110   Decr. H,L. Subr.
   73 176
   74 004
   75 051
   76 061
  177 007   End of Exec.-Loader System
  200 006   Beginning of ROM TTY Dump system
  201 013 — Baudot CR
  202 106
  203 350 ⇒ TTY Output
  204 004
  205 006
  206 157 — Baudot Fig's.
  207 106

04 210 350 ⇒ TTY Output
   211 004
   212 006
   213 043 — Baudot LF
   214 106
   215 350 ⇒ TTY Output
   216 004
   217 313
   220 106
   221 276 ⇒ TTY 2-Digit Translation & Output
   222 004
   223 314
   224 106
   225 260 ⇒ TTY 3-Digit Translation & Output
   226 004
   227 006
   230 023 — Baudot Space
   231 106
   232 350 ⇒ TTY Output
   233 004
   234 353
   235 364
   236 317
   237 300   Reserved For Future Interpreter
   240 300
   241 300
   242 106
   243 260 ⇒ TTY 3-Digit Translation & Output
   244 004
   245 106
   246 254 ⇒
   247 004
   250 110
   251 200   Start Again
   252 004
   253 007 —
   254 040
   255 013   Incr. D, E
   256 030
   257 007
   260 056   Byte → Baudot Translation
   261 004
   262 301
   263 044
   264 300   Mask, Shift, Pick Up Baudot
   265 002
   266 002
   267 004
   270 335
   271 360
   272 307
   273 106
   274 350 ⇒ TTY Output
   275 004
   276 056
   277 004
```

```
04 300 301
   301 044
   302 070
   303 012
   304 012
   305 012
   306 004
   307 335
   310 360
   311 307
   312 106
   313 350
   314 004
   315 301
   316 044
   317 007
   320 004
   321 335
   322 360
   323 307
   324 106
   325 350
   326 004
   327 006
   330 023
   331 106
   332 350
   333 004
   334 017
   335 067
   336 167
   337 147
   340 103
   341 053
   342 007
   343 127
   344 163
   345 063
   346 016
   347 037
   350 121
   351 131
   352 026
   353 330
   354 300
   355 300
   356 300
   357 021
   360 110
   361 355
   362 004
   363 200
   364 110
   365 350
   366 004
   367 007
```

Mask Shift Pick up Baudot

TTY Output

mask pick up Baudot

TTY Output

Baudot Space

TTY Output

Baudot Octal Table

Bit Timing

Serial Output Loop

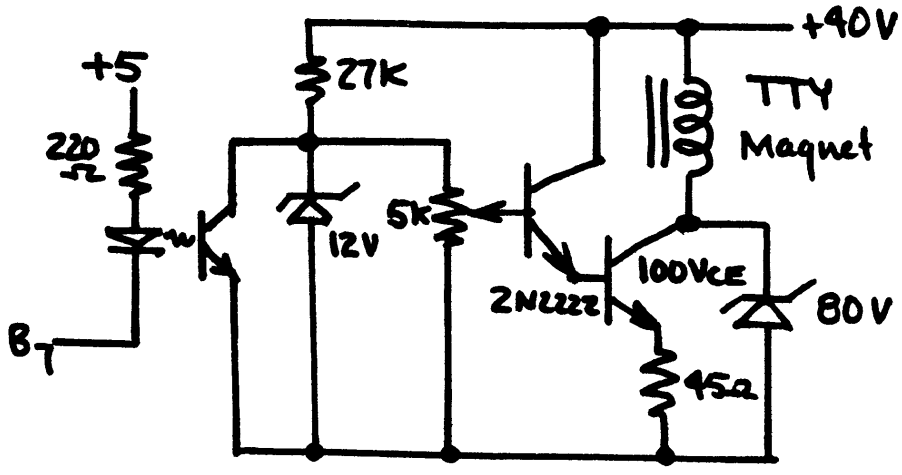
```
04 370 104
   371 000
   372 000
   373 000
   374 100
   375 000
   376 000
   377 000
```

Output To TTY B7 of Port 4
(Also on port 0 for effect)

Extra delay step included for faster clocked 8008's.

Isolated Magnet-Driver

(constant-current design insures fast attack and adjustable current; zener protects transistor and insures fast decay. Inductance in circuit requires high voltage for performance comparable to 120 volt systems)



Terry Ritter, 2524B Glen Springs Way, Austin, TX 78741 (512) 441-0036
suggests the following software standard:

On another topic, I propose some software possibilities which might become standards: It may not be all that apparent, at first, but an 8008 machine-language program will run only when the program is located in the memory locations for which it was written. There is no particular problem in loading the program anywhere desired; the problem is changing the jump addresses for the program to correspond to the new location.

I propose that the present or original location of each program be identified by a jump as the program's first step (see figure 1.). Additional space just after the jump address can be used as program ID bytes (see figure 2.), thus providing a convenient basis for program labeling and starting-address table-building programs. Note that the unconditional jump utilizes one of the 8008 "don't care" codes which can be easily identified by the relocatable loader as a "start of new program." Similarly, the last step of each program should be a special RETURN code which provides indication of the last step of program(s) in which all proceeding steps to the start of the program are to be located sequentially together.

Note also that a RETURN last step is standardized, rather than a HALT. The RETURN allows calling each program from a keyboard-controlled monitor program, then returning control to the monitor when the called program is complete.

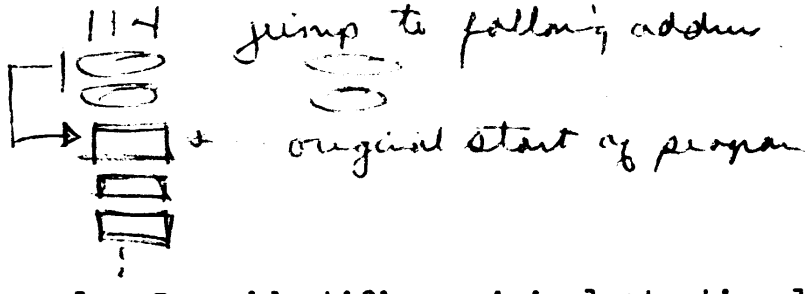


Figure 1. Jump identifies original starting location.

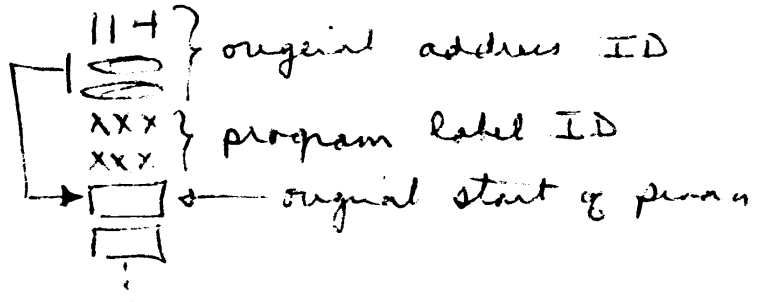


Figure 2. Program labels included.

Cordially

Terry F. Ritter
Vice President

PROPOSAL FOR UNIFYING INPUT/OUTPUT METHODS FOR 8008 AND 8080 USERS

By Brion Johnson
11 April 1975

1. There has been much discussion lately concerning a standard that would enable the ready exchange of programs between different systems. This would be especially good because the different essential programs on a system are much the same in nature but a lot of work to generate if a person starts from scratch. This proposal is aimed at those using INTEL 8008 or 8080 based systems, such as the MARK-8, MIL MOD-8, and ALTAIR 8800 to name a few, with a goal of making programs standardized to this proposal runable on any 8008 or 8080 system equipped with software as stated in this proposal.

2. A previous proposal has suggested that this standardization take place at a hardware level, by making specific I/O port allocations when the architecture is firmed up. This proposal does not use that approach because invariable the I/O configurations of every person's system is bound to be different due to personal choice, or pocketbook, or any number of other reasons, and also because the I/O structure of the 8008 differs radically from the I/O structure of the 8080.

3. The proposal is that program input/output be handled on a system level, which is the usual way that I/O is handled on most computers. Two elements are required, an IOCS (Input Output Control System - The term refers usually to a specific program that takes care of I/O responsibilities) and an exact format specification for calling on the IOCS to inform it of its task (here handled as a subroutine call).

4. The IOCS must (sorry) be written to fit the machine on which it will run. There is a bright side, though, as it may be the bare essentials and only take up about 40 to 50 locations or perhaps one may want to expand it to include such features as buffering, multi-level priority interrupt, usage with a comprehensive operating system supervisor, etc. Given time and sufficient interest, a typical small IOCS (bare bones) might be developed and published in this NL. This one might take and adapt as necessary to fit one's own system.

5. The IOCS proposed here works as follows; The program desiring an I/O service must tell the IOCS (A) The location or starting address of the data, (B) The number of 8 bit words to be transferred, (C) The format that the program looks at (the current format - binary, decimal, ASCII, etc - for outgoing data, or the format that incoming data is to be placed in), and (D) The device at which the operation is to be performed. When the IOCS receives this information, it proceeds with all of the housekeeping (is the device servicable, or is it busy, etc.) and gets everything set up to perform the I/O. Then it might go ahead and do the operation or if it is sophisticated enough, set up the interrupt system and return to the main program, bouncing back to send or receive the next word of data when the device signals that it is ready by causing an interrupt. The beauty of this is that both approaches "look" the same to the program, and hence the program may easily run on many different systems.

6. The format for calling the IOCS is as follows:

6.1 The A, B, C, and D registers contain data as follows:

6.1.1 The A register contains the lowest address of the block of data to be transferred.

6.1.2 The B register contains the number of 8 bit words to be transferred.

6.1.3 The C register contains the data format as follows:

D	D	D	D	D	D	D	D
7	6	5	4	3	2	1	Ø

D(7) is a 1 for output: A Ø for input.
D(6) thru D(4) are not used
D(3) is a 1 for ASCII: A Ø for other.
D(2) is a 1 for 5 level Baudot code: A Ø for other
D(1) is a 1 for Binary-Coded decimal (BCD), at two BCD digits per 8 bit word: a Ø for other.
D(Ø) is a 1 for binary data (8 binary bits): a Ø for other.

6.1.4 The D register contains a device code number. Note that this number does not specify input or output, and note also that this number may be readily changed to fit hardware by the IOCS.

6.1.5 Standard Device Code Allocations Are:

Code	Allocation
ØØØ	System Communications Device (TTY or operator I/O)
ØØ1	System Output Device (perhaps line printer)
ØØ2 thru ØØ7	System input/output
Ø1Ø thru 377	Not Allocated

6.2 Program transfer to the IOCS is made by addressing a subroutine jump to location ØØØ Ø7Ø such as CAL Ø7Ø ØØØ or RST 7.

6.3 Memory locations ØØØ Ø7Ø thru ØØØ Ø77 must not be used by the program (these contain the transfer instructions to the IOCS) and also memory locations 000 000 thru 000 007 must not be used unless for restarting and then must be considered volatile (as the IOCS may use it for interrupt in some cases of implementation). The IOCS is assumed to be in upper memory.

7. Programs written to this specification are assured of running on any system which has an IOCS written to this specification provided, of course, that there is room for both the IOCS and the program in memory... The program should occupy the lower end of memory with the exceptions as mentioned in 6.3. I hope that this may provide a workable solution to this very important problem.

8. One word concerning 8008 and 8080 differences... There will be no difficulty if 8080 users will limit themselves to only using the 8008 instruction set for programs to be exchanged among users. For those who would be tempted not to (and I don't blame them), I would like to see a program compatible with this proposal written in 8008 code that would take 8080 code and translate it to the equivalent 8008 listing at the assembly level..

REGISTER ASSIGNMENTS FOR MCS-8 SUBROUTINES

By T. W. Fuller, Santa Barbara

20 April 1975

As a follow-up on the IOCS register assignments proposed by Brion Johnson, the following discussion recommends changes to that proposal, as well as assignments for general subroutine transfers. This will make coding more compact, and easier to implement and understand.

There are basically two ways in which a subroutine may be called with the 8008 instruction set. These are: 1) Using the address stack in a CALL or RST instruction, and 2) jumping to the entry address of the subroutine with the return address contained in two of the scratch pad registers. Obviously, if there is no more stack space, using the second method would be preferable to wiping out the bottom address in the stack by using the first.

In either type of calling method, it may be desirable to save the contents of the registers before executing the meat of the subroutine, so they may be restored before returning to the calling program. The only way these will be store is by using registers L and H. Therefore they cannot be assigned a definite function.

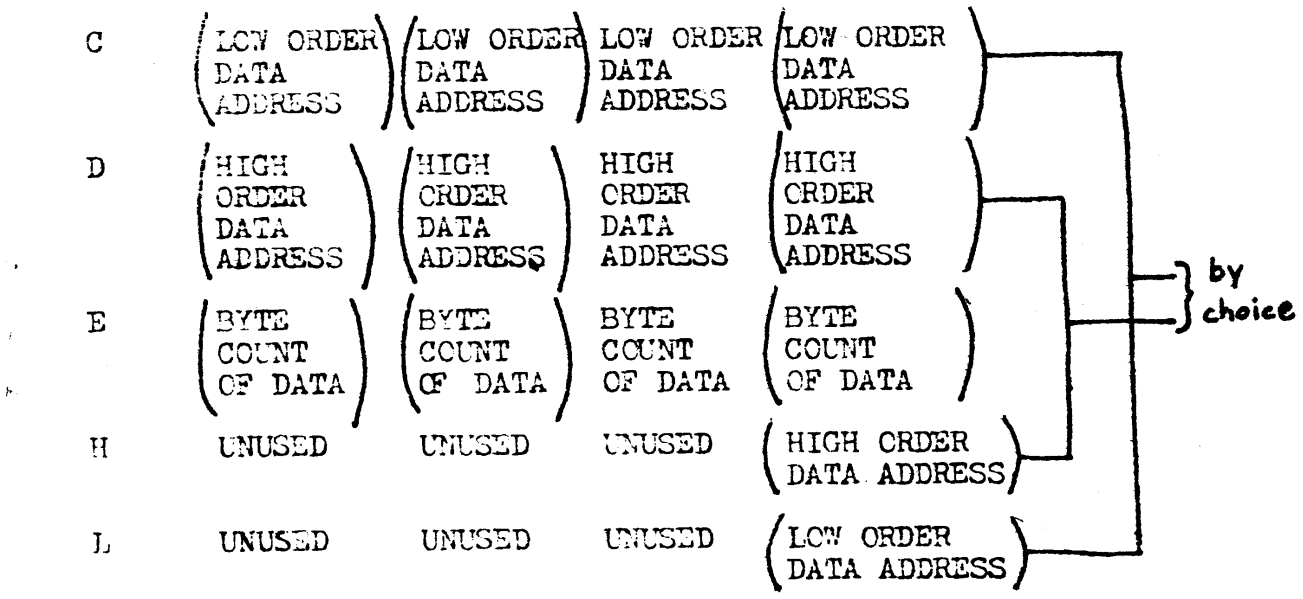
When the second, or jump entry method is used, two registers must contain the return address. It would be most convenient if they could be store directly in the JMP instruction which returns control to the main program. Also, when storing registers for later restoration, it is most convenient to store them in sequence in memory. Therefore, if registers A and B are stored in the second and third bytes of the returning JMP instruction, registers C, D, and E may follow them directly, saving 2 bytes of memory.

The only remaining problem is the manipulation of data, which appears to be well served by placing the starting address and byte count in the remaining 3 registers, as per Brion's proposal. No further information is basically necessary, and the subroutine would only be complicated by requiring more pointers. These registers may be used for any purpose if data is not to be transfered in quantities greater than 3 bytes.

When the CALL method is used, registers A and B are not required to indicate the return address and may be used as necessary to the subroutine.

With these assignments in mind, a slight shuffling of the proposed IOCS parameters is in order so that the assignments may be consistent, and they appear in the table below. Register assignments which are optional are enclosed in parentheses. Returned values may var, depending on how immediately their values will be used.

REGISTER	CALL METHOD	JUMP METHOD	CALL IOCS	RETURNED VALUES
A	(FUNCTION OF SUBROUTINE)	LOW ORDER RETURN ADDRESS	FORMAT OF TRANSFER	(FUNCTION OF SUBROUTINE)
B	(FUNCTION OF SUBROUTINE)	HIGH ORDER RETURN ADDRESS	DEVICE CODE	(FUNCTION OF SUBROUTINE)



The resulting house keeping code for the jump method might appear as follows:

```
SUBRTN:  MVI  L,RETURN+1(L)   'Set up to store registers'
         MVI  H,RETURN+1(H)
         MOV  M,A             'Store return address'
         INR  L
         MOV  M,B
         INR  L
         MOV  M,C             'Store remaining registers'
         INR  L
         MOV  M,D
         INR  L
         MOV  M,E
         .
         .
         (SUBROUTINE)
         .
         .
         MVI  L,RETURN+3(L)   'Set up to restore registers'
         MVI  H,RETURN+3(H)
         MOV  C,M             'Restore registers'
         INR  L
         MOV  D,M
         INR  L
         MOV  E,M
RETURN:  JMP  0               'Return'
         $B,4                 'Set aside block of 4 bytes'
                              'for registers B through E'
```

MODIFIC TION TO PROPOSED SOFTWARE I/O STANDARDS AS SUBMITTED
By Brion Johnson
By T. W. Fuller 13 April 1975

In Brion's proposal, the low order 4 bits of register D indicate the method used in encoding the data to be transferred. Each bit is assigned a different method, thus limiting the number of codes to 4.

One of the advantages of using such as system is the ease with which it can be implemented in a "bare bones" I/O control system (BBIOCS).

There are some facts to consider, however. There are a wide variety of I/O devices available for use in our "community", and not all of them use the same codes or characters sets. There are modifications in the number of special characters, and in whether or not shifting must be a separate operation for teletypes.

It would be advantageous to adopt an assignment in which the four low order bits are considered a single number indicating the coding of the data. In this way it would be possible to assign numbers 0 thru 3 to the proposed four codes, since they are common, and this would be easily incorporated into a BBIOCS. Any other coding or modifications to these four could be indicated by a larger number.

If it is desired to recognize modifications to the original four codes, it would be possible to assign them values whose lower two bits indicate the basic coding scheme, and the upper two bits would indicate the version of the modification. In this way a BBIOCS would not have to be concerned with the modifications, but just look at the lower two bits to determine the coding scheme.

There may still be room for different coding schemes among those values yet unassigned, although this might present a problem for the BBIOCS. It may be better to use of the unassigned bits (bits 4-6) as a flag indicating an odd-ball code.

An example of a BBIOCS to process register D, and select the proper I/O handler might be the following, assuming that registers A, B, C have ben used or saved, and that this routine and all handlers reside in page 0 of memory:

Addr.	Contents	Label	Instruction		Comments
000	303		MOV	A,D	Load A with spec.
001	044		ANI	203	Mask I/O bit and
002	203				code spec, clr C
003	022		RAL		Set C=I/O bit
004	100		JNC	JUST	If C=0 skip to
005	011				JUST
006	000				
007	004		ADI	010	Add 2*length of
010	010				Input Handler table
011	012	JUST:	RRC		Justify and block C
012	004		ADI	TABLE(L)	Add address of
013	026				Handler table
014	056		MVI	H,000	Clear high order
015	000				address
016	360		MOV	L,A	Set L to get handler add.
017	307		MOV	A,M	Load A with handler add.
020	066		MVI	L,JUMP+1	Set L to mod. JMP instr.
021	024				
022	370		MOV	M,A	Modify JMP instr.

```
023     104     JUMP:   JMP  000000    Jump to handler
024     000
025     000
026     ---     TABLE:  (Handler 1) Input Handlers
027     ---             (Handler 2)
030     ---             (Handler 3)
031     ---             (Handler 4)
032     ---             (Handler 5) Output Handlers
033     ---             (Handler 6)
034     ---             (Handler 7)
035     ---             (Handler 8)
```

Ted Lincoln 410 Bell Ave. Santa Ana, Cal. 92707. Finaly got the Mark -8 and TVT working in Dec. TVT modified for cursor control, erase and home controled from keyboard. It also has simplex duplex modes. Mini uses ttl memory instead of 1101 and contains a relay control and a sense board. (almost working) I have a DAC and mag tape modem in process. Delays in geting these on line are not technical but lack of time. (I am an engineer designing computer interfaces for a simulation lab.)

I'd like to make some suggestions.

1) Two modems will be better than one. Dr. Suding's design would be useful for telephone as well as ham. Another design should be adopted for use for high density recording.

2) When adding connecters to the Mark 8 use at least 100 to 120 pins. My choice is 120. Two connectors of 60 each. This allows bussing all important lines plus about thirty spares. Board location is then easily changed. Use I.C. sockets and headers for I/O signals. One sixteen pin socket for two inputs or outputs. Cost less than a dollar even from slip joints.

3) Standardizing data format on mag tape is not important. As long as the person giving you the tape explains the format used you can read into memory reformat and rerecord with your own system.

4) I've got a RTC designed and as soon as I am able to get enough time to get it working I will send along the info.

5) An idea to your readers. When mounting wirewrap IC sockets to vector boards. Drill out the holes of two opposite corners and mount PC board eyelets. Eyelets can be purchased from a local PC board manufacturer, Newark Electronics, etc. Most require a #51 drill. For example drill out holes on the board for pins #1 and #8. Insert eyelet. By holding the eyelet in the hole with a center punch and pushing on the other side of the eyelet with an automatic center punch the eyelet will peen to the board. Insert socket and solder base of pin to eyelet. Socket is now solid on board making it easier to wire.

Ted suggests the "Bugbooks" for those readers asking about books on basics. These start with "this is a diode" and end up with micro processor use and applications. Bugbook I and II, and laboratory workbooks are available for $18.95, and instructors manual for $3.50. The books are centered around EL instrument circuit designer sockets and "outboards", functional modules that plug into the sockets. Contact Edwards Associates, Calif. Federal Bldg., Suite 320, 608 Silver Spur Road, Palos Verdes Peninsula, CA 90274 (213) 377-0975. One of the neatest things available is a set of "bugbacks", a set of printed labels gummed on the back that illustrate IC pin connections. For debugging a tough circuit, these would be fabulous. $4.95 for a set of 500 individualized labels.

A TABLE OF DOUBLE PRECISION WORDS FOR AN 8 BIT MACHINE

If we wish to use numbers higher than 255 decimal in computer calculations using 8 bit words we find that we must concatenate two or more words which is called double or higher precision. Suppose we have the decimal number 2783, we would find that the binary equivalent is;

1 0 1 0 1 1 0 1 1 1 1 1 which in octal is 5337.

Now if we split this into two words of 8 bits each it would be

0 0 0 0 1 0 1 0 and 1 1 0 1 1 1 1 1

Note that the bit arrangement is the same but the octal is now 012 and 337. The first number is called the high order and the second number is called the low order of the double precision number.

Tables of decimal to octal numbers are found in many computer texts and following is the table of octal to double precision numbers.

00XX = 000 0XX	30XX = 006 0XX	60XX = 014 0XX
01XX = 000 1XX	31XX = 006 1XX	61XX = 014 1XX
02XX = 000 2XX	32XX = 006 2XX	62XX = 014 2XX
03XX = 000 3XX	33XX = 006 3XX	63XX = 014 3XX
04XX = 001 0XX	34XX = 007 0XX	64XX = 015 0XX
05XX = 001 1XX	35XX = 007 1XX	65XX = 015 1XX
06XX = 001 2XX	36XX = 007 2XX	66XX = 015 2XX
07XX = 001 3XX	37XX = 007 3XX	67XX = 015 3XX
10XX = 002 0XX	40XX = 010 0XX	70XX = 016 0XX
11XX = 002 1XX	41XX = 010 1XX	71XX = 016 1XX
12XX = 002 2XX	42XX = 010 2XX	72XX = 016 2XX
13XX = 002 3XX	43XX = 010 3XX	73XX = 016 3XX
14XX = 003 0XX	44XX = 011 0XX	74XX = 017 0XX
15XX = 003 1XX	45XX = 011 1XX	75XX = 017 1XX
16XX = 003. 2XX	46XX = 011 2XX	76XX = 017 2XX
17XX = 003 3XX	47XX = 011 3XX	77XX = 017 3XX
20XX = 004 0XX	50XX = 012 0XX	
21XX = 004 1XX	51XX = 012 1XX	* Example shown above.
22XX = 004 2XX	52XX = 012 2XX	
23XX = 004 3XX	53XX = 012 3XX *	
24XX = 005 0XX	54XX = 013 0XX	
25XX = 005 1XX	55XX = 013 1XX	
26XX = 005 2XX	56XX = 013 2XX	
27XX = 005 3XX	57XX = 013 3XX	

George L. Haller, Apr. 1975
Summer Address
Hound Ears Club
Blowing Rock, NC 28605

By Walter M. White, 343 S. Madison Ave., 18, Pasedena, CA 91101

MEMORY TEST PROGRAM FOR MARK-8

Purpose of the program is to thoroughly check the semiconductor memory used with the 8008 for ICs that will not accept data correctly or have problems in their internal address decoding circuitry.

The program starts at location 000 and halts at location 101 upon completion. Pseudo random bit patterns are written into the entire memory (except that portion where the program resides) and then are read back and checked against the pattern written. This process is repeated with 207 different starting points in the random bit pattern. Execution time is approximately 45 minutes for 16K of memory.

No peripheral devices are required, only the means of jamming a NO-OP instruction (LAA = 300_8) into the interrupt instruction port is necessary.

Upon detection of a read error, the program will halt. The LED register connected to output port Ø will indicated which bit of the 8 bit byte was incorrectly read, thus establishing which column the defective memory IC is in. Jamming a NO-OP into the interrupt instuction port will cause the program to halt with the high order address bits of the defective IC. The two displays enables the user to pinpoint the defective IC. The lowest order 8 bits of the address are not displayed as it is immaterial which bit inside the IC is defective, the entire IC must be replaced. Jamming a second NO-OP will cause the program to continue until completion or until another read error is detected. Note: if the program halts with all eight LEDs lit, it indicates a short on the address lines or a wiring error in the memory address circuitry.

LOCATION	INSTRUCTION		COMMENT
This section clears display and initializes registers			
000	046	LEI	Load register E with initial "random number"
001	001	001	
002	026	LCI	Load highest 6 bits of last address in
003	xxx	xxx	memory plus 1 (100_8 for 16K, 004_8 for 1K,
004	056	LHI	008_8 for 2K, etc.)
005	000	000	
006	066	LLI	Load starting memory test address
007	120	120	
010	250	XRA	Clear LED display
011	121	OUT	
This section writes random pattern into memory			
012	334	LDE	Store starting random number for each pass
013	106	CAL	Call random number
014	105	105	
015	000	000	
016	230	SBA	Set all bits of accum. equal to carry bit
017	370	LMA	Store
020	060	INL	Increment memory address location and
021	110	JFZ	continue writing
022	013	013	
023	000	000	
024	050	INH	
025	305	LAH	
026	272	CPC	Continue writing until maximum memory
027	110	JFZ	address is reached
030	013	013	
031	000	000	

This section reads data and checks it against pattern written

```
032     066  LLI     Load starting memory address
033     120  120
034     056  LHI
035     000  000
036     343  LED     Recover starting random number used for
037     106  CAL       last memory write pass
040     105  105     Call random number
041     000  000
042     230  SBA     Set all bits of accum. equal to carry bit
043     257  XRM     Compare with memory data
044     150  JTZ     Jump around error routine if data compares OK
045     056  056
046     000  000
047     121  OUT     Display bits in error
050     001  HLT     User jams a NO-OP to display address of bad IC
051     305  LAH     Display high order bits of memory that failed
052     121  OUT
053     001  HLT     User jams a NO-OP to continue with program
054     250  XRA     Clear display
055     121  OUT
056     060  INL     Increment memory address location
057     110  JFZ
060     037  037
061     000  000
062     050  INH
063     305  LAH
064     272  CPC     Continue reading and comparing until maximum
065     110  JFZ       memory address is reached
066     037  037
067     000  000
```

This section initializes the random number subroutine with a different number for the next write pass through memory.

```
070     343  LED     Recover starting random number used for
071     106  CAL       last memory pass
072     105  105     Call random number
073     000  000
074     074  CPI     Check if all random bit patters have been used
075     001  001
076     110  JFZ     Jump to memory write routine
077     004  004
100     000  000
101     001  HLT     Program stops here when complete
102     104  JMP
103     101  101
104     000  000
```

This section is a pseudo random number generating subroutine. It generates 207 of the possible 256 combination of 8 bits and can be used as the basis for a number of computer games. The main program above uses only the bit that is shifted into the carry position, not the actual random number generated.

```
105     304  LAE     Load accum. with previous random number
106     032  RAR     Rotate 3 bit positions
107     032  RAR
110     032  RAR
111     254  XRE     Exclusive OR with previous random number
112     032  RAR     Rotate new bit into carry
113     304  LAE     Load accum with previous random number
114     032  RAR     Rotate carry into A7 creating new random number
115     340  LEA     Save number in register E
116     007  RET
```

CROSS-LISTING
8008/8080 INSTRUCTION SETS

Mnemonic	8008	8080
Lr_1r_2	3XX	1XX
LrM	3X7	1X6
LMr	37X	16X
LrI	0X6	0X6
	XXX	XXX
LMI	076	066
	XXX	XXX
INr	0X0	0X4
DCr	0X1	0X5
ADr	20X	20X
ADM	207	206
ADI	004	306
	XXX	XXX
ACr	21X	21X
ACM	217	216
ACI	014	316
	XXX	XXX
SUr	22X	22X
SUM	227	226
SUI	024	326
	XXX	XXX
SBr	23X	23X
SBM	237	236
SBI	034	336
	XXX	XXX
NDr	24X	24X
NDM	247	246
NDI	044	366
	XXX	XXX
XRr	25X	25X
XRM	257	256
XRI	054	356
	XXX	XXX
ORr	26X	26X
ORM	267	266
ORI	064	366
	XXX	XXX
CPr	27X	27X
CPM	277	276
CPI	074	376
	XXX	XXX
RLC	002	007
RRC	012	017
RAL	022	027
RAR	032	037
JMP	1X4 *	303 *
JFC	100 *	322 *
JFZ	110 *	302 *

Mnemonic	8008	8080
JFN	120 *	362 *
JFEP	130 *	342 *
JTC	140 *	332 *
JTZ	150 *	312 *
JTN	160 *	372 *
JTEP	170 *	352 *
CAL	1X6 *	315 *
CFC	102 *	324 *
CFZ	112 *	304 *
CFN	122 *	364 *
CFEP	132 *	344 *
CTC	142 *	334 *
CTZ	152 *	314 *
CTN	162 *	374 *
CTEP	172 *	354 *
RET	0X7	311
RFC	003	320
RFZ	013	300
RFN	023	360
RFEP	033	340
RTC	043	330
RTZ	053	310
RTN	063	370
RTEP	073	350
RST	0X5	3X7
INP	1XX	333
		XXX
OUT	1XX	323
		XXX
HLT	00X	166
	or 377	

REGISTERS

REG.	8008	8080
A	000	111
B	001	000
C	010	001
D	011	010
E	100	011
H	101	100
L	110	101
M	111	110
	(spec. by H & L)	

*Bytes 2 & 3 not shown

By Roger L. Smith
4502 E. Nancy Lane
Phoenix, AZ 85040

SYSTEM/MINAC II Overview

By - Brion Johnson, 1423 Alta Vista Road, Santa Barbara, CA 93103

(1) General Description

SYSTEM/MINAC II is a general-purpose medium speed small scale digital computer processing system built around the Intel 8008-1 microprocessing chip. Architecture is buss-oriented, with a "wired or" cpu input buss shared by memory, interrupt, and input; and with a latched address and direct data cpu output buss. Physically, the processor is housed in a seven inch by 19 inch rack mount card cage with 12 card slots. Card slots 2 and 4 are reserved for the CPU board which contains the 8008-1 chip, but otherwise, the slots are bussed and may contain any card. The minimum operating configuration consists of the memory card (93410 - 256 locations) and the cpu card, used in conjunction with the front panel.

(2) Memory

Memory for the system consists of 8192 locations of 8 bits each (1 byte) organized in halfwords (2 bytes) with a write protect bit and program accessible flag bit associated with each halfword. The write protect bit prevents accidental erasure of information while the flag bit may be used for parity, word mark, EOF, etc. Data access time is 2.5 micro-sec, and full cycle time is 6 micro-sec. The storage medium is magnetic ferrite cores which retain information when power is removed. The unit is a model 28RVQ4096 made by Ampex Computer Products Corp.

(3) System Teletype

The teletype (used for system commands or as needed) is a Teletype Model 14 typing reperforator. The TTY uses a 5-level Baudot Code. Typing is don on a 3/8" wide gummed paper tape.

(4) System Teleprinter

Page Output is provided by a CW/895/UG teleprinter mfg. by Mite Corp. The teleprinter uses 5 level baudot code, and can type 76 characters per line at 40, 60, or 100 wpm.

(5) High Speed Tape Punch

High speed paper tape output is provided by a Teletype Corp. Model BRPE2 punch. Capacity is seven level at 1100 wpm.

(6) Paper Tape Reader

Two medium speed paper tape readers are available to read 8 level paper tape. These units are mechanical, manufactured by the Soroban Corporation.

(7) Magnetic Cassette

A cassette interface is proposed using FSK encoding upon unmodified audio consumer cassette tape recorders.

(8) Multiport

Bitwise input/output is available over multiport, originally planned for 8 each 8 bit words out and 4 each 8 bit words in.

Pages From Bob Cook's CREED Baudot TTY Manual.

4. The Character Set

4.1 Methods of Changing and Expanding

The current character set is described in the table in 4.2 and the typehead layout in 4.3. You will probably desire to replace some of the fractions with more useful characters. A proposed set of characters are shown in the table in 4.2 and a procedure for replacing characters is included in a later section. In order to have upper case H and S print it will be necessary to remove two springs to disable their function bars as is also described later.

The character set chosen is a compromise between several conflicting considerations, however, it is very close to the standard Baudot communications set. It is possible to create some additional characters by combining characters as follows: (!)=(')+(.), (+)=(-)+(|), (*)=(-)+(|)+(X), and ($)=(S)+(|). Since there is no backspace facility, the best way to achieve this result is to separately control the non-print and non-feed solenoids which are presently connected in parallel. This would require an additional drive circuit and rewiring the solenoids inside the machine in order to get access to them separately. Then it would be possible to turn on the non-feed solenoid, print multiple characters and turn off the non-feed solenoid just before the last of the multiple characters.

4.2 Code Table

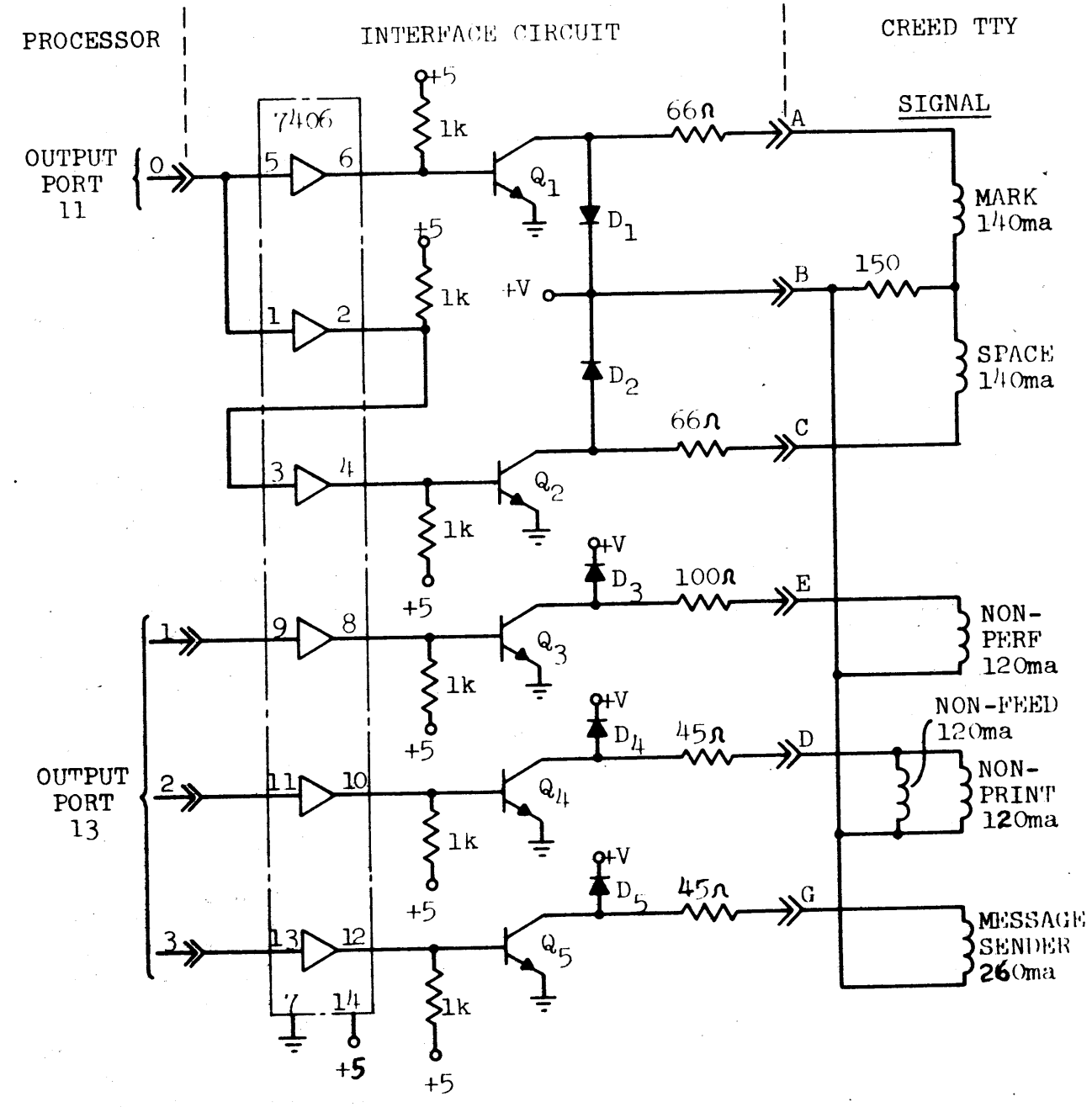
LTRS SHIFT	FIGS SHIFT	PROPOSED FIGS SHIFT	BINARY	OCTAL	LTRS SHIFT	FIGS SHIFT	PROPOSED FIGS SHIFT	BINARY	OCTAL
A	-	-	00011	03	Q	1	1	10111	27
B	5/8	?	11001	31	R	4	4	01010	12
C	1/8	:	01110	16	S		@	00101	05
D	1/3	=	01001	11	T	5	5	10000	20
E	3	3	00001	01	U	7	7	00111	07
F	2/3	&	01101	15	V	3/8	;	11110	36
G	'	'	11010	32	W	2	2	10011	23
H		%	10100	24	X	/	/	11101	35
I	8	8	00110	06	Y	6	6	10101	25
J	1/4	.	01011	13	Z	"	"	10001	21
K	1/2	(	01111	17	CR	CR	CR	01000	10
L	3/4	)	10010	22	SPACE	SPACE	SPACE	00100	04
M	#	#	11100	34	FIGS	FIGS	FIGS	11011	33
N	7/8	,	01100	14	LTRS	LTRS	LTRS	11111	37
O	9	9	11000	30	BLANK	BLANK	BLANK	00000	00
P	0	0	10110	26	LF	LF	LF	00010	02

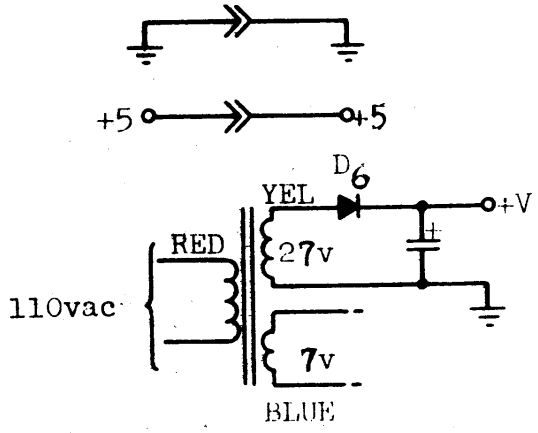
4.3 Typehead Layout

The diagram on page 4 is the original typehead layout specification for the Speigel Project. It will be very useful when you are changing typefaces on the typehead.

5. Machine description

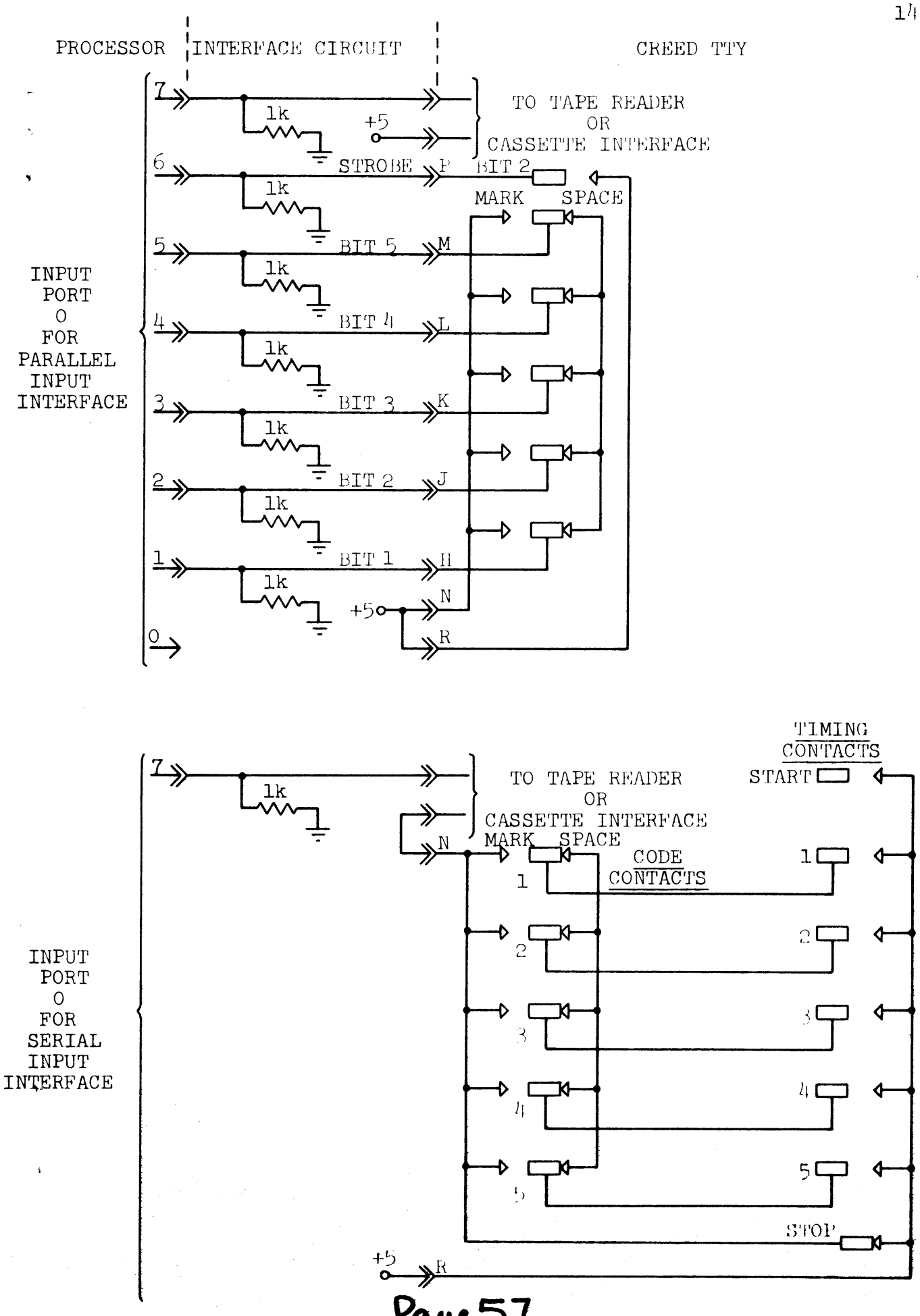
The tables and figures of this section will help you to find your way around the Creed. The component location figure and table and the special component location figure should get you oriented. The combination of the wiring diagram and wiring layout and color coding should enable you to work with the wiring if you desire to. The operations counter chart and figure will explain its operation. Some additional figures are placed elsewhere in the manual where it seemed appropriate.





COMPONENT	CODE
Q_1-Q_5	2N3725
D_1-D_5	1N625
D_6	1N4004

Page 56


PROCESSOR
INTERFACE CIRCUIT
CREED TTY
INPUT PORT 0 FOR PARALLEL INPUT INTERFACE
7
6
5
4
3
2
1
0
1k
+5
TO TAPE READER OR CASSETTE INTERFACE
STROBE P
BIT 2
MARK
SPACE
BIT 5 M
BIT 4 L
BIT 3 K
BIT 2 J
BIT 1 H
N
R
INPUT PORT 0 FOR SERIAL INPUT INTERFACE
TIMING CONTACTS
START
STOP
CODE CONTACTS

8. Basic Creed Monitor

The Basic Creed Monitor (BCM) takes up the first 256 word page of memory and performs many essential functions. The monitor commands listed below, which are entered via the Creed keyboard, largely replace the panel functions and are much more powerful and easier to use. The loader program is essential in order to get other programs into memory conveniently.

Although the BCM is intended to be in a Read Only Memory(1702A PROM) in page 0, it also requires an area of RAM (read/write memory) in page 2 for temporary storage and user options. The BCM uses 3 of the 8 restart instructions, but the other 5 are available for user use by means of pointers in RAM. There is also a pointer to allow the user to add his own monitor commands which are two letters, V followed by a letter chosen by the user.

A serial monitor will also be provided which will differ from the following listing in that a JMP RD instruction will be inserted at BCIN (331). Another version of the serial monitor (BCM-S2; the first serial monitor will be called BCM-S1) with reassigned addresses and two more commands (display H and display L) will be offered later.

A two page monitor will also be offered called BCM-2(on two 1702A's) which will perform a few more commands and do conversion between ASCII and Baudot as well as between a 6 bit code(5 Baudot bits and a case (upper=1 and lower=0) bit) and Baudot. The BCM-2 will also have about half of the dump(to paper tape or cassette interface) routine. The rest of the dump routine may be tailored to the desired format by utilizing the appropriate pointers in RAM and adding the correct subroutines. A couple of changes are necessary in the BCM listing included here to utilize it with the second page of the BCM-2. The changes are inserting the commands JFZ 407; JMP 400 at location 136, in order to extend the monitor command search. The second page of the monitor (see listing in 9.) will be located in page 1. Other versions of the monitor using other pages for RAM, e.g. page 10, and for the second page of the monitor will be offered if there is sufficient interest. It would be possible to order the first page of BCM-2 and load the second page in RAM, however it would require loading manually via switches the following instructions every time the processor is turned on: JMP CLP at 400; CFZ 1005 at 407 and JMP CLP at 412.

8.1 Basic Creed Monitor Commands.

0,1,...,7 - Shift buffer left 3 bits, put octal digit in low 3 bits and display.
L - Put buffer in L, clear buffer and display.
H - Put buffer in H, clear buffer and display.
M - Put buffer in memory word addressed by H:L, clear buffer, increment H:L and display buffer.
C - Display memory word addressed by H:L.
I - Increment H:L.
D - Decrement H:L.
O - Load memory from serial input (paper tape or cassette).
X - Execute loaded program.
V* - Execute user control routines or programs as defined by user in RAM.
A* - Display L.
B* - Display H.
S# - Put H:L in start address (1001).
N# - Put H:L in end address (1003).
Z# - Dump memory from start to end address. Format determined by user.

- Only in BCM-2.
* - Only in BCM-2 or BCM-S2.

8.2 Loader format

← { bbbb....bbbbbcdefghgh....ghbbbbbbb }

b - blank tape
c - low 4 bits of low 8 bits of load address
d - high 4 bits of low 8 bits of load address
e - low 4 bits of high 8 bits of load address
f - high 4 bits of high 8 bits of load address
g - low 4 bits of a data word
h - high 4 bits of a data word

The high order bit of each 5 bit word is ignored, except in looking for the blank header and trailer, i.e. the 5th bit may be 1 or 0 except that 0000 must always be 10000 rather than 00000.

8.3 Subroutine descriptions.

SUBROUTINE	ADDRESS	REGISTERS DESTROYED	DESCRIPTION
TIME	10	C	WAIT FOR 6.67 milliseconds
INHL	40	H,L	INCREMENT H AND L COMBINED
CA2	147	A,C,H,L	SEARCH TABLE- SEE COMMENTS IN LISTING
WD	227	A,B,C,D	COMBINE TWO 5 BIT WORDS, STORE IN MEMORY AND INCREMENT H:L
RD	250	A,B,C	GET 5 BIT WORD FROM SERIAL 100 WPM DEVICE AND PUT IN A (LOW 5 BITS)
CD	301	A,D,E	DECREMENT D:E
CI	311	D,E	INCREMENT D:E
BCOT	315	A,C	TRANSMIT LOW 5 BITS OF A TO CREED
BCIN	331	A,C	GET CHARACTER FROM CREED AND PUT IN LOW 5 BITS OF A
INII#	440	A,B,C	GET CHARACTER FROM CREED KEYBOARD, TRANSLATE TO ASCII AND PUT IN A
IN6#	454	A,B,C	GET CHARACTER FROM CREED KEYBOARD, TRANSLATE TO 6 BIT CODE AND PUT IN A
OTII#	617	A,B,C	TRANSLATE ASCII CHARACTER IN A TO BAUDOT AND TRANSMIT TO CREED PRINTER OR PUNCH
OT6#	642	A,B,C	TRANSLATE 6 BIT CHARACTER IN A TO BAUDOT AND TRANSMIT TO CREED PRINTER OR PUNCH.

8.4 Addition of 1702A's to Mark-8 memory board.

One or two 1702A's may be added to a Mark-8 memory board by a simple process as follows. Obtain a 24 pin socket and a piece of perf board about the size of the socket. Using a small drill and the perf board as a patern, drill holes for the socket in one of the blank areas above or below the 7442 decoder. Insert the socket and connect the power and ground leads. Connect the address and data output lines in parallel with those for the 1101's. Connect one of the enable outputs of the 7442 to the select input of the 1702A. If the desired enable output is already connected to a row of 1101's, e.g. the enable for page 0, then move to the 1702A and connect one of the other enables to the row of 1101's or if desired leave it unconnected. If information on the pin connections for the 1702A is desired, request when ordering the 1702A.

```
   * RESET 0
 0      JMP 1013  JUMP TO USER RST POINTER
 3 CL   LEB       PUT B IN LOW ADR
 4 CLA  SBA
        LBA
        OUT 10    DISPLAY BUFFER ON PANEL
        RET
   * RESET 1
10 TIME LCI 147   WAIT FOR 6.67 MS
12 TL   DCC
        JFZ TL
        RET
        HLT       UNUSED WORD
   * RESET 2
20      JMP 1016  JUMP TO USER RST POINTER
23 CC   LAM
        OUT 10    DISPLAY MEMORY ON PANEL
        RET
        HLT       UNUSED WORD
        HLT       UNUSED WORD
   * RESET 3
30      JMP 1021  JUMP TO USER RST POINTER
33 WD2  RLC
        NDI 360
        ORE
        LMA       FALL THRU TO DO INHL AND RET
   * RESET 4
40 INHL INL       INCREMENT H:L
        RFZ
        INH       IF L=0 THEN INCREMENT H
        RET
44 CH   LDB       PUT B IN HIGH ADR
        JMP CLA
   * RESET 5
50      JMP 1024  JUMP TO USER RST POINTER
53 CDA  DCE       CONTINUE DECREMENT D:E
        NDA
        RFZ
        DCD
        RET
```

Page 60

```
   * RESET 6
60      JMP 1027  JUMP TO USER RST POINTER
63 CX   JMP 1000  JUMP TO EXECUTE POINTER
66 CV   JMP 1032  JUMP TO EXTD CTL POINTER
   * RESET 7 - LAST WORD OF LAST JMP IS EXECUTED
   *     BY RESET 7 AS A ONE WORD INSTRUCTION
   *     WHICH IS A NOP FOR THE FOLLOWING
******* START MONITOR *******
71      LAI 377   INITIALIZE TTY OUTPUT
        OUT 11
        OUT 13
75 CLP  CAL BCIN  GET A WORD FROM CREED KEYBD
        LHC       H=0
        LHI L(TTBL-1)
        LCI 11    LOOP INDEX
105 CB  INL
        DCC
        JTZ CA    JMP IF NOT OCTAL DIGIT
        CPM
        JFZ CB    JMP IF NOT THIS DIGIT
        DCC       CORRECT OCTAL DIGIT
        LAB       GET CURRENT DATA FROM BUFFER
        RLC
        RLC       SHIFT LEFT 3 BITS
        RLC
        NDI 370   CLEAR LOW 3 BITS
        ORC       OR IN 3 NEW BITS
        LBA       PUT BACK IN BUFFER
        OUT 10    DISPLAY BUFFER ON PANEL
        JMP CLP   GO BACK TO MAIN LOOP
133 CA  CAL CA2   LOOK FOR A CONTROL CHARACTER
136     CFZ 1005  EXECUTE CONTROL ROUTINE IF Z=0
        JMP CLP   GO BACK TO MAIN LOOP
144 CA3 INL       INCREMENT TABLE ADR POINTER
        INL
        RTZ       RETURN IF END OF TABLE
   * TABLE SEARCH ROUTINE. START AT H:L ADR
   * AND GO TO END OF PAGE. EACH TABLE ENTRY HAS
   * TWO WORDS: CHARACTER AND LOW ADR. IF A IS
   * FOUND IN TABLE Z=0  AND JMP TO LOW ADR IN THI
   * PAGE IS CONSTRUCTED AT 1005. IF A IS NOT
   * IN TABLE THEN Z=1.
```

```
147 CA2  CPM        IS THIS AN OCTAL DIGIT
         JFZ CA3
153 CF   INL        CONTROL CHARACTER FOUND
         LCM
         LAH
         LLI 5      CONSTRUCT JMP INSTRUCTION AT
         LHI 2      1005
         LMI 104    INSERT JMP OP CODE
         INL
         LMC
         INL
         LMA
         LHD
         LLE
         RET
173 LOAD LLI 0      LOAD MEMORY FROM SERIAL DEV
         LHI 2      SET UP JMP AT EXEC. LOCATION
         LMI 104    INSERT JMP OP CODE
         RST 40     INCREMENT H:L
202 LD1  CAL WD     GET LOW ADR FROM TAPE
         JTZ LD1    SKIP HEADER
         CAL WD     GET  HIGH ADR FROM TAPE
         RTZ
         DCL
         DCL
         LLM
         LHA
220 X    CAL WD     LOAD LOOP
         RTZ        STOP IF BLANK TRAILER
         JMP X
    * GET TWO 5 BIT WORDS AND COMBINE THEM INTO
    * ONE 8 BIT WORD
227 WD   CAL RD     GET FIRST 5 BIT WORD
         RTZ
         NDI 17     MASK OUT ALL BUT 4 BITS
         LEA
         CAL RD     GET SECOND 5 BIT WORD
         RTZ
         RLC        SHIFT SECOND WORD IN ORDER
         RLC        TO COMBINE WITH FIRST
         RLC        WORD
         JMP WD2    CONTINUE AT WD2

    * GET A 5 BIT WORD FROM CREED
250 RD   INP 0
         NDA        DESTROYS A,B,C
         JTS RD     WAIT FOR START BIT
         LBI 100    LOAD MARKER BIT
257 RL   RST 10     WAIT 1/2 BIT TIME
         INP 0
         RST 10     WAIT
         NDI 200    MASK OUT THE BIT
         ADB        ADD A AND B AND CARRY = 0
         RAR
         LBA
         JFC RL     LOOK FOR MARKER BIT
         RRC
         LBA
         NDI 37     MASK AND CHECK FOR BLANK
         RTZ        RETURN IF BLANK WITH Z=1
         CPB        CLEAR Z BIT
         RET
301 CD   LAE        DECREMENT D:E
         JMP CDA    CONTINUE AT CDA
305 CW   LMB        WRITE B AT ADR IN MEMORY
         SBA
         LBA
         OUT 10     DISPLAY ON PANEL
311 CI   INE        INCREMENT ADR POINTER
         RFZ
         IND        IF E=0 INCREMENT D
         RET
315 BCOT ORI 140    BASIC CREED OUTPUT
         RAL        DESTROYS A,C
320 OUTL OUT 11     PUT OUT A SERIAL BIT
         RST 10     WAIT 1/2 BIT TIME
         RAR
         NDA
         RTZ        RETURN IF FINISHED WITH WORD
         RST 10     WAIT 1/2 BIT TIME
         JMP OUTL
331 BCIN INP 0      BASIC CREED INPUT
         NDI 100    DESTROYS A,C
         JTZ BCIN   WAIT FOR STROBE BIT
         RST 10     WAIT 1/2 BIT TIME
```

Page 61

```
340       INP 0      GET THE 5 PARALLEL BITS
          RST 10
          RST 10     WAIT FOR STROBE BIT TO GO AWAY
          RRC
          NDI 37     MASK OUT ALL BUT 5 BITS
346       RET
      * THIS TABLE HAS OCTAL CHARACTERS, CONTROL
      * CHARACTERS AND ADR OF CONTROL SUBROUTINE
      * NOTE THAT FIRST TABLE ENTRY IS ALSO RET
      * ABOVE, I E OVERLAP IS O K
346 TTBL      07     7
347           25     6
350           20     5
351           12     4
352           01     3
353           23     2
354           27     1
355           26     0
356           22     L
              L(CL) PUT BUFFER IN LOW ADR
360           24     H
              L(CH) PUT BUFFER IN HIGH ADR
362           06     I
              L(CI) INCREMENT ADR POINTER
364           34     M
              L(CW) WRITE BUFFER IN MEMORY
366           35     X
              L(CX) EXECUTE LOADED PROGRAM
370           30     O
              L(LOAD) LOAD MEMORY
372           11     D
              L(CD) DECREMENT ADR POINTER
374           36     V
              L(CV) EXTENDED CONTROL RTN
376           16     C
              L(CC) DISPLAY MEMORY ON PANEL
```

Page 62

```
              RAM LAYOUT
1000          JMP        EXECUTE LAST LOADED INSTRUCTION
1003                     END ADDRESS FOR DUMP PROGRAM
1005                     SAVE AND MONITOR EXECUTE AREA
1011                     CASE BIT FOR ASCII CONVERSION
1012                     OUTPUT PORT STATUS
1013          JMP        RST 00 USER POINTER
1016          JMP        RST 20 USER POINTER
1021          JMP        RST 30 USER POINTER
1024          JMP        RST 50 USER POINTER
1027          JMP        RST 60 USER POINTER
1032          LHI        ) ENTER HERE FROM USER COMMAND V
1034          LLI        ) LOAD ADR OF USER COMMAND TABLE
1036          CAL BCIN   GET NEXT CHARACTER FROM CREED
1041          CAL CA2    DO TABLE SEARCH
1044          JFZ 1005   EXECUTE COMMAND IF FOUND
1047          RET        RETURN TO MONITOR
1050 HDTL     JMP        GO TO HEADER OR TRAILER ROUTINE
                         OR INSERT  RET  IF NONE
1053 AOT8     NOP        NOP(LAA) IF START ADR IS TO BE
                         PUNCHED OR  INSERT  RET IF NOT
1054 OT8      JMP        GO TO SUBROUTINE TO PUNCH AN 8
                         BIT WORD AND INCREMENT H:L
-------------------------------------------------------
              BCM - 2  LISTING
 400          LHI 1      )
 402          LLI 366    )ADDRESS OF CONTROL TABLE
 404          CAL CA2    SEARCH CONTROL TABLE
 407 C2E      CFZ 1005   CALL CONTROL ROUTINE IF FOUND
              JMP CLP    RETURN TO MAIN LOOP
 415 CA       LAL        DISPLAY L
              OUT 10                ON PANEL
              RET
 420 CB       LAH        DISPLAY H
              OUT 10                ON PANEL
              RET
 423 CS       LLI 1      START ADDRESS
 425 CSN      LHI 2
              LME
              INL
              LMD
              RET
```

```
433 CW  LLI 3       STORE H:L IN END ADDRESS
        JMP CSN     GO TO COMMON PART
440 INII CAL IN6    ASCII INPUT FROM 5 BIT DEVICE
        LHI 1       INDEX INTO TABLE TO TRANSLATE
        ADI 100
        LLA
        LAM         GET ASCII WORD FROM TABLE
        LHB         RESTORE H AND L
        LLC
        RET
454 IN6 LBH         6 BIT INPUT FROM 5 BIT DEVICE
        LCL         SAVE H AND L
        LHI 2       CASE BIT ADDRESS - 1
        LLI 10
        LMC         SAVE L IN RAM
        INL
464 A   CAL BCIN    GET A 5 BIT CHARACTER FROM TTY
        CPI 37      IS THIS LETTER CHARACTER ?
        JFZ B       IF NOT GO TO B
        LMC         SET CASE TO LTR (0)
        JMP A
500 TRTB    000     BLANK TAPE(NUL)
501         105     E
502         012     LF
503         101     A
504         040     SPACE
505         123     S
506         111     I
507         125     U
510         015     CR
511         104     D
512         122     R
513         112     J
514         116     N
515         106     F
516         103     C
517         113     K
520         124     T
521         132     Z
522         114     L
523         127     W
524         110     H
525         131     Y
526         120     P
527         121     Q
530         117     O
531         102     B
532         107     G
533         377     FIG (NO CORRESPONDENCE)
534         115     M
535         130     X
536         126     V
537         377     LTR (NO CORRESPONDENCE)
540         000     BLANK TAPE (NUL)
541         063     3
542         012     LF
543         055     -
544         040     SPACE
545         100     @
546         070     8
547         067     7
550         015     CR
551         075     =
552         064     4
553         056     .
554         054     ,
555         046     &
556         072     :
557         050     (
560         065     5
561         042     "
562         051     )
563         062     2
564         045     %
565         066     6
566         060     0
567         061     1
570         071     9
571         077     ?
572         047     '
573         377     FIG (NO CORRESPONDENCE)
574         043     #
575         057     /
576         073     ;
577         377     LTR (NO CORRESPONDENCE)
600 B   CPI 33      IS THIS FIGURE CHARACTER?
        JFZ C       IF NOT GO TO C
        LMI 40      SET CASE TO FIG (40)
        JMP A       GET ANOTHER 5 BIT CHARACTER
612 C   DCL
        LCM
```

Page 63

```
614        LLC         SET UP C FOR INII
           LHB
           RET
617 OTI1   LBH         ASCII OUTPUT TO CREED
           LCL         SAVE H AND L
           LHI 1        ADDRESS 6 BIT TO ASCII TABLE
           LLI 100
625 OSLP   INL
           JTS DEF     IF END OF TABLE GO TO DEF
                       DEFAULT =   000
           CPM         IS THIS ENTRY = A?
           JFZ OSLP    IF NOT GO ON TO NEXT ENTRY
635 DEF    LAL
           SUI 100     CALCULATE 6 BIT EQUIVALENT
           LHB         RESTORE H AND L
           LLC
642 OT6    LBH
           LCL
644 OIIE   LHI 2       ADR OF CASE BIT - 2
           LLI 7
           LMB
           INL
           LMC
           LBA         SAVE 6 BIT WORD
           NDI 40      MASK OUT CASE BIT
           CPM         IS THE CASE BIT THE SAME?
           LMA         UPDATE CASE BIT
           JTZ SC      GO TO SC IF SAME CASE
           NDA
           LAI 37      PUT LTR IN A
           CFZ LFIG    IF IT WAS FIG THEN LOAD FIG
           CAL BCOT    SEND CASE SHIFT CHARACTER
675 SC     LAB         SEND OUT THE CHARACTER TO CREED
           DCL         RESTORE H AND L FROM RAM
           LCM
           DCL
           LHM
           LLC
           JMP BCOT
706 LFIG   LAI 33      PUT FIG IN A
           RET

711 DUMP   RST 1       WAIT
           RST 1
           RST 1
```

Page 64

```
714        CAL HDTL    PUNCH HEADER (HDTL IS A POINTER
           LHI 2       IN RAM)
           LLI 4
           LDM         )
           DCL         ) PUT END ADR IN D:E
           LEM         )
           DCL
           DCL
           CAL AOT8    PUNCH START ADDRESS(OPTIONAL)
           CAL AOT8
           DCL
           LLM         ) PUT START ADR IN H:L
           LHA         )
713 A      CAL OT8     PUNCH AN 8 BIT WORD
           LAD
           CPH
           JFZ A       IF NOT TO END THEN DO ANOTHER WORD
           LAE
           CPL
           JFZ A       IF NOT TO END THEN DO ANOTHER WORD
           JMP HDTL    DO TRAILER AND RETURN TO CALLING
                       PROGRAM
763          *** three unused words***
766        03      A
           L(CA)   DISPLAY L ON PANEL
770        31      B
           L(CB)   DISPLAY H ON PANEL
772        05      S
           L(CS)   STORE H:L IN START ADDRESS
774        14      N
           L(CN)   STORE H:L IN END ADDRESS
776        21      Z
           L(DUMP) DUMP MEMORY TO PUNCH OR TAPE
```

OCTAL KEYBOARD INPUT LOADER -- By William B. Severance, Jr.
Center Lovell, MN 04016
(207) 925-2271

As one soon tires of loading a mini-computer such as the Mark 8 via the interrupt-jam port data switches, it becomes apparent that a calculator style entry of the octal digits is needed. In the following two diagrams, such a device is described. One will note the following features:

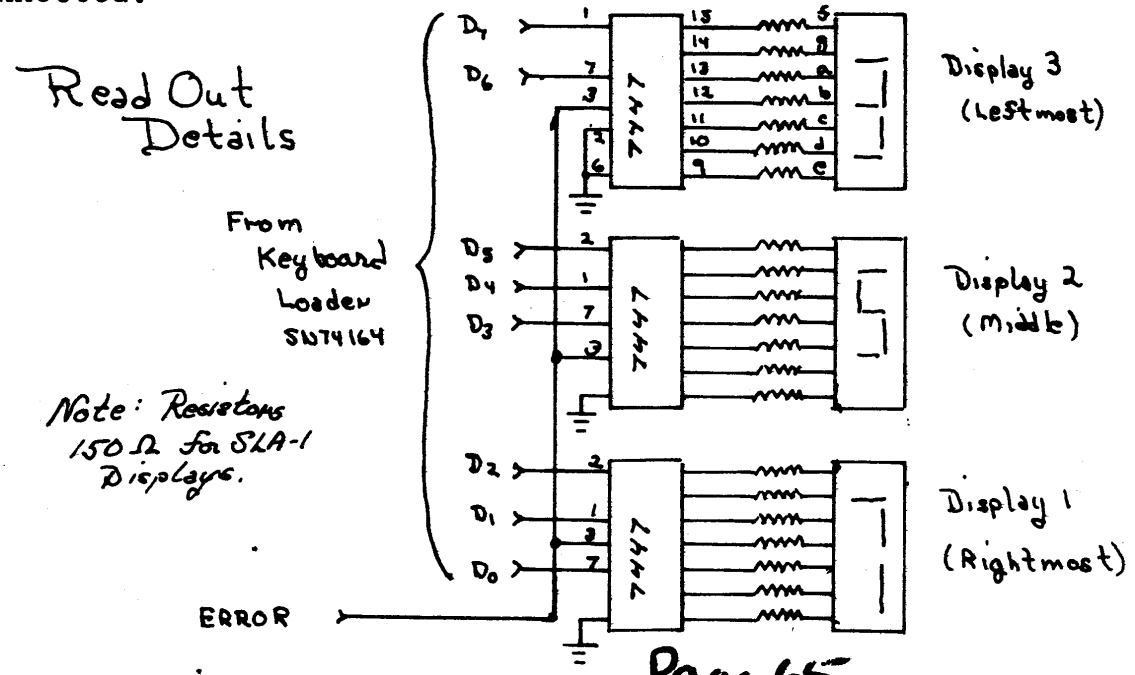
1) Digits are entered most significent digit first with automatic shifting of previously entered digits to the left with each new entry up to a maximum of 377.

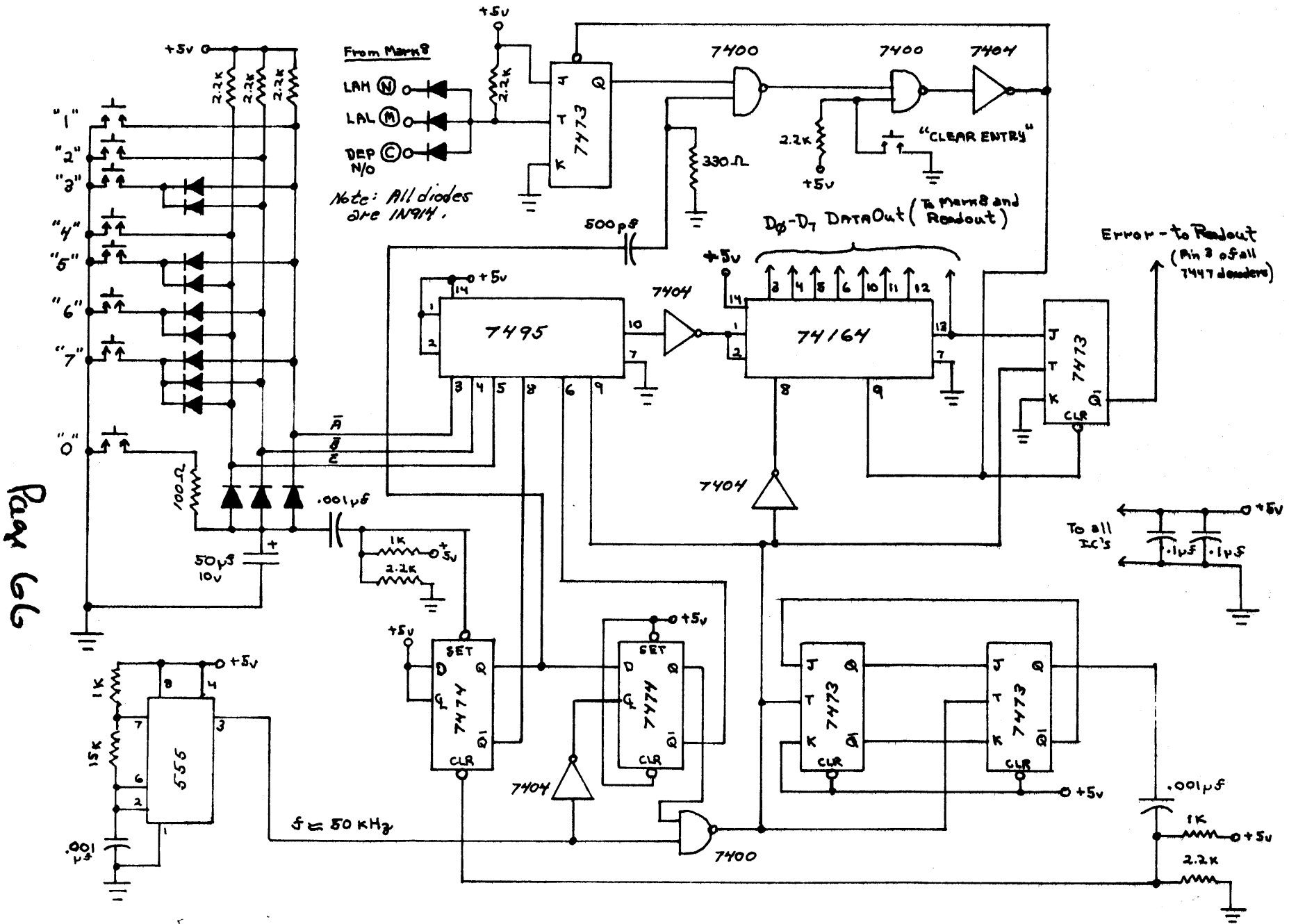
2) Should more than three digits be entered or the leftmost digit (representing bits 7 and 6) be greater than 3, the display will show an error indication of 888.

3) The "CLEAR ENTRY" key may be pressed at anytime to clear the display and output register. After an entered number is used (as signaled by pressing the Mark 8 LAH, LAL, or DEP keys), the next digit entered will automatically clear the output register and display before appearing, thus speeding up entry of one number after another.

4) Not only is this design simpler than that appearing in the Dec. 1974 issue of Popular Electronics, it is more convenient to use in that leading zeros need not be entered and there is automatic clearing of output register and display after an entry is used.

I am currently working on a PC board for this design and when ready will mail Xerox copy upon receipt of SASE. As for parts used, the following notes apply: Keyswitches are from Solid State Systems, Inc. and do not require additional debouncing--others might. Readouts are Opcoa SLA-1's. As for interfacing with Mark 8, you may simply replace the 8 interrupt port data switches with the keyboard loader and make the connections to the LAH, LAL, and DEP switches. Or, you may do as I've done and leave the interrupt port as is after adding a separate Jam input port to which the keyboard loader is permently connected.





Page 66

Octal Keyboard Input Loader — By Wm. Severance, Jr.

MARK-8 BUS SIGNAL DESCRIPTION

By Laurence L. Plate, Jr., 2320 Skyline Way, Santa Barbara, CA 93109

INTER-BOARD WIRE NO.	FUNCTION	SOURCE BOARDS	DESTINATION BOARDS	SYMBOL(S)
1	OUT PORT MSB R BIT	MA	CPU	A13,R2
2	not used			
3	COMMOM GROUND		ALL	GND,GROUND
4	STATE CONTROL SIGNAL	CPU	DI	S0,SL0
5	STATE CONTROL SIGNAL	CPU	DI	S1,SL1
6	+5 VOLT POWER		ALL	+5
7	-9 VOLT POWER		MA,CPU	-9
8	CPU READY LEVEL	MA	CPU	RDY,READY
9	MEMORY ADDRESS &	MA	CPU	D0
10				D1
11				D2
12				D3
13	INPUT BUS LINES *	DI	MM,OL	D4
14				D5
15				D6
16				D7
17	I/O OUTPUT SIGNAL **	CPU	***	OUT, OUTPUT
18	OUTPUT BUS LINES	CPU	MA,MM	D7
19				D6
20				D5
21				D4
22				D3
23				D2
24				D1
25				D0
26	MEMORY READ/WRITE SIGNAL	CPU	MM	R/W
27	I/O INPUT SIGNAL	CPU	DI	$\overline{IN}$
28	I/O OUTPUT SIGNAL	CPU	OL	$\overline{OUT}$
29	CYCLE CONTROL SIGNAL	MA	CPU	CC1
30	CYCLE CONTROL SIGNAL	MA	CPU	CC0
31	CPU LAL SIGNAL	CPU	MA	LAL
32	CPU LAH SIGNAL	CPU	MA	LAH
33	OUT PORT LSB R BIT	MA	CPU	A12,R1
34	I/O DATA ENABLE SIGNAL	CPU	DI,OL	DEN
35	I/O INTERRUPT SIGNAL	MA	CPU	INT
36	KEYBOARD INTERRUPT	MA	CPU	EX INT
37	CPU WAIT FLAG ****	CPU	N/C	FLAG
38	HIGH MEMORY ADD. BIT	MA	MM	A
39	DITTO & I/O PORT BIT	MA	MM,DI,OL	B
40	DITTO & DITTO	MA	MM,DI,OL	C
41	DITTO & DITTO *****	MA	MM,DI,OL	D

* Memory address lines and input bus lines are not common.

** Titus expansion signal for additonal output ports (note: it bypasses the RR≠00 condition,).

*** for Titus' port expansion logic.

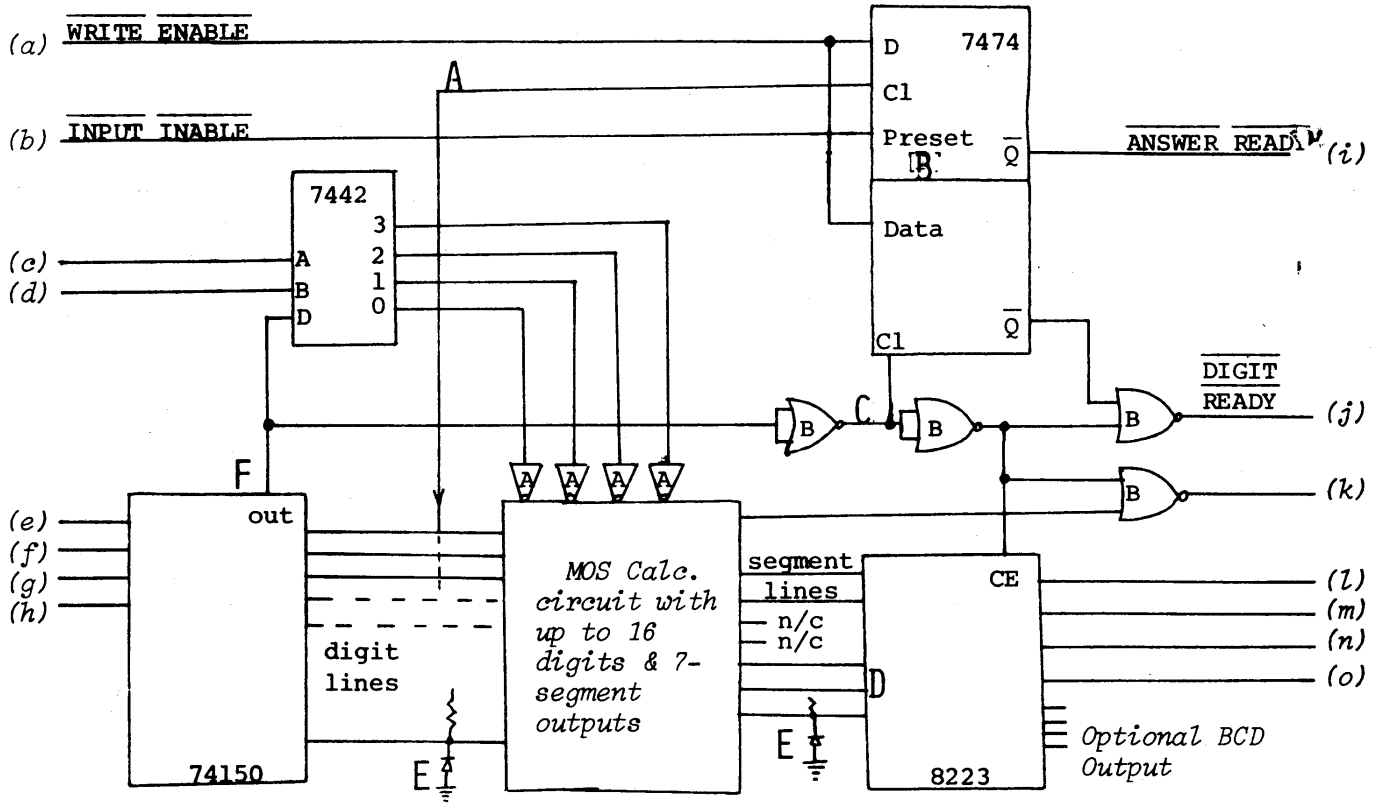
**** apparantely for slow memories

***** Wires 39-41 serve a double duty as shown above.

Prepared by Laurence L. Plate, Jr

Ken A. McGinnis
P.O. Box 2078
San Mateo, Ca. 94401

Universal Calculator Circuit with all control inputs and outputs negative true.



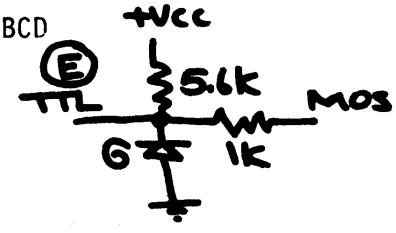
IC A is 1/6 7406 — *(c,d)* are BCD inputs to select a keyboard input (up to 4)
IC B is 1/4 7402 — *(e,f,g,h)* are BCD inputs to select a digit output (up to 16)
(l,m,n,o) are BCD outputs after conversion from 7-segments
(k) is a Decimal Point output

This circuit is similar to the one to be offered by the Digital Group, P.O. Box 6528 Denver, Colorado, 80206.

A from any digit which always appears in the answer
B INPUT ENABLE causes $\overline{Q}$ to preset so *(i)* = 1; When A appears with Write Enable *(i)* =0
C When selected digit appears with WRITE ENABLE, *(j)* is true
D segment outputs a,b,e,f,g are used as the 5 address lines to the 8223 PROM
E resistor-diode network suggested by the Digital Group as MOS to TTL converters
F 74150 output goes low when selected digit is true

TRUTH TABLE FOR PROGRAMMING 8223 FOR 7-SEGMENT CONVERSION TO BCD

Decimal Equivalent	Address Lines For Segment Outputs	Outputs Of ROM (BCD Equivalents)	
0	01111	0000	
1	00010	0001	
2	10111	0010	
3	10011	0011	
4	11010	0100	
5	11001	0101	
6*	11100	0110	*5 segment 6*
6*	11101	0110	*6 segment 6*
7	00011	0111	
8	11111	1000	
9	11011	1001	*5 or 6 segments 9*



****

* FLASH !! A 4K & 8K version of BASIC from the folks at MITS !! *

****

In the latest (& first) copy of the Altair Users Group newsletter, "COMPUTER NOTES", the people who brought you the Altair 8800 have announced the availability of a 4K and an 8K version of BASIC (Beginner's All-purpose Symbolic Instruction Code). Now, if you've had occasion to use BASIC in the past then you realize that it is unquestionably the language for the home computer. If you haven't had any experience with it....then by all means start doing some checking into the subject.

We've included the two ads from the newsletter for your information. In another section of the Altair newsletter the price for the 8K BASIC was given as $500 and the Extended BASIC was $750 for non-members. As you can see, the price drops rather sharply if you own an Altair 8800. And, if you own an 8800 you should be receiving the NL, but at $30 per year for non-owners we would be somewhat hesitant to recommend it (although it is a very nice publication). Some of the features of the NL include frank discussions of some of the "boo-boos" MITS may have made so that owners can get them straightened out; introduction to the MITS Service Department; software tips (which includes quite a bit on their new BASIC); new products & price lists; a software contest; maintenance contracts; questions & answers; and etc. (in case we've overlooked anything).

Another interesting paragraph in the NL mentioned that if you own a Mark-8 or a Shelby (?) or an M16 National Computer and you would rather have an Altair 8800, then MITS will offer you $150 on a trade-in.

4K Altair BASIC Language

STATEMENTS		COMMANDS	FUNCTIONS
IF...THEN[1]	DATA	LIST	RND
GOSUB	LET[2]	RUN	SQR
RETURN	DIM	CLEAR[7]	SIN
FOR	REM	SCRATCH	ABS
NEXT	RESTORE		INT
READ	PRINT[3]		SGN
INPUT	STOP		
END			

NOTES: [1]IF...THEN can be followed by a statement. Example: IF A<5 THEN PRINT B
[2]LET is optional in variable assignments. Example: A=5 is identical to LET A=5
[3]TAB(X) within PRINT statement tabs to print column X.
[7]CLEAR deletes all variables.

FEATURES

Multiple statements per line, separated by a colon ":" (72 characters per line)
Approximately 750 bytes available for program and variable storage before SIN or SIN, RND or SIN,RND, SQR are deleted.
"@" deletes a whole line and "←" (or underline) deletes last character typed.
Direct execution of any statements except INPUT.

Two character error code and line number printed when error occurs.
Example: ? US ERROR IN 50 would indicate a reference to an undefined statement in a GOTO, etc., during execution of line 50.

All results are calculated to at least six decimal digits of precision.
Exponents may range from 10^{-38} to 10^{37}.
Maximum line number of 65535.

8K Altair BASIC Language

STATEMENTS		COMMANDS	FUNCTIONS	
IF...THEN[1]	DIM	LIST	COS	POS
GOSUB	REM	RUN	LOG	RND
RETURN	RESTORE	CLEAR[7]	EXP	SGR
FOR	PRINT[3]	SCRATCH	TAN	SIN
NEXT	ON...GOTO	CONT[8]	ATN	ABS
READ	ON...GOSUB		INP[4]	INT
INPUT	OUT[5]		FRE[9]	SGN
END	DEF[6]			
DATA	STOP			
LET[2]				

NOTES:
[1]IF...THEN can be followed by a statement. Example: IF A<5 THEN PRINT B
[2]LET is optional in variable assignments. Example: A=5 is identical to LET A=5
[3]TAB(X) within PRINT statement tabs to print column X. SPC(X) prints X spaces
[4]INP returns status of a hardware I/O channel.
[5]OUT sets status of a hardware I/O channel.
[6]DEF allows for single variable single statement user defined functions.
[7]CLEAR deletes all variables.
[8]CONT continues program execution after Control C or STOP.
[9]FRE returns number of free bytes for program or variable storage. With a string argument, FRE returns amount of free string space.

FEATURES

Multiple statements per line, separated by a colon ":" (72 characters per line)
Approximately 2K bytes available for program and variable storage before ATN or ATN, COS, SIN, TAN are deleted.
"@" deletes a whole line and "←" (or underline) deletes last character typed.
Multi-dimensioned (up to 255) arrays for both strings and numbers.
Direct execution of any statements except INPUT.
Two character error code and line number printed when error occurs.
Example: ? US ERROR IN 50 would indicate a reference to an undefined statement in a GOTO, etc., during execution of line 50.
Control C -- interrupt program (prints BREAK IN LINE XX)
Control O -- toggles suppress output switch
All results are calculated to at least six decimal digits of precision. Exponents may range from 10^{-38} to 10^{37}.
Maximum line number of 65535.
AND, OR, NOT operators can be used in IF statements or formulas.
STRINGS
Maximum length = 255 characters
String concatenation (A$ + B$)
String functions:
LEN -- length of string.
ASC -- returns the equivalent ASCII decimal number for the specified argument.
CHR$ -- truncates the numeric formula to an integer, interprets the integer as a decimal number, and converts it to its equivalent ASCII character.
RIGHT$, LEFT$, MID$ -- Return substrings of specified string formulas; beginning at leftmost character (LEFT$) or ending at rightmost (RIGHT$) or beginning at specified position (MID$) of the string formula, and containing the number of characters specified by the numeric formula.
STR$ -- number converted to a string.
VAL -- string converted to a number.

MITS Operating System
(Package I)

The operating system is designed to facilitate assembly language program development on an ALTAIR 8800 with at least 8K bytes memory and a serial I/O board (for either teletype or COMPTER I/O).

The system monitor, which resides in the first 1K of memory, enables the user to load and execute programs stored on paper tape or other external device. The user can also write device drivers coded to suit his own particular I/O needs. Programs loaded and executed under monitor supervision can be passed parameters to control their operation.

The text editor provides facilities for editing a source program (usually assembly language) read in from an external device. The program is stored in an area of memory during the editing process and is written back out to an external device when editing is complete. The user can insert, delete or replace lines in the text buffer.

The assembler reads a source program from an external device and converts it into binary form in the ALTAIR's memory. Input can be read from any device, including the teletype keyboard. A second pass of the source can be made to generate an assembly listing. The assembler itself occupies approximately 3K of memory.

For the development of small programs, the monitor, assembler and text editor can be resident at the same time eliminating the use of external I/O for the storage of the source program on paper tape, etc.

A debugging package, DDT-8800, will be available in June.

SOFTWARE PRICES

NOTE: The software prices listed on your price sheet are incomplete. This is the complete, updated software price list.

ALTAIR 4K BASIC.....
Altair customers who have purchased an Altair 8800, 4K of memory, and one I/O board (Serial, Parallel, or audio-cassette)...ONLY $60.00
If teletype or terminal (CT256 or COMTER II) is also ordered.....N/C
NOTE: You can order software when you order hardware or you can order it later. Special prices apply to all Altair customers.

ALTAIR 8K BASIC.....
Altair customers who have ordered an Altair 8800, 8K of memory and one I/O board.....ONLY $75.00
If teletype or terminal (CT256 or COMTER II) is also ordered.....N/C

ALTAIR EXTENDED BASIC.....
Altair customers who have ordered an Altair 8800, 12K of memory and one I/O board.....ONLY $150.00
If teletype or terminal (CT256 or COMTER II) is also ordered.....N/C

ALTAIR PACKAGE ONE (assembler, text editor, system monitor).....
Altair customers who have ordered an Altair 8800, 8K of memory and one I/O board.....ONLY $30.00

ALTAIR DOS (Disk Operating System).....
When purchased with 88-DCDD.....n/c

4K BASIC, 8K BASIC, EXTENDED BASIC AND PACKAGE ONE available on paper tape (8 level) or audio cassette. Specify when you order.

DOS available on paper tape, audio cassette, or disk. Specify.

ALL SOFTWARE INCLUDES DOCUMENTATION. BASIC LANGUAGE COURSE SOON TO BE AVAILABLE (there will be a charge for this).

specifications, prices subject to change. also delivery.

Altair Extended BASIC Language

Extended BASIC has all the features of the 8K BASIC plus:

1. PRINT USING for formatted PRINT statements
2. DISK I/O (to ALTAIR floppy disk)
3. Double precision (13 digit accuracy) add, subtract, multiply, divide.

A minimum of 12K is required to support Extended BASIC. The first release of Extended BASIC is planned for July, 1975. The 4K and 8K BASICs are available today. Many more features, such as integer variables (16 bits) and ELSE clauses in IF statements, are scheduled for future implementation.

MICRO-8 COMPUTER USER GROUP
CABRILLO COMPUTER CENTER
4350 CONSTELLATION ROAD
LOMPOC, CA 93436