```
                    *******  *     *  *******
                       *     *     *  *
                       *     *     *  *
                       *     *******  *****
                       *     *     *  *
                       *     *     *  *
                       *     *     *  *******

 ******   *****   *     *  ******   *     *  *******  *******  ******
*        *     *  **   **  *     *  *     *     *     *        *     *
*        *     *  * * * *  *     *  *     *     *     *        *     *
*        *     *  *  *  *  ******   *     *     *     *****    ******
*        *     *  *     *  *        *     *     *     *        *  *
*        *     *  *     *  *        *     *     *     *        *   *
 ******   *****   *     *  *         *****      *     *******  *     *

*     *   *****   ******   ******   *     *    ***     *****   *******
*     *  *     *  *     *  *     *  *     *     *     *     *     *
*     *  *     *  *     *  *     *   *   *      *      *           *
*******  *     *  ******   ******     * *       *        *         *
*     *  *     *  *     *  *     *     *        *          *       *
*     *  *     *  *     *  *     *     *        *     *     *      *
*     *   *****   ******   ******      *       ***     *****       *
```

VOLUME 1 NUMBER 1

This is the first trial issue of "The Computer Hobbyist". Until recently the computer scene was like the early days of radio with only the most dedicated and wealthy of us taking such a complex subject seriously as a hobby. Now with the breakthrough in integrated circuit technology represented by the Intel 8008 "computer on a chip", computers have become a reasonable hobby for a number of people. The Computer Hobbyist is an attempt to provide this growing community with a journal which will encourage its growth.

"The Computer Hobbyist" will contain articles on both hardware and software. Each issue will also include a want ads column which is free to subscribers and a column about electronic surplus. "The Computer Hobbyist" will also, at the option of the subscriber, print his name and address so that it will be easier for interested persons to contact one another. Submission of articles by subscribers is encouraged.

"The Computer Hobbyist" will be published at monthly intervals and mailed to subscribers if there is sufficient support of the publication. Sufficient support will be considered as the receipt of at least 100 letters of intent to subscribe. This trial issue is typical of the minimum that will be offered in terms of subject matter and length. Subscription price will be 50¢ per issue. The decision to publish December´s issue will be based upon the response received by November 30. The December issue will be mailed to all those who have written in and they will be billed for one year´s subscription at that time. This issue is being sold at area Hamfests and mailed to persons expressing a previous interest in computers as a hobby.

"The Computer Hobbyist" will be published in Raleigh, North Carolina by Stephen Stallings, Hal Chamberlin, and Richard Smith. These people have been persuing computers as a hobby for a total of over 8 man years. Letters of intent and material to be submitted for publication should be sent to:

The Computer Hobbyist
520 Sorrell Street
Cary, North Carolina 27511

A GRAPHICS DISPLAY FOR THE 8008 PART 1 by Hal Chamberlin

A vector graphics display is probably the most flexible and most desirable peripheral you can interface to a personal computer system. With such a device you and your computer can plot graphs, plot mathematical curves, draw sound waves, display and edit music scores, display text including all kinds of special symbols, generate and maintain engineering drawings, play an infinite variety of games using the screen as a playing board, generate random kaliedoscopic patterns, etc. Part 1 will describe fundamental concepts and programming techniques of the display system. Part 2 will present the digital and analog circuitry of the display generator and interface. Part 3 will discuss large screen displays and give circuits for constructing a 12 inch screen display using a readily available surplus radar CRT.

Before going further, the limitations of this system should be mentioned. Since the display is generated and refreshed directly by the 8008 program, the slow speed of the 8008 will be evidenced by display flicker when a number of items are displayed. This can be allieviated to some extent by using a long persistance phosphor screen on the monitor, developing a tolerance to the flicker, using the faster 8008-1 chip, or a combination of the three. In particular, text will be limited to around 400 displayed characters. Nevertheless the usefullness and low cost of the display system will make constructing one a most worthwhile project.

A minimum display interface would provide an X position register, a Y position register, and a beam control bit. With these the display routine could move the beam around at will by loading the coordiantes of the desired positions into the X and Y registers with OUT instructions. If the beam control bit had been set on previously, a line (often called a vector) would be drawn from one point to the next. Leaving the beam control off would allow positioning without drawing. Such an interface can, theoretically, draw anything with the appropriate programming.

Some details have to be considered however if the expected results are to be obtained. If the lines between end points are to be straight, it is essential that the X and Y registers change simultaneously. Also a special analog circuit known as a vector generator is usually necessary for uniform line brightness and minimum bandwidth requirements of the deflection amplifiers. Since 4 bit X and Y resolution is not adequate, and the 8008 only outputs 8 bits at a time, some method of simultaneously updating X and Y will have to be found. Additionally, it would be nice to be able to avoid having a seperate OUT instruction to turn the beam on and off.

The method of display control chosen for this system utilizes four different OUT instructions. These four instructions will be given the symbolic addresses XMOVE, YMOVE, XSTOR, and YDRAW in the programming examples to follow. When a byte is sent to XMOVE or YMOVE, the beam immediately moves to the new position without being turned on. A byte sent to XSTOR is stored in an 8 bit buffer register only and does not affect the beam position. When a byte is sent to YDRAW, the Y position register is loaded from the output lines and simultaneously the X position register is loaded from the buffer register providing simultaneous update of X and Y. The beam is also turned on until the line is drawn and then is extingusihed in order to avoid a bright dot at the end of the line.

The data for generating a display can be obtained in a number of ways. In cases where the pattern to be drawn is repetitive such as a gameboard, it is often advantageous to compute the display data in line. In plotting routines, the X axis would be computed but the Y axis would be taken from a table in memory. In random figures such as the outline of a state, both X and Y would have to come from a list in memory. The example draw routine in appendix 1 draws the figure defined in a memory buffer. On entry, registers H and L should point to the first byte of the buffer which contains a count of X-Y pairs to be displayed before returning. Successive bytes are paired with X first. The routine moves to the first point and then draws until the last point is done and then returns. Several calls using different buffers are normally required to display a number of disjoint figures.

Character display for data readout or text editing is also possible using the basic interface. One would have a coordinate table in memory for each different character shape to be used. The display text routine would address the proper table according to the character codes fetched from the text buffer. Before character segments are drawn, their coordinates have to be added to the character position coordinates in order to position the character to the desired line and space. In a fast computer these steps may not limit the number of characters that can be displayed but with the 8008 only a couple dozen characters could be displayed with a reasonable refresh rate.

In order to increase symbol display capability, a minor deflection system can be added for about $20 worth of parts and one or two more OUT instructions. The minor system consists of another X and Y register, beam control, and analog adder to add the output of the minor system to the output of the major system. Now character shapes can be traced out with the minor system and the entire character positioned with one setting of the major system. An optional size register can be added to the minor system to determine the size of the characters.

The minor deflection system used here has a 3 bit X value, a 3 bit Y value, and a beam bit, thus a character segment can fit into one byte. These are organized in the byte as 0B XXX YYY. The unused bit is normally used by the display characters routine to signify the last segment in the character. X and Y are positive three bit numbers. As a result, the lower left corner of the character should be thought of as the origin. As an example, the sequence of minor deflection bytes for the letter "A" is 000B, 126B, 140B, 012B, 332B. The size register if used is one byte with the left 4 bits specifying X size and the right 4 bits specifying Y size. The size values can be thought of as binary fraction multipliers of the minor X and Y values, thus a size of 1000 (.5) would be roughly 1/2 the maximum size of 1111 (.9375). The maximum size is normally chosen to be 1/16 of the full screen dimension.

A naively written program loop for stepping through the stroke list of a character might be LAM, OUT, ORA, JTS, INL, JMP where H & L point to the strokes to be outputted and the assumption is made that the list is not split accross a memory page boundary. Using the above loop, 43 states are required for each stroke. The display character subroutine in appendix 2 makes a few minor alterations to the basic scheme in order to increase the speed to 22 states per stroke. Being a subroutine, a conditional return that uses 3 states when unsuccessful can be used rather than a conditional jump that uses 9 to detect the end of list. An 11 state JMP can be eliminated if the loop is expanded into straight line code. Little additional memory is used since the most complex ASCII character (a "B") requires only 11 strokes. The ORA used to set the condition code can also be removed if the stroke list is a list of consecutive differences between stroke bytes. In this case, an ADM instruction is used to load the next stroke from memory and simultaneously set the condition code. As a byproduct, the initial minor beam position can always be 0,0 thus eliminating a stroke from many characters such as A, B, D, etc. The program segments in appendix 2 are part of a complete character display package developed by the author. The package occupies 512 bytes and contains everything necessary to display formatted text from an ASCII string in memory using the standard 64 character set.

In conclusion it is felt that the interface described is adequate for general purpose graphic display applications within the limitations of the 8008. In the future, "The Computer Hobbyist" will publish a number of programs and routines that use this display interface.

APPENDDIX 1

```
* EXAMPLE CALLING PROGRAM FOR DRAWING
* A TILTED SQUARE

SQUAR SHL TLTSQ     SET ADDRESS OF LIST
      CAL DRAW      DRAW SQAURE ONCE
       .            CHECK FOR I/O DEVICE
       .            FINISHED
      JMP SQUAR     LOOP FOR REFRESH



* EXAMPLE DRAW SUBROUTINE
* ENTER WITH ADDRESS OF BUFFER IN HL
* FIRST BUFFER BYTE IS COORDINATE COUNT
* SUCCEEDING BYTE PAIRS ARE COORDINATES
* X IS FIRST IN BYTE PAIR

DRAW  LBM           GET COORD COUNT
      CAL INHL      BUMP TO NEXT BYTE
      LAM           GET X OF FIRST
      OUT XMOV      COORD AND OUTPUT
      CAL INHL      BUMP TO NEXT BYTE
      LAM           GET Y OF FIRST
      OUT YMOV      AND OUTPUT
DRAW1 DCB           DECREMENT COUNT
      RTZ           AND RETURN IF DONE
      CAL INHL      BUMP TO NEXT BYTE
      LAM           GET X COORD
      OUT XSTOR     AND STORE IN BUFFER
      CAL INHL      BUMP
      LAM           GET Y COORD
      OUT YDRAW     AND DRAW LINE
      JMP DRAW1     LOOP


* INCREMENT H AND L SUBROUTINE

INHL  INL           INCREMENT L
      RFZ           RETURN IF NO CARRY
      INH           INCREMENT H IF CARRY
      RET           RETURN


* TILTED SQUARE

TLTSQ DEF 5         NUMBER OF COORDINATES
      DEF -20,10    STARTING POINT
      DEF 0,80      CORNER COORDINATES
      DEF 80,60
      DEF 60,-20
      DEF -20,10
```

APPENDIX 2

```
* PORTION OF DISPLAY TEXT ROUTINE
* TEXT ADDRESS IN DE

DTXT  LAB           POSITION BEAM TO
      OUT XPOS      CHAR LOCATION
      LAC
      OUT YPOS
      LHD           GET CHARACTER FROM
      LLE           ASCII STRING
      LAM
      INE           BUMP TEXT ADDRESS
      JFZ *+4
      IND
      SUI ´!´       TEST IF CONTROL CHAR
      JTS CTRL      JUMP IF SO
      LLA           ADDRESS THE STROKE
      LHI H(CHTB)   TABLE ACCORDING TO
      LLM           CHARACTER CODE
      CPI ´D´-´!´   INCREMENT H IF IN
      JTC *+4       SECOND PAGE OF STROKE
      INH           TABLE
      CAL DCHR      DISPLAY CHARACTER
      LAB           INCREMENT X POSITION
      ADI 8         SET FOR 32 CHAR/LINE
      LBA
      JMP DTXT+1
CTRL   .

DCHR  XRA           SET MINOR TO 0,0
      OUT MINXY
      LAM           WAIT FOR SETTLE
      ORM           FETCH FIRST STROKE
      OUT MINXY     TEST AND OUTPUT IT
      RTS           RETURN IF END OF LIST
      INL           BUMP L
      ADM           FETCH NEXT STROKE
      OUT MINXY     TEST AND OUTPUT IT
      RTS           RETURN IF END OF LIST
      INL           REPEAT THIS SEQUENCE
       .            8 MORE TIMES

      ORG           ORG ON PAGE BOUNDARY
CHTB  DEF L(ZEXC)   POINTER TABLE TO
      DEF L(ZDQUO)  STROKE TABLES
      DEF L(ZSHRP)
       .

ZEXC  DEF 020B      EXCLAIMATION POINT
      DEF 120B-020B
      DEF 021B-120B
      DEF 326B-021B
ZDQUO DEF 014B      DOUBLE QUOTE
      DEF 116B-014B
       .
```

SURPLUS SUMMARY

Surplus Summary will be a regular column in "The Computer Hobbyist". The main purpose of this column will be to keep our readers informed of bargains and interesting offerings in the way of surplus. Many sources of excellent quality surplus cannot afford to advertise in popular magazines or they just do 'walk in' trade. We hope that readers will keep us informed about finds in their area.

Surplus Summary will also contain short articles about some of the more common surplus items when possible. These articles will contain application information for surplus items as well as information on what some of the items available really are. Many excellent items are overlooked because people do not know what part numbers mean etc. In this vein we are including, in this issue, material submitted by Joe Tolbert on teletype equipment. If you are looking for cheap computer peripherals teletypes are hard to beat. First however a few surplus listings.

ATLANTIC SURPLUS SALES
1902 MERMAID AVE.
BROOKLYN, NY 11224 PH. 212/266-2629

These people have good prices on many types of teletype equipment. They also have FERRANTI MKII & MKIII paper tape readers for $15. The units are old vacuum tube types but for that price who cares!

MNH APPLIED ELECTRONICS
BOX 1208
LANDOVER, MD 20785 PH. 301/773-0895

We visited this guy about a week ago. He has all kinds of IC's, both guaranteed and you-test-um. Two exceptional bargains are core driver diode arrays for 5¢ and 74S181's for $2.50. Also, if you like speed he has some 625 character a second paper tape readers for $95. Units are used but in great shape. He has full documentation, something that is invaluable.

OLSON ELECTRONICS
260 SOUTH FORGE STREET
AKRON, OHIO 44327

An old standard in the surplus market. Their latest catalogue has a power supply (#XX209-T) that puts out 12 volts DC at 1 amp. and costs only 99¢. Just add an LM309K and you have a handy 5 volt power supply.

Now the Teletype story.

Teletype Corp. gear is reliable, well made and can often be bought from hams (people engaged in amateur radio for a hobby) who have been using and trading them as surplus for years. Ham magazines such as 'QST', 'Ham Radio', and '73' frequently have ads for equipment, or if you have the opportunity, go to a hamfest. Hamfests are held all over the country and you will find most anything electronic being bought, sold, and traded at them, including teletypes. Here is a list of machines and their basic characteristics.

MODEL 12. A real antique, removed from service in the 1940's. Sets consist of a page printer and keyboard with parallel data, and a mechanical transmitting and receiving distributor (that's multiplexers and de-multiplexers to us digital types). You might give it to a museum if you find one.

MODEL 14TD. Five level paper tape equipment. Reads tape and transmits serial data on a current loop. Easily modified for parallel output. Speed is determined by gears. The most common speed is 60 words per minute, with 66,75, and 100 also offered. Current surplus prices run from $2 to $25.

MODEL 14RO. Reperforator. Receives serial data on a current loop and punches paper tape. Some units also print on the tape, but the printing is several characters behind the punches. There are also 14KSR's which are the same as a 14RO with the addition of a keyboard, but they are no more valuable since the keyboard generates serial data. Common speeds are 60, 66, and 75 words per minute. These units go for anywhere from $5 to $30.

MODEL 15. This is a five level page printer. They are usually set up for 60 words per minute, but 66 and 75 word machines sometimes pop up. There are two models, RO and KSR. RO stands for receive only. KSR stands for keyboard send receive. These machines are very common as are the model 19's. Current prices run from $10 to $100.

MODEL 19. This number refers to a set of machines which include a page printer with a special keyboard which has an intergral paper tape punch, and a model 14 tape transmitter. Presently these sets bring $10 to $150. This unit as well as all the units mentioned so far are almost indestructable. They will run forever. All the above units also use five level Baudot (Murray) code.

MODEL 20. A six level machine used by wire services for newspapers. Similar to model 15 except for expanded character set. Very rare.

MODEL 26. Light duty 60 word machine used for a while by the wire services. Rare.

MODEL 28. General number for a series of machines. Included in the series were RO's, KSR's, ASR's, TD's, and reperforators. ASR stands for automatic send and receive. This means that the unit has a built in reader and punch. These machines are also five level Baudot, but operate at 100 words per minute. They are more modern (slightly) than the 15's and just as durable. Model 28's are still in commercial use and therefore cost more. Current prices are: RO $75 and up, KSR $100 and up, ASR $100 and up, TD $25 and up, and reperforators $20 and up.

MODEL 29. Special version of model 28 built for IBM. Rare.

MODEL 32. Light duty Baudot machine. Runs 100 words per minute and is available as RO, KSR, and ASR. These units are presently in commercial service and can cost anywhere from $100 to $1200. Typical used machine brings $400.

MODEL 33. Light duty eight level ASCII machine. Same construction and types as 32. Very popular machine, used on many new mini-computers today. Units bring $100 to $1500. Typical used ASR brings $700.

MODEL 35. Eight level ASCII version of 28. Nice but expensive.

MODEL 38. Same as model 33 but platen accepts standard computer forms.

MODEL 40. Latest telêtype machine. Fast, beautiful and very expensive. Uses ASCII.

Future articles in "The Computer Hobbyist" will describe how to interface and program a teletype on an 8008 micro-processor. The staff has already interfaced a Kleinschmit TT24, a five level 100 word machine which is electrically equivalent to a 32 KSR. Had time permitted the article would have been in this issue.

NOTES ON THE 8008 INSTRUCTION SET

The 8008 is surely destined to become the most popular computer ever both in sheer numbers sold and in the number of people using and programming them. The instruction set of the 8008 will probably be the most praised and cursed ever since the PDP-8. In order to effectively utilize the 8008, both the experienced and the beginning programmer needs to become familiar with some simple tricks and techniques. The assembly language conventions used in "The Computer Hobbyist" for programming examples basically conform to those Intel uses in their manuals. The major exception is the upper byte operator, H( ), and the lower byte operator, L( ). The following discussion and examples assume a basic understanding of programming and of the 8008 instruction set such as can be obtained from the Intel manuals.

Memory addressing in the 8008 is accomplished by loading the desired memory address into registers H (upper byte or page number) and L (lower byte or byte number). Loading an arbitrary address into H and L can be accomplished either with the sequence LHI H(ADDR), LLI L(ADDR) or with SHL ADDR both of which use 4 bytes of memory and 16 states of time. If frequently referenced variables, flags, and data tables are kept in one page of memory, the LHI instruction can usually be omitted saving 2 bytes and 8 states. Often two simultaneous address indices are required such as for moving data. One effective method of handling this situation is to keep one index in registers D and E and the other in H and L. A short utility subroutine named SWAP is then called when necessary to exchange the two indices. An auxiliary register such as B or C is needed for execution of SWAP however. Other utility subroutines useful in memory addressing are INHL (increment H & L), DCHL (decrement H & L), ATHL (double add A to H & L), SFHL (double subtract A from H & L), and LDHL (load address pointed to by H & L into H & L). In most large programs these are called so frequently that it is advantageous to locate them in low memory at locations 10, 20, 30, etc. (octal) so that they may be called with the RST instruction. This not only saves 2 bytes per call, but also saves time as RST is a 5 state instruction.

Because of the short 8 bit wordlength of the 8008, unsigned arithmetic is often helpful in handling counts and other inherently non-negative numbers. Unsigned arithmetic allows such quantities to be as large as 255 instead of being limited to 127 as with signed arithmetic. The distinction is actually only one of interpretation as the same add, subtract, and compare instructions are used for either. The difference lies in the interpretation of the condition code after an add, subtract, or compare. The zero flag can be used as usual to detect equality or a zero result but the sign flag is relatively worthless. Instead, the carry flag is used in place of the sign flag. After an add or subtract, the carry should be off unless either a zero result occurred or an overflow or underflow occurred. The zero flag could then be tested to verify an overflow or underflow. For compares assume that A is in the accumulator and that it is being compared with B. For the case of A<B, carry will be on and zero will be off. When A=B, carry will be off and zero will be on. Finally, when A>B, carry will be off and zero will be off. If a habit is made of treating all inherently non-negative quantities as unsigned numbers, many potential bugs can be avoided.

Occasionally one needs to set A to 377B if A is negative and to 000B if A is zero or positive. A short two instruction sequence to accomplish this is RLC, SBA. It really does work!

Utility subroutines can often be made more utilitarian if they are written so that they can be chained together. Example 1 shows a simple add subroutine that adds the byte at (HL) to the byte at (DE) and stores the sum at (DE). Example 2 shows the same subroutine written for chaining. The first difference is the inclusion of an initialization entry point that clears the carry flag. The second difference is that the two address indices are decremented before returning in anticipation of a chained call. Now if a triple precision add is required, CAL ADD, CAL ADDO, CAL ADDO is all that is necessary after D, E, H, and L are set up. Chaining also eliminates the need for a loop counter and saves memory when 4 or less calls are chained.

Another trick that saves time, memory, and sometimes a subroutine stack location can be used whenever a subroutine calls another subroutine immediately before a return to the calling program. Instead of calling the subroutine, simply jump to it. Its return then gets back to the calling program.

Readers are invited to submit their own tricks and techniques for 8008 programming. If the originality and quantity warrants, these will be published as a regular feature of "The Computer Hobbyist".

EXAMPLE 1

```
* SIMPLE ADD SUBROUTINE

ADD    LAM           GET ADDEND
       RST SWAP      SWAP DE & HL
       ADM           ADD TO AUGEND
       LMA           STORE SUM
       RST SWAP      RESWAP HL & DE
       RET           RETURN
```

EXAMPLE 2

```
* ADD SUBROUTINE WRITTEN FOR CHAINING

ADD    ORA           CLEAR CARRY INITIALLY
ADDO   LAM           GET ADDEND
       RST SWAP      SWAP DE & HL
       ACM           ADD TO AUGEND WITH
*                    CARRY
       LMA           STORE SUM
       RST DCHL      DECREMENT DE FOR NEXT
*                    CALL
       RST SWAP      RESWAP DE & HL
       RST DCHL      DECREMENT HL FOR NEXT
*                    CALL
       RET           RETURN
```

*** CORRECTION ***

In a pre-publication handbill "The Computer Hobbyist" erroneously reported that SCELBI, a manufacturer of 8008 computer systems and kits, used tri-state and other difficult to find IC´s in their design. In reality, tri-state and other exotic logic was carefully designed out of their system. Also the maximum memory size was stated as 4K bytes when actually an expansion system is available to expand the memory up to 16K which is the maximum that can be addressed on the 8008. The staff regrets any adverse effect that these errors may have caused and will strive to prevent unconfirmed materal from appearing in this publication.

```
                    *******  *     *  *******
                       *     *     *  *
                       *     *     *  *
                       *     *******  *****
                       *     *     *  *
                       *     *     *  *
                       *     *     *  *******

  ******   *****   *     *  ******   *     *  *******  *******  ******
 *        *     *  **   **  *     *  *     *     *     *        *     *
 *        *     *  * * * *  *     *  *     *     *     *        *     *
 *        *     *  *  *  *  ******   *     *     *     *****    ******
 *        *     *  *     *  *        *     *     *     *        *  *
 *        *     *  *     *  *        *     *     *     *        *   *
  ******   *****   *     *  *         *****      *     *******  *     *

 *     *   *****   ******   ******   *     *    ***     *****   *******
 *     *  *     *  *     *  *     *  *     *     *     *     *     *
 *     *  *     *  *     *  *     *   *   *      *      *          *
 *******  *     *  ******   ******     * *       *        *        *
 *     *  *     *  *     *  *     *     *        *          *      *
 *     *  *     *  *     *  *     *     *        *     *     *     *
 *     *   *****   ******   ******      *       ***     *****      *
```

VOLUME 1 NUMBER 2 DECEMBER 1974

With this issue, THE COMPUTER HOBBYIST begins regular publication. It will be published once a month in Raleigh, North Carolina. Plans are to send out a new issue during the first week of each month. We appologize for not being on time this time, but after all it is only our second printing.

Since THE COMPUTER HOBBYIST has decided to publish regularly, we have secured a post office box for newsletter mail. Our new address is:

The Computer Hobbyist
Box 295
Cary, North Carolina 27511

Those who have written us letters of intent to subscribe will find a bill for one year's subscription at $6.00 for 12 issues printed on the inner side of the protective mailer. For those who are receiving a sample issue there is a subscription blank printed on the inner side of the mailer.

Because the name THE COMPUTER HOBBYIST is used many times in our text, it will be abbreviated as TCH in this issue and future issues. Also since there will be many schematic diagrams printed in TCH, we have included in this issue an article which we hope will standardize the notation used.

This issue contains part 2 of the graphics series which was started in issue 1, an article on logic notation, an article on interfacing teleprinters, and a write-up on a new 8080 microcomputer kit. Also included are most of the columns which will be standard in TCH. Your comments on the content of the newsletter are welcomed.

LETTERS

TCH will publish a few of our more interesting letters each month along with comments by the staff. This month we have a couple just to get started.

Gentlemen:

Having built a TV Typewriter, I would like to provide a few comments that may save future builders some time.

If on line one, to the left of each character displayed, several undesired bits of video appear, the output line register on the memory A board is probably being loaded during the previous 9-12 blank sequence. To correct this, take the output load command (pin 1, IC9, timing board) and OR it with the 9-12 blank clock (pin 11, IC7, timing board). Substitute this signal for the present output load command by breaking the connection leading to pin 45 on the timing board and jumpering wires to a spare OR gate in IC8.

If one is having difficulty stabilizing the display on an old AC-DC TV set, the problem may be due to AC ripple in the TV set. This can be verified by running the set on a 120VDC power supply.

One jumper connection on the timing board is easy to miss. If timing difficulties appear, check that pin 11, IC10 goes to test point Q (not point R as shown in the derived timing schematic).

Timing board problems can be easily traced with a TTL logic breadboard by replacing the 4.56 MHz oscillator with a variable audio oscillator and checking timing points with state indicators.

I recommend that one use reliable IC's in the project, especially on the cursor board where timing pulses are critical.

James T. Parker

Thanks. I'm sure that there are readers who will benefit from your advice. TCH hopes to be able to assist our readers with their problems, either hardware or software. If you have advice or a problem please send us a letter.

The Computer Hobbyist:

Consider this my letter of intent to subscribe.

You may wish to consider these suggestions:

Conduct a survey to determine content of future articles in larger editions by asking the subscriber's

1. Knowledge of computers, uses, operation, etc.
2. Specific areas of interest (software, storage, displays, input/output interfacing, etc.)
3. Uses of computer techniques (telemetry, signal processing, statistical operations, video games similar to "Odyssey", etc.)
4. Specialized equipment in possession or readily available (keyboards, displays, teleprinter gear, card hardware, mag. tape equipment, etc.)

This survey could easily be performed through a returnable form in a future issue.

Publish a list of material available for self-instruction in various phases of computer technology.

Include a want-ad section. Fees charged could help defray costs and maintain support of the publication even with a reduced number of subscribers.

Start a "Trivia" column for reader contribution of tips and shortcuts that individual subscribers have discovered, developed, concocted, or just want to pass on to the gang.

W. Smyth

TCH will conduct a survey in the January issue to determine the background and desires of readers. In the meantime, feel free to write us about any type of articles you would like to see. As for want ads, this issue begins a column which is free to subscribers. A trivia column is a good suggestion and will be started if sufficient material is submitted, so start sending in trivia!

THE 8080 IS HERE

It is getting to the point that every week when a new Electronic News, Computerworld, or Electronic Design magazine is received a new case of "future shock" leaves us reeling for some time. The January 1975 issue of Popular Electronics is no exception. In that issue another computer kit was announced using not the common, cheap (we thought) 8008 or 8008-1 chip but the new, super-performance 8080.

The company is MITS and the computer is dubbed the ALTAIR 8800. The price for a basic kit (less case, switches, power supply) of $298 left us here at TCH very skeptical as to the inclusion of the 8080 chip. The singles list price of $360 for the 8080 and the ambiguous language of the Popular Electronics article led us to a long and very informative conversation with H. Edward Roberts, president of MITS.

Yes, Virginia, there really is an 8080 in the basic, $298 kit. Not only that but the system is properly organized around a comprehensive, single bus, somewhat like DEC's UNIBUS but synchronous. 256 words of RAM are also included in the basic kit to get the buyer started. The lack of a power supply in the basic kit is a minor problem since MITS's philosophy is local regulators on each board. Other versions of the ALTAIR 8800 currently offered are a complete system kit with cabinet, console, and power supply for $397 and an assembled, checked out, 90 day warranted system for $498. First deliveries are slated for mid-January, 1975.

Large memory users have not been forgotten by MITS. Although the memory board supplied with the system uses 1K static RAM's and has a maximum capacity of 1024 words, larger boards with 4096 words each will be available. These will use the new 4K dynamic RAM chips which until now have been enthusiastically ignored even though the price per bit in 25 quantity from industrial distributors is actually less than surplus 1103's. The 4K RAM's being used are the TI style with 22 lead packages, the most common type. Price for the 4K board will be about $230.

Besides the computer, MITS has on their drawing boards or in the lab a complete line of peripheral gear for their systems. There will be four different types of general I/O interface boards, one 8 bit parallel and the other three serial using UART's. An audio recorder modem usable with one of the serial interfaces will be available. A 16 line by 64 character alphanumeric CRT display is planned using a high-quality monitor, which is required for such a long line. Both hard and floppy disk controllers are planned using, you guessed it, an 8080 chip for the control intelligence. These disk controllers will use the inherent direct memory access capability of the ALTAIR bus. Other more esoteric devices mentioned in our conversation include a 32 by 32 element image sensor, PROM programmer interface, and 8 bit analog-to-digital and digital-to-analog converters.

MITS has not forgotten software either. Naturally an assembler has been written. This one is unusual however in that it is a one-pass assembler. It will be free to customers who purchase more than 8K of memory with their ALTAIR 8800. Of course I/O handlers are being written for each peripheral device. In addition an operating system is being written. High-level languages are being considered with BASIC at the top of the list. Software will probably be for sale to anybody.

The natural question is how can MITS offer so much for so little and stay in business. The answer is both supprising and indicitive of near future trends. It is well known that Intel and others can be induced into giving deep discounts on complex IC's when quantities get into the range of 10K to 100K, particularly with the current slump in the economy. Deep discounts here mean a factor of five or so. Mr. Roberts said flatly in response to our question that his company believes the potential computer hobbyist market is larger than the amateur radio market. The latter has over 300,000 liscense holding members, supports several high quality publications and manufacturers, has several active nation-wide and worldwide organizations, and even has a yearbook. The prospect of a "computernik" on every block is truly mind-boggling. In the words of Mr. Roberts, the whole concept is truly "super!"

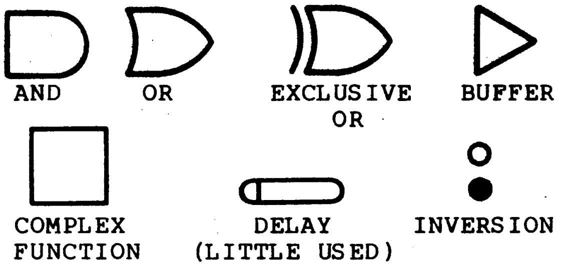
LOGIC SYMBOL CONVENTIONS
or
HOW TO READ TCH LOGIC DIAGRAMS

It is a historical fact that an intellectual discipline does not expand and mature until a suitable, human-engineered notation is developed for that discipline. The notation must lend itself to convenient, meaningful manipulations which aid the thought processes of the user. In this way new properties of the discipline are discovered, understood, and propogated.

The most obvious example of this principle is the development of mathematics. Roman numerals adequately represented numbers but could not be conveniently manipulated and combined. Arabic numerals and the base-radix method of notation helped unlock all of the "secrets" of mathematics since discovered. Music also was static in its development until the present notation was devised and adopted. In short, a good method of notation aids the thinking of the user and helps guide him to the desired conclusion.

Logic design is also an intellectual discipline which requires a good notation method. In the early days of computers only a few highly talented people practiced logic design. Symbologies such as IBM's square boxes for everything or DEC's schematicized diode-transistor symbols were developed for the purpose of documentation. These and other logic notations were of little value in the design or understanding of digital logic systems. Several years ago the Military realized that in the future it would become necessary for a great number of people of average intelligence to design, understand, and troubleshoot logic systems. With this in mind MIL-STD-806C was formulated. Logic diagrams in TCH will conform to the spirit of this standard.

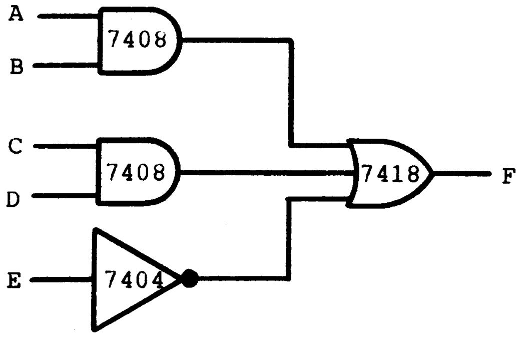
The 806C standard consists of two fundamental parts, the mnemonic logic shapes and the "dot convention." In addition, TCH will adopt some additional standards designed to reduce ambiguity and confusion. The basic logic shapes are shown below:



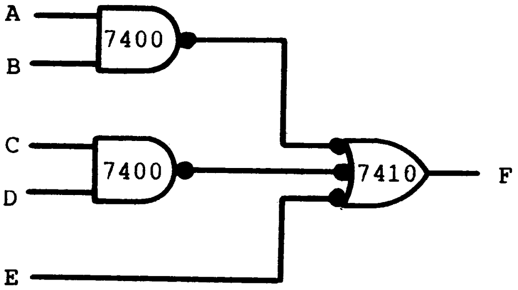
Note that the triangle by itself is a non-inverting buffer or amplifier. The solid dot and open circle are interchangable, with the former being easier to draw. Whenever a signal passes through a dot it is logically inverted, i.e., a one becomes a zero and vice-versa. Thus an inverter such as a 7404 may be drawn with a dot on the output or on the input. The same holds for the other logic symbols, dots are allowed on both the inputs and the outputs.

Figure 1 shows the truth table of the possibilities of AND and OR gates and dots. A little observation should reveal that the truth tables for the NAND and the OR-NOT gates are identical as are the NOR and AND-NOT gates. Likewise an AND gate and an NOR-NOT gate are the same and the OR and NAND-NOT gates are the same. From this the following rule can be formulated: The truth table or a gate is unchanged when the DOT/NO-DOT status of all lines is flipped and the AND and OR symbols are interchanged. This also holds for any number of inputs. This is actually just an expression of DeMorgan's theorem in this logic symbol notation. The rule also applies to groups of interconnected gates. Figure 2 shows the two representations of the common 7451 AND-OR-INVERT gate.

At this point the question arises as to why play around with dots and truth tables at all, why not just draw 7400's as NAND's and 7402's as NOR's, etc. Although doing so would certainly document the circuit, it would in most cases hide the logical functioning of the circuit and thus be bad notation. An example should serve to prove the point. Assume that for some reason the logic function $F=(A.B)+(C.D)+\overline{E}$ is needed (. means AND and + means OR) and that A, B, C, D, and E are available in their true forms only. TTL gates are now available to realize this function directly as shown below:



However the following will also work with some important advantages:

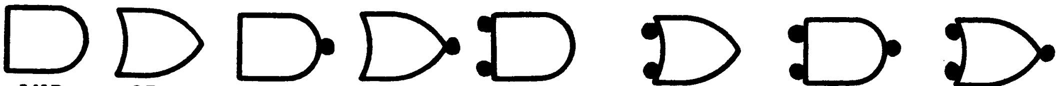


In order to understand how the above works consider the line at the output of the topmost 7400. A and B are ANDED together by the AND symbol and then the result is inverted by the dot at the output. The inverted result then travels along the wire and is inverted again at the 7410 input. Since the two inversions cancel, the OR symbol "sees" A.B at this input. The same holds for the other 7400. E is inverted as it enters the OR also giving Ē at that input as desired. Think of how difficult it would be to understand the circuit operation if all of the gates were drawn as NAND's.

Advantages of the second circuit include the elimination of an inverter, use of more common less costly IC's, 50% greater speed, and 40% less power consumption. In addition, if the F output needed to drive a heavy load, the 7410 could be replaced with a 7440 with little penalty.

Synthesis of the second circuit can be as straightforward as the first. One would first draw the circuit with AND and OR symbols directly from the equation. Then by either consulting figure 1 or applying the dot movement rule to available gates, the design of the second circuit will almost automatically emerge. In order to test your understanding of this principle, try the following example using no more than two full packages of gates.

$$\overline{F1}=(\overline{A}+\overline{B}).\overline{C}.\overline{D}$$
$$\overline{F2}=(\overline{I}.J)+(J.\overline{K})$$
$$F3=F1+F2$$



A	B	AND	OR	NAND	NOR	AND-NOT	OR-NOT	NAND-NOT	NOR-NOT
0	0	0	0	1	1	1	1	0	0
0	1	0	1	1	0	0	1	1	0
1	0	0	1	1	0	0	1	1	0
1	1	1	1	0	0	0	0	1	1

Fig. 1 Variations Of Two Input Gates

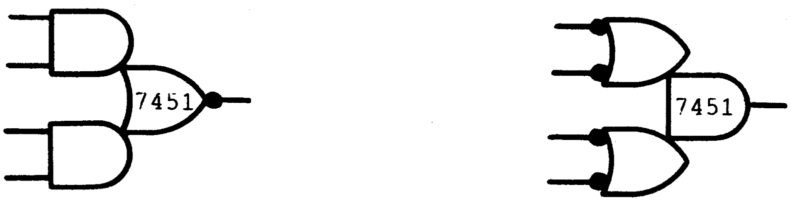


Fig. 2 Two Representations of 7451

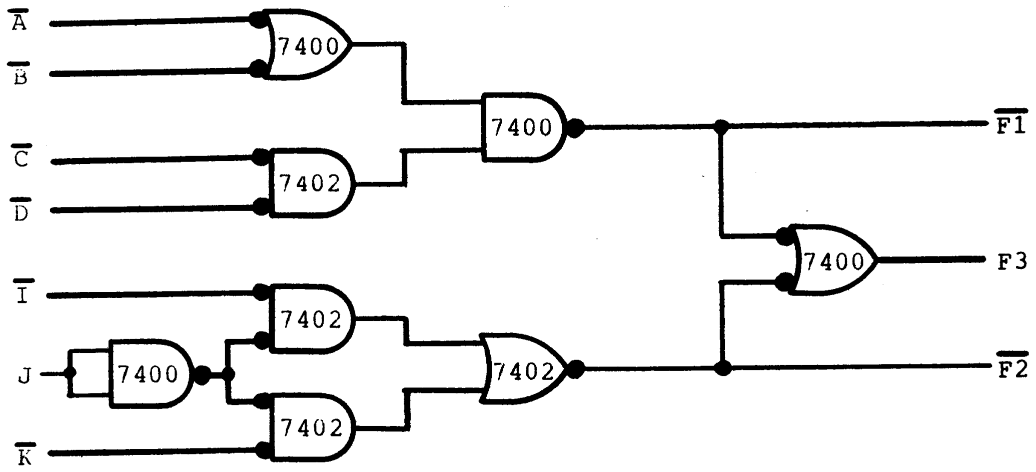
A line with dots on both ends is called a false or active-low line. The RESET or CLEAR inputs of many flip-flops and counters function when a zero is applied and have no effect when a one is applied, hence the term "active-low." Such inputs are always drawn with a dot. If a name is to be given to the line for some reason, a horizontal bar would be drawn above the name indicating that it is active-low.

When an inverter such as a 7404 is in a line, the dot may be either at the input or the output. The choice is usually made such that the inverter′s dot is placed on the line that already has a dot on the other end. Non inverting buffers such as 7407′s or 7417′s would have no dots if placed in an active-high line and dots on both input and output when placed in an active low line. The general rule is that the dot status of both ends of a line is the same unless a negation is specifically required. A notable exception to this is the $\overline{Q}$ output on flip-flops and single-shots. The dot is customarily omitted on such outputs and the bar above the Q is substituted for it.

Some flip-flops and counters have what is called a "dynamic" clock or "edge-triggered" clock. An honest edge-triggered clock responds only to the transition from one logic state to another. There are no restrictions on what other inputs can do when the clock is stationary or making the opposite non-triggering transition. Such dynamic inputs are denoted on TCH drawings by a small triangle adjoining the inside edge of the logic symbol at the input. A positive edge-triggered input responds to a zero-to-one transition and has no dot. A negative edge-triggered input responds to one-to-zero transitions and is drawn with a dot as well as the triangle. On flip-flops and counters that are not truly edge-triggered such as 74107′s and 74161′s, the dot is assigned to the clock according to the transition direction that causes the outputs to change. Thus the 74107 would have a dot on the clock input whereas the 74161 would not. Neither would have the triangle since there are restrictions on what the synchronous inputs (J, K, LE, CEP) can do when the clock is stationary in one of its two possible states (low for 74161, high for 74107) for predictable operation.

There are additional standards for data busses, bit numbering, MSI pin/function labelling and drawing organization that will be dealt with in future articles as the need arises.

ANSWER TO LOGIC DESIGN PROBLEM

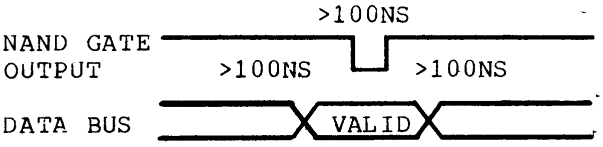


A GRAPHICS DISPLAY FOR THE 8008 Part 2 by Hal Chamberlin

This month the digital and analog circuitry necessary to generate X, Y, and Z deflection voltages for the display scope will be described. The major and minor deflection subsystems are described separately so that they may be built separately or the minor system omitted in purely graphic applications. The speed of the display circuitry has been chosen to match the speed of the 8008 or 8008-1. The vector draw time is 100 microseconds plus 20 us for settling and the stroke draw time is 40 us. This translates into a rather modest scope bandwidth requirement of 100kHz for X and Y and 2 mHz for Z. DC coupling is required however on all three axes. The included circuits have been thoroughly tested and have been operating on The Computer Hobbyist's demonstration 8008 system for two months.

All of the parts used in the display interface are standard and readily available with the possible exception of the two 8 bit digital-to-analog converters. The Motorola MC1408L8 is specified on the drawings because it is the least expensive 8 bit converter known to the author. For more precision or in the case of difficulty in getting the Motorola part, a DAC 371-8 from Hybrid Systems can be used. The price for two is $25.00 and throw in another dollar for handling and shipping. Type 748 op-amps with external compensation were used instead of 741's for two reasons. The first is that the bandwidth and slew rate of the 741 varies considerably from unit to unit, particularly with surplus parts. This is due to the on chip 30 pF MOS capacitor which suffers greatly from manufacturing tolerances. The second is that reduced compensation is possible in circuits with gain thereby increasing bandwidth.

The address decoding and data registers for the main deflection system is shown in Figure 1. The four NAND gates at the left decode the four OUT instructions; XMOV, YMOV, XSTOR, and YDRAW as described in the last issue. The inputs are connected as required by the host system such that the outputs pulse low cleanly when the output data bus contains valid data as shown below:



The following OR-NOT gates invert and transfer the output strobes to the three pairs of latches. For XMOV both the X storage latches and the X latches are enabled and the bus data ripples through the X storage latches into the X latches. For XSTOR only the X storage latches are enabled. For YMOV, the Y latches are enabled which loads them with bus data and the X latches are enabled which loads them with the contents of the X storage latches. A DRAW signal is also generated which triggers the vector generator. YMOV simply enables the Y latches to be loaded with the bus data. 7437's are used for the OR-NOT gates instead of 7400's since the clock inputs on the 2 7475's present 16 loads total. The 7404's connected to the data bus isolate the loading presented by the 7475 D inputs from the data bus. The logic is drawn for a false data bus. A true data bus can be used if Q and $\overline{Q}$ outputs are interchanged on the X and Y latches but not the X storage latches. This applies also to the minor system if used.

The signals leaving from the right of Figure 1 enter at the left of Figure 2 which shows the X and Y D-to-A converters. The MC1408 output acts like a programmable current source. When the binary input is all zeroes, the output is essentially an open circuit. When the input is all ones, the output sources a negative current equal to 255/256 of the reference current into pin 14. Intermediate binary inputs provide the equivalent fraction of the reference current to the output. This reference current is set to approximately 2 MA by the 2.7K resistor connected to the zener regulated +5.1 volt supply. The 0 to -2 MA analog output is summed with a +1MA fixed current in an op-amp which converts the resulting -1MA to +1MA current into a -2.5 to + 2.5 volt voltage. The 1K trimpot adjusts the output offset so that a zero word from the 8008 (10000000 to the MC1408) results in 0 volts at the op-amp output. The 500 ohm trimpot adjusts the output scale factor so that a -128 from the 8008 (00000000 to the MC1408) results in -2.5 volts output.

Figure 3 shows an alternate circuit used with the Hybrid Systems DAC 371-8. Basic operation is similar except that a positive current is sourced from the output and an internal 2MA reference is provided. The 5.1 volt reference has been made negative because of the positive output current. The op-amp is reconfigured for non-inverting operation. This setup is about twice as fast as the previous one, even though the DAC itself is slower, because the reduced compensation allowed by the 2.5 gain of the op-amp gives a faster slew rate. The speed difference is of no consequence in this display generator however.

The raw X and Y voltages from either DAC go to Figure 4 which is the vector generator and timing logic. The vector generator is based on CD4016 quad analog switches. Each switch has two signal terminals and a control terminal. When the voltage applied to the control terminal is equal to the VSS supply voltage, (-7.5 volts here) the switch is off and the two signal terminals are isolated. When the control terminal is at VDD (+7.5 volts) the switch is on and the signal terminals are connected together through about 300 ohms. The switch will behave properly however only when the signal voltage is confined into the range from VSS to VDD or +7.5 to -7.5 volts in this case. The VSS and VDD voltages are obtained from zener regulators powered from the +15 and -15 volt supplies used by the op-amps. The 2N3906 transistors form two level shifters to convert TTL 0 and +3 volt logic levels to +7.5 and -7.5 volt logic levels required by the switch control. Transistors slower than the 2N3906 should not be substituted in the level shifters.

In the quiescent state switches 1 and 3 are on and 2 is off. This allows the raw X voltage into the two storage capacitors which are charged to that voltage. The voltage accross C2 is buffered by the output amplifier and becomes the main X deflection voltage. This results in an output voltage range of -2.5 volts to +2.5 volts for full screen deflection, more than adequate for most scopes.

When a YDRAW is executed, the first single-shot is fired for a 20 microsecond settle delay. This single-shot through the OR-NOT gate turns switches 1 and 3 off. The voltage on C1 and C2 thus remains at the old value while RAWX and RAWY move toward and settle at their new values. Op-amp A2 is connected such that its output settles to twice the new value minus the old value. It can be shown that the output of A2 will always lie in the range of -7.5 to +7.5 volts, keeping the analog switch happy. The worst case occurrs when the old voltage is -2.5 volts and the new voltage is +2.5 volts (or vice-versa). 2*NEW-OLD = 2*2.5-(-2.5) = 5+2.5 = 7.5 volts.

When the first single-shot times out it fires the second single-shot which turns the beam on and turns switch 2 on. Switches 1 and 3 remain off because of the OR-NOT gate. An R-C delay has been added to delay gate response to the turnoff of the first single-shot until the second single-shot triggers, thus avoiding a glitch in the gate output. The integrating capacitor, C2, now starts charging toward the 2*NEW-OLD voltage through switch 2 and its series resistance along an exponential curve. The circuit is adjusted so that the second single-shot times out just as the voltage on C2 crosses the desired new endpoint voltage. At this time the beam is blanked, switch 2 turns off, and switches 1 and 3 turn back on. The C2 voltage adjusts itself to exactly the new voltage through switch 3 thus preventing any endpoint error from accumulating. The fact that the output voltage changes along a portion of an exponential curve has no bearing on the straightness of the displayed line provided both axes are identical. The beam velocity does change by a factor of two from beginning to end however. The only effect of this is a very slight difference in brightness between the beginning and the end of the line.

When an XMOV (YMOV for the identical Y circuitry) is executed, the raw X voltage changes to the new value but the timing single-shots are not fired. The switches remain in their quiescent state and the storage capacitors gradually charge to the new voltage. The beam also remains off and the net result is a rapid (about 25 microseconds) move to the new position.

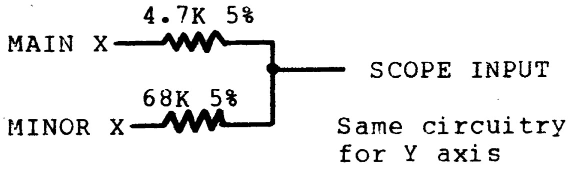
Adjustment of the vector generator is fairly simple. First adjust the two single-shots to 20 and 100 microseconds respectively. Finally adjust X and Y SLOPE so that there is no correction in output voltage at the end of a draw cycle. The visual effect of the slope controls is either an overshoot or a shortfall of the vector endpoint with respect to their desired position. A good test pattern is a simple square, preferably one with full scale coordinates.

Figure 5 shows the minor deflection system used for conveniently drawing characters or small, simple objects. Two more OUT instructions, MINSZ and MINXY are decoded by the NAND gates. The data inputs to the latches are taken from the buffering inverters used in the main deflection system. Seven bits are active in the MINXY latches with the eighth left for software use in marking the end of a minor coordinate list. Bits 0 through 5 are the minor X and Y coordinate bits. The latch outputs are fed through open collector inverters into a resistor array that is actually a pair of 3 bit multiplying digital-to-analog converters. The open collector outputs swing between approximately .15 volts for logic zero and the reference voltage supplied by the op-amps in the size register circuitry for logic one. The reference voltage can vary between .15 volt and 5 volts depending on the size register content. If a size register is not desired, the reference voltages can be obtained from op-amp buffered potentiometers tied between +5 and ground for manual variation or simply from +5 if size variation is not needed at all. The voltages at the open collector outputs are then summed in a binary weighted resistor network and sent to the minor deflection vector generator.

A four-pole lowpass filter is used for the minor deflection vector generator. The components of the filter have been chosen such that the step response assumes an "S" shape and is essentially complete in 50 microseconds. This scheme, although simple, gives good results only for the relatively short vectors drawn with the minor system. Besides filtering, the circuit has a high input impedance and low output impedance under steady state conditions which isolates the weighted resistor DAC output from the load. When a new word is sent to MINXY, a beam control single-shot is fired. At the same time the step change in minor X and/or Y voltage is smoothed into an S-shaped ramp by the lowpass filter. The single-shot times out in 40 to 50 microseconds when the transition is complete. The beam control single-shot output is NANDed with bit 6 of the MINXY latches and sent to the Z axis circuitry. Adjustment of the single shot duration should be such that the character stroke endpoints barely meet without bright dots at their juncture.

The size register and associated resistor network comprise two 4 bit digital-to-analog converters. The 1K pullup resistors at the latch outputs standardize the logic one level at +5 volts independent of individual latch characteristics. The op-amps buffer the size DAC output voltage against the varying load presented by the minor X and Y DAC's.

The main and minor X and Y coordinates are combined at the input terminals of the monitor scope with a resistive mixer as shown below:



If the leads between the display generator and the scope are much longer than a foot, they should be run through individual shielded cable. The outputs of the display generator should then be isolated from the cable capacitance with 470 ohm resistors at the display generator end of the line. Also the 4.7K resistor in the mixing network should be reduced the 4.3K.

Construction of the display generator should be done as compactly as is reasonable and preferably over a ground plane in order to minimize crosstalk coupling and ground loops. The lowpass filters in the minor system are particularly prone to VHF oscillation if not built compactly over a ground plane with power supply bypassing right at the transistors. A good material to use is phenolic microvectorboard copperclad on both sides. The copper can be removed from the edges of holes that receive components by handtwisting a new, sharp 1/8 inch drill bit in the hole a couple of turns.

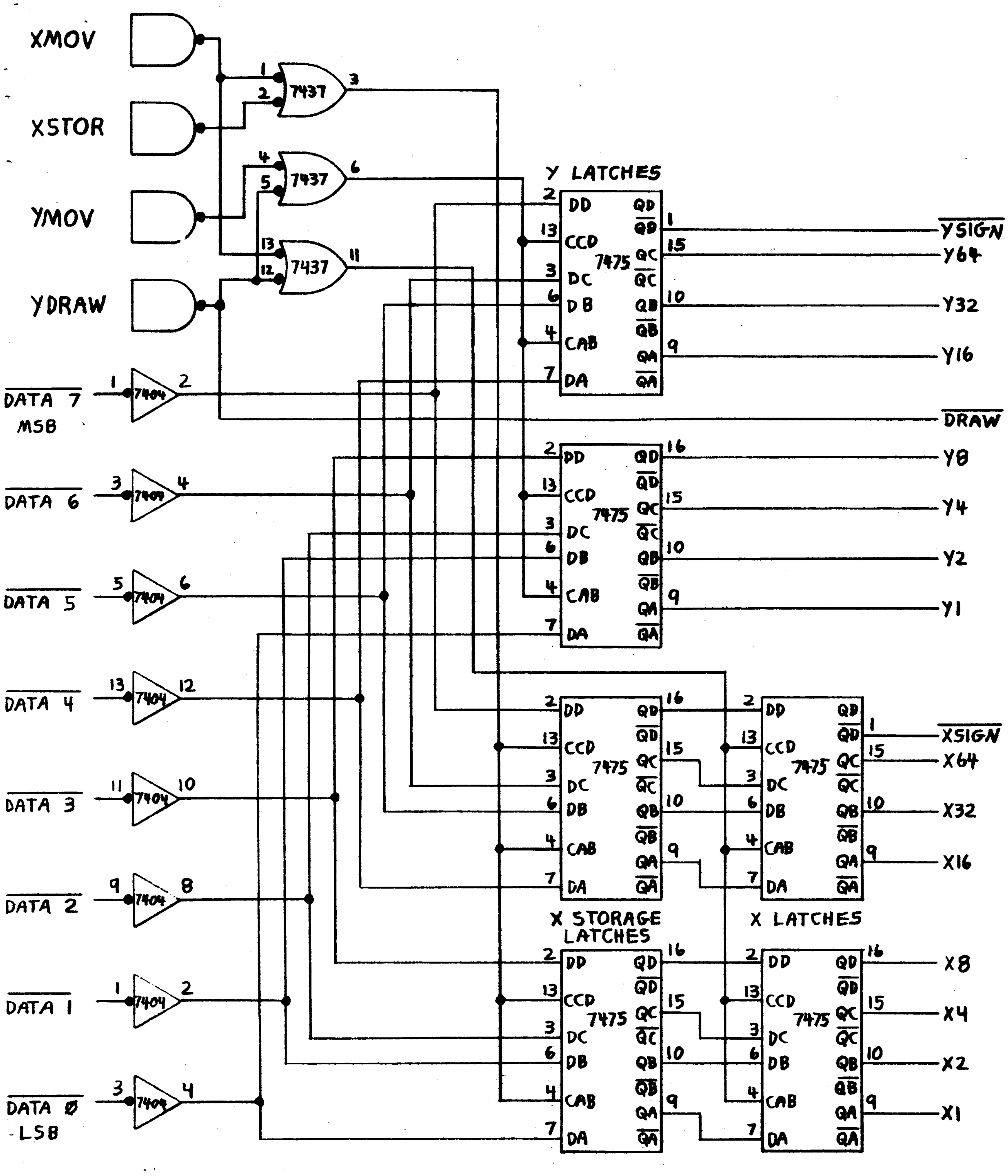


Figure 1. Address Decode and Main X Y Latches

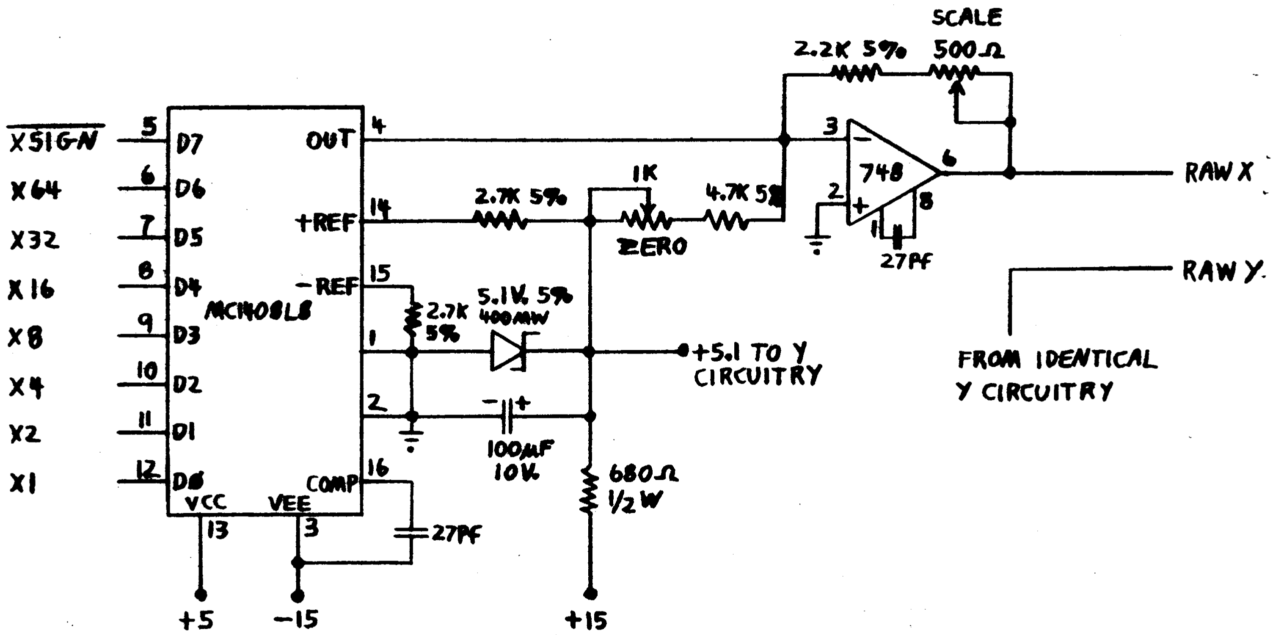


Figure 2. Main X and Y Digital-to-Analog Converters

With Motorola DAC

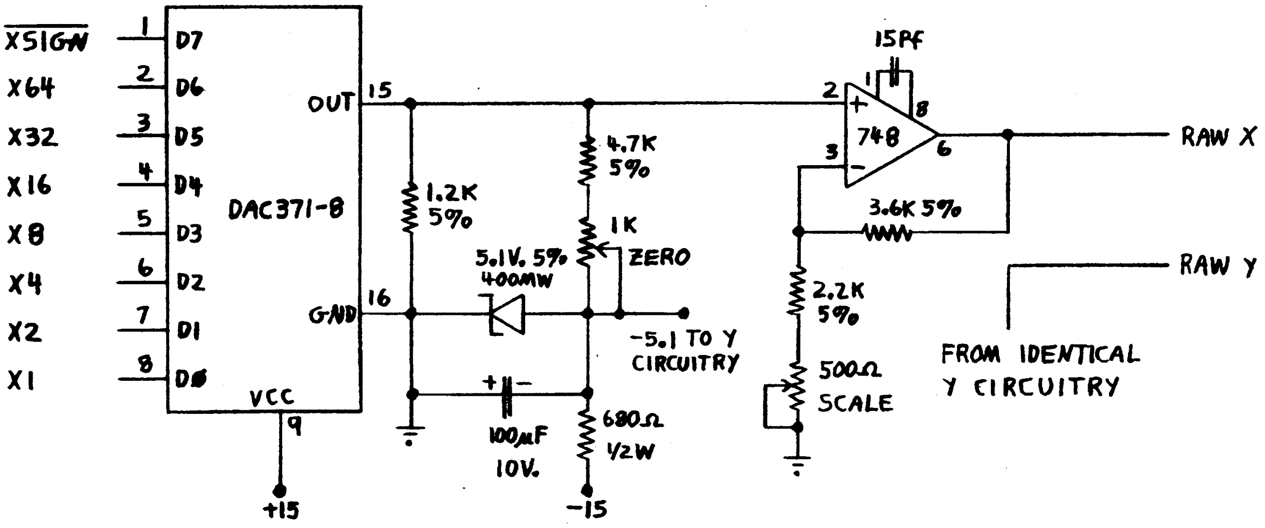


Figure 3. Main X and Y Digital-to-Analog Converters

With Hybrid Systems DAC

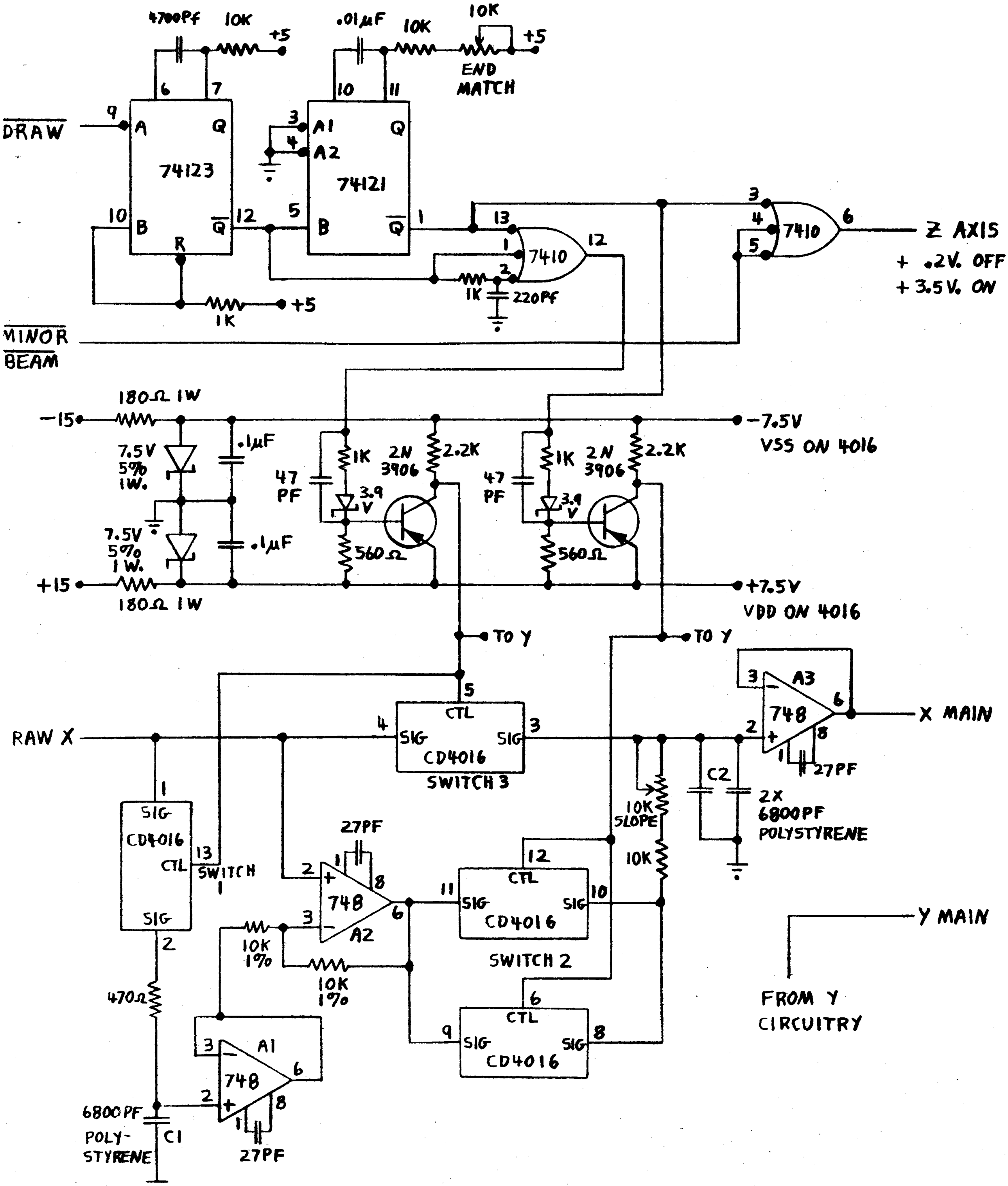


Figure 4. Main X and Y Vector Generator

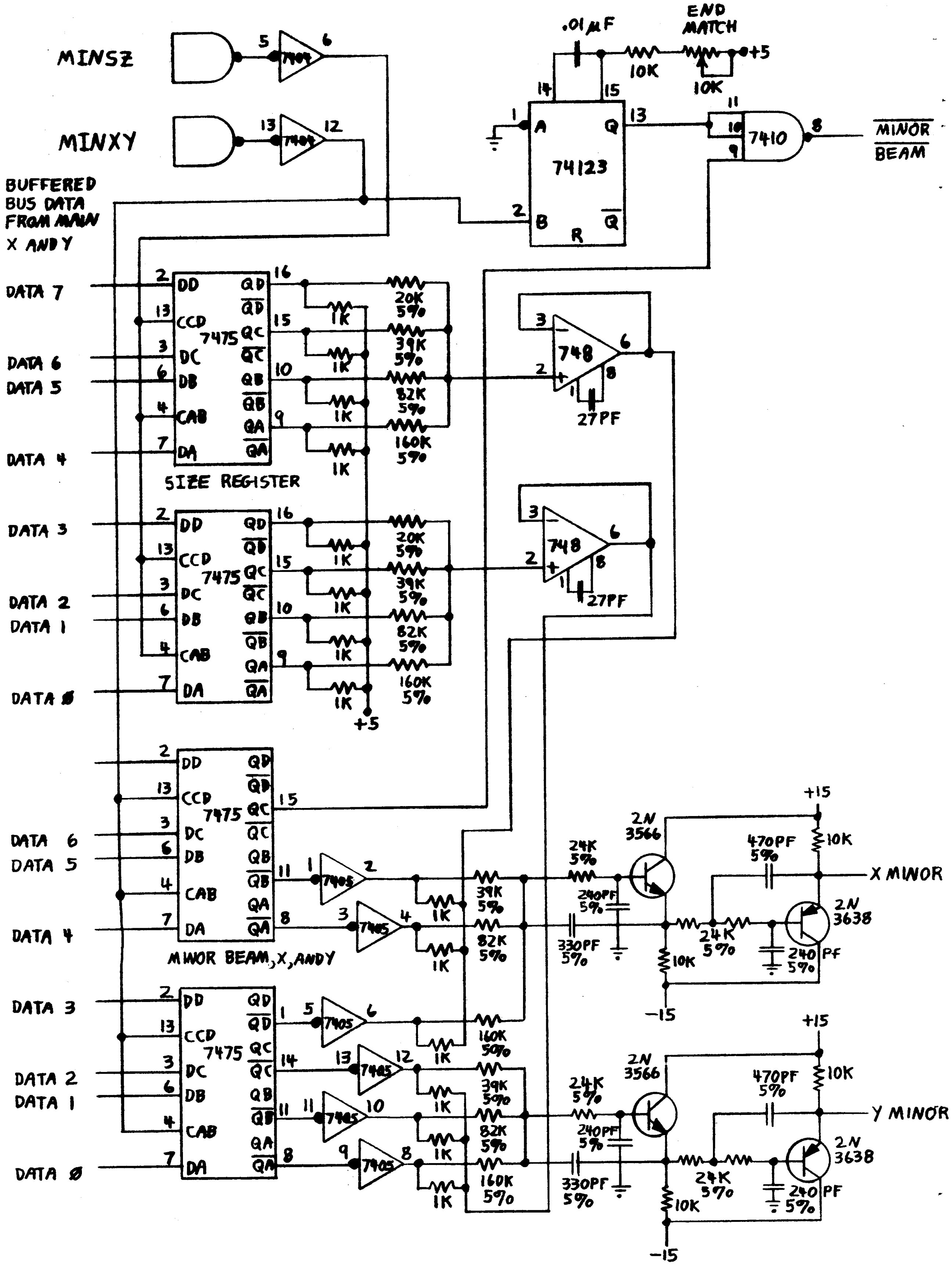


Figure 5. Minor Deflection System

Microvectorboard can be purchased from K A Electronic sales.

Take heed of the 1% and 5% tolerance specified on some components. Those not marked may be 10% tolerance. Unmarked resistors may be 1/4 watt if desired. Use only carbon or film resistors; wirewounds have too much inductance which may cause oscillation or poor performance. The +5, +15, and -15 power supplies should be semiconductor regulated so that ripple is less than 20Mv. peak-to-peak. The power drain will be less than 1 amp on +5 and .2 amp on +15 and -15.

Although excellent results can be obtained with an ordinary 5 inch oscilloscope, the full glory of graphics display can only be achieved with a larger screen. Part 3 in the January TCH will discuss a large screen magnetic CRT X Y Z monitor that has the necessary bandwidth and can be built with but one exception from readily available parts.

APPENDIX

Hybird Systems Inc.
87 Second Ave.
Burlington, Mass. 01803

INTERFACING A 5 LEVEL TELEPRINTER

This article will describe one possible way to interface a 5-level teleprinter to an 8008 system. As we mentioned in our last issue there are many 5-level machines which can be obtained cheaply. Some machines which qualify are Teletype models 12, 15, 19, 26, and 28, and Kleinschmidt models TT100 and TT117.

The interface consists of a power supply and a circuit to turn the current to the teleprinter magnet coil on and off. All intelligence and timing is contained in the program in the 8008. This allows for maximum flexibility with respect to machine type and speed. The circuit used is illustrated in Figure 1.

The NAND gate output represents the decode of the output instruction to the teleprinter. The 7474 latches bit 0 of the data bus. Timing should conform to Figure 2. If your data bus is true rather than inverted the 470 ohm resistor from the base of Q1 should connect to the true Q output of the 7474 rather than the inverted one. K1 can be any fast, light duty relay. Reed relays are ideal. The contacts are normally open. The value of R2 is determined by the coil current of K1 and Vcc. If the coil current is represented as I, then (I*R2)+Vcc=40 volts or less. The network formed by R2 and D1 clamps the voltage spikes generated when K1 turns off to less than 40 volts which is the breakdown voltage of the 2N2222. If Vcc is 12 volts and I is 50 ma, then R2 equals 560 ohms. Vcc should not exceed 24 volts and of course must not exceed the rated coil voltage of K1.

R3 and C1 form another spike suppression network, this one to protect the contacts of K1. In addition both networks help keep noise out of the logic system. R4 is used to limit the current in the teleprinter's selector magnet coils to 60 mA. This requires 390 ohms for the Kleinschmidt machines and 330 ohms for the Teletype models 15 and 19. It also makes the power supply look somewhat like a current source, which is desirable when driving the selector magnets. While the teleprinter is sitting idle this resistor will be carrying 60 mA. and will dissipate 1.4 watts. Two watts is the smallest recommended size for this resistor. While working on this circuit be careful as R4 will be quite warm and the voltage spikes generated when K1 opens can be supprisingly tingly.

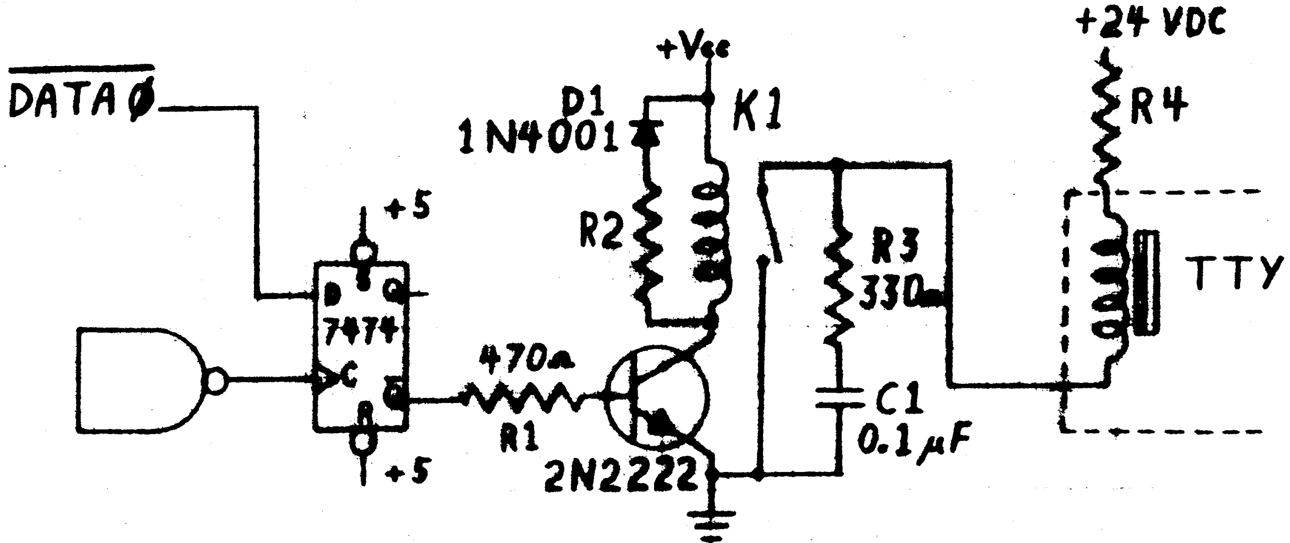


Figure 1. Details on K1, R2, and R4 in text.

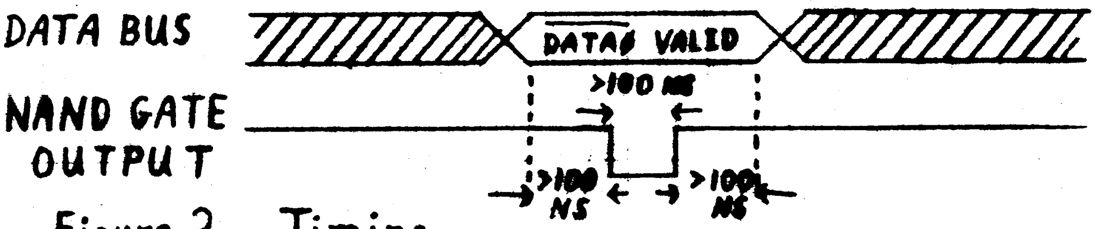


Figure 2. Timing

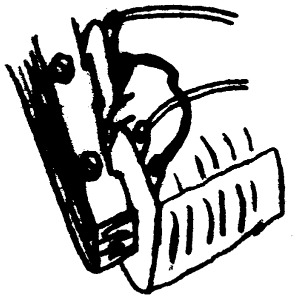


Figure 3. Teletype Corp.

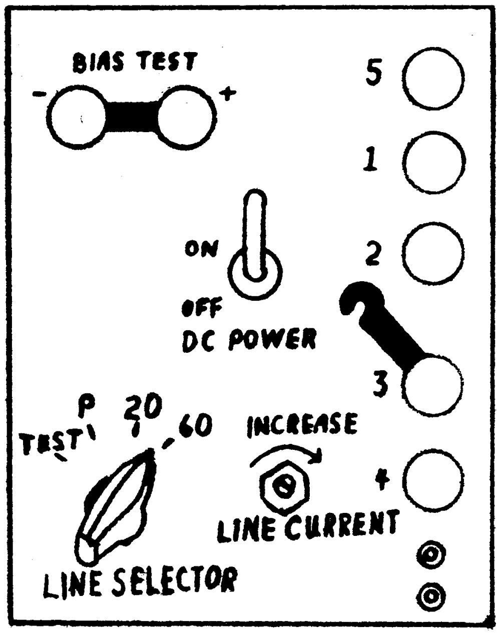


Figure 4. Kleinschmidt

So far all is fine, but we have not even told you how to locate the connections to the selector magnets on your teleprinter. Naturally this depends on the type of machine you have. Illustrations below will show connections to the more common Teletype model 15 and 19 and the Kleinschmidt model TT100 and TT117. If your machine is not one of the above we suggest that you contact a local ham for advice. In all cases the selector magnet coils need to be connected for a 60 mA circuit. This is mentioned because a few of the newer Teletype Corp. machines and all Kleinschmidt's can operate at 20 mA also. A Teletype Corp. machine properly wired for 60 mA will measure about 180 ohms accross the selector magnets and a Kleinschmidt set for 60 mA will measure about 60 ohms.

If your machine is a Teletype Corp. model 15 or 19 machine, the selector magnets are located on the left hand side of the machine. However, before you dig in, check to see if there is a cord with a 1/4 inch phone plug attached coming out of the machine. If so it should be connected to the selector coils. Measure the resistance accross it and/or connect 24 volts in series with 330 ohms to it and check to see if the armature pulls in on the selector magnet. If there is no cord on your machine, look for two coils mounted side-by-side horizontally, about one third the way up the left side of the machine. Looking at the coils from behind you should see their terminals as shown in Figure 3. The coils should be connected in series as shown, then check as above.

If your machine is a Kleinschmidt you will find a hookup box in the left rear corner of the machine. See Figure 4. The terminals labelled BIAS TEST should be connected together. The line selector switch should be set to 60. The D.C. power switch should be on in order to supply bias current needed by the coils. Connection to the coils is made on terminals 3 and 4. The positive connection from R4 must be connected to terminal 3. Again, to check out the circuit, an ohmmeter should read about 60 ohms when connected to terminals 3 and 4, and 24 volts in series with 390 ohms should actuate the armature of the selector magnet.

There are many adjustments on teleprinters, however most of them will be OK just the way they were when the machine was removed from service. The only adjustment the inexperienced should attempt is the range adjustment. If you decide to make this adjustment, start with a trial setting of 50. This adjustment will be found near the selector magnets on both machines, and is a lever on the Teletypes and a thumbwheel on the Kleinschmidts. While sending a steady signal of RYRYRYRY... find the minimum and maximum values at which the printer will copy correctly, then set the control halfway between the trial values. If you have problems with adjustments or connections try to find a knowledgeable, friendly ham radio man.

We hate to say it but there is not enough time or room to present the software for running a teleprinter in this issue. The necessary software has been written and tested though, and will be included in the next issue. For the pioneers, we will give you a few hints. Everything is done with timed program loops. Specifications for timing and bit patterns can be found in The Radio Amateurs' Handbook put out by the ARRL. As a general rule, Teletype Corp. machines will be 60 words per minute and Kleinschmidts will be 100 words per minute.

THE POWER OF AN 8008

This issue of TCH was generated on an 8008 driven system. All text was entered, edited, justified, and printed under the control of a stand-alone 8008. The system has 8K of 1103 memory and a floppy disk. The program is divided into 12 overlays and consists of over 10,000 8008 assembly language statements. The CRT display used for editing and formatting is probably the biggest ever with 69 lines of 102 characters capacity. The actual printing was done with offset masters which were photographically reduced from the printout from a Diablo Hytype I printer. Isn't it wonderful, the things you can do with a microprocessor.

SUBSCRIBER LIST

Each month TCH will publish the names and addresses of new subscribers who so desire. We hope this service will aid people in finding assistance and friends with a common interest. So far the following people have given us permission to publish their names:

Hal Chamberlin
P. O. Box 5985
Raleigh, NC 27607
Ph. 919/851-7225

C. N. Lee
College of the Albermarle
Elizabeth City, NC 27909

Jim Parker
Box 15388
North Carolina State University
Raleigh, NC 27607

Richard Smith
Box 5882
Raleigh, NC 27607
Ph. 919/467-3145 or 919/851-7223

Steve Stallings
520 Sorrell St.
Cary, NC 27511
Ph. 919/467-3145 or 919/851-7223

Joe Tolbert
Box 307
Ninety-Six, SC 29666

CLASSIFIED ADS

With this issue TCH begins publishing "want ads" submitted by our subscribers. There is no charge for ads but they must pertain to the general area of computers or electronics, and must be submitted by a non-commercial subscriber. At the present, commercial advertising is not accepted by TCH, however the matter is under consideration. Feel free to use them to buy, sell, trade, seek information, announce meetings, or for any other worthwhile purpose. Please keep length down to 10 lines or less. Now this month's ads, all 3 of them!

NEEDED: Information, hardware and software on the BIT 483 minicomputer. This machine was manufactured in Mass. but company no longer exists. R. E. Smallwood, 20-12 St., N.W., Calgary, Alberta, Canada T2N 1Y3

FOR SALE: Keypunch cards $15 per case (10,000 cards) plus shipping. These are standard data cards with columns ruled off in groups of 4 and a colored stripe accross top of card. Will send samples. Steve Stallings, 520 Sorrell St., Cary, NC 27511

INFORMATION: Will act as clearing house for people who own Computer Entry Systems surplus tape drives and need help getting them running. Gary Coleman, 530 Glaser Bldg., 11900 Carlton Rd., Cleveland, Ohio 44106

SURPLUS SUMMARY

This month we have in addition to surplus listings a special listing of places to get 8008's, and micro-computer kits.

K. A. Electronic Sales
1220 Majesty Drive
Dallas, Texas 75247

K. A. is selling microvectorboard for breadboarding with IC's. All types of clad and unclad board with holes on standard .1 inch centers is available. This allows wirewrap sockets to be simply dropped in and wired. Write for prices.

MNH Applied Electronics
P.O. Box 1208
Landover, Maryland 20785

MNH-AE has wire for your wire-wrapped projects. The price is $10 per 1000 feet of wire, and this is wire that was intended for wire-wrapping, not just wire that happened to be #30 AWG.

Les Veenstra
Action Technical Services
919 Crystal Springs Ave.
Pensacola, Florida 32505

We are told that Les has some overhauled model 33RO (receive only) Teletype machines which he will sell for $280.

Herbach & Rademan, Inc.
401 East Erie Avenue
Philadelphia, PA 19134

H&R has a number of keyboards available at prices ranging from $35 to $56. Two of the units available use Hall-effect keyswitches. Nice! Write for catalog Volume 40 No. 4.

Andy Electronics, Inc.
6319 Long Drive
Houston, Texas 77017
Ph. 713/641-0576

For those of you who cannot find a teletypewriter locally, you can buy a Kleinschmidt from Andy by mail order. They sell rebuilt model TT100's for $119. Or if you are brave, "as is" machines go for $59.

Next, here are the addresses of companies which offer micro-processor chips or micro-computer kits along with their pricing when available.

Scelbi Computer Consulting, Inc.
1322 Rear-Boston Post Road
Milford, CT 06460
Ph. 203/874-1573

8008 micro-computer blank board set $135
8008 micro-computer assembled & tested board set $440
8008 Micro-computer kit all parts including assembled boards $580

RGS Electronics
3650 Charles St., Suite K
Santa Clara, CA 95050

8008 Micro-computer kit, all parts except fuses $375
8008 chip, will sell if sufficient supply $50

Bill Godbout Electronics
P. O. Box 2673
Oakland airport, CA 94614
Ph. 415/357-7007

8008 chip $50

K. A. Electronic Sales
1220 Majesty Drive
Dallas, Texas 75247
Ph. 214/634-7870

8008 chip $66

International Electronics Unlimited
P. O. Box 1708
Monterey, CA 93940
Ph. 408/659-4773

IMP 16 Micro-computer kit $349.50

MITS, Inc.
6328 Linn N.E.
Albuquerque, NM 87108
Ph. 505/265-7553

8080 board set kit, includes 8080 $298
8080 micro-computer kit, complete $397
8080 Micro-computer, assembled & tested $498

FOR YOUR INFORMATION

The field of hobby computing is served by at least two publications other than TCH. The two best known are the publications of People's Computer Company, and Mark-8 User's Group, both of which contain great material.

People's Computer Company, P.O. Box 310, Menlo Park, CA 94025 issues a newsletter 5 or 6 times a year. A typical issue is 28 pages tabloid (11" X 17") and contains user oriented articles and lots of computer games. Regular subscription is $5 per year but PCC will let TCH readers subscribe for $3 until January 1, 1974.

Mark-8 User's Group, Cabrillo Comp. Center, 4350 Constellation, Lompoc, CA 93436 publishes a newsletter about twice a month. The newsletter was set up for owners of the Radio-Electronics Mark-8 micro-computer, but applies to the 8008 in general. It includes lots of tips and techniques for owners of the Mark-8. For information send them a self-addressed stamped envelope.

```
                      *******  *     *  *******
                         *     *     *  *
                         *     *     *  *
                         *     *******  *****
                         *     *     *  *
                         *     *     *  *
                         *     *     *  *******

 ******   *****   *     *  ******   *     *  *******  *******  ******
*        *     *  **   **  *     *  *     *     *     *        *     *
*        *     *  * * * *  *     *  *     *     *     *        *     *
*        *     *  *  *  *  ******   *     *     *     *****    ******
*        *     *  *     *  *        *     *     *     *        *  *
*        *     *  *     *  *        *     *     *     *        *   *
 ******   *****   *     *  *         *****      *     *******  *     *

*     *   *****   ******   ******   *     *    ***     *****   *******
*     *  *     *  *     *  *     *  *     *     *     *     *     *
*     *  *     *  *     *  *     *   *   *      *      *          *
*******  *     *  ******   ******     * *       *        *        *
*     *  *     *  *     *  *     *     *        *          *      *
*     *  *     *  *     *  *     *     *        *     *     *     *
*     *   *****   ******   ******      *       ***     *****      *
```

VOLUME 1 NUMBER 3   JANUARY 1975

Well here it is. Finally! This issue may be late, but we at TCH hope that its size and content make up for the delay. This issue contains four feature articles; Part 3 of the graphics series, Part 2 of the teletypewriter interface article, an article on how to program the graphics display to put a chess board and pieces on the screen, and an article on a cheap way to get card reader input to your system. This issue also contains a special treat, a centerfold, starring our graphics display!

All of the regular columns are included again with the addition of two new ones which will be seen from time to time. They are BOOK REVIEWS, and TRICKS AND TECHNIQUES.

Last issue we promised to take a survey of readers' interests and equipment. You will find the survey questions on the inner side of the mailing cover. Please answer them and fold and staple as indicated, so that our address and pre-paid postage are on the front and the return address blanks for your address are on the back. Results of the survey will be printed in our next issue if answers are returned soon enough.

Now two quick business items. First, several people have asked why we don't go to full size print and 8.5" by 11" paper which can be put in a ring binder. The reason is cost. The only way TCH can bring you this much material for 50¢ per issue, as compared to higher prices for similar journals, is to keep it small. Going to the larger format would double both the cost of paper and postage. Second, a reminder to those of you who have seasonal address changes (students, etc.), please keep us posted on where to send your TCH. We will attempt to follow you around.

LETTERS

TCH will publish a few of our more interesting letters each month along with comments by the staff.

Gentlemen:

I have just read issues 1 and 2 of TCH and find it to be the most exciting prospect I have found in many years. If you can pull together all the disciplines involved in recreational computer development, you will have accomplished a major miracle.

I have been participating in the recreational computer field for many years, but it has never been organized or fermenting as it is right now. It looks as if there will be many exciting years ahead!

I would like to make a few comments on SURPLUS SUMMARY in issue 1. I am an avid Teletype phreak and I have collected hardware, manuals, and literature for more than 20 years. Your column was an excellent tutorial on the various models and their capabilities. There are a few points that should be made, however, so that the neophyte will be prepared. First, the difference between a perforator and a reperforator should be made clear. Not understanding the two could be a major disaster! A perforator can only punch tape from a keyboard input. This means that you can prepare tapes for input, but you cannot punch tapes from your computer. The model 19 perforator, for example, punches tape from mechanical linkages to the keyboard which makes it nearly impossible to utilize as a computer output. A reperforator (abbreviation of receiving perforator) can receive from both keyboard and computer and would obviously be a better deal.

Your survey of equipment left out the Model 37 which is a real gem and might someday fall into the hands of computer hobbyists. It is ASCII oriented, runs at 150 wpm (50% faster than the 33/38), has software tab stop sets, half line spacing forward and reverse, a full 255 graphic character printing set, etc. A most remarkable machine.

An important addition to the model 38 description - it has upper and lower case printing capabilities. It also has two color ribbon capabilities.

There are a plethora of modification kits from Teletype that allow you to do special tricks on the various models. Be sure to investigate this too, since some of the things are fantastic!

Be sure to remember that the model 32, 33, and 38 are light duty machines. They are designed for limited use and will work for reasonably long periods but they do require maintenance and adjustment more often than the other models.

An addition to your literature list: RTTY, Box 837, Royal Oak, Michigan, 48068 is an excellent source of information, technical articles and classified ads on TTY and associated equipment. Subs are $3.00/year and it is an excellent magazine. The November issue has the results of a picture tape contest.

I have a large collection of manuals on Teletype, Kleinschmidt, Western Union, etc. digital devices and would be glad to answer any questions your readers might have,

THE COMPUTER HOBBYIST
Founded October, 1974

Stephen C. Stallings-Managing Editor
Hal Chamberlin-Contributing Editor
Jim Parker-Contributor
Jerry Ledbetter-Contributor
Joe Tolbert-Contributor
Edwin Tripp-Photographer
Richard Smith-Programming Consultant

THE COMPUTER HOBBYIST is published monthly near Raleigh, N.C. The yearly subscription rate is $6.00 for persons in the U.S.A., U.S. possessions, and Canada.

All correspondence should be addressed to: THE COMPUTER HOBBYIST
Box 295
Cary, N.C. 27511

Members of the staff of TCH can usually be reached by phone evenings and weekends at either 919/467-3145 or 919/851-7223.

THE COMPUTER HOBBYIST welcomes contributions from our readers. Material to be submitted should be typed or neatly written and must not appear to be soliciting business for any firm. If you wish your material returned, please include a stamped, self-addressed envelope.

but please include a self-addressed stamped envelope. I also have a number of minicomputers at my office and can exchange software on paper tape for various machines.

My company reports on the Government Surplus Sales from the Department of Defense and many times there is digital equipment available to the public. If any of your readers would like further information on buying equipment from the government, they can write to: DOD SURPLUS SALES, Box 1370, Battle Creek, Mich., 49016. Be sure to request a bidder's list application. We have picked up some nice items pretty cheap. You must have patience, however, because sometimes months will go by with nothing but clothing and tent stakes up for bid. We have all the records of past sales and bidders if anyone would like to check on anything.

Fred Hatfield K8VDU
Computer Data Systems, Inc.
1372 Grandview Ave.
Columbus, Oh. 43212
614/486-0677

Thanks for the info and a great letter. Say, how would you like to write for TCH?

Gentlemen:

Thank you very much for the first two issues of "The Computer Hobbyist", which you recently sent me. You're off to a fine start, and I hope these will be the forerunner of many issues to come. I'm enclosing my check for a year's subscription.

I'm pleased to see you have not overlooked us beginners with your article on Logic Symbol Conventions. Although I took a course on Digital Techniques at the University of Maine this past year, your presentation still helped to improve my understanding of the manipulation of logic symbols. Perhaps my appreciation of the article has a bit of malice in it in that I believe you goofed in your answer to your logic design problem. Here's why: With a little manipulation your function

$$\overline{F1}=(\overline{A}+\overline{B})\cdot\overline{C}\cdot\overline{D}$$

becomes

$$\overline{F1}=\overline{\overline{\overline{\overline{A}}+\overline{\overline{B}}}+\overline{\overline{\overline{C}}\cdot\overline{\overline{D}}}} = \overline{A\cdot B+C+D}$$

Likewise

$$\overline{F2}=\overline{I}\cdot J+J\cdot\overline{k}$$

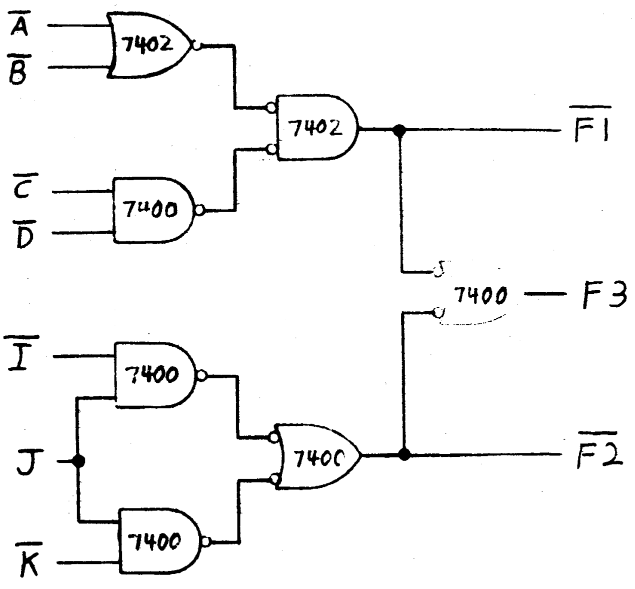
becomes

$$\overline{F2}=\overline{(I+\overline{J})\cdot(\overline{J}+K)}$$

However your logic diagram gives us

$$\overline{F1}=\overline{(A+B)\cdot C\cdot D}$$
$$\overline{F2}=\overline{(I\cdot J)+(J\cdot K)}$$

which are not the same as your starting functions, as a comaprison of truth table will show. I come up with the following solution for inputs A, B, C, D, I, J, K. This is efficient in requiring only seven gates, however five of them are NAND's. At the moment I don't see any way to implement the given functions with only a single 7400 and a 7402.



Norman F. Stanley

You are correct. The given problem and solution was a comedy of errors. The first error was that a list of available input variables was not given. The true form of A through K was intended. Then when the solution was drawn, variables A, B, C, D, I, and K were erroneously supplied in their complemented form. However, even if these corrections are made, the circuit generates F1 and F2 in their TRUE form rather than their complemented form as requested. This then invalidates the method of obtaining F3 and the problem cannot be solved using only two packages of common gates. Your solution is as good as any given the vague statement of the problem, and certainly indicates an understanding of logic symbol manipulation.

A GRAPHICS DISPLAY FOR THE 8008 Part 3 by Hal Chamberlin

In order to get the full usefullness of the graphics display system presented in the last two issues, a large screen X-Y monitor is needed. This article will discuss large screen CRT display technology in general and circuits for a monitor matched to the display generator given in part 2 in particular. The pictures in the centerfold of this issue were photographed directly from the monitor to be described with a 35mm SLR camera. In all cases the program for generating the display was less than 1024 bytes but the applications suggested obviously would require more memory. (See elsewhere in this issue an article describing the chessboard display program.)

Before continuing, the reasons why TV sets are unsuited for computer graphics display should be mentioned. The scanning circuits in a TV receiver are designed to operate at particular frequencies and waveforms only and hence cannot be used for deflection amplifiers in an X-Y graphics display. Alteration of TV scanning circuits is generally out also because their clever design utilizes distributed capacitance, leakage inductance, and other circuit "strays" in developing the scanning waveforms. In order to match the resolution of this X-Y display system with a raster scan display that is TV compatible would require the use of a 512 by 512 dot matrix. Since the 8008 cannot approach the dot-every-120 NS speed required to keep up with the display, a refresh buffer memory is needed. A buffer for 512 by 512 dots would have 262,144 bits or 32K bytes, twice the memory addressable by the 8008. Software for setting bits in the buffer according to lines in a display list can get slow but would be tolerable. An advantage of the dot matrix approach however is the ability to shade in a solid area of the image. It is interesting to note that many commercial systems for raster scan of computer graphics utilize a scan converter storage tube which converts X-Y computer input into TV compatible output.

Cathode ray tubes are of two basic types, electrostatically deflected, and magnetically deflected. The beam in an electrostatic tube is deflected by an electric field between a pair of deflection plates in the tube. Its primary advantage is the ease with which high-speed deflection amplifiers may be built. This is due to the very small amount of energy necessary to change the deflecting field strength between the plates. A typical value is 0.2 microjoules for a full screen sweep. Because of electron optics and physical laws, the beam cannot be deflected very much without severe defocusing. This translates into a large length-to-screen-size ratio and hence small screens for reasonable length tubes. The largest common size is 7 inches but they have been made as large as 19 inches which means the overall length was about 33 inches. A recent development called an expansion mesh now allows effective deflection angles up to 70 degrees peak-to-peak and thus tube proportions roughly that of TV picture tubes in the early '50s. These tubes are extremely expensive (over $1000) however, and while the focus is adequate, it still is not as good as most magnetic tubes.

Magnetically deflected tubes are much more common and can be cheaply made in large sizes. Deflection angles up to 114 degrees peak-to-peak are possible but the upper limit for precision performance is 50 to 70 degrees. Their primary disadvantage is a large deflection energy requirement which results in high power requirements at high frequencies. The deflection energy for the display to be described is about 625 microjoules or a factor of 3,125 higher than the typical O'scope tube.

A magnetically deflected tube may have either magnetic or electrostatic focus. The former usually has a better center-of-screen focus than the latter but quickly defocuses near the edge unless dynamic focus correction is applied. It also requires a focus coil or focus magnet with the former allowing front panel adjustment of focus. An electrostatic focus tube has an internal focus electrode which is connected to a variable focus voltage. Better uniformity of focus over the entire screen is usually obtained without dynamic correction in tubes of this type and thus is recommended for the display monitor.

Some older tubes require an ion-trap magnet near the base of the tube. These can be recognized by examining their gun structure. If the gun elements are not all concentric with a line from the center of the screen to the center of the base or do not have perfect radial symmetry, an ion-trap magnet will be needed.

The type number of most CRT's ends in the letter P followed by a one or two digit number. This is the phosphor type which determines the color and persistance of the screen. Long persistance phosphors especially suited for this display are P7, P14, P19, and P39. The first three are yellow-orange in appearance and P7 requires an amber filter to screen out its short persistance blue component. P39 is a new, very desirable type that is green in color and used in the newer commercial displays. Common short persistance phosphors that will work but provide little flicker filtering are P1, P4, and P31. The first and last of these are green while P4 is white and used in all television picture tubes. P4 may be improved somewhat with an amber filter. Avoid P11 and P16 which are blue or purple in color and of extremely short persistance.

The best CRT to use in building the display is a 10" or 12" round faced radar tube. These are readily available surplus for $20 or less, have a 52 degree deflection angle, and utilize long persistance phosphors. The screen size is large enough to fully utilize the system resolution but small enough to be convenient. The appendix lists some tube types that will work satisfactorily in the monitor.

The most important component of a magnetic deflection monitor is the deflection yoke which slips onto the tube neck and is pushed up against the bulb. The two primary parameters of the yoke are the winding inductance and the maximum deflection angle without the beam hitting the tube neck. In addition there are a number of construction methods for the magnetic core. Older television sets invariably used the saddle construction in which preformed coils are positioned like saddles inside a ring-shaped core. Newer sets use toroidal construction for the vertical axis in which two seperate coils are wound toroidally but are connected so that their fields oppose. This causes the field to jump accross the diameter of the core rather than circulate around it. The construction used for precision yokes is called stator construction because the core and windings are just like those found in the stator of larger AC motors.

Precision performance of deflection yokes depends on careful distribution of coil turns so that the magnetic field lines passing through the tube neck are straight, parallel, and uniformly spaced. Non-uniform deflecting fields result in geometric distortion of displayed patterns and astigmatic defocusing of the beam near the edges. Attaining the correct distribution by hand is easiest with stator construction because it is necessary only to get the proper number of conductors in each slot. TV manufacturers have not used this method however because of its higher cost and because the scanning circuits were too poor to take advantage of the greater precision possible.

In an X-Y monitor of this type the horizontal and vertical coils should be identical. Also low inductance coils are required for wide deflection system bandwidth. These requirements are not even approached by replacement yokes for TV sets. For this reason, the author, in cooperation with TCH, will make deflection yokes meeting the specifications in the monitor schematic available. See the appendix for details.

The schematic for one of the two identical deflection amplifiers is shown in Fig. 1. Basically the circuit is a power op-amp with current feedback so that yoke current is proportional to input voltage in spite of the yoke inductance. A 748 with light compensation is used for voltage gain and a two stage complementary symmetry transistor amplifier is used for current gain. The overall amplifier can provide up to + or - 6 amps of current continuously into a short circuit (the yoke is essentially a short at low frequencies), slew 10 amps in the yoke load in about 20 microseconds (fast full screen jump), accurately follow ramps up to .35 amp per microsecond (fast drawing), and respond to small signals up to 140 kHz (fast character drawing without character yoke). In addition the step response is smooth and essentially free of overshoot. Minimal compensation on the 748 is possible because at high frequencies the loop gain is small due to the high impedance of the yoke. A damping resistor is placed across the yoke for critical damping of the resonant circuit formed by the yoke inductance and distributed capacitance.

Each amplifier should be constructed inside a large, seperate heat sink at least 4 inches wide, 9 inches long, and 2 inches thick with fins. The 4 power transistors should be mounted with thin mica or mylar washers and white thermal grease onto the polished heat sink surface. The two large transistors are especially critical because at times one of them may be dissipating close to 100 watts. The 748 and associated components may be placed on a small piece of vectorboard mounted on short standoffs in the heat sink. The four bias diodes in the output stage should be glued directly to the center of the heat sink for thermal compensation. Each heat sink should be connected to ground and the bypass capacitors shown in the schematic connected between the indicated points and heat sink ground. When complete the two amplifiers may be bolted together with short standoffs and cooled with a whisper fan mounted on the end of the assembly so that air is blown parallel with the fins along their entire length.

The power supply in Fig. 2 is a conventional center-tapped bridge providing nominally + and - 15 volts under load. Each side is capable of supplying up to 10 amps continuously provided the other side is lightly loaded. The zener regulators prevent too much voltage from reaching the 748's when the supply is lightly loaded. Optimum performance is achieved with the 28 volt transformer specified although 24 volts will work with a small reduction in amplifier performance. The primary voltage of high quality 24 volt transformers can be boosted 18 volts with small filament transformers to provide 28 volts at the secondary. Don't use this though if the transformer gets hot with no load or if a significant external magnetic field can be felt with a screwdriver. The filter capacitors should not be skimped on or the ripple may be too much for the amplifiers to reject. Using two or three smaller capacitors in parallel is generally preferable to using a single large capacitor due to high ripple current under full load. The diodes should be of at least 15 amp rating or a 25 amp bridge can be used. Ten amp bridges seem to develop opens after a couple hundred hours in this circuit.

The Z-axis amplifier and CRT biasing circuits are shown in Fig. 4. The voltage applied to the first anode (G2) of the CRT is approximately 500 volts which is somewhat higher than usual. Although the control grid sensitivity is reduced, the focus is materially improved due to a smaller beam diameter in the neck region. There is 500 volts accross the focus control so a high quality molded control is advisable. The case of the focus control should be grounded to protect the user from arc-over should it occur. If a magnetic focus tube is used, the circuit in Fig. 5 will work with most focus coils from old TV's if their DC resistance is between 20 and 300 ohms. The 10KV power supply for the second anode is about right for tube sizes from 10" to 17". A 21" tube should be given about 15 KV for optimum focus and brilliance. The higher voltage reduces deflection sensitivity and as a result the overall picture size may be only slightly larger than that obtained on a 17" tube. It is possible however to beef up the deflection amplifiers and power supply to compensate if desired.

The Z axis amplifier uses a cascode circuit. This configuration has an inherently wide bandwidth because the collector-to-base capacitance of the output stage does not feed back into the input. The 2N3904 input transistor provides the current gain and the MPSU04 provides the voltage gain and high breakdown voltage capability. The contrast control determines the amount of local feedback in the input stage and hence its transconductance. The brightness control operates by injecting a variable bias current into the output stage. The input impedance is several K ohms and the circuit is fully DC coupled. As shown, response is flat to about 4 mHz but could be extended to over 20 mHz with minor modification. The voltage gain with full contrast is about -30 allowing a TTL level signal to fully modulate the beam. The Z axis amplifier should be mounted close to the CRT socket to avoid stray capacitance from the cathode lead which would reduce the bandwidth. Also, the leads to the brightness and contrast controls should be kept short. The filter capacitor for the Z-axis amplifier has been made large to prevent a bright spot on the CRT when power is turned off. It maintains cutoff potential between the control grid and the cathode until the filament has cooled. The 5 volt logic supply should shut down quickly however in order for this scheme to be effective.

The CRT should be mounted securely around the faceplate so that it will stay put with no additional support around the neck. The round radar tubes can be pressed into a round hole cut in soft, 1/2 inch fiberboard that is a teensy bit smaller than the maximum tube diameter. Shims may be needed inside the yoke to keep it snug against the neck and concentric around it. Additional support under the yoke will be needed to prevent stress on the front mounting when the unit is moved. If a TV picture tube is used, it should be fairly easy to salvage the mounting hardware from an old set that used that size tube.

This concludes the series on the graphics display. Builders are encouraged to send in photographs of their displays and descriptions of the software used.

APPENDIX

1. Some CRT's suitable for display

TYPE	FOCUS	SIZE	SHAPE	ANGLE	
7ABP	Elec	7"	Round	50	
7BP	Mag	7"	Round	50	
10FP	Mag	10"	Round	50	
10KP	Mag	10"	Round	52	
12ABP	Elec	12"	Round	52	
12AGP	Elec	12"	Round	42	*
12KP	Mag	12"	Round	54	
12DP	?	12"	Round	52	
12SP	Mag	12"	Round	52	
14EP	Mag	14"	Rect	70	
14QP	Elec	14"	Rect	70	
16AP	Mag	16"	Round	53	*
16RP	Mag	16"	Rect	70	
16STP	Elec	16"	Round	53	
16TP	Mag	16"	Rect	70	
16WP	Mag	16"	Rect	70	
17BP	Mag	17"	Rect	70	
17Cp	Mag	17"	Rect	70	*
17GP	Elec	17"	Rect	70	*
17HP	Elec	17"	Rect	70	
17LP	Elec	17"	Rect	70	
17QP	Mag	17"	Rect	70	
17TP	Elec	17"	Rect	70	*

* These are metal shell tubes. The majority of the tube envelope will be exposed high voltage. An insulated mounting is required.

NOTES: The Screen will not be fully utilized on 70 degree tubes. The letter A or B following the phosphor number indicates aluminized screen, very desirable.

CORRECTION: The + and - input pin connections on the 748 op-amps in the vector generator in last issue are reversed. If you have a January, 1974 copy of the National Semiconductor Linear Integrated Circuits data book, open it to page 2-179 and make a similar correction.

2. Sources of major components

7BP7A ($9.95), 12DP7 ($15), 12SP7 ($15), 16STP4A ($10) See Surplus Summary in this issue.

Power Transformer #28-4 ($20.22) Signal Transformer Co., 1 Junius St., Brooklyn, N.Y. 11212 Check also with Delta Electronics, Box 1, Lynn, Mass 01903 for possible substitution.

Transistors, Circuit Specialists Co., Box 3047, Scottsdale, AZ 85257

10KV Power Supplies (approx $10) B&F Enterprises, Box 44, Hathorne, Mass 01937. Also Meshna, Box 62, East Lynn, Mass 01904.

Deflection yoke ($15) Hal Chamberlin, Box 5985, Raleigh, NC 27607, Ph. 919/851-7225 Write or call for details. CELCO, 78G Constantine Dr., Mahwah, NJ 07430; and Syntronic Instruments Inc., 100G Industrial Rd., Addison, IL 60101 are commercial suppliers of precision and custom deflection yokes.

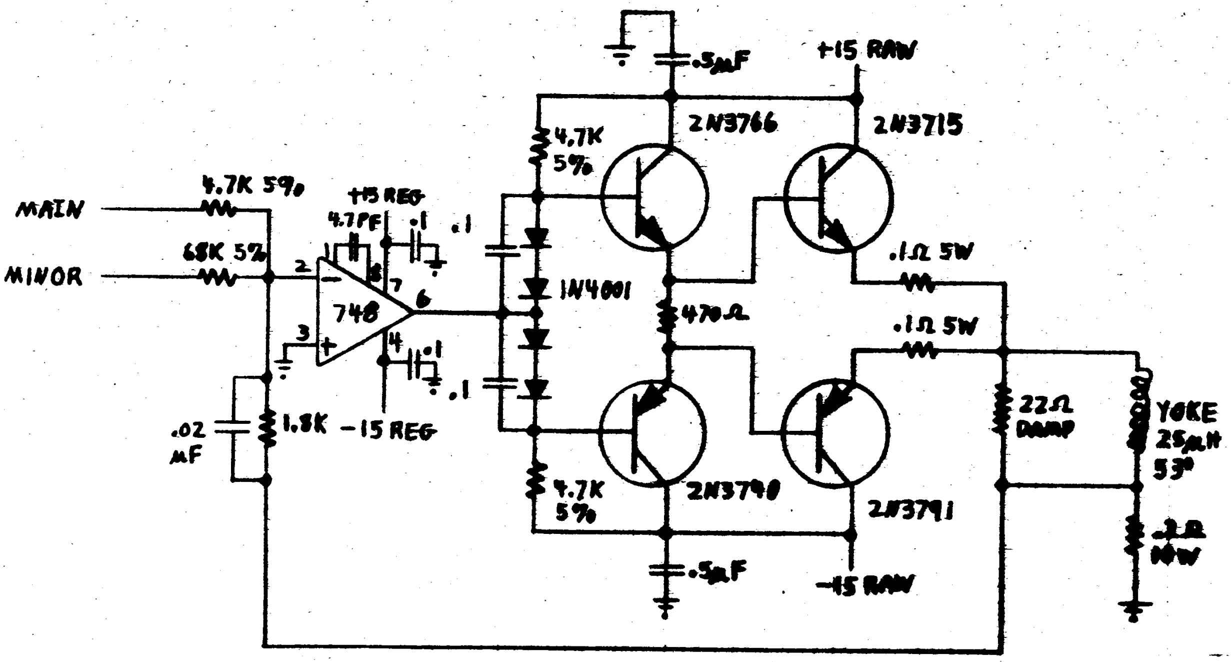


Fig. 1 Deflection Amplifier

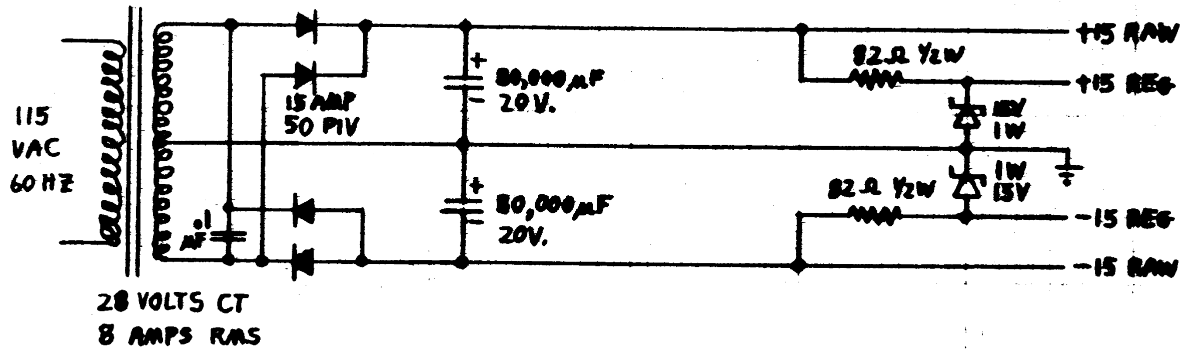


Fig. 2 Deflection Amplifier Power Supply

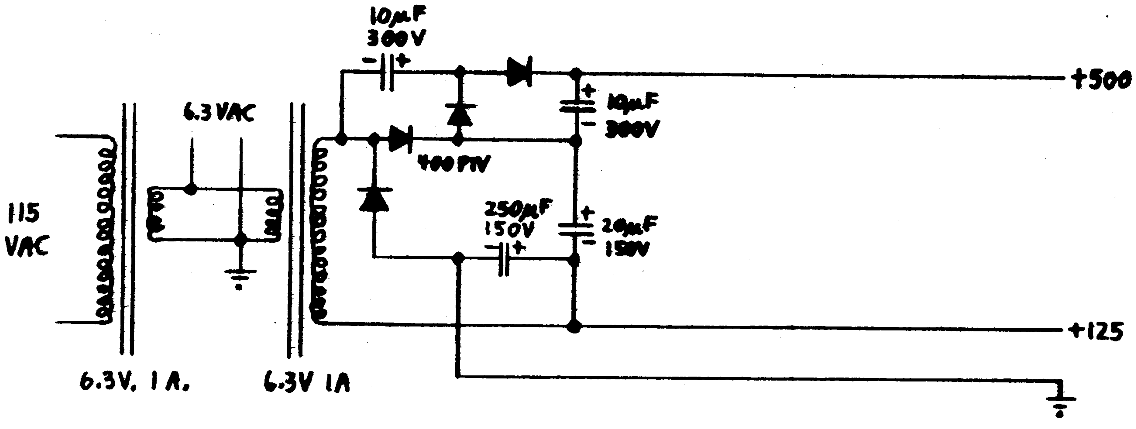


Fig. 3 CRT and Z Axis power Supply

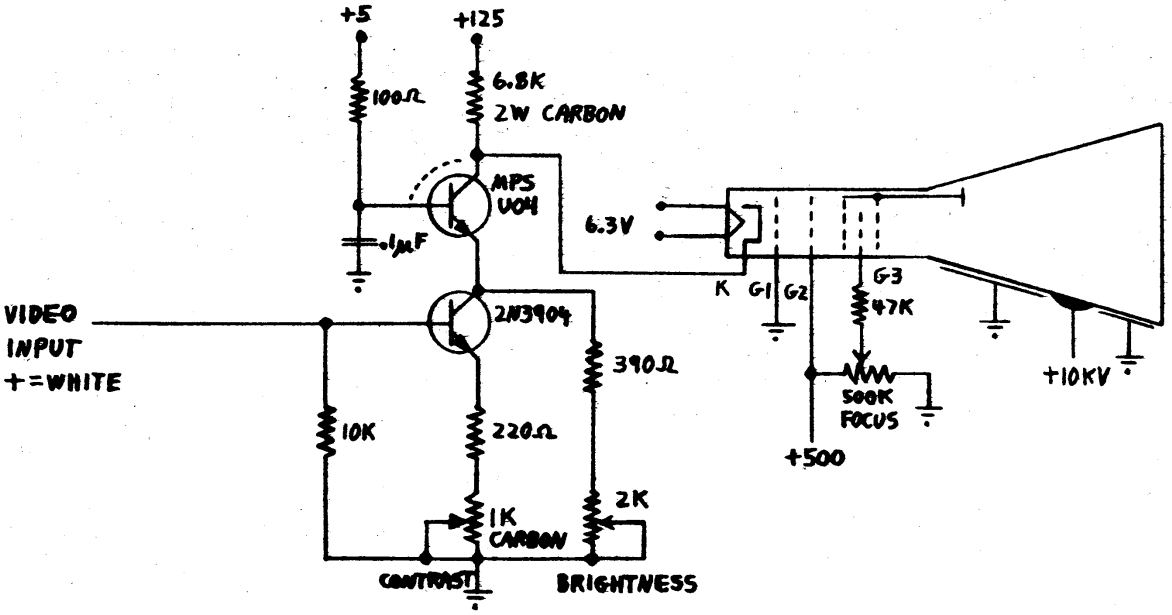
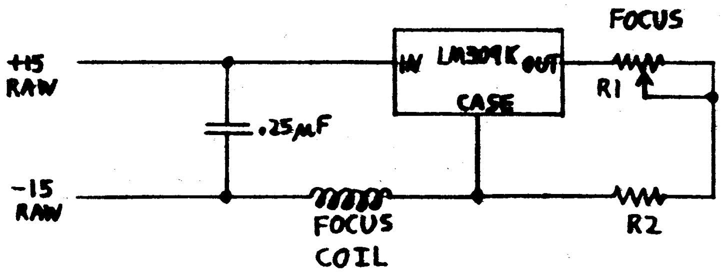
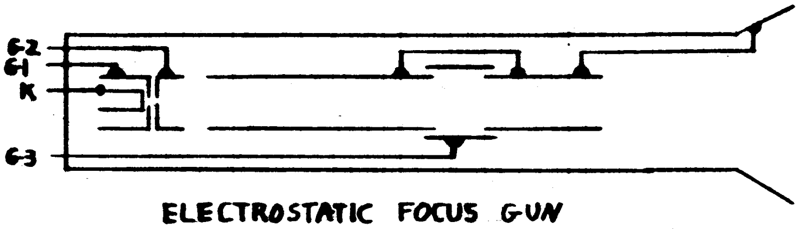


Fig. 4 CRT Biasing and Z Axis Amplifier



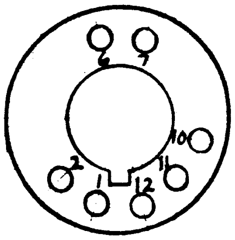
SELECT R1 AND R2
FOR 5 VOLT DROP WITH
NORMAL FOCUS CURRENT
THROUGH COIL. SELECT
R1 FOR ±30% RANGE.

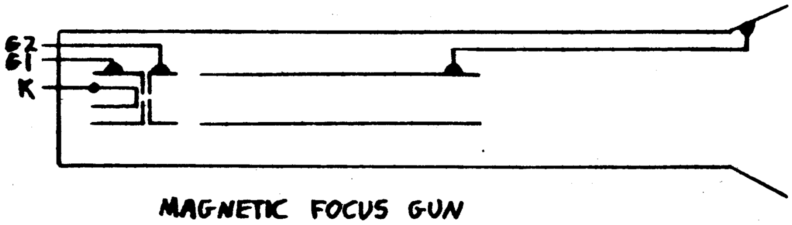
Fig. 5 Regulator for Magnetic Focus Coil



ELECTROSTATIC FOCUS GUN

BOTTOM VIEW





MAGNETIC FOCUS GUN

TYPICAL CRT BASE

PIN	FUNCTION
1	FILAMENT
2	G1 CONTROL GRID
6	G3 FOCUS ELECTRODE
7	OFTEN MISSING
10	G2 FIRST ANODE
11	K CATHODE
12	FILAMENT

BOOK REVIEW BY Hal Chamberlin

Microcomputer Design, Martin, Donald P., Martin Research LTD., 1825 S. Halsted St., Chicago, IL 60608

The industrial world, like the hobbyist world, has been caught relatively off-guard by the rapid introduction of microprocessor chips. The problem has not been a lack of desire to put the new devices to use but a lack of knowledge and the failure of traditional educational means in this interdisciplinary area. This book is an attempt to provide an understandable how-to-do-it reference manual on building microcomputers from microprocessor chips. The overwhelming emphasis is on the 8008 microprocessor but the 8080 is mentioned occasionally.

The book is sold in a manner that makes its price hard to pin down. For $100 Martin Research will supply the 300 pages looseleaf bound in two notebooks, a Microsystems International 8008 manual, and an 8008 chip. The chip supplied is tested for speed at 70 degrees C case temperature and packed in an individual envelope with the maximum speed stated to within a tenth microsecond for state time. The one supplied with this copy ran at 2.5 microseconds which is 8008-1 speed. For an additional $10, the 8008 will be replaced by an 8008-1 which is also tested for speed. With surplus 8008 prices running $50 to $80 and the difficulty of getting the manufacturer's manuals, the package seems to be a pretty good deal. The printing is of high quality but is difficult to read because it is on blood-red woven texture paper and done with a 10-pitch type size compressed to 12-pitch thus crowding the characters.

The leadoff chapters provide a tutorial explanation of sequential machines and basic computer concepts assuming only a knowledge of TTL logic design. Later the need for and design of the external logic necessary to make the 8008 function is developed. The remainder of the book is devoted to special topics such as I/O techniques, interrupts, direct memory access, expanding 8008 capabilities, and three complete microcomputer designs. One of the ideas that was new to us at TCH was the possibility of detecting the register self-load instructions (LBB, LCC, etc.) and picking off the register contents from the CPU bus at the proper time and latching them externally. Another was a method of clearing interrupt levels in a multilevel interrupt system using other valid variations of the RET instruction. Software discussion was limited to an interrupt save-restore routine and a few common programming techniques required for effective use of the microcomputer.

This book is directed mainly to industrial users of microcomputers who wish to use them as dedicated controllers or random logic replacements. As a result, many of the simplifying techniques presented may be of limited value in systems intended for general purpose use such as a hobbyist would require. Also, a large portion of the apparent simplicity of the designs presented is due to the extensive use of just-released, multifunction, low power Schottky MSI IC's. While industrial buyers may have little difficulty in obtaining these at reasonable prices, it may be some time before they are available mail order to hobbyists.

One possible objection that hard-nosed engineers might have is the lack of correlation between information stated or developed in the book and parameters stated by the manufacturers in their data sheets. A prime example is the allowable CPU bus loading which is specified as .44 mA maximum but is allowed by the author to reach 1.40 mA. Another is the use of a symmetrical clock waveform in violation of the manufacturer's recommended waveform (also done in the MARK-8 design). The author seems to be quite competent however and claims a detailed knowledge of the inner workings of the 8008 chip. The hobbyist should not experience any difficulty with these violations provided the microcomputer is kept at room temperature and the supply voltages are kept tightly regulated.

In summary, Microcomputer Design is a veritable computer-education-in-a-notebook and well worth considering.

SUBSCRIBER LIST

Each month TCH will publish the names and addresses of new subscribers who so desire. however the list will be limited to one page per issue to conserve space. We hope this service will aid people in finding assistance and friends with a common interest.

Roger W. Amidon K2SMN
Box 317
Skillman, NJ 08558
Ph. 201/359-4842

Jim Avery
Route 2 Box 13-A
Leesville, SC 29070

W. H. Burtner
RR2 Box 267
Valparaiso, Ind. 46383

James Bush
1201 NW 39 Ave.
Gainesville, FL 32609

Gary C. Chatters
2737 Hermina St.
Madison, Wis. 53704
Ph. 608/249-5560

Gary Coleman
Box 530, 11900 Carlton Rd.
Cleveland, Oh. 44106
Ph. 216/795-1317

Douglas B. Dyre
3202 Winchester Drive
Greensboro, NC 27406
Ph. 919/275-2459

Bill Dumke
Rt. 8, Box 496
Charlottesville, VA. 22901
Ph. 804/973-2894
804/296-0335

Edward C. Epp
Swan Lake Christian Camp
Viborg, SD 57070

James H Ewertz
908 Vernon St.
Garner, NC 27529
Ph. 919/779-0471

LTC F. H. Faulkner
58 Offutt Rd.
Bedford, MA. 01730

George Fischer
72 South Railroad Ave.
Staten Island, NY 10305

R. Story Forman
2421 NW Johnson
Portland, OR. 97210
Ph. 503/223-3478

Maury Goldberg
1618 James ST.
Syracuse, NY 13203
Ph. 315/479-7620

Dr. George L. Haller
1500 Galleon Dr.
Naples, FL 33940
Ph. 813/649-5007

Joseph A. Handley, Jr. K3LLL
10804 Joyceton Ct. Kettering
Upper Marlboro, MD. 20870
Ph. 301/336-9072

James C. Hart, Jr.
1841 Edgeland
Louisville, KY. 40204
Ph. 502/451-5138

Fred Hatfield
Box 27100
Columbus OH. 43212
Ph. 614/486-0677

Michael N. Hayes, MNH-AE
P.O. Box 1208
Landover, MD. 20785
Ph. 301/773-0895

LTC Donald R. Kelley
Facility Engineer
Fort Lesley J. McNair
Washington, D.C. 20319
Ph. 202/488-8055

Robert W. Kelley
5806 MT Terminal Dr.
Waco, TX. 76710
Ph. 817/772-7859

George Kowalski
2442 Sonoma Dr.
Colo. Springs, CO. 80910
Ph. 303/471-7635

Ed Lankford
511 Purnell Drive
Nashville, TN. 37211
Ph. 615/832-3552

Emerson Lawson
Jefferson Productions
One Julian Price Place
Charlotte, NC 28208
Ph. 704/374-3823

Philip Lohr
3917 Flowerfield Rd.
Charlotte, NC 28210
Ph. 704/364-8913

Micheal Pique
Dept. of Computer Science
New West Hall, Univ of NC
Chapel Hill, NC 27514
Ph. 919/933-5698

Larry Pleskac
938 Paula Street
Escondido, CA. 92027
Ph. 714/747-0274

Lester McFawn
2063 Triumph Dr.
Fairborn, OH. 45324

Terry F. Ritter, VP DANTCO
2524B Glen Springs Way
Austin, TX. 78741
Ph. 512/441-0036

Jeff Roberts
R. D. 2, Box 2075
Mount Holly, NJ 08060
Ph. 609/267-1403

Scelbi Computer Consulting
1322 Rear-Boston Post Rd.
Milford, CT. 06460
Ph. 203/874-1573

Brett Sher
Apt. 2701 330 W. Diversey
Chicago, IL 60657
Ph. 312/929-3228

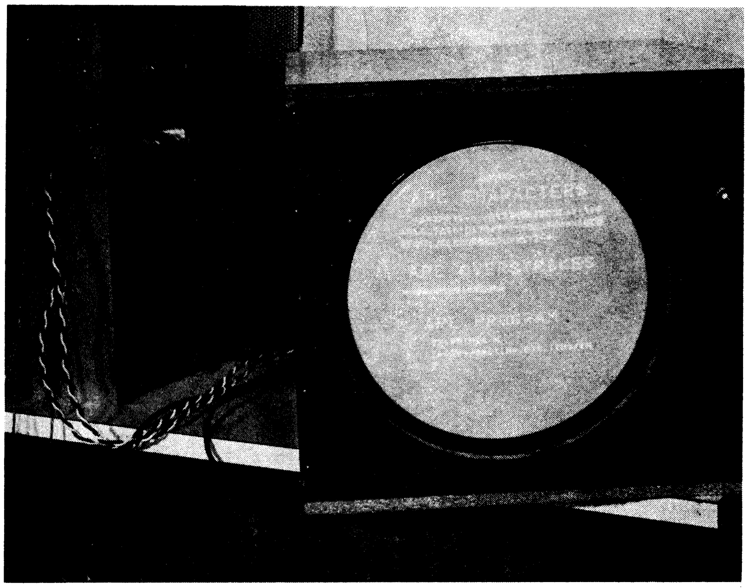
H. A. Singleton
1104 Parkmont Lane
Rockhill, SC 29730
Ph. 803/327-6670

Kenneth D. Slezak
652 South Sheraton Drive
Akron, OH 44319
Ph. 216/644-9619

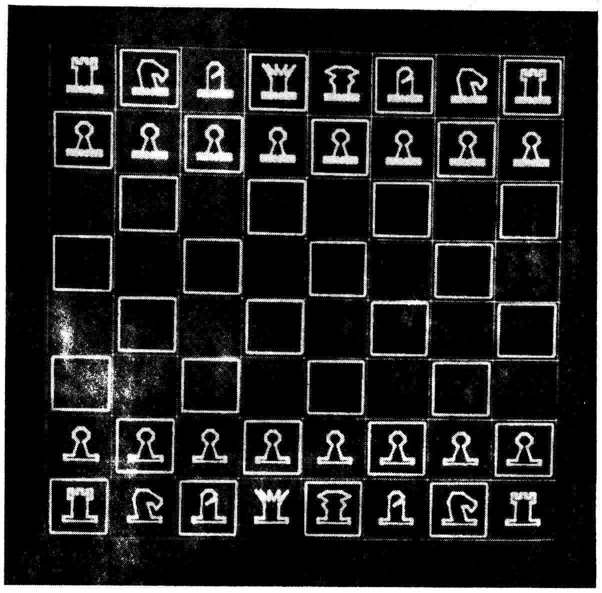
R. E. Smallwood
20-12 St. N. W.
Calgary, Alberta
Canada T2N 1Y3

Timothy Smith
475 Saint John's Street
Central Islip, NY 11722
Ph. 516/234-5245

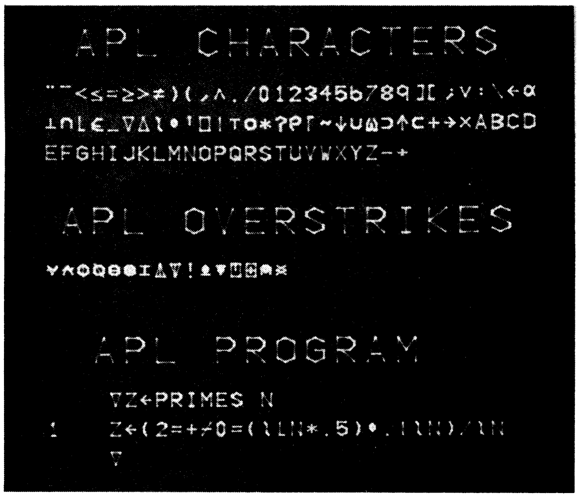
Sample Photographs of What TCH Has Done and
What You Can Do With a Graphics Display



TCH's Graphics Display System



Chess Board Display
See article on page 14.



APL CHARACTERS

¨¯<≤=≥>≠)(,∧./0123456789][;∨:\←α
⊥∩⌊∊_∇∆⍳∘'⎕|⊤○*?⍴⌈~↓∪⍵⊃↑⊂+→×ABCD
EFGHIJKLMNOPQRSTUVWXYZ-÷

APL OVERSTRIKES

⍱⍲⌽⍉⊖⍟⌶⍋⍒!⍎⍕⍞⌹⍝⍫

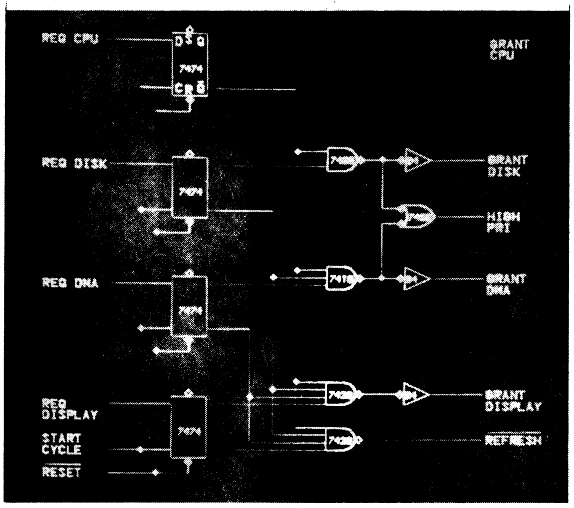
APL PROGRAM

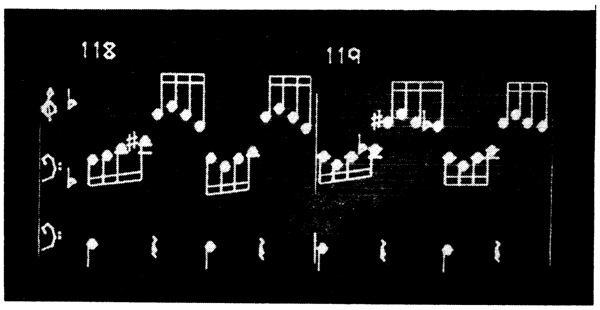
```
    ∇Z←PRIMES N
1   Z←(2=+⌿0=(⍳N*.5)∘.|⍳N)/⍳N
    ∇
```

APL Character Set

Photos by Edwin Tripp

Logic Diagram





Music Notation

CHESS BOARD DISPLAY by Jim Parker

A chessboard display is quite effective in demonstrating the versatility of one's computer system. The program discussed in this article displays a chessboard in the opening position using an 8008 microprocessor with at least 512 bytes of memory and the vector graphics display system presented in the first three issues of TCH. Although this program does not allow movement of the pieces by keyboard entry, such a program could be adapted since changing a single data table allows any chess position to be displayed. Two full pages of memory are required although they need not be consecutive. A discussion of the program itself follows. The author hopes that the ideas shown may help other programmers in coding similar display programs.

The program may be divided into two independent parts. One part uses the main deflection system to draw the checkerboard graphically and the other part uses the minor deflection system or character generator to draw the chessmen. The minor system saves on memory and refresh time. Subroutines are kept fairly general so that most of the work is specified by the four data tables. Taking a closer look, one should begin with the graphics routines. Note that the 8 X 8 grid is represented with octal coordinates as shown in Figure 1. The decision was made to specify all coordinates as positive since this simplifies keypunching data cards. Dark squares are indicated by alternately inscribing smaller squares on the board.

GRAPH is the fundamental graphics routine. It is a more general form of the DRAW routine discussed in the first issue of TCH. One provides it with a list of XY pairs, such as HORZ or CHK, headed by the number of pairs in that list. 200B is a reserved coordinate which instructs the routine to jump to the following XY pair without drawing a line. Thus 3 disjoint horizontal lines are drawn by the list HORZ. Registers DE are set before calling GRAPH to any desired XY displacement. Thus the nine horizontal lines are drawn by calling GRAPH to draw HORZ 3 times; each time with a different vertical displacement.

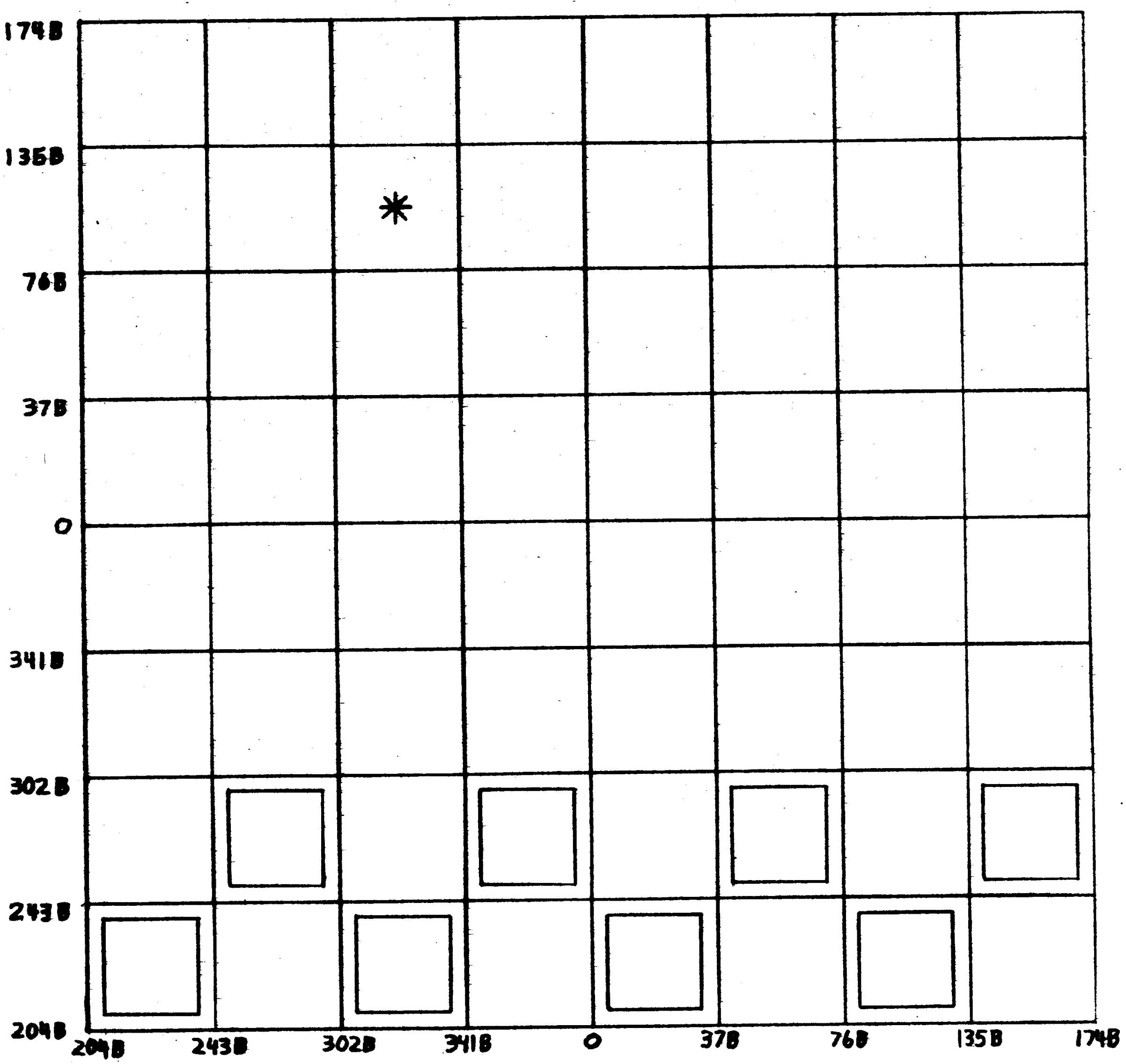
EDGRPH is the graphics edit routine. It is entered once to draw the entire board. One enters with HL set to GDISPT, the graphics displacement table. This table lists corresponding graph sets and displacements. Hence, the checks cover the board by calling GRAPH to draw a row of checks (as in Figure 1.) eight times. Each time the row is moved up and left or right one square. 377B is a reserved address which flags the end of the table. It is necessary that GDISPT and the graph sets be in the same page of memory since only the low order address of the graph sets are specified in GDISPT. Note that EDGRPH could be called additional times with other GDISPT tables and graph sets in order to draw the chess pieces but the inefficiency could cause more flicker and moving chess pieces on the board would be difficult unless the routine was modified.

CHAR is concerned with the second part of the program; drawing the chessmen. One enters the character routine with HL set to the address of the stroke list defining the chess piece or black base. A negative number in the list flags the end as described in the first issue of TCH. Expanding the loop into straight code to remove the inefficient jump instruction makes the called routine about 40% faster. However, the observable difference may not be worth the extra memory used.

EDCHAR is the character edit routine which determines the placement and color of each chess piece. The associated data table, CDISPT, lists a displacement and character address in that order for each chess piece. Since only the low order address is specified, CDISPT and the character set must be in the same page of memory. This subroutine is somewhat specialized in that it expects all character addresses to be in the lower half of a memory page. A low address greater than 200B tells the routine that a black chess piece is specified whose address is actually 200B less than indicated in the CDISPT table. BK must be aligned on the page boundary as EDCHAR addresses it absolutely. The chess pieces may be moved around by rearranging the entries in

CDISPT. Note the standard white side chess notation associated with each XY displacement. Squares not listed may have their displacement computed by adding (10B, 10B) to the lower left hand coordinates of any squares shown in Figure 1. If the pieces are not centered, one may have to adjust MINSIZ or reposition the chess piece by changing the displacement. If the list is shortened, the first entry in CDISPT must be decreased to indicate the number of chessmen in the list.

That covers the main ideas in the program. Readers are encouraged to develop programs that use this program to display a chess game between two players who enter their moves by keyboard. A more compatible notation may need to be developed. Other possibilities include a program that understands the game rules and flags checks and illegal moves. As a final challenge, the 8008 could play a human in a game where just a King and a Rook try to force a lone King into checkmate. In the future, supplementary programs may be published so please submit ideas.



* CHARACTER DISPLACEMENT FOR THIS SQUARE IS (312B, 106B)

Fig. 1 Chess Board Display Coordinates

```
*       CHESSBOARD DISPLAY

XMOV    EQU 10B
YMOV    EQU 11B
XSTOR   EQU 12B
YDRAW   EQU 13B
MINXY   EQU 14B
MINSZ   EQU 15B

        ORG 30000B

CHESS   LAI 377B        SET MINOR SYSTEM
        OUT MINSZ        FOR MAX DEFLECTION
LOOP    SHL GDISPT      GRAPH EDIT TABLE
        CAL EDGRPH      DRAW BOARD
        SHL CDISPT      CHAR EDIT TABLE
        CAL EDCHAR      DRAW PIECES
        JMP LOOP        REFRESH DISPLAY

*                       ENTER WITH HL SET
*                        TO GDISPT
EDGRPH  LAM             A=GRAPH ADDRESS
        CPI 377B        CHECK FOR END FLAG
        RTZ             RETURN IF SO
        INL             BUMP TABLE INDEX
        LDM             LOAD X DISPLACEMENT
        INL             BUMP TABLE INDEX
        LEM             LOAD Y DISPLACEMENT
        LBL             SAVE TABLE INDEX
        LLA             SET L TO GRAPH
*                        ADDRESS
        CAL GRAPH       DRAW GRAPH
        LLB             RESTORE TABLE INDEX
        INL             BUMP INDEX
        JMP EDGRPH      REPEAT

*       GRAPHICS DRAW SUBROUTINE
*       ENTER WITH XY DISPLACEMENT IN DE
*       ENTER WITH GRAPHICS ADDRESS IN HL

GRAPH   LCM             GET COORD COUNT
MOVE    INL             BUMP TO NEXT BYTE
        LAM             GET X COORD
        ADD             ADD X DISPLACEMENT
        OUT XMOV        MOV X WITH BEAM OFF
        INL             BUMP TO NEXT BYTE
        LAM             GET Y COORD
        ADE             ADD Y DISPLACEMENT
        OUT YMOV        MOV Y WITH BEAM OFF
DRAW    DCC             DECREMENT COUNT
        RTZ              AND RETURN IF DONE
        INL             BUMP TO NEXT BYTE
        LAM             GET X COORD
        CPI 200B        CHECK BEAM OFF FLAG
        JTZ MOVE        JUMP ON FLAG
        ADD             ADD X DISPLACEMENT
        OUT XSTOR       STORE X IN BUFFER
        INL             BUMP TO NEXT BYTE
        LAM             GET Y COORD
        ADE             ADD Y DISPLACEMENT
        OUT YDRAW       DRAW LINE TO (X,Y)
        JMP DRAW        LOOP
```

```
*               ENTER WITH HL SET TO
EDCHAR LBM              GET TABLE C
CTOP    INL             BUMP TABLE
        LAM             LOAD CHAR X
        OUT XMOV        SET X POSIT
        INL             BUMP TABLE
        LAM             LOAD CHAR Y
        OUT YMOV        SET Y POSITI
        XRA             A=0
        INL             BUMP TABLE I
        LDL             SAVE TABLE I
        ADM             LOAD CHAR AD
        JFS CNX         JUMP IF L<20
        XRI 200B        SUBTRACT 200
        LLI 0           BLACK BASE A
        LCA             SAVE A
        CAL CHAR        DRAW BLACK BA
        LAC             RESTORE A
CNX     LLA             LOAD CHAR ADD
        CAL CHAR        DRAW CHAR
        LLD             RESTORE TABLE
        DCB             DECREASE TABL
        JFZ CTOP        GET NEXT CHAR
        RET             RETURN

*       ENTER WITH HL SET TO CHARACT
CHAR    XRA             A=0
        OUT MINXY       SET MINOR TO 0
        DCL             CANCEL NEXT ST
CHAR1   INL             BUMP LIST INDE
        ADM             FETCH STROKE, T
        OUT MINXY       OUTPUT IT
        JFS CHAR1       LOOP IF NO FLAG
        RET             RETURN IF END O
*                        LIST

*       GRAPHICS SET
HORZ    DEF 6           3 HORIZONTAL LIN
        DEF 204B,204B,174B,204B
        DEF 200B
        DEF 204B,243B,174B,243B
        DEF 200B
        DEF 204B,302B,174B,302B
VERT    DEF 6           3 VERTICAL LINES
        DEF 204B,204B,204B,174B
        DEF 200B
        DEF 243B,204B,243B,174B
        DEF 200B
        DEF 302B,204B,302B,174B
CHK     DEF 20          1 ROW OF CHECKS
        DEF 002B,002B,002B,035B
        DEF 035B,035B,035B,002B
        DEF 002B,002B
        DEF 200B
        DEF 100B,002B,100B,035B
        DEF 133B,035B,133B,002B
        DEF 100B,002B
        DEF 200B
        DEF 176B,002B,176B,035B
        DEF 231B,035B,231B,002B
        DEF 176B,002B
        DEF 200B
        DEF 274B,002B,274B,035B
        DEF 327B,035B,327B,002B
        DEF 274B,002B
```

```
*       EDGRPH DATA TABLE
GDISPT  DEF L(HORZ),000B,000B
        DEF L(HORZ),000B,135B
        DEF L(HORZ),000B,272B
        DEF L(VERT),000B,000B
        DEF L(VERT),135B,000B
        DEF L(VERT),272B,000B
        DEF L(CHK),204B,204B
        DEF L(CHK),243B,243B
        DEF L(CHK),204B,302B
        DEF L(CHK),243B,341B
        DEF L(CHK),204B,000B
        DEF L(CHK),243B,037B
        DEF L(CHK),204B,076B
        DEF L(CHK),243B,135B
        DEF 377B       END OF TABLE FLAG

        ORG 30400B     NEXT PAGE OF MEMORY
*                      CHARACTER SET
BK      DEF 111B       BLACK BASE
        DEF 120B-111B,131B-120B
        DEF 140B-131B,151B-140B
        DEF 160B-151B,171B-160B
        DEF 070B-171B,161B-070B
        DEF 150B-161B,141B-150B
        DEF 130B-141B,121B-130B
        DEF 110B-121B,301B-110B
PN      DEF 101B       PAWN
        DEF 121B-101B,134B-121B
        DEF 125B-134B,126B-125B
        DEF 137B-126B,147B-137B
        DEF 156B-147B,155B-156B
        DEF 144B-155B,151B-144B
        DEF 171B-151B,170B-171B
        DEF 300B-170B
BP      DEF 101B       BISHOP
        DEF 121B-101B,126B-121B
        DEF 137B-126B,147B-137B
        DEF 156B-147B,134B-156B
        DEF 155B-134B,151B-155B
        DEF 171B-151B,170B-171B
        DEF 300B-170B
RK      DEF 101B       ROOK
        DEF 121B-101B,125B-121B
        DEF 115B-125B,117B-115B
        DEF 127B-117B,126B-127B
        DEF 136B-126B,137B-136B
        DEF 147B-137B,146B-147B
        DEF 156B-146B,157B-156B
        DEF 167B-157B,165B-167B
        DEF 155B-165B,151B-155B
        DEF 171B-151B,170B-171B
        DEF 300B-170B
QN      DEF 101B       QUEEN
        DEF 121B-101B,124B-121B
        DEF 107B-124B,125B-107B
        DEF 117B-125B,135B-117B
        DEF 137B-135B,145B-137B
        DEF 157B-145B,155B-157B
        DEF 177B-155B,154B-177B
        DEF 151B-154B,171B-151B
        DEF 170B-171B,300B-170B
NT      DEF 101B       KNIGHT
        DEF 121B-101B,115B-121B
        DEF 127B-115B,147B-127B
        DEF 175B-147B,174B-175B
```

```
        DEF 163B-174B,154B-163B
        DEF 144B-154B,151B-144B
        DEF 171B-151B,170B-171B
        DEF 300B-170B
KG      DEF 101B       KING
        DEF 121B-101B,123B-121B
        DEF 114B-123B,125B-114B
        DEF 106B-125B,127B-106B
        DEF 157B-127B,176B-157B
        DEF 155B-176B,164B-155B
        DEF 153B-151B,151B-153B
        DEF 171B-151B,170B-171B
        DEF 300B-170B

        ORG 30600B     NEXT HALF-PAGE
*                      EDCHAR DATA TABLE
CDISPT  DEF 32
        DEF 214B,253B QR2
        DEF L(PN)      WHITE PAWN
        DEF 253B,253B QN2
        DEF L(PN)      WHITE PAWN
        DEF 312B,253B QB2
        DEF L(PN)      WHITE PAWN
        DEF 351B,253B Q2
        DEF L(PN)      WHITE PAWN
        DEF 010B,253B K2
        DEF 1(PN)      WHITE PAWN
        DEF 047B,253B KB2
        DEF L(PN)      WHITE PAWN
        DEF 106B,253B KN2
        DEF L(PN)      WHITE PAWN
        DEF 145B,253B KR2
        DEF L(PN)      WHITE PAWN
        DEF 214B,106B QR7
        DEF L(PN)+200B BLACK PAWN
        DEF 253B,106B QN7
        DEF L(PN)+200B BLACK PAWN
        DEF 312B,106B QB7
        DEF L(PN)+200B BLACK PAWN
        DEF 351B,106B Q7
        DEF L(PN)+200B BLACK PAWN
        DEF 010B,106B K7
        DEF L(PN)+200B BLACK PAWN
        DEF 047B,106B KB7
        DEF L(PN)+200B BLACK PAWN
        DEF 106B,106B KN7
        DEF L(PN)+200B BLACK PAWN
        DEF 145B,106B KR7
        DEF L(PN)+200B BLACK PAWN
        DEF 214B,214B QR1
        DEF L(RK)      WHITE ROOK
        DEF 145B,214B KR1
        DEF L(RK)      WHITE ROOK
        DEF 214B,145B QR8
        DEF L(RK)+200B BLACK ROOK
        DEF 145B,145B KR8
        DEF L(RK)+200B BLACK ROOK
        DEF 253B,214B QN1
        DEF L(NT)      WHITE KNIGHT
        DEF 106B,214B KN1
        DEF (NT)       WHITE KNIGHT
        DEF 253B,145B QN8
        DEF L(NT)+200B BLACK KNIGHT
        DEF 106B,145B KN8
        DEF L(NT)+200B BLACK KNIGHT
        DEF 312B,214B QB1
DEF L(BP)     WHITE BISHOP
DEF 047B,214B KB1
DEF L(BP)     WHITE BISHOP
DEF 312B,145B QB8
DEF L(BP)+200B BLACK BISHOP
DEF 047B,145B KB8
DEF L(BP)+200B BLACK BISHOP
DEF 351B,214B Q1
DEF L(QN)     WHITE QUEEN
DEF 351B,145B Q8
DEF 1(QN)+200B BLACK QUEEN
DEF 010B,214B K1
DEF L(KG)     WHITE KING
DEF 010B,145B K8
DEF L(KG)+200B BLACK KING

END CHESS
```

TRICKS AND TECHNIQUES

This month's programming technique was submitted by Terry F. Ritter, Vice President, DANTCO, 2524B Glen Springs way, Austin, TX. 78741. Readers are encouraged to submit their own techniques or to comment on others that have been published.

In Vol. 1 No. 1, you invited submission of programming tricks and techniques; this is in response to that invitation, although it is more on the order of a systems technique than a trick.

I know from your examples that most, if not all of you are Assembly Language oriented. I, on the other hand, am quite convinced that assemblers, in general, are the most inefficient languages known, and are far less efficient even than machine language in program development. Let me save the detailed arguments for later, but one advantage that machine-language programming has over assembler is the availability of redundant machine codes. Examination of the 8008 instruction set will show several redundant codes; i.e., the 1X4 JMP, the 1X6 CAL, the 0X7 RET, and the 3XY (X=Y=7) NOP.

The principle problem with any machine code program, whether assembler-produced or directly coded, is that it must be located at a particular position in memory for proper operation. That is, there is no particular problem in loading the program anywhere desired; the problem is in changing the jump addresses for the program to correspond to the new location. This problem is partially solved by use of a relocatable system, but still lingers during program modification.

My approach is to use a particular machine code to indicate: 1) the start of a program, 2) where it was last correctly located in memory, and 3) a two or more byte program ID. Another redundant return code, of course, indicates the end of a particular program, group of programs, or programs and data which are desired to be grouped. The RETURN code is standardized so that an executive may call the program, then resume control after termination.

This approach allows complex keyboard editing of the machine code. A particular Executive command could cause a software system to insert a NOP and bubble up the program(s) until the end-of-program code was reached being careful not to bubble into another program without moving it also. It would then search from the beginning of the program (or all of memory if desired) for jumps pointing to the affected (moved) program area and suitably modify each. A similar command could cause deletion of a machine-code step (something not easily accomplished in certain delays where any NOP would still constitute a machine-cycle delay).

Labelling and structuring of variables also seems possible by using redundant codes, as well as the passing of variables through high-level language subroutine calls.

Example:

```
┌──── 124  Indicates start of pro-
│          gram system.
│     AAA  Identifies valid starting
│     AAA  location.
JUMP  LLL  Program label (ID bytes)
│     LLL
└───→ XXX  Program system
      XXX
       .
      XXX
      027  Indicates end of program
           system
```

INTERFACING A 5 LEVEL TELEPRINTER Part 2 by Steve Stallings

In our last issue we described the hardware needed to interface a five level teleprinter to your 8008 system. This article completes the interface with the needed running and testing software (courtesy of Richard Smith). The program consists of six parts.

The first part, Equates, must be custom tailored to your system. CNTREG is the equate for the output port that the teleprinter is to use. CNTTO designates which bit of the eight data bits is to be used. Setting CNTTO to 001B corresponds to using bit 0 (least significant bit). Other bits in the same port may be used for other purposes, however, one must realize that the teleprinter routine must be modified so that it does not cause problems with the devices using the other bits and vice-versa. TTDY1 specifies the delay count for one bit time. The value shown, 316B, corresponds to the use of an 8008-1 running with a state time of 2.5 microseconds and a 100 word per minute teleprinter. TTDY2 sets the delay time for the fractional stop bit. See the chart in Fig. 1 for the values of TTDY1 and TTDY2 needed for typical systems. PGAR and DTAR tell the assembler what memory areas to use for the program and temporary data storage.

Teletype Demo Program calls the initialization routine and then generates the ripple pattern text. If other patterns are desired, they may be put at MSG in place of the ripple pattern data. This routine works entirely with ASCII.

Teletype Initialization Rountine is used to get the teletype case shift in a known state, and to hold the teletype in a mark state until further data is sent.

Baudot Teletype Output Routine accepts Baudot characters and does the actual timing and transmission of the characters. If you wish to write your other programs to converse in Baudot rather than ASCII, you may call this routine directly.

ASCII Teletype Output Routine For Baudot Machine is a special routine to compensate for the differences between ASCII and Baudot other than character codes. It takes care of the case shifts on characters and suppressing unneeded case shifts for dual case commands (space, CR, LF, and NULL). This routine accepts Baudot with an extra bit to specify case. You may call the routine directly for output if you wish to use the special 6 bit code it requires.

ASCII to Baudot Translate Routine does just what its name says. Its input is ASCII and its output is the special 6 bit code required by the ASCII output routine.

Figure 2 shows a sample of the ripple pattern output produced by all of the above routines used together.

Fig. 1	8008 Standard 4 uS per state 500 kHz clock		8008-1 2.5 uS per state 800 kHz clock	
	TTDY1	TTDY2	TTDY1	TTDY2
60 WPM	322B	132B	Requires program delay loop change	
100 WPM	176B	066B	312B	127B

Fig. 2

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789-?:$!&'().,;/"A1B2C3D4E5F6G7H8I9J0A
BCDEFGHIJKLMNOPQRSTUVWXYZ0123456789-?:$!&'().,;/"A1B2C3D4E5F6G7H8I9J0AB
CDEFGHIJKLMNOPQRSTUVWXYZ0123456789-?:$!&'().,;/"A1B2C3D4E5F6G7H8I9J0ABC
DEFGHIJKLMNOPQRSTUVWXYZ0123456789-?:$!&'().,;/"A1B2C3D4E5F6G7H8I9J0ABCD
EFGHIJKLMNOPQRSTUVWXYZ0123456789-?:$!&'().,;/"A1B2C3D4E5F6G7H8I9J0ABCDE
FGHIJKLMNOPQRSTUVWXYZ0123456789-?:$!&'().,;/"A1B2C3D4E5F6G7H8I9J0ABCDEF
GHIJKLMNOPQRSTUVWXYZ0123456789-?:$!&'().,;/"A1B2C3D4E5F6G7H8I9J0ABCDEFG
HIJKLMNOPQRSTUVWXYZ0123456789-?:$!&'().,;/"A1B2C3D4E5F6G7H8I9J0ABCDEFGH
```

```
*       BAUDOT TELETYPE RIPPLE
*       PATTERN DEMO PROGRAM

*       EQUATES-SEE TEXT FOR DETAILS
CNTREG  EQU 026B         CONTROL REGISTER
CNTTO   EQU 001B         TTY SERIAL OUT BIT
TTDY1   EQU 316B         DELAY CNT FOR 1 BIT
TTDY2   EQU 130B         DELAY CNT FOR .42
*                         BIT
PGAR    EQU 30000B       PROGRAM AREA
DTAR    EQU 31000B       DATA AREA

*       TELETYPE DEMO PROGRAM

        ORG PGAR         SET THE ORIGIN
DEMO    LDI H(MSG-1)     SET D AND E TO ADDR
        LEI L(MSG-1)      THE MESSAGE-1
        LHI H(DTAR)      SET H TO ADDRESS
*                         THE DATA AREA
        CAL TTINT        INITIALIZE THE TTY
DEMO1   CAL INCR         POINT DE TO NEXT CH
        LBI MSGLEN       SET LOOP COUNT IN B
DEMO2   LHD              TRANSFER D AND E TO
        LLE               H AND L
        LAM              LOAD THE CURRENT CH
        LHI H(DTAR)      RESTORE HL TO ADDR
*                         THE DATA AREA
        CAL TTOUT        OUTPUT THE CHAR
        CAL INCR         POINT DE TO NEXT CH
        DCB              DECR LOOP COUNT
        JFZ DEMO2        GO FOR NEXT CH IF
*                         NOT ZERO
        LAI 015B         OUTPUT A CARRIAGE
        CAL TTOUT         RETURN
        LAI 012B         OUTPUT A LINE FEED
        CAL TTOUT
        JMP DEMO1        GO FOR NEXT CHAR

INCR    INE              DOUBLE INCREMENT D
        JFZ *+4           AND E BY 1
        IND
        LAE              RETURN IF NOT END
        SUI L(MSGEND)     OF MESSAGE
        LAD
        SBI H(MSGEND)
        RTC
        LDI H(MSG)       RESET D AND E TO
        LEI L(MSG)        THE START OF MSG
        RET               AND RETURN

MSG     DEF 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
        DEF '0123456789-?:$!&''().,;/"'
        DEF 'A1B2C3D4E5F6G7H8I9J0'
MSGEND  EQU *
MSGLEN  EQU MSGEND-MSG

*       TELETYPE INITIALIZATION ROUTINE

TTINT   LLI L(TTCSF)     SET THE CASE SHIFT
        LMI 0             FLAG TO LTRS
        LAI CNTTO        SEND A STOP BIT TO
        OUT CNTREG        THE TELETYPE
        LAI 8            DELAY FOR ONE CHAR
TTINT1  LLI TTDLY1        TIME
TTINT2  LAA
```

```
        LAA
        DCL
        JFZ TTINT2
        SUI 1
        JFZ TTINT1
        LAI 037B         OUTPUT A LTRS CODE
        CAL TTOUX
        RET              RETURN

*       BAUDOT TELETYPE OUTPUT ROUTINE

TTOUX   NDI 037B         ISOLATE THE BAUDOT
        ADA               CHAR AND SHIFT
        ADA               IT INTO POSITION
        ORI 002B         ADD A STOP BIT TO
*                         THE CHARACTER
        LHI 7            SET BIT CNT IN H
TTOUX1  ADA              SET THE CURRENT BIT
        LLA               IN THE CARRY AND
*                         SAVE THE OTHER
*                         BITS IN L
        SBA              OUTPUT THE CURRENT
        NDI CNTTO         BIT TO THE SERIAL
        OUT CNTREG        TELETYPE LINE
        LAL              RELOAD THE OTHER
*                         BITS INTO A
        LLI TTDLY1       LOAD THE DELAY CNT
TTOUX2  LAA               INTO L AND DELAY
        LAA               ONE BIT TIME
        DCL
        JFZ TTOUX2
        DCH              DECREMENT THE BIT
        JFZ TTOUX1        CNT AND LOOP IF
*                         NOT ZERO
        LLI TTDLY2       LOAD THE DELAY CNT
TTOUX3  LAA               INTO L AND DELAY
        LAA               FOR THE REMAINDER
        DCL               OF THE STOP BIT
        JFZ TTOUX3
        LHI H(DTAR)      RESTORE H TO ADDR
        RET               THE DATA AREA AND
*                         RETURN

*       ASCII TELETYPE OUTPUT ROUTINE FOR
*       BAUDOT MACHINE

TTOUT   CAL CHATB        TRANSLATE ASCII IN
*                         A TO BAUDOT IN A
        ORA              BRANCH IF IT IS A
        JTZ TTOUT3        NULL
        CPI 002B         BRANCH IF IT IS A
        JTZ TTOUT3        LINE FEED
        CPI 004B         BRANCH IF IT IS A
        JTZ TTOUT3        SPACE
        CPI 010B         BRANCH IF IT IS A
        JTZ TTOUT3        CARRIAGE RETURN
        LLI L(TTSAV1)    SAVE CHAR IN TTSAV1
        LMA
        LLI L(TTCSF)     BRANCH IF TTY IS IN
        XRM               CORRECT CASE SHIFT
        NDI 040B
        JTZ TTOUT2
        LAM              FLIP THE CASE SHIFT
        XRI 040B          FLAG
        LMA
        LAI 037B        LOAD THE CORRECT
        JTZ TTOUT1       SHIFT CODE INTO A
        LAI 033B
TTOUT1  CAL TTOUX       OUTPUT THE SHIFT
*                        CODE
TTOUT2  LLI L(TTSAV1)   RELOAD THE CHAR
        LAM              INTO A
TTOUT3  CAL TTOUX       OUTPUT THE CHAR
        RET              AND RETURN

*       ASCII TO BAUDOT TRANSLATE ROUTINE

CHATB   NDI 177B        ZAP THE PARITY BIT
        CPI 177B        REPLACE A RUB-OUT
        JFZ CHATB1       WITH A NULL
        XRA
CHATB1  CPI 'A'+040B    CONVERT A LOWER
        JTC CHATB2       LETTER TO UPPER
        CPI 'Z'+040B     CASE
        JFC CHATB2
        SUI 040B
CHATB2  LHI H(BDTAS-1)  SET HL TO ADDRESS
        LLI L(BDTAS-1)   BAUDOT TO ASCII
*                        TRANSLATE TABLE-1
CHATB3  INL             BUMP UP HL TO THE
        JFZ *+4          NEXT TABLE ENTRY
        INH
        XRM             BRANCH IF MATCH
        JTZ CHATB4       BETWEEN ENTRY
*                        AND ASCII CHAR
        JTS CHATB5      BRANCH IF END
        XRM             RESTORE A AND LOOP
        JMP CHATB3
CHATB4  LAL             LOAD L INTO A AND
        SUI L(BDTAS)     COMPUTE THE BAUDOT
*                        CHARACTER
        LHI H(DTAR)     RESTORE H TO ADDR
        RET              THE DATA AREA AND
*                        RETURN
```

```
CHATB5  XRM             RESTORE THE ASCII
        LHI H(DTAR)      CHARACTER IN A AND
*                        H TO ADDR THE DATA
*                        AREA
        CPI 040B        LOAD A BAUDOT NULL
        LAI 0            INTO A AND RETURN
*                        IF THE ASCII CHAR
        RTC              WAS A CONTROL CODE
        LAI 063B        LOAD A BAUDOT '?'
        RET              INTO A AND RETURN

*       BAUDOT TO ASCII TRANSLATE TABLE

BDTAS   EQU *           SET ADDR OF BDTAS
        DEF 000B,124B,015B,117B  00  LTRS
        DEF 040B,110B,116B,115B  04
        DEF 012B,114B,122B,107B  10
        DEF 111B,120B,103B,126B  14
        DEF 105B,132B,104B,102B  20
        DEF 123B,131B,106B,130B  24
        DEF 101B,127B,112B,000B  30
        DEF 125B,121B,113B,000B  34

        DEF 000B,065B,015B,071B  00  FIGS
        DEF 040B,043B,054B,056B  04
        DEF 012B,051B,064B,046B  10
        DEF 070B,060B,072B,073B  14
        DEF 063B,042B,044B,077B  20
        DEF 007B,066B,041B,057B  24
        DEF 055B,062B,047B,000B  30
        DEF 067B,061B,050B,000B  34

*       DATA STORAGE

        ORG DTAR        SET THE ORIGIN TO
*                        THE DATA AREA
TTCSF   DST 1           TELETYPE SHIFT FLAG
*                        0=LTRS 40B=FIGS
TTSAV1  DST 1           TELETYPE TEMPORARY
*                        STORAGE
        END DEMO
```

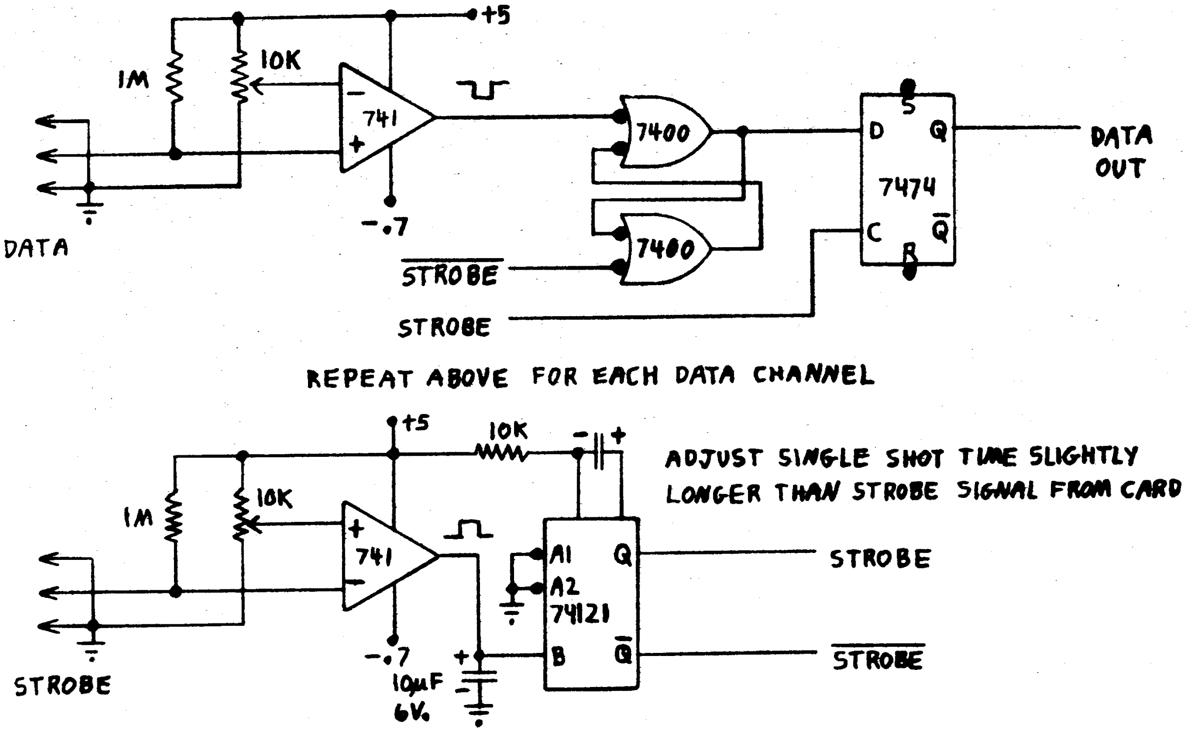
A CHEAP MARK SENSE CARD READER   by Joe Tolbert

John B. Kramer, 3400 Old Jonesboro Road, Hapeville, GA 30354 (Ph. 761-6030 after 5 PM.) is selling a card reader made for Western Union for $5 plus shipping for 12 pounds (your post office can tell you how much that will be). What you get for your money is a new card transport assembly driven by a 110 volt 60Hz motor with a set of switches to indicate things like "cards in hopper" etc., a couple of indicator lamps and pushbuttons, all brought out to a minature connector. These can be wired together with a 110 volt relay to give "start reading when start switch is operated", and "run until out of cards or reset is operated". A pair of gears, not supplied, set the speed of the reader. If you can't find gears to fit in your junk box, try rubber tires from a child's toy. With one inch diameter gears, it reads about 10 cards a minute.

The reader comes with a "stylus assembly" of 40 spring loaded contacts brought out individually to another minature jack. These contacts could be used to sense holes in the card by allowing them to contact the frame below the card. Although the brush centers don't exactly match IBM card hole centers, you are assured of at least one of them making good, solid contact through each hole. I have designed a system to use the reader for reading homemade mark-sense cards. It converts answers on a test (any number out of 5) into valid ASCII characters which are sent to a tape punch so that a "call-a-computer" terminal can be used to grade tests. This same idea could be used for 8 data channels for loading programs, or possibly up to 16 without too much problem.

The sensing circuit shown in the figures can be adjusted to work with up to a light line from a #2H pencil. An IBM electrographic pencil is not needed. This also means that you can't use carbon base ink to print the card forms.

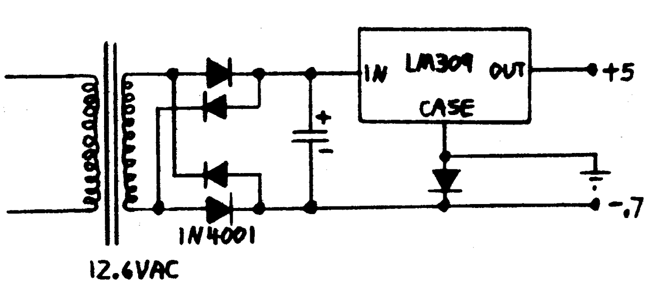
When a card starts into the reader a strobe pulse clears the 7400 S-R flip-flops and transfers what was in them to the 7474's and output lines. As marks are sensed in the data channels the + input of the op-amp goes low. When the voltage at the + input goes lower than that at the - input as set by the sensitivity control, the output voltage swings from about 5 volts to about 0 volts, pulling down the input of the 7400 cross connected gates and storing the information until a strobe pulse comes along. Please note that the op-amp is tied from +5 volts to -.7 volts. The -.7 volts can be obtained by passing current through a silicon diode. The -.7 volts is required so that the op-amp is guaranteed to pull down the TTL.



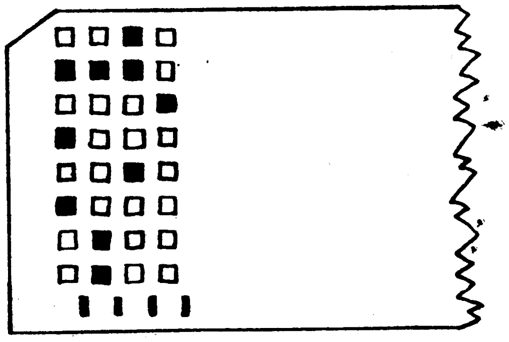
REPEAT ABOVE FOR EACH DATA CHANNEL

ADJUST SINGLE SHOT TIME SLIGHTLY LONGER THAN STROBE SIGNAL FROM CARD

Mark Sense Circuitry



Example Power Supply



Sample Mark Sense Data Card for 8 Bits Plus Strobe

SURPLUS SUMMARY

This month surplus information abounds. To lead off we have a special report contributed by Jerry R. Ledbetter.

Surplus In East Texas

Those readers of TCH who live in east Texas are fortunate in that there are many surplus places around Dallas and Houston. The unfortunate part is that not all of them are easy to find. Even though I had time for only a few of them I bought more useful electronic parts than I could cram in my suitcase. Here are several places that are worth looking up if you're in the area.

Special Sales Co.
12400 N. Central Expressway
Dallas, Texas 75208

This is the most commercial of the places that I have visited, but Frank Robertson (the owner) does keep a good stock of electronic surplus. There are no great buys here, but many good ones such as alpha-numeric keyboards with ASCII encoding for $25 and used but good MAN-3 displays for 30¢ each. Frank does back up his stock with a replacement guarentee, but doesn't like to "bargain". He carries many transistors and IC's both new and used and one of the largest selections of nuts and bolts hardware I've seen. One catch is that he does not do mail order business.

The next place is not a surplus house, but a scrap disposal agency for the Collins Radio Company. They have no address and about all I can tell you is that they are on Hilltop Rd. in Richardson, Texas which is a north suburb of Dallas. I stumbled upon this place one normal day when I was hopelessly lost. They have lots of parts (on circuit boards) and had some disk surfaces. I found plenty of usable parts there including some keyboard keys.

Another place in Dallas that's worth checking into is:

Altaj Electronics
Box 38544
Dallas, Texas 75238
Ph. 214/271-6440

They advertise in some of the larger magazines and have some very good prices. They will mail items but will not accept COD orders.

In Houston:

Radio Electronics & Supplies
1508 McKinney Ave.
Houston, texas

have some good buys from time to time. When I was there they didn't have very much that would have been useful to TCH readers except the usual assortment of resistors and caps, etc. Although they lean primarily toward HAMs this is one of those places where you never know what will turn up. Look into them if you're in Houston.

There are many more places in this area of Texas, but no single place would satisify every need. Some great buys turn up in unpredictable places, like an Augat prototype board with 300 wirewrap IC sockets that I bought for $10 from a guy who didn't know what it was. Personally I miss the days of scrounging around in the rain in a dirty scrap yard and finding only one single 2N3055 under 4 tons of metal. Now that's surplus hunting!

Now some mail-orderable items which we have run accross during the past month.

Mini-Micro Mart
C/O syracuse Management Services
1618 James Street
Syracuse, N.Y. 13203
Ph. 315/422-4467

Maury Goldberg of the Mini-Micro Mart has two exceptional offerings, however both are sold as OEM-type assemblies and require a fair amount of knowledge to fully interface. The first item is a digital cassette tape drive made by ICP. The second is a 132 column 30cps printer made by Univac (same as in 727 terminal). Both units are either new or very slightly used and both are available for about $200. Contact Maury for details.

Tri-Tek, Inc.
Box 14206
Phoenix, ariz. 85063
Ph. 602/931-6949

Tri-Tek's latest catalog contains two things of potential interest to a computer hobbyist. The first is a fine selection of large electrolytic filter caps.

Example: 60,000 mFd at 20 volts for $2. Four of these would make an excellent filter bank for a power supply for a graphics display like TCH's. Also available are General Instrument UART's (Universal Asynchronous Receiver-Transmitter) for $13.95. While this price is not fantastic, it is reasonable and UART's aren't advertized often.

Fair Radio Sales
1016 East Eureka Street
Box 1105
Lima, OH. 45802
Ph. 419/223-2196

Fair Radio is a good source of CRT's for a graphics display. You can get a 12SP7 CRT for $15 or a radar display assembly which nets you a 12DP7A, a nice chassis, and the orange filter needed for the P7 phosphor, all for 22.95. Unfortunately the yoke is not suitable for graphics.

Bill Godbout Electronics
Box 2673
Oakland Airport, CA 94614
Ph. 415/357-7007

Corky Deeds of Godbout Electronics called TCH to let us know of a special offer he would make to TCH subscribers. The offer is one 8008 CPU by Intel, eight 2102 RAM's, one 5203 U.V. erasable PROM, all for $100. Thats all the makings of a great system! Be sure to mention TCH.

PEOPLE'S COMPUTER COMPANY

We at TCH are printing the following special offer from People's Computer Company to our subscribers in return for similar mention in the PCC newspaper.

Send a letter mentioning TCH and $3 cash, check, or money order for a one-year subscription to People's Computer Company, our newspaper about computers for FUN! PCC is 28 pages tabloid (11"X17") published 5 or more times a year. It contains programming tips, book reviews, and lots of computer games. Regular subscriptions are $5 so here is your chance to save two bucks. Send to PCC, Box 310, Menlo Park, CA 94025. This offer expires March 1, 1975.

CLASSIFIED ADS

There is no charge for classified ads in TCH but they must pertain to the general area of computers or electronics, and must be submitted by a non-commercial subscriber. Feel free to use classified ads to buy, sell, trade, seek information, announce meetings, or for any other worthwhile purpose. Please submit ads on seperate sheets of paper and include name and address and/or phone number. Please keep length down to 10 lines or less.

WANTED: Game and puzzle software for computers. Will exchange picture tapes also. Fred Hatfield, 1372 Grandview Ave, Columbus, OH 43212

FOR SALE: Keyboards, by Microswitch, with documentation and cover $25. 74S181's $2.50. I gotta clear out my room. I have a lot of stuff to sell, send a SASE for a list. I have some memories, 2602's @ $8, core, and other types. Gary Coleman, 530 Glaser Bldg., 11900 Carlton Rd., Cleveland, OH 44106

WANTED: Source of P C boards for the TV Typewriter, also need information on Western Electric 202-C Dataphone. Have 15 National MM5058 shift registers at $5 each. Ed Lankford, 511 Purnell Dr., Nashville, TN 37211

WANTED: Digital Equipment Company modules for PDP/8 series computers. Also, software, manuals for obsolete machines. I am interested in obtaining cryptographic items and publications. Fred Hatfield, Computer Data Systems, Inc., 1372 Grandview Ave., Columbus, OH 43212, Ph. 614/486-0677

THE POWER OF AN 8008

Last month in this space we described the system that is currently being used to prepare TCH. What we failed to mention is the fact that the system belongs to and was designed by the employer of one of the staff members. Since the system will be a commercial product, it will not be possible to elaborate on the design, software, or capabilities of the system at this time. TCH will be developing its own system soon and may publish some aspects of it if interest warrants.

THE COMPUTER HOBBYIST

Volume 1 Number 4

February 1975

Alas, it is dated February and not March, so you did not miss an issue after all. It seems that TCH, like so many other things today, has problems being on time.

You will note that this issue is on 8 1/2 by 11 inch sheets which will fit into a ring binder. After a substantial number of requests for this format TCH accepted it as valid and then set about figuring out details such as reduction ratio in order to keep costs approximately the same as with the old format. Conceptually there are larger pages, with the same size print, but fewer of them. One bonus gained with larger pages is that circuit diagrams should be more readable.

With the last issue TCH sent out a survey form on interests and equipment to our subscribers. The response rate to that survey was 71%. This in itself shows one important fact, there IS interest out there. The results of the survey are shown below by category. The percentile numbers are the percentages of the responses indicating an interest in the item named.

Our first category was software articles. The results were:

Computer games	87%
Graphics software	83%
Calculator programs	68%
Accounting programs	39%
Operating systems	81%
Electronic RTTY station	27%
Streamlined BASIC	86%
Source statement editor	63%
Morse code sender	17%
I/O programming techniques	88%
Music composition	36%
HAM repeater control	11%
Assemblers	92%

TCH would like to publish articles on all of the big interest items, however the ones which are simpler and require less memory will come first. Unfortunately both BASIC and assemblers require lots of memory, but a simplified assembler will be looked at since it drew so much interest (92%).

The next category was hardware articles. The large amount of interest and the equality of the spread were impressive. Even the least popular item drew 29% in favor. The results were:

Incremental plotter	47%
AD/DA systems	56%
Touch-Tone receiver	36%
Modems	58%
Keyboard encoders	37%
General I/O concepts	80%
Prog. frequency synthesizer	29%
Memory controller and DMA	62%
Hardware multiply/divide	54%
Interfacing dynamic memory	74%
Raster scan CRT display	61%
Music synthesizer interface	31%
Cassette recorder interface	94%

It is obvious that the most desired article was one about a cassette recorder interface. Since this was anticipated, TCH has already been at work on a system for cassette recording. We abandoned sterotyped ideas, such as FSK, and searched for a simple, truly reliable system. The system as developed thus far utilizes less than 10 modules both digital and analog, achieves a transfer rate in excess of 400 bits per second, and has an error rate which is so low that we have not experienced any errors at all on 4 different recorders to date. Watch for an article in our next issue.

Articles on memory systems and raster scan CRT's are also in the works. In the way of I/O, TCH is looking at the possibility of a unified bus system which would make interfaces compatible between different microprocessors such as the 8008, 8080, IMP-16 the PACE (a soon to be produced 16 bit microprocessor single chip, and which is almost identical to the IMP-16). If this is possible TCH could offer circuits (and maybe PC cards) which could be used on most systems.

Our last category was equipment. The range in this category was amazing, from people with nothing but an interest in the subject to people with a little of everything. In this list the first percentage is for equipment people already have, the second is for equipment planned. The results were:

MARK-8	39%	2%
Homebrew 8008	8%	8%
MITS 8080 (ALTAIR 8800)	19%	37%
IMP-16 by International Elec. Unl.	0%	3%
Homebrew TTL CPU	6%	3%
Scelbi 8008	12%	2%
RGS 8008	1%	0%
Other 8008	2%	2%
Homebrew 8080	3%	11%
PDP-8	2%	2%
TTY	54%	11%
Serial Printer	8%	7%
Graphics CRT	2%	41%
Plotter	2%	13%
AD or DA	6%	14%
Modem	19%	22%
Moving head disk	1%	6%
R.E. T.V. Typewriter	32%	24%
Other CRT	11%	9%
Paper tape reader	39%	12%
Paper tape punch	34%	9%
Electronic keyboard	49%	13%
Fixed head disk or drum	1%	7%
Line printer	3%	17%
Card reader	2%	13%
Card punch	3%	11%
Cassette tape I/O	23%	67%
Other magnetic tape I/O	11%	9%
Floppy disk	6%	27%

It should be noted that at the time of the survey, MITS had only begun to ship ALTAIR 8800's, therefore the numbers beside it should be considered the number of orders sent in. Two figures are interesting in particular, one - over 50% had a teletype of some type, and two - 67% planned to have cassette I/O while 23% already did. This latter figure points out the need for some sort of standard cassette format, a need which TCH hopes to fill both by publishing hardware and software and by releasing programs on cassette.

LETTERS

We at TCH will publish a few of our more interesting letters each month along with comments by the staff. The following are excerpts from a long letter which Gordon French wrote to People's Computer Company and sent a copy to TCH for our use.

/ / /

I've been following the articles on the CRT display that "The Computer Hobbyist" is publishing. My opinion is that this is the best CRT display system that I've seen. It is reminiscent of the method used in the PDP-12/LINC-8 CRT displays. It is different than the TVT concept in that the beam does not scan, but is positioned by X and Y commands. In short it handles more like a plotter system. That is you can "beam on" [intensify point] (pen down) or "beam off"[lower intensity] (pen up). This yields a lot more flexibility to the charactcr of the displays that are possible. Figures, lines, curves, and kaliedoscope effects are relatively easy to do. The method has been around long enough to be good, practical, and versatile. They are describing the hardware and software and are publishing routines that draw the chess board and chess pieces. If this turns you on, their address is: The Computer Hobbyist, Box 295, Cary, North Carolina, 27511. Six inflated American dollars will get you a years worth of very good stuff. They are covering the area of logic symbols, surplus markets, and the letters from some of their subscribers are describing some of the equipment that has traditionally been available, like teletypes of all kinds, sorts, styles, and descriptions. They also have done an article on interfacing the Teletype model 15 to a home computer system. Since that was the machine that I used in my system (version one) I was interested in that very much. I still feel that for $50 to $75 a model 15 can't be beaten. I think that it is a whole lot less work to rewire this machine than to wire a TVT thingee from scratch. If you ever do undertake this project, you will learn all there is to know about teletype signals and it does make stepping up to an ASCII device a great deal easier to understand after you have made a model 15 work. You can, incidentally, still use ASCII for the other devices that you have hung on your system, and go through a short piece of translation code just before you need to print anything. If you are held up on account of you don't have a print type of device, you might ask the North Carolina boys for the back issues and take a look. There is information as to where to go and what to buy. There are outfits around Chicago (if my memory serves me) that buy and sell this type of gear. They cater to the radio-teletype boys. There is a variety of Baudot (five level code) machines that are 110 Baud machines.

/ / /

Somebody else who seems to be going gang busters is Carl Helmers, Jr. who is otherwise known as: M. P. Publishing Co., Box 378, Belmont, Mass., 02178 is publishing a series of booklets covering the construction of what he calls "The Experimenter's Computer System" or ECS-8 for short. Send him $21 for 12 issues of this system. He is developing it in serial form in the booklet. It is getting better and better as he goes along. He includes some choice coding charts which give instruction times and other valuable information that you would compile for yourself if you had the time. He has designed an LED display and hung it on an output port so that you can ship bits out there and take a look at them if you want to. He is into a really stable clock, and has published the best tutorial description of UART's that I've ever seen in print. The software to run his stuff is also in the booklet and the feeling that you come away with after reading his stuff a little while is that with just a little more effort and a few additional check lists, he is putting out the paperwork of a Heathkit. Other good stuff and ideas that he has come up with are that he has what he calls a kludge assembler system. It is a pre-printed form with the byte numbers and the space to fill in the instructions as you code them. It is precisely what I have been painstakingly doing for myself for several weeks now. He wants a nickle a sheet for them and instructions are written on the heel (so to speak). He is also aware of the software that is/or will be necessary to get all the stuff playing, and he is hard at work developing that too. It is a very creditable effort.

/ / /

One final note. Those of you who have already started your 8008 system need not panic or dispair because MITS is coming out with a better goodie. If you can finish your system as you originally set out to do, go right ahead. I personally think that the 8080 for all of it's obvious advantages is a lot more computer than I could use this year at home. By the time I have gotten to the end of developing my 8008 system to the full, memory costs, easier interface techniques and devices will be available. And more important I will know better just what it is that I intend to do withe the system. Understand that I'm not putting down the Altair 8800, far from it. But my experinece is that it takes a long time to develop anything that is really worth while, and I for one am not going to junk what I've started until I have squeezed out all that I can from a really inexpensive system. Memory cost is still the big hangup and it will continue to be for some time yet. Besides, a clipped canary at a tom-cat's convention has a better chance than I have of convincing my lady that I need a disk system for the living room.

Gordon A. French

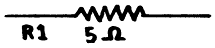
TCH wanted to let our readers know about about M. P. Publishing Co. but also we could not resist putting in Gordon's plug for TCH. Later (as soon as our copies arrive) TCH will be doing a book review on some of M. P. Pub's stuff.

Congratulations!

I would not be surprised if this publication grows <u>very</u> quickly.

Suggestions:

1. How about a section devoted to book reviews & new publications.
2. Perhaps when there are enough members your (our) organization could buy electronic components in large numbers and pass the savings along. (ie TTL ic's)
3. What if future issues were made 8 1/2 X 11 so that they could be more easily stored (ie, in binders). I have a strong feeling this publication might get relatively large.
4. When giving schematic diagrams it may prove helpful to use reference designations. For example:



If this were given in a certain issue and it was later realized that R1 should have been 5K instead of 5 ohms, it would be quite easy to correct. (ie in ckt. 33 chg. R1 to 5K) If this were not done the reader may have difficulty in figuring out which 5 ohm resistor should be changed to 5K.

Enough said.

I wish you all the best for the future.

Paul Costello

TCH will be offering book reviews from time to time, recently, however, no one here has had the time to read a book. As for collective buying see Surplus Summary and the Want Ads in this issue. And for number 3, you're holding it.

THE COMPUTER HOBBYIST
Founded October, 1974

Stephen C. Stallings - Managing Editor
Hal Chamberlin - Contributing Editor
Edwin Tripp - Photographer
Richard Smith - Programming Consultant

THE COMPUTER HOBBYIST is published monthly near Raleigh, N.C. The yearly subscription rate is $6.00 for persons in the U.S.A., U.S. possessions, and Canada. Subscriptions in other countries considered individually with rate variances based on postal charges.

All correspondence should be addressed to:

THE COMPUTER HOBBYIST
Box 295
Cary, NC 27511

Members of the staff of TCH can usually be reached by phone evenings and weekends at either 919/467-3145 or 919/851-7223.

THE COMPUTER HOBBYIST welcomes contributions from our readers. Material to be submitted should be typed or neatly written and must not appear to be soliciting business for any firm. If you wish your material returned, please include a stamped, self-addressed envelope.

HUMAN INTERFACE YOUR GRAPHICS DISPLAY

by Hal Chamberlin

In order for the graphics display described in issues 1, 2, and 3 of TCH to be truly interactive, a proportional type of human interface is needed. Commercial systems utilize light pens, joysticks, track balls, data tablets, and control dials for the interface. With one or more of these devices it is possible to enter graphical data and interact with it quickly and conveniently. In the case of interactive computer games, manual skill and reaction can be made a factor in the scoring. Of this list, joysticks and control dials are the easiest to obtain and interface. A control dial is simply a high quality potentiometer with a large knob. The program is able to accurately read the position of the dial whenever desired. Two dials and a simple program can simulate an Etch-A-Sketch, with one dial controlling X, the other controlling Y, and the path being stored in memory. Four dials and a push-button can be used with a program for completely general drawing. One pair of dials would control the starting point of a line, the other pair would control the ending point, and the push-button would store the line in memory and allow positioning of the next line. A more sophisticated program would allow the user to position symbols and connect them with lines such as would be done in preparing flowcharts or logic diagrams. A joystick is simply a pair of potentiometers mechanically linked to a handle so that horizontal motion moves one pot and vertical motion moves the other. Joysticks are easier to use for X-Y data than dials but cost $20 and up.

An interface for either a joystick or a dial is primarily an analog-to-digital converter and a stable voltage source. The potentiometer is tied across the voltage source and the wiper is connected to the analog-to-digital converter. As the user rotates the pot, a varying fraction of the reference voltage is seen by the A-D converter and the program. The reference voltage is chosen such that full rotation of the pot exactly spans the full range of the A-D converter. Additional pots may be connected across the same voltage source and the individual wipers connected to the inputs of a multiplexed A-D converter. The program then selects the control it wishes to read before initiating the conversion. The interface to be described uses a portion of the graphics display interface and a small handful of parts to provide four analog inputs for potentiometers and four digital inputs for function switches using one 8008 or 8080 input port.

Analog-to-digital conversion can be accomplished in a number or ways. The method used here uses the digital-to-analog converter from one axis of the graphics display, an analog comparator, and a short program to determine what the input voltage is. Figure 1 shows the circuitry for the 4 analog inputs and 4 digital inputs. The RAWX output from the display is applied to the - inputs of four op-amps used as comparators. Each analog input is then connected to the + input of the corresponding op-amp. The resistor-diode network at the comparator output clamps the + and - 15 volt swing to TTL logic levels to feed the open-collector input bus interface gates. The NAND gate at the top of the diagram is connected to the user´s system so that the desired input port address is decoded. The comparators can drive two TTL loads if your system requires a different input interface arrangement.

When an INP instruction is issued to the interface (it will be given the symbolic address GINP in the program listings) the lower 4 bits read into the accumulator will be the 4 comparator outputs. If an analog input voltage is lower than the RAWX voltage from the display DAC, then the corresponding bit read into register A will be ZERO. If the input voltage is higher, then the bit would be a ONE. A simple routine for analog-to-digital conversion would start by sending a trial value of -128 to XMOV and then testing the bit corresponding to the input of interest. If it is a ZERO, then the trial value is incremented and another test is performed. If it is a ONE, then the current trial value is the conversion result. A test for overflow after the trial value increment should be performed to prevent an endless loop if the analog input is out of range. This is a rather inefficient conversion algorithm since up to 256 iterations are required for a conversion.

The program listing in the appendix requires only 8 iterations to do a conversion. The algorithm is formally called successive approximation but computer science people should recognize it as a binary search. What we are actually trying to do is find the point in the sequence of trial values, -128, -127, ... 0, 1, ... 126, 127 where the comparator bit changes from 0 to 1. Since there is only one such point, the binary search method is applicable. In a binary search, the search interval is first divided into halves. Then a test is made to determine if the thing being searched for is in the first half. If so, then this half is divided in half and another iteration is performed. If it was not in the first half, then it must have been in the second half. In that case, the second half is divided in half and another iteration performed. It is easy to show that with 256 points in the search area, only 8 iterations are needed to guarantee that the transition point is found.

With binary numbers and binary arithmetic, dividing the intervals in half is simply a matter of bit setting and resetting. For illustration, consider the trial values as unsigned binary numbers from 0 to 255. Figure 3 shows how a comparator transition at 175 would be found. First the interval from 0 to 255 is divided in half at 128 by setting the most significant bit of the trial value to ONE. The trial value is then sent to XMOV and the comparator is read. The comparator bit is a ZERO indicating that this trial is too low and that the transition is in the upper interval, from 128 to 255. This interval is then cut in half by setting the next most significant bit giving a trial value of 192. This time the comparator returns a ONE indicating that the transition is in the lower half. In this case, the bit in the trial value that was just set is reset and then the next most significant bit is set for another iteration. The procedure is finished when all of the bits in the trial value have been tried. Since the display DAC uses two´s complement numbers and we want a two´s complement result from the conversion, the most significant bit of the trial value must be complemented when being sent to the display DAC and when being returned as the conversion result.

The conversion program includes a couple of NOOP instructions between changing the DAC and reading the comparator to allow 741´s to be used as comparators. These NOOP´s may be eliminated if 748´s with 3.3 pF compensation capacitors are used for the comparators. Since a portion of the display is used in the conversion sequence, sufficient time for the last display operation must be allowed before a dial is read. Also note that the final endpoint of the last vector drawn is lost when a conversion is done. A convenient way of organizing an interactive display program is to refresh the image once, read the dials and function switches, and then go refresh again.

Figure 2 shows a typical interface between potentiometers and push-button switches and the circuitry in Fig. 1. A pair of zener diode regulators provides noise-free reference voltages of +7.5 and -7.5 volts from the system +15 and -15 volt supplies. The two 5K pots are used to adjust the actual reference voltage accross the dial pots to +2.5 and -2.5 volts, thus matching the input range of the analog to digital converter. Adjustment is somewhat tricky since the two controls interact but should not be difficult. If joysticks are used instead of dials, more than + and -2.5 volts will probably be needed accross the internal pots since most joysticks utilize only a portion of the resistance element. In severe cases, higher voltage zeners may have to be substituted. The .1uF capacitors connected to the wipers suppress noise that may be generated when the pot is turned and the wiper momentarily breaks contact with the resistance element.

The cross-coupled 7400 NAND gates in the push-button interface eliminate possible contact bounce when the button is pressed or released. The button read subroutine in the appendix can be used if a pulse type of switch action is desired, i.e., only one operation for each depression of the switch. The routine works by comparing the present state of the four switch bits with their state when the routine was last called. If a switch was previously OFF but is now ON, then the corresponding bit in the return value will be a ONE. This scheme allows each button to be either continuous action or pulse action under program control.

In the future, TCH will publish interactive display programs using this human interface. As always, readers are invited to submit their own programs or articles for publication.

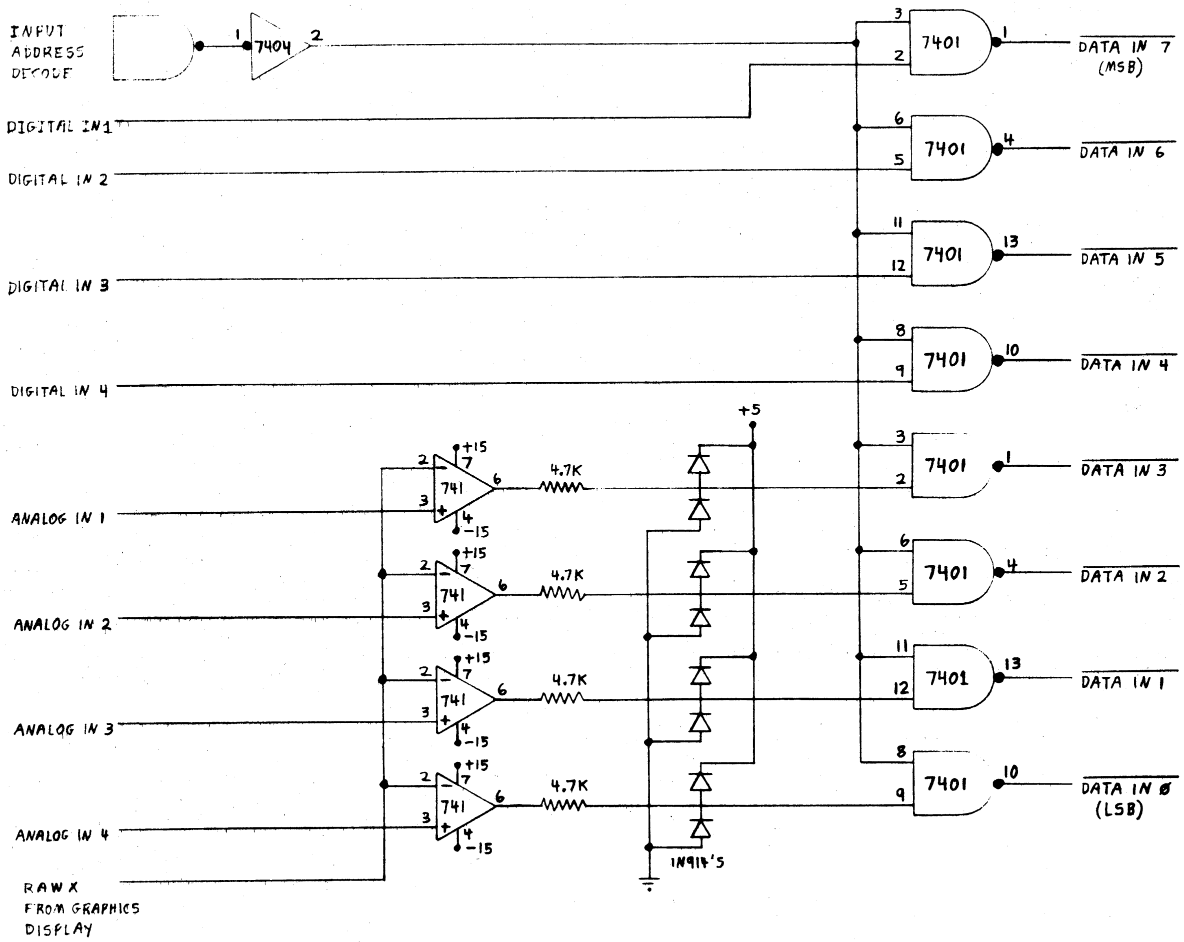


FIGURE 1 DIGITAL AND ANALOG INPUT INTERFACE

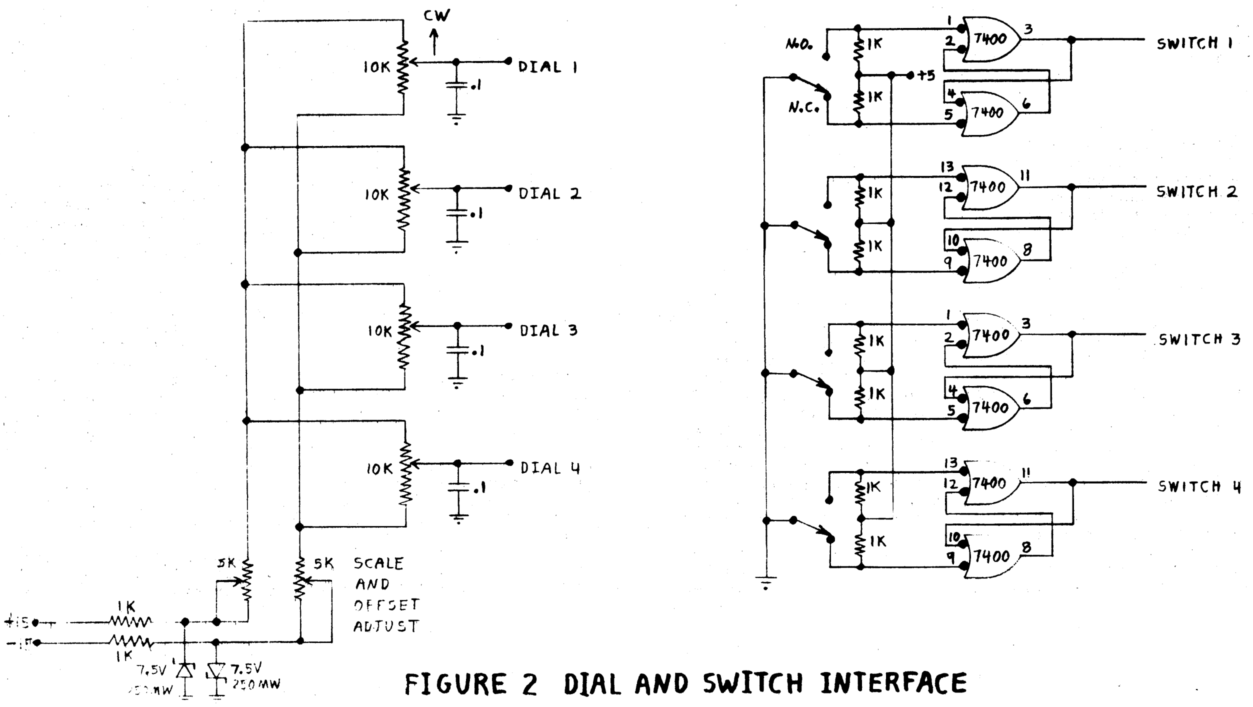
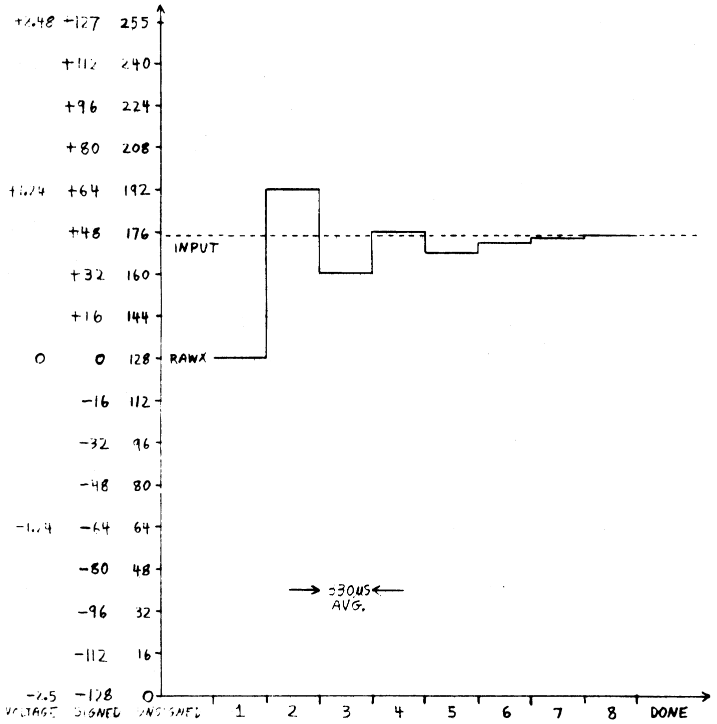


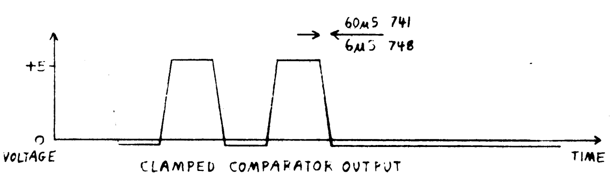
FIGURE 2 DIAL AND SWITCH INTERFACE

Figure 3 Binary Search for Transition at 175

STEP	TRIAL-BINARY	DECIMAL	COMPARATOR	ACTION
1	10 000 000	128	0	KEEP
2	11 000 000	192	1	DROP
3	10 100 000	160	0	KEEP
4	10 110 000	176	1	DROP
5	10 101 000	168	0	KEEP
6	10 101 100	172	0	KEEP
7	10 101 110	174	0	KEEP
8	10 101 111	175	0	KEEP

All bits tried, finished.





CLAMPED COMPARATOR OUTPUT

```
*       PROGRAM TO DISPLAY RUBBER BAND
*       LENGTH & ORIENTATION DETERMINED BY 4 CONTROL DIALS

        ORG 30000B

XMOV    EQU 10B
YMOV    EQU 11B
XSTOR   EQU 12B
YDRAW   EQU 13B
MINXY   EQU 14B
MINSZ   EQU 15B
GINP    EQU 6B

BAND    XRA                 ZERO MINOR SYSTEM
        OUT MINXY
        LAI 4               DIAL 4, Y2
        CAL RVCD
        LCA
        LAI 2               DIAL 2, Y1
        CAL RVCD
        LDA
        LAI 3               DIAL 3, X2
        CAL RVCD
        LEA
        LAI 1               DIAL 1, X1
        CAL RVCD
        OUT XMOV            MOVE TO  X1, Y1
        LAD
        OUT YMOV
        LAA                 WAIT FOR BEAM MOVE
        LAA
        LAA
        LAE
        OUT XSTOR           DRAW TO  X2, Y2
        LAC
        OUT YDRAW
        JMP BAND            LOOP

*       SUBROUTINE TO READ MANUAL INPUT SWITCHES
*       AND PROVIDE BOTH CONTINUOUS AND PULSE OUTPUT.
*       REQUIRES ONE TEMPORARY MEMORY LOCATION
*       CALL RMISO FOR INITIALIZATION
*       CALL RMIS TO READ SWITCHES
*       RETURN WITH CONTINUOUS OUTPUTS IN REG. B
*       AND PULSE OUTPUTS IN REG. A
*       USES REGISTERS A, B, H, AND L

RMISO   SHL RMIST           SET PREVIOUS SWITCH STATES
        LMI 0               TO OFF
RMIS    INP GINP            READ GRAPHICS INPUT
        NDI 360B            ISOLATE SWITCH BITS
        LBA                 SAVE IN B AS CONTINUOUS OUTPUTS
        SHL RMIST           ADDRESS PREVIOUS SWITCH STATES
        LAM                 GET THEM INTO A
        XRI 360B            FLIP THEM
        NDB                 AND WITH PRESENT STATE TO DETECT
*                           OFF-TO-ON TRANSITIONS
        LMB                 SAVE PRESENT STATE IN MEMORY
        RET                 RETURN, INDICATORS SET ACCORDING
*                           TO PULSE OUTPUTS

*       SUBROUTINE TO READ VARIABLE CONTROL DIALS
*       ENTER WITH DIAL NUMBER TO BE READ (1,2,3, OR 4) IN REG. A
*       EXIT WITH DIAL POSITION (-128 TO +127) IN REGISTER A
*       USES REGISTERS A, B, H, AND L

RVCD    LBA                 COMPUTE BIT MASK FROM DIAL NUMBER
        LAI 020B            START WITH MASK OF 00010000
RVCD1   RRC                 SHIFT MASK RIGHT UNTIL DIAL
        DCB                 NUMBER BECOMES ZERO
        JFZ RVCD1
        LBA                 SAVE BIT MASK IN B
        LHI 200B            INITIALIZE TRIAL VALUE IN H
        LLH                 INITIALIZE TRIAL BIT IN L
RVCD2   LAH                 GET CURRENT TRIAL VALUE IN A
        XRL                 FLIP CURRENT TRIAL BIT
        OUT XMOV            SEND TO DIGITAL-TO-ANALOG CONVERTER
        LAA                 WAIT FOR 741'S TO SETTLE
        LAA
        INP GINP            READ COMPARATORS & SWITCHES
        NDB                 MASK TO GET CHANNEL OF INTEREST
        JTZ RVCD3           JUMP IF TRIAL TOO HIGH
        LAH                 RETAIN TRIAL IF TOO LOW
        XRL
        LHA
RVCD3   LAL                 SHIFT TRIAL BIT RIGHT 1
        RRC
        LLA
        JFC RVCD2           DO ANOTHER ITERATION IF ALL BITS NOT TRIED
        LAH                 LOAD FINAL RESULT INTO A
        RET                 RETURN
```

8008 VS 8080 VS IMP-16 - WHICH MICROPROCESSOR FOR YOU?

by Hal Chamberlin

At this time there are three microprocessor chips or chip sets readily available to the hobbyist: the 8008, the 8080, and the IMP-16. The first two were pioneered by Intel and the last is a National Semiconductor invention. Chips and/or kits utilizing each of the three microprocessors are available from at least two sources catering to hobbyists as of this writing. This level of availability and popularity is not even approached by other microprocessors, therefore this discussion is being confined to these three.

Comparing computers is like comparing people; the conclusions depend on the application, the circumstances, and personal preference. The comparisons made will be based on use of the microprocessor as a general purpose computer. For our purposes a general purpose computer is one which has read-write memory for the bulk of its storage, which is expected to run a variety of programs, and for which the end use is the development and execution of programs written by the user. General purpose computers are also expected to be able to control a variety of input-output equipment. Instruction sets will be compared on the basis of assembly language programming. Speed will be compared on the basis of the time necessary for the machine to complete a non-trivial task. Complexity will be compared on the basis of ease of understanding microprocessor operation as well as the sheer number of parts required to implement a system. Finally, cost will be compared on the basis of minimum systems capable of assembling programs for themselves given the existance of suitable I/O devices.

Before getting into comparisons, we will take a brief look at the leading features of each microprocessor. Then the comparisons will be made in each performance area elaborating on individual features as necessary.

The 8008 was the first microprocessor to be introduced and the first to be available to the hobbyist. It has an 8 bit instruction and accumulator length. There are essentially only two memory addressing modes, immediate, and zero displacement indexed. Subroutine and branch addresses are full length absolute, allowing branching anywhere with one instruction. Subroutine return addresses are saved on an internal 8 level stack which puts a 7 deep restriction on subroutine nesting. Much of the instruction set power is derived from the six additional 8 bit index registers which may count, save, or address memory. The maximum directly addressable memory is 16K bytes, in addition, 8 input and 24 output devices may be directly addressed with one byte instructions. CPU speed is modest ranging from 20 microseconds for a register operation to 32 microseconds for a memory operation to 44 microseconds for a jump or call. A selected chip, the 8008-1, reduces these times to 12.5, 20, and 27.5 microseconds respectively. A single level of interrupt is provided but external hardware is necessary for complete status saving during interrupts. Interfacing the chip to the rest of the system is fairly involved and requires from 20 to 70 TTL packages depending on the system performance desired. The lower figure will barely function while the higher one includes a console, complete interrupt system, and dynamic memory interface with direct memory access capability. Most of the interfacing complexity can be blamed on overzealous designers trying to make-do with an 18 lead package. Present cost to the experimenter ranges from $40 to $80 with the "dash one" version bringing roughly 50% more.

The 8080 is Intel's sequel to the 8008. Basically it has more of everything. The instruction set contains all of the 8008 instructions making it upward compatible at the assembly language level. Major additions to the instruction set include direct load and store of the accumulator, double precision (16 bits) add and increment for address calculation, and a pushdown stack of indefinite length in memory thus allowing unrestricted subroutine nesting. Addressable memory has been increased to 64K bytes and addressable I/O devices have been increased to 256 inputs and 256 outputs at the expense on 2 byte I/O instructions. Execution speed has been considerably improved also. Register operations take 2 microseconds, memory operations require about 3.5 microseconds, and subroutine calls consume 8.5 microseconds. Interrupts work the same way as on the 8008 but everything required for complete status saving is provided as well as an interrupt enable/disable flag. Interfacing an 8080 is generally regarded as being simpler than interfacing an 8008. There is only a slight improvement in the minimum system, about 15 chips, but a full-bore system may be cut in half to 35 chips. The 40 lead package allows a seperate 16 bit address bus and 8 bit data bus, as well as simplified timing and control. Present cost to the hobbyist is about $160.

The IMP-16 is one of the older microprocessors and is still the only one with a 16 bit wordlength. The programmer is supplied with four 16 bit accumulators and a 16 word stack. The instruction set is typical of many 16 bit minicomputers, and in many ways resembles that of a NOVA. Four general address modes are provided, base page direct, program counter relative, and indexed using either accumulator 2 or accumulator 3. In addition, LOAD, STORE, JUMP, and CALL can be indirect addressed using any of the addressing modes to get to the address pointer. Two memory modification instructions are provided, ISZ (Increment memory, Skip if Zero), and DSZ which allows much counting and indexing to be done in memory freeing the registers for arithmetic. The stack is used for subroutine return addresses but can also be used for saving registers and status. A unique feature is the availability of an extended instruction set chip which provides automatic multiply, divide, double word add and subtract, and byte manipulation. The CPU can address 64K words but this should be held to 32K if the byte instructions are used. The I/O instructions can also address 64K devices. Another unique feature is that several bits of input and output are provided by the microprocessor itself making communication with a teletype possible without any interface at all. Speed is good ranging from 4.2 microseconds for a register operation to 7 microseconds for a memory operation. A multiply takes about 160 microseconds which is still considerably faster than a software routine would be. The IMP-16 provides two priority levels of interrupt and all of the hardware necessary for complete status save/restore. Interfacing is conceptually simple and requires 25 to 50 packages depending on system sophistication. Part of this number is due simply to the fact that 16 bits are to be handled rather than 8. The microprocessor is in the form of five 24 lead packages which for the most part are simply wired in parallel. The extended instruction set resides in a sixth package. Present cost of the standard chip set is about $160. The extended instruction set chip is available only from National at this time for $80.

One of the most important performance areas of a microprocessor is the instruction set. A good instruction set should be well organized so that it is easy to learn, powerful so that complex routines can be coded with a small number of instructions, memory efficient so that complex routines require only small amounts of memory, and time efficient so that only a small number of memory cycles is necessary to complete a task. In addition, performance should be equally high on both character oriented tasks and numerically oriented tasks.

Instruction set organization is best on the 8080 closely followed by the 8008 with the IMP-16 being somewhat disorganized. Consequently, the beginner will find the 8008/8080 the easiest to learn. Experience has shown that beginners prefer simple instruction sets and that they retain a certain "fondness" for their first machine long after they have graduated into much more sophisticated endeavors. The experienced programmer however should experience little difficulty keeping the little quirks, distinctions, and special cases straight when working with the IMP-16.

Instruction set power is best on the IMP-16, followed by the 8080 with the 8008 a distant third. Based on actual experience, it may require as few as one half as many IMP-16 instructions to program a task as 8008 instructions. The 8080 falls about midway between the extremes. There are many reasons why the IMP-16 is superior. Memory addressing is much more flexible due to the four addressing modes and indirect addressing capability. An additional advantage is that the arithmetic word length is the same as the address length. Since the return addresses are put on a stack in all three machines, multiple entrypoint subroutines are easy but the IMP-16 also allows multiple return points (return to CALL+1 on error, CALL+2 otherwise, etc.) with no additional instructions. The 8080 is a big improvement over the 8008 because registers may be saved on the stack when they are used by a subroutine and then restored unaltered upon return. This allows subroutines to be called as needed without regard to which registers they may destroy. Note however that this capability may be added to the 8008 quite simply. The direct load and store instructions of the 8080 also reduce the number of lines of code in a program.

Memory efficiency of the instruction set is best on the IMP-16 but is closely followed by the 8080. The 8008 is not as bad as might be presumed but is definitely inferior. In terms of numbers, the 8080 may require 10 to 15 percent more memory bits and the 8008 20 to 40 percent more. Note that these figures are based on optimized programs written by experienced programmers. The spread can be much greater with inexperienced programmers or hastily written programs. It is also interesting to note that instruction set organization and memory efficiency are usually conflicting requirements. This is because many of the lesser used possible operation combinations have been culled from a memory efficient set in order to reduce the number of bits required to encode the instruction. Implied operands are also utilized in order to free up bits for other uses. Experienced programmers are able to plan ahead and avoid having these restrictions become restrictive. The 8008 and 8080 are as good as they are because many of the instructions are a single word (8 bits) long whereas the minimum instruction length in the IMP-16 is 16 bits. This is somewhat offset by the three word (24 bit) instructions of the 8008 and 8080 which in most cases would only require 16 bits in the IMP-16. A fringe benefit of high memory efficiency is that the shorter programs will load faster regardless of the loading method.

Time efficiency is by far the best on the IMP-16 with the 8008 a distant second and the 8080 a slightly poorer third. On a classic minicomputer, a machine cycle was the same as a memory cycle in most cases. As a result, a time efficient instruction set meant a faster machine without faster hardware. Microcomputers on the other hand may have very few of their machine cycles being memory cycles. As a result, time efficiency may have little relation to actual machine speed but does represent the potential speed with an optimized CPU. Time efficiency can be important in multiprocessor systems with a shared memory where more memory cycles increases the probability that a CPU will have to await its turn. The IMP-16 has a high time efficiency mainly because twice as much data is fetched in each memory cycle. Further improvement is due to the instruction set power, requiring fewer instructions to be fetched. The 8080 has poorer time efficiency then the 8008 mainly because the stack is in memory. A subroutine call, for example, requires 5 memory cycles, 3 to fetch the instruction and two to stack the return address.

Historically some minicomputers were better at handling character oriented tasks and others were well adapted to number crunching tasks. Microcomputers are no exception. Most micros have been optimized for character handling because of expected high usage in terminals and the 8008 and the 8080 belong to this class. The IMP-16 on the other hand is much better at numerically oriented tasks and was aimed more toward machine tool control and industrial monitoring. Interestingly, use of the extended instruction set on the IMP-16 greatly improves both character handling and arithmetic capability.

One performance area of interest to hobbyists is the suitability of a machine for running a BASIC system. The IMP-16 and the 8080 are about equal in their ability to compile BASIC quickly but the IMP-16 without the extended instruction set may execute BASIC twice as fast. This is due mainly to the all floating point arithmetic that BASIC requires. The extended instruction set may double the speed again if a lot of multiplies and divides are done. The 8008 can of course run BASIC also but compile and execution speeds are likely to be one tenth of the 8080.

One other property of an instruction set is the ease with which it may be assembled, either by hand or with an assembler program. In this respect, the 8008 comes out on top with the 8080 next and the IMP-16 last. Use of the mnemonics and format recommended by the manufacturer is assumed in making this comparison. 8008 code is easy to hand assemble because the octal notation used corresponds to the various fields in the instruction word. Assemblers for 8008 code can also be quite simple because instructions require at most one operand and very few instruction formats exist. Further simplification results from all addresses being absolute and all mnemonics being three characters. 8008 assemblers run on an 8008 can be as small as 2.5K bytes for a limited implementation but 4K bytes is more realistic when providing an easy to use assembler. If the hexadecimal notation recommended by Intel is used with the 8080, hand assembly is definitely more difficult. The assembler also has a tougher time with the two operand format and other niceities defined for the 8080. The Intel version of the 8080 assembler requires 8K bytes but it should be noted that it provides macro capability. Hand coding and assembling for the IMP-16 is harder yet due mainly to relative addressing considerations and a wider variety of instruction formats. National's version of the assembler requires 4K words and can produce relocatable object code and handle external symbols.

Speed in a hobby computer system can be a two-edged sword. A high speed microprocessor requires higher speed in other system compontents such as memory in order to realize its higher speed. An 8008 for example can run at full speed with memories as slow as 3 microseconds access but the 8080 will have to wait on memories slower than 520 nanoseconds and the IMP-16 requires 420 nanoseconds. If the ready line is used on the 8008 and 8080 to permit the use of slower memories, the wait will be in increments of whole machine cycles which is 4 microseconds on the 8008 and 500 nanoseconds on the 8080. Thus if memory is a tad slow, one cycle will be added to each three cycle memory access sequence slowing the system down an average of 25 percent to 30 percent. The IMP-16 does not have a ready line, rather the user stretches one of the clock periods in a cycle long enough to permit memory access. This scheme has the advantage that the stretch can set to the exact amount needed. The higher time efficiency of the IMP-16 instruction set will greatly reduce the performance impact of a slow memory as compared to the 8080.

The ranking on interface complexity is 8080 (least), IMP-16, and 8008. The comparison is based on sophisticated general purpose implementations having a complete I/O interrupt facility, software console using ASCII I/O, and a generalized input/output/memory bus allowing simultaneous direct memory access without affecting the CPU. The ranking is the same whether parts count or conceptual complexity is being considered. The difference between the 8080 and the IMP-16 is primarily due to the wider word of the IMP-16 and less confusing discussion of chip interfacing in the Intel manual. The 8008 is just plain difficult to understand and interface correctly but once that is done, either by the user, a magazine, or a manufacturer, the system should operate just as well.

Software support is often a big issue among industrial users of microprocessors. Unfortunately, the majority of the software they are fighting over is unavailable to the hobbyist because of high prices. It is not unusual for a program such as an assembler to cost as much as a handful of microprocessors. 8008 and 8080 users can look to Scelbi and MITS for some software at reasonable prices even if they did not purcahse their machines from these sources. National has an excellent body of software for the IMP-16 but the package price is $200 for object tapes and source listings. The hobbyist will have to depend on himself, kit manufacturers, and publications such as TCH for most of his software in the near future. Ultimately, the level of software support will be directly proportional to the popularity of the microprocessor.

There are a number of other performance areas that are only of minor interest to hobbyists. Although there is a large spread in the maximum memory size, all three machines are likely to have ample addressing capability for the hobbyist's memory budget. The same may be said relative to addressable I/O devices. Power supply voltages and power consumption are also usually of minor importance. All of the microprocessors can be successfully operated from standard +15, +5, and -15 system supply voltages using simple, fail-safe, zener regulators. Package size and pinout are unlikely to be factors in hobbyist use.

This brings us to a comparison of overall system cost. First, the spread in chip cost is roughly from $50 to $150 , so the spread in system cost would be $100 at most. An 8008 requires more interfacing circuitry however which reduces the spread somewhat. After enough memory to do assemblies or run BASIC and a few I/O devices are added, the $75 difference left may be small compared to the total investment. Nevertheless, an 8008 system will be the least expensive followed by an 8080 system closely followed thereafter by an IMP-16.

Which microprocessor for you? The answer still depends on the application, circumstances, and personal preference, but hopefully the decision can be made with more authority after reading this article. TCH will be giving space to all three systems in the future and any others that can match their availability and popularity.

Each month TCH will publish the names and addresses of new subscribers who so desire, however the list will be limited to one page per issue to conserve space. We hope this service will aid people in finding assistance and friends with a common interest.

John E. Stauffer
3415 Race St.,Apt. E
Drexel Univ.
Phila., PA 19104

Paul Satterlee, Jr.
2710 Sevier Ave. Apt. 6-I
Knoxville, TN 37920
Ph. 615/577-2209

Tom Parquette
116 Sanford Ave.
Clinton, NY 13323

Lawrence R. Rowland
515 E. Call St., Apt. 1
Tallahassee, FL 32301
Ph. 904/222-6056

Wayne E. Davies
24121 Marathon St.
Mission Viejo, CA 92675
Ph. 714/830-9627

L. Scheppmann
P. O. Box 65
Descanso, CA 92016

Al Sardo
PO Box 6678
San Jose, CA 95150

E. N. Noren
111-11033-127 St.
Edmonton, Alberta
Canada

Jim McGinness
PO Box 9628
San Diego, CA 92109
Ph. 714/272-4878

Allyn Rothman
Box 24019
Jerusalem, Isreal

Dana D. Craig
67 Hill Sreet
Norwood, Mass. 02062
Ph. 617/762-4221

Paul R. Costella
3859 Brighton Drive
Calgary, Alberta
Canada T2L 1G7
Ph. 403/282-3107

Ed Berlin, Jr.
Skunk Hollow Rd.
Huntington, NY 11743
Ph. 516/427-0047

Andy Barlak
Box 88
Sointula, British Columbia
Canada VON 3E0
Ph. 604/973-6910

Jon Tyler
5625 John R. Rd.
Troy, Mich. 48084
Ph. 313/879-9335

Alan F. Neel
1435 Judson
Boulder, CO 80303
Ph. 303/499-0656

Robert W. Cook
25W178-39th street
Naperville, Ill. 60540
Ph. 312/357-3021

Phil Macgregor
2804 Broadway Ave.
Pittsburgh, PA 15216

Don Davis
196 Dell Ave.
Pittsburgh, PA 15216
Ph. 412/561-7963

J. Scott Williams
P.O. Box 932
Bellingham, WA 98225

Jeb Boswell (WIGJU)
29 Kenilworth Street
Newton, MA 02158
Ph. 617/969-2740

Gregory Ehrler (K2ALX)
105-53 87th St.
Queens, NY 11417
Ph. 212/848-3339

Buster Killion
2773 Winrock
Altadena, CA 91001
Ph. 213-798-2977

O. F. Hamm
4751 Louisiana Ave.
St. Louis, MO 63111

Henry S. Birdseye, Jr.
401 Tennessee NE
Albuquerque, NM 87108
Ph. 505-266-1322

William C. Parrish
127 Winfield Ave.
Jersey City, NJ 07305

John L Marshall
Box 242
Renton, WA 98055

William A. Smyth, Jr.
1216 Bluefield Dr.
Columbia, SC 29210
Ph. 803-772-4795

Lawrence J. Stein
151 Kline Boulevard
Colonia, NJ 07067

Harvey A. Sugar
4301 57th Ave, Apt. 1
Bladensburg, MD 20710

Robert Swartz
195 Ivy Lane
Highland Park, IL 60035

Robert W. Thomas
910 Sonman Ave.
Portage, PA 15946
Ph. 814-736-9656

J. M. Whitehurst
204 S. Elm St.
Greenville, NC 27834
Ph. 919/758-2488

Robert Q. Childress
127 West 5th St.
Cookeville, TN 38501

Lt. Col. Stanley S. Jurgielewicz
6414 Noble Drive
McLean, VA 22101

James A. Rawlings
211 Milo Place
San Ramon, CA 94583

Stephen Ciarcia
41 Hilltop Drive
West Hartford, CT 06107

Norman F. Stanley
P. O. Box 723
Rockland, ME 04841
Ph. 207/594-7053

Daniel H. Baker
1112 Florida Avenue
Bristol, TN 37620
Ph. 615/968-8400 (office)
615/968-9261 (home)

James E. Smith, Jr.
3242-G Calumet Dr.
Raleigh, NC 27610
Ph. 919/833-7879

T. H. Pearson
648 Sherwood Forest Blvd.
Baton Rouge, LA 70815
Ph. 504/275-8190

David Zethmayr
413 S. 8th Ave.
Lagrange, IL 60525
Ph. 312/354-4193

Jerry W. Ward
1125 Gorsuch Ave.
Baltimore, MD 21218
Ph. 301/889-1608

Wm. J. Schenker, MD
Apt. 4, 660 Tempe Court
Pleasant Hill, CA 94523
Ph. 415/687-8804

Frank R. Mallalieu
Encom Prods. Co. Inc.
P. O. Box 300
Oxford, PA 19363
Ph. 215/932-9265

Neil A. Benson
10040 Nicollet
Bloomington, MN 55420
Ph. 612/372-7581 day
612/881-6972 evening

Richard C. Bemis
402 So. Humboldt
Denver, CO 80209

J. Sikorski III
1410 Lathrop
River Forest, IL 60305
Ph. 312/649-8028

Richard P. Brennan
601 So. Knight
Park Ridge, IL 60068
Ph. 312/696-4318

Clifford Wetmore
2184 N. Rosewood Lanee
St. Paul, MN 55113

Don Allison
Box 1685
Craig AFB, AL 36701
Ph. 205/875-7382

James M. Lamiell
1615 Colorado Blvd.
Denver, CO 80220
Ph. 303/388-7531

David F. Stout
717 Saranac Drive
Sunnyvale, CA 94087
Ph. 408/736-0846

Carl Geiger
7160 W. Genesee St.
Fayettville, NY 13066
Ph. 315/637-8443

Dan Derbick
6622 Montrose
Chicago, IL 60634
Ph. 312/867-9059

R. G. G. Guenther
1100 Main St. W.
Hamilton, Ontario
Canada L8S 1B3
Ph. 416/522-4750
or 416/529-5632

Lee C. Hanson
51CES, Box 648
APO, San Francisco 96366

George B. McConnell
1120 Ware St.
Vienna, VA 22180
Ph. 703/938-5998

SURPLUS SUMMARY

To start off this month, we want to repeat from an ad seen in a ham magazine. For sale, RCA 70/15 computer. If our information is correct, the 70/15 is a small (8K bytes) computer from the mid 60's with an instruction set similar to the IBM 360 series. Its certainly not a microcomputer, but then microcomputers are not for everyone. If you are interested, write to:

Douglas Craton
5625 Balfrey Drive
West Palm Beach, FL 33406

Maury Goldberg of Mini-Micro Mart has asked TCH to notify our readers who may have written him that their letters have not gone astray. He has simply gotten more mail than he can quickly process. Matters have been complicated at Mini-Micro Mart by the fact that Micro-Systems International, a Canadian firm which second sources Intel parts and which supplied some of Mini-Micro Mart's items, was closed down by it's parent company Northern Electric Ltd. Despite the fact that the Mart's mailbags will be further loaded, TCH wishes to mention that they have several items of interest. First item is a nice encoded ASCII keyboard, with parity and two key rollover, for $40.00. Next the Mart now has UART's for $9.95. And last the Mart has power supply components and kits at reasonable prices. For a flyer send a SASE to:

Mini-Micro Mart
1618 James Street
Syracuse, NY 13203

Since last issue TCH has run accross a new source of goodies. This source usually has only a few of each item but they are well worth memtioning and they do send flyers. In flyer #91 they have listed 7BP7 CRT's. This 7" CRT will work fine for a graphics display. There are only 4 left but they are available for $10 including shipping. Also a one only item, an eight level paper tape punch of the BRPE type (parallel interface, medium speed) for $29, catalog part #4097.

BVE Enterprises
Box 73
Paramus, NJ 07652
Phone 201/265-7075
Note: This number is answered by an answering machine when no one is around.

One of our subscribers, Al Sardo, sent TCH a flyer with some interesting items. He has 1702A erasable 256 byte PROMS for $13.00 and Signetics 2516 vertical scan character generators for $3.00. Also available, 4004, 8008, and 8008-1 microprocessor chips and PROM programming services. Write to: 4 - 2, 5-8 25

Al Sardo
2032 S. W. Expressway
San Jose, CA 95126

Finally, someone with a reasonably priced printer for your system. Wilcox Enterprises is offering used, but guaranteed, Creed model 75 teletypes. The Creed model 75 is a 5 bit Baudot machine which has a printer, keyboard, and paper tape punch but no reader. The unit prints at 100 words per minute (10 cps) and prints 71 characters per line on 8.5 inch wide paper. The price of $125 also includes a manual on the TTY, parts and plans for an 8008 interface, and program listings for an 8008 support package.

Wilcox Enterprises
25 W 178 - 139th Street
Naperville, IL 60540
Ph. 312/357-3021

And now last but not least, TCH is proud to announce that we are making reasonably priced 2102's available to our readers. The 2102 is a 1K static RAM which operates from a single +5 volt supply and is fully TTL compatible. This is the same RAM used in the TV Typewriter II and several microcomputer kits. The parts we offer are MM2102N's which were obtained through normal distributor channels from National Semiconductor Corp. and are guaranteed by TCH for 90 days which is the same guarantee National extends to us. Prices are $4.50 each or 8 for $35.00, limit 32 per customer per order.

The Computer Hobbyist
Box 295
Cary, NC 27511

CLASSIFIED ADS

There is no charge for classified ads in TCH but they must pertain to the general area of computers or electronics, and must be submitted by a non-commercial subscriber. Feel free to use classified ads to buy, sell, trade, seek information, announce meetings, or for any other worthwhile purpose. Please submit ads on separate sheets of paper and include name and address ana/or pnone number. Please keep length down to 10 lines or less.

WANTED: 1101 memory chips. Also have experience in area of computer power supplies and application info on components. To request advice or information send SASE. William C. Parrish, 127 Winfield Ave., Jersey City, NJ 07305

BENEFIT from Amateur Computer, quantity purchasing power. We will be making a quantity buy of 2102 1K sataic RAM's 650 nsec. or faster, May 31st. We expect the price to be between $4.25 and 5.25 each. Write for more information, please enclose self-addressed stamped envelope. James Fry 4249 N. Lockwood, Toledo, OH 43612

FOR SALE: Elliot high speed paper tape reader. 1000 char. per sec. 5, 6, or 8 level tape, start/stop within 1 char. In excellent condition with book. Don Davis, 196 Dell Ave., Pittsburgh, PA. 15216, Ph. 412/561-7963

SERVICE AVAILABLE: I Am setting up a 5203 PROM programming service if anyone is interested. Contact: David Yulke, 121 Liberty Ave. Selden, NY 11784, Ph. 516/698-0551

WANTED: 35 Mhz or higher dual beam or dual trace oscilloscope with 5" CRT & plug-in amplifier(s). Scope must be in working condition and CRT must not have any trace burns on screen. Send all particulars including condition, model make, age if known, available accessories and documentation, and your price. Give your address and phone number. If I am interested I will contact you within 15 days. Philip M. Lohr, 3917 Flowerfield Rd., Charlotte, NC 28210

FOR SALE: 18x1K Ampex complete core memory system $100. Will supply schematic for examination. Unit appears new. I've never used it. Write Lee Hanson, PSC Box 648, APO S.F. 96366

WANTED: Info on Burroughs dual main frames. Need "Series L Tech Manual". Try 'Tech' of 1969 to '71 and probable #'s B2500, B3500, B5500, 6503, 6504, 6506, 7504, 7506 plus more descriptive information available. Will pay necessary expenses & accept toll calls relating to information. Karl Shattuck, 5 N 482 Pine Lane, Wood Dale, IL 60191, Ph. 312/766-0840

FOR SALE: 2 IBM 727 Tape Drives for sale -$200 ea. with diagrams for converting to solid state & maintenance manual. Also have Century Data Model 110 Floppy Disk Drives, complete -$275 ea. Buster Killion, 2773 Winrock, Altadena, CA 91001, Ph. 213/798-2977

FOR SALE: Several Wangco Model 7 Tape Decks , 12.5 ips, 800 cpi, read after write, 7 or 9 channel, unused. Make offer. John Marshall, Box 242, Renton, WA 98055

WANTED: Have parts for RE TV Typewriter. Would like to find someone to assist on completion and debug. George Haller, 1500 Galleon Dr., Naples, FL 33940

THE COMPUTER HOBBYIST
Box 295
Cary, NC 27511

ADDRESS CORRECTION REQUESTED

FIRST CLASS MAIL

THE COMPUTER HOBBYIST

Volume 1 Number 5 March / April 1975

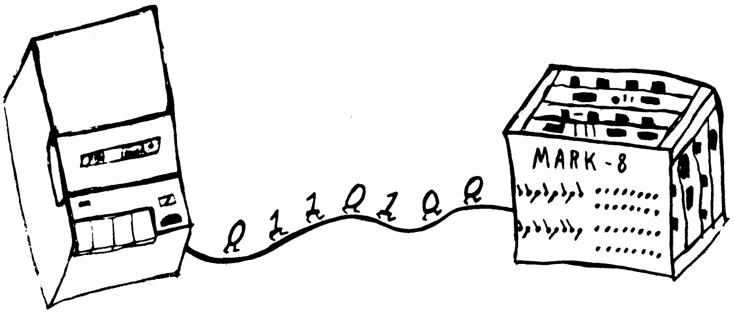
With this issue TCH recognizes the fact that publication is hopelessly behind schedule. Consequently this issue is designated the March/April issue and the expiration date of all subscriptions is extended one month.

To change the subject, things have begun to really move in the area of computers for a hobby. Chips and kits abound and prices are coming down constantly. This past month another first, at least as far as TCH knows, happened. A club was formed for people interested in amateur computers. They haven't chosen a name yet but 32 people showed up for the first meeting in Gordon French's garage in Menlo Park, CA. Fred Moore seems to be organizing things. If you live in the area contact Fred at 2100 Santa Cruz Ave., Menlo Park, CA 94025.

In the interest of promoting computers as a hobby, TCH will be going to the world's largest get-together of electronics buffs, the Dayton, Ohio hamfest. The Dayton Hamfest will be held at the HARA Arena on April 25, 26, and 27. TCH will be there Saturday and Sunday with a complete system including our graphics display and cassette system (see articles this issue). See you there!

One last note, back issues of TCH are available on a continuing basis. The cost is the same as for regular issues, 50 cents each. If possible, when ordering, please include issue numbers desired, caution there was no November 1974 issue. October 1974 was our trial and first issue. December was the first regular issue.

TCH AUDIO CASSETTE DATA RECORDING STANDARD
PART 1 by HAL CHAMBERLIN



According to the survey taken in the January issue, the one hardware article topic most asked for has been data recording on audio cassette tape recorders. This state of affairs is interesting in that it indicates widespread dissatisfaction with the numerous techniques and systems that are now avaiable. It has been our experience that often a hobbyist will read a description of an audio cassette system and then reject it without actually trying it out. This is a sad but true situation. Some publications have printed a number of different techniques and let their readers select as they please. Finally, there are as many different cassette recorder interface kits as there are companies producing them. Standardization?

In this pair of articles, TCH will develop an audio cassette recording standard, publish proven hardware and software utilizing the standard, and the fully commit itself to using and promoting the standard. How do we plan to be successful in doing this when so many others have failed? The first step is understanding the dynamics of standards acceptance. A prerequisite for acceptance is technical competence of the standard, that is, performance and reliability of equipment using the standard must be on a par with any existing equipment. The chances for acceptance are even better if the standard actually advances the state of the art. Once this prerequisite is met, acceptance is directly proportional to the benefit the user can anticipate if he adopts the standard. In the commercial world, this anticipation of benefit is created mostly by the marketing activities of the standards makers. Additional factors that can ease the marketing effort required are simplicity, low cost, and compatibility at widely varying performance levels.

TCH has a technically competent standard that meets or exceeds the performance of any system we know of in actual use. It is simple both in concept and execution, thus low in cost. Upward compatibiltiy with a higher performance digital cassette system is possible. The user benefits from adopting the standard will be numerous. All programs published in TCH will be available in both commented source form and in object form on standard cassettes. Plans are to cover the 8080 and IMP-16 as well as the 8008. Other programs which are discussed in TCH and which were submitted in machine readable form will also be available. TCH will also be active in looking for public domain programs. We will support the hardware interface with detailed functional descriptions and low cost printed circuit boards. Read and write software will be published and described in detail. Plans are to offer this software as a single 1702 or 5103 PROM custom programmed as to the address page numbers and I/O port numbers desired by the user. We can also burn the cassette software into a subscriber's PROM of the above types for $1.00 and a self addressed stamped box. A bootstrap load program requires only 32 bytes and can be burned into a single 8223 bipolar PROM in lieu of the complete package in the larger ROM. This offer applies only to the TCH standard cassette software package. Besides promoting the standard, we hope this offer will soften the impact of MIL's withdrawal from the marketplace on those who were planning to use their MONITOR-8 ROM and associated cassette tape system. We realize, of course, that ours is not the only audio cassette system that works satisfactorily but we also realize that the time for standardization is overdue and we do think that our system is currently the best available.

The audio cassette recording standard consists of three distinct parts. These are the recording technique, the physical data format, and the logical data format. In this article the first two parts will be developed and described with the logical format discussion to appear in the next issue along with the formal PC board and PROM offer.

In choosing a recording technique and accompanying modulation method, we decided to first take a closer look at the properties of the communication channel, in this case an inexpensive tape recorder. After these properties were understood, we could design a modulation method that utilizes the predictable properties and rejects the effect of the unpredictable properties of the channel. In this way an inherently high performance and reliable recording technique would be developed which, hopefully, could then be simplified without sacrificing too much of the technique's potential.

An audio tape recorder presents a relatively noise-free channel of fairly wide bandwith. Additive noise is 40 or more dB below the signal at frequencies above a few hundred hertz increasing 6 dB per octave at lower frequencies. The noise amplitude distribution is Gaussian indicating a lack of noise bursts. Rolloff at the band edges of approximately 100 Hz and 5 kHz is smooth and gradual. The smooth rolloff implies good transient response and fairly accurate reproduction of signal waveshapes. Timebase distortion however is quite severe in inexpensive recorders. To begin with, the variance in average tape speed is wide, as much as 10% due to loose manufacturing tolerances and other factors such as weak batteries. To this 10% is added low frequency wow due to variations in motor loading as the tape spools rotate and medium frequency flutter due to eccentric or crooked capstans. Wow and flutter can reach several percent before they are objectionable in speech recording which is what these recorders are intended for anyway. Higher frequency flutter due to tape and bearing stiction may be present to some extent. Gain distortion is also present and may be severe. A static gain error exists to the extent that recording and playback levels vary from run to run, recorder to recorder, and to the extent that battery voltage changes. Short-term gain variations due to tape coating variations are also experienced. Last but not least, many low-cost recorders have an automatic gain control circuit which is active during recording. These are very effective in maintaining constant signal level on the tape even though the input varies widely. They also can cause severe even-order harmonic distortion particularly at low frequencies.

Contrast these characteristics with those of a typical switched telephone line. The average signal-to-noise ratio is 20 to 35 dB depending on the quality of the circuit but independent of frequency. However the noise distribution is far from Gaussian. Noise bursts can occur frequently with amplitudes that may obliterate the signal. Cutoff at the band edges at 300 Hz and 3300 Hz is very sharp and steep. This sharp cutoff implies a very poor transient response. Ringing may last for several tens of milliseconds after a wideband pulse. Telephone lines however do not have any timebase distortion. Transmitted tones may experience an occasional slight frequency shift if they happen to pass through some of the older single-sideband type carrier equipment along the way, but this is not the general case. Gain errors are mostly static and do not change once the connection is established. The telephone system uses more elaborate automatic gain control circuits called companders. These act instantly and their effect at the transmitter is cancelled at the receiver by an expansion circuit. Being more expensive, their contribution to harmonic distortion is minimal and what they do introduce is mostly odd-order harmonics.

Given the broad differences between these two audio channels one would expect the optimum data transmission methods to be different also. This is in fact the case yet the vast majority of audio cassette data recording techniques published so far utilize methods originally designed for telephone line transmission. These methods use the stable timebase and/or amplitude characteristics of the telephone line as the carriers of information. They are also designed to avoid the band edges and associated ringing which would smear the bits. Some of the more recent cassette designs have recognized the poor timebase and amplitude characteristics of tape and have attempted to minimize their effects with brute-force techniques such as very wide frequency deviations, speed tracking circuits, etc.

TCH will be using pulse modulation in its audio cassette recording standard. Pulse modulation utilizes the good transient response of the recorder for the encoding of data. Speed variation tolerance can be made as wide as desired in trade for data rate without speed tracking circuits. Amplitude variation tolerance is also wide and can be traded for data rate. Note that no recording method can compensate for complete signal drop-out lasting over half a bit time, an error correcting code is required to do that.

Pulse modulation is perhaps the easiest of all modulation methods to understand. Figure 1 shows how a ZERO and a ONE are encoded. Simply stated, a ZERO is a single isolated pulse while a ONE is a pair of pulses far enough apart to be detected as seperate pulses but not so far apart as to be detected as two isolated pulses. It is easily seen that pulse modulation is self-clocking and is truly bitwise asynchronous. The tradeoff between bit rate and timing tolerance is also apparent. One limiting factor is the pulsewidth received from the recorder and processing circuits. The other is the minimum spacing between bits which is proportional to the maximum bit rate. An additional factor is the pulse amplitude. Too high an amplitude makes the pulses wider while too low an amplitude increases the chances of a missed pulse. The processed pulse width from inexpensive recorders is in the range of .2 to .5 milliseconds. If a data rate of 500 bits per second is desired, the pulse spacing for ZEROES would be 2 milliseconds and for ONES would be 1 millisecond.

Assuming the data is recorded at the proper speed, the speed tolerance on playback can be determined with a little graphical analysis as shown in Fig. 2. The data is decoded by the use of two fixed time delays. When the first pulse of a bit arrives, a .75 millisecond time delay is started. The pulse must decay within this period to avoid an error. After the .75 millisecond delay, a 1.0 millisecond "decoding window" delay is started. If a pulse or any part of a pulse is seen during this time, a ONE is decoded, otherwise a ZERO is assumed. Searching for the next bit pulse resumes immediately after the window time. The first drawing in Fig. 2 shows the timing relationships for correct playback speed. The second drawing shows the maximum overspeed that can be tolerated and the third drawing shows the maximum underspeed allowable. In preparing these figures, the maximum pulsewidth of .5 millisecond was assumed . Speed tolerance can be further improved at the expense of amplitude tolerance by lowering the playback level to obtain shorter pulsewidths.

While on the subject of speed tolerance we will examine the common 11 bit teletype code and show why it is particularly speed sensitive. In decoding the waveform one initially looks for the leading edge of the start bit, a ONE-to-ZERO transition. Once the transition is found, a clock is started that runs at twice the bit rate. We should be at the center of a bit on odd counts of the clock. The usual procedure is to verify the start bit on the 1 count, strobe the data bits at counts 3, 5, 7, 9, 11, 13, 15, and 17, and check the parity bit at count 19. An additional two counts are taken before the wait for another start bit is resumed. This is essentailly what a UART does when receiving. Figure 3 shows this action at ideal playback speed, 5% fast and 5% slow. Note that in the last two cases bit strobing is perilously close to the bit edges. Add to this an uncertainty in detecting the start bit edge and some bit smearing at high baud rates and the error rate might be unacceptable. One method that may be used to improve the speed tolerance is to cut down on the number of bits in a character. Reduction to 4 data bits and elimination of parity would allow + or - 10% variation.

One particular advantage of pulse modulation over the various forms of tone modulation is its adaptability to a higher performance digital cassette system. TCH is researching the potential of a PHI-DECK (Individualized Instruction, Inc., P.O. Box 25308, Oklahoma City, OK 73125) as a high-performance, completely computer controlled upgrade for audio cassette users. The accompanying controller/interface would be capable of reading and writing both audio cassette data and high density digital cassette data. This is possible because the recording technique is essentailly the same. Performance of the PHI-DECK system would be in the range of 800 bits per inch

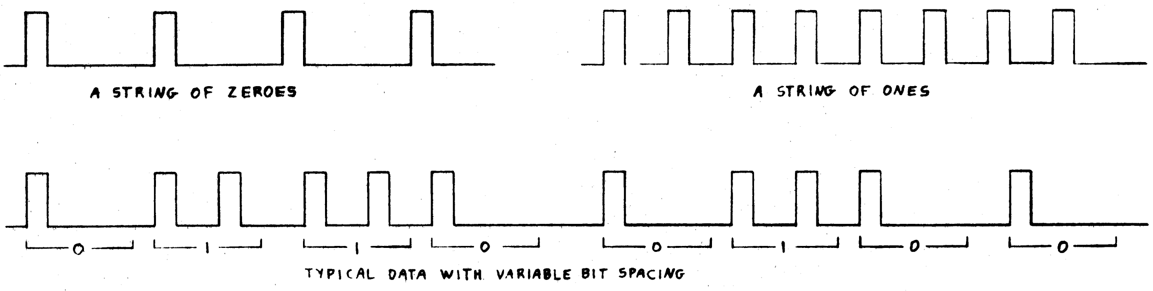


Figure 1 Pulse Modulation ZERO and ONE Encoding

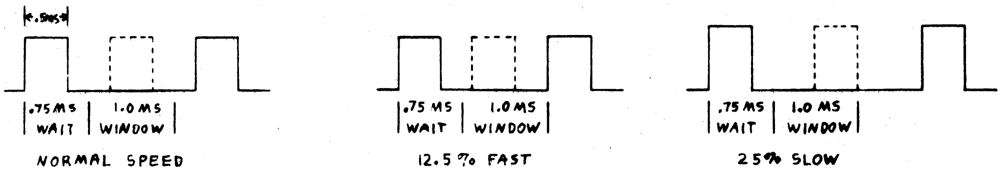


Figure 2 Pulse Modulation Speed Tolerance

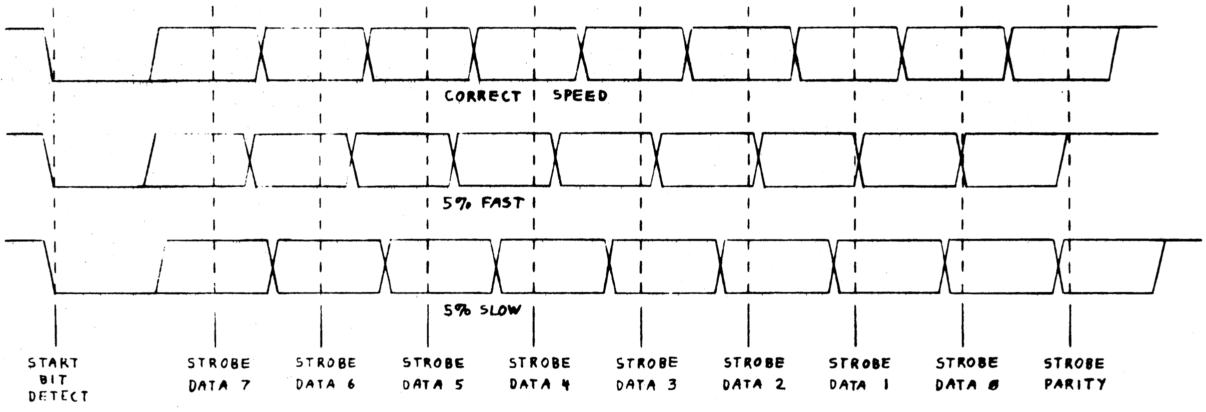


Figure 3 11 Bit Teletype Code Speed Tolerance

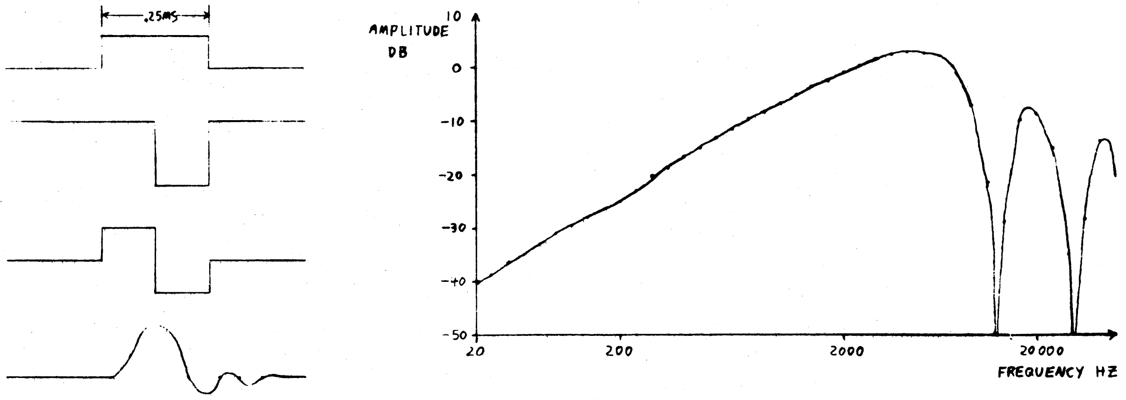
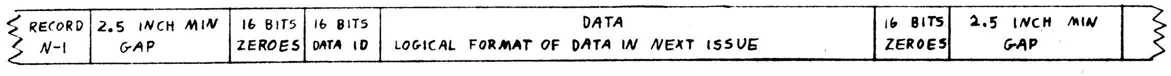


Figure 4 Pulse Waveform and Spectrum



RECORD N-1	2.5 INCH MIN GAP	16 BITS ZEROES	16 BITS DATA ID	DATA LOGICAL FORMAT OF DATA IN NEXT ISSUE	16 BITS ZEROES	2.5 INCH MIN GAP

←—— DIRECTION OF TAPE MOVEMENT ONE TRACK SHOWN

and 5000 bits per second. Two of these decks could support a nice, speedy operating system with text editor and assembler for considerably less money than a floppy disk system. Although you may not need a high-performance cassette system now, its nice to know that if you ever do, it will be completely compatible with all of your old tapes.

An interesting point is the fact that a very similar pulse modulation technique is used commercially on magnetic storage media. Prime examples are ANSI standard digital cassettes, 1600 BPI IBM 9-track tape, and floppy disks. Names such as "phase encoding" and "double frequency recording" are often associated with variations of pulse modulation. A requirement imposed in order to get higher bit densities is that the bit rate must be constant so that synchronous decoders can be used for greater reliability.

The performance of pulse modulation in an audio recorder depends heavily on the pulse shape used. One consideration is the spectrum of the pulses. Power below 1 kHz will serve only to blur the pulses on playback. A pulse shape that minimizes low frequency power is therefore desirable. Another consideration is the automatic gain control of the recorder. The inexpensive circuits used make their adjustment based on the peak signal amplitude in one direction, that is, a half-wave rectifier and peak holding capacitor. A symmetrical pulse is desirable to insure that the AGC circuit sees the actual amplitude of the signal input. Figure 4 shows the pulse shape chosen, a finite width doublet pulse, and its associated power spectrum. As can be seen, the majority of the power is centered at 4 kHz with a smooth rolloff on the lower frequencies. The doublet pulse is easily generated from two ordinary rectangular pulses as shown. Also shown is a typical playback waveform from an inexpensive recorder.

The second part of TCH's standard is the physical data format on the tape. Factors influencing the choice of format include tape start-stop distance, AGC response time, random pops and voice messages between data records, and throughput.

Some space is required between data blocks on the tape so that the recorder can be stopped at the end of a data block for processing and then be restarted and up to speed before the beginning of the next block. It is desirable that this space be large to accomodate all possible recorders. It should be small however for a high throughput rate on short records such as lines of text. Oddly enough, a cheap recorder starts and stops faster than an expensive one due to a lighter or non-existant capstan flywheel. Our experience at TCH has been that the real limiting factor is the "turn-on time" of the amplifiers in the recorder. This time lag is due to charging of bypass and coupling capacitors in the cheap circuitry. One solution if this is a problem is rewiring the microphone switch so that only the motor is controlled while the amplifiers are left powered up. In due consideration of the above factors, a minimum gap between records will be specified as 2.5 inches. This can be implemented with a 1.5 second delay between motor on and start writing and a .5 second delay after motor off to allow stopping. This should be ample for most recorders.

Since pulse modulation is self-clocking and asynchronous by bit it should be possible to simply write and read data as a bit stream combining groups of 8 bits into bytes starting from the first bit. However utilization of some common commercial techniques offers a significant improvement in accuracy and noise rejection. In recorders with AGC, the gain will be pumped up to maximum prior to the writing of bits. The first couple of bits written while the gain is adjusting itself may not be of as good a quality as the rest of the bits. Additionally a speed tracking decoder, if used, needs a few bits to initially adjust itself. Thus a standard data record will have at least 16 ZEROES prefixed onto it. These will be called a "sync pattern" and allow time for AGC action and decoder synchronization. Tremendous noise rejection is gained by the inclusion of a "data ID" pattern between the pattern and the data. The data ID is a unique sequence of 16 bits that must be seen before actual data transfer is started. The pattern chosen, 1000100110101111, is one that could be generated by a 4 bit feedback shift register acting as a pseudo-random sequence generator. It has the desirable properties of a leading ONE, and at least one occurrence of 000, 111, 0010, 1101, and 1010. Recognition of the ID marks off the first bit of true data and thus the byte boundaries even if some of the initial ZEROES are lost due to AGC action or initial garbage from the decoder. This method of marking the beginning of data is so effective that voice messages and even music may be interspersed between data records with no effect on the computer. It will simply skip over the trash since the chance of a valid data ID being seen is practically nil. TCH actually played a prerecorded cassette into the system with no effect. Another benefit is that if due to gross misadjustment the data is error prone, the 16 bit ID will probably be missed inhibiting reading until the problem is corrected. After the data bits, a 16 bit trailer of ZEROES will be written. The first couple of these are required by some decoding schemes in order to insure that the last actual data bit reaches the computer. The rest may be utilized as a sync pattern if read backwards is ever attempted.

The detailed discussion of the data bits will appear in part 2 on the logical data format. In general however they will be grouped 8 to the byte with error checking implemented using a standard 16 bit cyclic redundancy character (CRC). This is much better than parity or checksum techniques and requires very little more software. One item to note is that after the 48 bits of overhead for a record, all bits read are useful data bits. This means that for long records such as assembled programs, the data rate closely approaches the bit rate.

Listed in the appendix is a summary of the standard as developed so far. Look elsewhere in this issue for an interface circuit implementing the physical aspects of the standard usable on any computer.

APPENDIX

I. PHYSICAL STANDARD
    A. Modulation method - pulse
        1. Zero is an isolated pulse
        2. One is a pair of pulses
        3. Timing
            a. bit spacing 2 ms. min., 50 ms. max.
            b. second pulse of a one follows first by 1.0 ms.
        4. Pulse shape for recording - doublet
            a. high time .125 ms.
            b. low time .125 ms.
            c. total time .250 ms.
            d. to be recorded without tape saturation
    B. Data record format
        1. Interrecord gap
            a. 2.5 inches minimum
            b. voice message may be present in gap
            c. produced with 1.5 second minimum delay between motor turnon and write data, and .5 second minimum delay between motor turnoff and next operation
        2. Preamble
            a. 16 minimum bits of zeroes
            b. 16 bit data ID
                i. binary 1000100110101111
                ii. 2 byte octal 211 257
                iii. hexadecimal 89AF
        3. Data (details in next issue)
        4. Postamble
            a. 16 minimum bits of zeroes
    C. Tape tracks
        1. Half-track monaural is standard
        2. Outside tracks (channel A) if stereo must be used

II. LOGICAL STANDARD
    (developed in next issue)

LETTERS

We at TCH will publish a few of our more interesting letters each month along with comments by the staff.

Gentlemen:
Dick Bemis just gave me your inquiries about the cassette standard. Unfortunately, every designer views "his design is best", so there will not be perfect agreement. However, "independent judges" should consider the following when selecting a standard.

1. Minimum software requirement, especially on "Cassette to Memory" (since you have to "key in" this unless in ROM).
2. Simplest hardware circuitry - Costwise and tuning.
3. Most relaible operation - Noise immunity, dropout resistance, speed fluctuation proof, varying levels, tone purity.
4. Possible compatibility with other designs.
5. Number in use.

Perhaps your readers would make the best ultimate judges. Have each designer detail the merits of his system, or have an independent party parallel the qualifications. I have included my views on the following sheet.
. . . . . .

Dr. Robert Suding

Dr. Suding could not be more correct about each designer viewing his own system as best. The staff of TCH certainly thinks that Hal Chamberlin's system, which we are adopting and presenting as a standard is the best design currently available. While we must admit that on points 4 and 5 of Dr. Suding's letter we have no leg to stand on we feel that our merits in the first three considerations more than make up for it. A bootstrap loader for Hal's design will fit in 32 bytes or one 8223 PROM. The design uses 16 chips, which are common TTL and requires no tuning or timed program loops, as it is crystal controlled. Reliability is outstanding. To find out why, read Hal's article in this issue.

Gentlemen:
Enclosed is what I believe to be the program for your TTY Ripple Demo. Please look it over and let me know if I've done it right, and could you please explain the BDTAS TABLE. I started with 000B and went to 124B,015B,177B then to 040B and assigned the addresses sequentially. Was that correct? And for TTCSF I just put in 000 each time, is that correct?
If I am wrong could you not only answer me but put an explanation of your programming technique in the next issue of TCH?

Lee C. Hanson

Your work looks basically correct Lee, though I have not had time to check it 100%. Your handling if BDTAS data and TTCSF is correct. TTCSF was a reserved storage location for use by the program (DST means Define STorage), and its contents at startup time are unimportant. TCH chose not to print the object code because of space restrictions and because it would be different for each person's system due to memory locations and I/O port assignments. Another possible confusion factor is the fact that Intel switched mnemonic systems on the 8008 to be compatible with the 8080. TCH will be changing over soon, however we will not change from octal to hex despite Intel's preferences. In the future TCH will be able to furnish cassette tapes of assembler output listings, both source and object code.

SURPLUS SUMMARY

Finally, 8080 chips are becoming available at reasonable prices. TCH has found the following sources:

Mini-Micro-Mart
1618 James Street
Syracuse, New York 13200
8080 for $169.95

Electronic Discount Sales
138 N. 81st Street
Mesa, AZ 85207
8080 for $175

Also IMP-16 chip sets are available now from Poly-Paks for only $99. TCH will be running a series of construction articles on the IMP-16 soon.

Poly Paks
Box 942 B
Lynnfield, Mass. 01940

For people who are interested in a high-performance, completely computer controlled cassette tape drive, there is the Phi-Deck. This unit costs $95 or buy two for $189 and get tape heads and a motor control circuit board included. For detailed information write:

Individualized Instruction, Inc.
P.O. Box 25308
Oklahoma City, OK 73125

THE COMPUTER HOBBYIST
Founded October, 1974

Stephen C. Stallings - Managing Editor
Hal Chamberlin - Contributing Editor
Jim Parker - Contributor
Edwin Tripp - Photographer
Richard Smith - Programming Consultant

THE COMPUTER HOBBYIST is published monthly near Raleigh, N.C. The yearly subscription rate is $6.00 for persons in the U.S.A., U.S. possessions, and Canada. Back issues of TCH may be ordered at fifty cents per copy. Subscriptions in other countries considered individually with rate variances based on postal charges.
All correspondence should be addressed to:

THE COMPUTER HOBBYIST
Box 295
Cary, NC 27511

Members of the staff of TCH can usually be reached by phone evenings and weekends at either 919/467-3145 or 919/851-7223.
THE COMPUTER HOBBYIST welcomes contributions from our readers. Material to be submitted should be typed or neatly written and must not appear to be soliciting business for any firm. If you wish your material returned, please include a stamped, self-addressed envelope.

Brooks Radio and TV has been in the surplus business for at least 15 years. Their distinctive ads can usually be found in the hobby electronics magazines. They deal mostly in TV related components. Items of interest to TCH readers are high-voltage silicon rectifiers, flyback transformers, and color TV crystals as used in the audio cassette interface. The crystals are priced at 2 for $1.89.

Brooks Radio & TV Corp.
487 Columbus Ave.
New York, NY 10024

TCH received another flyer from Atlantic Surplus. They have all types of teletype gear but of special interest are Teletype Corp. model 15's for $80 checked out, and Lorenz model 15's for $85. Lorenz is a German firm that bought rights to the Teletype Corp. design and then added a few extra niceities such as a copy lamp and front panel carriage return and paper feed controls. The Lorenz units are also newer and smoother running. Write for a flyer to:

Atlantic Surplus Sales
3730 Nautilus Ave.
Brookly, NY 11224
Ph. 212/ 52-0349

D.M. Miller of Santa Ana, CA sent TCH this list of goodies which we chose to list here rather than as a want-ad. His prices are excellent.

PERIPHERAL EQUIPMENT	CONDITION	PRICE
Line Printers		
CDC 9340A	Good	$1000 - $3000
ODEC 1321	New & Used	$1000 - $2500
Digitronics 200	Used	$1250
Mag Tapes		
Ampex TMZ	New	$1500
Ampex TM7	Used	$500
Wangco Formatter (Phase encoded)	New	$1000
Pertec 37.5 IPS	Used	$750
Disk Drives		
(2311) Honeywell DSU 160	New	$1500 - $2500
(2310) IOMEC	Used	$750
Card Readers		
Mohawk 1000 CPM	Used	$500 - $800
Mohawk 300 CPM	Used	$200 - $400
Paper Tape Punches		
Digitronics	New	$250
Drums		
Vermont Research	New	$500

Contact: D.M. Miller
1191 Risa Place
Santa Ana, CA 92705
Ph. 714/838-0070 after 7PM PST

TCH still has 2102's and now has 2102-2's. The 2102 is a 1K static RAM which operates from a single supply and is fully TTL compatible. The 2102 has an access and cycle time of 1us and the 2102-2 has an access and cycle time of 650ns. The parts we offer are MM2102N's and MM2102-2N's which were obtained through normal distributor channels from National Semiconductor Corp. and are guaranteed by TCH for 90 days which is the same guarantee which National extends to us. Prices are $4.00 each or 8 for $31 for 2102's and $4.50 each or 8 for $35.00 for 2102-2's.

The Computer Hobbyist
Box 295
Cary, NC 27511

RANDOM NUMBER GENERATOR By: Jim Parker

When it comes to pseudo-random number routines, most of the ones I've seen use the idea of multiplying a number by some ugly constant, and then adding the remainder from a subsequent division. This works fine on a big machine with multiply and divide hardware, but the chore seems overbearing for the 8008. By far, the best idea I've heard yet for a random number generator is one suggested by Hal Chamberlin. His method simulates a hardware shift register with the input being fed from an EXCLUSIVE OR of two of the bits in the register. Don Lancaster discusses this hardware method of generating random numbers in a recent issue of Radio Electronics magazine and in his "TTL Cookbook". Basically this routine simulates a 31 bit shift register with the input being fed from an EXCLUSIVE OR of bit positions 28 and 31. This shift register can be clocked (2**31)-1 (read as 2 to the thirty-first power minus one) or 2,147,483,647 times before it repeats. The routine shifts the register 8 times to get one random byte but since 8 and (2**31)-1 have no common factors, we can accept (2**31)-1 bytes before they repeat. The routine executes in about 1300 machine states so if you ran it at top speed on an 8008-1, you would generate 300 random bytes per second for over 11 weeks before repeating.

The initial value of the four bytes in SHIFT determine where, in the over 2 billion byte sequence, the random numbers begin. By initializing SHIFT with the same value each time, we would get the same sequence of random bytes. There are a number of ways to initialize SHIFT. This routine simply depends on whatever "garbage" was at that location before loading the program. There is less than a one in 2 billion chance that random memory would initialize SHIFT with the disallowed state of all zeroes. One last note; the routine can be speeded up 8 times by removing the LBI 8, DCB, and JFZ RTOP instructions but the effective randomness in the short run (8 successive bytes) will be reduced. However, this change may be useful for some applications such as RAM testing or audio noise generation.

```
*                               THIS SUBROUTINE RETURNS
*                               A RANDOM BYTE IN REG A
*                               AFFECTS REGS A,B,H,L

RAND     SHL    SHIFT+3         PT TO SHIFT BYTE 4
         LBI    8               SET FOR 8 SHIFTS
         LAM                    LOAD SHIFT BYTE 4
RTOP     RLC
         RLC                    MOVE BIT 28
         RLC                    TO POSITION 31
         XRM                    X OR BITS 28 & 31
         RAL
         RAL                    MOVE NEW BIT INTO CARRY
         DCL
         DCL
         DCL                    PT TO SHIFT BYTE 1
         LAM                    LOAD SHIFT BYTE 1
         RAL                    ROTATE THRU CARRY
         LMA                    SAVE
         INL
         LAM                    LOAD SHIFT BYTE 2
         RAL                    ROTATE THRU CARRY
         LMA                    SAVE
         INL
         LAM                    LOAD SHIFT BYTE 3
         RAL                    ROTATE THRU CARRY
         LMA                    SAVE
         INL
         LAM                    LOAD SHIFT BYTE 4
         RAL                    ROTATE THRU CARRY
         LMA                    SAVE
         DCB
         JFZ    RTOP            REPEAT 8 TIMES
         RET
SHIFT    DST    4               SIMULATED SHIFT REGISTER
*                               STORAGE, MUST NOT CROSS
*                               A PAGE BOUNDARY
```

TCH STANDARD AUDIO CASSETTE INTERFACE

By: Hal Chamberlin

In this article a universal interface that implements the audio cassette data recording standard will be described. A printed circuit board containing the interface circuitry is being developed and will be announced in the next issue. The cassette software being developed is designed to use this interface circuit also. The interface is universal and may be used with virtually any computer. In most cases, the only additional logic required to connect to the computer I/O bus is a couple of 7430's to decode the device address. Connection to a port-oriented I/O system can be accomplished with no additional logic at all. The interface also has daisy-chaining capability where a number of units may be connected in parallel and selected under computer control.

The first problem encountered in designing the interface was the question of the hardware/software split. It is possible to minimize the hardware (down to one or two IC's for TCH's standard) at the expense of complex software and machine dependence. One may also strive for a completely automatic interface that presents the data on a silver platter directly into memory. The final choice depends on many factors including personal preference.

One of the factors considered in our design was machine independence. Many audio cassette systems depend on timed loops in the software for write timing and read decoding. Supporting software for such systems is of course critally dependent on the instruction timing of the CPU used. Many CPU's do not have definite instruction timing because of asynchronous logic or occasional pauses for dynamc memory refreshing. In particular, interrupt activity during tape operations would have to be avoided. Although these limitations apply mostly to larger systems, one CPU currently being used by hobbyists has an instruction time dependent on the type of memory used. In a mixed memory system (very likely) the timed loops would have to be matched according to their location in memory. In order to gain any degree of machine independence, the timing functions will have to be done by the interface and communicated to the CPU via status bits. Another feature added to simplify connection to any computer is jumper selection of input data polarity and an output driver socket that accepts a variety of open collector, tri-state, inverting and non-inverting gates.

Another factor considered in the design is ease of adjustment once the interface is built. Due to the pulse modulation data recording technique used, the usual tuning adjustments are not needed at all. The interface timing signals could be generated with one-shots quite efficiently with respect to IC count but would have to be adjusted after assembly and possibly readjusted during the unit's life. Since these times are in the micro and millisecond range, a triggered sweep scope would be needed for the adjustments. TCH decided to implement all of the critical timing by means of counting down a crystal oscillator. This raised the package count from 8 to 16 however the increase is well justified in terms of ease of construction and alignment. The net result of this design philosophy is a cassette interface that should work satisfactorily without any adjustments as soon as construction is completed. Optimum performance of the analog section can be assured with a single adjustment that requires only the use of a DC voltmeter.

Additional design considerations included the use of standard readily available components only and the desirability of single 5 volt supply operation. The final circuit may not have the least parts count or even the lowest cost but does possess most of the qualities needed for the implementation of a standard.

Before delving into circuit operation, the logical operation of the interface will be described. The discussion will be based on the 8008/8080 but applies equally well to any machine. The interface requires one input port or address and one output. The four least significant bits of the output control the interface operation. The remaining bits may be utilized for selecting units in a multitape system. The four control bits are MOTOR, MODE, WRITE ENABLE, and DATA. All except WRITE ENABLE are latched in the interface each time the output is addressed. When the MOTOR bit is a one, a relay will be closed applying power to the recorder. The MODE bit conditions the internal circuitry for reading when it is a ZERO and for writing when it is a ONE. The user may also connect an LED to indicate when to switch the recorder to record mode or make direct connection to the recorder for automatic record/playback switching. The WRITE ENABLE and DATA bits are functional only in the write mode. Whenever a word is sent to the interface with WRITE ENABLE and MODE on, a sequence is started for writing the pulse pattern required to encode the state of the DATA bit. Data should not be changed until the write bit operation is complete. Since WRITE ENABLE is not latched, the program need not reset it between each bit. Once the motor is running and the recorder is in the record mode, a bit string may be written by simply outputting a sequence of words, one for each bit to be written. The words would appear, in binary, as XXXX111D, where X indicates user option, and D is the data bit to be written.

The four least significant bits of the input indicate status of the cassette recorder interface. The remaining bits may be either ZEROES or ONES depending on the particular machine used. The input bits are MOTOR BUSY, WRITE BUSY, CLOCK, and DATA. Whenever the status of the motor control bit is changed, a timeout is started which causes MOTOR BUSY to be a ONE for the duration of the timeout. The delay for turn on is 1.5 seconds and for turn off is .5 seconds thus complying with the record gap length standard. A "write record" routine should first turn the motor on and then wait for MOTOR busy to return to ZERO before bits are written. After the record is written, the motor should be turned off and another wait on status performed before the routine returns unless an adequate delay can be assured before the routine is entered for the next record. Another possibility is to have the write routine check for MOTOR BUSY being off before the motor is turned on and not wait after writing. This is both efficient and safe and is probably the best way to handle things.

The WRITE BUSY status bit comes on during the write bit sequence. The program should wait until this bit returns to ZERO before the next bit is written. Also, no outputs to the interface that could change the DATA bit should be done while WRITE BUSY is a ONE. The time lag between recognizing the end of busy status and initiating writing of the next bit is added to the normal bit spacing on the tape and therefore will decrease effective data rate if the lag is excessive. An efficient program would have the next data bit all ready to go while WRITE BUSY is being waited upon.

CLOCK and DATA are used for reading the bits from the tape. The interface is "listening" whenever the MODE control bit is a ZERO. The CLOCK bit is normally a ZERO. When a bit or other noise of sufficient amplitude is "heard", CLOCK goes to a ONE for awhile. When it returns to ZERO, the data bit read will appear in DATA. It remains there, unchanged, until CLOCK makes another ONE-to-ZERO transition. A read routine should read the DATA bit separately after the ONE-to-ZERO transition is detected to eliminate possible errors due to differential propagation delays. For tapes played at normal speed, the CLOCK bit is ONE for about 50% of the bit time and so should not be missed even if parts of the read routine are lengthy.

Connections to the interface consist of 12 wires to the computer and 6 wires to the cassette recorder. The connections to the computer are listed below:

1. +5 VOLTS POWER
2. GROUND
3. OUTPUT STROBE
4. MOTOR CONTROL
5. MODE
6. WRITE ENABLE
7. WRITE DATA
8. INPUT ENABLE
9. MOTOR BUSY
10. WRITE BUSY
11. CLOCK
12. READ DATA

The OUTPUT STROBE and INPUT ENABLE signals provide easy interfacing to bus structured machines and daisy-chaining capability. These signals are active-low since in all probability they would be driven by NAND gates. They are provided with pullup resistors to keep the interface idle if connection to the CPU is broken. Data is accepted into the interface input latches when OUTPUT STROBE is active (logic ZERO). The data from the computer should not change when OUTPUT STROBE is active or within 30 ns of it being active. The minimum pulsewidth of the strobe is 350 ns and the maximum width is just short of 2 ms. Too short a pulse may result in failure to write even though data will be latched. Too long a pulse will cause multiple writing of the same data bit. Nearly all CPU's generate strobes in the range of 400 ns to 4 us so this restriction should cause no problems. The status output drivers of the interface are activated when INPUT ENABLE is active. There is no restriction on the timing of INPUT ENABLE except that imposed by the CPU's I/O system. It may be exercised as frequently and rapidly as desired with no effect on interface operation.

The connections to the recorder are AUDIO IN, AUDIO OUT, and MOTOR RELAY. The majority of inexpensive recorders either have three miniture phone jacks or a DIN plug. In the case of three jack recorders the labels are usually EARPHONE, MIC, and SW. Often the MIC label is between two of the jacks. One of the two jacks, usually the smaller one, is the switch jack and the other is the microphone signal jack. If you have the owners manual consult it to be sure of the jack designations. AUDIO IN on the interface should be connected to the EARPHONE jack; AUDIO OUT to the MIC jack, or if your recorder has one, the AUX or LINE IN jack; and MOTOR RELAY to the switch jack. The

100 ohm resistor across the AUDIO OUT terminals on the interface should be omitted if a high level input such as AUX is used. Slightly better results can be expected if a high level input is used since one stage of amplification would be bypassed. Shielded cables should be used for the signal connections. It may be desirable to disable the speaker cutout contacts on the earphone jack so that the audio can be heard at all times. This allows voice messages to be heard as well as audible feedback of proper recorder operation. Pulse modulation is less grating on the ear than tone modulation because of its wider spectrum.

Figure 1 shows the complete schematic of the cassette recorder interface. If some of the logic symbols are confusing, consult the December 1974 TCH for an explanation. The convention we will be following for labelling MSI circuits will be fully described in a future issue. In the meantime, correlate the pin numbers on our diagram and the labelling on a data sheet to resolve any confuson. This is the same circuit that will be offered in PC form next issue.

The digital portion of the circuit is fully synchronous throughout thereby avoiding problems with glitches and races in the logic. The clock source is a crystal oscillator using a common color TV subcarrier crystal of frequency 3.579545 mHz. Probably more of these have been made than any other single frequency. They should be available at any TV repair shop. See also Surplus Summary in this issue for the address of Brooks Radio & TV, another excellent source. Fortunately this frequency, when divided by 7168=2X14X16X16, yields 499.378 Hz which is within .12% of the desired bit rate of 500 BPS.

The crystal oscillator consists of four interconnected inverters and provides CLOCK and $\overline{\text{CLOCK}}$ at 3.58 mHz to the rest of the interface at full TTL fanout capability. A counter chain consisting of a 7474 D-flop and three 74161 synchronous binary counters is driven by CLOCK. All of the counter outputs change simultaneously following the ZERO to ONE transition of CLOCK. The reset input to the 7474 is used to enable counting of the entire chain. This signal is applied only when the flop is already reset and removed only immediately following a ZERO to ONE transition of CLOCK thus eliminating any possibility of a glitch on the output. The 74161 following the 7474 is connected for division by 14. This is accoplished by using the load and data inputs to set the counter to 0010 on the next count after 1111. Since counts 0000 and 0001 are skipped, division is by 14. The next 74161 simply performs a divide by 16 synchronously with the preceeding counters. The last divide by 16 stage is used to break the bit period of 2 ms into 16 sub-periods of 125 us each for the various timing functions. An excellent description of the 74161 counter can be found both in Fairchild and Texas Instruments IC catalogs.

Two halves of a 74107 J-K flip-flop are used to control the operation of the interface. Under normal idle conditions both flops are off and the counter chain is disabled. Except at power on, all counters in the chain are in their base state while idling. The $\overline{\text{CLOCK}}$ signal from the 3.58 mHz oscillator serves to sample the J and K inputs to the flops on its 1-to-0 transitions which are equivalent to 0-to-1 transitions of CLOCK. The counter chain is enabled if either flop is on. The write flop is conditioned to set when MODE, WRITE ENABLE, and OUTPUT STROBE are all ONE. When set, the counter chain is enabled and goes through a complete cycle of sixteen 125 us. periods. The write flop is conditioned to reset at the end of the counter cycle when all of the counter bits are ONES. The next clock pulse then overflows the counters to their base state and resets the write flip-flop which holds them there. WRITE BUSY status is connected to the write flop. If MODE is ZERO and a pulse is received from the analog circuitry then the read flip-flop is conditioned to set. However only 14 of the 125 us periods are used for reading a bit to allow for upward speed variation. The read flop is conditioned to reset when the third 74161 is just about to go to 1110 from 1101. When it does reset on the next clock pulse, the lower order counters overflow to their base state and the last counter does indeed go to 1110. However on the following clock pulse all ZEROES are loaded into the last counter returning it to the base state also.

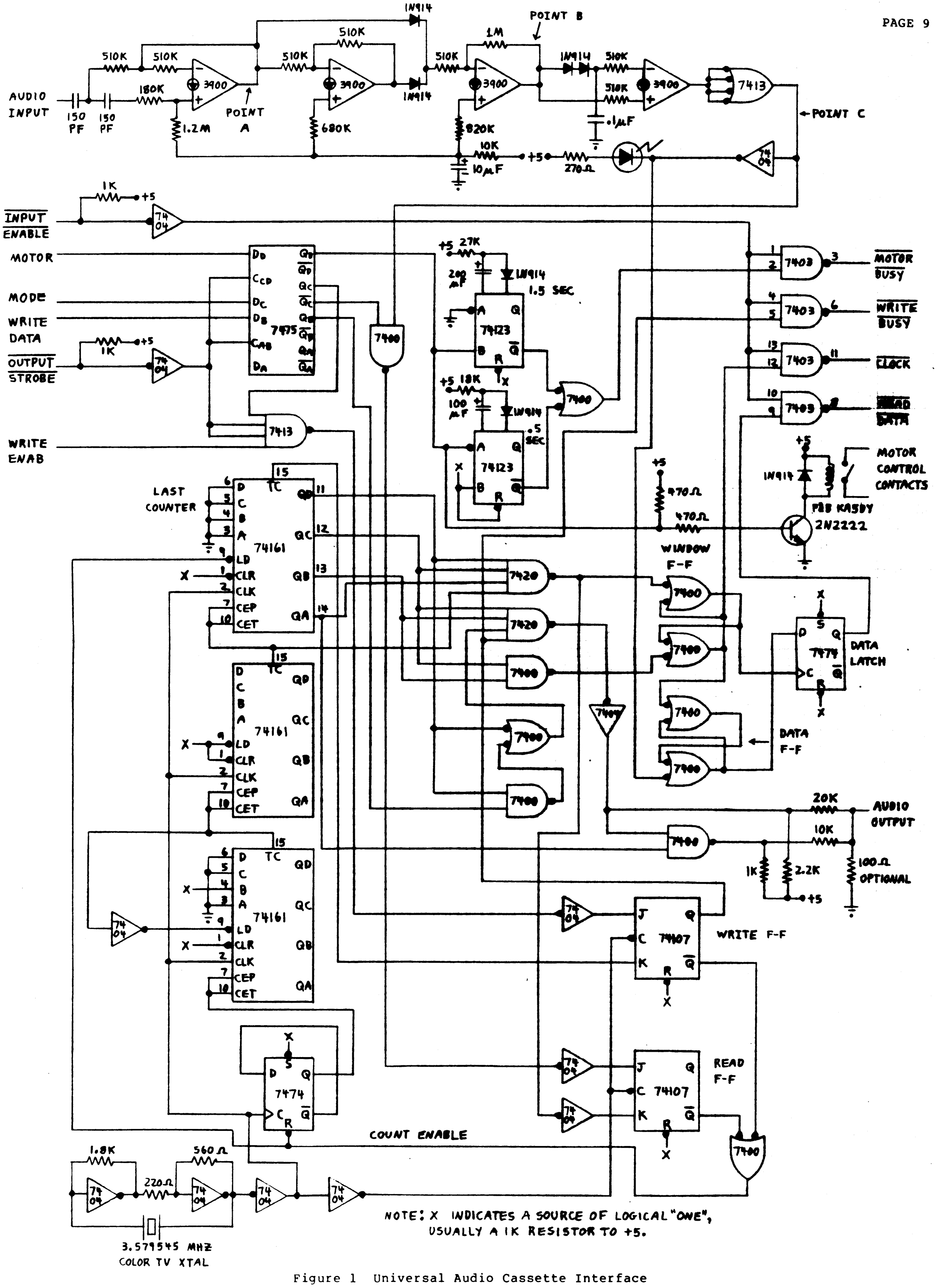
The audio signal generation circuitry is enabled when the write flip-flop is on and the counters are going. The first doublet pulse of the bit written is generated on counts 0110 and 0111 unconditionally. If a data ONE had been latched into the interface when the write command was issued, another doublet pulse is generated on counts 1110 and 1111 exactly a half bit period later. The resistor network used to generate the doublet waveform is actually a very simple digital-to-analog converter. The two pullup resistors to +5 at the gate outputs serve to standardize the output voltage swing independent of gate characteristics. When idle, the inverter output is 0 volts and the NAND gate output is +5 volts giving a voltage of 3.33 at the audio out terminal. During the first half of the pulse such as at a count of 1110, the inverter output goes to +5 volts and the gate remains at +5 giving an output of +5 volts. The NAND gate output drops to zero during the second half of the pulse and due to the fact that its weighting resistor is 1/2 the value of the inverter's, pulls the audio output down to 1.66 volts. The audio output voltage swing is reduced by a factor of 66 for use with a microphone level input if the 100 ohm resistor to ground is included but the waveshape is unaffected.

The motor relay circuit consists of a high current transistor driver and two single-shots. When the MOTOR latch contains a ONE, the 2N2222 transistor is turned on and drives the relay coil. The relay specified requires about 170 mA at 5 volts and needs a low saturation voltage driver to insure closing. The diode across the coil suppresses inductive kickback which would otherwise destroy the transistor when it was turned off. The 470 ohm resistor to +5 from the latch output insures adequate source current from the TTL to drive the transistor base. A 1.5 second single-shot is triggered when the MOTOR latch makes a 0-to-1 transition and a .5 second single-shot is triggered on the opposite transition. The two single-shots outputs are ORed together and become the MOTOR BUSY status signal. The timing components shown were computed using the manufacturer's data but in all probability will give too long a delay due to capacitor tolerance. While this will not affect compatibility with the record gap standard, it can reduce the throughput of the system. A resistor in parallel with the designated timing resistor can be used to bring the time down if maximum performance is desired. TCH's board will have an extra pair of pads for this purpose. Low voltage electrolytics such as 6 volts will give better long-term stability than higher voltage units in IC single-shot circuits.

The playback signal processing circuitry uses four amplifiers from an LM3900. This IC functions very well on a single 5 volt power supply. This first stage is an active highpass filter with a corner frequency of approximately 2kHz. This was necessary since low frequency noise on the tape can be a significant fraction of the pulse amplitude. The theoretical voltage gain is 2.5 but in practice the output signal is about the same amplitude as the input. A full wave rectifier using two diodes is fed with an in-phase signal from the filter stage and an out-of-phase signal from the unity gain inverter second stage. Full wave rectification of the signal is desirable to make the circuit insensitive to exact waveshapes and signal polarity. Under no-signal conditions the DC voltages at the diode anodes should be equal. If they are not equal to within 100 mV or so, the 1.2 meg bias resistor in the filter stage may be adjusted until they are equal. Note that some current is always flowing through one of the rectifier diodes and that their forward drop is of no significance. The third stage isolates the full wave rectifier from the detector and also provides a gain of two. The detector compares the rectified pulses with the short-term average background noise in the system. The .luF capacitor charges to two diode drops or about one volt below the rectified noise level. When a pulse comes through, the diodes reverse bias holding the capacitor charge. If the pulse amplitude is more than one volt negative, the comparator flips. The two diodes thus serve as a switch and as a threshold element. The relatively slow switching of the comparator stage is sped up to a TTL compatible value by a 7413 Schmidt trigger as well as buffered for greater fanout. The detected "pulse" from this processing chain may actually be a burst of closely spaced pulses and the following digital circuitry will have to allow for it. Note that except for the high pass filter the entire chain is DC coupled. This makes the unit relatively immune to signal overloads. An LED is connected to the detected pulse output to aid the user in setting the playback level.

If a pulse is detected while MODE is a ZERO, the read flip-flop turns on and starts the counter. Two pairs of cross-coupled NAND gates connected as R-S flip-flops to the last divide by 16 counter are used to distinguish between a ONE and a ZERO. When the counter reaches 0110 after .75 millisecond, the first gate pair is flipped starting the decoding window interval. This removes the reset from the second gate pair allowing it to catch and hold any pulses seen during the window interval. When the counter reaches 1101 and is just about to flip to 1110, the decode window is turned off. While turning off, a 7474 D-flop catches and holds the final state of the data detector flip-flop which is reset at the same time. Thus if no pulses were seen during the window interval, the data detector would have remained off. The 7474 holds the data until the end of the next bit allowing maximum time for the CPU to read it. The window flop is buffered and becomes the CLOCK status bit.

The status drivers are shown as open collector NAND gates. If the sockets are wired as indicated, inverting or non-inverting tri-sate gates may also be plugged in if desired for compatibility with nearly any kind of bus structure. The data inputs are shown for true data.



NOTE: X INDICATES A SOURCE OF LOGICAL "ONE", USUALLY A 1K RESISTOR TO +5.

Figure 1 Universal Audio Cassette Interface

WRITE ENABLE may be inverted with one of the spare gates and the other inputs inverted by switching connections to Q and $\overline{Q}$ on the 7475 latches.

Construction of the cassette recorder interface should not be difficult especially if TCH's board is used. One recommendation that always holds true is to use prime quality first-run IC's unless you have a good oscilloscope or a fairly sophisticated IC test setup. This is particularly true with respect to the LM3900 linear circuit which could be of very poor quality and still be judged "functional". Expect to pay 75 cents to a dollar for this part. Five percent resistors are recommended in all places except those used as pullups which may be 20%. Wiring layout is fairly non-critical but remember to adequately bypass the power with a .luF disk for every 2 or 3 IC's. The 5 volt power supply should be within 5% and free of ripple since the analog circuitry is powered from the same supply.

Setting up the recorder should not be difficult either. If your recorder has an automatic gain control for recording, you will not have to worry about recording level. If a level meter is supplied instead, adjust the recording level as you would for normal sound recording with the computer outputting a continuous string of ONES. On playback, adjust the level up slowly from zero until the signal monitor LED in the interface lights with a data tape. Optimum level is about 6dB higher than this or roughly half again as loud. Once the operating range of the recorder's level controls is found, it should be marked for quick setup. If the recorder is used solely for data, it will probably be possible to place some tape over the controls and forget about them. If the recorder has a single tone control, it should be in the full treble position. If two tone controls are provided, they should both be set to zero or flat. In general, an expensive recorder works no better than a 30 or 40 dollar cheapie except in the area of mechanical ruggedness.

Tape quality is important as in all digital applications. Bits are packed nearly 300 to the inch thus even a small but gross imperfection may obliterate a bit. Cassettes intended for Hi-Fi application should be suitable in general. One way to cut cassette cost and still use expensive tape is to make use of cassette repair kits. These consist of an empty cassette put together with screws and leader tape already bonded to the hubs and cost around a dollar. Since many short cassettes will probably be needed for commonly used programs, the tape from an expensive cassette can be cut up and loaded into several of the empty cassettes.

Best results in duplicating tapes will be obtained if they are read into the computer and then written back out. This prevents buildup of background noise and cumulative speed error. It also assures the best possible signal quality for the recipient of the copy.

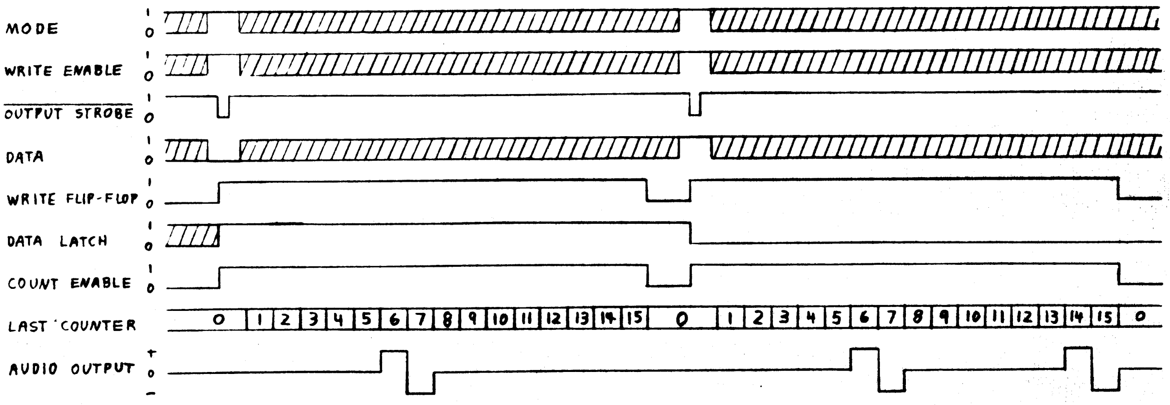


Figure 2 Interface Timing WRITE

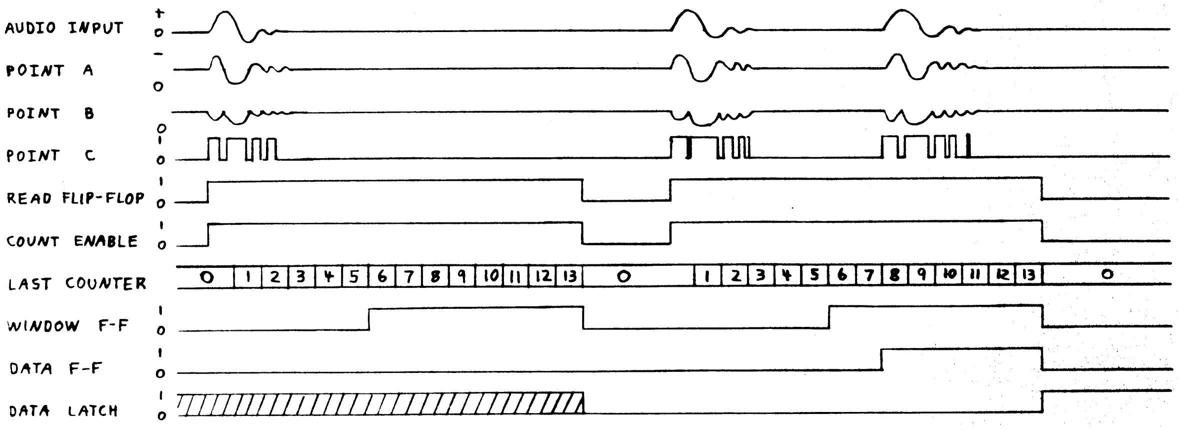


Figure 3 Interface Timing READ

DRAWING WITH YOUR POT CONTROLS By: Jim Parker

If you have already built the pot controls and digital switches as shown in the last issue of TCH, you may like to try using this program. It lets you draw figures on the graphics display by positioning lines and then storing them on the screen. You simply turn the 4 pots until you see a line where you want it and then push digital switch #1 to store it. You may repeat this process and store up to 255 lines on the screen. The lines are stored in memory in a sequential fashion so that if you do not have 5 pages of RAM you may still store as many lines as you have memory for. Here is a run down on the 4 switch functions which you may use:

Switch 1 - store the current line on the screen
Switch 2 - reset the screen
Switch 3 - remove the last line stored (decrement memory)
Switch 4 - restore last line erased (increment memory)

The program needs the RMIS, RMISO, and RVCD subroutines exactly as they are shown in issue 4 of TCH. The BAND routine is made into a subroutine with only one change, the JMP BAND instruction is replaced by a RET instruction.

```
XMOV    EQU   10B          GRAPHICS DISPLAY
YMOV    EQU   11B          I/O ADDRESS EQUATES
XSTOR   EQU   12B
YDRAW   EQU   13B
MINXY   EQU   14B
MINSZ   EQU   15B
GINP    EQU   6B

*                          MAIN PROGRAM
DRAW    SHL   GTABLE
        LMI   0            SET TABLE SIZE=0
        CAL   RMISO        INITIALIZE SWITCH ROUTINE
TOP     SHL   GTABLE
        CAL   GRAPH        DRAW GRAPH TABLE
        CAL   BAND         DRAW LINE FROM POT CONTROLS
        CAL   RMIS         READ SWITCHES
        RAL                TEST FOR SW1= ON
        JTC   SW1          JUMP IF SO
        RAL                TEST FOR SW2= ON
        JTC   DRAW         RESET IF SO
        RAL                TEST FOR SW3= ON
        JTC   SW3          JUMP IF SO
        RAL                TEST FOR SW4= ON
        JTC   SW4          JUMP IF SO
        JMP   TOP          LOOP

*                          ADD POT CONTROLS POSITION
*                          TO GTABLE
SW1     LAI   1
        CAL   RVCD         GET POT 1 VALUE
        SHL   GTABLE
        LBM                REG B * 4 + 1 = GTABLE
*                             BYTE SIZE
        INB                INCREMENT
        JTZ   TOP          RETURN IF OVERFLOW
        LMB                LOAD NEW GTABLE SIZE
        CAL   INHL         PT TO NEXT BYTE
SW11    DCB                COUNT THRU GTABLE
        JTZ   SW12         UNTIL YOU PT TO FIRST
        CAL   INHL         NEW BYTE IN GTABLE
        CAL   INHL
        CAL   INHL
        CAL   INHL
        JMP   SW11
SW12    LMA                LOAD POT 1 VALUE IN TABLE
        CAL   INHL
        LMD                LOAD POT 2 VALUE IN TABLE
        CAL   INHL
        LME                LOAD POT 3 VALUE IN TABLE
        CAL   INHL
        LMC                LOAD POT 4 VALUE IN TABLE
        JMP   TOP

SW3     SHL   GTABLE       DECREMENT GTABLE SIZE
        LAM
        ORA
        JTZ   TOP          RET IF UNDERFLOW
        SUI   1
        LMA
        JMP   TOP

SW4     SHL   GTABLE       INCREMENT GTABLE SIZE
        LBM
        INB
        JTZ   TOP          RET IF OVERFLOW
        LMB
        JMP   TOP

*                          SUBROUTINE TO TAKE A
*                          TABLE OF (X1,Y1) & (X2,Y2)
*                          POINTS AND CONNECT THEM
*                          WITH LINES PAIRWISE
GRAPH   LBM                GET TABLE COUNT
        INB                CANCEL NEXT STEP
LOOPG   DCB                DECREMENT TABLE COUNT
        RTZ                RET IF FINISHED
        CAL   INHL
        LAM                GET X1
        OUT   XMOV
        CAL   INHL
        LAM                GET Y1
        OUT   YMOV
        CAL   INHL
        LAM                GET X2
        OUT   XSTOR
        CAL   INHL
        LAM                GET Y2
        OUT   YDRAW        CONNECT WITH A LINE
        JMP   LOOPG

INHL    INL                BUMP L
        RFZ                RET IF NO OVERFLOW
        INH                BUMP H
        RET

*       PLACE BAND, RMISO, AND RMIS SUBROUTINES HERE
*       THEY MAY BE FOUND IN TCH VOL. 1  NO. 4

GTABLE  DST   1021         USES 4 BYTES FOR EACH LINE
        END   DRAW
```

CLASSIFIED ADS

There is no charge for classified ads in TCH but they must pertain to the general area of computers or electronics, and must be submitted by a non-commercial subscriber. Feel free to use classified ads to buy, sell, trade, seek information, announce meetings, or for any other worthwhile purpose. Please submit ads on seperate sheets of paper and include name and address and/or phone number. Please keep length down to 10 lines or less.

SERVICE AVAILABLE: I am setting up a 5203 PROM programming service if anyone is interested. Contact: David Yulke, 121 Liberty Ave.,Seldon, NY 11784, Ph. 516/698-0551

FOR SALE: Have two each 8263 and 8267 for Mark-8. All four chips for $18. If desired will sell these IC's plus all MK-8 circuit boards except output board for $25. Front panel has tested IC's installed. T.F. Caldwell, Box 116, Burgess, VA 22432

BENEFIT from Amateur Computer quantity purchasing power. We will be making a quantity buy of 2102 1K static RAM's 650 nsec. or faster, May 31st. We expect the price to be between $4.25 and $5.25 each. Write for more information, please enclose self-addressed stamped envelope. James Fry 4249 N. Lockwood, Toledo, OH 43612

WANTED: December 1972 Popular Electronics in good condition (Volume 2, Number 6). Dana Craig, 67 Hill Street, Norwood, MA 02062

THE COMPUTER HOBBY-IST
Box 295
Cary, NC 27511

ADDRESS CORRECTION REQUESTED

FIRST CLASS MAIL

THE COMPUTER HOBBYIST

Volume 1 Number 6 May 1975

Well folks, here it is; TCH for MAY. This issue contains hardware articles including the rest of the dope and the printed circuit layout for the TCH cassette interface. Our next issue will be devoted mostly to software and will have read/write and basic utility programs for the cassette interface.

For those of you who live within reasonable distance, TCH will be going to the Atlanta Hamfest, July 5 and 6. We will have a system with graphics, TTY, and cassette running.

Also with this issue some new policies need mentioning. Forign subscriptions are beginning to come in regularly now, so postal rates have been researched and subscription fees set. See the publication facts block inside for details. Advertizing is being accepted for the first time with this issue, if interested, please inquire. Lastly, TCH is seeking paid contributors. We need some more variety and just plain help. Please write or call for details.

TCH AUDIO CASSETTE DATA RECORDING STANDARD PART 2

by Hal Chamberlin

Before starting the logical format discussion, one update needs to be made on the physical format. Through use of the interface on a number of recorders, we found that the attack time of the automatic gain control on some recorders was longer than anticipated. This meant that most or occasionally all of the 16 initial ZEROES of a record would be distorted. The problem is worse if the recorder had been idle several minutes prior to writing. In order to minimize this difficultv, the standard will be ammended to require a minimum of 32 ZEROES before the ID rather than 16. Keep in mind that this is a minimum specification. More can be used if desired without problems but the overall throughput on short records will be reduced slightly. The complete ammended standard, both physical and logical, appears in the appendix.

Although it was not made clear in part 1, TCH's audio cassette standard is block oriented. This means that the data is divided up into small blocks on the tape. The recorder can be commanded to stop after a block to allow processing of the data and then restarted for the next block without loss of data. The block length is usually determined by the type of data being handled. Program source text or manuscript text would normally be organized with an intergral number of lines per block. This allows easy editing and processing of the textual material. TCH is keeping its subscription files on audio cassettes. Each block contains the name and address of a subscriber and information regarding the status of his subscription. Even binary programs lend themselves well to block organization. Most large programs have breaks in the sequential flow of instructions for data storage or to start a particular data table on a page boundary. Each contiguous section of code is broken into one or more blocks. Areas of memory skipped or reserved for data storage are not written on the tape. The proper load addresses for the blocks of code are part of the block itself so the user does not have to worry about getting the program loaded in the proper area of memory. Relocatable object code from relocatable assemblers is also easily accommodated with the relocation flags being part of the record. In short, block organization is necessary if audio cassettes are to be used as a general purpose mass storage computer peripheral.

The logical data format is concerned with the data bits between the end of the data ID and the beginning of the 16-bits-of-zeros postamble described in part 1. This block of useful data is further broken down into three fields. The first field is one byte (8 bits) and contains the count of the number of data bytes in the second field. Thus the second field may have a minimum length of zero or a maximum length of 255 bytes. The case of a zero length is used as a file mark block to indicate that the last block of data in the file has been read. The second field contains the "meat" data bytes. The format of these data bytes is defined by the program in use and is not part of the standard as such. The third field is two bytes in length and contains the Cyclic Redundancy Check (CRC). This is similar to a checksum and is used to verify the accuracy of data transfer. The CRC is a function of the first two fields. Given this standard data format, it is possible to write a general copy program that can duplicate a tape without having to know the details of the data organization on the tape.

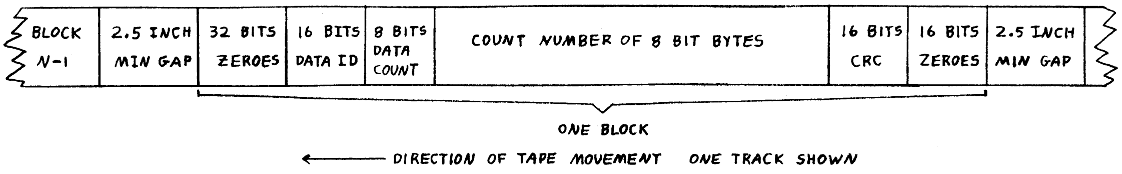
The most popular methods of data validity checking are vertical parity, longitudinal parity, checksum, and cyclic redundancy check. Vertical parity involves adding an extra bit to each byte such that the total number of ONES in the result is odd. Longitudinal parity requires an additional byte to be added at the end of a data block. Each bit in this additional byte is such that the total number of ONES in the corresponding bit position of the data bytes and the check byte is odd. Both parity methods will detect all single bit errors. Multi-bit errors may or may not be detected depending on the distribution of the errors. In particular, an even number of identical errors will cancel and therefore not be detected. Of the two, longitudinal parity would be preferred for block oriented data since the overhead is much smaller and its error detecting capability is about the same as vertical parity when data is serial by bit.

Checksum involves adding up all of the data bytes ignoring overflow and appending the sum onto the end of the data. When reading, the data bytes read are also added up and the sum is compared with the checksum byte at the end of the record. Checksum is somewhat better than longitudinal parity because the carries from bit-to-bit tend to propogate the effect of error bits to higher order positions of the check byte, thus reducing the possibility that another identical error in the same bit position will cancel the effect of the first. The checksum however is only guarenteed to detect single bit errors.

The cyclic redundancy check is by far the best of all of the simple data verification methods. The 16 bit CRC used by TCH detects all possible 1,2, and 3 bit errors in blocks shorter than 32,000 bits (our maximum block length is 2056 bits) and 99.99847% of all other possible errors. This same CRC technique is used commercially for data transmission and magnetic data recording.

Unlike the other verification methods, the cyclic redundancy check is computed on a bit-by-bit basis. A 16 bit CRC 'register' is defined by the read/write cassette tape program. This register is cleared to zeroes before the data bits are transferred. Each time a bit is transferred it is "combined" with the current contents of the CRC register. The proceedure for combining a data bit with the CRC is as follows:

1. The data bit is exclusive-ored with the leftmost bit of the CRC register.
2. The entire CRC register is shifted left one position, a ZERO bit replaces the vacated rightmost bit.
3. If a ZERO was shifted out in step 2, the computation is finished.
4. If a ONE was shifted out in step 2, a 16 bit constant (called the CRC polynomial) is exclusive-ored with the CRC register after which the computation is finished.



This algorithm is very effective because not only are errors shifted to other bit positions but their effect is multiplied in that one bit error will change several of the CRC register bits. Thus compensating errors become very obscure and therefore very unlikely.

Appendix 2 shows the 8008 code necessary to combine the leftmost bit of register A with the CRC register kept in D and E. With a total length of only 20 bytes, the routine is not really much longer than a parity or checksum routine.

The write record routine appends the final state of the two CRC register bytes onto the end of the data record. When reading back, the CRC register should be zero after the data and the CRC bytes have been read and combined with the CRC register. A non-zero result indicates an error. TCH's audio cassette support package will handle the computation and verification of the CRC.

While on the subject of the cassette support software, Appendix 3 lists the subroutines provided and a brief description of their function. There is a transfer vector containing jumps to the individual routines at the beginning of the ROM. This allows changes to be made to the cassette package without affecting the call addresses of the subroutines. The organization and choice of subroutine functions chosen offers maximum flexibility as well as upward compatibility with the Phi-Deck system TCH is developing. The ROM listing with object code, flowchart, and description will appear in the next issue.

This concludes the series on the audio cassette standard we propose. While it may seem more complicated than other schemes presently in use, it is because we have attempted to cover all feasible applications of audio cassette data recording rather than just memory dump and restore. Also, upward compatibility with high performance digital cassettes was deemed important. We welcome any constructive comments. Those having merit will be published and considered as ammendments to the standard.

APPENDIX 1

I. PHYSICAL STANDARD
  A. Modulation method - pulse
    1. ZERO is an isolated pulse
    2. ONE is a pair of pulses
    3. Timing
      a. bit spacing 2 ms. min., 50 ms. max.
      b. second pulse of a ONE follows first by 1.0 ms
    4. Pulse shape for recording - doublet
      a. high time .125 ms.
      b. low time .125 ms.
      c. total time .250 ms
      d. to be recorded without tape saturation
  B. Data record format
    1. Interrecord gap
      a. 2.5 inches minimum
      b. voice message may be present in gap
      c. produced with 1.5 second minimum delay between motor turnon and write data, and .5 second minimum delay between motor turnoff and next operation
    2. Preamble
      a. 32 minimum bits of ZEROES
      b. 16 bit data ID
        i. binary 1000100110101111
        ii. 2 byte octal 211 257
        iii. hexadecimal 89AF
    3. Data bits (see logical standard below)
    4. Postamble
      a. 16 minimum bits of ZEROES
  C. Tape tracks
    1. Half-track monaural is standard
    2. Outside tracks (channel A) if stereo must be used

II. LOGICAL STANDARD
  A. Bit packing
    1. Bits are packed 8 to the byte
    2. Leftmost bit of the byte is transferred first
    3. No additional overhead bits are used
  B. Data record format
    1. One byte count
      a. count of number of data bytes following
      b. minimum is 1
      c. maximum is 255
      d. a zero count is used as an end of file mark
    2. Count number of data bytes
      a. text record format (to be decided)
      b. 8008/8080 object code record format (detailed in next issue)
      c. other user defined record format
    3. Cyclic redundancy check, 2 bytes (16 bits)
      a. CRC-16 used, polynomial - 1000 0000 0000 0101
      b. leftmost bit written first
      c. computed on the count and the data bytes

APPENDIX 2

```
*       CRC CALCULATION ROUTINE
*       COMBINES LEFTMOST BIT OF REG. A WITH CRC
*       KEPT IN REG. D AND E

CTCC    NDI 200B    ISOLATE LEFTMOST BIT OF A
        XRD         EXCLUSIVE OR THE DATA BIT INTO THE
        LDA            HIGH ORDER BIT OF THE CRC
        LAE         SHIFT THE CRC LEFT BY 1
        ADA
        LEA
        LAD
        RAL
        LDA
        RFC         RETURN IF NO CARRY OUT
        LAD         EXCLUSIVE OR THE CRC POLYNOMIAL
        XRI 200B       INTO THE CRC
        LDA
        LAE
        XRI 005B
        LEA
        RET         RETURN
```

APPENDIX 3

CTROM	CTRS	Cassette Tape Read Start Turns motor on in unit selected by unit code in A, zeroes CRC register, searches incoming data for data ID and returns when found.
CTROM+3	CTRB	Cassette Tape Read Byte Reads 8 bits from the incoming data and assembles them in A. Combines each bit read with the CRC register.
CTROM+6	CTRE	Cassette Tape Read End Verifies CRC and sets condition code zero if no error and turns the motor off.
CTROM+9	CTWS	Cassette Tape Write Start Waits on motor status from interface. Turns motor on in unit selected by unit code in A, waits on motor status from interface, writes 32 bit preamble and 16 bit data ID, and zeroes CRC register.
CTROM+12	CTWB	Cassette Tape Write Byte Disassembles byte in A and writes the 8 data bits. Combines each bit with the CRC register.
CTROM+15	CTWE	Cassette Tape Write End Writes 16 bit CRC register contents onto tape, writes 16 trailing ZEROES, and turns the motor off.
CTROM+18	CTIPL	Cassette Tape Initial Program Load Reads a record into memory at a fixed location (user specified when ROM burned) and branches to it if no error. Halts if error.

EDITORIAL

Beginning with this issue TCH will be expanded to include a monthly editorial. Unlike our other articles and commentaries the editorial will be mostly opinion, sometimes that of a single staff member, and other times that of TCH as a whole. Reaction to the editorials, either supportive or dissenting, is welcome.

In reading other hobbyist publications, talking to people at the Dayton Hamvention, and scanning technical magazines, I am constantly made aware of the presence of MITS. Much of what is written and said about MITS is contradictory and controversial making it difficult to determine the truth independently. A look at the facts and some clear thinking is required in order to understand how this situation arose.

Most of us were introduced to MITS by an article in the January and February 1975 issues of Popular Electronics concerning their major product, the Altair 8800 computer. Initially the entire hobbyist community including TCH (see Vol. 1 No. 2) was elated at the prospect of a high performance computer at an almost unbelievable price. The passage of time aided by some unfortunate circumstances has, however, created a flurry of unflattering rumors. The basis of most of these rumors can be derived from shipping delays, unusual concern over MITS's marketing plan, and the "sour grapes" complex.

MITS has indeed experienced delivery problems. Orders for computers have taken as long as three months to be filled and shipments of extra memory and peripherals were delayed until the backlog of computer orders was cleared out. When looking at the situation from the individual hobbyist's point of view it is easy to see why there is a great deal of concern. Your money is gone and you have had nothing to show for it for two months. Visions of another "ripoff" cross your mind. "At this rate it will be obsolete before I get it."

The problem of course is that MITS has been swamped with orders in the thousands. I imagine that they even knew they would be swamped. However convincing investors and banks that a product will be successful beforehand is virtually impossible. Without ample financing, initial deliveries are going to be slow. More than one company has failed because they could not expand rapidly enough to meet the demand for their products. At any rate, 3 months delivery has been par to excellent in the minicomputer industry. Larger computers are announced a year or more before the first one is delivered. I have no doubts that deliveries will improve substantially with time.

Excepting IBM, I have never seen so much attention given to a company's marketing strategy as the hobbyists are giving to MITS's. Charges of "loss leader sales", "locking the users in", baiting, and using reject parts are common. Some people even worry about MITS's financial stability and how the Altair price can be so low. The most common theory is that they are selling the CPU at a loss in order to lock users into buying needed extra memory and peripherals at inflated prices.

The "loss leader sales" technique of introducing buyers to a product line is as old as free enterprise itself. It is the fairest of all promotional tools because the consumer can't loose. The intelligent buyer (I assume we would fall into that class) can take advantage of the leader item and pass up the other merchandise if its price-performance is unacceptable. The preceeding is irrelevant however because I believe MITS is making a reasonable profit on CPU's. It is amazing how the bottom drops out on component cost, especially semiconductors, when quantities reach 10,000 and more. Yes, I do believe they will ship that many systems by the end of the year!

Intelligent users and a free marketplace will not allow any computer company to truly lock its customers into using only its products. IBM has tried everything in the book to make independent add-ons difficult to design and expensive to implement yet the plug compatible manufacturers keep coming back with lower cost, higher performance peripheral equipment. I can find no evidence at all that MITS is trying to make interfacing difficult. On the contrary, their universal bus architecture makes add-ons easy for anybody. Already competitive plug compatible memory boards are available and you can bet there will be plug in replacements for all of their peripherals if indeed they can be sold profitably for less than MITS is asking. The universal bus also makes it easy for kit companies and publications like TCH to design complete, plug-in boards for peripherals not available form MITS. Nobody is going to be locked in unless they refuse to use the key!

Now, at the risk of stepping on somebody's toes, the last basis for rumor will be discussed. I think we all agree that the MARK-8 article and kit in Radio-Electronics (July 1974) introduced computers as a hobby to the masses of electronic hobbyists. However, the 8008 microprocessor chip had been around for over 2 years prior to that article and the MARK-8 design is not particularly original or unusually cost-effective. The real breakthrough was convincing the editors of a hobbyist electronics magazine that such an article would be well received and not over the heads of the readers. The problem is that many hundreds (perhaps even thousands, I don't have the exact figures) of MARK-8's were paid for and built before MITS came on the scene. ZOWIE! The MARK-8 had just been obsoleted by a machine that cost about the same (during the initial promotion anyway) and was far superior in almost every respect. A natural defense mechanism for those who felt "stuck" with the MARK-8 or other low-performance homebrew system is to criticize or downgrade the new system, the familiar "sour grapes" complex. This phenomonen is common in the computer industry and should not supprise anyone. A case in point is the reaction of IBM 360, 370-155, and 370-165 users to the introduction of virtual storage.

This situation is, of course, unfair to both MITS and newcomers to the ranks of computer hobbyists. If you feel your system is obsolete consider the following. Your decision to build a system was an economic one. That is, the level of performance and flexibility being offered was judged to be worth the cost. The introduction of a new system is not going to slow down your old system. If your system met your needs before, it will meet them just as well now unless the ability to do more has made you want to do more.

In summary, I believe MITS to be a fine upstanding company that has been unjustly criticized. If you still doubt their integrity, talk to an owner of their machine. There are thousands out there already.

LETTERS

We at TCH will publish a few of our more interesting letters each month along with comments by the staff.

Dear Sirs:

I am writing in response to the "Computer Bits" column in Popular Electronics. I am extremely interested in the graphics terminal by Hal Chamberlin, which was mentioned. Any additional information about this terminal and any other data of interest to a microprocessor "nut" would be greatly appreciated. I am reasonably impressed with the ALTAIR 8800, but if sufficient data is available, I may brew my own machine from scratch, having had a pretty good grounding in digital circuitry and construction techniques. In either case (ALTAIR or home-brew), I will need one or more terminals. The graphics capabilities of the aforementioned CRT display appeal more to me than the "TV Typewriter II", which had been my choice to date. I am interested in subscription data for any newsletters available. Thank you for your attention to this inquiry.

P.S. In all of the DP journals I see the term OEM used frequently, but cannot find a definition. Can you tell me what the initials stand for?

James H. Nestor, Ed.D.

James is one of many who saw our mention in P.E. Thank you P.E.! We have been swamped in mail. To answer James' question: OEM is an abbreviation for Original Equipment Manufacturer and refers to companies who construct equipment for use by others. OEM also usually implies quantity purchases, though not in all cases. OEM's however are always expected to be self-reliant as opposed to a retail customer who may expect and get lots of application help and service from the supplier.

THE COMPUTER HOBBYIST
Founded October, 1974

Stephen C. Stallings - Managing Editor
Hal Chamberlin - Contributing editor
Jim Parker - Contributor
Edwin Tripp - Photographer
Richard Smith - Programming consultant

THE COMPUTER HOBBYIST is published monthly near Raleigh, N.C. The yearly subscription rate is $6.00 for persons in the U.S.A., U.S. possessions, Canada, and Mexico. Subscriptions in all other countries are $8.50 per year for surface mail or $13.50 per year for air mail. All back issues are available and cost fifty cents each for persons in the U.S.A., U.S. possessions, Canada, or Mexico. For all other countries back issues cost 75 cents each by surface mail or $1.25 each by air mail. Remittances from countries other than U.S.A should be in the form of a bank or postal money order.

THE COMPUTER HOBBYIST is seeking paid contributors. Material to be submitted should be typed or neatly written and must not appear to be soliciting business for any firm. For details, please inquire.

Advertizing space is now available in TCH. Please inquire for rates and formats.

All correspondence should be addressed to:

THE COMPUTER HOBBYIST
Box 295
Cary, NC 27511

Members of the staff of TCH can usually be reached by phone evenings and weekends at either 919/467-3145 or 919/851-7223.

ADD A DATA STACK TO YOUR 8008 by Hal Chamberlin

Besides higher speed, the most significant improvement offered by the 8080 is the addition of a general purpose stack capability. Using the stack, the programmer may save registers used in subroutines and interrupt service routines and then later restore them. Arguments to subroutines may also be pushed onto the stack. In the 8080 the stack is kept in main memory and addressed by means of a stack pointer register. One inconvenience of the 8080 stack is that data may be pushed and popped only in byte pairs creating wasted space if only a single register needs to be saved. Also, the stack pointer MUST be set up at the beginning of a program before any subroutines are called and kept valid at all times or very strange things may happen.

Much of the programming convenience of the 8080 stack may be had on an 8008 system with the addition of about 6 IC's and the use of one input and one output address. The basic stack is 16 elements deep which is generally adequate for register saving applications. Addition of more chips and substitution of 256 X 4 RAM's for the 16 X 4 RAM's gives a 256 element capacity, ample for almost any use. In either case the added hardware provides both a stack pointer and a dedicated memory. The stack pointer does not need to be initialized and thus the stack is always ready for use, or it can be completely ignored by programs that don't need it without problems.

Programming the data stack is quite simple. The output address associated with the stack is given the symbolic name STPSH for STack PuSH and the input address is given the name STPOP for STack POP. When an OUT STPSH is executed by the program, all of the existing data (or garbage) in the stack is conceptually pushed down one location and the byte in register A is written into the top location which was vacated. When an INP STPOP is executed, the contents of the top location are read into register A and then all of the lower data in the stack moves up one location and the top location is lost.

An obvious application of the stack is in writing subroutines that do their job without destroying any registers. A simple example is the exchange HL and DE subroutine in Appendix A. First register A is pushed onto the stack. Then registers H and D, and L and E are exchanged using A. Finally the original state of register A is restored by popping it off the stack and the subroutine returns. Because of the push-down nature of the stack, one subroutine that uses the stack may call another subroutine that uses it and so on without loss of data as long as the stack's capacity is not exceeded. The only requirement is that all of the data saved on the stack by a subroutine be popped back off before it returns.

Appendix B shows a completely general interrupt service routine that uses the data stack to save all registers and the state of the conditions (C, Z, S, P). When entered, register A is first pushed onto the stack. Then the remaining 6 registers are saved one at a time by first loading then into A and then pushing A onto the stack. None of the instructions necessary to do this affect the condition flags. Finally a chain of conditional jumps is executed to create a "magic number" in A that reflects the state of the conditions. After A is pushed onto the stack, the interrupt may be serviced without restrictions on register usage.

In order to return to the interrupted program, first the magic number is popped off and added to itself with an ADA instruction. The number is such that the proper conditions are set when it is added to itself. Next the 6 index registers are popped off and restored in reverse order from which they were saved. Finally A is restored and a RET instruction is executed. This method of complete status saving may be modified for use by a debug program. Debug can be entered by a console interrupt and the user may examine things. Then program execution may be resumed with no loss of data. These two programming examples should serve to illustrate the use of the stack.

The stack is implemented with an up-down counter and a random access read-write memory. Rather than the data moving when pushes and pops are executed, the up-down counter acts as a pointer to the top element on the stack and the pointer moves. The logic is set up so that when an OUT STPSH is decoded, the counter first counts up one notch and after sufficient time for the address to settle in the RAM, a write pulse is generated to write the data from A into the RAM. The write pulse delay can be fairly short (50 NS or so) in the 16 element stack but must be at least 200 NS for the slower MOS RAM used in the 256 element version. It is possible that a timing problem may arise in a system using the 8008-1 if the output data is not valid for the sum of write pulse delay and write pulse width (950 NS) required by the MOS RAM. (Timing given is for the 2101 RAM. Matters are improved if 2101-1, 2101-2, or 9101 RAM is used.) There should be no problems with the bipolar RAM in the 16 element version.

When an INP STPOP is recognized, the contents of the currently addressed location are simply gated onto the input bus. The counter counts down one notch at the end of the INP instruction thereby addressing the next lower element on the stack.

Figures 1 and 2 show the logic diagram and timing chart respectively for a 16 element data stack. A bus type of I/O system (as opposed to a "port" type) is assumed. As shown, any system with either seperate data input and output busses or a bidirectional bus may be used. Some systems may use an output bus with TRUE data and an input bus requiring FALSE data. In this case, the 7401's may be omitted and the TTL RAM outputs tied directly to the input bus. The two single-shots, SS-1 and SS-2, are used to time the sequence of events for a stack push. First, NAND gate number 1 recognizes the coincidence of the STPSH device code on the address bus and an output strobe pulse or its equivalent. The gate output triggers SS-1 which increments the stack pointer counter when its cycle is finished. An RC network between the two single-shots delays firing of SS-2 until the counter has settled down and the RAM's recognize the new address. The write enable is connected to SS-2 which allows data on the CPU output bus to be written into the newly addressed RAM location.

The occurrence of an INP STPOP is detected by NAND gate number 2. As long as the gate is satisfied, data from the RAM is placed on the CPU input bus. At the end of the INP instruction when the NAND gate output goes back to a ONE, the counter decrements to address the next lower element on the stack. A 7400 connected as an OR-NOT enables the memory when either a push or a pop is being executed and disables it otherwise.

The logic necessary for a 256 element stack is essentially the same as for the 16 element version. The major differences are that a seperate single-shot should be used to time the write delay and that a buffer is absolutely necessary to drive the CPU input bus. If the polarity of the input bus is the same as that of the output bus or it is the same bus, 8093 or 74125 non-inverting quad tri-state buffers are convenient to use. Open-collector 7401 gates may be used instead if the input bus is inverted. The chip enables on the MOS RAMS should be grounded so that the chip is always enabled. The connection to the bus drivers is left as it was for the 16 element version however. The timings for the write delay and write pulse width single-shots can be set to the minimum values allowable for the standard 2101 RAM. If a 9101 is used (see surplus summary), the timing may be speeded up considerably. An 8101 may require somewhat slower timing. In any case be sure to check the data sheet for the RAM being used.

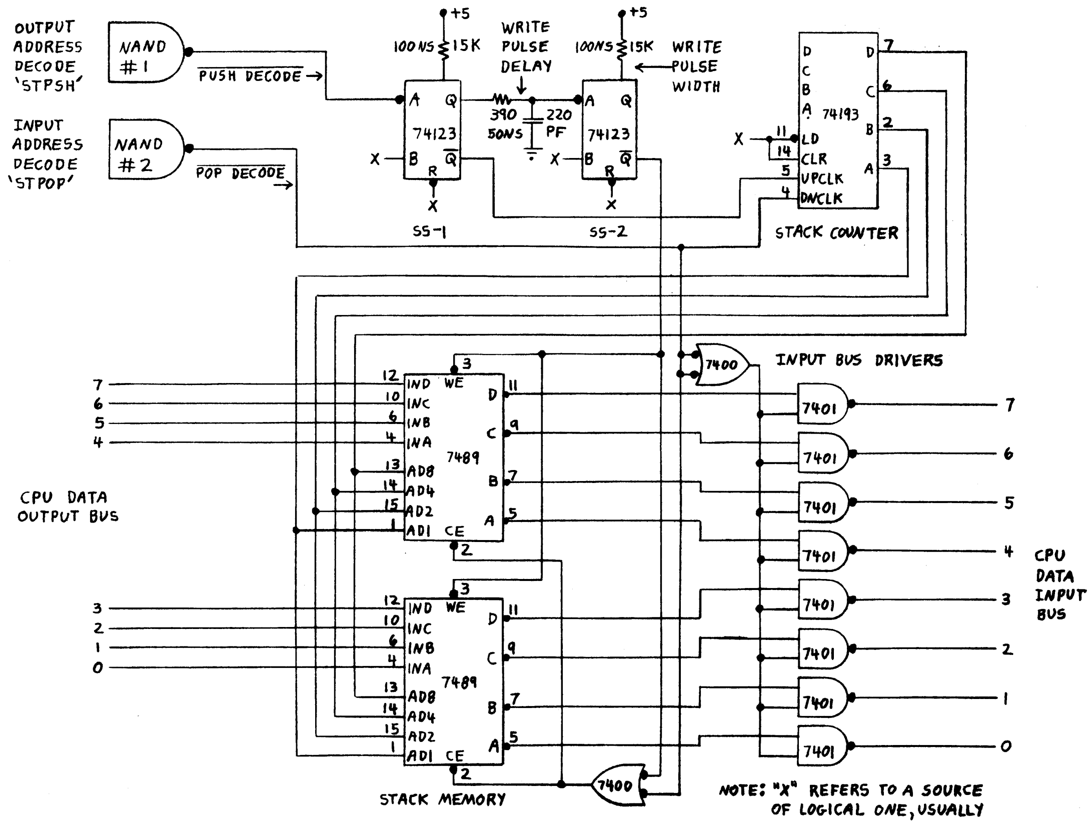
After writing a few programs using the stack you will wonder how you got along without it. The size and speed of some routines may be improved by a factor of two if use of the stack alleviates the need to constantly reference memory. An overall improvement of 10 to 20% can be expected on large programs such as assemblers. The biggest improvement however will be in coding time since register usage will not have to be carfully planned in advance.

APPENDIX A

```
*      SUBROUTINE TO EXCHANGE DE AND HL
*      USES ONE STACK LOCATION
*      USES NO REGISTERS

XCDEHL OUT STPSH    SAVE A ON THE STACK
       LAH          EXCHANGE H AND D
       LHD          USING A
       LDA
       LAL          EXCHANGE L AND E
       LLE
       LEA
       INP STPOP    RESTORE A FROM STACK
       RET          AND RETURN
```

APPENDIX B

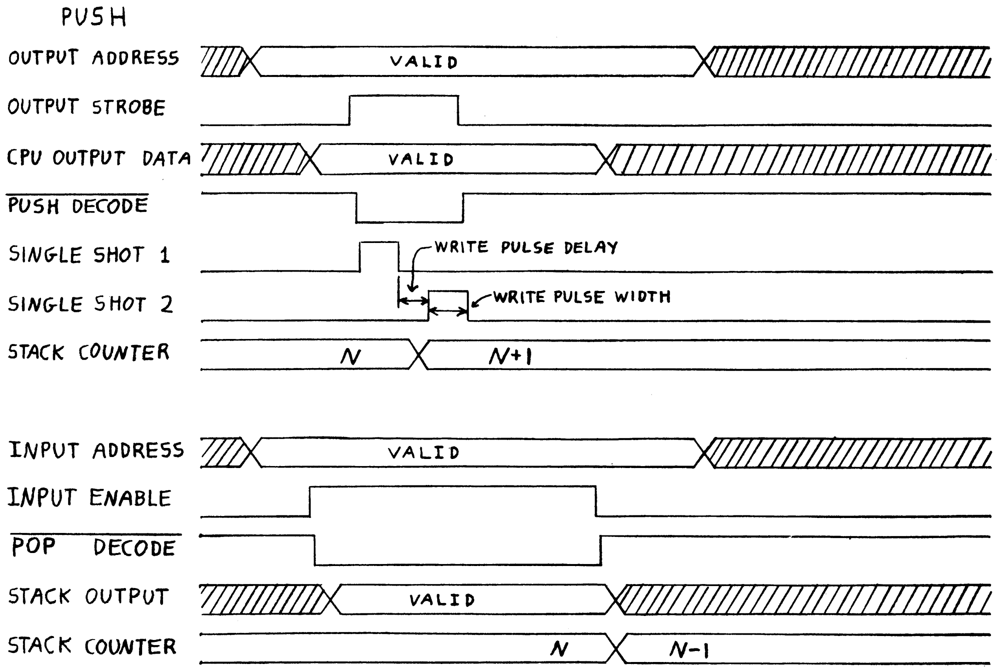
```
*      GENERAL PURPOSE REGISTER AND CONDITION
*      SAVE ROUTINE
*      USES 8 STACK LOCATIONS
```



NOTE: "X" REFERS TO A SOURCE OF LOGICAL ONE, USUALLY A 1K RESISTOR TO +5

FIGURE 1. 16 ELEMENT DATA STACK

FIGURE 2. STACK TIMING



```
GSAVE  OUT STPSH    SAVE A ON THE STACK
       LAB          SAVE B
       OUT STPSH
       LAC          SAVE C
       OUT STPSH
       LAD          SAVE D
       OUT STPSH
       LAE          SAVE E
       OUT STPSH
       LAH          SAVE H
       OUT STPSH
       LAL          SAVE L
       OUT STPSH
       LAI 0        CLEAR A
       RAR          PUT CARRY IN HIGH ORDER
       JTZ GSAV3    JUMP IF ZERO FLAG IS ON
       LBI 170B     PUT INTO B THE BIT MASK TO
       JTS GSAV1      TURN OFF THE ZERO FLAG AND
       LBI 030B       RESTORE THE SIGN FLAG
 GSAV1 JTP GSAV2    OR IN A 004B IF PARITY
       ORI 004B       INDICATOR IS OFF
```

```
GSAV2  ORB          COMBINE B AND A
GSAV3  OUT STPSH    SAVE MAGIC NUMBER ON STACK

  *       REGISTER AND CONDITION RESTORE ROUTINE

 GRSTR INP STPOP    RESTORE MAGIC NUMBER FROM STACK
       ADA          ADD IT TO ITSELF TO RESTORE CONDITIONS
       INP STPOP    RESTORE L
       LLA
       INP STPOP    RESTORE H
       LHA
       INP STPOP    RESTORE E
       LEA
       INP STPOP    RESTORE D
       LDA
       INP STPOP    RESTORE C
       LCA
       INP STPOP    RESTORE B
       LBA
       INP STPOP    RESTORE A
       RET          RETURN WITH STATUS RESTORED
```

The TCH Cassette Interface Printed Circuit Board

Thanks to the efforts of (would you believe) our programming consultant, Richard Smith, TCH now has a printed circuit version of the audio cassette interface. The board is laid out for double-sided with plated through holes, however care was taken to avoid lands which connect to IC's on the top side of the board. Thus problems that have occurred with the Mark-8 boards are avoided. Lands do connect to some component leads on the top side, however, these are easily soldered and wires can be put through the remaining via holes. TCH's prototype was constructed in this manner with no problems.

Several minor improvements have been made to the originally published circuit. Unit select logic and an additional relay was added so that two recorders could be controlled from the interface. The use of two recorders will greatly facilitate functions such as program text editing, data file maintenance, and program assembly. In order to ease connection with any computer, jumpers were added so that either true or inverted data could be accepted from the CPU output bus. Two circuit improvements were also made. Trials with several recorders showed that some units generate a lot of electrical noise when turned on and off (spikes exceeding 600 volts were observed). Therefore, an arc suppression circuit was added to the motor control contacts of the relays. The extra parts can be omitted if you have a quiet recorder. Also we discovered that some crystals might try to oscillate at 3 times their fundamental frequency. An additional R and C added to the oscillator will prevent this. For your convenience, the schematic is reprinted in this issue with all updates included.

Some more notes on the relays. Our unit was constructed with 6 volt relays running from the +5 volt logic supply. If this is done, either keep the relay and logic power wires seperate up to the power supply terminals or put at least 100 uF of bypass capacitor on the card between +5 and ground. If you do not have 6 volt relays, the seperate power connections will allow you to use up to 24 volts to power the relays. This voltage is limited by the 2N2222 driver transistors.

Construction of the board is straightforward, just follow the assembly diagram as shown. Resistors are mounted on end to conserve space and to allow either 1/4 or 1/2 watt units to be used. I/O connections to the CPU and power supply are made through a DIP type pattern on the board. You may either solder wires in or install a DIP socket and attach a DIP plug to your cable. I/O connections to the recorder are made through shielded cables for signal and a twisted pair for power. Signal connections are made directly to the relay contacts but ground pads are provided for the shields.

Three of the resistors have dual sets of pads so that a selected resistor can be put in parallel for optimum adjustment. These are R31, R34, and R46 which are the motor delays and the balance adjustment for the full wave rectifier respectively. If you wish to optimize your unit, use the next largest value of resistor and try parallel resistors until the delays and the rectifier balance are set as close as you desire. The unit should meet specifications however even if no adjustment is attempted. See vol 1, no. 5 for details on the adjustments.

Now how do you get your printed circuit board? There are three possibilities. You may do your own by copying our layout either with layout tape, ink on mylar, or a copy camera. The published patterns are exact 1 to 1 reproductions. You might get a friend to purchase and produce several boards from a set of contact prints of the layout which we can supply for $10 per set (specify either positives or negatives). Or you can purchase a fully plated through, glass-epoxy board from TCH for $15 which includes postage and a reprint of the assembly diagram, schematic, and parts list. The boards should be available by the time you read this. Also, while they last, we will include two 6 volt, three pole double throw relays for an additional $5.00. We have not worked out the details on the cassette ROM's yet so that offer will have to wait until the next issue.

PARTS LIST

R1 - R4	100 Ohms
R7, R8	100 Ohms, optional
R9, R10	220 Ohms
R11	270 Ohms
R12 - R16	470 Ohms
R17	560 Ohms
R18 - R25	1 K
R26	1.8 K
R27	4.7 K
R28 - R30	10 K
R31	18 K
R32, R33	20 K
R34	27 K
R35	180 K
R36 - R42	510 K
R43	680 K
R44	820 K
R45	1 MEG
R46	1.2 MEG
R47 - R49	Optional parallel trim
C1	27 pF
C2, C3	150 pF
C4 - C16	.1 uF disk
C17, C18	100 uF electrolytic 6 volt min
C19	220 uF electrolytic 6 volt min
IC1 - IC3	74161
IC4	7474
IC5	LM3900
IC6	7420
IC7	7413
IC8, IC11	7400
IC12, IC18	"
IC9, IC10	7404
IC13	74107
IC14	74123
IC15	7475
IC16, IC17	7403 or bus driver
J1 - J7	Jumpers, bold lines are for true input signals from CPU, fine lines for false
K1, K2	3 pole single throw relay, P & B KA14-DY or similar
D1 - D8	1N914 or other Si diode
L1	Light emitting diode
X1	3579.545 kHz crystal color TV burst freq
Q1, Q2	2N2219 or 2N2222

Note: Due to an oversight there is no R5 or R6

I/O PLUG CONNECTIONS

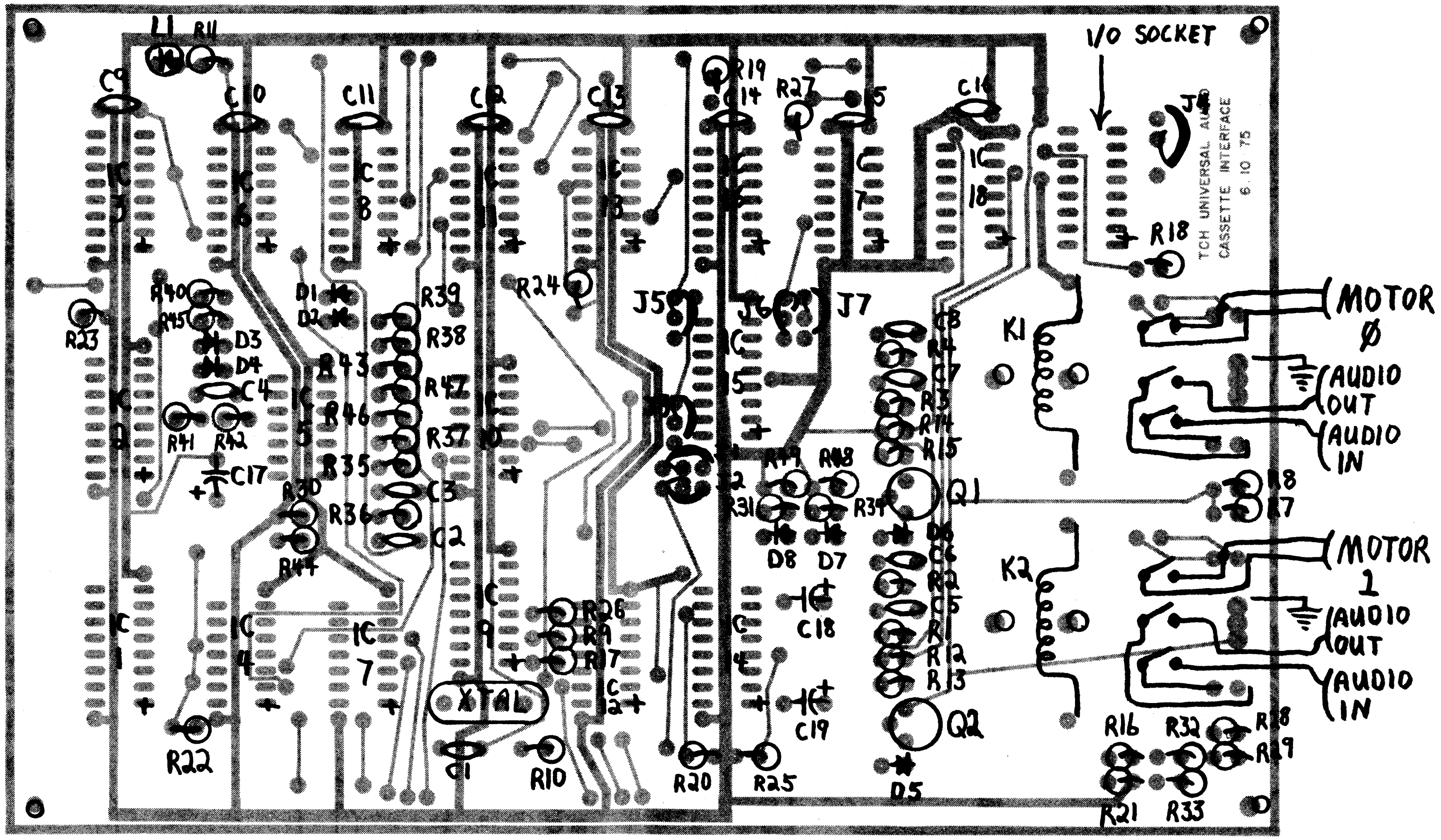
1. MOTOR BUSY
2. SELECT READBACK
3. READ CLOCK
4. WRITE BUSY
5. READ DATA
6. OUTPUT STROBE
7. WRITE ENABLE
8. INPUT ENABLE
9. GROUND
10. GROUND
11. LOGIC +5 POWER
12. RELAY POWER
13. UNIT SELECT
14. WRITE DATA
15. MOTOR
16. MODE

ASSEMBLY DRAWING

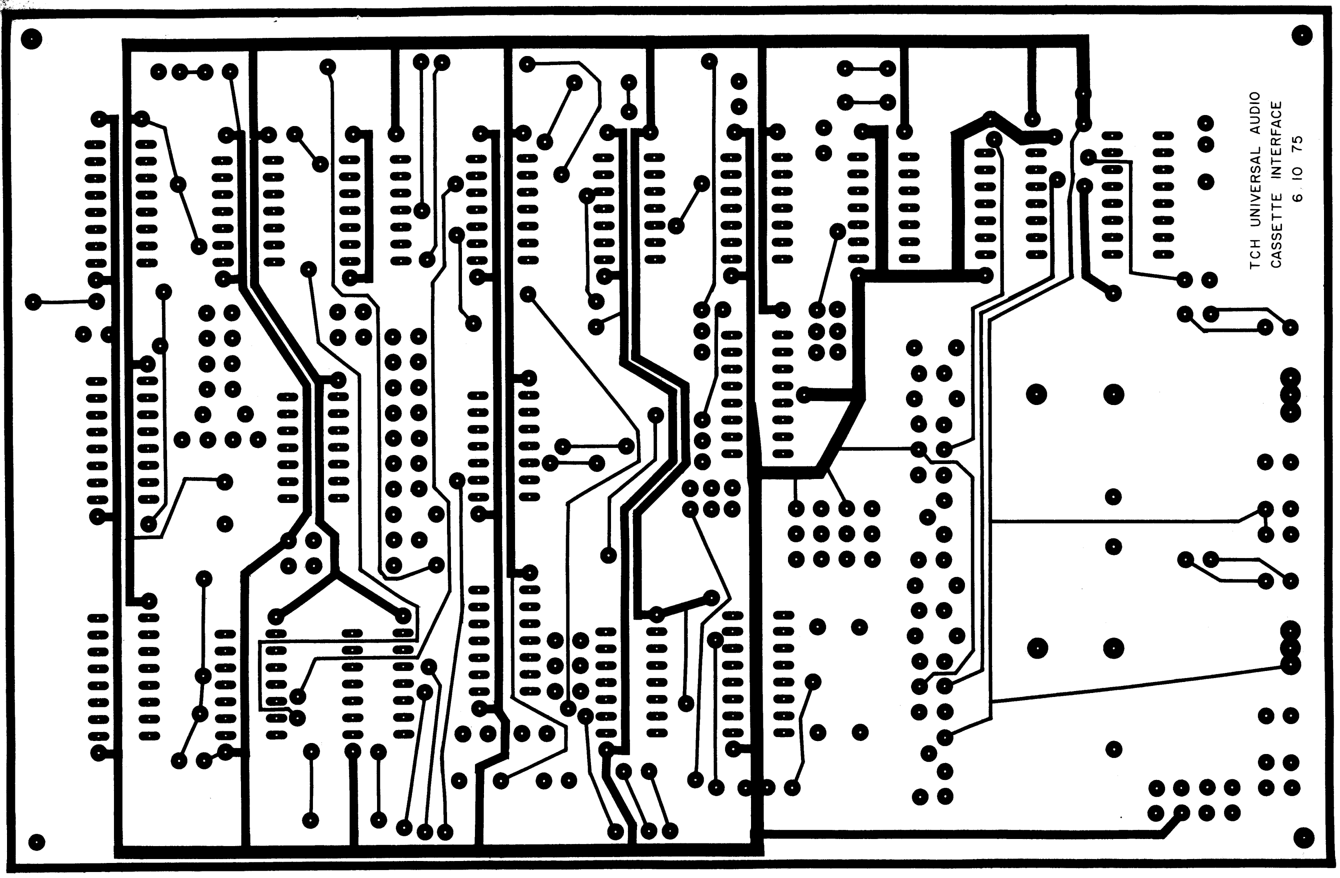
Pin # 1 All IC's same direction

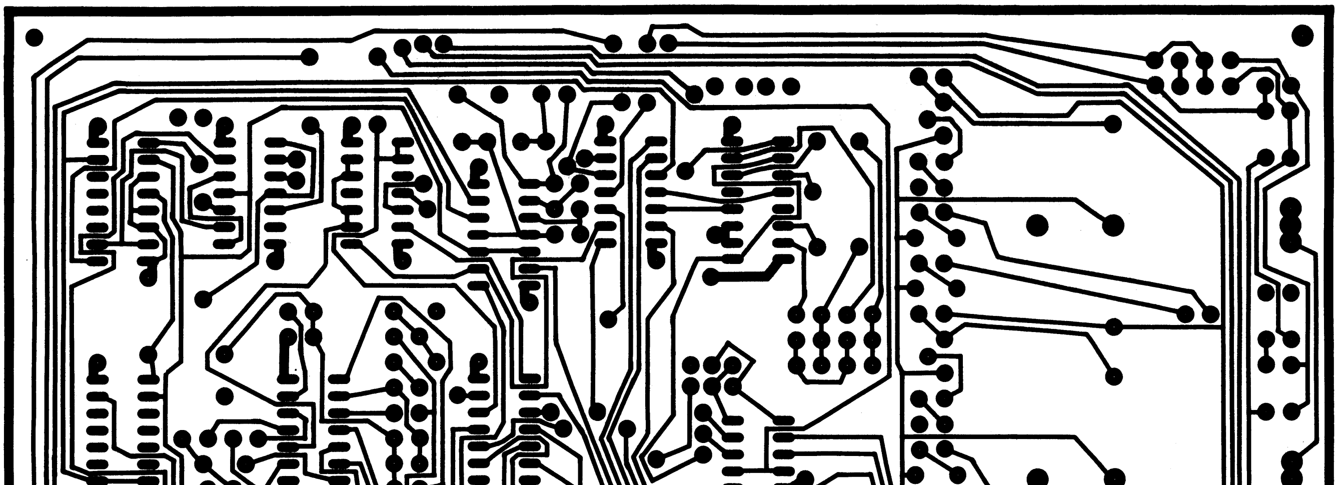
R7, 8, 47, 48, 49 Optional, See text

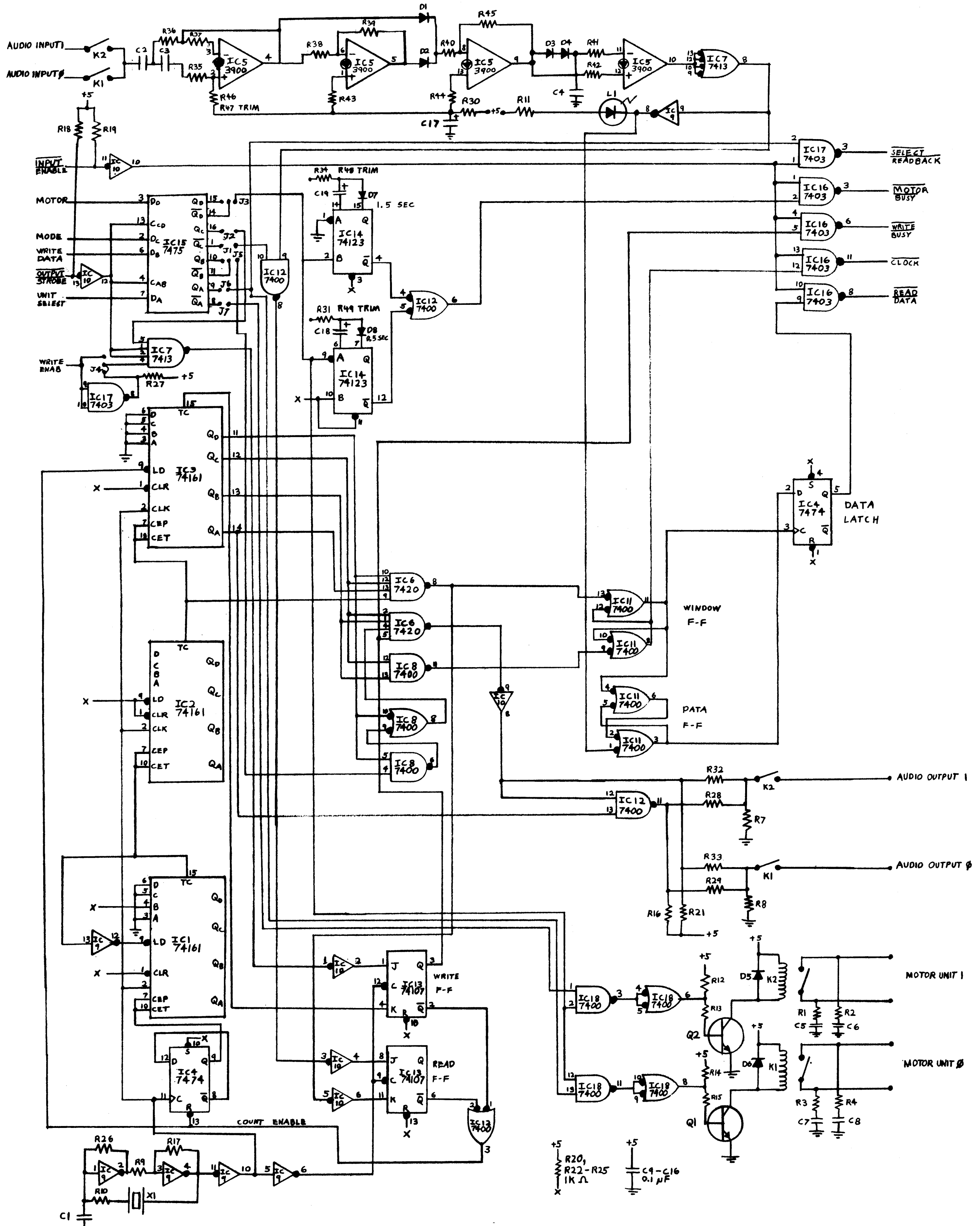
○ Denotes Mounting Holes



Resistor   Ceramic disc cap.   + Pin # 1


TCH UNIVERSAL AUDIO
CASSETTE INTERFACE
6 10 75


Wiring


AUDIO INPUT1
AUDIO INPUTØ
K2
K1
C2
C3
R36
R37
R35
R46
R47 TRIM
IC5 3900
D1
R34
R38
R43
D2
R45
R40
R44
R30
R11
C17
D3
D4
R41
R42
C4
L1
IC7 7413
IC9
+5
R18
R19
INPUT ENABLE
IC10
MOTOR
MODE
WRITE DATA
OUTPUT STROBE
UNIT SELECT
IC15 7475
J1
J2
J3
J4
J5
J6
J7
IC12 7400
R48 TRIM
C19
D7
1.5 SEC
IC14 74123
R31
R49 TRIM
C18
D8
0.5 Sec
IC17 7403
SELECT READBACK
IC16 7403
MOTOR BUSY
WRITE BUSY
CLOCK
READ DATA
WRITE ENAB
R27
IC3 74161
IC2 74161
IC1 74161
IC4 7474
DATA LATCH
IC6 7420
IC8 7400
IC11 7400
WINDOW F-F
DATA F-F
R32
R28
R7
R33
R29
R8
R16
R21
AUDIO OUTPUT 1
AUDIO OUTPUT Ø
IC13 74107
WRITE F-F
READ F-F
IC18 7400
R12
R13
R14
R15
Q1
Q2
D5
D6
R1
R2
R3
R4
C5
C6
C7
C8
MOTOR UNIT 1
MOTOR UNIT Ø
COUNT ENABLE
R26
R17
R9
R10
X1
C1
R20, R22-R25 1K Ω
C9-C16 0.1 µF

COMPUTER SNOWFLAKE DISPLAY by JIM PARKER

It is fascinating to watch the endless variety of patterns that snowflakes have. This program flashes through a countless number of "snowflakes" on an 8008 equipped with at least one page (256 bytes) of RAM and Hal Chamberlin's graphic display. Its always amazing to me how so many programs turn out to be easy to write with just an 8008 and a graphics display. I hope you will find that writing your own variations to this program are well worth the effort.

The program consists of two main parts. The first part uses the random number generator discussed last month to generate a table of from 4 to 7 random points connected by lines. The last point is made the same as the first point in order to draw a closed figure. The second part of the program takes this random figure and "reflects" it about the X, Y, 45 degree, and -45 degree axes to produce a symmetrical pattern. This is easily done by drawing GTABLE 8 different ways. Each time SYM1 is called, it exchanges the present X and Y coordinates in GTABLE and then draws them. Each time SYM2 is called, it inverts the present X coordinates in GTABLE and then draws them. By alternately calling these two routines four times each, we display the eight combinations of (Y,X), (-Y,X), (X,-Y), (-X,-Y), (-Y,-X), (Y,-X), (-X,Y), and (X,Y) in that order for every (X,Y) point originally in GTABLE. It can be shown that by doing this, we get a pleasing pattern with 4 lines of symmetry. The image is drawn 64 times before computing a new set of random numbers for the next image. Thus each image remains on the screen for approximately 3/4 of a second.

What about variations? For starters, use the same SYM1 and SYM2 routines but place a "hand drawn" figure in GTABLE. The first entry in GTABLE tells how many XY pairs are in the table, followed by the (X,Y) points which are to be connected. Avoid using -128 since it cannot be negated into +128. GTABLE should not cross a page boundary, that is, the H register should not have to change while addressing it. Another variation is to randomly add 0, 1, 2, or 3 to each coordinate instead of completely changing it every time. This produces a "blooming" effect. I hope that all computer artists will respond to this article and send in their own ideas for computer art.

```
        ORG     31000B

XMOV    EQU     10B
YMOV    EQU     11B
XSTOR   EQU     12B
YDRAW   EQU     13B

SNOW    LCI     13          SET LOOP1 COUNT
        LDI     H(GTABLE)
        LEI     L(GTABLE)   DE PTS TO GTABLE
LOOP1   CAL     RAND        RETURNS RANDOM BYTE IN A
        CPI     -128
        JTZ     LOOP1       GET ANOTHER BYTE IF A=-128
        LHD
        LLE                 HL INDEXES GTABLE
        LMA                 SAVE RANDOM BYTE
        INE                 BUMP GTABLE INDEX
        DCC                 CHECK LOOP COUNT
        JFZ     LOOP1       LOOP 13 TIMES
        SHL     GTABLE      PT TO GTABLE
        LAM                 GET RANDOM TABLE COUNT
        NDI     3
        ADI     4           A=4 TO 7
        LMA                 SAVE NEW COORD PAIR COUNT
        INL
        LDM                 D= FIRST X COORD
        INL
        LEM                 E= FIRST Y COORD
        RLC                 MULT TABLE COUNT BY 2
        ADI     L(GTABLE)-1
        LLA                 INDEX LAST X COORD
        LMD                 LAST X COORD = FIRST X COORD
        INL
        LME                 LAST Y COORD = FIRST Y COORD

        LDI     64          DRAW SNOWFLAKE 64 TIMES
LOOP2   SHL     GTABLE
        CAL     SYM1        EXCHANGE X AND Y AND DRAW
        SHL     GTABLE
        CAL     SYM2        INVERT X AND DRAW
        DCD                 DECREMENT COUNT
        JFZ     LOOP2       LOOP 256 TIMES
        JMP     SNOW        DRAW NEXT SNOWFLAKE
```

```
*                           RETURNS RANBOM BYTE IN REG A
RAND    SHL     SHIFT+3     PT TO SHIFT BYTE 4
        LBI     8           SET FOR 8 SHIFTS
        LAM                 LOAD SHIFT BYTE 4
RTOP    RLC
        RLC                 MOVE BIT 28
        RLC                 TO POSITION 31
        XRM                 X OR BITS 28 & 31
        RAL
        RAL                 MOVE NEW BIT INTO CARRY
        LLI     L(SHIFT)    PT TO SHIFT BYTE 1
        LAM                 LOAD SHIFT BYTE 1
        RAL                 ROTATE THRU CARRY
        LMA                 SAVE
        INL
        LAM                 LOAD SHIFT BYTE 2
        RAL                 ROTATE THRU CARRY
        LMA                 SAVE
        INL
        LAM                 LOAD SHIFT BYTE 3
        RAL                 ROTATE THRU CARRY
        LMA                 SAVE
        INL
        LAM                 LOAD SHIFT BYTE 4
        RAL                 ROTATE THRU CARRY
        LMA                 SAVE
        DCB
        JFZ     RTOP        REPEAT 8 TIMES
        RET

*                           ENTER WITH HL PT TO TABLE
*                           X'=Y
*                           Y'=X
*                           CONNECTS (X',Y') COORDS
*                           AFFECTS REGS A,B,C,H,L
SYM1    LBM                 GET TABLE COUNT
        INL                 PT TO X COORD
        LCM                 C=OLD X
        INL                 PT TO Y COORD
        LAM                 A=OLD Y
        OUT     XMOV        OUTPUT NEW X
        DCL                 PT TO X COORD
        LMA                 SAVE X'=Y
        LAC                 A= OLD X
        OUT     YMOV        OUTPUT NEW Y
        INL                 PT TO Y COORD
        LMA                 SAVE Y'=X
SYM11   DCB                 DECREMENT COUNT
        RTZ                 RETURN IF FINISHED
        INL                 PT TO NEXT X COORD
        LCM                 C= OLD X
        INL                 PT TO Y COORD
        LAM                 A= OLD Y
        OUT     XSTOR       OUTPUT NEW X
        DCL                 PT TO X COORD
        LMA                 SAVE X'= Y
        LAC                 A= OLD X
        OUT     YDRAW       OUTPUT NEW Y
        INL                 PT TO Y COORD
        LMA                 SAVE Y'= X
        JMP     SYM11       LOOP

*                           ENTER WITH HL PT TO TABLE
*                           X'= -X
*                           Y'= Y
*                           CONNECTS (X',Y') COORDINATES
*                           AFFECTS REGS A,B,H,L
SYM2    LBM                 GET TABLE COUNT
        INL                 PT TO X COORD
        XRA                 SET A=0
        SUM                 SET A= -X
        LMA                 SAVE X'= -X
        OUT     XMOV        OUTPUT NEW X
        INL                 PT TO Y COORD
        LAM
        OUT     YMOV        OUTPUT SAME Y
SYM21   DCB                 DECREMENT COUNT
        RTZ                 RETURN IF FINISHED
        INL                 PT TO NEXT X COORD
        XRA                 SET A=0
        SUM                 SET A= -X
        LMA                 SAVE X'= -X
        OUT     XSTOR       OUTPUT NEW X
        INL                 PT TO Y COORD
        LAM
        OUT     YDRAW       OUTPUT SAME Y
        JMP     SYM21

GTABLE  DST     15          SPACE FOR UP TO 7 COORDINATES
SHIFT   DST     4           32 BIT SHIFT REG USED BY RAND

        END     SNOW
```

SURPLUS SUMMARY

A new source for cheap 8080's has been found. Though he may not always have any on hand, the units he has are tested good in an ALTAIR 8800. The price is excellent, $80.00!

John Burgoon
Solid State Music
2102A Walsh Ave.
Santa Clara, CA 95050
Ph. 408/246-2707

Several people have written TCH asking for a source of information on the BIT 480 computer. Well, Stephen Corwin can help. Send him a SASE at:

4422 Clifford Rd.
Cincinnati, OH 45236

TCH no longer has 2102's however John Clark does. He has first quality National Semiconductor 2102-2 plastic package 1K RAM's. The cost is $3.00 each, any quantity. A minimum order of 8 pieces applies however. Send cash, money order, Master Charge, Bank Americard or he will ship C.O.D. with 25% deposit.

John A. Clark, Jr.
Box 5314
Charlotte, NC 28205
Ph. 704/332-3313

Mini-Micro-Mart has those hard-to-find 256 by 4 static RAM's. These are known variously as 8102's, 2101's, and 9102's depending on the manufacturer and the specs. As many of you know, the basic memory board of the ALTAIR machine uses these. According to MMM, the cheapest and easiest to obtain number is the Fairchild 9102 plus it has the best specs, however they can get the other numbers.

Mini-Micro-Mart
1618 James Street
Syracuse, NY 13200

CLASSIFIED ADS

There is no charge for classified ads in TCH but they must pertain to the general area of computers or electronics, and must be submitted by a non-commercial subscriber. Feel free to use classified ads to buy, sell, trade, seek information, announce meetings, or for any other worthwhile purpose. Please submit ads on seperate sheets of paper and include name and address and/or phone number. Please keep length down to 10 lines or less.

FOR SALE: New VARO 15KV rectifier - 50 MA, fast recovery -$1.00 each. TWO extender boards - 8 X 10 inches, 50 contacts per side, 8 contacts per inch, fair condition, sockets not included - $2.50 each. Nine track, 1/2 inch tape heads, unused, no specs - $10 each. All orders must include a SASE. Neil A. Benson, 10040 Nicollet, Bloomington, MN 55420, Ph. 612/372-7515

HP 55/65 BUFFS! Anyone interested in creating or swapping unusual or limited use programs? Am willing to discuss forming a group with TCH as a central clearinghouse. I can usually be found at all S.C. hamfests plus those in Augusta, Charlotte, and Shelby. W. Smyth, 1216 Bluefield Dr., Columbia, SC 29210

HELP! I have a Burroughs B250 CPU with manuals, extra power supplies, spare modules and core planes. 4800 bytes of core. Would like $200 for the whole batch. NO SHIPPING! Do you know anybody who would like this monster? If so call 313/538-1204 and ask for Jim or call 313/229-4091 and ask for Gene. J. A. Prest, 18704 Glatonbury Rd., Detroit, MI 48219

FOR SALE: HF Data Modems (2) Collins TE216D-16D, 1200/2400 BPS. Make offer. Otis Robinson, 7048 Lee Park Ct., Falls Church, VA 22042 Ph. 703/534-9069

FOR SALE: Brand new Intel 8080 and 16 new 2102 RAM's, $160. Jim Rea, 731 E. Harmony, Fullerton, CA 92631, Ph. 714/871-2220

1ST PRIZE: 16 BIT MICROCOMPUTER CHIP!

CONTEST!

GODBOUT

BILL GODBOUT ELECTRONICS
BOX 2355, OAKLAND AIRPORT, CA 94614

2nd prize: 8080 cpu

3rd prize: 8008 cpu

We were 1st to offer the 8008 to hobbyists over 16 months ago; now we're setting the pace again with a powerful new 16 bit microcomputer IC in a 40 pin DIP, made by: the SECRET MICROCOMPUTER Co!

YOU MAY WIN ONE OF THESE CHIPS --- SIMPLY:

1) Reveal the Secret Microcomputer Co.'s true identity
2) Tell us in 25 words or less why you should receive a free chip

If you can convince our jaded judges, in a form suitable for use in this family (?) magazine, you win.

★ GOOD LUCK!

CAVE grafix

FINE PRINT: ALL ENTRIES MUST BE POSTMARKED BY JUNE 31 AND BE IN OUR HANDS BY JUL 7, 1975; ENTRIES BECOME PROPERTY OF BILL GODBOUT ELECTRONICS. ALL CONTESTANTS RECEIVE A DATA SHEET ABOUT OUR FIRST PRIZE FOR THEIR TROUBLE. WINNER WILL BE NOTIFIED BY AUG. 1, 1975. IF YOU DON'T WIN ANYTHING THIS TIME AROUND DON'T FEEL TOO BAD; ENTER OUR COMING CONTEST FOR A COMPLETE 16 BIT MICROCOMPUTER KIT. THESE CONTESTS SPOTLIGHT PRODUCTS TO BE INTRODUCED BY US IN THE FALL OF '75. SEND ENTRIES TO "TCH CONTEST", BOX 2355, OAKLAND AIRPORT, CA 94614.

Due to publication delays, Godbout Electronics has agreed to move deadlines for the contest as follows: Entries must be mailed by July 15, received by July 22, and winners will be notified by Aug. 15.

THE COMPUTER HOBBYIST
Box 295
Cary, NC 27511

ADDRESS CORRECTION REQUESTED

THE COMPUTER HOBBYIST

Volume 1 Number 7

By the time you read this TCH will have gone to another hamfest, this one in Shelby, N.C., so Hi all you folks whom we just saw. TCH will be going to some other hamfests with our system this fall. On September 13 we will be at the ARRL national convention in Reston, VA at the Sheraton Motor Inn. Hamfests in Gaithersburg, MD and Stone Mountain, GA are also possibilities for this fall.

Upcoming articles for TCH include a series on the IMP-16 starting with the next issue. Also, TCH now has an Altair 8800, so issue 8 should usher in a series of articles on I/O and interfacing.

Now a few words about TCH's health and our future. We are slow (and how) in cranking these things out. Consequently, we are abandoning our claim to be a monthly and only claiming to be an "almost" monthly, i.e., we reserve the right to publish that often if we can manage. Don't be upset about your hard earned coins which you sent us however, for they will still get you the same number of issues (12 for $6). For those of you who are already in our cassette based record keeping system, your mailing label begins with "Sub #### Last ##". The number after "Last" is the issue number with which your subscription expires. If this number differs from your records please notify us immediately. If your mailing label has any other format, hang in there; you will be in the new system in a couple of months. As for TCH's health otherwise, its good. We now have over 1500 subscribers and have yet to see slack in our mail. We are also planning to go to 3RD class postage. This will make getting TCH slower but will allow us to put out larger issues with no increase in postage. Also TCH is now actively seeking outside authors and advertizers.

One last item, TCH wants to publish a list of clubs accross the country, so if you want your club listed drop us a card and you will be in our next issue. Be sure tc include important facts such as meeting times and locations and names of club leaders.

TCH AUDIO CASSETTE STANDARD ROM

by: Richard Smith and Hal Chamberlin

The TCH audio cassette ROM provides basic support software for the TCH universal audio cassette interface. Functions include reading a record into a buffer in main memory, writing a record from a buffer, IPL (initial program load, cold start, bootstrap, etc.) from cassette tape, and interface control on both units. At the beginning of the ROM is a jump vector with an entry for each of the four main functions: CTROM - read record, CTROM+3 - write record, CTROM+6 - Control routine, and CTROM+9 - IPL routine. This jump vector allows software compatibility with possible updates to this program and similar packages for the Phi-Deck by keeping the call addresses of the routines invarient.

The read record routine reads the next record from the tape into a user specified buffer area in memory. When called, register A should contain a unit select mask specifying the tape unit to be used. A mask of all zeroes specifies unit 0 while a mask with the unit select bit (octal 020) on specifies unit 1. Equates between the unit number and the unit select mask are a convenient way to keep this straight. H and L should have the address of the buffer that will receive the data read. When the read record routine returns, register B will contain the length of the record read in and H & L will point to the next available buffer word. The condition code indicates whether a CRC error was encountered during the read. Everything is OK if the ZERO flag is on. Note that the read routine uses all of the registers and except for B, H, and L they will be in an undefined state on return.

The write record routine is the exact complement of the read routine. Register A contains the unit select parameter as before and H & L points to the buffer to be written from. Register B specifies the length of the record to be written. Since errors cannot occur when writing, the only return information is the content of H & L which points one beyond the last byte written.

At this time the control routine does nothing except reset the interface. It was included for future compatibility with a Phi-Deck interface in which case it would handle rewinding, beginning of tape search, etc.

The IPL routine can be used to read a cassette tape file into memory and branch to it. It is particularly useful after power on or a major crash since one needs only to branch to it (CTROM+9) and put the appropriate cassette into the recorder to recover. Preprogrammed into the ROM is the address in memory to read into and jump to after reading. The IPL program data format is simply memory image starting from the load address. The routine can handle multiple records so program length is unrestricted. It stops reading when an end-of-file record (zero length) is encountered and branches to the program loaded. 8080 users need not be concerned with the stack pointer since the routine initializes it to a preprogrammed address. This address should be specified when ordering the ROM. Since the 8080 predecrements the stack pointer when pushing, the address specified should be one more than the address of the top of the stack. This IPL routine requires 10 stack bytes but for future compatibility a few more should be reserved. If an I/O error is encountered during loading, the routine halts by means of an infinite loop.

Anpendix I shows an example program which copies the tape on unit 0 onto unit 1. The first step taken by the program is to reset the tape interface by calling CTCN. Next, a record is read into a memory buffer starting at BUFF. This buffer should be 255 bytes long to accomodate the longest possible record. If no read errors were detected, register B is tested for zero to determine if an end-of-file record was read. If so, an end-of-file record is written by calling CTWR with a length specification of zero and the program halts. Otherwise H & L are reset to the beginning of the buffer, A is set for unit 1, B is left unchanged and CTWR is called to write the record just read. The program then loops to read the next record.

Due to Murphy's 19th law (a program is always 30 bytes too long), the coding in the ROM can be tricky in places. Hopefully this discussion, the detailed flowcharts, and the commented listings for 8008 and 8080 will help. One overriding consideration was that no memory be used for temporary storage, i.e., all counters, pointers, etc. should be kept in registers. In general the program structure follows the record structure with a separate routine for each component of the record format. Perhaps the most common trick is to fall into an adjacent subroutine rather than calling it if that is the last operation to be performed, thus saving 3 bytes. Elaborate equates are used to make the program more readable.

The control routine resets the interface by sending it a word of all zeroes. This selects unit 0, turns off both motors, and sets read mode.

After turning on the specified unit's motor, the read routine goes into a loop to search for the data ID. The ID is located by defining a 16 bit shift register using B (high part) and C. After shifting BC left by one and reading the next bit from the tape into bit 0 of C, the contents of BC are compared with the ID pattern. The only way to have an equal compare is if the last 16 bits read were the data ID pattern. Clever arrangement of the code and registers results in the CRC register (registers D and E) being zero when the ID is detected. Next, the record length is read in. Saving the record length for use by the calling program while also using it for a loop count presented some problems as all of the registers were already used. The 8080 solution is simple, push it onto the stack. The trick on the 8008 is to save the length in the buffer position reserved for the next byte. When a data byte is to be stored in the buffer, the count is loaded back into A and the byte stored from C. This works fine as long as data is read into operable memory. When the byte count reaches zero, the CRC is checked by simply reading in two more bytes but not storing them. If the CRC is zero after this, the record was OK. Finally the motor is turned off, the CRC register is tested for zero, and a return is made.

CONTINUED ON PAGE 4

LETTERS

We at TCH will publish a few of our more interesting letters each month along with comments by the staff.

Editor:

I'm curious about the May 1975 issue. I received it on July 7. Since it was first class are you getting cheated by paying first class and getting bulk service?

My main reason for writing is to compliment the writer of the May editorial. It is a damn good, well balanced piece of journalism, even though I don't totally agree with it. Hope to see more like it in future issues.

I like the graphics display and will probably try to get one together in the future. Was wondering if the MP digital display might not be interfaced with the CRT portion of the display without too much trouble.

You guys really put together good information and detailed descriptions--thanks.

Interested in your PC board for the cassette interface, but due to vacation expenses will be unable to order until the end of the month. If the supply is limited, could you save me one?

Bill Fuller

No the post office isn't cheating us, we are just slow and tried to keep the dates on the issues sequential in spite of that fact. For more comments see the discussion on the front page.

Your letter and the two following are just some of the many we received mentioning the Altair editorial. They are typical of the opinions expressed. TCH is glad you folks do read and care.

The MP digital display cannot be interfaced to the TCH graphics system, however any programs written for it should be easily convertable to run TCH's display (the reverse is NOT true).

Cassette boards are being supplied on a continuing basis. Notice will be given before they are discontinued. To date over 75 units have been shipped. Also due to our early success TCH is regrettably out of relays so do not order them now.

Gentlemen:

As an owner of an Altair 8800 and a computer hobbyist I would like to comment on your editorial in the May '75 issue of TCH.

I purchased the basic computer during the initial promotional effort by MITS. I have always considered it an excellent buy. Components are of good quality and the kit assembly instructions were above average.

MITS's marketing strategy seemed to be directed toward the average hobbyist on a limited budget. The basic 8800 fullfilled this requirement nicely. The MITS add-ons and peripherals are definitely overpriced and, I suspect beyond the means of most 8800 purchasers. I happen to agree with the "loss leader" theory.

In addition, MITS has not offered a single piece of software at a reasonable cost. Few people have the time and resources for software development. It should not be necessary to point out the usefullness of a computer without software. MIL's MONITOR-8 package, which was free for the asking, showed what can be done.

I have asked MITS for additional technical data for trouble-shooting and maintenance if this should become necessary. Their answer "data is not available at this time" is somewhat difficult to swallow. Another ripoff?

Fortunately others have tried to fill some of the gaps: The Digital Group, MiniMicroMart, Processor Technology, and TCH of course. This should not relieve MITS from their responsibilities implied in advertisements and in the PE article which started it all.

Your primary concern should be a look at a system (CPU, hardware, and software) from the standpoint of the hobbyist. Although I agree with your editorial, this is one point you obviously missed.

I enjoy your publication and it has helped me greatly getting started in the micro-computer business. Hope to see more articles on 8080 based systems. Keep up the good work.

A. P. Stumpf

Gentlemen:

I have just received, and devoured in one day, the first six issues of TCH. Excellent!

I particularly appreciated the editorial in issue #6. Your analysis displays both understanding and courage. Understanding of the interaction of economics and technology in the fast developing areas of computers and microelectronics, and courage to present unpopular facts. It is unfortunate, but many people find it easier or more comforting to blame others, often "big business", than to recognize their own motivations, or lack of foresight. We should all firmly affix in our mind that even the ALTAIR 8800 will be obsolete, though still useful, in at most two years. We should further recognize that even IBM can not limit the speed or spread of technological development.

All computer hobbyists should stop and consider where they will be, if in the next few years Digital Equipment Corp. manages to produce the LSI-11 for about $100. We are in an exciting field, but that excitment is because of, not in spite of. technological progress. If we recognize that, in the not too distant future, home computers will be as ubiquitous as component Hi-Fi equipment; we must expect tremendous improvements in both computer power and economies of production. As the market expands, more and more capital will be invested in research and production improvements, and we will be the beneficiaries.

Computer hobbyists should not glory in the exclusiveness of their hobby. They should anxiously look forward to the entrance of many participants, because these participants will attract real big business, and capital.

Micheal A. Sicilian

THE COMPUTER HOBBYIST
Founded October, 1974

Stephen C. Stallings - Managing Editor
Hal Chamberlin - Contributing editor
Jim Parker - Contributor
Edwin Tripp - Photographer
Richard Smith - Programming consultant

THE COMPUTER HOBBYIST is published at intervals approaching monthly near Raleigh, N.C. The subscription rate for 12 issues is $6.00 for persons in the U.S.A., U.S. possessions, Canada, and Mexico. Subscriptions in all other countries are $8.50 per year for surface mail or $13.50 per year for air mail. All back issues are available and cost fifty cents each for persons in the U.S.A., U.S. possessions, Canada, or Mexico. For all other countries back issues cost 75 cents each by surface mail or $1.25 each by air mail. Remittances from countries other than the U.S.A should be in the form of a bank or postal money order.

THE COMPUTER HOBBYIST is seeking paid contributors. Material to be submitted should be typed or neatly written and must not appear to be soliciting business for any firm. For details, please inquire.

Advertizing space is available to interested forms. please inquire for rates and formats.

All correspondence should be addressed to:

THE COMPUTER HOBBYIST
Box 295
Cary, NC 27511

EDITORIAL

Unlike ~~our other articles and commentaries the edit~~orial will be mostly opinion, sometimes that of a single staff member, and other times that of TCH as a whole. Reaction to the editorials, either supportive or dissenting, is welcome.

Are console control panels really necessary in hobby computer systems? By control panel I mean the array of toggle switches, push-buttons and lights often seen on the front panels of computers. Lets look at some history and see how the traditional control panel evolved.

Early computers were used in much the same way a lot of people are using their Mark-8's and Altair's now, programs were entered and results obtained from the front panel. One major difference though was that these early machines displayed all of their registers and major logic states as well for maintenance purpose. The control panels on these beasts were indeed large and impressive and still persist in the minds of the public through the efforts of Hollywood. Large computers now (IBM 370-168) have all of their console functions microprogrammed and channelled through an ordinary keyboard and alphanumeic CRT display, no more switchs and lights.

Minicomputrs have also gone through much the same evolution. Early ones displayed all of their registers any of which could be loaded from the console switches. Later the console was reduced to simply a data bus monitor and console access was essenially restricted to memory. Now most minicomputers are priced less console which is a $200 to $500 optional feature.

Why the trend away from consoles? The answer is that software and a basic I/O device such as a teletypewriter can provide much more convenient and extensive console functions than even an elaborate control panel. Any serious system is going to have the requesite I/O device anyway. Even a simple console emulator program (we will call it a "DEBUG" program) allows memory contents to be printed and entered at typing speed, allows registers to be examined and changed, and can start user program execution at a particular address all with simple keyboard commands.

Powerful debugging aids are available in the more advanced debug programs. These can take the form of either "trace" or "breakpoint" functions. A typical trace routine will print the instructions executed and all of the registers that are modified between trace limit addresses. The trace limits are set by the user from the keyboard. Trace routines work either by simulating execution of the user's program or by clever use of the interrupt system. The major disadvantage of trace, slow execution outside the trace limits when trace is enabled, is avoided by the breakpoint function. Typically the user can set one or more breakpoints at particular points in his program with keyboard commands. When program execution hits a breakpoint, the breakpoint address and the registers are printed and then execution resumes. The breakpoint routine works by inserting calls to itself in the breakpoint locations and keeping track of the instructions replaced by the calls. Breakpoint routines are typically smaller but require more planning to use effectively. Either method substitutes nicely for single cycle console controls and provides a written record of program execution for detailed study. After all, large computer users have only a memory dump after catastrophic program failure to go on.

The fact is that microprocessors were never intended to have consoles. For that matter, the older ones (including the 8008 and 8080) were designed as logic replacements and dedicated controllers rather than general purpose computers for problem solving. These two facts are readily apparent to an engineer who tries to design a good console or integrate the microprocessor into a general purpose system. Nevertheless both tasks can be accomplished such as in the MITS system or the Mark-8 system. Considerable money is spent in either case on console logic, quality toggle switches and buttons, and a well labelled front panel. Often compromises are made to accomodate the single cycle function without letting the bus control logic get out of hand. As mentioned before, any serious user of these systems has or will have the keyboard-printer or display necessary to support a software console and once he tries a debug program, the hardware console may never be used again! Two interesting side points are that most complaints about existing hobby computers relate to console malfunctions and that the newer systems such as SPHERE have no control panel.

How would a consoleless computer look and operate? There would probably be three switches and two or three lights. One switch would be a power on-off toggle, one a reset button, and one an interrupt button. The first light would be a power on indicator, the second would indicate when the CPU is halted and the optional third light would indicate whether interrupts were enabled or not. Some processors may require a start button if interrupt doesn't work when the machine is halted, e.g. the IMP-16.

Now how would one get this underendowed creature started when the power is turned on? In a well designed system, a "power on reset" circuit (POR) will effectively press the console reset button when power is first applied. The reset signal should go to all of the peripherals resetting them to an idle state and it should cause the CPU to jump to a program stored in read only memory. At this point two possibilities exist. One is that enough ROM is available to hold the entire debug program in which case the restart procedure is complete. The other possibility is that a small program called a bootstrap loader is all that is in the ROM. The loader would then proceed to read the debug program into regular read-write memory from an input device such as a paper tape reader, cassette, etc. and jump to it. The bootstrap data format is generally simplified and inefficient in order to minimize the bootstrap loader size. In either case we now have control of the system through debug program commands. Ideally the console interrupt button would be seperate from the I/O interrupt system and always enabled. Pressing the interrupt button would cause an "interrupt entry" into the debug program which would save the registers, status, and return address for examination and alteration by the user. A debug command would be available for resuming execution at the point of interruption. That is all there is to it.

Some microprocessors are better adapted to this mode of operation than others. In particular the Motorola 6800 and the National PACE and IMP-16 seem to be designed with this specifically in mind. Nevertheless, any of the popular chips can be effectively operated without a console.

BOOK REVIEW by Hal Chamberlin

Machine Language Programming for the 8008, Wadsworth, Nat, Scelbi Computer Consulting Inc., 1322 Rear Boston Post Road, Milford, CT 86460

We have been constantly getting requests for information and articles on basic level machine language programming for all of those hobby computers out there. Well here is the answer! This 168 page book has all of the answers and guidance that a beginning programmer could want. It is far more detailed and down to earth than even DEC's classic Small Computer Handbook that until now had been the best tutorial publication available on the programming of small binary computers. The coverage is so broad and well done that some of the articles that we had planned on programming will have to be altered or dropped to avoid duplication.

As the title implies, the contents are directly applicable to the 8008 microprocessor. The large number of fully commented example programs and routines are directly usable on the 8008. Neither the 8080 nor any other microprocessor was mentioned but most of the concepts and techniques presented would be applicable to other machines. Many of the added instruction functions on the 8080 are developed as subroutines in the text, thus where a subroutine might be called in a sample program, the 8080 user could simply supply the appropriate instruction.

One noticable feature of the book was a complete lack of any commercialism whatsoever. As the reader may know, Scelbi is a manufacturer of 8008 based systems for hobbyists and schools and also sells extensive software for the systems. Scelbi computers are seldom mentioned and none of the text or examples made use of or even mentioned any particular features of the Scelbi system. Even the chapters on input/output programming were kept general; never mentioning the hardware or software techniques used in Scelbi I/O devices. In short, one would never suspect that the book was written by a manufacturer of the computer it describes.

The book is in the form of 8 1/2 by 11 inch pages bound in a soft cover report binder with metal binding tabs through the three hole punching. Offset printing of good contrast on one side of the pure white medium weight paper is employed. The type was obviously set on a teletype machine in all caps with a cloth ribbon so the character quality was not particularly good but nevertheless easy to read. The absence of typographical errors indicates that the author made effective use of his editor program. Space utilization on the printed side of the paper was good due to single spacing and narrow margins. Some of the simpler drawings were formed with teletype characters much like those seen in IBM manuals.

In the introduction a very convincing (and accurate) argument for machine language versus high level language programming of hobby computers is presented. The first chapter which is 21 pages long gives an original, truly readable description of the 8008 instruction set. An interesting approach is taken in explaining the op-codes. Rather than utilizing binary op-codes and then having to explain binary-to-octal conversion so early, octal op-codes are used initially. The result is that explanation of operation encoding is much simpler. For example, the load register immediate instructions have the form 0R6 where R is replaced with the register number to be loaded.

Now that the beginning programmer has been introduced to the "tools" he will use, he should be ready for the second and third chapters which discuss the steps used in program development and some necessary programming skills. A clear, accurate understanding of the problem to be programmed and the desirability of a flowchart are emphasized as prerequisites to a smooth, rewarding program development cycle. Number conversion is taken up as a programming skill along with the use of memory maps and coding sheets. Through the use of examples, the desirability of using an editor and assembler for long programs (over 100 instructions) is demonstrated. Manual coding is developed as an actual hand assembly process so that the use of an assembler should come quite naturally to the reader when he becomes advanced enough to need it.

Chapter 4 will be of great value to the beginning programmer because it is the chapter on basic programming techniques. In 37 pages the discussion proceeds from how to clear the accumulator to development of search and sort routines. Several utility routines, some of which substitute for 8080 instructions, are also developed in this section and their use is explained. Only the simplest search and sort algorithms that get the job done are presented. The more advanced and efficient methods are left to the computer science text books. The examples are always based on actual, concrete requirements, never on abstract or theoretical considerations.

I suspect that many people may buy this book solely for the contents of chapter 5. This is the section on arithmetic and is 45 pages long. The discussion starts with multiple precision add and subtract and proceeds to develop general purpose multiple precision add, subtract, and complement routines. After a general discussion of binary fractions, floating point notation is introduced. From here to the end of the chapter the six floating point operations (addition, subtraction, multiplication, division, ASCII-to-float, and float-to-ASCII) are discussed in detail and the algorithms converted into assembly language code. In other words, chapter 5 includes a floating point package. The four math routines were punched up and assembled at TCH and they appear to work correctly (be sure to consult the errata sheet supplied with the book). We have not tried the conversion routines but they should work also. Extensive use is made of subroutines developed in earlier chapters. According to the author, the code was optimized for ease of explaination and understanding and as a result is apt to be both time and space inefficient. The reader is then encouraged to rewrite the routines for greater efficiency once he thoroughly understands them in the present form. Probably the greatest speed gain will be in the multiply and divide routines and the greatest space gain in the conversion routines. Scelbi offers a listing with object code but no comments to owners of the book for $5.00. It is well worth the price if an assembler is not available.

The last four chapters discuss such diversified topics as input/output programming, real-time programming, and creative programming concepts. The depth of discussion is not very great but enough is said to get the reader thinking in the correct terms.

In summary, Machine Language Programming for the 8008 is a must acquisition for the beginning programmer, especially a hardware man. Even the experienced programmer should sit down with a copy for an hour or so because he is bound to discover something he has not known or thought of before.

CONTINUED FROM PAGE 1

The read bit routine (CTRD) waits for the clock from the interface to make a high-to-low transition and then reads a bit into the low order position of C shifting the remaining bits left. The bit read is also combined with the CRC register in DE by CTCC before returning. Thus all bits read will be factored into the CRC. The read byte routine (CTRB) simply calls the read bit routine 8 times to accumulate a new byte in register C. Due to the lack of registers, the count is done by setting a dummy bit in position 0 of C and calling CTRD until it is shifted into position 7. The routine then falls into CTRD for the eighth bit.

Operation of the write routine is similar to read. When entered, CTWR waits until the motor busy status becomes zero to insure a proper record gap after the last write operation. If more than .5 second has elapsed since the last operation, the status will already be zero so there is no waiting. The motor on the specified unit is then turned on and another wait on motor status is performed before writing is started. The 32 leading ZERO bits are written by effectively calling the write byte routine (CTWB) 4 times with zero data. The data ID is then written by two calls to CTWB with the appropriate data. At this point the CRC register in DE is zeroed in preparation for the remainder of the record. Writing of the record length is a bit tricky in order to conserve space. The length is first loaded into C from B and a jump into the middle of the write data bytes loop is taken. The byte count in B is incremented in order to compensate for the additional pass thorough the bottom of the loop. The routine still works properly for zero length and maximum length records in spite of this trick. Since HL is incremented at the top of the loop, it will point to the last byte written + 1 on exit. When writing the CRC, the low half is saved in B while writing the high half first because the act of writing the CRC also changes the CRC. Finally the trailing zeroes are written, the motor is turned off and a return is executed without waiting for the motor to stop.

The write bit routine shifts register C left by one and writes the bit shifted out. The test for write busy is done first to overlap the serialization processing with the time necessary to write a bit. It is necessary to read the the unit select bit from the interface so it can be combined with the write command bits sent to the interface. Otherwise, the unit select bit would be destroyed. Each bit written out is combined with the CRC as in the read bit routine. The write byte routine (CTWB) writes register C by calling CTWD 8 times. Bit counting is done by calling CTWD once and then appending a "stc_" bit in the vacated position 0. When CTWD has been called enough times so that C contains 10 000 000, the remaining bits have been written and the routine returns.

During the course of writing these routines, several interesting observations were made. One was that the 8080 code was not significantly shorter. The two byte I/O instructions and the all-in-registers requirement were mainly responsible. Another was that there is a lot of work between an operable program and a fully optimized one (notice that we changed our minds on the organization since issue 6). Finally, this program might serve as a reasonable, notrivial benchmark for comparing microprocessors.

APPENDIX 1

```
;       EQUATES

CTROM   EQU  ________      ; CASSETTE TAPE ROM (CTROM)
CTRR    EQU  CTROM+0       ; READ RECORD ROUTINE
CTWR    EQU  CTROM+3       ; WRITE RECORD ROUTINE
CTCN    EQU  CTROM+6       ; CONTROL ROUTINE
CTIPL   EQU  CTROM+9       ; IPL ROUTINE

CTUNO   EQU  000Q          ; UNIT 0 SELECT
CTUN1   EQU  020Q          ; UNIT 1 SELECT


;       COPY PROGRAM

COPY:   MVI  A,0           ; RESET THE CASSETTE TAPE INTERFACE.
        CALL CTCN          ;
LOOP1:  LXI  HL,BUFF       ; READ INTO THE BUFFER A RECORD FROM
        MVI  A,CTUNO       ;    CASSETTE TAPE UNIT 0.
        CALL CTRR          ;
        JNZ  ERROR         ; BRANCH IF AN I/O ERROR.
        MOV  A,B           ; BRANCH IF AN END OF FILE RECORD.
        ORA  A             ;
        JZ   EOF           ;
        LXI  HL,BUFF       ; WRITE OUT THE RECORD TO CASSETTE TAPE
        MVI  A,CTUN1       ;    UNIT 1.
        CALL CTWR          ;
        JMP  LOOP1         ;
EOF:    MVI  A,CTUN1       ; WRITE THE END OF FILE RECORD TO
        CALL CTWR          ;    CASSETTE TAPE UNIT 1.
        HLT                ; DO A DONE HALT.
        JMP  COPY          ; GO START ANOTHER COPY.
ERROR:  HLT                ; DO A HARD ERROR HALT.
        JMP  ERROR         ;
```

NEW PRODUCTS

With this issue we are starting a new products column. In it we will list some of the new commercial products of interest to the computer hobbyist. Comments made will, in general, be condensed from the manufacturer's literature. Occasionally we may make a comment of our own if a particular feature is unusually impressive. Listing in this column, of course, does not imply endorsement of the product by TCH.

A new, nicely packaged microcomputer kit is being offered by Comp-Sultants. It is based on the Intel 4040 CPU chip. The basic machine has 256 bytes of program memory, (the 4040 has separate program and data memory) one input port, one output port, a control panel, and of course the CPU chip. An unusual feature is that the entire basic machine, including the control panel, fits on one large PC board. The machine is housed in a handsome but inexpensive metal cabinet. There is sufficient room in the cabinet to expand the memory to 8K words (each 2K memory board adds 8 I/O ports as a bonus). The basic kit costs $275 and the assembled unit costs $375. These prices were taken from a press release and do not agree with the glossy sheet price of $300 and $400 respectively.

COMP-SULTANTS, Inc.
P.O. Box 1016
Huntsville, Ala. 35807

Cramer Electronics, a well known industrial distributor of electronic components, has introduced a line of computer kits called Cramerkits. A Cramerkit consists of all of the parts (IC's, resistors, caps, etc.) necessary to build a microcomputer along with a documentation package including circuit diagrams and wirelists. NO interconnection hardware is supplied. The buyer is expected to build the system on wire-wrap cards or the equivalent. Cramerkits have or will be introduced for every major MOS microprocessor and probably some of the bipolar microprocessors. Available now are kits for the Intel 8080, the TI 8080, and the Motorola 6800. All kits have 1024 bytes of RAM, 1024 bytes of eraseable ROM (using the new 2708 8K bit easy-to-program eraseable PROMS), 4 input ports, 4 output ports, basic control panel, audio cassette interface (as published in Popular Electronics Sept. 1975), and RS-232 or TTY current loop serial I/O. The PROM comes with a debugging program and cassette read/write routine already programmed in. The TI kit includes a TMS-5501 "utopia chip" (UART, 5 interval timers, 8 level priority interrupt control, 2 I/O ports) whereas the other kits rely on software for these functions. The price of $495 is the same for any of the kits. A 2708 PROM programmer kit which connects to two of the output ports will be available shortly for around $70.

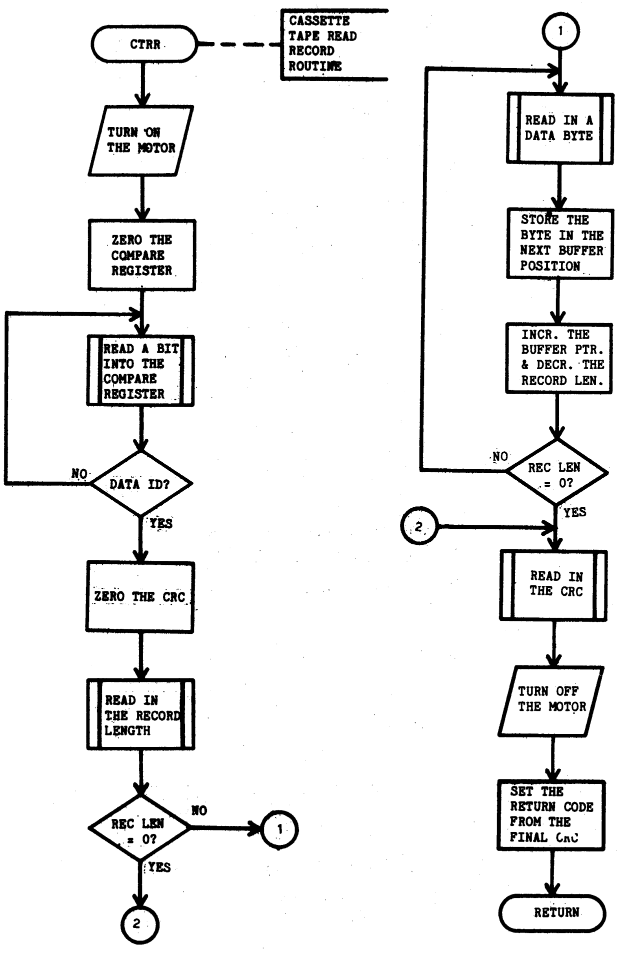
Cramer Electronics, Inc.
85 Wells Avenue
Newton, Mass 02159

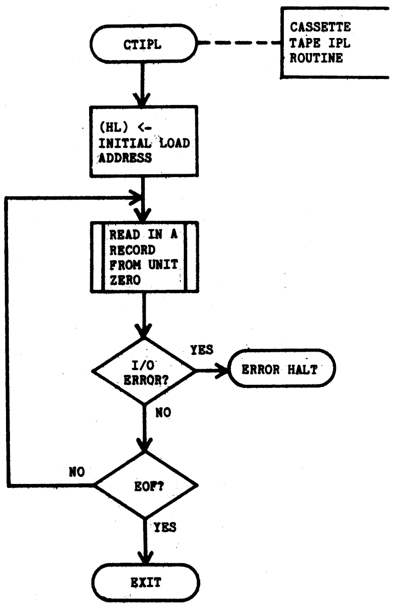
USE IT FOR WHAT IT WAS INTENDED?

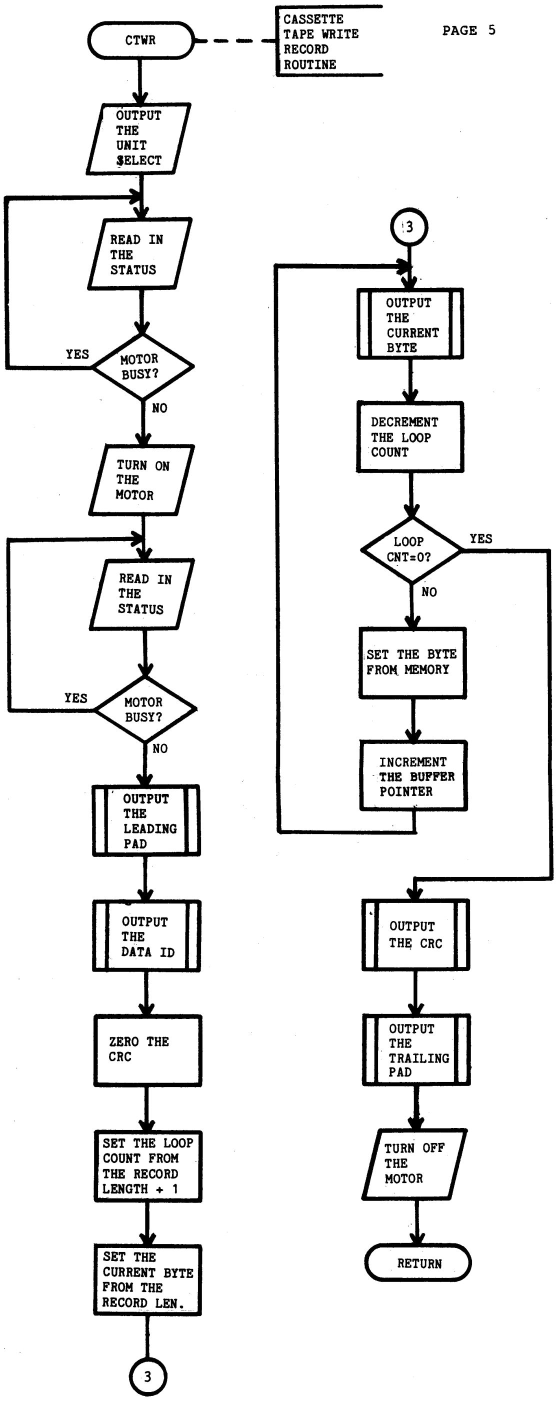
The Cyclops computer compatible TV camera (see Popular Electronics February 1975) is the best example of looking beyond the manufacturer's spec sheet we have seen in a long time. It seems that the "special image sensor chip" used is simply an MK4008 1K dynamic RAM with the metal cap replaced by a glass one. This dynamic RAM is unusual in that reading a location does not refresh it; a special refresh cycle is used instead. The rate of charge leakage from the storage capacitors in the dynamic cells is dependent on temperature and, you guessed it, ambient light. To use the modified memory as an image sensor, ONES are written into all locations. Then the memory is scanned repetitively with a scan count maintained for each bit position. The number of scans before a given bit returns to ZERO is inversely proportional to the amount of light falling on that bit.

Replacing the cap on the memory requires clean room conditions and a controlled atmosphere. A kit including a PC board and all parts is available however from H. Garland, 26655 Laurel Lane, Los Altos, CA 94022 for $55.00.

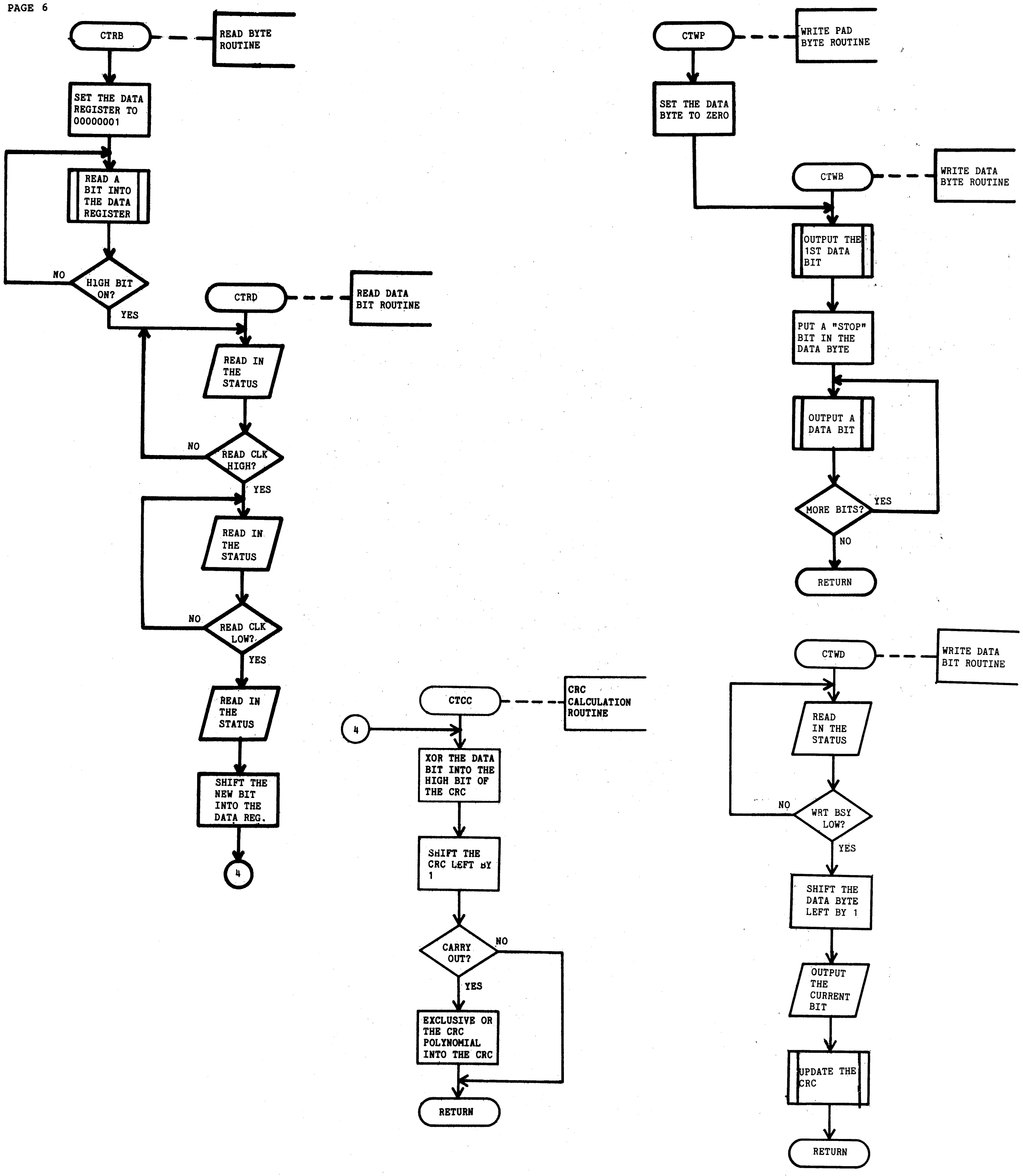
I wonder if one of the new 4K RAM's could be used for a 64 X 64 image sensor?


CTRR
CASSETTE TAPE READ RECORD ROUTINE
TURN ON THE MOTOR
ZERO THE COMPARE REGISTER
READ A BIT INTO THE COMPARE REGISTER
DATA ID?
NO
YES
ZERO THE CRC
READ IN THE RECORD LENGTH
REC LEN = 0?
NO
1
YES
2
1
READ IN A DATA BYTE
STORE THE BYTE IN THE NEXT BUFFER POSITION
INCR. THE BUFFER PTR. & DECR. THE RECORD LEN.
REC LEN = 0?
NO
YES
2
READ IN THE CRC
TURN OFF THE MOTOR
SET THE RETURN CODE FROM THE FINAL CRC
RETURN


CTIPL
CASSETTE TAPE IPL ROUTINE
(HL) <- INITIAL LOAD ADDRESS
READ IN A RECORD FROM UNIT ZERO
I/O ERROR?
YES
ERROR HALT
NO
EOF?
NO
YES
EXIT


CTWR
CASSETTE TAPE WRITE RECORD ROUTINE
OUTPUT THE UNIT SELECT
READ IN THE STATUS
MOTOR BUSY?
YES
NO
TURN ON THE MOTOR
READ IN THE STATUS
MOTOR BUSY?
YES
NO
OUTPUT THE LEADING PAD
OUTPUT THE DATA ID
ZERO THE CRC
SET THE LOOP COUNT FROM THE RECORD LENGTH + 1
SET THE CURRENT BYTE FROM THE RECORD LEN.
3
3
OUTPUT THE CURRENT BYTE
DECREMENT THE LOOP COUNT
LOOP CNT=0?
YES
NO
SET THE BYTE FROM MEMORY
INCREMENT THE BUFFER POINTER
OUTPUT THE CRC
OUTPUT THE TRAILING PAD
TURN OFF THE MOTOR
RETURN

PAGE 6


CTRB
READ BYTE ROUTINE
SET THE DATA REGISTER TO 00000001
READ A BIT INTO THE DATA REGISTER
HIGH BIT ON?
NO
YES
CTRD
READ DATA BIT ROUTINE
READ IN THE STATUS
READ CLK HIGH?
NO
YES
READ IN THE STATUS
READ CLK LOW?
NO
YES
READ IN THE STATUS
SHIFT THE NEW BIT INTO THE DATA REG.
4
CTCC
CRC CALCULATION ROUTINE
4
XOR THE DATA BIT INTO THE HIGH BIT OF THE CRC
SHIFT THE CRC LEFT BY 1
CARRY OUT?
NO
YES
EXCLUSIVE OR THE CRC POLYNOMIAL INTO THE CRC
RETURN
CTWP
WRITE PAD BYTE ROUTINE
SET THE DATA BYTE TO ZERO
CTWB
WRITE DATA BYTE ROUTINE
OUTPUT THE 1ST DATA BIT
PUT A "STOP" BIT IN THE DATA BYTE
OUTPUT A DATA BIT
MORE BITS?
YES
NO
RETURN
CTWD
WRITE DATA BIT ROUTINE
READ IN THE STATUS
WRT BSY LOW?
NO
YES
SHIFT THE DATA BYTE LEFT BY 1
OUTPUT THE CURRENT BIT
UPDATE THE CRC
RETURN

```
LOCN    CODE        SOURCE STATEMENT

                    *
                    *      TCH STANDARD CASSETTE TAPE ROM FOR THE 8008
                    *      ---
                    *
                    *      PROGRAM BY RICHARD M. SMITH
                    *


                    *      I/O DEVICE ADDRESSES

000016              CTCTL  EQU 016B          CASSETTE TAPE CONTROL REGISTER
000001              CTSTS  EQU 001B          CASSETTE TAPE STATUS REGISTER


                    *      CONTROL REGISTER BITS

000020              CTUS   EQU 020B          UNIT SELECT
000010              CTMTC  EQU 010B          MOTOR CONTROL
000004              CTWRM  EQU 004B          WRITE MODE
000002              CTWRE  EQU 002B          WRITE ENABLE
000001              CTWRD  EQU 001B          WRITE DATA


                    *      STATUS REGISTER BITS

000010              CTMTB  EQU 010B          MOTOR OPERATION BUSY
000004              CTWRB  EQU 004B          WRITE BUSY
000002              CTRDC  EQU 002B          READ CLOCK
000001              CTRDD  EQU 001B          READ DATA


                    *      UNIT NUMBERS

000000              CTUN0  EQU 000B          UNIT 0
000020              CTUN1  EQU CTUS          UNIT 1


                    *      MISCELLANEOUS EQUATES

100005              CTCRC  EQU 100005B       CRC POLYNOMIAL (CRC 16)
104657              CTDID  EQU 104657B       DATA ID
000000              CTLDA  EQU 000000B       LOAD ADDRESS FOR IPL


                    *      ORIGIN SET

037400                     ORG 037400B       ORG AT THE LAST PAGE OF MEMORY.

                    *      JUMP VECTOR

037400  104 035 077        JMP CTRR          READ RECORD ROUTINE
037403  104 213 077        JMP CTWR          WRITE RECORD ROUTINE


                    *      CASSETTE TAPE CONTROL ROUTINE

037406  250         CTCN   XRA               RESET THE CASSETTE TAPE INTERFACE.
037407  135                OUT CTCTL
037410  007                RET               RETURN.


                    *      CASSETTE TAPE IPL ROUTINE

037411  056 000     CTIPL  LHI H(CTLDA)      SET THE LOAD ADDRESS IN HL.
037413  066 000            LLI L(CTLDA)
037415  006 000     CTIP1  LAI CTUN0         READ A RECORD FROM CASSETTE TAPE
037417  106 035 077        CAL CTRR             UNIT 0.
037422  110 022 077        JFZ *             LOOP FOREVER IF AN I/O ERROR.
037425  301                LAB               LOOP IF THE RECORD IS NOT AN END OF
037426  260                ORA                  FILE RECORD.
037427  110 015 077        JFZ CTIP1
037432  104 000 000        JMP CTLDA         BRANCH TO THE LOADED PROGRAM.


                    *      CASSETTE TAPE READ RECORD ROUTINE

037435  064 010     CTRR   ORI CTMTC         TURN ON THE MOTOR ON THE SPECIFIED
037437  135                OUT CTCTL            UNIT.
037440  016 000            LBI 0             ZERO THE COMPARE REGISTER IN BC.
037442  321                LCB
037443  302         CTRR1  LAC               READ THE NEXT BIT INTO THE COMPARE
037444  200                ADA                  REGISTER
037445  301                LAB
037446  022                RAL
037447  310                LBA
037450  106 145 077        CAL CTRD
037453  301                LAB               LOOP IF NOT THE DATA ID. (ALSO
037454  054 211            XRI H(CTDID)         ZERO THE CRC IN DE IF THE DATA
037456  330                LDA                  ID.)
037457  340                LEA
037460  302                LAC
037461  054 257            XRI L(CTDID)
037463  263                ORD
037464  110 043 077        JFZ CTRR1
037467  106 133 077        CAL CTRB          READ IN THE RECORD LENGTH AND SAVE
037472  312                LBC                  IT IN B.
037473  301                LAB               LOAD IT INTO A AND BRANCH IF IT IS
037474  260                ORA                  ZERO.
037475  150 117 077        JTZ CTRR3
037500  370         CTRR2  LMA               SAVE THE RECORD LENGTH IN MEMORY.
```

```
037501  106 133 077        CAL CTRB          READ IN A DATA BYTE AND SWAP IT
037504  307                LAM                  WITH THE RECORD LENGTH IN
037505  372                LMC                  MEMORY.
037506  060                INL               BUMP UP HL BY 1.
037507  110 113 077        JFZ *+4
037512  050                INH
037513  011                DCB               DECREMENT B AND LOOP IF IT IS NOT
037514  110 100 077        JFZ CTRR2            ZERO.
037517  310         CTRR3  LBA               SAVE THE RECORD LENGTH IN B.
037520  106 133 077        CAL CTRB          READ IN THE CRC.
037523  106 133 077        CAL CTRB


                    *      MOTOR OFF ROUTINE

037526  250         CTMOF  XRA               RESET THE CASSETTE TAPE INTERFACE.
037527  135                OUT CTCTL
037530  303                LAD               SET THE RETURN CODE FROM THE CRC.
037531  264                ORE
037532  007                RET               RETURN.


                    *      READ BYTE ROUTINE

037533  026 001     CTRB   LCI 001B          SET C TO 001B.
037535  106 145 077 CTRB1  CAL CTRD          READ IN 7 DATA BITS.
037540  302                LAC
037541  260                ORA
037542  120 135 077        JFS CTRB1


                    *      READ DATA BIT ROUTINE

037545  103         CTRD   INP CTSTS         WAIT FOR THE READ CLOCK TO GO HIGH.
037546  044 002            NDI CTRDC
037550  110 145 077        JFZ CTRD
037553  103         CTRD1  INP CTSTS         WAIT FOR THE READ CLOCK TO GO LOW.
037554  044 002            NDI CTRDC
037556  150 153 077        JTZ CTRD1
037561  103                INP CTSTS         ISOLATE THE DATA BIT IN THE CARRY.
037562  032                RAR
037563  302                LAC               ROTATE THE BIT INTO C.
037564  022                RAL
037565  320                LCA
037566  012                RRC               PUT THE BIT IN THE HIGH BIT OF A.


                    *      CRC CALCULATION ROUTINE

037567  044 200     CTCC   NDI 200B          EXCLUSIVE OR THE DATA BIT INTO THE
037571  253                XRD                  HIGH ORDER BIT OF THE CRC.
037572  330                LDA
037573  304                LAE               SHIFT THE CRC LEFT BY 1.
037574  200                ADA
037575  340                LEA
037576  303                LAD
037577  022                RAL
037600  330                LDA
037601  003                RFC               RETURN IF NO CARRY OUT.
037602  303                LAD               EXCLUSIVE OR THE CRC POLYNOMIAL INTO
037603  054 200            XRI H(CTCRC)         THE CRC.
037605  330                LDA
037606  304                LAE
037607  054 005            XRI L(CTCRC)
037611  340                LEA
037612  007                RET               RETURN.


                    *      WRITE RECORD ROUTINE

037613  135         CTWR   OUT CTCTL         OUTPUT THE UNIT SELECT AND SAVE IT
037614  320                LCA                  IN C.
037615  103         CTWR1  INP CTSTS         WAIT FOR THE MOTOR TO STOP.
037616  044 010            NDI CTMTB
037620  110 215 077        JFZ CTWR1
037623  302                LAC               TURN ON THE MOTOR.
037624  064 010            ORI CTMTC
037626  135                OUT CTCTL
037627  103         CTWR2  INP CTSTS         WAIT FOR THE MOTOR TO COME UP TO
037630  044 010            NDI CTMTB            SPEED.
037632  110 227 077        JFZ CTWR2
037635  106 332 077        CAL CTWP          OUTPUT THE 4 BYTES OF LEADING PAD.
037640  106 332 077        CAL CTWP
037643  106 332 077        CAL CTWP
037646  106 332 077        CAL CTWP
037651  026 211            LCI H(CTDID)      OUTPUT THE DATA ID.
037653  106 334 077        CAL CTWB
037656  026 257            LCI L(CTDID)
037660  106 334 077        CAL CTWB
037663  036 000            LDI 0             ZERO THE CRC IN DE.
037665  343                LED
037666  321                LCB               LOAD THE RECORD LENGTH INTO C, BUMP
037667  010                INB                  IT UP BY 1 IN B, AND BRANCH INTO
037670  104 301 077        JMP CTWR4            THE WRITE LOOP.
037673  327         CTWR3  LCM               LOAD THE NEXT DATA BYTE INTO C.
037674  060                INL
037675  110 301 077        JFZ *+4
037700  050                INH
037701  106 334 077 CTWR4  CAL CTWB          OUTPUT THE CURRENT BYTE.
037704  011                DCB               DECREMENT B AND LOOP IF IT IS NOT
037705  110 273 077        JFZ CTWR3            ZERO.
037710  314                LBE               OUTPUT THE CRC.
```

```
037711  323                LCD
037712  106 334 077        CAL CTWB
037715  321                LCB
037716  106 334 077        CAL CTWB
037721  106 332 077        CAL CTWP          OUTPUT THE TRAILING PAD OF ZEROS.
037724  106 332 077        CAL CTWP
037727  104 126 077        JMP CTMOF         GO TURN OFF THE MOTOR.


                   *       WRITE ZERO PAD BYTE ROUTINE

037732  026 000    CTWP    LCI 0             ZERO C.


                   *       WRITE BYTE ROUTINE

037734  106 351 077 CTWB   CAL CTWD          OUTPUT THE 1ST DATA BIT.
037737  020                INC               PUT IN A "STOP" BIT IN C.
037740  106 351 077 CTWB1  CAL CTWD          OUTPUT BITS UNTIL A STOP BIT IS THE
037743  302                LAC                  ONLY BIT LEFT.
037744  200                ADA
037745  110 340 077        JFZ CTWB1
037750  007                RET               RETURN.
```

```
                   *       WRITE DATA BIT ROUTINE

037751  103        CTWD    INP CTSTS         WAIT FOR THE WRITE BUSY TO GO LOW.
037752  044 004            NDI CTWRB
037754  110 351 077        JFZ CTWD
037757  302                LAC               SHIFT THE BYTE IN C LEFT BY 1.
037760  200                ADA
037761  320                LCA
037762  103                INP CTSTS         ISOLATE THE UNIT SELECT AND CURRENT
037763  032                RAR                  DATA BIT IN A.
037764  002                RLC
037765  044 021            NDI CTUS+CTWRD
037767  064 016            ORI CTWRM+CTWRE+CTMTC OUTPUT THE BIT.
037771  135                OUT CTCTL
037772  012                RRC               ISOLATE THE DATA BIT IN THE HIGH BI
                   *                            OF A.
037773  104 167 077        JMP CTCC          GO UPDATE THE CRC.

037776                     END
```

```
                 ;        TCH STANDARD CASSETTE TAPE ROM FOR THE 8080
                 ;        ---
                 ;
                 ;        PROGRAM BY RICHARD M. SMITH
                 ;


                 ;        I/O DEVICE ADDRESSES

000200           CTCTL    EQU  200Q           ; CASSETTE TAPE CONTROL REGISTER
000200           CTSTS    EQU  200Q           ; CASSETTE TAPE STATUS REGISTER


                 ;        CONTROL REGISTER BITS

000020           CTUS     EQU  020Q           ; UNIT SELECT
000010           CTMTC    EQU  010Q           ; MOTOR CONTROL
000004           CTWRM    EQU  004Q           ; WRITE MODE
000002           CTWRE    EQU  002Q           ; WRITE ENABLE
000001           CTWRD    EQU  001Q           ; WRITE DATA


                 ;        STATUS REGISTER BITS

000010           CTMTB    EQU  010Q           ; MOTOR BUSY
000004           CTWRB    EQU  004Q           ; WRITE BUSY
000002           CTRDC    EQU  002Q           ; READ CLOCK
000001           CTRDD    EQU  001Q           ; READ DATA


                 ;        UNIT NUMBERS

000000           CTUNO    EQU  000Q           ; UNIT 0
000020           CTUN1    EQU  CTUS           ; UNIT 1


                 ;        MISCELLANEOUS EQUATES

000200           CTHCRC   EQU  200Q           ; CRC POLYNOMIAL (CRC 16)
000005           CTLCRC   EQU  005Q           ;
000211           CTHDID   EQU  211Q           ; DATA ID
000257           CTLDID   EQU  257Q           ;
000000           CTLDA    EQU  000000Q        ; INITIAL LOAD ADDRESS FOR IPL
000400           CTSTE    EQU  000400Q        ; END OF STACK FOR IPL


                 ;        ORIGIN SET

177400                    ORG  177400Q        ; SET THE ORIGIN TO THE LAST
                                              ;    PAGE OF MEMORY.


                 ;        JUMP VECTOR

177400  303 042 377       JMP  CTRR           ; READ RECORD
177403  303 135 377       JMP  CTWR           ; WRITE RECORD
177406  303 036 377       JMP  CTCN           ; CONTROL ROUTINE


                 ;        CASSETTE TAPE IPL ROUTINE

177411  041 000 000 CTIPL:  LXI  HL,CTLDA     ; INITIALIZE THE LOAD ADDRESS
177414  061 000 001         LXI  SP,CTSTE     ;    AND THE STACK POINTER.
177417  345                 PUSH HL           ; SAVE THE LOAD ADDRESS.
177420  076 000     CTIP1:  MVI  A,CTUNO      ; READ A RECORD FROM CASSETTE
177422  315 042 377         CALL CTRR         ;    TAPE UNIT 0.
177425  302 025 377         JNZ  $            ; LOOP FOREVER IF I/O ERROR.
177430  170                 MOV  A,B          ; LOOP IF THE RECORD IS NOT AN
177431  267                 ORA  A            ;    END OF FILE RECORD.
177432  302 020 377         JNZ  CTIP1        ;
177435  311                 RET               ; BRANCH TO THE LOADED PROGRAM.
```

```
                    ;        CASSETTE TAPE CONTROL ROUTINE

177436  257         CTCN:    XRA  A              ; CLEAR THE CASSETTE TAPE
177437  323 200              OUT  CTCTL          ;    INTERFACE.
177441  311                  RET                 ; RETURN.


                    ;        CASSETTE TAPE READ RECORD ROUTINE

177442  366 010     CTRR:    ORI  CTMTC          ; TURN THE SPECIFIED UNIT´S
177444  323 200              OUT  CTCTL          ;    MOTOR ON.
177446  001 000 000          LXI  BC,0           ; ZERO THE COMPARE REGISTER IN
                                                 ;    BC.
177451  171         CTRR1:   MOV  A,C            ; READ THE NEXT DATA BIT INTO
177452  207                  ADD  A              ;    THE COMPARE REGISTER.
177453  170                  MOV  A,B            ;
177454  027                  RAL                 ;
177455  107                  MOV  B,A            ;
177456  315 255 377          CALL CTRD           ;
177461  170                  MOV  A,B            ; LOOP IF THE COMPARE REGISTER
177462  356 211              XRI  CTHDID         ;    DOES NOT EQUAL THE DATA
177464  127                  MOV  D,A            ;    ID. (ALSO ZERO THE CRC IN
177465  137                  MOV  E,A            ;    IN DE IF THE DATA ID.)
177466  171                  MOV  A,C            ;
177467  356 257              XRI  CTLDID         ;
177471  262                  ORA  D              ;
177472  302 051 377          JNZ  CTRR1          ;
177475  315 243 377          CALL CTRB           ; READ IN THE RECORD LENGTH AND
177500  101                  MOV  B,C            ;    SAVE IT IN B AND ON THE
177501  305                  PUSH B              ;    STACK.
177502  171                  MOV  A,C            ; BRANCH IF IT IS ZERO.
177503  267                  ORA  A              ;
177504  312 120 377          JZ   CTRR3          ;
177507  315 243 377 CTRR2:   CALL CTRB           ; READ AND STORE THE NEXT DATA
177512  167                  MOV  M,A            ;    BYTE.
177513  043                  INX  HL             ;
177514  005                  DCR  B              ; DECREMENT B AND LOOP IF IT
177515  302 042 377          JNZ  CTRR2          ;    IS NOT ZERO.
177520  315 243 377 CTRR3:   CALL CTRB           ; READ IN THE 2 CRC BYTES.
177523  315 243 377          CALL CTRB           ;
177526  301                  POP  BC             ; POP THE RECORD LENGTH INTO
                                                 ;    B.


                    ;        MOTOR OFF ROUTINE

177527  257         CTMOF:   XRA  A              ; RESET THE CASSETTE TAPE
177530  323 200              OUT  CTCTL          ;    INTERFACE.
177532  172                  MOV  A,D            ; SET THE RETURN CODE FROM THE
177533  263                  ORA  E              ;    FINAL CRC.
177534  311                  RET                 ; RETURN.


                    ;        CASSETTE TAPE WRITE RECORD ROUTINE

177535  117         CTWR:    MOV  C,A            ; OUTPUT THE UNIT SELECT AND
177536  323 200              OUT  CTCTL          ;    SAVE IT IN C.
177540  333 200     CTWR1:   IN   CTSTS          ; WAIT FOR THE MOTOR TO STOP.
177542  346 010              ANI  CTMTB          ;
177544  302 140 377          JNZ  CTWR1          ;
177547  171                  MOV  A,C            ; TURN ON THE MOTOR.
177550  366 010              ORI  CTMTC          ;
177552  323 200              OUT  CTCTL          ;
177554  333 200     CTWR2:   IN   CTSTS          ; WAIT FOR THE MOTOR TO COME
177556  346 010              ANI  CTMTB          ;    UP TO SPEED.
177560  302 154 377          JNZ  CTWR2          ;
177563  315 323 377          CALL CTWP2          ; OUTPUT THE 4 BYTES OF
177566  315 323 377          CALL CTWP2          ;    LEADING PAD.
177571  016 211              MVI  C,CTHDID       ; OUTPUT THE DATA ID.
177573  315 330 377          CALL CTWB           ;
177576  016 257              MVI  C,CTLDID       ;
177600  315 330 377          CALL CTWB           ;
177603  021 000 000          LXI  DE,0           ; ZERO THE CRC IN DE.
```

```
177606  110                   MOV  C,B              ; LOAD THE RECORD LENGTH INTO
177607  004                   INR  B                ;    C, BUMP IT UP BY 1 IN B,
177610  303 215 377           JMP  CTWR4            ;    AND BRANCH INTO THE WRITE
                                                    ;    LOOP.
177613  116         CTWR3:    MOV  C,M              ; LOAD THE NEXT DATA BYTE INTO
177614  043                   INX  HL               ;    C.
177615  315 330 377 CTWR4:    CALL CTWB             ; OUTPUT THE CURRENT BYTE.
177620  005                   DCR  B                ; DECREMENT B AND LOOP IF IT
177621  302 213 377           JNZ  CTWR3            ;    IS NOT ZERO.
177624  103                   MOV  B,E              ; OUTPUT THE CRC.
177625  112                   MOV  C,D              ;
177626  315 330 377           CALL CTWB             ;
177631  110                   MOV  C,B              ;
177632  315 330 377           CALL CTWB             ;
177635  315 323 377           CALL CTWP2            ; OUTPUT THE TRAILING PAD.
177640  303 127 377           JMP  CTMOF            ; GO TURN OFF THE MOTOR.


                              READ BYTE ROUTINE

177643  016 001     CTRB:     MVI  C,001Q           ; SET C TO 001Q.
177645  315 255 377 CTRB1:    CALL CTRD             ; READ IN 7 DATA BITS.
177650  171                   MOV  A,C              ;
177651  267                   ORA  A                ;
177652  362 245 377           JP   CTRB1            ;


                    ;         READ DATA BIT ROUTINE

177655  333 200     CTRD:     IN   CTSTS            ; WAIT FOR THE READ CLOCK TO
177657  346 002               ANI  CTRDC            ;    GO HIGH.
177661  302 255 377           JNZ  CTRD             ;
177664  333 200     CTRD1:    IN   CTSTS            ; WAIT FOR THE READ CLOCK TO
177666  346 002               ANI  CTRDC            ;    GO LOW.
177670  312 264 377           JZ   CTRD1            ;
177673  333 200               IN   CTSTS            ; ISOLATE THE DATA BIT IN THE
177675  037                   RAR                   ;    CARRY.
177676  171                   MOV  A,C              ; SHIFT THE BIT INTO C.
177677  217                   ADC  A                ;
177700  117                   MOV  C,A              ;
177701  017                   RRC                   ; PUT THE NEW BIT IN THE HIGH
                                                    ;    BIT OF A FOR THE CRC
                                                    ;    UPDATE.


                    ;         CRC CALCULATION ROUTINE

177702  346 200     CTCC:     ANI  200Q             ; EXCLUSIVE OR THE DATA BIT
177704  252                   XRA  D                ;    INTO THE HIGH ORDER BIT
177705  127                   MOV  D,A              ;    OF THE CRC.
177706  353                   XCHG                  ; SHIFT THE CRC LEFT BY 1.
177707  051                   DAD  HL               ;
177710  353                   XCHG                  ;
177711  320                   RNC                   ; RETURN IF NO CARRY OUT.
177712  172                   MOV  A,D              ; EXCLUSIVE OR THE CRC
177713  356 200               XRI  CTHCRC           ;    POLYNOMIAL INTO THE CRC.
177715  127                   MOV  D,A              ;
177716  173                   MOV  A,E              ;
177717  356 005               XRI  CTLCRC           ;
177721  137                   MOV  E,A              ;
177722  311                   RET                   ; RETURN.


                    ;         WRITE ZERO PAD BYTE ROUTINE

177723  315 326 377 CTWP2:    CALL CTWP             ; OUTPUT THE 1ST PAD BYTE.
177726  016 000     CTWP:     MVI  C,0              ; ZERO C.


                    ;         WRITE BYTE ROUTINE

177730  315 345 377 CTWB:     CALL CTWD             ; OUTPUT THE 1ST DATA BIT.
177733  014                   INR  C                ; PUT A "STOP" BIT IN C.
177734  315 345 377 CTWB1:    CALL CTWD             ; OUTPUT BITS UNTIL THE STOP
177737  171                   MOV  A,C              ;   BIT IS THE ONLY BIT
177740  207                   ADD  A                ;   LEFT.
177741  302 334 377           JNZ  CTWB1            ;
177744  311                   RET                   ; RETURN.


                    ;         WRITE DATA BIT ROUTINE

177745  333 200     CTWD:     IN   CTSTS            ; WAIT FOR THE WRITE BUSY TO
177747  346 004               ANI  CTWRB            ;   GO LOW.
177751  302 345 377           JNZ  CTWD             ;
177754  171                   MOV  A,C              ; SHIFT THE BYTE IN C LEFT BY
177755  207                   ADD  A                ;    1.
177756  117                   MOV  C,A              ;
177757  333 200               IN   CTSTS            ; ISOLATE THE UNIT SELECT AND
177761  037                   RAR                   ;
177762  007                   RLC                   ;
177763  346 021               ANI  CTUS+CTWRD       ;
177765  366 016               ORI  CTWRM+CTWRE+CTMTC ; OUTPUT THE BIT.
177767  323 200               OUT  CTCTL            ;
177771  017                   RRC                   ; POSITION THE CURRENT DATA
177772  303 302 377           JMP  CTCC             ;    BIT IN THE HIGH BIT OF A
                                                    ;    AND GO UPDATE THE CRC.


                              END                   ;
```

COMPUTER PING-PONG by Jim Parker

One of the more entertaining ways to use a graphics display is to play ping-pong on it. This program simulates a Ping-Pong game in a way very similar to the television version that has recently become popular. The game allows two players to move "paddles" up and down so as to "hit" a ball back and forth across the screen. The first serve is begun by pushing a button. A miss scores a point for the other player and automatically causes another serve to the player who missed last. The first player to score eleven points wins the game. The first game always begins with a serve to the player on the right. After that, the following games begin with a serve to the side that lost the last game.

The program runs on an 8008-1 system with at least 3 pages of RAM and Hal Chamberlin's graphics display and pot controls. The plans for these I/O devices were published in Volume I, issue numbers 1, 2, 3, and 4 of TCH. As an option, a speaker can be used to provide a pop sound when the ball and paddle collide. Readers with an 8080 CPU should also be able to use this program although it may need to be slowed down a little. More will be mentioned on this later.

Motion of the ball is done by showing successive "frames" of the ball shifted slightly over, much like a movie. An interesting feature of this game is that the ball can travel with 32 different velocities, 16 towards the left and 16 towards the right. A basic knowledge of vectors is required to understand how this is done. Simply put, the velocity (which specifies both speed and direction) can be broken up into X and Y (horizontal and vertical) components. The square root of (X**2+Y**2) determines the speed while the arc tangent of (Y/X) determines the direction angle with respect to the horizontal axis. For example, if the ball was displaced 1 unit up and one unit to the right every frame, it would appear to move at a speed of v2 units per frame at an angle of 45o. A problem arises when you try to keep the speed below 2 units per frame (as was done in this program) and still provide a large variety of direction angles. If you use only integers, the possible (X,Y) component velocity combinations are soon exhausted. For example, if you displace the ball 1 unit right and 2 units up every frame, the speed will be v5 or greater than 2 units per frame. To solve this problem, double precision (16 bit) arithmetic was used. The most significant 8 bits specify the ball's X or Y position on the display while the least significant 8 bits act as the numerator of a fraction with the implied denominator of 256. This two byte combination can be treated as if the whole thinq was multiplied by 256 and converted to a 16 bit integer with an implied division by 256 to restore it back to normal. Thus the DBLADD routine can perform a simple 16 bit integer addition to compute the next ball position. Note that multiplication or division by 256 is accomplished by shifting the 16 bit integer left or right 8 bits with respect to the binary point. Thus the 16 bit ball position is displayed by regarding only the most significant byte. This has the same effect as dividing the 16 bit integer by 256 (shifting it right with respect to the binary point) and ignoring any fractional part of the quotient.

The use of fractions allow a much greater range of velocity (X,Y) components between 0 and 2 so that a wide range of angles are possible all with a speed of 2 units per frame. As mentioned before, the ball's next position is computed by adding the 16 bit X position to the 16 bit X displacement, the 16 bit Y position to the 16 bit Y displacement, and regarding only the most significant bytes of these two sums. Due to the fact that the fractional part of the ball's position is not displayed, there is a slightly ragged appearance in the ball's path across the screen although it is certainly not very noticable. To avoid complex subroutines to compute square roots and arc tangents, a simple velocity table is provided with 16 different velocities. Fifteen of these move the ball at a speed of 2 units per frame and one moves the ball horizontally at 1 unit per frame. All of the velocities move the ball towards the right. To move the ball left, the X velocity component is negated. Note that two's complement arithmetic is used to handle negative numbers. I like to negate numbers simply by subtracting them from zero which is represented the same as 2**16 or 65536 for 16 bit numbers. Thus negative 100 would be represented as 65536-100=65436. To prove that this works, try adding -100 to +300 using 16 bit unsigned integers. Well, 65436+300= 65736 but that number requires 17 bits to represent. Since the most significant bit will be lost and that bit represents 2**16=65536, we must subtract it from the above addition. Thus 65736-65536=200 which is the correct result. If you are familiar with modular arithmetic, you can think of this as MOD 2**16 arithmetic. There are other ways to deal with two's complement arithmetic but I prefer the above method.

In the velocity table, all values were computed with the help of a pocket calculator. They are listed in the order of least significant X, most significnat X, least significant Y, and most significant Y. Multiply the most significant value by 256 and add that to the least significant value to get the numerator of a fraction in the range of 0/256 to 65535/256. If the number is greater than or equal to 2**15=32768, its most significant bit (called the sign bit) will be ONE and the number will be negative. Convert these to a positive number by subtracting them from 65536 as mentioned before. You can then confirm for yourself the ball velocities by dividing the numbers by 256 and plugging them into the square root and arc tangent formulas given previously.

When the ball is served or when the ball is hit, a random number generator indexes the ANGLE table so as to project the ball in an unpredictable manner. Bouncing the ball off the top or bottom, however, is simulated by simply negating the ball's Y velocity component. Basic physics predicts this result assuming there is no energy lost in the bounce.

The power of flow charting is well illustrated here for without the appending chart, it would be very difficult to explain or even write this program. By glancing over it, you should get a general idea of how the program runs. Please refer to the Volume 1, Number 4 issue of TCH for a detailed description of the pot controls and switch device used. Before switch 1 is pushed, the program merely draws the score from the last game, reads two pot controls and positions the two paddles accordingly, and draws the board (a square that defines the outer boundary). All this is done by calling the DRAW subroutine. If you are unfamiliar with how the CHAR, RVCD, and GRAPH subroutines work, you should refer to the example software programs listed in the first four issues of TCH. Basically the DRAW subroutine does its job by filling in the correct parameters for the other three subroutines just mentioned.

When switch 1 is pushed, the program begins a new game. It resets the score, picks a starting position for the serve somewhere near the middle of the board, and picks a serve velocity at random from the ANGLE table. The program is then ready to enter the major loop headed by the name NXPOS. The loop is passed through everytime the next position of the ball is displayed and examined. There are four tests made on the ball's position. Due to the wrap-around effect of the board the test for end zone can be reduced to one test. Similarly, so can the test for the ball touching the top or botton. It works like this:

There are two Y coordinate positions at the top and two at the bottom that indicate when the ball has reached the edge. Thus if the ball's most significant Y position is 126, 127, 128 (=-128), or 129 (=-127), then the program needs to change the ball's Y direction. This is tested by subtracting 126 from the ball's Y position. If the ball is at the edge, the resulting difference will be 0,1,2, or 3. A CPI 4 is performed on the result. This sets the flag flip-flop the same way as if 4 had been subtracted from the number. Only if the number is 0,1,2, or 3 will the carry flag (or the borrow flag in this case) be set by subtracting 4. A similar technique is used for testing the other zones. Some readers may find this technique useful in other programs for testing if a number is in a certain range. Note that the X position of the ball is not reloaded after every test. This can be done if you keep track of the amount you subtracted from the number for the previous test. Thus 104 is subtracted first to see if the ball's X position is 104 and then 44 is subtracted from that to see if the X position is 148 (=-108) or 149 (=-107).

The result of the above tests determine what the program does next. If the ball is not touching any special boundaries then the program jumps back to NXPOS and displays the next position of the ball. As long as the ball doesn't reach the paddle zones or end zone, the program will continue to loop through NXPOS, showing the ball shifted over slightly each time. This causes the ball to appear in motion. When the ball reaches a paddle zone, the program jumps to a routine that tests the position of the corresponding paddle. The paddle is 16 units high so a test is made to see if the ball's Y position minus the Y position of the bottom of the paddle is less than 16. If it is, the direction of the ball is changed and a new ball velocity is picked at random from the ANGLE table. The ball must also be cleared from the paddle zone or an interresting bug will occur. If the ball travels into the zone faster than it leaves, it will not be out of the zone on the next time around the loop. The program would the change the direction of the ball again. The ball may bounce back and forth in this zone many times until it finally escaped. To prevent this from happening, the program actually relocates the ball just outside of the paddle zone when it is hit. Also a speaker is popped. This is done by executing an input instruction to the keyboard (which has the speaker for feedback while typing) on TCH's demo system. If the ball misses the paddle then the program allows the ball to keep traveling in the same direction.

When someone misses the ball, the ball will reach the end zone. This causes the program to test the ball's direction to see who missed, and then to increment the score accordingly. Score keeping is handled somewhat unconventionally in this program in order to save on memory and program complexity. Each character can be drawn using the minor deflection system with 16 bytes or less of data. Because of this, the data for the characters begin at memory locations separated by exactly 16 bytes per character. Since the table of characters for the score is arranged in ascending order, the score counter needs only to keep track of the low address of the character and have that value incremented by 16 every time the player scores another point. Note that ten and eleven are treated as if they were single characters. The program knows when the game has ended by testing for when the score counter addresses the character eleven. If the game is not over, the program will delay a few seconds and then jump back to the part of the program that initiates a serve. If the game is over, the program will jump back to the beginning, displaying the final score and waiting for switch 1 to be pushed to begin another game.

Readers with an 8080 machine and an assembler should have little problem using this program. Since the 8008 instruction set is a subset of the 8080 instruction set, you should be able to translate every instruction on a one to one basis. The real problem in running the program is slowing it down. If you are driving the graphics display as published in the first three issues of TCH, your main concern should be in slowing down the subroutines that draw on the display since most of the execution time is spent driving the display. This can be done by adding inefficient NOOP´s (such as an even number of XTHL´s) in the critical loops. If the speed of the 8080 can match the speed of Hal Chamberlin´s display, then the program should be slow enough for two human players. Caution should also be used in the D/A routines for the pot controls to avoid sampling the comparators before they have settled. If you have both a fast machine and a faster graphics display, then you can either introduce delay loops, reduce the size of the entries in the ball velocity table, or make the paddles wider.

If you have a slower machine such as an 8008 instead of an 8008-1, you can improve the efficency of the display subroutines, make the paddles smaller, or increase the size of the entries in the ball velocity table. If you choose the last suggestion, you must also increase the width of the paddle zones or the ball may penetrate right through the paddles.

As usual, there are a large number of variations that can be tried. A simple variation is to reduce the thickness of various boundaries so as to make them penetratable 50% of the time. This is guaranteed to challenge even the most skilled players. To make even a more realistic game, the velocity of the ball after being hit could be determined by how quickly the paddle changed position prior to being hit. Even fancier versions would be a program that would occasionally make the ball loop-the-loop. A game for 1, 3, or 4 players could also be developed. In the case of a single player, provide a goal on the opposing side so that the player could hit the ball in. Hopefully this program will open the doorway to a whole series of simulation game programs. Hobbyists are encouraged to make every effort to build their own graphics display and enjoy the demo programs that are published by TCH.

PING-PONG PROGRAM LISTING

```
LOCN      CODE       SOURCE STATEMENT

000000                         ORG   0

000006               GINP      EQU   6
000010               XMOV      EQU   10B
000011               YMOV      EQU   11B
000012               XSTOR     EQU   12B
000013               YDRAW     EQU   13B
000014               MINXY     EQU   14B
000015               MINSZ     EQU   15B

000000  106 032 001 PING       CAL   DRAW     DRAW SCORE, BOARD AND PADDLES
000003  115                    INP   GINP
000004  022                    RAL
000005  100 000 000            JFC   PING     WAIT FOR SWITCH 1 TO BE PUSHED
000010  056 002 066            SHL   SCORE
000013  025
000014  076 000                LMI   0
000016  060                    INL
000017  076 000                LMI   0        RESET SCORE TO 0

000021  106 361 000 SERVE      CAL   RAND     GET RANDOM BYTE IN ACC
000024  044 177                NDI   177B
000026  024 100                SUI   64       IN RANGE OF -64 TO 63
000030  066 052                LLI   L(BALPOS+3) PT TO MOST SIG. Y BALL POS
000032  370                    LMA            RANDOM VERTICAL SERVE POS. NEAR MIDDLE
000033  066 050                LLI   L(BALPOS+1)   PT TO MOST SIG. X BALL POS.
000035  076 000                LMI   0        HORIZONTAL SERVE POS. AT MIDDLE
000037  106 270 000            CAL   RDISP    PICK BALL VELOCITY FROM TABLE
000042  066 050      NXPOS     LLI   L(BALPOS+1)
000044  307                    LAM
000045  121                    OUT   XMOV
000046  066 052                LLI   L(BALPOS+3)
000050  307                    LAM
000051  123                    OUT   YMOV     POSITION BALL´S X AND Y COORD.
000052  066 066                LLI   L(BALL)
000054  106 127 001            CAL   CHAR     DRAW BALL
000057  106 032 001            CAL   DRAW     DRAW SCORE,BOARD, AND PADDLES
000062  036 027                LDI   L(XDISP)  D= ADDRESS OF XDISP
000064  046 047                LEI   L(BALPOS)  E= ADDRESS OF BALPOS
000066  106 333 000            CAL   DBLADD   COMPUTE NEW BALL X COORD
000071  030                    IND            D= ADDRESS OF YDISP
000072  040                    INE            E= ADDRESS OF BALPOS Y COORD
000073  106 333 000            CAL   DBLADD   COMPUTE NEW BALL Y COORD
000076  066 052                LLI   L(BALPOS+3)
000100  307                    LAM            GET BALL Y COORD
000101  024 176                SUI   126
000103  074 004                CPI   4        CHECK IF TOUCHES TOP OR BOTTOM
000105  066 031                LLI   L(YDISP)
000107  142 350 000            CTC   NEGATE   CHANGE Y DIRECTION IF SO
000112  066 050                LLI   L(BALPOS+1)
000114  307                    LAM            LOAD BALL X POSITION
000115  024 150                SUI   104
000117  074 002                CPI   2        SEE IF IT TOUCHES RIGHT PADDLE BOUNDARY
000121  140 212 000            JTC   RPAD     CHECK PADDLE POSITION IF SO
000124  024 054                SUI   44
000126  074 002                CPI   2        SEE IF IT TOUCHES LEFT PADDLE ZONE
000130  140 217 000            JTC   LPAD     CHECK PADDLE POSITION IF SO
000133  004 026                ADI   22
000135  074 004                CPI   4        SEE IF BALL REACHES END ZONE
000137  140 145 000            JTC   OFF      JUMP IF SO
000142  104 042 000            JMP   NXPOS    COMPUTE NEXT BALL POSITION

000145  066 033      OFF       LLI   L(DIRECT)
000147  307                    LAM            GET BALL DIRECTION
000150  260                    ORA            SET SIGN BIT
000151  160 161 000            JTS   OFFL     RECORD SCORE ACCORDINGLY
000154  066 025                LLI   L(SCORE)
```

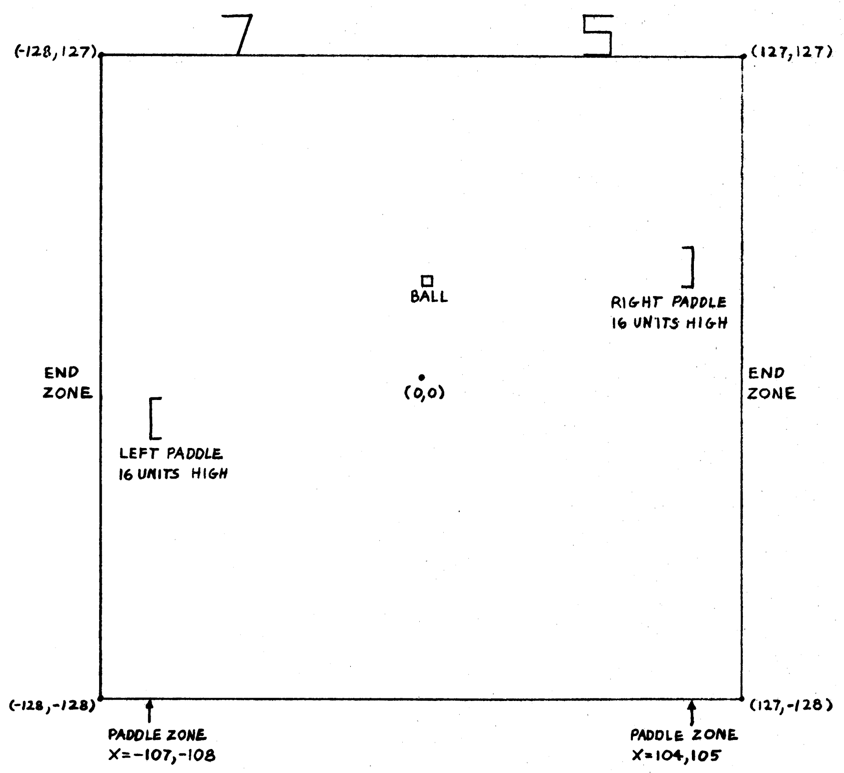
```
000156  104 163 000            JMP   POINTS
000161  066 026      OFFL      LLI   L(SCORE+1)
000163  307          POINTS    LAM
000164  004 020                ADI   20B      BUMP SCORE INDEX
000166  370                    LMA
000167  074 260                CPI   260B     CHECK FOR END OF GAME
000171  150 000 000            JTZ   PING     JUMP IF SO
000174  066 034      LOOP      LLI   L(DELAY)  DELAY A WHILE BEFORE NEXT SERVE
000176  317                    LBM
000177  011                    DCB
000200  371                    LMB
000201  150 021 000            JTZ   SERVE    SERVE NEXT BALL AFTER DELAY
000204  106 032 001            CAL   DRAW     DRAW BOARD WHILE WAITING
000207  104 174 000            JMP   LOOP     DELAY LOOP

000212  006 004      RPAD      LAI   4        SET TO READ RIGHT DIAL
000214  104 221 000            JMP   HIT      CHECK FOR PADDLE HIT

000217  006 001      LPAD      LAI   1        SET TO READ DIAL 1

000221  106 137 001 HIT        CAL   RVCD     GET DIAL VALUE
000224  056 002 066            SHL   BALPOS+3
000227  052
000230  227                    SUM            GET DISTANCE BETWEEN BALL AND BOTTOM
                     *                        OF PADDLE
000231  004 020                ADI   16       SET CARRY IF BALL TOUCHES PADDLE
000233  100 042 000            JFC   NXPOS    LET BALL PASS IF NO CONTACT
000236  107                    INP   3B       POP SPEAKER
000237  066 033                LLI   L(DIRECT)
000241  307                    LAM            GET BALL DIRECTION
000242  054 200                XRI   200B
000244  370                    LMA            INVERT BALL DIRECTION
000245  106 270 000            CAL   RDISP    GET RANDOM BALL VELOCITY
000250  066 033                LLI   L(DIRECT)
000252  307                    LAM            GET BALL DIRECTION
000253  260                    ORA            SET SIGN BIT
000254  066 050                LLI   L(BALPOS+1)   PT TO  THE BALLS X POSITION
000256  076 147                LMI   103      CLEAR FROM RIGHT PADDLE
000260  160 042 000            JTS   NXPOS    LOOP IF BOUNCED OFF RIGHT PADDLE
000263  076 226                LMI   150      CLEAR FROM LEFT PADDLE
000265  104 042 000            JMP   NXPOS    LOOP

                     *         PICKS RANDOM BALL VELOCITY FROM TABLE

000270  106 361 000 RDISP      CAL   RAND     GET RANDOM BYTE
000273  044 074                NDI   74B
000275  066 272                LLI   L(ANGLE)
000277  206                    ADL
000300  360                    LLA            INDEX ANGLE TABLE AT RANDOM
000301  307                    LAM            TRANSFER 4 BYTES FROM ANGLE TABLE
000302  060                    INL            TO XDISP AND Y DISP
000303  317                    LBM
000304  060                    INL
000305  327                    LCM
000306  060                    INL
000307  337                    LDM
000310  066 027                LLI   L(XDISP)
000312  370                    LMA
000313  060                    INL
000314  371                    LMB
000315  060                    INL
000316  372                    LMC
000317  060                    INL
000320  373                    LMD
000321  066 033                LLI   L(DIRECT)
000323  307                    LAM            GET BALL DIRECTION
000324  066 027                LLI   L(XDISP)
000326  260                    ORA            SET SIGN FLAG
000327  160 350 000            JTS   NEGATE   NEGATE X DIRECTION IF NEEDED
000332  007                    RET            RETURN
```

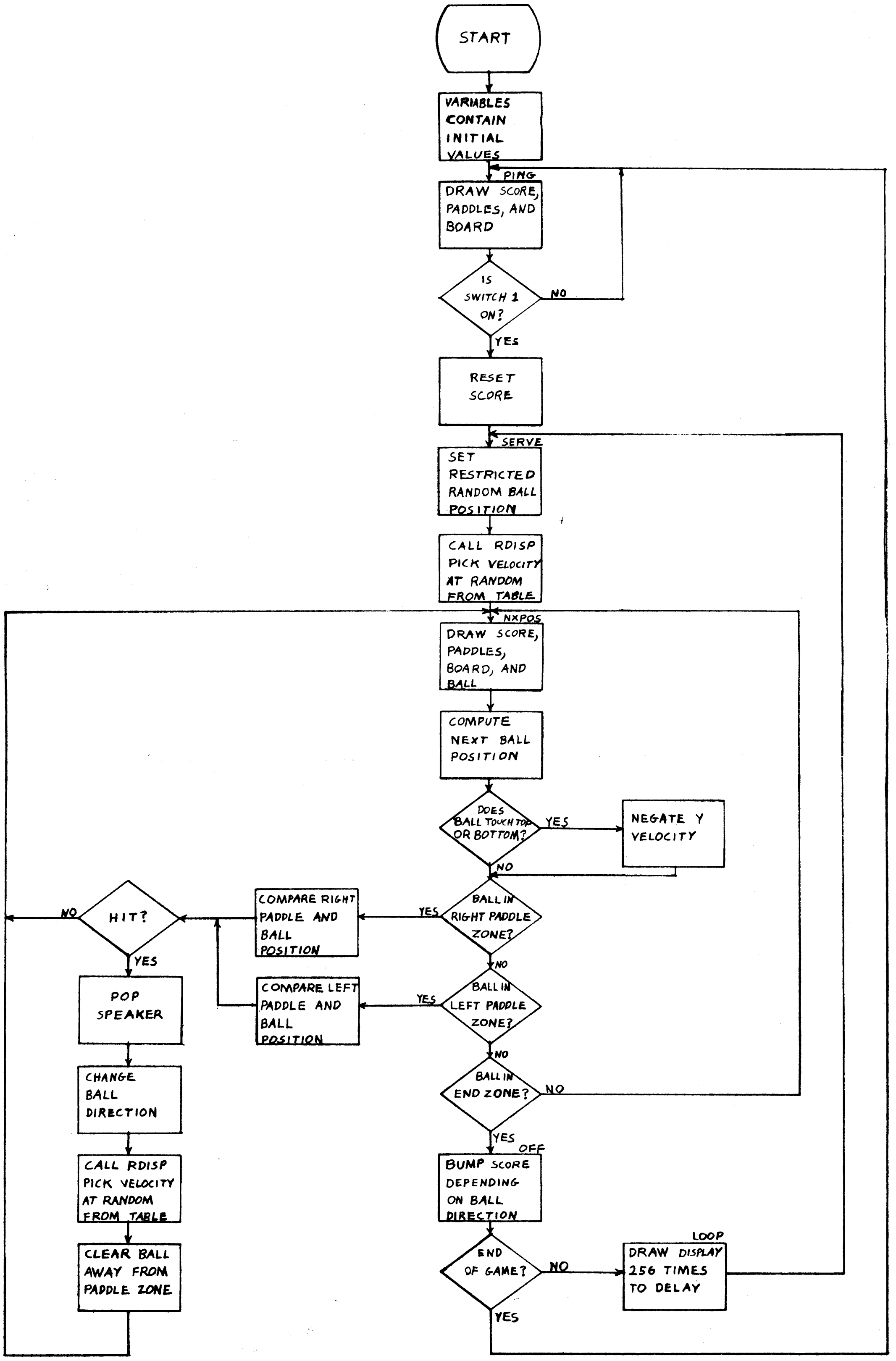
```
                        *       16 BIT ADD ROUTINE

000333  363         DBLADD  LLD
000334  307                 LAM             LOAD LSB OF VALUE TO ADD
000335  364                 LLE
000336  207                 ADM             ADD BOTH LEAST SIG. BYTES TOGETHER
000337  370                 LMA             SAVE
000340  030                 IND
000341  363                 LLD
000342  307                 LAM             LOAD MOST SIG BYTE OF VALUE TO ADD
000343  040                 INE
000344  364                 LLE
000345  217                 ACM             ADD WITH CARRY
000346  370                 LMA             SAVE
000347  007                 RET             RETURN

                        *       16 BIT NEGATE ROUTINE

000350  250         NEGATE  XRA             ZERO REG A
000351  227                 SUM             LOAD NEGATIVE OF MEMORY
000352  370                 LMA             SAVE
000353  060                 INL
000354  006 000             LAI    0        ZERO REG A WITHOUT RESETTING CARRY
000356  237                 SBM             LOAD NEGATIVE OF MEMORY WITH BORROW
000357  370                 LMA             SAVE
000360  007                 RET

                        *       RETURNS RANDOM BYTE IN REGISTER A

000361  056 002 066 RAND    SHL    SHIFT+3         PT TO SHIFT BYTE 4
000364  375
000365  016 010             LBI    8               SET FOR 8 SHIFTS
000367  307                 LAM                    LOAD SHIFT BYTE 4
000370  002         RTOP    RLC
000371  002                 RLC                    MOVE BIT 28
000372  002                 RLC                    TO POSITION 31
000373  257                 XRM                    X OR BITS 28 & 31
000374  022                 RAL
000375  022                 RAL                    MOVE NEW BIT INTO CARRY
000376  066 372             LLI    L(SHIFT)        PT TO SHIFT BYTE 1
000400  307                 LAM                    LOAD SHIFT BYTE 1
000401  022                 RAL                    ROTATE THRU CARRY
000402  370                 LMA                    SAVE
000403  060                 INL
000404  307                 LAM                    LOAD SHIFT BYTE 2
000405  022                 RAL                    ROTATE THRU CARRY
000406  370                 LMA                    SAVE
000407  060                 INL
000410  307                 LAM                    LOAD SHIFT BYTE 3
000411  022                 RAL                    ROTATE THRU CARRY
000412  370                 LMA                    SAVE
000413  060                 INL
000414  307                 LAM                    LOAD SHIFT BYTE 4
000415  022                 RAL                    ROTATE THRU CARRY
000416  370                 LMA                    SAVE
000417  011                 DCB
000420  110 370 000         JFZ    RTOP            REPEAT 8 TIMES
000423  007                 RET

000424  307         DRAW1   LAM             LOAD SCORE INDEX
000425  360                 LLA             INDEX SCORE CHAR. TABLE
000426  006 177             LAI    177B
000430  123                 OUT    YMOV     POSITION CHARACTER AT TOP OF DISPLAY
000431  007                 RET
                        *   DRAWS BOARD, PADDLES, AND SCORE

000432  006 377     DRAW    LAI    377B
000434  133                 OUT    MINSZ    SET MINOR DEFLECTION SYSTEM FOR MAX RANGE
000435  056 002 066         SHL    SCORE
000440  025
000441  106 024 001         CAL    DRAW1
000444  006 260             LAI    260B
000446  121                 OUT    XMOV     POSITION RLEFT SCORE
000447  106 127 001         CAL    CHAR     DRAW IT
000452  066 026             LLI    L(SCORE+1)
000454  106 024 001         CAL    DRAW1
000457  006 100             LAI    64
000461  121                 OUT    XMOV     POSITION RIGHT SCORE
000462  106 127 001         CAL    CHAR     DRAW IT
000465  066 104             LLI    L(BOARD)
000467  106 200 001         CAL    GRAPH    DRAW OUTSIDE SQUARE
000472  006 001             LAI    1
000474  106 137 001         CAL    RVCD
000477  123                 OUT    YMOV     POSITION LEFT PADDLE
000500  006 224             LAI    -108
000502  121                 OUT    XMOV
000503  056 002 066         SHL    LPADD
000506  147
000507  106 127 001         CAL    CHAR     DRAW LEFT PADDLE
000512  006 004             LAI    4
000514  106 137 001         CAL    RVCD
000517  123                 OUT    YMOV     POSITION RIGHT PADDLE
000520  006 151             LAI    105
000522  121                 OUT    XMOV
000523  056 002 066         SHL    RPADD
000526  127
```

```
                        *       DRAWS WITH MINOR DEFLECTION SYSTEM

000527  307         CHAR    LAM             LOAD BYTE FROM MEMORY
000530  260                 ORA             SET SIGN BIT
000531  131                 OUT    MINXY    DRAW BYTE
000532  063                 RTS             RETURN IF LAST BYTE
000533  060                 INL             BUMP MEMORY INDEX
000534  104 127 001         JMP    CHAR     DRAW NEXT BYTE FROM TABLE

000537  310         RVCD    LBA             COMPUTE BIT MASK FROM DIAL NUMBER
000540  006 020             LAI    20B      START WITH MASK OF 00010000
000542  012         RVCD1   RRC             SHIFT MASK RIGHT UNTIL DIAL
000543  011                 DCB             NUMBER BECOMES ZERO
000544  110 142 001         JFZ    RVCD1
000547  310                 LBA             SAVE BIT MASK IN B
000550  056 200             LHI    200B     INITIALIZE TRIAL VALUE IN H
000552  365                 LLH             INITIALIZE TRIAL BIT IN L
000553  305         RVCD2   LAH             GET CURRENT TRIAL VALUE IN A
000554  256                 XRL             FLIP CURRENT TRIAL BIT
000555  121                 OUT    XMOV     SEND TO DIGITAL-TO-ANALOG CONVERTER
000556  300                 LAA             WAIT FOR 741'S TO SETTLE
000557  300                 LAA
000560  115                 INP    GINP     READ COMPARATORS & SWITCHES
000561  241                 NDB             MASK TO GET CHANNEL OF INTEREST
000562  150 170 001         JTZ    RVCD3    JUMP IF TRIAL TOO HIGH
000565  305                 LAH             RETAIN TRIAL IF TOO LOW
000566  256                 XRL
000567  350                 LHA
000570  306         RVCD3   LAL             SHIFT TRIAL BIT RIGHT 1
000571  012                 RRC
000572  360                 LLA
000573  100 153 001         JFC    RVCD2    DO ANOTHER ITERATION IF ALL BITS NOT TRIED
000576  305                 LAH             LOAD FINAL RESULT INTO A
000577  007                 RET             RETURN

                        *       DRAWS WITH MAJOR DEFLECTION SYSTEM

000600  317         GRAPH   LBM             GET TABLE COUNT OF COORD PAIRS TO DRAW
000601  250                 XRA
000602  131                 OUT    MINXY    RESET MINOR DEFLECTION SYSTEM
000603  060                 INL
000604  307                 LAM
000605  121                 OUT    XMOV     POSITION BEAM X COORDINATE
000606  060                 INL
000607  307                 LAM
000610  123                 OUT    YMOV     POSITION BEAM Y COORDINATE
000611  011         GRAPH1  DCB             DCREMENT COUNT
000612  053                 RTZ             RETURN WHEN DONE
000613  060                 INL
000614  307                 LAM
000615  300                 LAA             DELAY
000616  300                 LAA             DELAY
000617  125                 OUT    XSTOR    LOAD X COORD
000620  060                 INL
000621  307                 LAM
000622  127                 OUT    YDRAW    DRAW LINE
000623  104 211 001         JMP    GRAPH1   LOOP

                        *                   ALL VARIABLES MUST BE LOCATED IN THE SAME
                        *                   PAGE

001000                      ORG    1000B
001000              DATA    EQU    *                BEGINNING OF DATA AREA

001000  040 131 136         DEF    40B,131B,136B,147B               ZERO
001003  147
001004  167 176 171         DEF   167B,176B,171B,160B
001007  160
001010  340                 DEF   340B

001020                      ORG    DATA+20B
001020  040 160 050         DEF   040B,160B,050B,157B               ONE
001023  157
001024  345                 DEF   345B
001025  000 000     SCORE   DEF    0,0      KEEPS PLAYERS' SCORE
001027  000 001     XDISP   DEF    0,1      PRESENT BALL X VELOCITY COMPONENT
001031  000 000     YDISP   DEF    0,0      PRESENT BALL Y VELOCITY COMPONENT
001033  000         DIRECT  DEF    0        INDICATES BALL DIRECTION
001034  000         DELAY   DEF    0        USED TO DELAY BETWEEN SERVES

001040                      ORG    DATA+40B
001040  026 137 157         DEF    026B,137B,157B,154B              TWO
001043  154
001044  111 110 360         DEF    111B,110B,360B
001047  000 000 000 BALPOS  DEF    0,0,0,0    PRESENT BALL POSITION
001052  000

001060                      ORG    DATA+60B
001060  030 170 177         DEF   030B,170B,177B,137B               THREE
001063  137
001064  074 334             DEF   074B,334B
001066  000 110 111 BALL    DEF    0,110B,111B,101B
001071  101
001072  300                 DEF    300B

001100                      ORG    DATA+100B
001100  050 157 124         DEF   050B,157B,124B,364B               FOUR
001103  364
```

```
001104  005             BOARD   DEF    5
001105  200 200 200             DEF    -128,-128,-128,127
001110  177
001111  177 177 177             DEF    127,127,127,-128
001114  200
001115  200 200                 DEF    -128,-128

001120                          ORG    DATA+120B
001120  030 150 161             DEF   030B,150B,161B,163B              FIVE
001123  163
001124  133 136 366             DEF   133B,136B,366B
001127  000 120 127 RPADD       DEF    0B,120B,127B,307B
001132  307

001140                          ORG    DATA+140B
001140  046 136 123             DEF   046B,136B,123B,120B              SIX
001143  120
001144  150 153 323             DEF   150B,153B,323B
001147  020 100 107 LPADD       DEF    20B,100B,107B,327B
001152  327

001160                          ORG    DATA+160B
001160  050 167 327             DEF   050B,167B,327B                   SEVEN

001200                          ORG    DATA+200B
001200  020 160 167             DEF   020B,160B,167B,127B              EIGHT
001203  127
001204  120 024 364             DEF   120B,024B,364B

001220                          ORG    DATA+220B
001220  050 157 137             DEF   050B,157B,137B,126B              NINE
001223  126
001224  125 134 354             DEF   125B,134B,354B

001240                          ORG    DATA+240B
001240  040 131 136             DEF   040B,131B,136B,147B              TEN
001243  147
001244  167 176 171             DEF   167B,176B,171B,160B
001247  160
001250  140 000 120             DEF   140B,000B,120B,010B
001253  010
001254  117 305                 DEF   117B,305B

001260                          ORG    DATA+260B
001260  040 160 050             DEF   040B,160B,050B,157B              ELEVEN
```

```
001263  157
001264  145 000 120             DEF   145B,000B,120B,010B
001267  010
001270  117 305                 DEF   117B,305B

                    *                                 TABLE OF BALL VELOCITIES

001272  144 000 012 ANGLE       DEF    100,0,10,254        -79 DEG
001275  376
001276  304 000 047             DEF    196,0,39,254        -67 DEG
001301  376
001302  034 001 126             DEF   28,1,86,254          -56 DEG
001305  376
001306  152 001 226             DEF   106,1,150,254        -45 DEG
001311  376
001312  252 001 344             DEF   170,1,228,254        -34 DEG
001315  376
001316  331 001 074             DEF   217,1,60,255         -22 DEG
001321  377
001322  366 001 234             DEF   246,1,156,255        -11 DEG
001325  377
001326  000 002 000             DEF   0,2,0,0              0 DEG FAST
001331  000
001332  000 001 000             DEF   0,1,0,0              0 DEG SLOW
001335  000
001336  366 001 144             DEF   246,1,100,0          11 DEG
001341  000
001342  331 001 304             DEF   217,1,196,0          22 DEG
001345  000
001346  252 001 034             DEF   170,1,28,1           34 DEG
001351  001
001352  152 001 152             DEF   106,1,106,1          45 DEG
001355  001
001356  034 001 252             DEF   28,1,170,1           56 DEG
001361  001
001362  304 000 331             DEF   196,0,217,1          67 DEG
001365  001
001366  144 000 366             DEF   100,0,246,1          79 DEG
001371  001

001372              SHIFT       DST   4       RANDOM NUMBER REGISTER
                    *                         SET ANY WAY EXCEPT ALL ZEROES

001376                          END   PING
```




START
VARIABLES CONTAIN INITIAL VALUES
PING
DRAW SCORE, PADDLES, AND BOARD
IS SWITCH 1 ON?
NO
YES
RESET SCORE
SERVE
SET RESTRICTED RANDOM BALL POSITION
CALL RDISP PICK VELOCITY AT RANDOM FROM TABLE
NXPOS
DRAW SCORE, PADDLES, BOARD, AND BALL
COMPUTE NEXT BALL POSITION
DOES BALL TOUCH TOP OR BOTTOM?
YES
NEGATE Y VELOCITY
NO
BALL IN RIGHT PADDLE ZONE?
YES
COMPARE RIGHT PADDLE AND BALL POSITION
HIT?
NO
YES
POP SPEAKER
CHANGE BALL DIRECTION
CALL RDISP PICK VELOCITY AT RANDOM FROM TABLE
CLEAR BALL AWAY FROM PADDLE ZONE
NO
BALL IN LEFT PADDLE ZONE?
YES
COMPARE LEFT PADDLE AND BALL POSITION
NO
BALL IN END ZONE?
NO
YES
OFF
BUMP SCORE DEPENDING ON BALL DIRECTION
END OF GAME?
NO
LOOP
DRAW DISPLAY 256 TIMES TO DELAY
YES

SURPLUS SUMMARY

Guess what! IMP-16 chip sets have hit a new low in price. This is perfect timing, next issue of TCH will begin a series on the IMP-16. The new deal is from Poly-Paks who, in their latest flyer, have the 5 chip set listed for $49.99.

Poly Paks
Box 942B
Lynnfield, Mass. 01940

Components useful for graphics displays have finally shown up as surplus. Suntronix has a batch of Sanders 720 CRT heads. These heads are "dumb" units, i.e., they have no buffer or character generator, all data comes in over a cable. However they are great for graphics because they used stroke type character generation rather than raster scan. The power supply, CRT, deflection amplifiers, and possibly even the DAC's from the unit could be utilized. Unfortunately, Suntronix is not shipping at this time due to pending court action by Sanders regarding interpretation of the word "scrap" in the contract.

Suntronix
6 King Richard Drive
Londonerry, NH 03053
Ph. 603/434-4644

Want an easy to use but somewhat weird replacement for paper tape? well Delta t has one. Its an 8 track incremental magnetic tape recorder, except the "tape" is a 16 MM magnetic film cartirdge. The unit will read and write 330 bytes per second asynchronously (you need not control data rate as long as you do not exceed 330 bytes/sec.). From their pictures the unit looks well built and they seem to have all the necessary items including interface from the 12 volt logic to TTL and the weird cartridges. The recorder goes for $250.00, the modifications for TTL (assembled, installed, and tested) adds $100, and extra cartridges go for $20.00. For details write:

Delta t
11020 Old Katy Road
Suite 204
Houston, TX 77043

TCH cassette boards are still available and will be until notice is given otherwise. The relay offer is another matter however, the supply is nearly exhausted so if you order them send two seperate checks, one for the boards, and one for the relays which we could return when the supply is gone.

CLASSIFIED ADS

There is no charge for classified ads in TCH but they must pertain to the general area of computers or electronics, and must be submitted by a non-commercial subscriber. Feel free to use ads to buy, sell, trade, seek information, announce meetings, or for any other worthwhile purpose. Please submit ads on seperate sheets of paper and include name and address and/or phone number. Please keep length down to 10 lines or less.

MEDICAL APPLICATIONS: I am interested in contacting individuals with a serious interest in the application of microprocessor technology to medical instrumentation, and automated diagnostic systems. Please contact James A. Willis, 3013 Woodlawn Ave., Falls Church, VA 22042. Ph. 703/532-8242

FOR SALE: TMS 4030 ZA0248 Dynamic RAM's. 420ns access time, 690ns cycle. Ideal for 8080 and IMP-16 microprocessors. $13 each, 8 or more $10 each. Andy Pitts, PO Box 5734, Winston Salem, NC 27103. Ph. 919/765-1277

FOR SALE: Precision 1% metal film 1/4 watt 100 ohm resistors, $2.00 per 50. Postpaid. C. Funk, 711 Eno Street, Hillsboro, NC 27278

FOR SALE: 10 CPS hard copy teleprinter with keyboard. Exact equivalent of TTY KSR. Serial in/serial out. Only one available. This will go fast at only $300. M. W. Smith, 4355 S. High Street, Englewood, CO 80110

POWER SUPPLY: I have a quantity of 5V 10 amp highly regulated power supplies taken from keyboard terminals. I will provide schematics and plans for obtaining -5V, -9V, and -12V. $25 plus postage on 15 pounds. Grant Runyan, 1146 Nirvana Rd., Santa Barbara, CA 93101

CASSETTE TAPE ROM ORDER FORM

CPU Type □ 8008 □ 8080

Status input device address: _______

Control output device address: _______

Memory page allocated to the ROM: _______

Memory load address for the IPL program: _______

Stack address for the IPL program (8080 only): _______

(Please specify all device and memory addresses in octal)

NAME ____

STREET ____

CITY _______________ STATE ___________ ZIP ________

We can program 1702, 1702A, and 5203 PROMS. At this time, TCH can only program customer supplied ROMS. The programming charge of $2.00 covers programming, verification, and return shipment. An octal listing of the ROM contents will also be enclosed. We are using an unmodified Intel PROM programmer and are strictly following the manufacturer's programming recommendations. Any PROMS that we cannot program successfully are definitely bad and will be returned. The printout will have the errors marked to aid you in obtaining a refund from your supplier. Unfortunately bad PROMS require as much effort as good ones so there will be no refund.

THE COMPUTER HOBBYIST
Box 295
Cary, NC 27511

ADDRESS CORRECTION REQUESTED

```
***** *   * *****    ****  ***  *   * ****  *   * ***** ***** ****    *   *  ***  ****  ****  *   *  ***   ***  *****
  *   *   * *       *     *   * ** ** *   * *   *   *   *     *   *   *   * *   * *   * *   * *   *   *   *   *   *  
  *   *   * *       *     *   * * * * *   * *   *   *   *     *   *   *   * *   * *   * *   *  * *    *   *       *  
  *   ***** ****    *     *   * *   * ****  *   *   *   ****  ****    ***** *   * ****  ****    *     *    ***    *  
  *   *   * *       *     *   * *   * *     *   *   *   *     * *     *   * *   * *   * *   *   *     *       *   *  
  *   *   * *       *     *   * *   * *     *   *   *   *     *  *    *   * *   * *   * *   *   *     *    *   *   *  
  *   *   * *****    ****  ***  *   * *      ***    *   ***** *   *   *   *  ***  ****  ****    *    ***   ***    *  
```

Volume 1 Number 8

As promised, this issue of TCH starts two new series of articles, one on an IMP-16 system and the other on interfacing to the Altair 8800. Both of these series will include detailed discussions and examples of known good circuits. Upcoming issues of TCH will also see a new series, this one on a floppy disk interface that is buffered and can be hooked to any processor. MOS Technology has created quite a stir with their 6502 (see article inside) and we at TCH would like to know if readers are interested in a series about it.

There is a recording standards conference sponsered by BYTE Magazine coming up this month in Kansas City. This will be the first effort in the hobbyist computer industry to get the different manufacturers together along with users. TCH will be represented at the conference by Hal Chamberlin and Richard Smith. With luck this will be the beginning of a new era for the hobbyist. This conference could also lead to some standardization of interfaces in general, thus allowing one manufacturer's peripherals to plug into another's system.

This issue also contains two other items that should be mentioned. In the letters column there is a discussion of some errors in the listing of the 8080 code for the cassette interface. The second item is a listing of clubs. This is something many people have asked about since TCH ceased printing the listing of our subscribers.

One final matter, TCH special offerings. TCH is now sold out of both 2102's and relays for the cassette board. See note inside if you still need relays. The cassette boards and ROM programming offer are still good. As always, all back issues are available. Also, with this issue TCH starts offering a universal wire wrapping card and 5 volt regulator kits for the card, see the Altair interfacing article inside.

INTERFACING THE ALTAIR 8800 PART 1

by Hal Chamberlin

This is the first of what will probably be a long series of articles on direct interfacing of peripherals and memory to the ALTAIR 8800 parallel bus. Additional topics to be covered include direct memory access interfacing, a simple non-vectored interrupt system, noise control measures for large systems, and bus and power supply loading considerations. All circuits and techniques will be tested on a near minimal Altair 8800 with 1K of original memory. Most of the concepts are equally applicable to other just announced kits (see the new products column).

Although MITS is generally responsive to user's needs, particularly in software, we felt that there was a gap in the technical information on the bus system and interfacing. With nearly 10,000 machines in the field, we think that this will be particularly valuable information for our readers. Please keep in mind that we are not trying to compete with MITS or suggest that users not buy their boards. If a standard I/O board will do the job in a straightforward manner, by all means purchase one -- it will probably be less trouble and expense in the long run. On the other hand, many people have their own whizbang golly-gee I/O device to interface which is either impractical or too expensive to do with standard I/O interface cards. Music synthesizer control or graphics display interfaces are excellent examples. Additionally, an infinite variety of multi-function custom interfaces are possible with just enough of a particular resource to satisfy the application. The Altair is a great machine and full realization of its I/O system makes it even greater!

By far the best way to build a custom interface is on a so-called "wirewrap card" with wirewrap IC sockets and wrapped interconnections. Unfortunately, wirewrapping has a "high cost" stigma that has prevented many hobbyists from considering it. A very low cost wrapping tool ($1.70) made by Cambion and available through Solid State Systems exists however and is generally satisfactory for hobbyist use. If extensive custom interfacing is anticipated, Godbout has a relatively inexpensive ($47.50) motor driven wrapping tool. Unwrapping tools are outrageous for what they are ($5.58 from S.S.S.) but the time saved in making changes and correcting errors make them worth it. Wire for wrapping tends to be expensive also (1.5¢ per foot) but fortunately it goes a long way. Kynar insulation is easier to strip from the #30 solid conductor than is Teflon. The scissors type of stripper very carefully adjusted is OK for stripping Kynar insulation. It might be wise to buy such a stripper, adjust it once, and use it only for wirewrap wire. Wirewrap IC sockets are available from many sources with Solid State Systems being particularly good during one of their sales.

Wire wrap cards are now available from TCH for $20 and are made by the same PC shop that made our cassette borads. The wirewrap card is the same size as the other Altair cards and has a 100 contact edge connector for plugging into the bus. Also provided are power and ground connections for TTL IC's, numerous bypass capacitor pad sets, and two locations for 7805 regulators to supply regulated +5 from the unregulated bus +8. A photo of the artwork and an IC placement form are in the appendix. As a service, TCH also has a 5 volt regulator kit consisting of a 7805, heatsink, and two 47uF 10 volt bypass capacitors for $2.50 when purchased with a board. Connections between the custom I/O device and the interface card are made with DIP plugs into a row of 6 socket positions at the top of the board. Typically a short cable would go from the interface board to a rugged socket mounted in the holes provided on the Altair's back panel.

The only major problem with wirewrap boards is that they cannot be plugged into consecutive sockets since the spacing is only 3/4 inch. Alternate socket positions will have to be used making separate purchase of motherboards and 100 pin sockets advisable.

Now that the preliminaries are out of the way, lets get down to business with a discussion of system loading considerations. The Altair has three system resources that are subject to overloading. These are the I/O address space, the built-in power supply, and the parallel bus itself. We will take an objective, engineering approach in recommendations regarding allowable loading of these resources.

Since the 8080 CPU has the capability of addressing 256 input and output devices, there is generally no problem in having enough addresses to go around as there would be in an 8008 based system. The only real concern is that the user's custom interface addresses not interfere with any standard interface addresses. This is particularly true if one ever plans to use standard software or distribute his own software. We have not seen a comprehensive list of preferred device addresses for MITS interfaces (jumpers select the actual address) but the few that their software uses are all low addresses. One would probably be safe in assigning custom interface addresses in ascending order starting from 200 octal.

The internal power supplies have the following stated capacities: +8 unregulated at 8 amps, +16 unregulated at .8 amps, and -16 unregulated at .3 amps. All of these are connected to bus lines and are available for use by any plugin card. The +8 volt supply is regulated down to +5 volts on each card to power the TTL logic. This supply is the one most likely to be overloaded. The +16 supply is used primarily by the 8080 CPU and the 4K dynamic memory boards, as well as by op-amps on some standard interfaces. The -16 supply is very small and is used only for substrate bias on the 8080 and 4K chips (negliable drain) and to power op-amps. This supply may be overloaded if a large number of 1702 or 5203 PROMS are used on custom boards.

The primary guideline on power loading is don't overload the supplies. Since heat generated in the transformers is proportional to the square of the current drawn, even a minor overload will result in gross overheating. MI.S's approach to preventing +8 power supply overload is to specify that no board is allowed to draw more than .5 amp. Thus a fully stuffed system with 16 cards will not draw more than 8 amps. Considerations on the + and -16 volt supplies are not as straightforward but the basic idea is to insure that any sensible combination of up to 16 cards be guarenteed not to overload any power supplies. This was undoubtedly a wise move considering the large number of non-technical people buying Altairs.

When custom interfaces are designed or alternate source interfaces are purchased, it is the user's resposibility to insure that no power supplies are overloaded by his intended configuration. On custom interfaces we would generally recommend waiving the "half amp" rule and try to cram as many different interfaces as possible onto each board. Besides saving space and cost, one can actually get more I/O function per amp! The reason for this is that the 8T97 bus drivers are real power hogs (65 MA typical) and every separate board will likely need a few.

CONTINUED ON PAGE 4

LETTERS

Gentlemen,

I have just completed a satisfactory one page write and read to/from tape using the TCH cassette interface and associated software. I would like to point out a couple of errors I have discovered in the program. Specifically the address specified for the jump line 177515 of the "Tape Read Record Routine" should read 107 instead of 042 and also the register specified for MOV on line 177512 should be C, not A. (change instruction to 161 from 167)

In addition you should indicate that the ROM software is based on the assumption of normal (uninverted) input to the data buss and replacement of IC 16 and 17 with non-inverting buffers or gates. As it is the program will not work with the interface if built per the schematic using the 7403's specified for IC 16 and IC 17. I finally did stumble across the suggestion of the DM8094 in the hardware description, but only on the third reading. No mention of this is made in the software description.

Also, I wonder if you have thought about using some sort of shift-register approach for the DID recognition on read. As it is, a lost bit on the leading zeroes can cause complete loss of sync with resulting loss of the record.

Nevertheless, I'm quite impressed with the overall product. Hope to be able to use it as an inexpensive replacement for a disk system. Will need to go to a deck such as the Phi-Deck with remote rewind, etc. for that. Wonder if you will be looking into any special prices on those. Also wonder about your thoughts on record-playback electronics for them. I've looked at the control electronics and that aspect seems quite good; the record-playback bugs me though.

Thanks again for a darn good job on the cassette system so far. Keep up the good work.

Fred LaPlante

Several readers caught the errors you mentioned in the 8080 version of the cassette software, the result of hand translation of known good 8008 code into untried 8080 code. The data ID recognition routine does indeed simulate a 16 bit shift register. It should tolerate all sorts of garbage at the beginning of a record as long as the DID itself is intact. TCH has two Phi-Decks and is working on a high-performance interface for them.

Gentlemen:

You ought to be ashamed of yourselves! Your 8080 cassette software had 4 bugs in it, 2 of them fatal. Address 177512 is 167, MOV M,A must be 161, MOV M,C. Address 177515 is 302:042:377 must be 302:107:377.

Also in the read data bit routine, the JZ and JNZ on the clock bit should be interchanged to be consistant with the flow chart and comments. 177661 is 302:255:377 should be 312:255:377 and 177670 is 312:264:377 should be 302:264:377. These are the only bugs I've found so far -I hope they are just typing errors, but that "MOV M,A" looks suspicious.

I might be starting some undergrad research here at UW, and I'm going to stick with microcomputers and peripherals. If you folks can suggest some interesting projects, either hardware or software, I'd appreciate it.

P.S. I hope you haven't programmed any 8080 ROM's yet!

Paul Gumerman

We always knew those programmers at the University of Waterloo were sharp (WATFOR, WATFIV, SPITBOL, PL/C). Fortunately, these errors were corrected in the ROM's that have been processed but a couple with incorrect load address and/or stack address may have escaped before a bug in our ROM customizer program was discovered. Interesting note, we have not come across any bad PROMS yet. As for projects, why not build a graphics display and experiment with software for automated drafting using cassettes for drawing storage.

Sirs:

An aquaintance was kind enough to loan me a few of the recent TCH issues, and I appreciated very much the informative articles on a Cassette Standard. Having personally traversed the problems involved in cassette interface design, I am painfully aware of some of the difficulties, and I appreciated finding out why my various designs failed. Two comments I should like to make with respect to your proposed standard:

1. One system requirement for the wide acceptance of a standard (aside from those propoqed with economic clout, a la IBM) is that it admit to a variety of implementations. In particular, not all users will be satisfied with the particular hardware/software mix defined by your circuit; some, like myself, prefer to minimize the external hardware at all costs when software can do the job; others wish to minimize the software at the expense of extra hardware (within reason). With the exception noted under item 2 below, this design does admit to a wide variety of implementations because of the basic simplicity of the format. In fact, I intend to alter my own cassette design to be compatible.

2. I was absolutely astounded that any serious attempt at proposing a standard should so blatently ignore one of the few widespread existing standards in the field of the serialization of parallel data, that is, that the least significant bit is transmitted first. If it were not for the facts outlined in item 1 above, this would be relatively unimportant; however since in this case it is not the hardware that does the serial-to-parallel and parallel-to-serial conversion, there is the non-trivial factor of user and programmer experience to consider also (in systems with other serial I/O, it is convenient to use common subroutines for some of the common functions such as serial conversion, CRC conversion, etc.). The real kicker is this: with a trivial modification to your basic circuit, a $20 USRT can be added and the whole system run byte-parallel on the processor's interrupt system (the minimum software version I mentioned in item 1 above), in a completely data-compatible mode, except that every USRT on the market follows the existing serialization standard and transmits and receives the least significant bit first, which would make all of the data upside-down in the processor (on input) or on tape (on output). Fortunately it is not necessary to alter either the physical standard or your already-designed and running hardware to correct this fault; the serialization sequence is entirely controlled by software, and to correct the problem in the software is not a major difficulty. The Data ID need not be altered physically if it is reinterpreted: as two bytes transmitted serially becomes (in hex) 91, F5 or (in octal) 221, 365.

As I mentioned earlier, I have gone through considerable gyrations in getting my own cassette interface running, and I have developed a paper which traces my experience in connection with a philosophy of peripheral design for microprocessors. I expect to present the paper at an upcoming meeting of the Bay Area Homebrew Computer Club, and would consider an opportunity to publish it to a wider audience. The title of the paper: The Quest for a 25¢ Cassette Interface, or A Microcomputer Design Philosophy". If published in TCH, I would include schematics and program listings to correspond to the TCH standard.

Tom Pittman

It is indeed true that the ASCII standard for data transmission in serial form calls for sending the least significant bit first. The reason for this is obvious, ASCII is basically a 7 bit code with the eighth used for parity. If you want both the high order bit to be the parity bit and the parity bit to be transmitted last, then the least significant bit must be transmitted first. IBM, true to form, ~~transmits~~ the ~~most signi~~ficant bit first in their data communications equipment. At any rate, the TCH cassette format is intended to be transparent, that is, not tied to any particular character code. There is no agreement on bit order when data is written onto magnetic media. Ignoring other factors, most-significant-bit-first is the logical way to do it because a printout of the bit stream can then be read simply by drawing lines separating groups of 8 bits. Likewise, a scope display of the signal can be easily read. The USRT problem can be solved simply by reversing the leads to the parallel data I/O leads on the IC package.

Your article proposal sounds great, so don't waste any time getting us a draft of it.

THE COMPUTER HOBBYIST
Founded October, 1974

Stephen C. Stallings - Managing Editor
Hal Chamberlin - Contributing editor
Jim Parker - Contributor
Steven Roberts - Contributor
Clyde Butler - Photographer
Richard Smith - Programming consultant

THE COMPUTER HOBBYIST is published at intervals approaching monthly near Raleigh, N.C. The subscription rate for 12 issues is $6.00 for persons in the U.S.A., U.S. possessions, Canada, and Mexico. Subscriptions in all other countries are $8.50 per year for surface mail or $13.50 per year for air mail. All back issues are available and cost fifty cents each for persons in the U.S.A., U.S. possessions, Canada, or Mexico. For all other countries back issues cost 75 cents each by surface mail or $1.25 each by air mail. Remittances from countries other than the U.S.A should be in the form of a bank or postal money order.

THE COMPUTER HOBBYIST is seeking paid contributors. Material to be submitted should be typed or neatly written and must not appear to be soliciting business for any firm. For details, please inquire.

Advertizing space is available to interested firms. Please inquire for rates and formats.

All correspondence should be addressed to:

THE COMPUTER HOBBYIST
Box 295
Cary, NC 27511

EDITORIAL

One would think that computer hobbyists, of all people, would be cool, level-headed, and logical in their thinking. But let the subject of conversation turn to octal versus hexadecimal notation for the Intel micros (8008 and 8080) and a shouting match may develop. One thing we can agree on though is that Intel created the problem by introducing and supporting the 8008 for more than two years with octal notation and then switching to hex notation for both machines when the 8080 was introduced.

Both notations are simply a more convenient way for human beings to communicate long strings of ONES and ZEROES. They are based on a fundamental law of human memory which states that the difficulty experienced in memorizing a string of symbols does not depend on the size of the alphabet from which they were drawn. An example would be the relative difficulty in memorizing a sequence of 30 random English words and a sequence of 30 bits.

Lets first look at the merits of the two number systems themselves without regard to a particular machine. Octal notation was invented sometime in the '50s for use on early binary computers. Its development was natural for two reasons. First, characters were 6 bits long in those days and binary computers frequently used 36 bit words. Second, octal notation confines itself to the already familiar decimal digits 0-7. This was important back then since line printers often had a 13 character set ($-.0123456789) to facilitate high speed operation. Hexadecimal notation was introduced by IBM along with the 360 line of computers in 1964. According to Knuth (1) the proper term for base 16 notation is "sexadecimal" but IBM probably didn't like the idea of "sex notation" or "sex dumps". The 360's were the first widely used computers that had an 8 bit byte thus making hex a reasonable choice. By itself, hexadecimal notation is probably the better of the two because it is more compact requiring less print space and time (except on Baudot TTY and Selectric typewriters where case shifting makes it slower) and memorization effort. Octal on the other hand is easier to learn and understand and can be easily displayed on 7-segment readouts using common decoder-drivers. Since hex is associated with IBM, some companies refuse to use it even though they should.

Now let's look at octal versus hex on the Intel micros. Since the machine word is 8 bits long, hex is a good first choice. However, the instruction set designers set up all of the instruction formats so that the bit field boundaries matched octal digit boundaries. This is evident on any instruction that referrs to registers because the register number (A=0, B=1, . . . M=7) becomes a digit in the 3 digit octal equivalent of the instruction. For example, LBM load B from memory, is coded 317 (in 8008) where 3 is a two bit "op code", the 1 is register B, and the 7 is memory pointed to by registers H and L. Things are shifted around in the 8080 but the octal oriented structure is unchanged. It should be obvious that with hex notation this correspondance between numeric instruction code and instruction function is lost requiring the some 200 instructions to be memorized separately or looked up in a table.

While octal is ideal for instructions, it is less than perfect for addresses. The problem arises because addresses are two bytes (16 bits) long and 3 digit octal notation implies a 9 bit number with the leftmost bit always zero. If addresses are considered to be 16 bit numbers, then the address 132764 is in page 265 at word 364. Note that the 6 digit octal address cannot be split into a 3 digit octal page and 3 digit word directly whereas the hex equivalent, B5F4 page B5 word F4, can. Actually the octal can be split with a little thought. To go from 6 digit address to page and word using the above address as an example, take the left 3 digits and double the octal number mentally (132 X 2 = 264). If the fourth digit from the left is 4, 5, 6, or 7, add one to your result (264 + 1 = 265) and you will have the page number. If the fourth digit is 0, 1, 2, or 3, the last 3 digits are already the word number. Otherwise subtract 4 from the fourth digit and you will have the word number (764 - 400 = 364). Going from page and word to 6 digits is also easy. First look at the page number. If it is odd, add 400 to the word number to form the last 3 digits of the result. If the page was even, then the last three digits are the same as the word number. The upper 3 digits are simply the page number divided by two in octal with the remainder discarded. Even these mental gymnastics are, I think, preferable to memorizing the hex op codes.

The ideal solution would, of course, be a notation that is good for both instructions and addresses. Some have suggested octal instructions and hex addresses. A better solution is called "split octal" notation. A split octal address is written as 265:364 which is page 265, word 364. The colon separates the two halves and reminds you that this is a split octal number, not a natural number. The only potential problem is in converting split octal to decimal. The digit weights are now 16384, 2048, 256 : 64, 8, 1. Possibly a better method of conversion with a pocket calculator is to convert each half separately and then multiply the page number by 256 and add in the decimal word number.

TCH will be shifting to split octal notation in our 8008/8080 program listings when our new assembler is finished. We think that it offers the best balance between ease of learning and ease of use. For other machines, we will follow the manufacturer's recommendations unless they show the same indecision already demonstrated by Intel.

[1] Knuth, Seminumerical Algorithms, Addison-Wesley Publishing Co., 1969.

CLUBS ETC.

Since last issue the staff of TCH has been collecting information about clubs and other organizations which have sprung up to get computer hobbyists together. These organizations along with what information we were able to find are listed below:

LOS ANGELES, CALIFORNIA
Southern California Computing Society
Club newsletter - Interface
Contact - Hal Lashlee
Box 987
South Pasadena, CA 91030
213/682-3108

NORFOLK, VIRGINIA
Peninsula Computer Hobbyist Club
Contact - Larry Pollis
2 Weber Lane
Hampton, VA 23663
803/723-3117
Officers - Larry Pollis, President
Frank Pards, Vice President

NEW JERSEY
Amateur Computer Group of New Jersey
Club newsletter - ACGNJ NEWS
Contact - Sol Libes
995 Chimney Ridge
Springfield, NJ 07081
201/889-2000 day
201/277-2063 evenings

ATLANTA, GEORGIA
Atlanta Area Microcomputer Hobbyist Club
Club newsletter - AAMHC Newsletter
Contact - Richard Stafford
3144 Parkridge Crescent
Chamblee, GA
404/455-0118

SAN FRANCISCO, CALIFORNIA
Homebrew Computer Club
Club newsletter - Homebrew Computer Club Newsletter
Contact - Homebrew Computer Club
Box 626
Mountainview, CA 94040

SAN DIEGO, CALIFORNIA
San Deigo Computing Society
Club newsletter - Personal Systems
Contact - Personal Systems
10137 Caminito Jovial
San Diego, CA 92126
714/223-7853

CANADA
Tenative formation
Contact - G. Pearson
861 - 11th. Street
Brandon, Manitoba
CANADA R7A 4L1

DALLAS, TEXAS
North Texas No Name Computer Club
Contact - Bill Fuller
2377 Dalworth 157
Grand Prarie, TX 75050

Thats it for computer clubs, however TCH feels that two other groups are worth mentioning. Here they are:

BIT Users Association
3010 4th. Avenue South
Minneapolis, MN 55411
612/824-8247

This group, organized by Richard Koplow, is continuing support for the products of the now defunct BIT company. This should be of interest to those of you who acquired BIT-480 computers.

HP-65 Users Club
2541 West Camden Place
Santa Ana, CA 92704

This group is concerned with programmable pocket calculators (they no longer confine themselves to HP-65's) and they put out a fine newsletter, HP-65 NOTES.

Since it is inevitable that some of the above information is incorrect or out dated we will be printing updates next issue. Also if new groups start or we missed a current one, please let us know.

CONTINUED FROM PAGE 1

With several interfaces on one card, only one set of bus drivers is needed with multiplexors directing data from each interface to the bus drivers. Bus loading is also reduced since only one set of receivers is needed. The real limit is 2 amps set by the 7805's used on TCH's wirewrap board.

What about additional booster supplies if the standard ones are used to capacity? Adding the booster supply is somewhat tricky. Simple paralleling across the supply to be boosted is not recommended unless the components used are identical to the original. There is also a lack of space if the I/O connections are used and a fan is installed (recommended when the system becomes half filled). The only reasonable way to connect the booster supply is to cut the power bus for that voltage somewhere and make sure that the cards are installed such that neither the original nor the booster is overloaded. We would not recommend any substantial increase in the amount of +8 or +16 volt power available unless better internal ventilation is provided for. It would be permissable to double the -16 supply capacity by substituting a larger transformer and filter cap if necessary to power up to 4K worth of eraseable PROMS.

To illustrate the above points and also put a current controversy to rest lets compare MITS's 4K dynamic memory card with the 4K static memory cards available from alternate sources. The only real technical difference is in power resource use. The static RAM boards are stuffed full of 2102 equivalents (32 of them) and draw all of their power (about 1.5 amps) from the +8 unregualted supply. This power drain is constant, thus 4 cards for 16K would use up 6 amps leaving very little for I/O interfaces after the CPU and original 1K card are added in. Dynamic memory on the other hand draws power only when it is addressed (operating power) and very little when it is not addressed (standby power). Furthermore, most of the power is taken from the +16 volt supply; the +8 volt requirements are less than .5 amp to power the TTL overhead circuitry. If MITS had chosen to put 8K or even 16K of memory on a card, then even the TTL overhead power would have been negligible on a per K basis. Since only one group of 8 RAM chips can be addressed at a time, the power drawn from the +16 supply by a bunch of dynamic cards is only slightly more than that drawn by one card. The Altair could easily power 60K of dynamic memory but that much static memory would need a whopping 22 amps. So while it is true that one static card takes only a little more power in watts than one dynamic card, there is no comparison when several are used. As before, the main consideration is that power is a limited resource and it is up to the user to decide how that resource is divided up.

Bus drive capability is another limited resource. The 8T97 drivers used by MITS can drive 48 MA at .5 volts. The drive capability at the more common spec point of .4 volts is somewhat less, probably around 32 MA. MITS specifies a maximum bus loading of one low power TTL load (.16 MA at .4 volt) per card. Assuming a 32 MA drive capability, this figures out to a maximum of 200 boards in a system. MITS however routinely violates their own loading rules by connecting two and sometimes three low power loads to some bus lines. For examples, look at the address decoding on nearly any board, especially the 1K static board. If you don't intend to ever expand beyond the 16 board maximum in the case, it would be perfectly permissable to have one regular TTL load per card on a bus line and still have 6.4 MA of reserve drive capacity in a full system.

Besides DC current loading of the bus lines there is AC capacitive loading to worry about. It is this capacitive loading that contributes to internally generated noise. The noise arises from the fast, powerful bus drivers charging up the load capacitance when logic states are switching. The 8T97's can supply a surge of almost a quarter amp for each output. If a large number of bus lines change state at once such as an address change from 077:377 to 100:000 the large current surge flowing through the ground lines can induce noise in the critical strobe lines and possibly cause false triggering. As capacitive loading increases, the charging current's amplitude and duration also increases. The capacitance of a gate input and associated wiring is about 7 pF regardless of whether it is low power, standard, or Schottky. Thus, in order to reduce noise generation, each bus line should connect to a minimum of gate inputs through short connecting wires. In no case should a bus line be allowed to connect to an I/O cable that leaves the card without buffering. We have found that low power Schottky (74LS series) devices make excellent bus receivers. They are as fast as regular TTL, can drive 5 TTL loads, and have an input current of only .36 MA. They are also becoming more available on the hobbyist market. Purchase of a couple dozen 74LS04's should satisfy the bus receiver requirements of several boards. 8097's (also 8095, 8096, and 8098) can drive the bus as well as 8T97's although they are not as fast and don't have as much surge current capacity, thus cutting down on noise generation somewhat.

Appendix 1 gives the timing diagrams for three types of data transfer cycles in the Altair, input, output, and read memory. Write memory will be discussed in a future article. The times given are measured values observed in our Altair. These are subject to some variation since most of them depend on characteristics of the 8080 chip itself. The relative timing relationships should remain the same however.

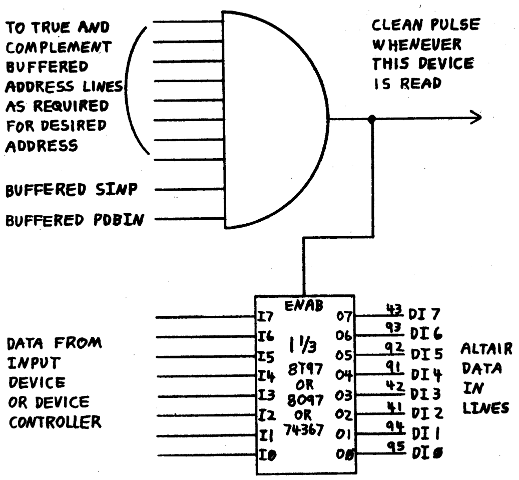

TO TRUE AND COMPLEMENT BUFFERED ADDRESS LINES AS REQUIRED FOR DESIRED ADDRESS
BUFFERED SINP
BUFFERED PDBIN
CLEAN PULSE WHENEVER THIS DEVICE IS READ
ENAB
I7
I6
I5
I4
I3
I2
I1
I0
1 1/3 8T97 OR 8097 OR 74367
O7
O6
O5
O4
O3
O2
O1
O0
43 DI 7
93 DI 6
92 DI 5
91 DI 4
42 DI 3
41 DI 2
94 DI 1
95 DI 0
DATA FROM INPUT DEVICE OR DEVICE CONTROLLER
ALTAIR DATA IN LINES

FIG. 1 SIMPLIFIED INPUT PORT SCHEMATIC

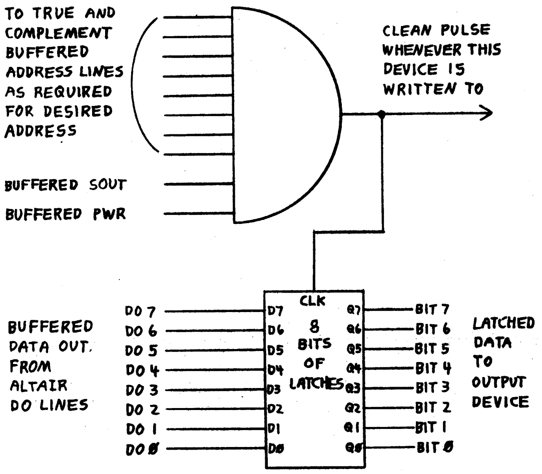

TO TRUE AND COMPLEMENT BUFFERED ADDRESS LINES AS REQUIRED FOR DESIRED ADDRESS
BUFFERED SOUT
BUFFERED PWR
CLEAN PULSE WHENEVER THIS DEVICE IS WRITTEN TO
CLK
8 BITS OF LATCHES
BUFFERED DATA OUT. FROM ALTAIR DO LINES
DO 7
DO 6
DO 5
DO 4
DO 3
DO 2
DO 1
DO 0
D7
D6
D5
D4
D3
D2
D1
D0
Q7
Q6
Q5
Q4
Q3
Q2
Q1
Q0
BIT 7
BIT 6
BIT 5
BIT 4
BIT 3
BIT 2
BIT 1
BIT 0
LATCHED DATA TO OUTPUT DEVICE

FIG. 2 SIMPLIFIED OUTPUT PORT SCHEMATIC

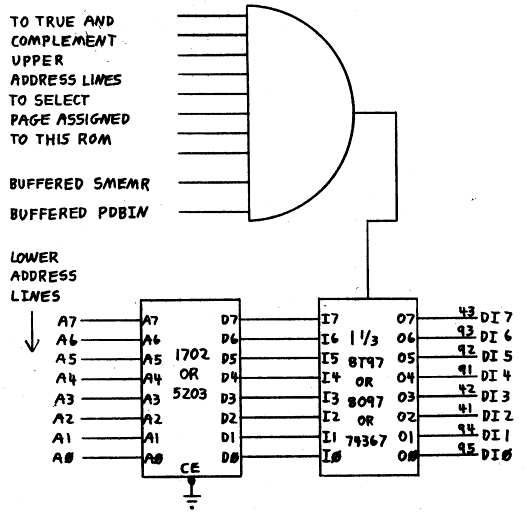
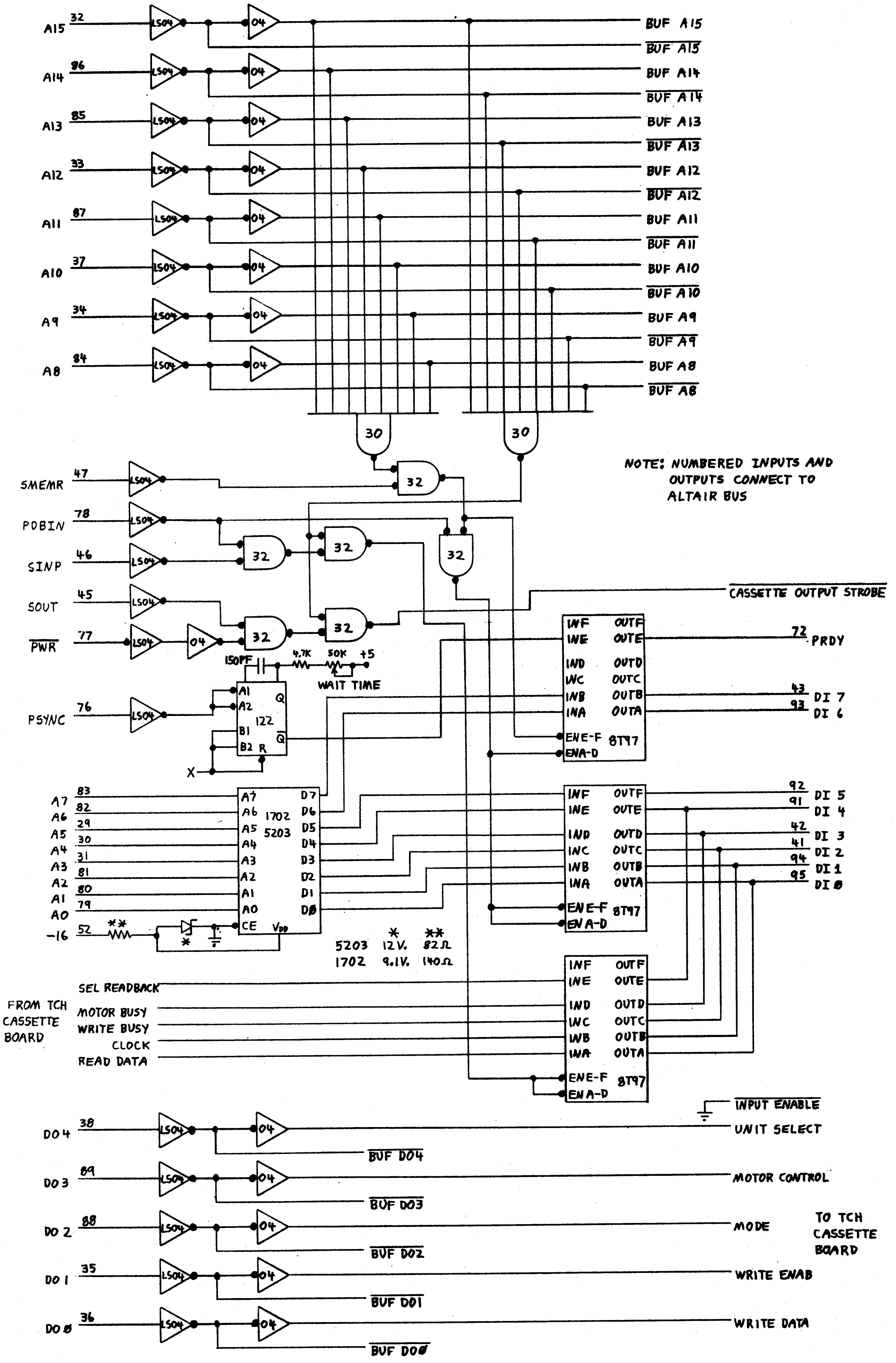
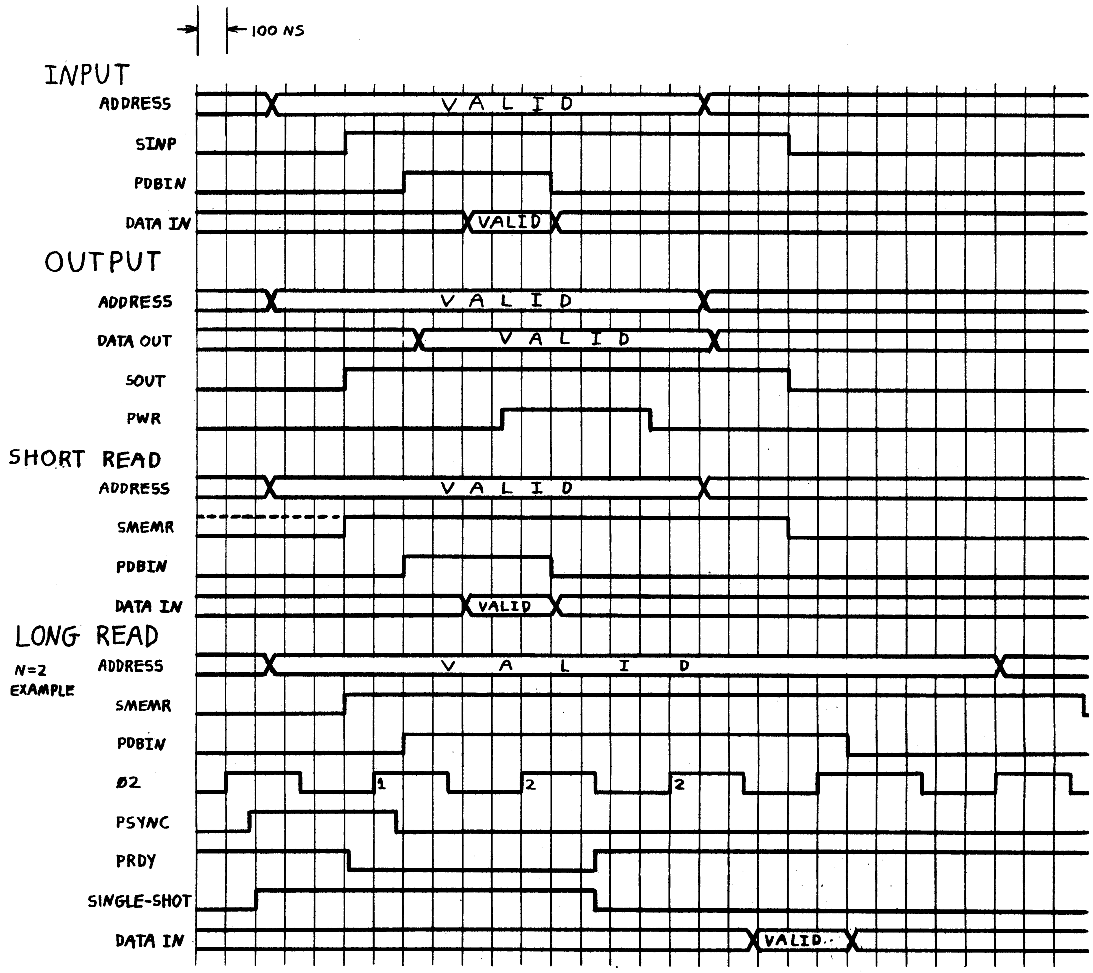


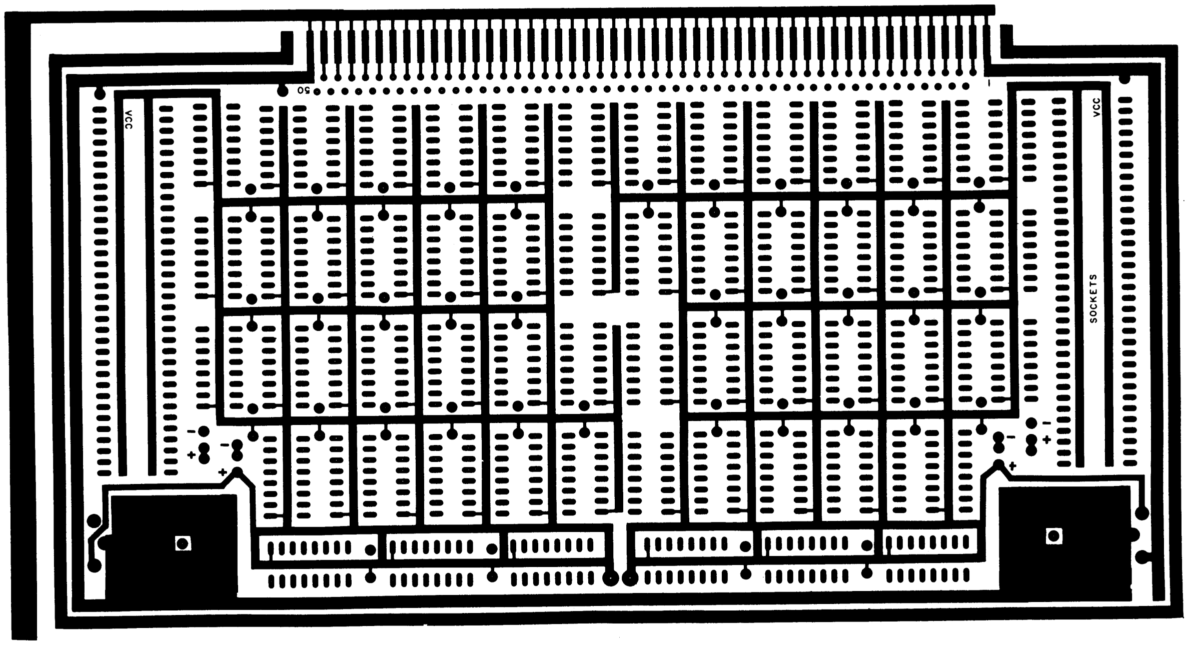
FIG. 3 SIMPLIFIED ONE PROM INTERFACE

FIG. 4 EXAMPLE INTERFACE FOR CASSETTE AND PROM



APPENDIX 1. ALTAIR BUS TIMING





An input cycle is distinguished by the signal SINP (pin 46) going high on the bus. The input device address should have already settled on the address bus lines before SINP becomes active. An interesting property of the 8080 is that the 8 bit I/O device address is duplicated in the high and low order halves of the 16 address lines. This property can be used to distribute address buss loading evenly in a system with many I/O devices. After both the address and SINP have settled, PDBIN (pin 78) goes high. The coincidence (AND function) of the proper address, SINP, and PDBIN indicates that the input device data should be gated onto the data-in bus, lines DI0 through DI7. This is typically done with tri-state drivers such as 8T97's. A delay of up to 200NS is permissable from when PDBIN becomes active to when valid data is required to be on the input bus. If a clean pulse to inform the input device that it has been read is needed, it should be the coincidence of proper address, SINP, and PDBIN. At the end of the input cycle, PDBIN disappears first followed by the address, and then finally SINP disappears. That is all there really is to an input operation! Once the bus buffering overhead is taken care of, an 8 bit input port can be as simple as a 10 input AND gate equivalent to detect the coincidence of the address and control signals, and 8 bits of tri-state buffers (see figure 1).

An output cycle is distinguished by the signal SOUT (pin 45) going high on the bus. As with the input cycle the device address is duplicated on the high and low halves of the address bus and is settled before SOUT goes high. Shortly after SOUT becomes active, the output data from the CPU settles on the 8 data out (DO) bus lines. Finally, after everything is settles and valid, PWR (pin 77) becomes active for almost exactly 500 NS signalling that the DO bus data should be latched into the output device's register. Thus the trigger pulse used to clock the data from the DO bus into the device register is the AND combination of proper address, SOUT, and PWR. Since valid address, valid data, and SOUT brackets the PWR pulse, the register used to receive the data may be either leading or trailing edge triggered. Even level triggered devices such as 7475's or a 74100 may be used. This "load register" pulse is also clean and may be used to inform the output device that new data has been loaded into its register. Note that the PWR signal appears on the Altair bus inverted ($\overline{\text{PWR}}$) but is easily inverted again in the buffering process. As with input, once the bus buffering overhead has been taken care of, an 8 bit output port can be as simple as a 10 input AND equivalent and an 8 bit register (see figure 2).

We will discuss read memory now instead of later so that readers interfacing the TCH audio cassette board can also include an erasable PROM with cassette software on their interface. A look at the short memory read cycle timing diagram will show that it is the same as the input cycle except that read memory is distinguished by the presence of SMEMR (pin 47). The data out from the memory should be gated onto the DIN bus lines at the coincidence of proper address range, SMEMR, and PDBIN. Address range means the range of addresses assigned to the particular block of memory. For the case of a single 1702 or 5203 PROM, the circuitry can be very simple as shown in figure 3 once the required buffering has been done. For the case of reading multiple PROMS, the chip selects should be decoded directly from the buffered address lines in order to give the PROM maximum time to respond.

Unfortunately, almost any erasable PROM likely to be available for a reasonable price will be too slow to use the simple short read cycle. The standard speed for 1702A's is 1.0uS with speed selections from 650NS to 2.5uS available from the manufacturer. National's 5203 has similar speed capability. As can be seen in the short read timing, there is approximately 650 NS available from when the address settles to when valid data must be on the DIN lines. Allowing for buffering delays and variations in the 8080 chip, the short read should not be used if the PROM access time is greater than 500NS.

The PRDY line (pin 72) in the Altair can be used by the memory interface to instruct the CPU to provide a long read cycle rather than the short cycle. This technique allows additional time for memory access in increments of 500NS. The long read timing diagram shows the PSYNC (pin 76), PRDY, and Ø2 (pin 24) signals in addition to the others used for memory read. The CPU makes the decision whether to execute a short or long read based on the state of the PRDY line at the rising edge of Ø2 designated with a 1. If PRDY is high at this time, a short read is done; if it is low, the CPU enters the "wait" phase of a long read cycle. The PRDY line has a pullup resistor attached so that if PRDY is not connected, short reads automatically occur.

While the CPU is waiting, both SMEMR and PDBIN are active meaning that data from the memory is being gated onto the DIN bus before it is valid. Fortunately, the CPU only looks at the data in the 300NS period immediately before PDBIN drops. Also, once the wait phase is entered, the CPU examines the PRDY line at every rising edge of Ø2 designated with a 2. The first time a ONE is seen, the cycle is completed and PDBIN drops.

Since memory chips don't have a pin to tell when the data is valid, the user must provide a time delay equal to or longer than the data sheet specified access time to drive the PRDY line. Also, the circuit should drive PRDY only when the block of memory it serves is addressed. In the Altair, it is difficult to tell from looking at just the address and SMEMR lines when the memory access delay should start (note that PDBIN starts after the CPU has made its decision). Fortunately a signal, PSYNC, is available which marks the start of a data transfer cycle. Somewhat later, one of the "S" signals (SOUT, SINP, SMEMR, etc.) distinguishes what kind of cycle it will be. An easy way to provide the memory access time delay is to trigger a single-shot unconditionally with PSYNC. The coincidence of proper address range and SMEMR is then used to gate the single-shot output onto the PRDY line through a tri-state or open collector gate. If this block of memory was not addressed after all, the single-shot times out harmlessly without affecting PRDY. The optimum time setting is T=(500N)+150 NS where N is an integer such that T+350 is greater than the memory access time. In the case of unknown access time (likely with surplus PROMS) a trimpot can be connected to the single-shot to vary the time from about 500NS to 3uS. The pot can then be adjusted for the fastest reliable operation. Note that the single-shot must be of the retriggerable variety (74122, 74123) if the delay is longer than 1.2uS. Otherwise the single-shot may not have recovered from a preceeding short cycle.

As an example to illustrate some of the concepts discussed, figure 4 shows a possible schematic for an interface between the TCH cassette board and the Altair. Also included is provision for a 1702 or 5203 PROM containing the cassette software. This circuitry would use only about 1/3 of the available space on a wirewrap board leaving room for other interfaces.

The cassette board in TCH's demonstration system is mounted in a separate Ten-Tek box with two relay override switches and four LED indicators mounted on the box front. Since there is a two foot cable between the cassette board and the Altair, all of the signals sent to or received from the cable are buffered. To use this circuit, the cassette board should be jumpered for true data input and the 7403's replaced with 7408's or 74125's so that the data out will also be true. The 74125's would allow a number of cassette boards to be paralleled if desired. Altair bus lines DO0 - DO4 are buffered by 74LS04's and reinverted by 7404's. While the second set of inverters could have been eliminated if the cassette board was jumpered for inverted data, it is quite likely that the buffered DO lines might be needed elsewhere on the board. Complete separation of signals sent to or received from a cable from other circuitry is always desirable for maximum noise immunity. The 5 bits coming back from the cassette board connect to a single 8T97 or equivalent to drive the DIN lines. The various control signals are also buffered with 74LS04 inverters.

The address decoding is done in a completely general way in anticipation of added interfaces on the same board. Each of the high 8 address lines is double inverted to provide both the true and complemented form. By connection of 8 input NAND gates to various combinations of true and complemented address lines, any I/O address or page address can be selected. Using the address connections shown, the cassette interface is at 200 octal and the cassette ROM is at page 377.

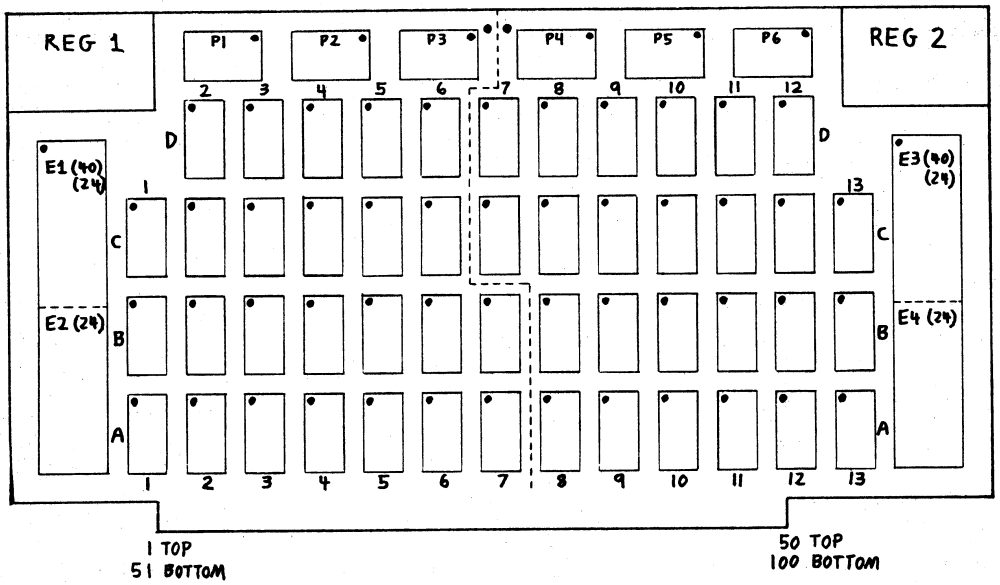
Type 7432 positive OR gates are used as NAND-NOT gates (see TCH # 2) to detect the coincidences of addresses and control signals. This eliminates the need for many inverters. The cassette output strobe is the coincidence of cassette address, SOUT, and PWR. The bus driver for the data back from the cassette is enabled by the coincidence of cassette address, SINP, and PDBIN. The drivers on the cassette board itself are always enabled by grounding the INPUT ENABLE signal to the cassette board.

If only one PROM is to be used, it is permissable to connect its address inputs directly to the low 8 address lines as shown. Although MOS circuits present no DC loading, the address input capacitance of several PROMS in parallel would increase the noise on the bus. The outputs of the PROM are only rated for one TTL load so the PROM chip enable is grounded and the DIN bus is driven with 8T97's or equivalent. The driver is enabled by the coincidence of ROM page address, SMEMR, and PDBIN. A 74122 retriggerable single-shot is triggered every time PSYNC goes up. An unused section of an 8T97 gates the inverted single-shot output onto the PRDY line when ROM address and SMEMR are present.

Power for the TTL and cassette board is taken from a 7805 regualtor on the wirewrap board. When only one PROM is used, the -9 or -12 volts required can be derived from the -16 bus supply with a simple zener diode regulator. If two or more PROMS are interfaced, a series regulator of some sort should be used to conserve power.

Additional interfaces can be added later to the board with very little additional circuitry since most of the bus signals are already buffered. If a number of input type interfaces are added (those that would drive the DIN lines), it might be wise to use only one set of 8T97's and a multiplexor to select data fed to the 8T97's. This would reduce capacitive loading on the DIN bus substantially. For example, we will probably put a keyboard interface, a graphics display interface (the XYZ signal generator would be in the display cabinet), and a couple more PROMS on our experimental board. As before, if the standard products meet your needs, use them. If not, the information presented here should make the job of custom interfacing easier and surer.

TCH 8080 WIRE-WRAP BOARD



TOP VIEW

NEW PRODUCTS

It seems that everybody and his brother is offering microcomputer kits of some sort these days. Three different ones have come to our attention in the last few weeks. In many cases the prices this month are half of what they were last month thanks to the bold lead taken by MOS Technology with their $20 microprocessor offering.

MITS has announced their long rumored MC6800 system. Logically it is called an ALTAIR 680 and not suprisingly it looks like a baby ALTAIR 8800. MITS has gone to larger, somewhat denser boards in the ALTAIR 680. For example, the CPU board has the MC6800 chip, 1K bytes of 2102 RAM memory, and a serial I/O port. In addition there are four sockets for 1702 erasable PROMS. This represents a significant departure from the ALTAIR 8800 philosophy which would have required 4 separate, smaller boards for the same function. The front panel is a separate board. In order to accomodate the slow speed of the 1702 PROMS, the clock frequency to the MC6800 is 500kHz, half of the maximum specified by Motorola.

The complete machine is enclosed in a smaller Optima cabinet (11"wide, 4 3/4" high, 11" deep) and includes built-in power supply. The kit prices reflect the recent deep price cuts by the semiconductor companies on microprocessor products. During the promotional period until the end of the year a complete kit (except erasable ROM's) is $293 or $420 for an assembled and tested unit. For the hardware nuts on limited budgets, the CPU board only is $180 for a kit or $275 assembled. Available software includes PROM monitor, assembler, debug package, and editor. We do not know whether the software is included or priced spearately.

MITS
6328 Linn N.E.
Albuquerque, NM 87108

The first kit we know of to use the MOS Technology microprocessor is being offered by Microcomputer Associates and is called, appropriately, JOLT. It sells for $249 as a kit and uses the "little bit of everything on the CPU board" philosophy that will be the hallmark of second generation computer kits. On a single board approximately 4" by 6 1/8" and using only 11 chips, there are 576 bytes of RAM (512 for program, 64 for stack), 1024 bytes of ROM with a debug monitor masked into it, a serial I/O port, and two 8 bit parallel I/O ports. The secret to the small number of parts is use of the MOS Technology MCS6530 "kitchen sink" chip. This 40 pin DIP has 64 bytes of RAM, 1024 bytes of masked ROM, two programmable I/O ports that work like Motorola's PIA, and an interval timer. Although the JOLT literature doesn't mention the timer, it should be there nevertheless. Use of the timer is far better than timed loops since it doesn't tie up the CPU and is not affected by memory waits.

An unusual feature of their debug monitor is a claimed self-adjustment to any serial data rate between 10 and 30 characters per second. This feature may have been necessary however since the on-board clock is not crystal controlled.

Other items in the JOLT line include a RAM board, an I/O board, a power supply, and a universal wire-wrap board. The RAM card sells for $265 as a kit and includes 4K of one microsecond static RAM, probably 2102's. This leads us to believe that the CPU clock is significantly slower that the 1mHz maximum in order to accomodate write cycles into the slow RAM (see article on MOS Technology chips elsewhere in this issue). CPU speed was not mentioned in the literature we received. The I/O card costs $96 as a kit and provides four 8 bit programmable I/O ports and two interrupt requests. It uses two of the PIA chips now made by Motorola and soon to be made by MOS Technology. The power supply is $145 as a kit and provides +5, +12, and -10. The current ratings were not specified but enough power for CPU, 4K of memory, and an additional I/O card is claimed. The wire-wrap card is the same size as the others and sells for $25 in single quantities.

The JOLT boards are interconnected by means of flat cable and Scothflex connectors rather than edge fingers and edge sockets. An "accessory bag" at $40 is available with the flat cable and connectors necessary to connect two boards together. Each additional card would require an additional accessory bag or equivalent. Software beyond the ROM based debug monitor was not mentioned.

Orders sent before Nov. 10, 1975 consisting of CPU cards and other items will receive a 20% discount on the other items.

Microcomputer Associates, Inc. Dept. A.
111 Main St.
Los Altos, CA 94022

SPHERE is now selling their CPU board separately. It too has a little bit of everything except memory which is a whopping 4K bytes on the CPU board. This was accomplished by using only eight 4K dynamic chips tucked away in the corner of the board. Having the memory timing and refresh control on the same board as the CPU greatly simplifies dynamic RAM interfacing. Sockets for four 1702 PROMS are provided with two of them used by their console replacement debug monitor. A real-time clock, 16 bits of programmable I/O lines, and a serial teletype interface are also included on their 6.5" by 8 1/8" CPU card. The mail order price for the complete CPU board to hobbyists is $350 and is a limited time offer.

SPHERE Corp.
791 South 500 West #3
Bountiful, Utah 84010

Software is the other half of a complete system that is becoming available from multiple sources and at greatly reduced prices. MITS has just cut their 4K and 8K BASIC package prices to non-Altair owners. The 4K package now sells for $150, down from $350 and the 8K package is now $200, down from $500. This move clearly indicates MITS's intention to remain competitive in software as well as hardware. It probably also indicates that the development costs have been recovered. The high quality of their package is generally unquestioned, although some have commented on its speed.

Although there is a tremendous interest in BASIC, many hobbyists may prefer FORTRAN for solving really complex mathematical problems. Another advantage is that most commercial scientific application packages such as circuit analysis programs are written in FORTRAN. Additionally, FORTRAN provides much better argument passing capabilities to subroutines than does BASIC.

Mini-Software Inc. now has a FORTRAN compiler for the 8080 as well as a BASIC compiler. Both are two-pass compliers meaning that the source code is translated into a pseudo-assembly language first which is read back to be translated into object code.

The two phases of the compiler require 14K bytes each and also rely on user supplied I/O routines which would occupy the remaining 2K on a 16K system. One shortcut taken for simplicity is a requirement that all subroutines be compiled along with the main program. These sizes provide enough table area to compile programs as large as 1000 statements. An additional 8K for the compiler's tables would more than double the maximum program size.

The run-time package is 6K and the generated code is quite compact (GOTO 25, 2 bytes; IF(A.EQ.B)GOTO 15, 7 bytes) so it would be possible to run a non-trivial compiled FORTRAN program on an 8K machine. Maximum object code size is 32K and maximum data area size is 16K although these limits may be expanded with a slight reduction in speed. The object code is interpreted by the run-time routines rather than being executed directly.

Total compilation speed is about 30 statements per minute exclusive of I/O time which would be substantial without digital cassette or floppy disk. This is not bad considering that the compiler itself is written in FORTRAN and what is run on the 8080 is actually a compiled version of the compiler. The arithmetic package is a little strange. Integers are 40 bits (5 bytes) long and floating point numbers are 48 bits long. This affords roughly 11 digits of accuracy rather than the more common 7 digits. Floating add takes 2.5 Ms, floating multiply takes 24 MS, and floating divide takes 17 MS. At the expense of 5 significant bits, the multiply speed may be doubled.

For the most part the compiler accepts ANSI standard FORTRAN. Most restrictions are minor hassles regarding reserved words and the placement of blanks. The two major restrictions are no three dimensional arrays and no quoted character strings in FORMAT statements.

The complete FORTRAN system sells for $350 and BASIC sells for $400. Other major software products are an 8080 macro cross assembler for $100 and a FORTRAN cross compiler for $200. These cross products allow programs for the 8080 to be assembled or compiled on a large machine and run on a small 8080 system. A 50% discount is extended to non-commercial hobbyist users who prepay their orders.

Mini-Software
P.O. Box 7438
Alexandria, VA 22307

AN IMP-16 MICROCOMPUTER SYSTEM PART 1 by Hal Chamberlin

The IMP-16 microprocessor chip set made by National Semiconductor is perhaps the best microprocessor to use if a high performance, flexible hobby computer system is desired. Much emphasis has been given to the Intel 8008/8080 in the past and very little has been written on the IMP-16 even though it has been around quite awhile and is just as available as the 8080. There are many reasons for this and I hope that this series of articles will negate some of them and increase hobbyist awareness of this high-performance alternative.

Before continuing, a word should be said about National's PACE microprocessor. PACE is basically a one chip (instead of five) IMP-16. Improvements offered by the PACE are provisions for unlimited stack depth, five levels of vectored interrupt built-in, and reduced support IC package count. Sacrifices made include slower speed (2us state time instead of 1.4us), non-availability of extended instruction set CROM's and reduced flexibility or omission of some of the instructions. Nevertheless PACE is an excellent microprocessor that should be considered when an 8 bit machine is not enough but the full-bore system to be discribed is not quite necessary. As far as the hobbyist is concerned, PACE is about as far along as the Motorola 6800, good availability can be expected toward the end of this year. Conversely, the IMP-16 will not be discontinued any time soon, in fact, a bipolar chip set equivalent has been rumored.

Since adequate coverage of this topic will require a lot of space, it will be presented in four consecutive parts. This part will describe the system features and fundamental operation concepts. Hopefully , there will be enough detail to enable the reader to decide if this is the system for him. Part 2 will detail the system bus controller and timing generator. Logic diagrams of this portion of the machine will also be published. Part 3 will deal with the microprocessor chips themselves and the interface to the system bus described in Part 2. Again complete diagrams will be given. Part 4 will conclude the series with a presentation of an 8K by 16 dynamic memory board and a general discussion of I/O device interfacing with examples.

This implementation of the IMP-16 is original as far as can be determined. The CPU and memory design is part of a computerized music synthesis system being built by the author. Little circuitry was borrrowed from National's implementation, the IMP-16C. Weird or non-standard IC's were avoided as much as possible. Those that remain are listed in Appendix 1 along with the quantity required so that orders can be placed early.

Fundamentally, all system components communicate over a set of 44 lines which, for lack of a better name, will be called the PUNIBUS (Processor UNIfied BUS). This bus structure has much of the power found in Digital Equipment Company's UNIBUS used in PDP-11 computers but is simpler to interface to and is synchronous, having a definite cycle time of 700 ns. Another major difference is that it is intended to be a backplane bus with a maximum length of two feet or so. The length restriction allows ordinary TTL gates and standard tri-state devices to be used for bus driving. Thus in a system it is necessary that at least a portion of each I/O device interface be on a board that plugs into the system backplane. In actual practice (details and examples in Part 4) several simple device controllers or a complete complex one can fit on a single board.

The bus cycle time of 700 ns was chosen to be compatible with the IMP-16 microcycle time of 1.4 us and with 4K dynamic RAM chips. The same timing generator is used for bus timing and CPU timing thus reducing circuitry and insuring synchronization between the CPU and the bus. Timing signals are provided on the bus for operation of all popular 22 pin 4K dynamic RAMS. The newer 18 pin and 16 pin 4K RAMS can also be supported by changing one of the two 8223 PROM's that are used to generate the timing signals. Alternatively, you can use those cheap 1103's and 5260's that are available everywhere. As designed, the bus controller has no provisions for waiting on slow memory (slower than 400 ns access). This was done to reduce parts count and because adequately fast dynamic RAM's cost less per bit than the slower static RAM's anyway. The only real problem created by the lack of a wait is the inability to use most eraseable PROMS for program storage.

At the beginning of every bus cycle a decision is made as to which device will get the bus cycle. The CPU, memory refresher, and up to 5 direct memory access devices can be supported. If more than one device is requesting, the cycle is awarded on the basis of priority. The normal arrangement gives the CPU highest priority, memory refresher lowest priority, and the 5 DMA's are in between the extremes. This arrangement of priorities has several interesting properties. Since the CPU cycle is synchronized with the bus cycle and it has the highest priority, the CPU never has to wait. Conversely, the CPU will not hog the bus either because it will use at most 25% of the available bus cycles. The memory refresher is connected so that it is always requesting cycles. If neither the CPU nor any DMA devices are requesting the bus, then the cycle goes to the refresher. Adequate refreshing is assured (at least 64 cycles every 2 milliseconds) if as little as 2.3% of cycles are unclaimed. This allows totally transparent dynamic memory operation with the CPU running full speed and over one million words per second aggregate DMA rate. The bus could support up to 4 IMP-16 CPU's simultaneously. One CPU would be the master which would never wait and which would control the bus while the others would have lower priorities and may have to wait.

Input/output devices are addressed as memory locations and are connected to the bus as if they were memory. All of the memory reference instructions in the IMP-16 set can therefore be used for I/O operations. The regular IMP-16 I/O instructions are not used. I/O devices are assigned to addresses FF00 to FF7F hex and the distinction between input and output is governed by whether a memory read or a memory write is executed by the program. An interesting side-effect is that any DMA device also has access to the I/O system.

A tabulation of the bus signals appears in Appendix 2. Note that data and addresses are time multiplexed over a single set of 16 lines. Note also that timing signals are provided to identify what is on the bus or what should be gated onto the bus. The multiplexed bus not only saves connections but also saves IC's since fewer bus drivers and receivers are needed. Memory boards using 4K RAM's do not require address registers because they are built into the RAM chips. I/O device address decoding circuits will however require the use of a flip-flop to remember that it was addressed after the address disappears from the bus. Regulated supply voltages of +5, +15, and -15 are provided on the bus. Virtually any digital or linear IC can be operated from these voltages with perhaps a zener diode added. Two lines are reserved for +5 and three for ground to insure a noise-free +5 supply and ground. Ten amps from the +5 volt supply and 2 amps each from the 15 volt supplies should be enough to power a medium sized system with 16K of memory and 150 chips of I/O interfacing.

The IMP-16 chip set has two levels of interrupt built in. The lower level can be masked off and is called the I/O or general interrupt. The interrupt request bus line is connected to this interrupt. The higher level cannot be disabled and is called the control panel interrupt. It is normally connected to a console interrupt button and causes a return to the system monitor.

Built into the CPU board are two sockets for 8223 bipolar PROM's. This gives 32 words of permanent storage at locations FFE0 through FFFF which is adequate for a bootstrap loader. Whenever the power is turned on or the console reset button is actuated, the CPU clears its registers and branches to location FFFE in the loader. The loader program then reads in the operating system software from an input device such as a paper tape reader, cassette, floppy disk, eraseable ROM or even (shudder) a bank of switches or keyboard. The floppy disk is particularly convenient, start-up or recovery would require less than a second.

The operator's control panel consists of three push-buttons and two lamps. One button is the system reset button, one is used to restart the system after a HALT instruction, and the last is the control panel interrupt button. The lamps monitor the running and interrupt enable status of the CPU. Effective communication with the machine then requires a debug-monitor software package, an ASCII keyboard, and a display or teleprinter; items that should be present in any high-performance system. It is interesting to note that even in National's complete microcomputer system, the IMP-16L, the extensive console provided is implemented with software in a ROM.

One of the major attractions of the IMP-16 over other microprocessors is the availability of extended instruction set CROM's. One of these is called simply the "extended instruction set CROM". Included are 15 by 16 bit multiply, 31 by 16 bit divide (both of which take an average of 150 microseconds), 32 bit add and subtract, byte manipulation using byte memory addressing, and a shift and count instruction, potentially useful for floating point. The microprogrammed multiply and divide routines can be corrected to obtain the more common 16 X 16 and 32/16 operations with a few additional regular instructions. Another CROM which has just been announced is called the "POWER I/O" CROM. Included are high-speed (approx. 90K words/sec) microprogrammed block I/O transfer instructions, a block move from one area of memory to another, masked memory search, and machine status save/restore in a microprogram controlled stack in memory. One possible drawback to using these instructions on long blocks of data is that interrupts cannot be recognized until they are finished resulting in possible millisecond long interrupt latencies. Up to 4 CROM's may be used in a system. Others will doubtless come out as the masking charge is about the same as for a ROM and National has a microprogram prototyping kit available. Of particular interest, of course, would be a floating point CROM. (A "Power Math" CROM is rumored which supposedly handles all of the time-consuming functions of floating point software.)

Backing the IMP-16 is a substantial body of essential software. Although National asks $200 for a package including source listing and object tapes, the material is not copyrighted. We understand that the $200 pays mostly for "field support" functions. National asks only that they not be "bugged" if the software was not bought from them. TCH is punching source decks from the listings for modification. The modified software will be available at the end of this series. The code is well organized and thoroughly commented which makes understanding and further modification much easier. Input/output is generally done in only one subroutine thus alteration for different I/O schemes is easily accomplished.

The assembler requires 4K words to load and has many features not often found in micro- or even minicomputer assemblers. Conditional assembly, local symbols, relocatable or absolute code generation are all provided. Symbols require three or four words so an 8K system would allow about 1150 symbols. The relocating, linking loader allows each module of a massive program to be assembled seperately. When the entire program is to be run, it links the segments together and relocates them so that no memory "holes" appear between segments. The editor is essential when preparing large programs unless you have a keypunch and card reader. It works like the typical minicomputer editor in that a source file and keyboard commands are accepted as input and a destination file is produced as output. A memory buffer is used to allow random access to a portion of the text for editing. The debug program is quite extensive and is just the thing for replacing the typical computer console. It offers memory search, memory move and fill, and up to 5 separate breakpoints for program tracing. Many other typical functions are provided and the command structure makes expansion into a full scale operating system feasible. The CPU diagnostic program is very elaborate and very thorough. If it will run, you are assurred of a properly functioning system. If not, the error analysis usually pin-points the problem.

And last but not least are physical and packaging considerations. The board size used in the music system from which this design was taken is 8" by 8" with a 100 contact edge finger similar to that used in the Altair. The 64 square inches of space available is sufficient to place the CPU, bus controller, and bootstrap ROM on the same board. In the memory department there is enough space for 8K words using 22 pin 4K RAM's or probably 16K words using 16 or 18 pin 4K RAM's. One could also expect to fit a floppy disk controller or display generator on a single board. Using the 100 contact edge connector, up to 54 uncommitted pins are available for external world connections to each board. The mother board for the system is built in sections of 6 sockets each. The socket patterns are such that wire-wrap edge sockets will fit with 1.25 inches between boards. The wide spacing allows effective use of wire-wrap boards using even long pin, high profile sockets. A card file for standard rack mounting will hold two sections of the mother board for a total of 12 card positions, adequate for all but the largest systems.

Existing boards for the system include a wire-wrap board and a memory board. The wire-wrap board has 24 14-pin patterns, 44 16-pin patterns, and 10 24-pin patterns. Two pair of the 24 patterns can be combined to hold one 40 pin DIP each. One big advantage in using the board is the power and ground planes, bypass capacitor pads, and lands to the power and ground pins of the IC's. The memory board has a capacity of 8K 16 bit words using popular 4K RAM's such as the TMS-4030. It is complete and just plugs into the PUNIBUS. Jumpers select the block of 8K assigned to each board. A system is already operating at TCH and is essentially bug free. If you just can't wait for the next three parts to be published, blueprints of the CPU and memory board are available now for $4.00 but without the descriptive writeups. The memory board, wirewrap boards for CPU and interface construction, backplane boards and a card file with 12 card capacity are also available. Write the author in care of TCH for details.

APPENDIX 1

Integrated circuit list for IMP-16 CPU card.
* Denotes less common item. ** A real pain.

QUAN	TYPE
2	7400
* 1	74LS00
2	7402
2	7404
* 1	74LS04
* 2	74S04
1	7410
1	7413
* 1	74LS20
* 1	74LS21
1	7432
1	7442
* 1	74LS54 (Needed only for extended CROM)
4	7474
1	7493
2	74125 or 8093
* 1	74148
9	74173 or 8551
1	74174
* 1	74S174
* 1	8097 or 8T97
2	8223 (Programmed with CPU timing)
2	8223 (Programmed with IPL routine)
**2	8334 (9334)
* 5	8833
* 2	MH0026

Integrated circuit list for 8K by 16 memory card.

QUAN	TYPE
3	7404
1	7437
1	7474
4	74173 or 8551
* 2	8098 or 8096
* 1	MH0026
1	741
32	TMS4030 or 2107A or 2107B

APPENDIX 2 PUNIBUS SIGNAL SUMMARY

PIN	SIGNAL NAME	DESCRIPTION
1	+5	Regulated power
2	GROUND	
3	+15	Regulated power
4	-15	Regulated power
5	BUS DATA 14	Data bus
6	BUS DATA 15	Data bus most significant bit
7	BUS DATA 12	Data bus
8	BUS DATA 13	Data bus
9	BUS DATA 10	Data bus
10	BUS DATA 11	Data bus
11	BUS DATA 8	Data bus
12	BUS DATA 9	Data bus
13	BUS DATA 6	Data bus
14	BUS DATA 7	Data bus
15	BUS DATA 4	Data bus
16	BUS DATA 5	Data bus
17	BUS DATA 2	Data bus
18	BUS DATA 3	Data bus
19	BUS DATA 0	Data bus least significant bit
20	BUS DATA 1	Data bus
21	BUS I/O ADDR	Low when data bus has value in range of X'FF00 - X'FF7F
22	GROUND	
23	BUS ADDR ENAB	High when controlling device should gate address onto bus
24	BUS CE	Chip enable clock for 4K RAM's
25	BUS WRITE ENAB	Strobes data from bus during memory write or output
26	BUS MDR STROBE	Strobes data output register on 8K memory board
27	BUS CLOCK	17.1459 mHz signal 2 X frequency of middle C
28	BUS DATA OUT ENAB	Gates data onto bus during memory read or input
29	BUS DATA OUT STB	Strobes data from bus during memory read or input
30	GROUND	
31	BUS DATA IN ENAB	Gates data onto bus during memory write or output
32	BUS INT REQ	Wire-or I/O interrupt request
33	BUS RESET	low during power-on or reset
34	BUS WRITE CYCLE REQ	Wire-or direct memory access write request
35	BUS GRANT 4	Binary code of DMA
36	BUS GRANT 2	device in control
37	BUS GRANT 1	of the bus
38	reserved	
39	reserved	
40	reserved	
41	BUS REQ 1	Highest priority
42	BUS REQ 2	Wire-or lines for requesting DMA
43	BUS REQ 3	cycles
44	BUS REQ 4	Grant code corresponds to request
45	BUS REQ 5	number
46	POWER OK	Pulled down by power supply when voltages are OK

DEAL - A CARD DEALING SUBROUTINE FOR THE 8008
by Steven Roberts

The Random Number Generator proposed by Jim Parker in TCH Volume 1, Number 5, provides the basis for a card dealing subroutine which, in turn, can provide the basis for a number of interesting games. This routine returns to the calling program an ASCII string naming the card dealt, and in register B, a six bit card code to simplify logical operations involving the card's identity. In addition, a card list is created which is used by the routine to verify that a newly chosen card has not been dealt since the last shuffle; this list and its associated pointer may be used by the calling program to determine the number and identity of cards dealt. When all 52 cards have been dealt (as determined by cardlist pointer = maximum cardlist address), the word "EMPTY" is placed in the ASCII buffer (CRDBUF) and returned to the caller. Shuffling is accomplished simply by resetting the list pointer. Upon doing so, the word "SHUFFLE" is placed in the ASCII buffer.

In addition to the card code in reg. B and the contents of the ASCII buffer, the user has access to additional information. The card code is found also in the memory location CRDCOD (which contains a 377 octal if all 52 cards have been dealt). POINTR yields the number of cards dealt, and they may be found, in card code format, in the 52 locations defined as CRDLST. The format of the returned ASCII string is shown by the sample run, and the organization of the CRDCOD byte is shown in the program listing.

The simplest use of DEAL is in the application used to generate the sample run; the program involves only a call of SHUFFL and then a call of DEAL and a print of CRDBUF for each card. Card games of any description could make use of the subroutine, however - BLKJAK and POKER are currently under development, and others will likely follow.

```
*       DEAL PROGRAM BY STEVEN ROBERTS
*       CALL SHUFFL TO SHUFFLE DECK
*       CALL DEAL TO GET EACH CARD
*       RETURNS A 10 CHARACTER ASCII STRING IN CRDBUF
*       AND A 6 BIT CARD CODE IN REG. B
*       BITS 2 - 3 SUIT CODE
*       BITS 4 - 7 RANK CODE

        ORG 0           PROGRAM ORIGIN

SHUFFL  LAI L(CRDLST)   POINTER VALUE FOR START OF CARD LIST
        SHL POINTR
        LMA             LOAD NEW POINTER VALUE
        SHL SUITS+38    POINT TO THE WORD "SHUFFLE"
        LEI 10
        LCI L(CRDBUF)
        JMP MOVLOP      GO MOVE IT TO CRDBUF

DEAL    CAL RAND        CALL RANDOM NUMBER GENERATOR
        NDI 077B        REMOVE BITS 6 AND 7
        LDA             SAVE THE REST
        NDI 017B        REMOVE BITS 4 AND 5 LEAVING CARD VALUE
        LCA             SAVE RESULT IN C
        SBI 015B        IS IT OUT OF RANGE?
        JFC DEAL        IF SO, REDEAL
        LAD             OTHERWISE, RESTORE THE 6 BIT CODE
        NDI 060B        REMOVE BITS 0 - 3, LEAVING SUIT CODE
        LDA             RETURN THAT TO D
        ORC             REASSEMBLE CARD CODE
        LBA             SAVE IT IN B
        SHL POINTR      ADDRESS CARD LIST POINTER
        LEM             POINTER TO E
        LAI L(CRDLST+52)  LOAD A WITH MAX POINTER VALUE
        CPE             IS POINTER AT MAX?
        JTZ EMPTY       IF SO, GO PUT "EMPTY" INTO CRDBUF
        SHL CRDLST      ADDRESS START OF CARD LIST FOR SCAN
SCAN    LAM             CARD TO A
        CPB             IS IT = TO NEW CARD CODE (IN B)?
        JTZ DEAL        IF SO, REDEAL
        LAL             OTHERWISE, MOVE CURRENT LIST ADDRESS TO A
        CPE             SAME AS POINTER?
        JTZ STORE       IF SO, CARD IS VALID (NOT DEALT BEFORE)
*                       GO STORE
        INL             OTHERWISE, INCREMENT ADDRESS AND CHECK
*                       ANOTHER
        JMP SCAN        LOOP UNTIL EITHER COMPARISON IS VALID
STORE   LMB             STORE NEW CARD, SINCE IT IS VALID
        INL             UPDATE POINTER
        LEL             SAVE IN E
        SHL POINTR      ADDRESS POINTER BUFFER
        LME             STORE IT
        LAI L(CARDS)    BASE ADDRESS OF ASCII CARD VALUES
        LLC             CARD VALUE TO C
        ADL             ADD TO BASE ADDRESS
        LLA             SAVE THAT IN L, NOW POINTING TO NUMERIC VALUE
        LAM             ASCII CARD VALUE TO A
        SHL CRDBUF      ADDRESS FIRST LOC. OF ASCII BUFFER
        LMA             STORE IT (A,2,3,....T,J,Q, OR K)
        INL             POINT TO SECIND LOCATION IN CRDBUF
        LMI ' '         CLEAR SECOND LOCATION
        LAD             SUIT CODE TO A
        RAR             ROTATE RIGHT
        SHL SUITS       ADDRESS START OF SUITS BUFFER
        ADL             ADD NEW SUIT CODE TO BASE ADDRESS
        LLA             PUT THAT IN L, NOW POINTING AT CORRECT SUIT
*                       NAME

MOVE    LEI 8           PREPARE FOR 8 ITERATIONS
        LCI L(CRDBUF+2) POINT WITH C TO SUIT FIELD IN CRDBUF
MOVLOP  LAM             PULL CHARACTER FROM SUIT NAME
        LDL             SAVE ADDRESS IN D
        LLC             CRDBUF ADDRESS TO L
        LMA             STORE CHARACTER
        LCL             CRDBUF ADDRESS BACK TO C
        LLD             SUIT NAME ADDRESS BACK TO L
        INC             INCREMENT BOTH
        INL             ADDRESSES
        DCE             DECREMENT COUNTER REGISTER
        JFZ MOVLOP      IF NOT 0, GO BACK AND MOVE ANOTHER CHARACTER
        SHL CRDCOD      OTHERWISE, ADDRESS CARD CODE BYTE
        LMB             SAVE IT
        RET             RETURN TO CALLING PROGRAM

EMPTY   SHL CRDBUF      POINT TO CARD BUFFER
        LMI ' '         CLEAR THE CARD BYTE
        INL
        LMI ' '         CLEAR THE SECOND BYTE
        LBI 377B        "EMPTY" CODE TO CARD CODE
        SHL SUITS+32    POINT TO THE WORD "EMPTY"
        JMP MOVE        GO MOVE IT INTO CRDBUF, RETURNING WHEN DONE

RAND                    RANDOM NUMBER SUBROUTINE FROM TCH # 5

        ORG 400B        ALL DATA IN SAME PAGE

CRDLST  DST 52          LIST OF CARDS DEALT IN 6 BIT CODE
SUITS   DEF 'HEARTS  SPADES  CLUBS   DIAMONDSEMPTY   SHUFFLE
CARDS   DEF 'A23456789TJQK'
CRDCOD  DST 1           CARD CODE RESULT OF DEAL
*                       BITS 2 - 3  SUITS  00=HEARTS, 01=SPADES
*                       10=CLUBS, 11=DIAMONDS
*                       BITS 4 - 7 RANK  0000=A, 0001=2, 0010=3
*                       0011=4, 0100=5, 0101=6, 0110=7, 0111=8,
*,                      1000=9, 1001=T, 1010=J, 1011=Q, 1100=K
CRDBUF  DST 10          ASCII BUFFER 10 CHARACTERS
POINTR  DST 1           POINTER TO LOCATION IN CRDLST
        END
```

SAMPLE RUN OUTPUT WITH ADDED CALLING ROUTINE AND PRINT ROUTINE

```
  SHUFFLE
8 DIAMONDS
8 HEARTS
4 HEARTS
4 SPADES
J DIAMONDS
4 DIAMONDS
9 SPADES
6 DIAMONDS
T SPADES
A SPADES
A CLUBS
8 CLUBS
7 CLUBS
K HEARTS
2 CLUBS
Q SPADES
9 DIAMONDS
3 HEARTS
6 HEARTS
T DIAMONDS
5 CLUBS
5 SPADES
3 CLUBS
A HEARTS
Q CLUBS
6 SPADES
7 HEARTS
K DIAMONDS
Q DIAMONDS
5 HEARTS
J SPADES
K SPADES
J HEARTS
6 CLUBS
T HEARTS
Q HEARTS
3 DIAMONDS
9 CLUBS
5 DIAMONDS
K CLUBS
9 HEARTS
3 SPADES
2 HEARTS
2 DIAMONDS
7 DIAMONDS
J CLUBS
8 SPADES
A DIAMONDS
4 CLUBS
T CLUBS
2 SPADES
7 SPADES
  EMPTY
```

WHAT THIS COUNTRY NEEDS IS A GOOD $20 MICROPROCESSOR!

Well, it has one! The MOS technology (NOT Mostek) MCS6501 and MCS6502 are out, cost only $20 and $25 each respectively in quantities of one, and are being shipped to anybody who wants one. The best part is that the $20 price is not a limited time come-on, it is the true, factory direct, single quantity price. In larger quantities they cost less! Better yet, the chip doesn't limp along like an 8008, its speed and programming ease equals or surpasses chips such as the 8080 and Motorola 6800. With the $25 MCS6502, you can forget about clock drivers because the chip has a built-in oscillator, just connect a 1mHz crystal or R-C and, go. These things are going to take over the world!

Availability is always a big problem with any new product but as far as we can tell, it won't be any problem with these chips. MOS Technology seems to be trying a bold new marketing technique - direct from the factory mail-order sales. Among our staff members, 5 of the chips have been orderd and received from four separate orders. Only one order was placed on letterhead, the other three were personal and one of those was handwritten! All four orders were acknowledged within 10 days and a shipping date was given (Sept. 20 in all cases). By Oct. 6 all 5 chips were in-hand along with some manuals that were ordered. The invoices give a clue to their unusually efficient handling of factory orders. At this time only 5 different items are offered (MCS6501 $20, MCS6502 $25, hardware manual $5, programming manual $5, cross assembler manual $4). The invoices already had these items and prices preprinted so preparing them was essentially a check-off operation. All that seems necessary is a check or money order for the exact total price and a legible listing of the items desired. We sincerely hope that this marketing plan proves successful and that some other manufacturers adopt it. We can do our part by confining information requests to manual orders.

The MCS6501 has been advertised as bus compatible with the Motorola MC6800 with a similar but more powerful instruction set. After looking over the detailed interface specifications we would have to say that the bus is similar to the MC6800 but unlikely to be compatible in any but the simplest systems. The instruction set would be better described as "heavily influenced by the 6800". The user should probably consider the MOS Technology chips as new and different and not try to compare them with the Motorola chip.

TCH will be working with the built-in oscillator version, the MCS6502 and recommends that readers do likewise to avoid clock driver problems. The remainder of this discussion is applicable directly to that chip although all software remarks apply equally to all chips in the family. Interfacing to memory and peripherals seems simple but some problems do arise that tend to dirty up an otherwise clean design. The chip has a READY line but it only works on read cycles; write cycles ignore it and proceed at full speed. This means that although slow erasable ROM's may be used, high speed RAM is needed unless the clock itself is slowed down. One solution (the one used by TCH) is to use 4K dynamic RAM's, the slowest of which are fast enough to avoid problems and cost only $6 to $12 each.

Believe it or not the MCS6502 has three "no connections" on its 40 lead DIP package, a sin punishable by 60 days of 8008 system design! One signal that could have been provided is a "read enable" which would distinguish between active read cycles that actually fetch data and passive ones that occur when internal processing is taking place such as adding a base address to an index register. Memory refreshing could then be done during the passive cycles without slowing the system down. As it is, every non-write cycle must be considered as an active read cycle. The other two no connections could have been connected to selected status bits such as interrupt enable and overflow.

On the other hand a couple of unusual signals are provided that increase flexibility. One, called SYNC, signals instruction fetch read cycles. This can be used to implement a single instruction console control if desired. A better use is in memory refresh and direct memory access circuitry since a write cycle (which cannot be delayed) will never immediately follow an instruction fetch cycle. Another interesting signal is called SET OVERFLOW which will set the overflow status bit when pulsed. The only statement about possible uses given by the manual is that it is reserved for use with future MCS650X family products. This probably means that an extended arithmetic chip is planned that would signal overflow through this line. If overflow status was available as an output, then serial I/O might be implemented with no additional hardware. One other feature that many people might consider a drawback is a non-tri-state address bus, i.e., the address lines from the chip are always on. In a system with much memory or I/O however the address lines will be buffered by TTL. Tri-state buffers and the READY line can be used to provide a tri-state address bus if desired. According to the manual, substantial chip area was saved on the address buffers thus lowering costs.

Other architectural details of the MCS 6502 are similar to those of the MC6800. Input/output is accomplished with the regular memory reference instructions as if the I/O device ports were in fact memory locations. The rich repertorie of memory reference instructions makes this an even more effective method of handling I/O operations. If I/O registers are wired so that they can be both read and written by the CPU, then the modify memory instructions (increment. ecrement, and shift memory) can also modify the I/O registers. This could be a powerful tool in handling high speed I/O devices such as digital cassettes or floppy disks, particularly in special purpose systems. A transfer vector containing three full addresses occupies the last six memory locations. One is the start address after a system reset and the other two are the service routine addresses for the regular and the non-maskable interrupts respectively. High memory should therefore be ROM so that system reset and console interrupt (using the non-maskable interrupt) will automatically jump to the debug program in ROM. As in the 6800, the first 256 bytes of memory are very easy to access and are termed the base page. Also, the stack pointer in the MOS Technology chips is only 8 bits long and can only point to locations in the second 256 bytes of memory, thus low memory should be the system RAM. Internal architectural improvements include a degree of "pipelining" which greatly reduces the number of cycles necessary to execute an instruction. Most instructions require only one more clock cycle than memory cycles for execution and some such as the immediates and jumps use no additonal cycles.

In some ways the MC6800 instruction set was trimmed down for the MCS6502 and in others it was beefed up or changed. The MC6800 has a total of eight bytes of internal registers whereas the MCS6502 has only six thus further reducing the chip area. One cut was in the stack pointer (16 bits to 8 bits) and the other cut was in working registers (two 8 bit accumulators and a 16 bit index to one 8 bit accumulator and two 8 bit indexes). It is this shifting around of registers and lengths that make the two machines very different when it finally comes to programming them.

Most programmers will have to make a major adjustment in their thinking about indexing in this machine. The roles of "base address" and "displacement" have been reversed from the norm seen in minicomputers and the Motorola chip. The indexed instructions on the MCS6502 supply a 16 bit base value as part of the instruction itself and then add the 8 bit index register contents to that base value to get the effective address. An immediate problem is seen in manipulating large tables (greater than 256 bytes) since manipulating the index registers alone is not enough to scan the whole table; the base value must be periodically changed. Fortunately, two indirect addressing modes are provided using pointers on the base page so that the base value in the instruction itself need not be modified for large table scans. Programming for small tables and short string moves however is a model of simplicity and efficiency.

There are several other points about the MCS6502 instruction set worth mentioning. Only two arithmetic instructions are provided, add with carry and subtract with borrow. In order to do a regular add or subtract it is necessary to clear the carry first with a one byte, two cycle instruction. If indeed only two arithmetic instructions were allowable then this was clearly a better choice than providing only single precision arithmetic capability on an 8 bit machine. There is a decimal mode bit in the processor status register which causes the arithmetic instructions to consider the operands as two digit BCD numbers rather than 8 bit binary numbers. Since the processor status is saved on the stack during interrupts, there is no problem in getting interrupted during decimal operations. However the interrupt service routine will have to issue a "clear decimal mode" instruction if the main program is expected to ever set it. There is one processor status bit unused in the MCS6502 which could have controlled whether or not the carry flag was included in the arithmetic instructions but according to the manual it is reserved for use in future family members. Jump conditions are somewhat limited as compared with the MC6800 but if speed is important it should be noted that the MCS6502 can do two conditional jumps in the time required for the MC6800 to do one although 4 bytes of memory would be used.

MOS Technology clearly intends to build a software compatible microprocessor family with the MCS6501 and MCS6502 as the first members. New products to be available soon are 3 different 28 pin microprocessors which are performance identical to the 40 pin versions except for maximum memory size (4K bytes) and bus control functions. These chips would be cheaper yet and should find their way into hobbyist peripheral controllers sooner than you think. (Imagine a printer controller using a more powerful microprocessor that the system it connects to!) Also available in November will be a 2mHz 6502 equivalent (twice as fast). Since the required memory access time will be 300NS or less for full speed operation, there should be renewed interest in 4K RAM's since they can easily go that fast. Future plans seem to call for a 16 bit chip in which all of the registers would be 16 bits long and the unused op-codes in the 8 bit machines would be for 16 bit operations. The unusual feature is that memory would still be 8 bits wide allowing the new chip to be a plug in replacement.

In summary, the speed and power of the MOS Technology chips far outstrips their lowball price. The manuals are excellent and should be obtained by anyone interested in further details.

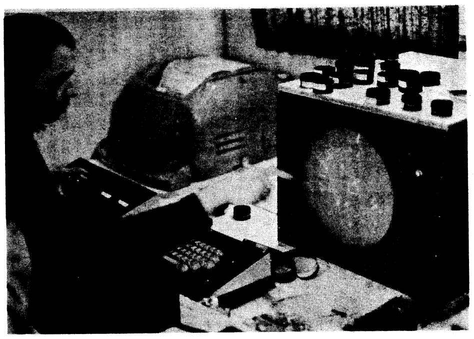
MOS Technology
Valley Forge Corporate Center
950 Rittenhouse Rd.
Norristown, PA 19401

PHOTOS

CHAMBERLIN ON THE ROAD WITH TCH

Somebody is taking pictures of what?
SHELBY, N.C. HAMFEST

And this is how its done.
SHELBY, N.C. HAMFEST



And this is the intersection
where we got off of the interstate.
DAYTON, OHIO HAMFEST

And what else did you plug in backwards?
ARRL CONVENTION, RESTON, VIRGINIA

Several of our readers pointed out some errors in the 8080 version of the cassette ROM. Two of the errors prevent the program from working at all. The corrections are listed below:

LOCATION	WAS	SHOULD BE	COMMENTS
177512	167	161	MOV M,C instead of MOV M,A
177516	042	107	Jump address was wrong
177661	302	312	Read data bit on trailing
177670	312	302	edge of clock

All of the erasable ROM's we have programmed have these corrections. There are no known errors in the 8008 version of the program.

There is an operational problem we have experienced that audio cassette users should be aware of. Most of the cheaper recorders have a very small capstan diameter and rather soft pressure roller. The higher RPM of the small capstan cuts down on required flywheel size and the soft rubber reduces the need for precision mechanical alignment. If the recorder is allowed to sit for a long period of time with the mechanism engaged but the motor off, the capstan can put a dent into the rubber and a crease into the tape. The minor speed wiggle from a dented pressure roller is of no consequence but the crease in the tape can be severe enough to completely lift the tape from the head for several bit times. Since the tape normally stops in the gaps between data blocks, any creases will not be discovered until the tape is erased and reused or data is written on the flip side. Thinner tape should be less subject to creasing but causes other problems with inexpensive recorders. The only real solution is to manually disengage the recorder if it will not be addressed for awhile.

Although TCH has exhausted its supply of 6 volt 3PDT relays for the cassette boards, they are a perfectly standard Potter & Brumfield part. An exact replacement is P&B number KA-14 DY with a typical unit quantity price of $4.50. Similar relays with coil voltages up to 40 volts can be used. The relay supply voltage (+) should be connected to pin 12 of the I/O connection socket on the cassette board.

CLASSIFIED ADS

There is no charge for classified ads in TCH but they must pertain to the general area of computers or electronics, and must be submitted by a non-commercial subscriber. Feel free to use ads to buy, sell, trade, seek information, announce meetings or for any other worthwhile purpose. Please submit ads on separate sheets of paper and include name and address and/or phone number. Please keep length down to 10 lines or less.

FOR SALE: Wanlass Model 200-IC-1 power supply. 3.6 - 6.3 VDC, 25 Amp output, .25% line and load regulation with crowbar protection. Original cost $250. Also Heath Impscope Model EV-3. Both excellent condition. Reasonable offers accepted. David Milhouse, 2823 Griffa Ave., Columbus, Indiana 47201

WANTED: Altair got you down? Sell me your used 8800 and recover some of your hard earned cash. Get back into computing when the smoke clears. Ken Hopper, 4201 S. Bowman Ave., Indiannapolis, Indiana. 307/787-8661

WANTED: Any information on the COQAR System 4. I have some tapes and would be willing to copy and swap them. W. B. Llewellyn, 3523 N. Druid Hills Rd., Decatur, GA 30033

FOR SALE: R.P.C. 4000 Computer system. 8K by 32 bit drum processor with hardware multiply and divide, 60 CPS card reader, 30 CPS card punch, 10 CPS parallel entry teletype. Software includes FORTRAN, PINT, RUNE, PERT-2, ACT IV, and an assembler. All hardware documentation except assembler. $800 at Wilmington Del. You pick up or ship. Also 8K by 32 bit magnetic drum memory, $150. MITS "ALTAIR" computer equipment assembled for 55% of MITS cost. Paul Gumerman, 101 Stonecrop Rd., Wilmington, Del. 19810

WANTED: IMP-16 Assembler. Jim Gaudreault, 7909 14th Ave., Hyattsville, MD 20783. 301/434-2482

FOR SALE: Used 1101A. I have 48 used 1101A's. Each one has been tested and working. All are ceramic DIP. Will sell as a lot for $36 on a first come basis. These would be ideal for a Mark-8 project. Roberto Denis, 11080 NW 39Th St. Coral Springs, FL 33065. 305/752-7067

SALE OR TRADE: HP-55 programable calculator with all accessories for a basic minicomputer (8008 - 8080 - IMP-16 etc.) Contact: Tony Demase, 8108 Westmoreland Avenue, Pittsburgh, PA 15218. Ph. 412/242-5780

POWER SUPPLY: I have a quantity of 5V 6A highly regulated power supplies taken from keyboard terminals. Schematics included and plans for obtaining -5V, -9V, and -12V. $25 plus postage on 15 pounds. Edward G. Runyan, 1146 Nirvana Rd., Santa Barbara, CA 93101.

WANTED: Two used MARK-8's in good working condition for educational use. Units should be enclosed in professional appearing case. Will give $165 each. Indicate prices desired for any accessories. Jordan Kreindler, PhD, 1363 Pine Ridge Rd., Montgomery, Ala. 36109

SERVICE: You want to save $100 on an 8800 but don't want to put it together yourself? Or maybe you need a parallel I/O card but don't want to spend more than $30. You wish you could get that vectored interrupt card with your $40. Well for heaven's sake write: Steve Witham, 168 Painter Road, Media, PA 19063

FOR SALE: GE card reader - 300 CPM. Used, good condition $75 - you pay shipping (about 50lb.) Also complete set of PC boards for Mark-8 (from Techniques) - CPU board, front panel and address latch board have most IC's installed, but no CPU chip - $35 for the set. Donald Bailey, 6 Jay Drive, Concord, NH 03301

FOR SALE: DIAN 9030 teleprinter (RS-232 compatible), used but recently factory-rebuilt and in perfect condition. Prints and transmits full ASCII code at 110, 150, or 300 baud, using 5x7 dot matrix, on standard computer paper from 5" to 15" wide. Why pay MITS $1500 for a TTY when you can get much more in a DIAN for $1300 . . . or make an offer! Call 404/483-4570 after 6:00. Bud Pass, 1454 Latta Lane, Conyers, GA 30207

ALTAIR 8800: SAVE OVER $200 Assembled, tested, and completely operational. Includes 1K RAM board with 256 words installed, complete documentation including news letters. Quality built by electrical engineer. ONLY $500 postpaid. D. Schreiter, 9032 Whitehaven Drive, St. Louis, MO 63123

SURPLUS SUMMARY

For you people in the Illinois area, Wilcox Enterprises has come up with some interesting goodies. Included are IBM input/output Selectrics Model 731 for $750, Dura Business Machines Model 10 (this machine includes a Selectric printer, paper tape reader, and paper tape punch) $390, Flexowriters $150 and $250 for the mono and dual case machines respectively, CDC Typetronic system components including paper tape readers and punches for $75, and receive only Model 33 teletypes for $375. The prices are not steals but they are good. The catch? These units must be picked up, they will not be shipped. Actually this is a good approach since you get to examine before purchase.

Wilcox Enterprises
25 W 178 - 39. Street
Naperville, Illinois 60540

Another source for few of a kind but very worthwhile items is a guy called Gary Coleman. Available items range from tape decks, to modems, to keyboards, to IC's. For a list send a SASE to:

Gary Coleman
14058 Superior Road, Apt. 8
Cleveland, OH 44116

Looking for a CHEEP printer? Here is one way to get one. Richard Page informs us that he has several unusual model 28 receive only teletypes. Unusual? Well they run 115 words per minute, have the fractions character set (have 1/4, 1/2, etc. in place of :, ;, (,), ?, and !), have sprocket feed platens, print 5 characters per inch (36 characters per line), and feed at 2 lines per inch. The other unusual thing is that this well known, rugged machine is being sold for $50. Furthermore an industrious person can purchase and install the parts to convert it back to a standard model 28. For further details contact:

Richard Page
223 East Howard Street
Pontiac, Illinois 61764
815/844-5550

Here is a tip in reverse. Les Veenstra of Action Technical Services has asked us to tell you folks that he is completely out of those rebuilt 33RO's. Apparently people receiving back issue sets are still writing him!

Gary Fishkin of Rochester, NY has informed us that he has some digital cassette decks for sale. They are International Computer Products DigiDeck model 63. He is selling the decks for $100 while they last. You can obtain a complete technical manual directly from ICP for $5 by writing to ICP, Box 34484, Dallas, Texas 75234. To order a deck write to:

Gary Fishkin
Box 349
25 Andrews Memorial Drive
Rochester, NY 14623

THE COMPUTER HOBBYIST
Box 295
Cary, NC 27511

ADDRESS CORRECTION REQUESTED

THE COMPUTER HOBBYIST

Number 9 February, 1976

TCH SUPER SIMPLE FLOPPY DISK INTERFACE Part 1.

As the title implies, TCH has developed a simple, inexpensive controller for floppy disk drives. In fact, it has been running in our 8008 system since December 1975. Like the cassette interface, graphics display, and other projects, the floppy disk controller will run on any kind of computer. This part 1 will describe the characteristics and operation of floppy disk drives in detail as well as give an accurate estimate of the costs involved. The experience gained in transferring 2300 subscriber records to floppy disk and then updating several hundred of them for address changes as well as experience with a graphic image storage and retrieval system convinces us that the design is bug-free and reliable. Nevertheless, circuit diagrams will be held until part 2, to be in issue #10.

Floppy disk systems are not expensive! At least not as expensive as the $1200 to $3000 tags on commercially available systems would lead one to believe. Here are the hard facts on the cost of floppy disk system components. The drive is the worst part. As will be detailed later, drives from the various manufacturers are nearly identical and this applies to the single quantity price tag too; $625 to $750. The first quantity price break is about $100 and occurs at 25 units for most manufacturers. In 100-up quantities, prices range from $400 to $450. This is an example of what standardization and competition can do. How about the controller? With TCH's interface and less than 400 bytes of software, the controller is at most $60 worth of components, circuit board included! Twenty-five packages of very mundane TTL, two 2102 memory chips, and a little software that fits neatly into two 1702's is all it takes. The power supply is not a hassle either. Most drives require +5 volts at about an amp, and +24 volts at 1.5 amps, loosely regulated. Some also need a negative voltage at 50 MA or so. Twenty dollars more worth of components should take care of the power supply. All floppy disk drives have attractive front bezels. Packaging can be as simple as sitting the unit on the table. Mounting in a panel or large box however is just a matter of cutting a rectangular hole in the box front. Floppy disks to keep the critter fed are typically $7.50 each. Minimum order quantities are either 5 or 10 disks. This is not bad considering that the 300K byte capacity of one disk is equivalent to two C-60 cassettes recorded on both sides with a TCH audio cassette interface.

What kind of applications do the characteristics of floppy disks make possible? An obvious one is maintaining lists of information that are subject to frequent change and which need to be printed periodically, such as address lists. Maintaining such a list on tape is a pain at best and certainly requires two drives. With a floppy, any record can be retrieved, updated, and stored again in a few seconds. Text editing, whether it be manuscripts, or programs can be readily handled. The 300K byte capacity of the disk allows massive program text files of 100 to 200 pages of assembly language code to be on one disk. Assembling such programs is very rapid due to the fast transfer rate. Of course keeping extensive program files for debugging and demonstration purposes is a natural application. All kinds of special purpose applications present themselves. For example, Jim parker has written a graphics operating system for use with a floppy. Files of images, parts of images, and character shapes are stored on the disk. New images may be created as a combination of stored images and new graphic input. The stored images can be moved around in the new image and in some cases expanded or shrunk also. The new images can likewise be stored on the disk for later recall or incorporation into other images. The same concepts could also be applied to electronic music composition. A practical small business accounting system virtually requires disk storage, especially to implement the automatic functions that really make such systems pay off. Game applications where the program "learns" by maintaining a file of previous experience are almost endless. In each of these application suggestions, the fast random access and update capability of the floppy disk is central with the other characteristics being of less importance.

That should be enough for vague generalities, lets look now at the individual system components and some of their typical specifications. First we have the disk itself, often called a "diskette" which is an IBM term. The disk is a thin circular sheet of mylar (actually die cut from a wide roll of computer tape) coated on one or both sides with magnetic oxide material (see figure 1). The outside diameter is 7.88 inches and a 1.5 inch round hole is punched in the center for slipping over the drive spindle. An additional small hole is punched 1.5 inches from the center and is called the index hole. For environmental protection, this flimsy disk is enclosed in a sealed, semi-rigid envelope lined on the inside with a felt-like material. The envelope is 8 inches square and has a center punchout so the drive spindle can reach the disk. There is also a radial slot to allow the recording head to reach the disk, and a small hole to allow a lamp-photocell assembly to see the index hole as it rotates by. Finally, an optional write protect hole is punched into one edge of the envelope. Like cassettes, the physical dimensions of floppy disks are standardized, so all manufacturer's products are the same although price and quality could vary (they don't really, the only significant difference is purchasing hassle). There is one major variation however. The single index hole disk just described is the IBM standard but also available are disks which have 32 "sector holes" evenly spaced around the disk on the same circumference as the index hole (see figure 2). These are frequently called "hard sectored" diskettes because the extra holes define 32 distinct, physical sectors on the disk. Every major manufacturer except IBM makes both styles and sells them for the same price. TCH will be using the hard sectored variety for reasons that will be made clear later.

It presently costs manufacturers about $1.50 to make a diskette. Much of this cost is due to the "initializing" process where a predefined data pattern is written all over the disk. This pattern is required by systems using the one-hole diskettes but not by hard sectored systems even though the manufacturers write it on their 33 hole disks anyway. There is great potential for lower cost diskettes ($1.98 each) as usage and competition increases.

Next we have the diskette drive (see figure 3) The drives available from the 8 or so major manufacturers are all so much alike that any differences are usually confined to how the door closes and how fast the salesman talks. There are a few significant differences however which will be noted. First there is a door which opens to expose a narrow slot. The disk envelope is pushed all the way into the slot and the door is closed, trapping it there. Since the door is about all the operator sees, there has been considerable innovation in this area. Door action ranges from a simple hinge and handle to push-button latches which pop the diskette out like a toaster when released. A prime example of poor human engineering is the fact that the disk can be inserted into the drive in 8 different ways and only one of them works. When the door closes, cam action moves the disk against a cone shaped spindle which pokes through the large center hole. A spring loaded female cone then engages from the other side clamping the disk against the drive spindle. The spindle is driven at a constant speed of 360 RPM by a synchronous motor causing the disk itself to rotate inside the felt lined envelope. Except for one manufacturer, there is a noticable lack of innovation in the spindle drive. Typically, a large wheel fixed to the spindle is belt driven by an 1800 RPM motor mounted in the corner of the drive. The belt is best described as a precision rubber band. Several drive manufacturers even use the same identical Bodine 4-pole, split capacitor, sync motor! Pertec has taken a different approach on the FD-400. They use a 24-pole 3-phase motor driven by a transistor inverter powered from the 24 volt DC supply. The motor runs at 360 RPM, thus it drives the spindle directly. The only drawback is a higher acoustic noise level.

The head is on a carriage that can be moved radially, in towards the center or out towards the edge of the disk by a lead screw driven by a stepping motor. This carriage can be positioned at any one of 77 discrete points spaced about .02" apart by stepping the motor one direction or the other. Seventy-seven concentric tracks are thus defined on the disk surface with track 0 being the outermost and track 76 the innermost. A sensor, either a microswitch or lamp-photocell detects when the head is positioned at track 0. A step count must be kept to ascertain the other 76 possible positions. Some drives have two mechanical stops that prevent the head from being positioned beyond the allowable range while others have only one or none. Stepping speed is really the only parameter where there is significant competition. Many drives can step 100 times per second but several advertise a rate of 166 steps per second. Sycor beats them all at 400 steps per second for a maximum positioning time of 190 milliseconds.

CONTINUED ON 4

LETTERS TO THE EDITOR

Gentlemen:

In regards to TCH's IMP-16 CPU, my thanks for a much needed construction article. Several friends and I are convinced this is the CPU we have been waiting for and have started building one.

After spending many hours searching supplier catalogs for hard to find IC's; looking for the lowest cost source on other IC's and parts via price breaks, etc., we decided to compile a list of parts by supplier. This list may be of benefit to other hobbyists desiring to build the TCH IMP-16 CPU. I would be glad to send a copy to other hobbyists for $1.00 to cover postage and copying costs.

Also, we would be interested in communicating with other hobbyists constructing the TCH IMP-16 CPU about additional hardware and software plans. It would seem that a TCH IMP-16 User's Group would be a real possibility.

Thanks again for leading the way with what promises to be a superb series of construction articles for the home computer hobbyist.

Fred Holmes
101 Brookbend Ct.
Maudlin, SC 29662

It is easy to spot the enthusiasm in letters such as the above. Many thanks for the letters of encouragement from all quarters on our construction projects. Send parts list orders directly to Mr. Holmes at the above address.

Gentlemen:

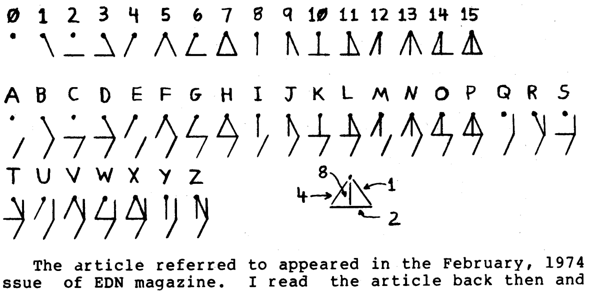
Your editorial in the Vol. 1 No. 8 issue deals with octal VS hexadecimal. Your decision is to adopt a "split octal" notation. The decision is based upon accommodation to the Intel microprocessors.

Several years ago we faced the same call for decision. We went base-16. Reason--the square roots of round numbers are round numbers. Also the fourth roots. For our Altair computer we have changed all instructions over to base 16. This makes the instruction set difficult to use--as you have noted. But this instruction set is transitory. There is little doubt in this quarter that the base-16 system is the system of the future and that the octal system will fade like an old soldier.

Appended is a photo of our Altair with the front panel altered to accommodate the base-16 Computer Compatible Digit. Also a copy of the instruction set. Also a reprint of an EDN article discussing the system.

Incidentally we have interfaced a combinational keyboard to the Altair. With eight keys we send in the operational codes directly. No encoding. Cuts costs drastically. It took perhaps a day to learn the keyboard. We also have the keyboard interfaced to a printer for Computer Compatible Characters. On the next page appear the digits of the base 16 numbering system, followed by the lower case alphabet, followed by some text.

R. O. Whitaker



The article referred to appeared in the February, 1974 issue of EDN magazine. I read the article back then and was frankly intrigued with the ideas presented. However two years of time and experience have taken their toll. First, the advent of the $10 calculator proved that 7-segment displays and associated decoding can be done economically. Second, with the present configuration of the digit, there could be great confusion between 9 (ᐱ) and 12 (ᐱ) when handprinted by a sloppy person. One only has to read as many scrawled addresses as we do to comprehend that problem. Finally, the economics of change and the general conservatism of people are tremendous obstacles to overcome. Look at the acceptance of the Dvorak typewriter keyboard (invented in 1932 by Dr. August, proven in tests to be 50 to 100% faster than conventional keyboard with fewer errors) for one example. Also look at the ongoing metric conversion effort in this country. It is barely started, will take 10 to 20 years to complete, will ultimately "cost" billions, and, sadly, had to be administered by the government.

EDITORIAL

By now nearly everyone should be aware of the results of the BYTE Standards Conference that was held in Kansas last November. We won't attempt to explain in detail the recording method agreed upon since this information will be available in Byte and Popular Electronics. However since Hal Chamberlin and Richard Smith were there for the whole meeting and took an active part in the proceedings we can give a "behind the scenes" report on just what transpired and how.

First, a rundown on the participants. Two groups were very conspicuous by their abscence. Scelbi Computer Consulting wasn't there. If they had been, they probably would have influenced the discussions considerably since they were making audio cassette systems long before the rest of us had thought of the concept. Another absent organization was The Digital Group of Denver, only about 300 miles from Kansas. As most of us know, they have been quite successful to date in penetrating the audio cassette interface market. Just before the conference they announced a speed tolerant version of their interface which would certainly have been of interest to the other participants. HAL Communications, well known for TV Radioteletype gear, came but left upon realizing the mismatch between their equipment offerings and the purpose of the conference. SPHERE and Popular Electronics left after the first two sessions (there were 4 in all). Les Solomon of P.E. was vocal in his support of the HIT format but unfortuantely could not answer the numerous technical questions flying about. He was particularly concerned about tape and recorder certification procedures. Other active groups in attendance who lasted through all 4 sessions were Godbout, Processor Technology, Southwest Technical Products, Lee Feldenstein (Pennywhistle 103), Ed Roberts of MITS, and Harold Mauch of Pronetics. MITS did not really try to infulence the proceedings but did occasionally bring up terms like "defacto standard" in relation to their recording method. Harold Mauch probably had the greatest influence since he has participated in digital cassette standards activity and has had much experience in the design of tape interfaces for industrial applications. We feel that TCH had a significant influence in issues such as record format and motor control.

Oddly enough, Carl Helmers, editor of BYTE was not there to run the show. Instead, Virginia and Manfred Peschke, the publishers were. Virginia essentially took notes and Manfred coordinated the sessions. While having considerable programming experience with large-scale computer equipment, Manfred had some difficulty with all of the technical terms being thrown around. In a way this was good since it forced the participants to be clear and unambiguous in their explanations and arguments. Incidentally, the conference was anything but a fun and games exercise. It was straight work all day Friday and Saturday morning with very little time off for socializing.

In the first session an organized engineering approach was taken to the problem at hand. Goals were stated and intended applications of the standard were established. Then the characteristics of low cost recorders were discussed and listed. Finally, a workable set of performance specifications was established. Up to this point, nothing had been said about any particular recording

THE COMPUTER HOBBYIST
Founded October, 1974

Stephen C. Stallings - Managing Editor
Hal Chamberlin - Contributing editor
Jim Parker - Contributor
Clyde Butler - Photographer
Richard Smith - Programming consultant

THE COMPUTER HOBBYIST is published at intervals approaching monthly near Raleigh, N.C. The subscription rate for 12 issues is $6.00 for persons in the U.S.A. and U.S. possessions. For Canada and Mexico the rate is $7.50 for 12 issues. Subscriptions in all other countries are $11.50 for surface mail and $16.00 for air mail. All back issues are available and cost 65 cents each for persons in the U.S.A., U.S. possessions, Canada, or Mexico. For all other countries back issues cost $1.00 each by surface mail or $1.50 each by air mail. Remittances from countries other than the U.S.A should be in the form of a bank or postal money order.

THE COMPUTER HOBBYIST is seeking paid contributors. Material to be submitted should be typed or neatly written and must not appear to be soliciting business for any firm. For details, please inquire.

Advertizing space is available to interested firms. please inquire for rates and formats.

All correspondence should be addressed to:

THE COMPUTER HOBBYIST
Box 295
Cary, NC 27511

method or existing system. The result of the morning session was a fairly complete body of engineering data needed to develop a recording method to meet the stated goals.

After lunch, each of the participants was asked to explain his existing or proposed system in about 5 minutes. TCH was first and there were approximately 8 people in all who spoke. Three of the formats were so similar that they could be considered identical (all variations of the "Lancaster format" finally adopted). The others were different and familiar to most of us. After some random discussion, someone pointed out that the formats which are based on the 11 unit teletype code (standard asynchronous communication) would not need a computer to encode and decode them. Within a matter of minutes, a vote was taken and the decision made that the standard would be asynchronous, 8 bit character oriented data. This of course essentially eliminated the HIT format and the TCH format from further consideration. We also realized that because of the inherent speed sensitivity of the asynchronous format and the fact that the Lancaster method is the only reasonable way to recover a speed tracking clock for a UART that the eventual outcome would in fact be the Lancaster method.

The first part of the evening session was devoted to other possible modulation methods that would be speed tolerant and still give an 11 unit asynchronous character structure. TCH still advocated the single pulse/double pulse method of encoding zeroes and ones but in the end endorsed Harold Mauch's position paper detailing the Lancaster method. The remainder of the evening and the Saturday morning session was spent formalizing the results and discussing other issues such as preambles, headers, checksums, interblock gap lengths and garbage rejection during gaps. Unfortunately due to lack of time and energy no decisions were reached on most of these data formatting issues. Thus, at this time, the BYTE standard consists of a method of putting 8-bit bytes on tape; essentially a paper tape analog. As a result, there is still some work to be done before hobbyists can freely exchange anything except ASCII text.

In interpreting the results of the conference it is important to remember that it is an interchange standard and not necessarily the only acceptable or even the best way to store data. After all, paper tape is the interchange standard for minicomputers but most all serious data storage takes place on some other medium. The BYTE standard does not seem to have slowed the development and introduction of new cassette interfaces. However, compatibility with the BYTE standard, perhaps by flipping a switch, will become an important interface attribute.

CLUBS ETC.

Each issue we will publish pertinent information about new computer clubs and significant changes in the status of old ones.

LONG ISLAND, NEW YORK
Long Island Computer Association
Contact - Jerry Harrison, Chairman
36 Irene Lane E.
Plainview, NY 11803

SOUTH MIAMI, FLORIDA
South Florida Computer Group
Contact - Terry Williamson
P.O. Box 430852
So. Miami, FL 33143
305/271-9909

WISCONSIN
Wisconsin Area Computer Hobbyists
Contact - Don Stevens
P.O. Box 159
Sheboygan Falls, Wis. 53085

DETROIT, MICHIGAN
Tenative formation (Dearborn, Dearborne Heights, Detroit)
Contact - Robert Tater
8476 Nightingdale
Dearborn Heights, MI 48127
313/279-0099

PITTSBURGH, PENNSYLVANIA
Pittsburgh Area Computer Club
400 Smithfield St.
Pittsburgh, PA 15222
Contact - Eric S. Liber
1156 Pennsbury Blvd. N.
Pittsburgh, PA 15222
Officers: Eric S. Liber, President
Fred Kitman, Secretary-treasurer

NORTH READING, MASSACHUSETTES
Alcove Computer Club
Contact: John P. Vulla, President
230 Main St.
North Reading, Mass. 01864

NEW YORK, NEW YORK
N.Y. City Micro Hobbyist Group
Contact - Robert Schwartz
1E, 375 Riverside Dr.
New York, NY 10025

MIAMI, FLORIDA
Miami Computer Club
Contact - John Lynn
13431 SW 79TH ST.
Miami, FL
305/271-2805

It is about time that the Raleigh, NC area had a hobby computer club. With this aim TCH is sponsering a "seed" meeting. It is not our intent to run a club but merely to start one, so come with organizational ideas in hand. The meeting will take place at the Plantation Inn on US highway # 1 approximately 2 miles north of Raleigh. Please park and enter at the rear of the main building. The get-together will start at 1:00 PM Sunday, March 14 and will last until folks get tired. For the sake of those who want to find out what hobby computing is about TCH will have both its 8008 system and an Altair on demo. The graphics display, cassette interface, and floppy disk will also be demonstrated.

NOTES ON TCH

Starting with this issue all mailing labels will contain your subscription number and the issue with which your subscription will expire. This is possible because all subscription records are now maintained on floppy disk. In the past records had been a combination of flip-file cards, two paper tape formats, and cassette tape. Now that the disk system is implemented all address changes to date have been verified and entered. If your newsletter is incorrectly addressed and you have not changed address in the last two weeks, then we missed your change of address, please resubmit it.

Effective immediately rates for back issues and foreign subscriptions are increasing. As you might suspect this is partially due to increased postal rates. The other factor is that TCH was originally planned for 1 ounce editions, but recent editions have consistantly been larger. Regular subscriptions in the U.S.A will not increase bacause they are mailed third class where-as the back issues and foreign cannot be. This change affects only remittances from this date foreward. Items already paid for will be shipped at the original price. New rates for backissues and foreign are shown below:

Backissues in U.S.A., Canada, Mexico	$ .65
Backissues foreign, surface mail	$1.00
Backissues foreign, air mail	$1.50
12 issue subscription U.S.A.	$6.00
12 issue subscription Canada, Mexico	$7.50
12 issue subscription foreign, surface	$11.50
12 issue subscription foreign, air	$16.00

Cassette interface boards, 8080 wirewrap boards, and regulator kits are still being shipped. Some orders for the 8080 wirewrap board have been delayed however. This is because TCH was faced with back orders from both our vendors on the heatsinks for the regulator kits. Hopefully this is now cleared up. Also cassette ROM programming and IMP-16 documentation is still being offered.

TCH is still seeking articles from outside authors. So far only two articles have been submitted (one of them was published last issue), but many people have inquired about writing for TCH. Rather than send out a flock of letters to potential authors, the needed information will be presented right here.

What is TCH looking for? General interest articles about both software and hardware. General interest means that it is not about your personal unique system or device which no one else would care to duplicate. Also articles which primarily describe some company's product are not suitable for TCH. We will leave those to the big guys who have more space to fill. What does that leave? Plenty! Stories about the application of readily available devices such as joysticks, oscilloscopes, TV sets, electronic music devices, tape recorders, and miscellaneous nifty IC's. Programs for any of the common microprocessors are of course interesting to many people.

What is it worth and how should it be submitted? TCH will pay $20 per page as printed for material. This includes drawings, listings, and photos. Material submitted should be typed or neatly written. If you must send your only copy of something, please say so or it may not be returned. It is desirable for programs to be accompanied by a flowchart. Any pictures should be black and white. If any editing other than grammatical corrections is needed, the author will be notified before publication, therefore please include your phone number.

What about advertising? TCH is now accepting paid advertising. Four formats are offered. Full page (7.5" X 10.5"), half page (7.5" X 5" or 3.75" X 10.5"), and quarter page (3.75" X 5") are available and cost $70, $40, and $25 respectively. This price is for "camera ready" copy. Ads can be set at extra charge. TCH reserves the right to reject any ad and prices are subject to change as our subscriber base of 2,300 continues to expand. For further details write to: TCH, Box 295, Cary, NC 27511.

CONTINUED FROM 1

Normally the head surface protrudes through one of the slots in the disk envelope and is held at a barely grazing distance from the disk surface. To read or write, a pressure pad is lowered to the other side of the disk directly opposite the head to press the magnetic coating against the head. All drives use the guts of a relay to move the pressure pad back and forth. The so-called head load and unload then is really pad load and unload and is accompanied by a resonant clank. The Pertec FD-400 is an exception since both the head and the pad move towards each other during head loading. Head load time is in the range of 10 to 50 milliseconds.

Another lamp-photocell assembly in the drive detects the passage of index or sector-index holes. A simple timing circuit is required to tell the difference between the index hole and the sector hole. Some drives may require readjustment of the photocell amplifier gain in order to resolve the closely spaced holes.

All floppy disk drives employ what we call a "flux transition interface" for getting data bits in and out of the drive. For our purposes at this time a "flux transition" has exactly the same meaning as "pulse" did in describing the TCH audio cassette interface (see Vol. 1 #5). Data on the disk is represented as an isolated flux transition (pulse) for a ZERO and two closely spaced flux transitions for a ONE. Unlike the tape however, the spacing between individual bits is not allowed to vary. Using numbers, the bit spacing is 4uS and the transition-to-transition spacing for a ONE is 2uS. When writing, the controller must supply properly timed pulses to the drive which translates them into flux transitions on the disk surface. When reading, the drive picks up the flux transitions, converts them to pulses, and sends them back to the controller with the original timing nearly intact. This data coding method permits over 300,000 bytes to be recorded on a single diskette. Another method, called Miller encoding or MFM, can double the storage capacity and data rate if the drive is capable of pulse timing errors of less than 1uS. This "double density" technique will be explored more thoroughly in a future article.

Most of the drive manufacturers offer numerous extra-cost options for special applications or to simplify controller design. The features that have been discussed are present on the base models of all manufacturer's drives.

Now we come to the sticky problem of data format on the disk. Actually everyone agrees on the basic record format; some leading ZEROES for synchronization, a data ID pattern, useful data (fixed length), CRC characters, and some trailing ZEROES. The difficulty arises because the data capacity of a full track which is about 4K bytes is inconveniently large. Two fundamental methods of breaking up a track into shorter records called sectors have evolved. The differences relate to how the desired sector is located without having to read or write all of them.

The IBM method is called "soft sectoring" and utilizes the one hole pre-initialized diskettes. Each sector consists of an "ID record" followed by a data record. The ID record contains the sector number of the following data record. When a read or write is to be performed, the disk controller logic must search for the ID record that matches the desired sector number (the sectors are not necessarily in order) and then read or write the following data record. The ID and data records are distinguished by a special byte called an address mark. Address marks consist of a peculiar pulse pattern that does not conform to the usual double frequency method of data encoding. A special decoder is required that can recognize these odd pulse patterns (there are actually 4 different ones) and reliably distinguish among them. Using this sectoring scheme there is room for 26 sectors on a track. Additionally, a track is reserved for an index, two are reserved for alternates (in case of a damaged disk), and one is just plain reserved. The total number of useful data bytes is thus 73*26*128 or 242,944 bytes. It should be noted that actual data exchange with IBM equipment requires far more than adherence to the physical data format; the data itself must be formatted and indexed according to a complex set of "logical format" rules.

Use of the 33 hole diskettes is an alternative sectoring method. The individual sectors are delineated by pulses from the photodector in the disk drive. Determining when the desired sector has been reached is simply a matter of counting pulses starting at the index. A sophisticated controller can maintain a continuous count so that at any point in time it knows what sector is coming up next. Records are simply written and read between the two sector pulses that define the desired sector. With hard sectoring, 32 sectors of 128 bytes each will fit on a track for a total capacity of 77*32*128 or 315,392 bytes. A potential drawback of hard sectoring apart from non-IBM compatibility is a lack of "sector address verification" preceeding a read or write. In other words, a disk drive malfunction such as missed stepping motor pulses or mis-counting of sector pulses may cause reading or writing of incorrect sectors without an error indication. A simple solution which has been found to be effective is to initialize the CRC register to the track and sector number before reading or writing instead of initializing it to zeroes. Then if an incorrect sector is read, a CRC error will be signaled. A read before a write will at least assure that the head has been positioned at the correct track.

TCH has chosen the hard sectored approach for a floppy disk interface. Actually, anything else could hardly be called "super simple". The main problem with the IBM format in a mainly software driven interface is the limited amount of time between the ID record and the following data record. There is simply not enough time for software to decode the bit stream and decide if the desired sector is next before it arrives. Also, IBM compatability precludes consideration of double density data recording.

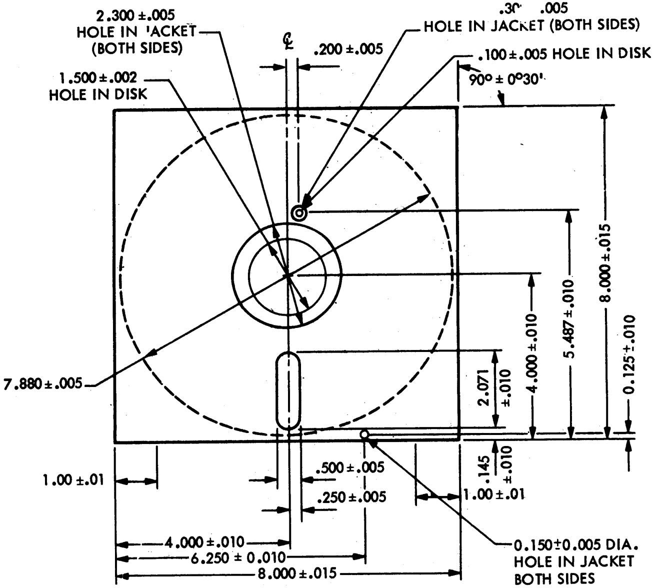
Why are commercial floppy disk controllers so expensive? The typical controller operates almost totally automatically. The CPU simply tells it what track number, what sector number, and a starting address in memory and the controller takes care of the rest via a direct memory access interface. Step counts are computed, sectors are counted (or recognized), data ID's are recognized, CRC's are computed and checked, data is serialized and deserialized, and memory addresses are counted. Additionally, the commercial jobs sometimes have elaborate error recovery logic built-in such as automatic retry. Also, multiple drives can be handled, usually with simultaneous stepping and sector counting on several drives. IBM compatible controllers often include logic for initializing diskettes as well as reading and writing and some can handle both IBM and hard sector formats. What this adds up to is a lot of IC packages, as many as 200 for an IBM compatible unit. For comparison, count up the number is IC's in an Altair or even a NOVA or LSI-11. Although some of the controller functions can be handled by a microprocessor, many of them occur at such high speed that they must be handled by hardware. Of course, these commercial controllers offer very high performance such as constant high transfer rate (reading and writing consecutive sectors) and very little load on the host processor.

How do we plan to reduce the controller complexity to only 27 packages? The obvious answer is to do everything possible in software. For example, software will issue step-in and step-out pulses to the head motor and keep track of which track the head is on. Software will wait for the index pulse and count sector pulses until the desired sector is reached. Software will serialize and deserialize the bit stream and calculate the CRC in much the same manner as was done in the audio cassette interface. Software will also control loading and unloading of the head. Hardware is still required for some functions. Encoding and decoding the double frequency pulse train, distinguishing index from sector, and starting and stopping data transfer immediately at sector boundaries are major hardware functions. The only high-speed operation is transferring the bits to and from the disk at 250,000 per second. Two 2102 memory chips will be used as a buffer between the fast disk transfer rate and the slower CPU/program transfer rate. All timing will be handled by the interface making it CPU speed independent. Believe it or not, the interface will require only one input port and one output port with 4 bits used in each!

What performance characteristics of the floppy disk are sacrificed with the super simple interface? The primary one is transfer rate. An overall average transfer rate of one sector per disk revolution can be expected which is about 750 bytes per second. A special high performance routine may be able to achieve up to 3000 bytes per second. All of the other desirable characteristics are maintained however, especially the random access and update capability. An advantage of the software driven interface is greater reliability; freedom from infrequent and often puzzling disk controller malfunctions.

How useful is a floppy disk system without a "disk operating system" monitor program? Conceptually, a floppy disk is simply a collection of 2464 individual records of 128 bytes each. Each record is addressable much like a memory location. All kinds of dedicated applications such as those described earlier can be easily written without even knowing what a disk operating system is. The user would develop storage and indexing techniques that are best suited for his application. Take for example the storage method used in our floppy based mailing list program. There are 1000 subscribers on a disk and two sectors allocated to each subscriber. To find a subscriber, the program simply computes ((Sub. No.) MOD 1000)*2+126 to get a composite sector number. The track is the quotient when the composite is divided by 32 and the sector is the remainder.

The super simple interface along with the 400 byte support software can also be used with existing disk operating systems. All that generally needs to be done is to locate the subroutine that actually handles reading and writing on the disk and replace it with calls to the support package. More extensive modification is required if the original disk system had a different sector size or different number of sectors per track.



NOTE:
ALL DIMENSIONS SHOWN IN INCHES

FIG. 1 DISKETTE PHYSICAL DIMENSIONS

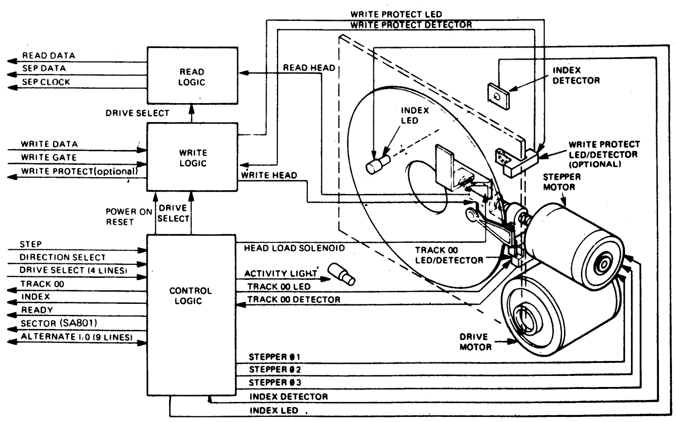


FIG. 3 TYPICAL FLOPPY DISK DRIVE

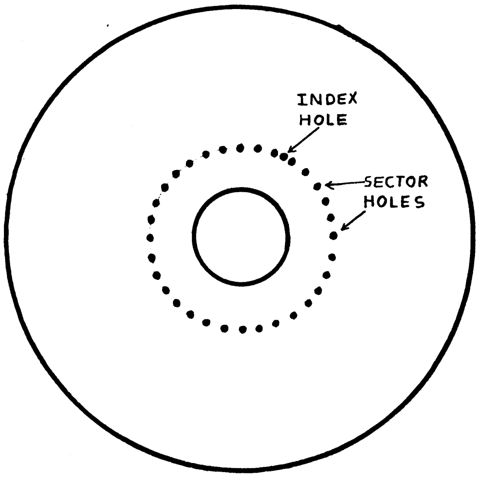


FIG. 2 HARD SECTORED DISKETTE

In issue number 10, the design, construction, operation, and programming of the floppy disk interface will be described in our usual level of detail. Also there will be a listing of the floppy disk routine necessary to operate the interface. In order to formulate support plans beyond that point, we would like to have some reader feedback. In particular, is there interest in PC boards and PROM programming for the floppy disk interface as there was for the TCH audio cassette interface? A simple postcard will do. We intend to remain the best publication for the serious computer hobbyist if not the biggest or most regular.

Following is a listing of the major floppy disk drive manufacturers. With one exception, a TCH staff member has seen a sample drive from each of the companies listed.

FLOPPY DISK DRIVE SUPPLIERS

Orbis Systems
14251 Franklin Ave.
Tustin, CA 92680

California Computer Products
2411 W. LaPalma Ave.
Anaheim, CA 92801

PERTEC, Peripheral Equipment Div.
9600 Irondale Ave.
Chatsworth, CA 91311

Control Data Corp.
P.O. Box 0
Minneapolis, MN 55440

Shugart Associates
435 Indio Way
Sunnyvale, CA 94086

SYCOR
100 Phoenix Dr.
Ann Arbor, MI 48104
(We have not seen this drive)

Remex
1733 Alton St.
Santa Ana, CA 92705
(These people resell Orbis drives,
may be easy to deal with)

DISKETTE SUPPLIERS

DYSAN Corporation
2388 Walsh Ave.
Santa Clara, CA 95050
(We have gotten excellent
service from these people)

AN IMP-16 MICROCOMPUTER SYSTEM PART 2 by Hal Chamberlin

Things are changing so rapidly that the first paragraph of these installments will have to be devoted to news items. Poly-Paks no longer has IMP-16 sets. We don't know if IEU still has them or not. However all surplus IMP-16 chip sets come through Godbout so perhaps some letters will persuade him to sell them directly. Of course all National distributors have some; TCH has gotten them this way for $160. The real problem is that they hit the surplus market too early. We got some more data on the "power math" CROM. Basically it provides instructions for operating on 32 bit binary fractions (mantissas) such as 32X32 add, subtract, multiply, divide, and normalize. The user need only code exponent handling and the result is a floating point package with 32 bit mantissas (10 decimal digits) and 16 bit exponents (10**10000 anybody?) with a 100 uS add time and a 600 uS multiply time. The bad news is that "power math" and the extended CROM share some op-codes so they cannot normally be used together. There is a way to enable one or the other using a status flag however (status flags can be saved during interrupt). Implementation of the scheme requires the use of a 74LS260 in place of the 74LS54. PC layout of the CPU board is planned but some readers couldn't wait and have already started to wire-wrap CPU boards. At least 3 TCH staff members will be building IMP systems and at least one of them will have a floppy disk so software support will not be lacking toward summer. Quick note: do not buy plain 2107 4K RAM's for this system! They have a different pinout, are very slow, and in a word, totally obsolete. TMS4030, TMS4060, 2107A, and 2107B are all fine as well as most gradeouts. The author has a limited supply of TMS4030-ZA0248 4K RAMS tested for operation in this system for $7.50. An error was made in the parts list for the memory board. Rather than three 7404's, it should be two 7404's and a 7440.

Now with the news out of the way, let us take a top-down approach to describing the PUNIBUS controller. The bus controller runs continuously, non-stop, from power-up to power-down crunching out 1.43 million cycles per second or one cycle every 700NS. All memories in the system likewise operate at this cycle rate. Each cycle is awarded on the basis of priority to one of 7 possible requesters. The highest priority requester is the CPU. Below the CPU are 5 direct memory access (DMA) devices. The lowest priority requester is the memory refresher which is always requesting bus cycles. Thus if the system is idle, that is, CPU halted and no DMA activity, all of the bus cycles are being awarded to the memory refresher. During operation, cycles that are unclaimed by the CPU or DMA are also awarded to the refresher. The PUNIBUS controller always generates the timing signals necessary for data transfer regardless of which requester controls the particular cycle. Thus DMA devices in the system don't have to generate any timing of their own, instead they just sit and respond to control signals issued by the PUNIBUS controller.

Any device interfaced to the bus that is not a possible DMA requester is expected to behave as if it was a memory. At the beginning of every bus cycle a 16 bit address is established. This address specifies either an actual memory location or a peripheral device register. There are only two types of bus cycles; a read cycle and a write cycle. During a read cycle, data is read from a memory or peripheral register into the CPU or DMA device. During a write cycle, data is written from the CPU or DMA device into a memory or peripheral register. The CPU or DMA device awarded the cycle determines whether a read or a write cycle is to be performed. The memory refresher, of course, always does read cycles. Undefined operations such as addressing non-existant memory or writing into a read-only peripheral register are not harmful and function as NO-OP bus cycles.

Figure 1 shows the timing relationships of the PUNIBUS. Although actual times in nanoseconds are given, it is important to note that correctly designed interfaces to the bus will work properly even with considerable variation in the timing details as long as the basic relationships are retained. This allows flexibility to change the details to accomodate other CPU's such as a bipolar IMP or a down-spec chip set without obsoleteing memory and peripheral designs.

As can be seen, a bus cycle starts with the signal BUS ADDRESS ENABLE (BAE) going high and terminates when it goes high again for the next cycle. Actually though, some preparation takes place toward the end of the previous cycle. An internal "priority strobe" is generated which causes the BUS REQUEST (BR) lines including CPU and refresh request to be examined to determine who will get the next cycle. The determination is made and the three bit grant code of the winning requestor is placed on the BUS GRANT (BG) lines immediately before the cycle commences with BAE going high. At this time the one requestor whose code is on the BG lines is expected to gate a 16 bit address onto the BUS DATA (BD) lines as long as BAE is high. Any BD lines not specifically driven will assume a ONE level because of pullup resistors. If a write cycle is to be executed, the BUS WRITE REQUEST (BWR) line should be pulled down during BAE time, otherwise a read cycle will be automatically assumed. This address phase of the cycle is identical for both read and write operation.

After the address phase we have the data transfer phase which is different for read and write cycles. In the case of a read cycle, the bus controller generates two signals, BUS DATA OUT ENABLE (BDOE) and BUS DATA OUT STROBE (BDOS) which control the data transfer from memory or peripheral register to CPU or DMA device. BDOE first goes high to cause the addressed memory or peripheral to gate its data onto the BD lines. BDOS is bracketed by BDOE and can be used to strobe data from the bus into the CPU or DMA device's data register on its trailing edge. The timing of this pair of pulses is chosen to allow memories sufficinet access time and to allow the IMP-16 chip set to grab the data directly from the bus with no intervening latches.

During the transfer phase of a write cycle, BDOE and BDOS remain inactive while BUS DATA IN ENABLE (BDIE) and BUS WRITE ENABLE (BWE) control the data transfer from CPU or DMA to memory or peripheral. BDIE becomes active first causing the CPU or DMA device to gate the data to the written onto the BD lines. BWE which is bracketed by BDIE then becomes active causing the memory or peripheral to accept and store data from the bus. The timing shown for these signals was chosen to be compatible with the 4K RAM's used in this system.

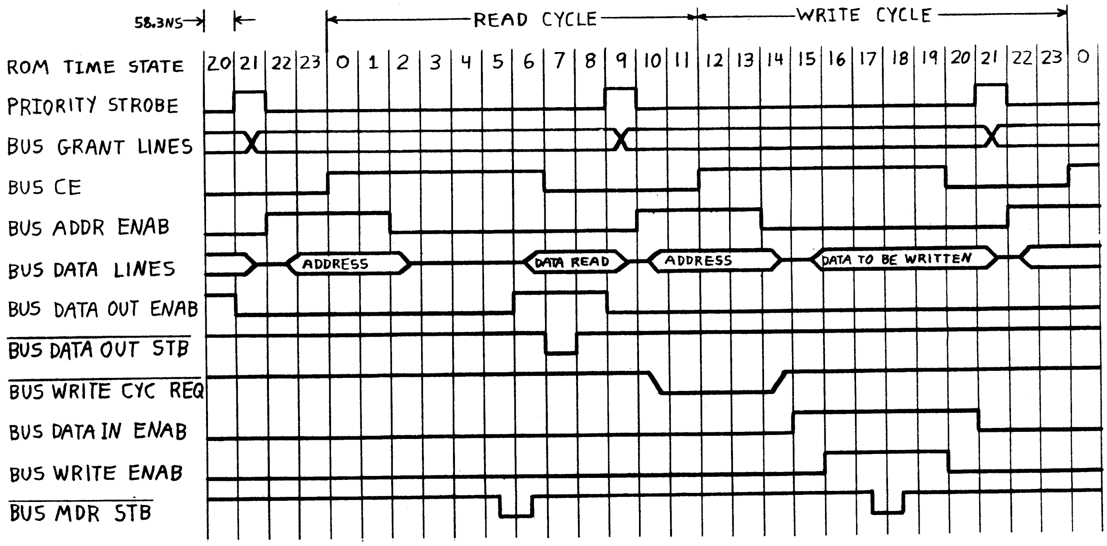


FIGURE 1. PUNIBUS TIMING RELATIONSHIPS

The responsibilities of a memory board or peripheral interface are quite simple. During the address phase of each cycle, pertinent information about the 16 bit address on the BD lines must be latched on each interface board. Generally, a memory interface using 4K RAM's need only latch a single bit since the RAM chips have built-in latches for address and chip select. The single bit needing a TTL latch simply indicates whether the board was addressed or not. Likewise, a peripheral register can decode its address directly from the BD lines and use a flip-flop to remember if it was addressed. In either case, the leading edge of BUS CE is used for address strobing since it always occurs when the address is valid. Once a memory or peripheral has latched the fact that it was addressed, it either sends its data out if it sees BDOE or accepts new data in if it sees BWE. Thus memories and peripheral registers are passive, merely responding to bus signals as they occur.

Four "convenience" signals are provided on the bus. One which has already been mentioned is BUS CE. Memory boards using 22 pin 4K RAM's can simply amplify this signal to NMOS levels and apply it to the Chip Enable clock input of the RAM chips. Its function within the RAM is to start up the memory cycle and also strobe the on-chip address and chip select latches. Another signal provided specifically for memory boards is BUS MDR STROBE. Its purpose is to strobe the data out latches on the memory board when data out from the 4K RAM's is valid. Some unfortunate timing constraints on both TMS4030 and 2107 type RAM's require latches to hold the data after it disappears from the RAM outputs. Although BDOE could have been turned on earlier with the leading edge strobing the latches, excessive noise generation would have resulted. BUS I/O ADDR is a signal that goes low whenever the binary value on the data lines is between FF00 and FF7F hexadecimal. This range of addresses is normally assigned to peripheral devices. Use of this signal in decoding I/O addresses can save a 9-input AND gate equivalent on each interface card. BUS CLOCK is provided as a convenient high frequency clock with .005% accuracy. Its frequency is such that when put through a 16 bit divider, the resulting frequency is middle C, 261.625 Hz. Additionally, 12 cycles of this clock make up one bus cycle whose length is actually 699.88NS.

Two signals are involved with power-on reset and console reset. The POWER OK bus line should be pulled low by an external circuit associated with the power supply when all supply voltages are present and stabilized. This circuit should also be connected to the console reset push button so a power on sequence can be simulated without loosing memory contents. A simple delay circuit is shown in figure 2 which functions quite well. Alternatively, a true power monitor can be built using zener diodes to sense when the supply voltages are actually present. BUS RESET is generated in response to POWER OK by the CPU board. It resets the CPU and should reset all peripheral interfaces to a safe, idle condition when it goes low. It has no effect on the bus controller or memory refresher however.

The interrupt system uses the very simple software polling technique described elsewhere in this issue. The BUS INT REQ (BIR) line is a wire-or line with pullup resistor which is pulled low by any device that wants to request an interrupt. The CPU responds, provided its master interrupt enable is on, by calling a subroutine at 0001 and simultaneously turning master interrupt enable off. After saving status, the program can look at the status register of each possible interrupting device to determine who is requesting. This search can be as fast as 9.8uS per device with proper use of the SKAZ (SKip if And is Zero, ANDs addressed memory location with a register and skips the next instruction if the result is zero) instruction. The device service routine then turns off the interrupt request for that particular device and turns master interrupt enable back on. Priority in the case of simultaneous interrupts is determined by the order of scanning. Nested interrupts can also be programmed. Thus the interrupt system essentially works like that on a PDP-8. The usual interrupting device interface also has an interrupt enable for each device making non-interrupt I/O programming possible if desired. More details on I/O interfacing and interrupts will be given in part 4.

Figures 3 and 4 show the timing generator and bus controller. Since this circuitry is on the CPU board, some CPU circuitry has encroached which will be described in part 3. See TCH #2 if any of the logic gate symbols are confusing. You will note that inputs always enter from the left of a drawing and outputs leave at the right. All signals going offpage are given a name and should mate with similarly named signals on the other pages. If an offpage signal has a number on it, it goes to the CPU board edge connector. If the number is 46 or less, it is a bus signal and is available at the same pin number of any board in the system. Some signals shown in figure 3 and 4 will not be mated until part 3.

The heartbeat of the system is the 17.145893 mHz oscillator in figure 3. Its output drives a hex latch and is buffered to drive the BUS CLOCK line. The latch and two 32 word by 8 bit bipolar PROM's make up the bulk of the timing generator. As can be seen, 6 of the 16 PROM outputs go to the 74S174 hex latch and 5 of these are fed back to the PROM address inputs. The result is that every cycle of the 17 mHz clock causes the PROM-latch combination to take one step in a programmed sequence. Using the PROM pattern in figure 5, this sequence is 24 steps long and takes 1.4uS to step through thus matching the minimum IMP-16 microcycle time. In order to avoid glitches at the PROM outputs when the address changes, the sequence of addresses has been chosen such that only one address line changes at a time. Figure 6 shows the PROM pattern in time sequence rather than address sequence. The 8 addresses not normally used all point to time state zero to avoid a possible lockup condition. The sequence of addresses was also chosen so that a decoder could be used to generate the 4-phase non-overlapping clock needed by the IMP-16 chips from 3 of the address bits.

The remaining 11 PROM bits are the various system timing signals. Those prefixed RAW require additional gating before being used; the others are ready to go. BUS MDR STROBE goes through the latch to effect an additional 30NS delay. The purpose of the flip-flop connected to the 4-phase decoder is to insure that the CPU starts up on phase 1 after a system reset. Although 8223 PROM's with pullup resistors are shown, a tri-state PROM such as an 82123 can be used without the resistors.

System reset and power up control are handled by the two 7413 Schmidt triggers and other discrete circuitry at the bottom of figure 3. The first 7413 gives a snap-action response to BUS POWER OK which may be a slowly changing signal. The R-C network and second 7413 provide a signal that tracks BUS POWER OK but with a several millisecond delay. This delayed signal, after inversion, becomes BUS RESET. The transistors apply -12 volt power to the IMP chips when bus power is OK and remove it otherwise. BUS RESET also controls application of the 4-phase clocks to the microprocessor. Thus the timing relationship between power application and removal and clock application and removal is such that the IMP is properly initialized.

The logic in the upper third of figure 4 modifies some of the timing signals from the PROM according to bus cycle type; read or write. Flip-flop 1 samples BUS WRITE REQUEST at the leading edge of BUS CE and retains the read/write decision for the remainder of the cycle. The network at the top of the page consisting of a 7432 and 7410 delays the fall of BUS CE by 50NS during write cycles. It behaves as a simple inverter during read cycles. Lengthening BUS CE during write cycles only provides improved timing margins for writing into 4K RAM's without unnecessary power dissipation during read cycles. The gates on BDOS and BDOE gate these signals on for read cycles and off for write cycles. Likewise, BDIE and BWE are gated on for writes and off for reads.

The network starting with the 74LS21's is a partial address decoder. If the address on the bus is between FF80 and FFFF, flip-flop 2 is set indicating that the on-board bootstrap ROM has been addressed. If the address is between FF00 and FF7F, BUS I/O ADDR is activated to inform peripherals that an I/O address is on the bus.

The next group of logic is the cycle request and grant priority logic. Gating for CPU cycle request and the 5 DMA request lines go into a hex latch that is strobed by PRIORITY STB near the end of each bus cycle. The latches are necessary to hold the input to the priority encoder constant throughout the next cycle. The 74148 determines the highest priority input present (active low, A is highest, H is lowest priority) and outputs a 3 bit code identifying that input. The G input is not used in this drawing but could be used for a sixth DMA request along with a latch. The H input is refresh request which is always present.

The bottom of figure 4 is the refresh logic for all dynamic memory in the system. A 74LS20 detects the coincidence of refresh grant (111) and BAE which indicates that the refresh address should be placed on the bus. The output thus enables an 8097 which gates the 6 significant refresh bits onto the bus. The other 10 bits assume a logic 1 and the bus controller assumes a read cycle. When the 8097 is gated off again, a 6 bit counter made from a 7474 and a 7493 counts up one notch in preparation for the next refresh cycle. Two 8556 tri-state counters could have replaced the 7474, 7493, and 8097 used here but they were too hard to get to justify their use.

That concludes the description of the bus controller. Everything else in the system is just a collection of bus interfaces. Although the remainder of this series will be specifically concerned with IMP-16 interfacing to the bus, the basic concepts and bus structure can be used with any microprocessor. In fact, an essentially identical bus system was used in the design of a super 8008 system over three years ago.

In the next issue a brief description of the IMP-16 chip set will be given along with the remainder of the CPU board schematic and accompanying discussion.

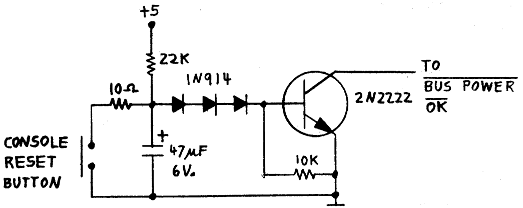


FIG. 2 SIMPLE POWER OK CIRCUIT

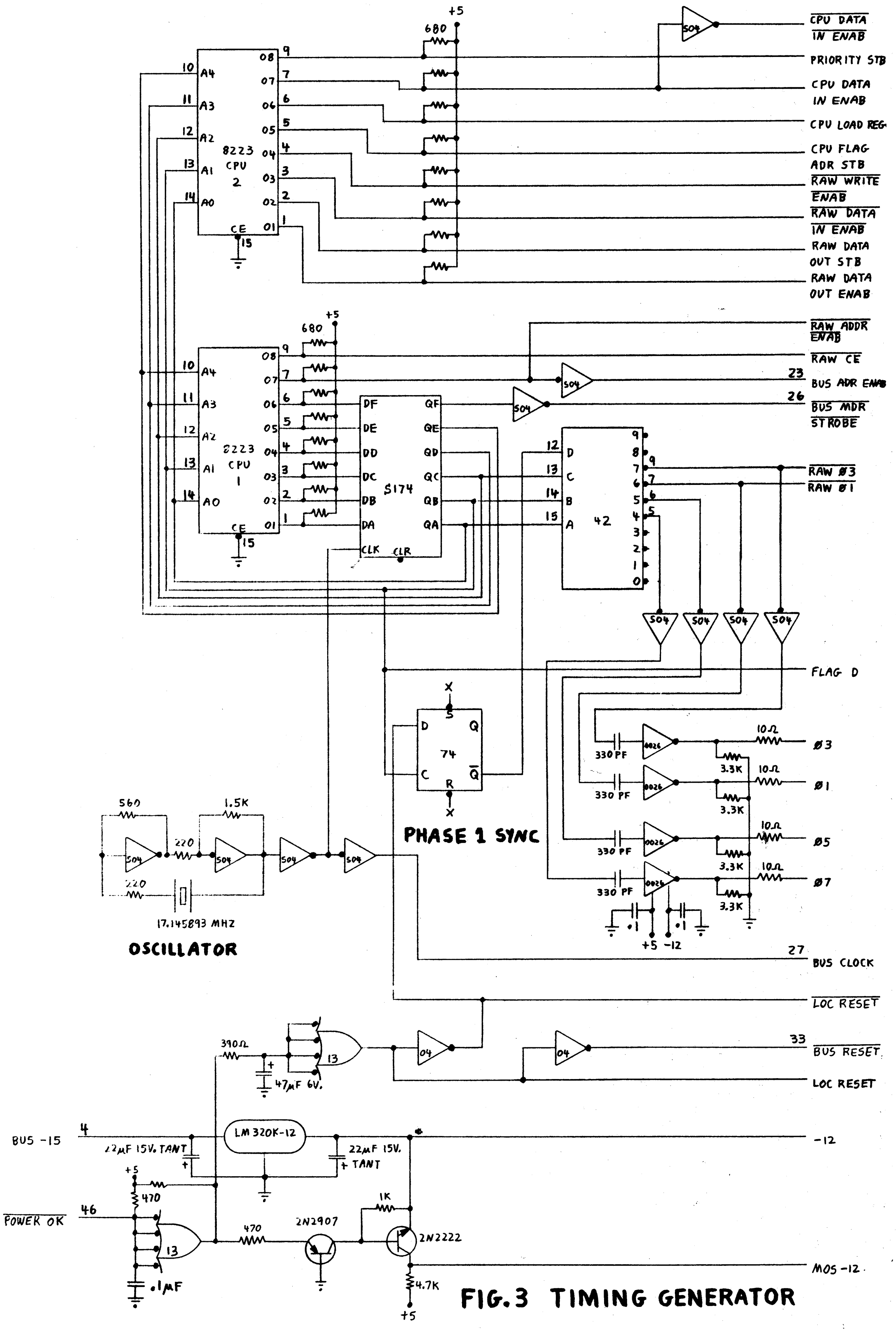


FIG. 3 TIMING GENERATOR

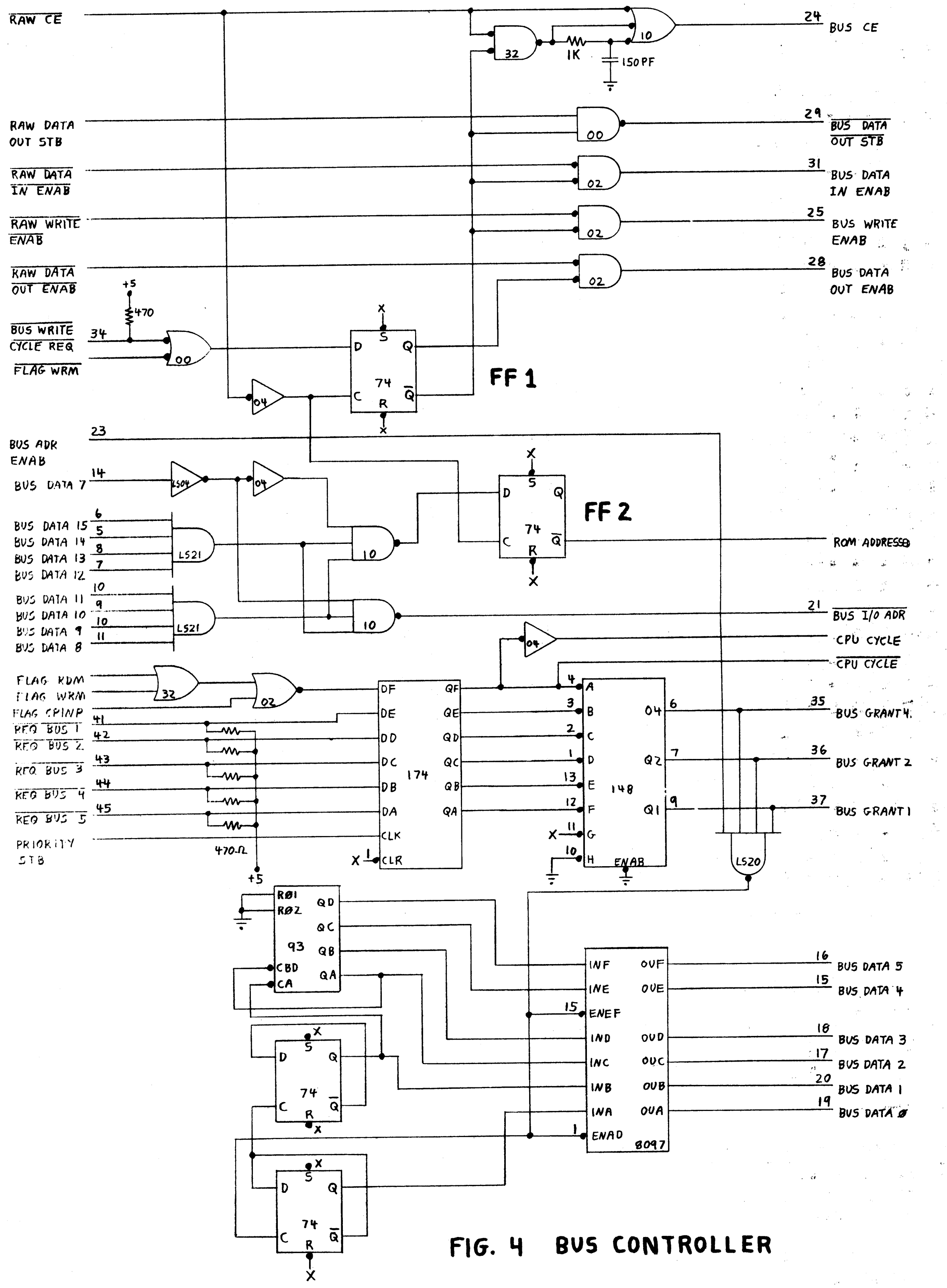


FIG. 4 BUS CONTROLLER

		CPU-1								CPU-2							
ROM ADDRESS	TIME STATE	RAW CE	RAW ADDR ENAB	RAW MDR STROBE	NEXT ROM ADDR 16	NEXT ROM ADDR 8	NEXT ROM ADDR 4	NEXT ROM ADDR 2	NEXT ROM ADDR 1	PRIORITY STROBE	CPU DATA IN ENAB	CPU LOAD REG	CPU FLAG ADDR STB	RAW WRITE ENAB	RAW DATA IN ENAB	RAW DATA OUT STB	RAW DATA OUT ENAB
0	16	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	1
1	15	0	1	0	0	0	0	0	0	0	0	1	0	1	0	0	1
2	23	1	0	0	0	0	1	1	0	0	0	1	0	1	1	0	1
3	-	X	X	X	0	0	1	1	0	X	X	X	X	X	X	X	X
4	-	X	X	X	0	0	1	1	0	X	X	X	X	X	X	X	X
5	14	0	1	0	0	0	0	0	1	0	0	1	0	1	1	0	1
6	0	0	0	0	0	1	1	1	0	0	0	1	0	1	1	0	1
7	7	1	1	0	0	1	1	1	1	0	0	1	0	0	0	1	0
8	17	0	1	1	0	1	1	0	0	0	0	1	0	0	0	0	1
9	10	1	0	0	1	1	0	0	1	0	0	1	0	1	1	0	1
10	-	X	X	X	0	0	1	1	0	X	X	X	X	X	X	X	X
11	9	1	1	0	0	1	0	0	1	1	0	0	0	1	1	0	1
12	18	0	1	0	1	1	1	0	0	0	0	1	0	0	0	0	0
13	-	X	X	X	0	0	1	1	0	X	X	X	X	X	X	X	X
14	1	0	0	0	1	1	1	1	0	0	0	1	0	1	1	0	1
15	8	1	1	0	0	1	0	1	1	0	0	1	0	1	0	0	0
16	21	1	1	0	1	0	0	1	0	1	0	1	0	1	1	0	1
17	-	X	X	X	0	0	1	1	0	X	X	X	X	X	X	X	X
18	22	1	0	0	0	0	0	1	0	0	0	1	0	1	1	0	1
19	5	0	1	1	1	0	1	1	1	0	0	1	0	0	0	0	1
20	20	1	1	0	1	0	0	0	0	0	0	1	0	1	0	0	0
21	13	0	0	0	0	0	1	0	1	0	0	1	0	1	1	0	1
22	-	X	X	X	0	0	1	1	0	X	X	X	X	X	X	X	X
23	6	0	1	0	0	0	1	1	1	0	0	1	0	0	0	0	0
24	-	X	X	X	0	0	1	1	0	X	X	X	X	X	X	X	X
25	11	1	0	0	1	1	1	0	1	0	0	1	0	1	1	0	1
26	3	0	1	0	1	1	0	1	1	0	0	1	0	1	0	0	1
27	4	0	1	0	1	0	0	1	1	0	0	1	0	0	0	0	1
28	19	1	1	0	1	0	1	0	0	0	0	1	0	0	0	1	0
29	12	0	0	0	1	0	1	0	1	0	0	1	0	1	1	0	1
30	2	0	1	0	1	1	0	1	0	0	0	1	1	1	1	0	1
31	-	X	X	X	0	0	1	1	0	X	X	X	X	X	X	X	X

FIG.5 TIMING ROMS IN ADDRESS SEQUENCE

		CPU-1								CPU-2							
TIME STATE	ROM ADDRESS	RAW CE	RAW ADDR ENAB	RAW MDR STROBE	NEXT ROM ADDR 16	NEXT ROM ADDR 8	NEXT ROM ADDR 4	NEXT ROM ADDR 2	NEXT ROM ADDR 1	PRIORITY STROBE	CPU DATA IN ENAB	CPU LOAD REG	CPU FLAG ADDR STB	RAW WRITE ENAB	RAW DATA IN ENAB	RAW DATA OUT STB	RAW DATA OUT ENAB
0	6	0	0	0	0	1	1	1	0	0	0	1	0	1	1	0	1
1	14	0	0	0	1	1	1	1	0	0	0	1	0	1	1	0	1
2	30	0	1	0	1	1	0	1	0	0	0	1	1	1	1	0	1
3	26	0	1	0	1	1	0	1	1	0	0	1	0	1	0	0	1
4	27	0	1	0	1	0	0	1	1	0	0	1	0	0	0	0	1
5	19	0	1	1	1	0	1	1	1	0	0	1	0	0	0	0	1
6	23	0	1	0	0	0	1	1	1	0	0	1	0	0	0	0	0
7	7	1	1	0	0	1	1	1	1	0	0	1	0	0	0	1	0
8	15	1	1	0	0	1	0	1	1	0	0	1	0	1	0	0	0
9	11	1	1	0	0	1	0	0	1	1	0	0	0	1	1	0	1
10	9	1	0	0	1	1	0	0	1	0	0	1	0	1	1	0	1
11	25	1	0	0	1	1	1	0	1	0	0	1	0	1	1	0	1
12	29	0	0	0	1	0	1	0	1	0	0	1	0	1	1	0	1
13	21	0	0	0	0	0	1	0	1	0	0	1	0	1	1	0	1
14	5	0	1	0	0	0	0	0	1	0	0	1	0	1	1	0	1
15	1	0	1	0	0	0	0	0	0	0	0	1	0	1	0	0	1
16	0	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	1
17	8	0	1	1	0	1	1	0	0	0	0	1	0	0	0	0	1
18	12	0	1	0	1	1	1	0	0	0	0	1	0	0	0	0	0
19	28	1	1	0	1	0	1	0	0	0	1	1	0	0	0	1	0
20	20	1	1	0	1	0	0	0	0	0	1	1	0	1	0	0	0
21	16	1	1	0	1	0	0	1	0	1	0	1	0	1	1	0	1
22	18	1	0	0	0	0	0	1	0	0	0	1	0	1	1	0	1
23	2	1	0	0	0	0	1	1	0	0	0	1	0	1	1	0	1

FIG.6 TIMING ROMS IN TIME SEQUENCE

CLASSIFIED ADS

There is no charge for classified ads in TCH but they must pertain to the general area of computers or electronics, and must be submitted by a non-commercial subscriber. Feel free to use ads to buy, sell, trade, seek information, announce meetings, or for any other worthwhile purpose. Please submit ads on separate sheets of paper and include name and address and/or phone number. Please keep length down to 10 lines or less.

HELP WANTED: I have some core stacks that I wish to lash up to my Altair 8800. I have no info on them except that I believe they are from Burroughs equipment. There are no drivers or sense amps, only core frame and glass diodes galore. Would appreciate hearing from anyone that may be able to help. Stanley D. Davis, RD 1, Stittville, NY 13469

FOR SALE; MITS RS-232 serial I/O board for the Altair 8800 (88-SIOA), assembled and tested, $80. Expander motherboard (88-EC), $8. Processor Technology MB-1 full-width 16-slot heavy-duty motherboard for the 8800, $25. David Richards, 6655 Hill St., El Cerrito, CA 94530, Ph. 415/529-0759

WANTED: Floppy disk drives and matrix printer. Send information including condition and prices to Fred Holmes, 101 Brookbend Ct., Mauldin, SC 29662.

FOR SALE: Wangco model 7 tape drives, NFE model 250 cassette drives (Cybertronics sells an Altair interface board), 4Kx12 memory systems, Tally line printers, paper tape readers, paper tape punch, TMS 2501 NC's, 3002, 3003, 3113, 7491, 1414L, 710, 741. Please inquire and make offer. Also interested in corresponding with hobbyists interested in COSMAC. John L. Marshall, Box 242, Renton, WA 98055

FOR SALE: Tired of waiting for MITS memory boards? I have two MITS 4K dynamic memory boards for sale at the kit price of $195 each. These boards were carefully assembled by an electrical engineer and factory checked. All bits certified. H. S. Corbin, 11704 Isben Drive, Rockville, MD 20852, Ph. 301/881-7571

FOR SALE: BELL 103 compatible modem PC cards. These are field rejects but include documentation. $17.50 postpaid. D. Blevins, 1857 Babe Ruth Ct., San Jose, CA 95132

NEW CLUB: If you live in the metropolitan New Orleans area and are interested in computers, you are invited to join our group. Whether your interest is hardware, software, applications, or just general interest we welcome your input. For further details, please write or call: Emile Alline, 1119 Pennsylvania Ave. Slidella, LA 70458, Ph. 504/641-2360

FOR SALE: ASR 33 in good condition with stand and some extra features. Sprocket feed platten, self contained power supplies and relays for tape reader control - ready for 6 wire computer hookup. Local pickup only. $425.00. Neal Sheffield, Jr., 108 Elmwood Terrace, Greensboro, NC 27408, Ph. 919-275-7720

WANTED: Software for business applications that will run on 8080 micro computers. Will buy, trade, or distribute for costs plus royalities (where required). John Lynn, 13431 SW 79 st., Miami, FL 33183, Ph. 305/271-2805

INTERFACING THE ALTAIR 8800 PART 2

by Hal Chamberlin

Many applications of hobby computers can benefit from the use of interrupts. MITS and a couple of other alternate sources have had "vectored interrupt controller" cards announced for some time but until recently have been unable to deliver. The problem is that the vectored interrupt cards use the Intel 8214 vectored interrupt IC which was announced over a year ago but was not available in volume until now. Although these cards offer many advantages in large systems with heavy use of interrupts and stringent response time requirements, we feel that not enough attention has been given to the interrupt capability built into the basic Altair. This article will discuss the effective use of this "free" resource in the design of custom I/O interfaces. A keyboard interface with interrupt capability will be used as a model to illustrate the concepts presented.

First, let us discuss what interrupts are, what they are good for, and the three popular implementation methods that are used in minicomputers.

Often in a computer system one has a program crunching away in the CPU and an impatient I/O device that occasionally wants attention from the CPU (and, of course, a different section of the program). Many examples come to mind but we will use the case where the CPU is busy drawing on a graphics display (see TCH #1) and the impatient I/O device is a human at a keyboard typing in commands to alter the image. In computer jargon, the routine doing the drawing is called a "background task" and the routine that reads characters from the keyboard and interprets them is called a "foreground" task.

Programming this application on a system without interrupts would involve three major routines. The first is an initialization routine. Its job is to set up the initial display list, initialize the various software flags, and enable the keyboard for the first user command. Normally, the initialization routine is executed only once, just after entering the program. Following initialization, a branch is taken to the draw routine which sits in a loop drawing and refreshing the graphic image from data in the display list. Periodically, such as once per image refresh, the draw routine should test the keyboard to see if anything has been typed in. If so, a branch is taken to the third routine, a command interpreter. Its job is to read the character from the keyboard, process it, re-enable the keyboard, and finally jump back to the draw routine. Processing the character frequently amounts to storing it away in memory, a very quick operation. When a complete command has been stored, character processing would also include execution of the command such as line erase, change scale, etc.

This scheme can be made to work very well but let us look at the possible limitations. First, the keyboard test must be placed at a point in the draw routine that is executed frequently enough to satisfy the person at the keyboard. Once per refresh may not be enough if the image is complex and the typist is fast. On the other hand, testing the keyboard once per line drawn would not be good either because a large percentage of CPU time may be spent simply testing. In some applications, a suitable test point in the background routine may not exist. What is really needed is a way for the keyboard to "interrupt" the draw routine and cause a subroutine call to the keyboard routine. This is, in fact, what interrupt hardware accomplishes.

Now let us look at how a very simple (but completely effective in this example) interrupt scheme for the keyboard would work. First, the keyboard interface would have two control/status flip-flops instead of the usual one. These would be called BUSY and DONE. These flip-flops can be altered by both an OUT instruction and key action and can be read with an INP instruction. Given two flip-flops, there are four possible combinations to represent operating "states" of the keyboard. If both BUSY and DONE are off, the keyboard is in an idle state, i.e., not being used. If a program needs a character from the keyboard, it should set the BUSY flip-flop but leave DONE off. The keyboard would now be in a "busy" state waiting for the operator to press a key. TCH has found that a light-emitting diode connected to the BUSY flip-flop and mounted on the keyboard cover is very effective in informing the operator that a key may be pressed. When a key is finally pressed, BUSY is automatically turned off and DONE is turned on signifying that the keyboard register now has a valid character but the program has not yet read in the character. When the program does read the character, DONE is turned off returning the keyboard to an idle state. We have also found that a small speaker inside the keyboard case connected so that it clicks when the program reads the keyboard gives valuable feedback to the operator, reducing keying errors. Both BUSY and DONE being on simultaneously is a meaningless situation.

Now, how would these two control/status flip-flops be connected to the Altair for interrupts? First, the keyboard only requires service (reading a character) when the DONE flip-flop is on. So to implement interrupts, we would tie DONE to the PINT line (pin 73) on the Altair bus. Whenever the Altair sees the coincidence of PINT and interrupt enable (set with the EI instruction), it will execute a CALL to location 000:070 (split octal notation, see issue #8) and turn interrupt enable off. There must, of course, be an "interrupt service" routine at 000:070 or a jump to one elsewhere.

Using the same display example, we would still have three major routines in a system with keyboard interrupts. The initialization routine would first force the keyboard into an idle state (it may have been left in an undefined state by the previous program) then it would set BUSY thus enabling the keyboard. It would also execute an EI to enable interrupts on the Altair. The draw routine is the same as before but now it doesn't have to test the keyboard flags. When the operator hits a key, BUSY will be turned off and DONE turned on by the keyboard interface logic. DONE being on and the Altair interrupts being enabled will force execution of a CALL to 000:070 when the current instruction is finished. The CALL stores where it came from on the stack and disables interrupts as if a DI instruction was executed. The interrupt service routine at 000:070 should also save status and any registers it uses on the stack before doing anything else. At this point, a character is read from the keyboard, DONE is turned off thus idling the keyboard, and the character is acted upon. After processing the character, the interrupt service routine turns BUSY on to re-enable the keyboard, restores registers and status from the stack, re-enables the Altair interrupt system, and finally executes a RET instruction. The display routine is now executing again and since no status or registers were changed, it is not even aware of the interruption. Of course the display was stopped momentarily while interrupt service was in control but generally the time is so short that the interruption is not noticable.

So far we have discussed a system in which only one device had interrupt capability, namely the keyboard. Most of the fancier uses of interrupts are in systems where several peripherals can interrupt and the programmer desires simultaneous I/O, that is, more than one I/O device running at a time. In addition to the hardware and software previously described, a method must be found to identify which device caused a given interrupt, and to resolve the conflict that exists when two or more interrupts occur simultaneously. In fact, the only real difference between various interrupt systems is in how these two functions are performed.

The most obvious, best, and expensive multiple interrupt scheme is called hardware vectored interrupts. Recall the keyboard/display example in which the keyboard service routine had to be at location 000:070. With hardware vectored interrupts, other devices would cause calls to other locations. In other words, hardware takes care of identifying which device is interrupting, and automatically branches to the service routine for that device. This is in fact what the vectored interrupt cards implement.

A similar system often used in minicomputers is called software vectored interrupts. All interrupts cause the CPU to branch to the same location. A common interrupt service routine then issues an INTA instruction (INTerrupt Acknowledge) which causes the interrupting device to return its device address. Using the device address, the common interrupt service routine can set up an N-way jump to a service routine specific to that device. This method uses less hardware than the previous method but is somewhat slower because of the time taken by the common service routine. Both methods have the advantage that the time between interrupt and entry to the matching service routine (called interrupt latency time) is independent of the number of possible interrupt sources.

What about the case of two or more simultaneous interrupts? In both vector schemes each device is assigned a "priority", usually by how it is wired into the system. When two or more interrupt requests are pending, the device with the highest priority gets serviced. When the service routine turns off the interrupt request in the highest priority device and returns, another interrupt occurs immediately from one of the remaining devices. Eventually all interrupts get serviced, in order of decreasing priority. Some really sophisticated interrupt systems even allow a higher priority device to interrupt the service routine for a lower priority device! This feature is often found in process control systems and is called "nested priority interrupts".

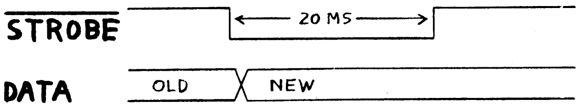


FIG. 1 CLARE-PENDAR KEYBOARD TIMING

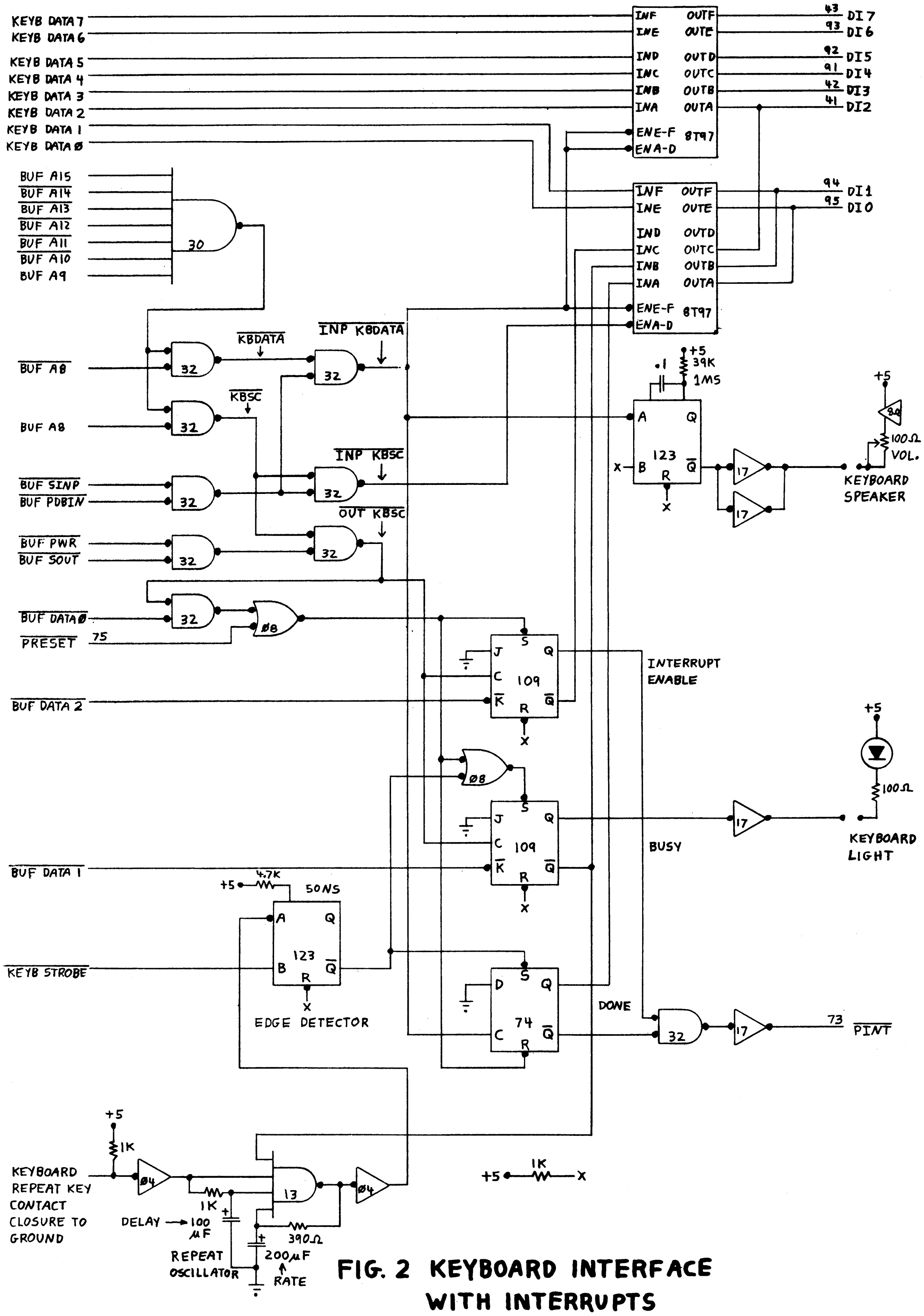


FIG. 2 KEYBOARD INTERFACE WITH INTERRUPTS

The last method is called "software polled interrupts" which is what we will be using for multiple interrupts on the Altair. It too is often found in minicomputers, most notably the PDP-8. Again, all interrupts cause the CPU to branch to the same location; 000:070 in the Altair. Here a common service routine tests (polls) the DONE status bit of every device that can cause interrupts, one device at a time. When one is found, a branch is taken to the service routine for that device. The priority is determined by the order of DONE status testing which means that priorities can be changed with software. Of course the disadvantage of polled interrupts is that interrupt latency time is longer if a lot of devices must be tested. Fortunately, the latency time is shortest for high priority devices since they are tested first. In fact, for most instruction sets, the first two or three devices in a polled interrupt system may get faster service than in a software vectored system because of the time spent in setting up the N-way branch with the latter. On the Altair, using polled interrupts and no memory waits, the worst case latency time to the top priority device is 33 microseconds. An additional 12 microseconds is added for each successively lower priority device in the polling chain. Note that these times include saving A and status on the stack. Simultaneous interrupts are disposed of in the same manner as with a vectored interrupt system. Again, they will be serviced in order of decreasing priority until all are taken care of.

So much for theory, now let us go through the design of a custom interface for a keyboard with all of the bells and whistles, including interrupt capability usable alone or in a polled interrupt system. The keyboard to be used for illustration purposes is a Clare-Pendar unit being sold by The Digital Group and by Herbach and Rademan. This, incidentally, is a very nice keyboard with a full upper and lower case ASCII character set and reasonably good "feel". A unique feature is an "upper case" latching mode key. When on, the alphabetics always come out as upper case but the numerics and specials are not affected. When off, it acts as a standard upper and lower case typewriter keyboard.

The keyboard interface consists of three status and control flip-flops. Two of these are, of course, BUSY and DONE as described earlier. The third is an interrupt enable flip-flop that applies only to the keyboard interface. If it is off, the keyboard cannot cause interrupts. If it is on, an interrupt request is generated whenever DONE is also on as described earlier. The inclusion of an interrupt enable for each device allows both interrupt and non-interrupt I/O to be mixed. It also allows interrupts from selected devices to be inhibited momentarily if necessary. Finally, it offers compatibility with old programs that were not written to handle interrupts.

Three distinct I/O instructions are used with the keyboard interface. OUT KBSC (output to KeyBoard Status and Control) is used to control the state of the three control/status flip-flops. INP KBSC is used to read the state of the same three flip-flops. INP KBDATA is used to read the content of the keyboard output register. In our system, KBSC was assigned to I/O address 202 octal and KBDATA was assigned to 203. Since these differ by only one bit, keyboard address recognition can be simplified. On input from KBSC, bit 0 is the DONE flip-flop, bit 1 is the BUSY flip-flop, and bit 2 is the state of the interrupt enable flip-flop for the keyboard. In the polled interrupt system it is advantageous to place DONE status at bit 0 since a RAR can then be used for a quick test of DONE. On output to KBSC, three bits are significant. If bit 0 is a ONE, all three status/control flip-flops are reset. If bit 1 is a ONE, BUSY is set. If bit 2 is a one, interrupt enable is set. DONE is reset automatically when the keyboard data is read (INP KBDATA executed). Striking a key, of course, will reset BUSY and set DONE.

Appendix A lists the significant portion of keyboard initialization, common interrupt service, and keyboard interrupt service routines. The initialization routine first clears the keyboard to an idle state and then sets BUSY and keyboard interrupt enable. Finally it enables the Altair interrupt system and jumps to the background routine. Any interrupt will cause entry into the common interrupt service routine. First, A and CPU status is pushed onto the stack since the polling chain will alter these. The chain itself consists of three instructions repeated as many times as there are devices that can cause interrupts. If the keyboard is the first device found with DONE on, a jump is taken to KBSRV. Within KBSRV the remaining registers are saved on the stack and the keyboard data is read in thus resetting DONE. After processing the character, keyboard BUSY is set again, the registers are restored, and a return to the interrupted program is executed.

One important consideration when writing programs that may be interrupted is that the stack pointer cannot be fooled with. This means, for example, that subroutines cannot use INX SP and DCX SP in retrieveing arguments from the stack. The reason of course is that an interrupt may strike when the stack pointer has been temporarily moved and stack data may be destroyed by the register save/restore code in the interrupt service routine. Instead, the stack pointer must be loaded into H&L and then H&L used to access arguments on the stack. Another possibility is to disable interrupts while the stack pointer is being manipulated but this may make interrupt latency quite long.

Figure 1 shows a timing diagram for the keyboard output signals. The strobe is generated whenever a key is pressed and the output register holds the key code stable until the next key is pressed. Unfortuantely, there is no ENABLE input to the keyboard so there is nothing to prevent the operator from striking another key and changing the output register contents before the previous keystroke is acted upon. Interrupt capability, of course, reduces this problem by insuring fast response at all times. Most keyboards available to the hobbyist work or can be made to work like the Clare-Pendar unit.

The actual logic to implement the keyboard interface is shown in figure 2. For the most part it is just an application of the input and output interface concepts presented in part 1. Rather than redraw all of the bus buffers that would be required if the keyboard were the only interface on a board, the prefix BUF will be used to designate a buffered bus signal. If there are already a couple of interfaces on the board, then most, if not all of the BUF prefixed signals will already be available.

Keyboard address recognition is performed by a 7430 and two sections of a 7432 connected as NAND-NOTs. One way to understand this configuration is to consider the 7430 as recognizing whether the address pertains to the keyboard, i.e., either 202 or 203 octal. The two 7432 sections then distinguish between 202 and 203 by looking at the least significant address bit.

Emerging from the mass of gates at the top of the drawing are three signals corresponding to the three possible I/O instructions for the keyboard. The topmost goes low when address 203 (KBDATA), SINP, and PDBIN coincide and causes a set of 8T97's to gate data from the keyboard data register onto the Altair DI lines. Additionally it fires a one-shot which drives a speaker in the keyboard for an audible click when the program reads data from the keyboard. Incidentally, error beeps and other audible signals can be easily generated with this setup using simple program loops. The next lower signal goes low when address 202 (KBSC), SINP, and PDBIN are coincident. It enables another section of an 8T97 to gate the state of the three control/status flip-flops onto the DI lines for input into A. The last signal goes low when an OUT KBSC is executed.

The logic around the control/status flip-flops is actually simpler than it looks. Note that interrupt enable and BUSY are upsidedown, that is, they are ONE when Q is low and ZERO when Q is high. This saved two inverters in our own system since $\overline{\text{BUF DATA 1}}$ and $\overline{\text{BUF DATA 2}}$ were already available. The 7432-7408 combination provides reset logic for the flip-flops. First, a system reset (called $\overline{\text{PRESET}}$ in the Altair manual) can reset all three control/status flops. However an OUT KBSC with bit 0 being a ONE can also reset all of the flops. Note that the direct inputs to the flip-flops are used which means that reset will prevail if a conflicting command is given. Interrupt enable and BUSY are clocked at the trailing edge of the strobe generated during OUT KBSC. If DATA 2 is a one, then interrupt enable will be turned on at this time. Similarly, if DATA 1 is a ONE, then DONE will be turned on. ZERO data will leave the corresponding flip-flop unchanged since it is a J-K type of flop. A strobe from the keyboard will reset BUSY and set DONE. Finally, reading the keyboard data register will reset DONE by clocking a ZERO into it.

As mentioned before, an interrupt request should be given to the Altair when keyboard DONE and interrupt enable are on simultaneously. Fortunately the Altair interrupt request line, $\overline{\text{PINT}}$ pin 73, is a "wire or" line. That means that open collector gates may be tied directly to the line in order to form the logical OR of many possible interrupt requests. Normally the line is pulled high (which means "no interrupt request" since it is an inverted signal) by a resistor to +5 on the CPU board. However a device requesting an interrupt can pull it low with an open collector gate such as a 7401. There is no practical limit to the number of interrupt requests that can be tied to $\overline{\text{PINT}}$ provided the wiring is not so extensive as to pick up noise. In figure 2 a left over portion of a 7432 and a 7417 are used to pull $\overline{\text{PINT}}$ down when interrupt enable and DONE are both on.

Since figure 1 shows that keyboard data is not valid until the trailing edge of the keyboard strobe, the other half of the 74123 one-shot is used as an edge detector. The resulting narrow width pulse directly resets BUSY and sets DONE. The 7413 and discrete components form a repeat oscillator since one is not provided on the keyboard itself. When the repeat key is pressed, there is a short delay and then the oscillator starts firing the strobe single shot thus simulating multiple depressions of the last character key struck. Note that BUSY must be on for the oscillator to run. With the components shown, the delay is about 100 MS and the repeat rate is 30 per second.

In the next issue, read/write memory interfacing will be discussed. Although some may want to build ordinary memory boards using the techniques to be described, the intent is for specialized memory systems.

APPENDIX A

```
         .
         .
*        KEYBOARD INITIALIZATION
*
         MVI  A,001Q  RESET ALL CONTROL FLOPS
         OUT  KBSC       IN THE KEYBOARD
         MVI  A,006Q  SET KEYBOARD INTERRUPT ENABLE
         OUT  KBSC       AND BUSY, TURNS KEYBOARD
         .               LIGHT ON
         .
         .
         EI           ENABLE ALTAIR INTERRUPTS
         JMP  MAIN    JUMP TO MAIN BACKGROUND PROGRAM
```

```
*        COMMON INTERRUPT SERVICE ROUTINE
*
         ORG  070Q    MUST BE AT LOCATION 000:070
         PUSH PSW     SAVE A & STATUS ON THE STACK
         INP  DEV1SC  TEST DONE IN DEVICE 1
         RAR          ROTATE DONE INTO CARRY
         JC   DEV1SV  JUMP TO DEVICE 1 SERVICE ROUTINE
*                        IF ITS DONE WAS ON
         INP  KBSC    TEST DONE IN KEYBOARD
         RAR
         JC   KBSRV   JUMP TO KEYBOARD SERVICE IF ON
         INP  DEV3SC  TEST DONE IN DEVICE 3
         .
         .
         JMP  ERROR   SPURIOUS INTERRUPT, HARDWARE FAILURE
```

```
*        KEYBOARD SERVICE ROUTINE
*
KBSRV    PUSH B       SAVE REMAINING REGISTERS
         PUSH D          ON THE STACK
         PUSH H
         INP  KBDATA  GET CHARACTER FROM KEYBOARD, ALSO
*                        RESETS DONE AND CLICKS SPEAKER
*
         .            INTERPRET CHARACTER AND ACT ON IT
         .
         MVI  A,002Q  SET BUSY ON TO ENABLE FOR NEXT
         OUT  KBSC       CHARACTER
         POP  H       RESTORE REGISTERS
         POP  D
         POP  B
         POP  PSW     RESTORE A AND STATUS
         EI           ENABLE ALTAIR INTERRUPTS
         RET          RETURN TO INTERRUPTED PROGRAM
```

SURPLUS SUMMARY

There is a new publication out which is certainly worthy of note in this column. It is ON-LINE. ON-LINE is a want-ads flyer service specifically for computer nuts. Lots of good listings. ON-LINE is issued 18 times per year and goes for $3.75/year. For a free sample issue write to:

ON-LINE
24695 Santa Cruz Highway
Los Gatos, CA 95030

Notice to you folks who wanted to build graphics systems: yoke cores for the yokes that Hal Chamberlin offered have been delivered now that Stackpole is off strike. The yokes are available for $15.00 with a 2 week delivery, see issue #3 for details.

For those who are into games or just analog inputs in general, James Electronics has a joystick for $9.95. Outputs are four 100K pots, two on each axis. Write to:

James Electronics
Box 882
Belmont, CA 94002
Ph. 415/592-8097

Want to try your hand at building a CRT monitor? If so Meshna is offering an excellent kit of parts including a 9" green phosphor (P39) tube with socket, flyback, HV rectifier, HV cap, magnetic shield, and deflection yoke for $20.

MESHNA
Box 62
East Lynn, MA 01904

ALTAIR 8800 BUS SIGNALS

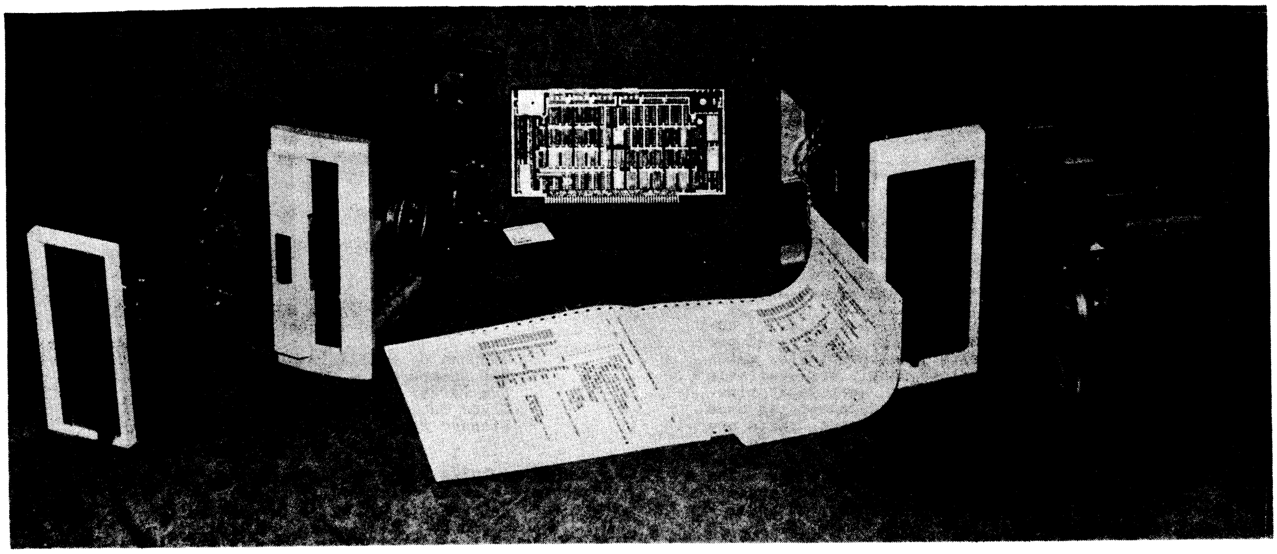
TCH has had one unusual but worthwhile request since our last issue. Two readers have asked that we list all the signals used on the Altair backplane. Their purpose is simple, though they do not own or plan to own Altairs, they would like to utilize some of the plug compatible boards being offered. The bulk of the signals would eventually be explained in the Altair interfacing series but for the sake of conciseness and convenience here they are:

PIN	NAME	DESCRIPTION
1	+8V	Unregulated input to 7805 regulators
2	+16v	Unregulated input to +12 regulators
3	XRDY	Anded with PRDY and goes to 8080 RDY
4	VI0	Vectored interrupt request 0
5	VI1	Vectored interrupt request 1
6	VI2	Vectored interrupt request 2
7	VI3	Vectored interrupt request 3
8	VI4	Vectored interrupt request 4
9	VI5	Vectored interrupt request 5
10	VI6	Vectored interrupt request 6
11	VI7	Vectored interrupt request 7
18	STA DSB	Status buffer disable
19	C/C DSB	Command/control buffer disable
20	UNPROT	Input to memory protect circuitry on mem bd
21	SS	Indicates machine is in single step mode
22	ADD DSB	Address buffer disable
23	DO DSB	Data out (from CPU) buffer disable
24	02	Phase two clock TTL levels
25	01	Phase one clock TTL levels
26	PHLDA	Hold acknowledge, buffered 8080 output
27	PWAIT	Wait acknowledge, buffered 8080 output
28	PINTE	Interrupt enable, buffered 8080 output
29	A5	Buffered address line 5 (32)
30	A4	Buffered address line 4 (16)
31	A3	Buffered address line 3 (8)
32	A15	Buffered address line 15 (32768)
33	A12	Buffered address line 12 (4096)
34	A9	Buffered address line 9 (512)
35	DO1	Buffered data out line 1
36	DO0	Buffered data out line 0, least sig. bit
37	A10	Buffered address line 10 (1024)
38	DO4	Buffered data out line 4
39	DO5	Buffered data out line 5
40	DO6	Buffered data out line 6
41	DI2	Data input line 2
42	DI3	Data input line 3
43	DI7	Data input line 7, most sig. bit
44	SM1	Latched 8080 M1 status
45	SOUT	Latched 8080 OUT status
46	SINP	Latched 8080 INP status
47	SMEMR	Latched 8080 MEMR status
48	SHLTA	Latched 8080 HLTA status
49	CLOCK	2 mHz clock, crystal controlled
50	GND	Logic and power ground return
51	+8V	Unregulated input to 7805 regulators
52	-16V	Unregulated input to negative regulators
53	SSW DSB	Sense switch disable (special for console)
54	EXT CLR	Clear signal for I/O devices
68	MWRT	Write enable signal for memory
69	PS	Indicates if addressed memory is protected
70	PROT	Input to memory protect circuitry on mem bd
71	RUN	Indicates machine is in run mode
72	PRDY	Anded with XRDY and goes to 8080 RDY
73	PINT	Input to 8080 interrupt request
74	PHOLD	Input to 8080 hold request
75	PRESET	Clear signal for CPU
76	PSYNC	Buffered 8080 SYNC signal
77	PWR	Buffered 8080 write enable signal
78	PDBIN	Buffered 8080 DBIN signal
79	A0	Buffered address line 0 (1)
80	A1	Buffered address line 1 (2)
81	A2	Buffered address line 2 (4)
82	A6	Buffered address line 6 (64)
83	A7	Buffered address line 7 (128)
84	A8	Buffered address line 8 (256)
85	A13	Buffered address line 13 (8192)
86	A14	Buffered address line 14 (16384)
87	A11	Buffered address line 11 (2048)
88	DO2	Buffered data out line 2
89	DO3	Buffered data out line 3
90	DO7	Buffered data out line 7, most sig. bit
91	DI4	Data input line 4
92	DI5	Data input line 5
93	DI6	Data input line 6
94	DI1	Data input line 1
95	DI0	Data input line 0, least sig. bit
96	SINTA	Latched 8080 INTA status
97	SWO	Latched 8080 WO status
98	SSTACK	Latched 8080 STACK status
99	POC	Clear signal during power-up
100	GND	Logic and power ground return

THE COMPUTER HOBBYIST

VOLUME I NUMBER 10

AUGUST 1976



The Complete System! TCH Super Simple Interface, Your Favorite Floppy Disk Drive, and Disk Read/Write Software.

TCH SUPER SIMPLE FLOPPY DISK INTERFACE Part 2.

Judging from the response to part 1 of this series there seems to be a great deal of interest in floppy disk systems. In this issue we will give a complete schematic of the interface along with a full technical description. Also basic programming concepts will be discussed but the complete listing of the 8080 support software will be held until part 3. A printed circuit board is being layed out and 1702/5203 PROM programming will be available but neither will be formally announced until part 3 is published. Please do not order boards or programming until they are formally announced.

Although the controller may be easily wire-wrapped, many readers would much rather have a printed circuit board to cut down on construction effort and minimize the possibility of wiring errors. Thus begins the delimma of choosing a board size, interface configuration, etc. Our usual policy is to make projects operable and equally applicable to any kind of computer from a 4004 on up. On the other hand, a vast majority of readers requested a plug-in Altair (or IMSAI or Polymorphic, etc.) compatible board to even further minimize expense and wiring effort.

After much debate, the following compromise emerged. The circuit supplied on the interface board and the board dimensions would be 100% Altair compatible. Jumpers would be available to adapt the card to nearly any manufacturer's floppy disk drive. Also sockets and addressing logic for two 1702 erasable PROM's would be provided so that the driver software could actually be a part of the interface. In order to provide universal applicability to any computer, the circuitry would be restricted to cover about 75% of the board area. The remaining space would have socket patterns for any additional logic necessary to interface the unit to another computer's bus. Alternatively, connection to a single input and output port (8 bits on input and 4 bits on output) can be accomplished directly. Packaging in the latter case would probably consist of mounting the controller card in a separate case along with the floppy drive and its power supply.

What this amounts to is a very convenient and inexpensive installation for the majority of readers. The remaining readers should not find installation any more difficult than if a "non-specific" board layout such as was done with the audio cassette interface had been used. Although microcomputer bus structures differ widely in details, essentially the same logic functions are required to interface to any of them. Board layout will be such that I/O pin assignments can be easily changed or the edge connector cut off and direct wire connections used instead. While specific instructions cannot be given for most computers (we have no way of testing them), the discussion and explanation of how the Altair interface works should give the reader a pretty good idea of how to connect the unit to another computer. Readers are encouraged to send in connection details and software used in their own non-Altair installations.

Here is a short summary of the interface characteristics. Jumpers are available to adapt the interface to any of the floppy disk drives listed in Appendix 1. Other drives may require the addition of a slight amount of circuitry to meet their signal requirements. The interface can handle either one or two floppy disk drives. For two drive operation, drives with a so-called "daisy chain" interface are required. This means that there is a DRIVE ENABLE line that enables the drive to accept commands and to return status. Thus, two drives may be connected in parallel except for the DRIVE ENABLE line. Two drives without this feature could also be supported but additional multiplexing circuitry would have to be constructed in the free IC matrix area. It should also be possible to support two drives of different make but additional circuitry may again be required.

The interface board contains a data separator and a sector separator. The former decodes the double frequency waveform from the drive read electronics into data and clock pulses. The latter separates the index hole pulse from the sector hole pulses. These circuits require adjustment however which can only be done with the aid of a good oscilloscope. If possible, the reader should try to acquire a disk drive with a built-in data separator and sector separator. This means that any needed adjustments were made at the factory. Also, many drives utilize complex separation circuitry that outperforms the simple one-shot circuits utilized in this interface.

CONTINUED ON 4

EDITORIAL by Hal Chamberlin

What is the first thing that comes into your mind when you see a commercial product price list that looks like this: 1-4 $X, 5-24 $.9X, 25-49 $.75X, 50-99 $.65X, 100-up $.5X? Group purchase, of course! The enticement of that "100-up" figure is directly proportional to the value of X. Unfortunately, the chances of success with a group purchase seem to be inversely proportional to X^2.

Before getting into the economic dynamics of group purchasing, let's try to figure out why the price list is the way it is. One obvious and often cited reason is that the proportional cost of paperwork is less if one order is for 50 units instead of 50 orders for one unit. This is certainly true, however it cannot explain the often hundreds of dollars per unit discount on big ticket items such as printers or disk drives. Further, many OEM manufacturers base their discounts on yearly contracts with multiple small quantity releases allowed spread throughout the year.

A related reason for discounts is the per-unit cost for a salesman's time. Great effort often must be expended in selling an industrial customer on the merits of a device against its competitors.

Another significant cost is field service support. With a large customer, a couple of phone calls may solve a design related problem on 50 units at once. With "oneseytwosey" customers, dozens of individual contacts with a variety of people would be necessary to service an equal number of devices.

The major portion of the larger discounts is, I believe, related to production scheduling. The cost per unit will be much lower if raw materials inventories match demand, the correct number of trained people are available (excess manpower is expensive but so is overtime at time-and-a-half), and the proper automated equipment is used full-time. Remember also that many component parts are bought from price lists that read much like the one in the first paragraph. The ideal situation is an exact knowledge of production requirements up to a year ahead. The chances of a wrong guess are much greater with today's volitile economy and the cost of a wrong guess is equally high with today's interest rates. Also note that list prices are maximums. If the order is very large or follow-on business has the potential of becoming large, further price reductions are likely through negotiation.

So to justify that low 100-up price the supplier is looking to process one order, unsolicited by his sales force, single party responsibility for field service, firm commitments on quantities and delivery dates, and a high probability of repeat business. A successful group purchase will have to give the supplier at least some of these expectations.

Now let's look at the other side of the coin, the group purchase organizer. There is a big difference between drumming up an order for 100 type 2102 memory chips at $1.60 each and getting together 50 orders for LSI-11 CPU's at $650 each. About $32,340 worth of difference. That's big money requiring professional legal and accounting help. Further, no one wants to be left holding that size of bag and the vendor will make sure that he doesn't. All too often the organizer gets cold feet when the orders start coming in and the deal starts taking shape. The vendor starts asking embarassing questions regarding the factors mentioned earlier, particularly field service.

Now let's assume the existence of an organization big enough to tackle a group purchase of a big-ticket item, such as the Southern California Computer Society. For an example I will use the recent LSI-11 group purchase. One quantity price, $990; 50 quantity price, $653; with SCCS overhead, $660. Except for one item, the planning and execution was perfect; big organization, professional help, frank discussions with DEC at the outset, and extensive publicity. Incidentally, individual orders were firmed by a $100 deposit requirement, refundable if the deal fell through.

The sequence of events was roughly as follows: Summer 1975 announcement of plan; first surge of inquiries and orders built confidence; October 31 deadline and not enough orders; order rate trails off short of goal; no communication with initial participants; passage of time (4 months); commercial broker offers LSI-11's for a substantial discount though not as great as SCCS; no communication; some participants threaten or execute withdrawal (deposit checks were not cashed); SCCS admits failure, privately; LSI-11's procured at higher price from broker and distributed to remaining participants.

Note that two failure modes were operating. One, a simple shortfall of interest, which is so obvious as to be often overlooked. The other is the passage of so much time that market conditions had changed and respondants took advantage of the changed market. Lack of communication did little to ease the wait and maintain confidence. Worst of all, the failure of this superbly organized group purchase will greatly discourage both organizers and participants in future efforts.

But let's not let the preceeding discussion discourage group purchases altogether. They can be successful if certain "rules of thumb" are followed. First, don't make it too large. A circle of a half-dozen friends is ideal, certainly no larger than the local club. Remember, the organizer will be responsible for defective unit returns, etc. and should be within easy reach of all participants. Second, get it organized and executed quickly. Time from announcement to order placing should be a very few weeks maximum. If more time is needed to publicize and gather orders, the deal is probably too big. Third, it should be understood by all participants that orders are binding. This policy may reduce interest but will prevent a possible fiasco when collection time rolls around.

So where does that leave the big deals, the ones that can save substantial amounts of money? The LSI-11 effort could not have been any smaller (the $990 price was for 1-49 units). If the price differential is large enough, brokers and distributors will step in, accept the responsibilities, invest a quantity of money, and pocket a portion of the differential for their efforts. This functioning of the free enterprise system is good for it benefits everybody involved.

THE COMPUTER HOBBYIST
Founded October, 1974

Stephen C. Stallings - managing editor
Hal Chamberlin - Contributing editor
Jim Parker - Contributor
Clyde Butler - Photographer
Richard Smith - Programming consultant

THE COMPUTER HOBBYIST is published at intervals approaching monthly near Raleigh, N.C. The subscription rate for 12 issues is $6.00 for persons in the U.S.A. and U.S. possessions. For Canada and Mexico the rate is $7.50 for 12 issues. Subscriptions in all other countries are $11.50 for surface mail and $16.00 for air mail. All back issues are available and cost 65 cents each for persons in the U.S.A., U.S. possessions, Canada, or Mexico. For all other countries back issues cost $1.00 each by surface mail or $1.50 each by air mail. Remittances from countries other than the U.S.A. should be in the form of a bank or postal money order.

THE COMPUTER HOBBYIST is seeking paid contributors. Material to be submitted should be typed or neatly written and must not appear to be soliciting business for any firm. For details, please inquire.

Advertising space is available to interested firms. Please inquire for rates and formats

All correspondence should be addressed to:

THE COMPUTER HOBBYIST
Box 295
Cary, NC 27511

LETTERS TO THE EDITOR

Dear TCH,

I am thrilled by your idea of a super simple floppy disk. But don't stop at PC cards!! Why not consider a group purchase of disk drives? $450 is a very attractive price! Or perhaps the Southern California Computer Society (being so large) would be willing to co-ordinate it.

Keep up the superb work.

Judson B. Ellmers

Dear Sirs:

I read with interest the floppy disk article in the last TCH. I think before I would be interested in PC boards and programming, I would be interested in finding out if there is a current group purchase effort ongoing. Just the drive is still a big ticket item for most people, and I suspect most potential users would jump at the chance to save $100 to $300. If SCCS or some other club does not step in and volunteer, why not consider organizing such a purchase?

Elmer Beachley

CONTINUED ON 3

Gentlemen,
In response to your article on the floppy disk interface, I think its a great idea! A price of $650 for the drive however is beyond my range. If we could coordinate a bulk-rate deal from the manufacturers to shave off $200 or so, I would seriously consider going that route.

As it stands, though, your tape interface will have to do.

Bob Walden

Gentlemen:
You wanted feedback on the floppy disk article. Here it is:

1. The article was great!
2. How about a dual floppy?
3. I would like a group purchase so we all could have a $400 floppy
4. How about a (nearly) complete kit (floppy, PC board, Altair interface)

Like most of your readers, I wish you could write more sooner, however you don't print garbage so it is well worth the wait. Keep up the good work.

Ken McGinnis, MD

Dear Sirs:
With reference to requests for feedback on floppies - yes, I am interested in boards, PROM's, etc. for a floppy I/F. Also how about a group buy of drives from somebody? Also some words (at least) about dual or even quad floppy disk systems, i.e. what kind of increased controller complexity, power requirements, etc.

Floppies are the only way to go for mass storage for any reasonably sophisticated application, so keep it coming!

Dave Warner

Gentlemen:
I definitely am interested in your floppy disk interface. I will want PC boards if they can be made available (for Altair preferably). I would also be interested in PROM programming assuming your locations would fit in my system. Also, any chance of a group purchase on drives to get the individual cost down?

Dr. Kelley

Gentlemen:
I'm very interested in board for floppy disk controller. I suggest you include a wire-wrap area on the board so it can be customized. I have 2 drives (Memorex 651, the original of all the others) and will want to do multiplexing.

Del Blevins

Gentlemen:
Yours is undoubtedly the best hobbyist computer publication! Now follow up with some ideas that will popularize and standardize your floppy disk.

1. Make the drives available through SCCS group purchase
2. Make the interface electronics on an Altair plug-in PC board
3. Push your floppy in the other hobbyist computer mags to make it a (semi) standard. This is where the tape unit ran out of steam!

Gary Alevy

Dear Sirs:
I am interested in the floppy disk interface cards if they become available, also PROM programs. Can a group order on the drives be arranged; if so I'll sign for 4!!

Keep up the great work.

Jerry Hewett

Gentlemen:
I vote yes for floppy disk PC boards & PROM programming.

B. R. Brunson

This is only a sample of the boxfull of mail we received concerning the super simple floppy disk interface. We took most of the suggestions received into consideration when designing the final version of the interface which is described in the feature article of this issue.

The group purchase is another matter however (see editorial). One new entrant (General Systems International, GSI) into the disk drive field greatly reduces the advantage of a group purchase by virtue of their flat discount curve. Single quantity price on an excellent drive (we bought one and it is indeed excellent) is $550. Also, at the recent Computerfest in Cleveland, one manufacturer's booth was selling brand new, completely compatible drives for $350 each and another was selling them for $400!

CONTINUED ON 10

CONTINUED FROM 1

An on-board crystal oscillator is provided for non-Altair installations or where an accurate 2mHz source is not available. The oscillator uses a 4mHz crystal which is cheaper and smaller than a 2mHz unit. If a 2mHz clock is available, the parts associated with the oscillator may be omitted.

Also provided on the interface board is a "power on reset" circuit. This prevents the interface from coming up in a write mode for example. When installed in an Altair, the console reset switch will also reset the interface to an idle state when actuated.

The key interface element is the 2048 bit buffer. Associated with the buffer is an 11 bit "buffer pointer" register. The contents of the buffer pointer point to a location in the buffer itself. Commands are available to increment the pointer and to reset it to the beginning of the buffer. Additional commands can be used to write a ONE or a ZERO into the buffer location pointed to by the pointer. When a disk sector is read or written, the bits stream either to or from the buffer by means of hardware controlled automatic incrementing of the pointer after every bit transfer. Two ordinary, cheap 2102 1K RAM's are used in the buffer. Only about 1300 bits are actually needed for a full sector including overhead.

The input port associated with the disk interface consists of 8 "status" bits. Bit 0 indicates the state of the buffer bit currently pointed to by the buffer pointer. Bit 1 is a ONE if the head is currently positioned at track 0; it is a ZERO if the head is at any other position. Bit 2 is a ONE when the index pulse from the disk is detected. Bit 3 likewise senses sector pulses. If the disk drive has the write protect hole sensor and the write protect hole on the disk is not covered up, bit 4 will be a ONE indicating that the disk is write protected. Otherwise it will be a ZERO indicating that writing is permitted. The write protect sensor is usually a $10 to $25 option. Some drives have a "watchdog monitor" circuit that senses illegal commands or drive malfunctions. If this circuit is triggered, it sets a flip-flop in the drive that locks out further operation until it is reset. Bit 5 is connected to this flip-flop and reads as a ONE if it had been set. The bit will always be ZERO if everything is OK or if the drive lacks this feature. Bit 6 is a ONE if the disk drive is ready. The drive is ready only if power is applied, a diskette is loaded, and the door is closed. Most drives include a speed sensor circuit so that ready status is not posted until the disk is up to speed.

All of the preceeding bits except bit 0 indicate the status of the currently selected drive in a dual drive system. Bit 7 is a "select readback" which indicates which disk drive is currently selected. If drive 0 is selected, it is a ZERO, otherwise it is a ONE if drive 1 is selected.

The output port assigned to the disk interface works in a somewhat strange manner. Only the 4 low-order bits (bits 0-3) are used. These 4 bits can specify one of 16 possible commands to the interface. To execute a command, the program would send the appropriate 4 bit command code to the disk interface output port address.

Command codes 0 and 1 are used to store a 0 or a 1 respectively into the buffer at the current location. After the bit is stored, the pointer is automatically incremented to point to the next buffer position. Command code 2 causes the head motor to step out toward the edge of the disk one track position. Command code 3 likewise does a "step in" operation. Code 5 sets the head load flip-flop which presses the head against the disk in preparation for a data transfer. Code 4 resets this flip-flop. Codes 6 and 7 reset and set respectively the "above track 43" flip-flop which controls the signal amplitude during write. Less amplitude is needed for the inside tracks where data is packed more densely. Not all drives actually need this signal however.

Command code 8 is called execute read. When issued, the interface latches the command until the leading edge of the next sector pulse is detected. At this time the bit stream from the disk is directed to the buffer which stores it. The transfer is stopped when another sector pulse is seen. Command code 9, execute write, works in a similar manner except that the bit stream is read from the buffer and written onto the disk. In both cases, hardware controls the exact point where reading and writing is started and stopped thus eliminating concern over exact software response times.

Command code 10 resets the buffer pointer to the beginning of the buffer. Code 11 increments the pointer, Neither command alters the information stored in the buffer. Command code 12 is used to reset the fault latch present in some disk drives. It should only be issed in response to a fault indication in bit 5 of the disk interface status byte. Code 13 is a spare which is not used by the printed interface circuitry. Code 14 causes drive 0 to be selected and code 15 causes drive 1 to be selected. Although drive selection doesn't matter in a single drive installation, the software should still explicitly select drive 0 to allow for compatibility with future expansion.

Now lets run through some disk drive control sequences in order to better understand interface operation and programming requirements. A fundamental task is positioning the head to a desired track number. Positioning to track 0 is easy. First the track 0 status bit should be examined. If it is a ONE, the head is already at track 0 and the operation is finished. If not, a command code 2 should be issued to step the head one position toward track 0. Since the stepping motor in the drive does not respond instantly, a 10 millisecond delay should be allowed before testing track 0 again or commanding another step. The best way to provide this delay independent of CPU speed or external hardware is to wait for two sector pulses to pass the sector sensor. After the delay, the program would loop back and test for track 0 again and take another step if necessary.

If a track number other than 0 is desired, it is necessary to know what track the head is currently at. Then the current track is subtracted from the desired track. If the result is negative, step out commands (code 2) are needed. If the result is positive, step in commands (code 3) are needed. A zero result of course means that the head is already at the desired track. The number of step commands needed is the absolute value of the difference. As before, 10 milliseconds (two sector times) should be allowed between step commands. Also, an additional 10 milliseconds (20 total) should be allowed after the last step for the motor assembly to stop vibrating.

Whenever a new program is loaded, part of the initialization procedure should be to move the head to track 0 and store 0 in the memory location used to hold the current head position. This will insure that the current head position assumed by the program agrees with the actual head position. In a two drive system, a current head position will have to be maintained for each drive.

After the head is at the correct track, it is still necessary to do the read or write operation in the proper sector. The most straightforward approach is to wait until an index pulse is detected and then count the appropriate number of sector pulses. When the head is scanning the sector just before the desired sector, the execute read or execute write command should be issued. After two more sector pulses have passed, the operation is complete. Note that sectors are numbered consecutively with the index pulse occurring in the middle of sector 0. This means that to read or write in sector zero, 31 sector pulses will have to be counted which is nearly a full disk revolution.

Of course the head must be loaded against the disk before a read or a write can be done. In order to minimize disk and head wear, the head is normally lifted during idle periods. Head load times (the time between the head load command and when reading or writing can be done) vary but 40 milliseconds (8 sector times) is generally adequate. This time may be overlapped with other preparatory operations such as stepping or waiting for the desired sector to come around. However, adequate time must be insured in cases where read or write preparation may require less than 8 sector times.

Before a write operation can be performed, the correct bit pattern must be set up in the buffer. The sector data format to be used is quite similar to that employed in the TCH audio cassette system. First there are 128 bits of ZEROES used in this case to synchronize the data separator during read and to allow for variation in sector sensor placement. Following the leading ZEROES is an 8 bit "start of data" ID pattern which is a constant 10010011. Eight bits is generally adequate in floppy disk applications since there is much less garbage at the beginning of a sector to reject than in an audio cassette application. Following the ID are 1024 data bits organized as 128 bytes. The most significant bit of the byte is written first. After the data are 16 bits of cyclic redundancy check (CRC). The CRC is computed and written in the same way as in the audio cassette system (see issue #5). The difference is that the CRC register is initialized to sector number (lower byte) and track number (upper byte) instead of all ZEROES. Trailing ZEROES fill out the remainder of the sector. The disk software is responsible for creating this data pattern in the buffer prior to issuing the execute write command. Note that the buffer pointer must be reset to the beginning of the data pattern before the write is executed.

When reading a sector back, the raw data bits from the entire sector are stored in the buffer starting from the current buffer position. Generally about 1300 bits are transferred. The program should reset the buffer pointer and start scanning the buffer contents until the 8 bit start of data ID pattern is located. If more than a couple hundred bits are scanned without seeing the ID, an error should be signalled since either the ID has been obliterated or the sector has never been written on. It is a good idea to clear the buffer before reading because otherwise a blank sector will leave the buffer contents unchanged and the read routine will process data left over from the previous operation.

After the ID has been located, the CRC should be initialized to the track and sector number just read. Then groups of 8 bits are taken from the buffer, assembled into bytes, stored into memory, and combined with the CRC until 128 bytes have been processed. The two CRC bytes should then be combined with the CRC and the CRC checked for for a ZERO result. If the CRC is ZERO, everything is OK. If it is non-zero, either a data transfer error has occurred or the track and/or sector just processed is not the one desired. Errors of the latter type should never occur if the drive is in good mechanical condition.

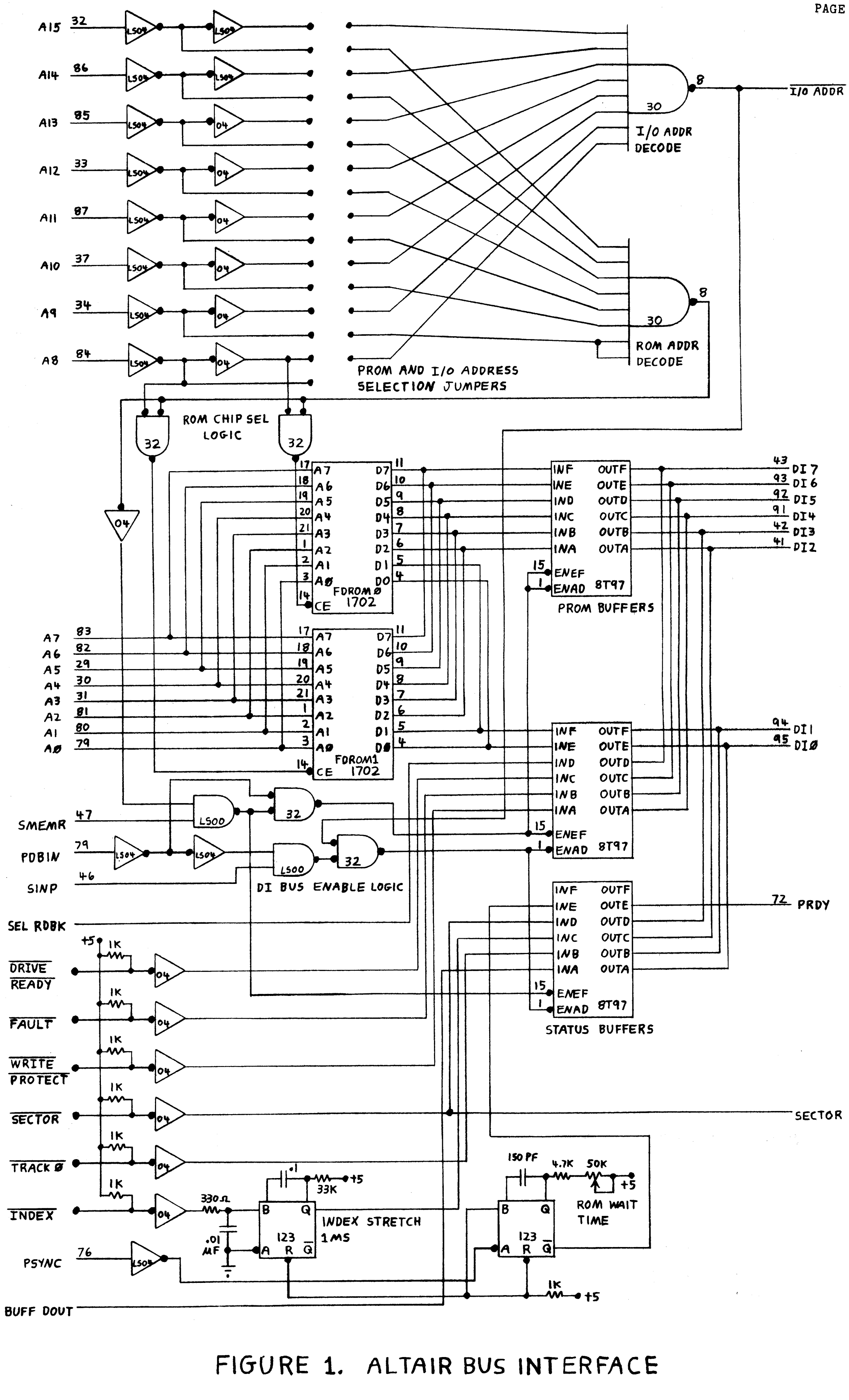


FIGURE 1. ALTAIR BUS INTERFACE

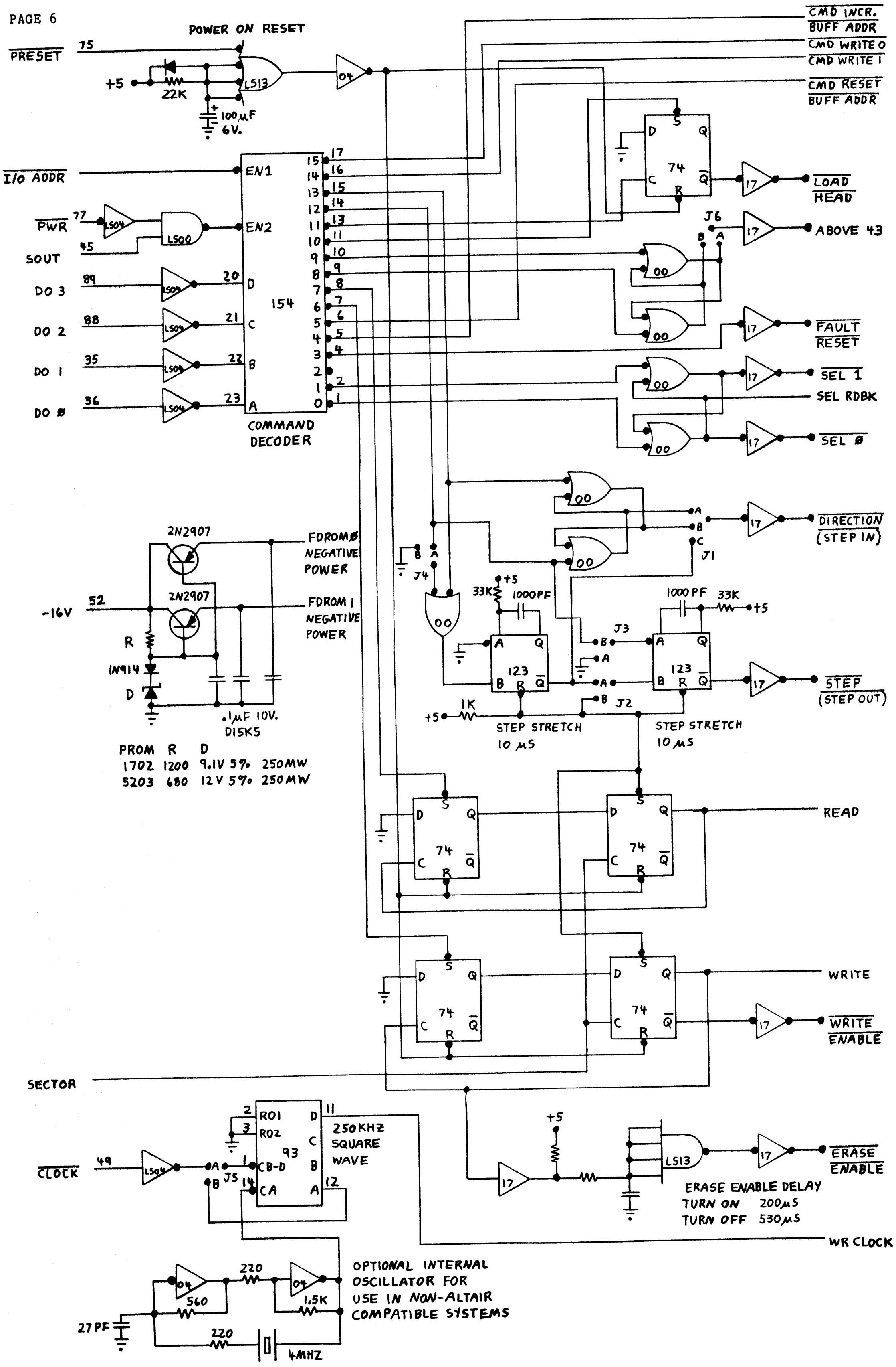


FIGURE 2. COMMAND LOGIC

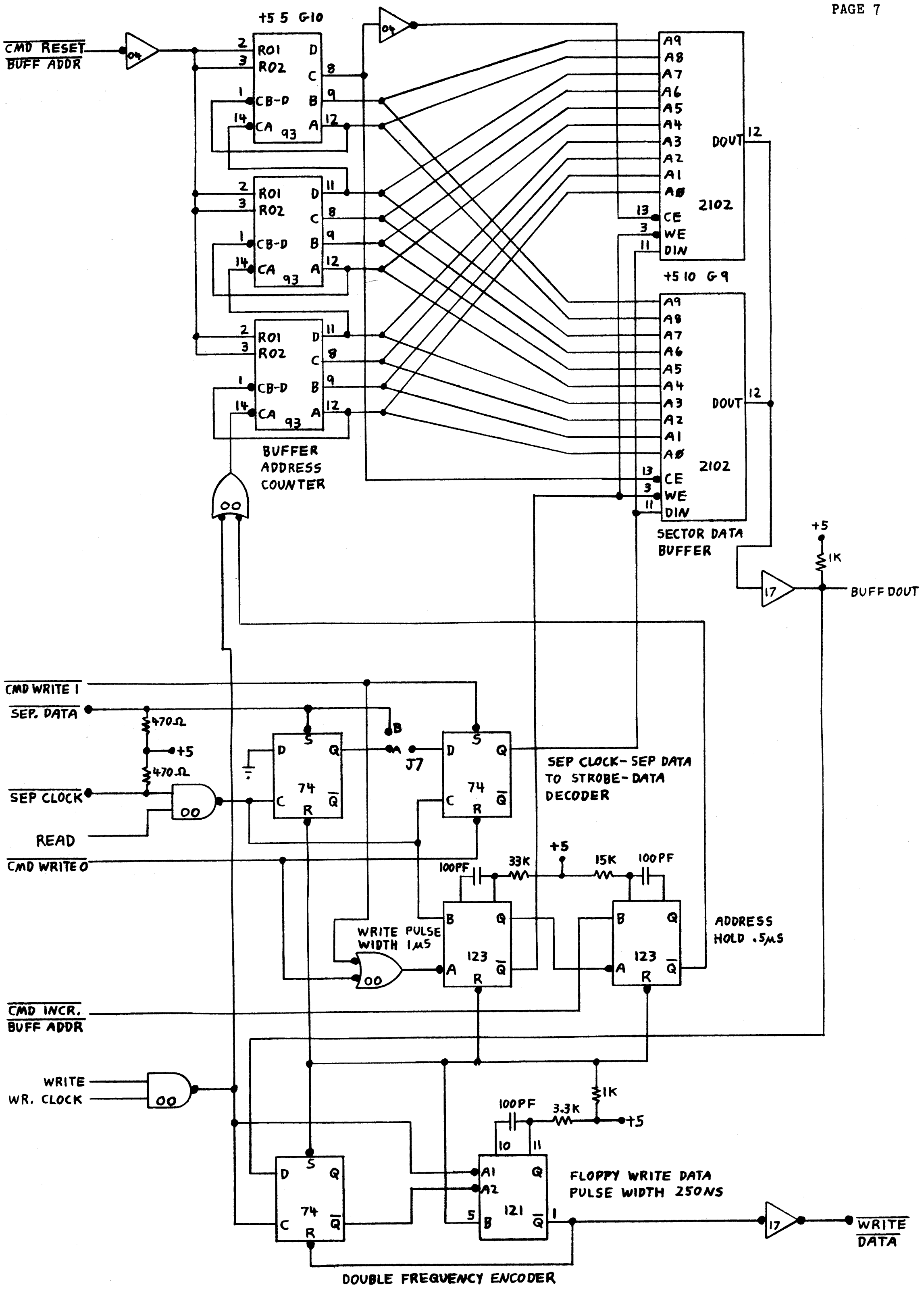


FIGURE 3. BUFFER CONTROL LOGIC

The preceeding discussion covers fairly well the basic programming considerations for the interface. Obviously, what has been saved in hardware complexity has shown up as added software complexity. The interface board has sockets and address decoding for two 1702 type erasable PROMS for a total of 512 bytes. The basic disk software can easily fit into these PROM's and relieve the user of detailed disk programming. This is particularly important when bringing up the system after power on. An "initial program load" routine in the PROM can automatically read track 0 sector 0 from the disk into memory and branch to it. Only three bytes need to be toggled in for a branch to the IPL routine. The remainder of the disk ROM software provides a read and a write subroutine which takes care of all of the operations described. The user need only load the track number, sector number, and memory address into registers and call the appropriate routine.

One problem with a completely ROM based disk support package is that the current track position for each drive needs to be saved somewhere between calls. One possibility is to have the calling routine pass the current track number as an argument and save it in read/write memory on return. However it would be much more convenient if the support package could somehow keep track of the track positions itself without using dedicated RAM locations. The solution employed is to use the buffer in the interface itself to save the current track positions. The track numbers are simply serialized into 8 bits each and saved in the first 16 buffer locations. The current track position values and the disk drives are initialized by calling a control subroutine in the package and requesting a seek to track 0 on each drive. More details on the floppy disk support package will be given in part 3 along with a complete assembled listing.

Now let us take a look at the interface schematic diagram. Several Altair interface concepts that have already been covered in detail in issues #8 and 9 will only be briefly described here. See the referenced issues for a more thorough discussion. Signals that go offpage and have only a name, mate with similarly named signals on other pages. Offpage signals which have a number shown are Altair bus signals. Signals which have large dots at their entry/exit points are generally connections to the disk drive cable.

The address decoding logic is shown at the top of figure 1. The high 8 address lines are double inverted to provide buffered true and complement forms of each bit. Jumpers are placed between the 7430 inputs and true or complement address bits to select the I/O and ROM addresses. The top 7430 decodes the input and output port address assigned to the disk interface. Note that the input and output port addresses must be the same. The other 7430 selects a pair of page addresses for the 512 bytes of PROM. The ROM page addresses must be an adjacent even-odd pair such as 374 and 375 octal. Odd-even pairs such as 375 and 376 cannot be properly decoded.

Two 7432 NAND-NOT gates are used to enable one ROM or the other if the ROM is addressed. The least significant address bit determines which ROM is enabled. Both ROM's are disabled when not addressed thus reducing the power drawn from the negative Altair supply. Eight bits of 8T97 (or 8097 or 74367) are enabled by the coincidence of ROM addressed, SMEMR and PDBIN which gates the ROM data onto the Altair DIN bus. A wait delay single-shot is fired by PSYNC and gated onto the PRDY line by the coincidence of ROM addressed and SMEMR. The single-shot may be adjusted for 0 to 5 wait states which will accomodate PROM's as slow as 3 microseconds. Since the speed of surplus erasable PROMS is seldom known, the single-shot may be adjusted for the fastest reliable operation.

Also on this page is the disk status and input port logic. Status signals from the disk drive are usually in inverted form and require pullup resistors. The 1K resistor value shown is somewhat larger than that recommended by most drive manufacturers. If the disk cable is kept to less than 5 feet, a fair fraction of an amp can be saved by using 1K pullups. The sector separators on some drives put out a very short index pulse. This is stretched to approximately 1 millisecond by a one-shot to insure that the program sees it. In a dual drive installation, all of the output signals from the drive are tied together. The drive select line then determines which drive is enabled to send its status back to the interface. Eight more 8T97 sections gate the status bits onto the DI bus and are enabled by the coincidence of I/O address, SINP, and PDBIN.

The reset signal generator is shown at the top of figure 2. This circuit generates a reset when power is turned on or when the PRESET bus line is held low by the front panel reset switch. At power on time, the 100uF capacitor is not charged and holds the 74LS13 inputs low. The capacitor gradually charges toward +5 and eventually crosses the upper threshold of the Schmidt trigger thus releasing the reset. The diode prevents the capacitor from discharging through the gate inputs when power is removed.

A 74154 decoder converts the command code sent to the output port into one of 16 short pulses. Since the bus DO buffering also inverts the 4 command code bits, the decoder outputs are backwards. Code 0 appears as a pulse at the "15" output of the decoder, 1 at 14, etc.

The commands associated with write current amplitude switching and drive selection simply operate set-reset flip-flops constructed from NAND gates. The head load flop is a 7474 and can be set by command. It can also be reset by command or by the reset signal generator. Type 7417 high current buffers are used to drive the disk input signal lines which can require as much as 40MA each on some types of disk drives. The fault reset command is simply buffered and sent directly to the drive.

The logic associated with stepping the head to different tracks is more complex in order to accomodate a variety of disk drives. Many drives have "step in" and "step out" control inputs. For this type of drive jumper 1 is set to position C, jumper 2 to position B, jumper 3 to position B, and jumper 4 is set to position B. With the jumpers in these positions, the two single shots merely stretch the .5 microsecond pulses from the command decoder into 10 microsecond pulses required by the disk drive.

Other drive types have a "step" line and a "direction" line. To move the head, first the direction signal is set up and then the step signal is activated to make the head move in the selected direction. For this type of drive, jumper 1 is set to position A or B, jumper 2 to A, jumper 3 to A, and jumper 4 to position A. When in these positions, the flip-flop formed from two 7400's remembers which direction the step command was for and drives the direction line to the disk drive. The left single-shot delays the actual step pulse for 10 microseconds to allow time for the direction select to set up in the drive. The right single-shot gives a 10 microsecond step pulse to the drive after the delay. Jumper 1 positions A and B select the polarity of direction select to match the disk drive being used.

Four sections of 7474 flip-flops are associated with the execute read and execute write commands. When the command is decoded, the left flip-flop of the corresponding pair is set. This then conditions the right flip-flop to be set on the leading edge of the next sector pulse. When it is set, the left flip-flop is immediately turned off. The result is that the READ or WRITE signals generated by this logic are on for exactly the full sector time of the sector immediately following the one during which the command was issued. All of these flip-flops are forced off during reset to prevent spurious commands during power up.

The WRITE ENABLE signal to the disk drive is active while WRITE is true. This causes current to be passed through the read/write head which allows writing of new data while the strong magnetic field simultaneously erases old data.

In order to insure some blank space between data tracks to prevent possible overlap, the disk head has two "trim erase" gaps that run parallel to the recorded track along each edge. These gaps trail the read/write gap somewhat and are controlled by the ERASE ENABLE signal. It is necessary to delay ERASE ENABLE some so that only the newly written data is trimmed off. The 74LS13 and associated buffers provides the proper delay to compensate for the gap spacing. The 5% tolerance on the two resistors and capacitor is important to insure proper trim erasing. Also, the capacitor should be a mylar or other temperature stable type; ceramic types should not be used. Some drives have the the trim erase delay built-in and factory adjusted.

The write circuitry requires an accurate, perfect square wave at the bit frequency of 250kHz. In Altair systems, an accurate 2mHz signal is available from the bus. This signal is divided by 8 with a 7493 counter to provide a 250kHz square wave. The J5 jumper may be moved to position B and the components for the internal oscillator installed if a 2mHz source is not available. The crystal should be well within 1% of 4mHz.

The top of figure 3 shows the buffer and buffer pointer register which is an 11 bit counter. The buffer is made of two 2102 static memory chips configured as 2048 words of 1 bit each. The address inputs on the 2102's are connected to the least significant 10 bits of the buffer pointer. The eleventh bit selects one RAM or the other through the chip enable input. A 7417 is used to buffer the memory data output since more than one TTL load is being driven.

The buffer pointer counter is built with 7493 ripple counters. It may be reset with a reset buffer pulse from the command decoder. A 7400 acting as an OR-NOT allows either of two signals from the read/write logic to increment the counter.

The remaining circuitry is probably the most difficult to understand part of the interface. Two D-type flip-flops and two single-shots are used to decode data read from the disk and to control writing into the buffer. When reading from the disk, separated clock pulses clock both flip-flops. The clock pulses are gated by the execute read command so that they do not affect the circuit at other times. Separated data pulses, which are present only for ONES, can directly set the first flip-flop. If a ONE is read from the disk, the data pulse sets the first flop and the next clock pulse transfers the ONE to the second flop while simultaneously resetting the first flop. If a ZERO is read, the first flip-flop remains a ZERO which is transferred on the next clock pulse as before. The clock pulses also trigger a buffer write sequence generator made from two one-shots. The

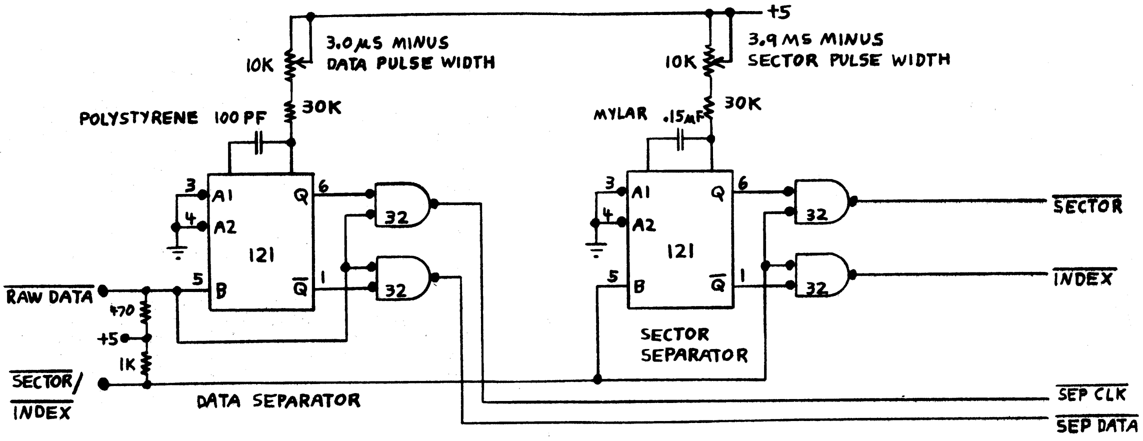


FIGURE 4. DATA AND SECTOR SEPARATORS

first one-shot connects to write enable on the 2102 RAM's and allows them to accept data from the second flip-flop described above. The second single-shot delays incrementing of the buffer pointer until well after the write operation. The timing used is slow enough to allow use of any kind or speed of 2102's.

The write 0, write 1, and increment buffer address command pulses sneak into this circuitry in order to accomplish their functions. Increment buffer address fires only the "address hold" single-shot. Write 0 and write 1 command pulses directly reset or set the second flip-flop and then trigger the write sequence generator.

Writing is not quite so bad. When execute write is active, the 250kHz write clock is gated into the write data encoder. Negative edges of write clock strobe the current buffer output data into a 7474 flip-flop and also increment the buffer pointer to read out the next bit. If a ONE was latched, the single-shot is fired to send a data pulse to the disk drive and the flip-flop is immediately reset to ZERO. If a ZERO was latched, nothing happens. Positive edges of the clock always fire the single-shot for interleaved clock pulses. The disk drive internally converts the write data pulses to magnetic flux transitions on the disk surface.

Figure 4 shows the optional data and sector separators. The components for these may be installed if the drive being used does not have internal data and/or sector separators. The two circuits are identical except for time delay values. A 74121 precision single-shot is used in each. The trailing edge of every clock pulse or sector pulse fires the single-shot which defines a "window" for detecting the data or index pulse. Using the trailing edge and a large timing resistor minimizes recovery time problems with the 74121. Type 7432 NAND-NOT gates route clock and sector pulses to one output and route data and index pulses to the other output under control of the single-shot. When adjusting the single-shots, a good scope is needed. For both circuits, the time should be adjusted so that the single-shot turns off midway between the data or index pulses and the following clock or sector pulses. This time is 3 microseconds minus the data pulse width for the data separator and 3.9 milliseconds minus the sector pulse width for the sector separator.

That about covers the hardware side of the disk interface. Next time a 512 byte ROM based software package for the 8080 will be discussed and formal announcement of available floppy disk related components will be made.

APPENDIX 1. FLOPPY DISK DRIVE CHARACTERISTICS

CHARACTERISTIC	INNOVEX 220	PERTEC FD400	GSI 105	SHUGART SA800	SHUGART SA801	REMEX RFD7400	CDC 9400	CALCOMP 140
STEP METHOD	STEP/DIR	STP I/O	STEP/DIR	STEP/DIR	STEP/DIR	STEP/DIR	STP I/O	STEP/DIR
J1	A	C	A	A	A	A	C	B
J2	A	B	A	A	A	A	B	A
J3	A	B	A	A	A	A	B	A
J4	A	B	A	A	A	A	B	A
UNIT SELECT	YES	NO[1]	YES	YES	YES	NO[1]	NO[1]	YES
READY SENSOR	YES	YES[2]	YES	YES	YES	YES	NO[3]	YES
READY GATED BY SELECT	YES	-	NO[4]	YES	YES	-	-	NO[4]
SECTOR SEPARATOR	YES	NO[5]	YES	NO	YES	OPT[6]	NO	OPT
DATA SEPARATOR	YES	NO	YES	OPT[7]	OPT[7]	OPT	YES	YES[8]
ABOVE 43 NEEDED	J6=B	J6=A	J6=A	NO	NO	J6=A	J6=A	J6=A
ERASE ENABLE NEEDED	NO	YES	YES	NO	NO	YES	NO	NO
WRITE PROTECT	OPT	NO	OPT	OPT	OPT	OPT[9]	NO	NO
FAULT SENSOR	YES	NO	YES	NO	NO	YES	YES	NO
MINIMUM STEP TIME MS	10	10	8	10	10	6	10	6
ADDITIONAL SETTLE TIME	10	20	12	8	8	24	10	10
HEAD LOAD TIME	30	40	40	35	35	50	60	16

NOTES: 1. Multiplexor (2 74157) will have to be constructed in wire-wrap area
2. Switch on door, delay will have to be constructed or operator will have to exercise care
3. Either ignore READY from interface or detect presence of sector pulses with retrig. one-shot
4. Multiplexor (1/2 7450) will have to be constructed in wire-wrap area
5. Gain of sensor may have to be reduced to resolve closely spaced holes
6. Option consists of aperture wheel, allows use of one hole disk for hard sector operation
7. Data separator is standard on model 1 drives
8. Separated data is in NRZ form, move J7 to position B
9. Manually operated switch when disk is inserted

APPENDIX 2. FLOPPY DISK INTERFACE COMMANDS AND STATUS

COMMAND CODES (octal)

0 Store 0, increment buffer pointer
1 Store 1, increment buffer pointer
2 Step head out towards lower track numbers
3 Step head in towards higher track numbers
4 Lift head away from disk surface
5 Lower head onto disk surface
6 Set high write current, tracks 0 - 43
7 Set low write current, tracks 44 - 76
10 Read next sector from disk into buffer starting at current buffer position
11 Write next sector from buffer to disk starting at current buffer position
12 Reset buffer pointer to zero
13 Increment buffer pointer
14 Reset fault condition
15 (spare)
16 Select drive 0 (primary drive)
17 Select drive 1 (secondary drive)

Status bits

0 State of bit at current buffer position
1 Head is at track 00 if bit is a ONE
2 Is a ONE during duration of index pulse
3 Is a ONE during duration of sector pulse
4 Is a ONE if disk in drive is write protected
5 Is a ONE if a fault condition detected
6 Is a ONE if disk drive is ready
7 Indicates which drive was last selected

CONTINUED FROM 3

Dear Sirs:

A number of us have read with great enthusiasm your article on a floppy disk interface. Yes we would be interested in PC boards for such, and are anxiously awaiting the next TCH!

I'm uncertain if you have been informed of our group. We call it the Mid Michigan Microcomputer Group - M^{3}Gr, and it consists of about 20 - 25 members. About 4 to 5 of us have 8008 machines, while there are about 10 8080's (Altairs). I have a MIL MOD-8 with MONITOR-8, TVT-1 with UART connected to the MIL CPU, and a total of 4K memory (2K monitor ROM, 2K RAM). It works very nicely.

We also are interested in your graphics display system. Do you have, or are you planning to manufacture boards of layouts for boards? Are there any updates to your graphics articles?

Thanks for the tremendous TCH! We are looking forward to the next issue. Thanks also for the inside on the "Kansas City Standard".

Richard F. Shultz

An improved (higher speed, better accuracy) graphics circuit is under development. A vector generator board, deflection amplifier board, and CRT power supply board are planned but a few months off. In the meantime, the old circuit slightly modified is now available from SUNTRONIX as a PC board that plugs into their surplus Sanders 720 display units. See the article in the September issue of 73 magazine.

CLASSIFIED ADS

There is no charge for classified ads in TCH but they must pertain to the general area of computers or electronics and must be submitted by a non-commercial subscriber. Feel free to use ads to buy, sell, trade, seek information, announce meetings, or for any other worthwhile purpose. Please submit ads on separate sheets of paper and include name and address and/or phone number. Please keep length down to 10 lines or less.

FOR SALE: 8080 TRACE program - Simplifies and speeds program debug. TRACE enables the user to scan all or selected portions of a program's execution for problem analysis. The PC, SW, and A-L contents are displayed; SW and A-L registers are sense switch selectable. Program listing and description for $7.50. ALTAIR ACR compatible tape included for $10. For additional information send SASE to: R. Rydel, 14021 Cricket Lane, Silver Spring, MD 20904.

FOR SALE: (1) Altair 8800 kit - new, unbuilt sell for $400. (2) MOD-8 - debugged, running with Monitor-8 in ROM complete set of boards with 1702A programming station, all IC's socketed, $300. (3) CREED TTY - $100. (4) Digital Group cassette interface - $20. (5) TVT-1 plus KBD-1 (SWTP keyboard) with UART board modified to work like 32 character 16 line KSR 33! Works nicely with MIL MOD-8 microcomputer above. Send SASE if interested to: R. Shultz, 611 N. Dexter, Lansing Mich., 48910.

PAPER TAPE splices - prepunched for any kind of 8 channel tape. $3.50 per hundred, post paid. H. S. Corbin, 11704 Ibsen Drive, Rockville, MD 20852

FOR SALE: Core stacks - 9 4Kx40 20mil and 2 8Kx36 80mil, $1.00/Kbyte. 5262 2K RAM's, $1.00 each. 1101's, 75¢ (16 only). PC edge connectors, .156" centers, non-standard solder terminals, gold palted: 22x1 - 10¢, 43x2 and 58x2 - $1.50. 5mHz Heath tube type scope, AC coupled, $75. Add shipping on all items. SASE for info. S. Wiebking, 919D Magellan Circle, Dallas, TX 75218. Ph. 214/328-4035.

FOR SALE: 1 inch computer tape at bargain prices! Scotch 861 and 871 on 3600' reels in original boxes. List price $38, my price $6/reel plus postage. Jim Stitt, 311 N. Marshall Road, Middletown, Ohio 45042.

FOR SALE: Univac model 490 computer. CPU with 32K core and power supply. C. E. Fry Jr., Box 507D Geyser Rd., Pittsburgh, PA 15205.

DO YOU KNOW of any minicomputer or microprocessor systems capable of synthesizing or recognising human speech? Any information on hardware or software handling of speech I/O will be appreciated. A summary of all information received will be sent to contributors requesting it. B.W. Klatt, R.R.1, Oliver, B.C., Canada.

FOR SALE: MITS RS-232 serial I/O board for the Altair 8800 (88-SIOA), assembled and tested, $80. MITS 256 word static memory, exapndable to 1K words (88-MCS), assembled and tested, $60. Expander motherboard (88-EC), $8. David Richards, Lawrence Berkley Labortory, University of California, Berkely, CA 94720. LBL ext-5776 or 529-0759 evenings.

FOR SALE: Factory assembled Altair 8800 and Processor Technology 3P+S I/O board - $650. M. A. Keane, 6242 Malvern, Troy, Mich. 48084.

THE BETA terminal owner's group of the Computer Hobbyist Group of North Texas is interested in establishing communications with owners of terminals that use the Univac 0769 series print mechanism. We are looking for Beta keyboards (microswitch # 53SW1-2). One of our members has some spare parts for BETA terminals for sale, Contact L. G. Walker, RT. 1 Box 272, Aledo, TX 76008, Ph. 817/244-1013.

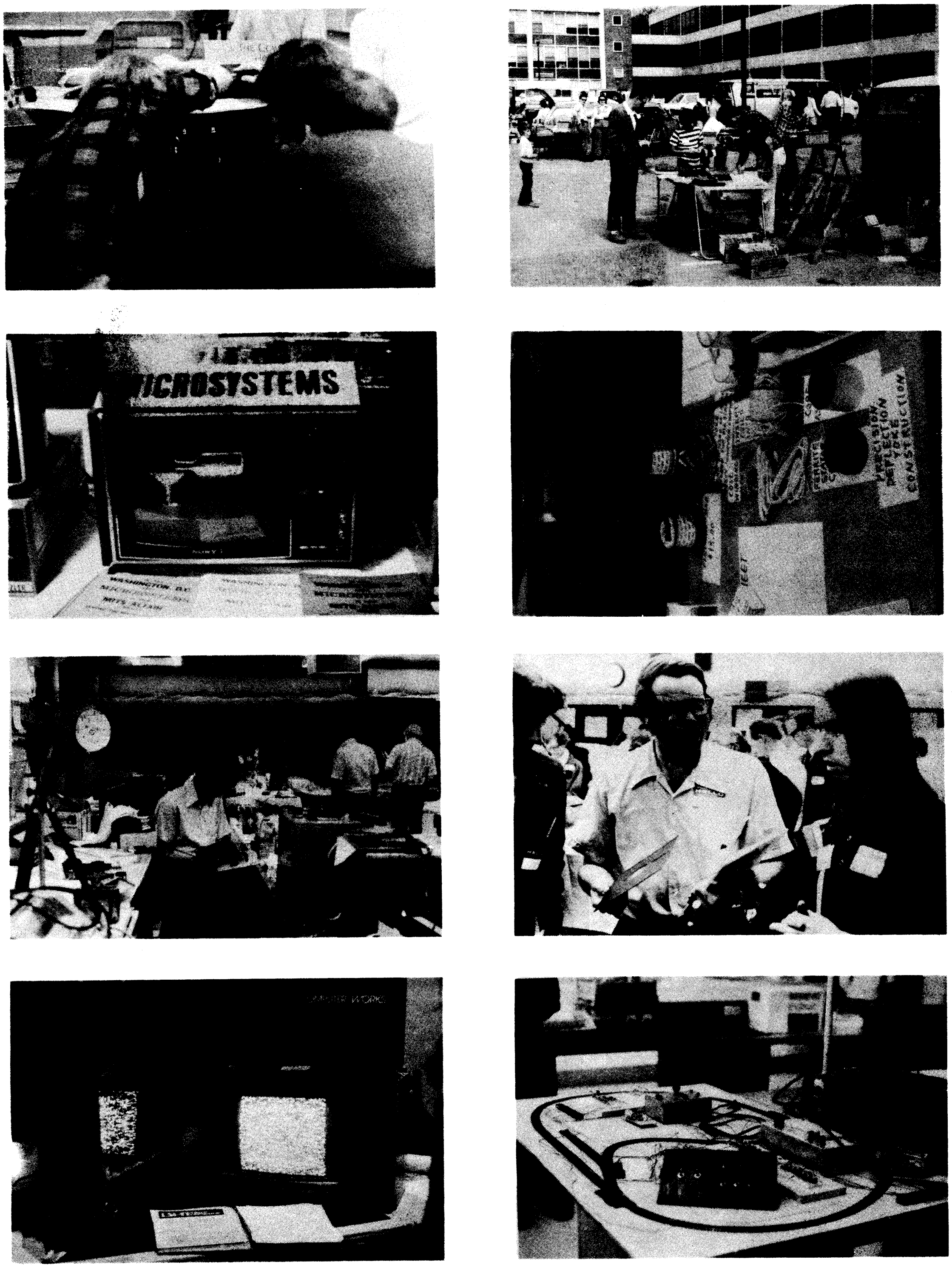
MAG MODULE - Info needed: Ampex 3227339-10F core memory "module" containing nineteen 3227xxx-10 series boards with unknown potted components. Need full schematics, operating voltages, timing; others welcome. Terry Ritter, 2524 Glen Springs, Austin, TX 78741, Ph. 512/441-0036.

FOR SALE: Teletype DRPE Super high speed paper tape punch - 240 characters/sec standard 8 level 1" tape. New complete $300 plus shipping. New dot-matrix line printer 180 char/sec. Simple parallel interface - $2000. (prices shown are U.S. or Canadian) John Youngquist, Box 122, Fort Erie, Ontario, Canada.

WILL TRADE Bicycling magazines, pre-1950 radio magazines, or comics (1950-1976) for microcomputer magazines or books, or a micro computer, or peripherals. Ian MacMillan, P.O. Box 128, Mount Royal, Quebec, Canada H3P 3B9.

FOR SALE/TRADE - HP5601A medical monitor, HP7848A 8 channel chart recorder, ATL profile monitor 17LP7A tube, with precision yoke, rack mounting, separate power supply, 2HP 1120A 500mHz active probes & 1 HP1122A power supply, all accessable. Motorola SLN state of the art freq. std. $3x10^{-11}$, 26VDC aerospace component, F.J.W. infra-red viewer with accessory IR source 6929 tube, very compact, uses one hearing aid battery! TT550/G paper tape reader - optical, 150-2400 baud, all IC circuitry, with literature. More weird & wounderful toys, write. I need - Selectric I/O, floppy disk, many 2102's, teletype #33, printer - mechanics only, YIG components. Ray Kajma Jr., 308 Flordia St., Farrell, PA 16121, Ph. 412/347-1736.

FOR SALE - Computer grade power supply. This is an insurance salvage deal, 20 units are available. Independent outputs - 5VDC @ 12 amps, 15VDC @ 2.8 amps, 15VDC @ 2.8 amps. All outputs are filtered, regulated, and variable. Modern L-shaped open-frame construction. $100 plus 10% for shipping and handling. M. D. Rivers, 28 Leyfred Terrace, Springfield, Mass., 01108.



Some scenes from the world's first ComputerFest held in Trenton, N.J.