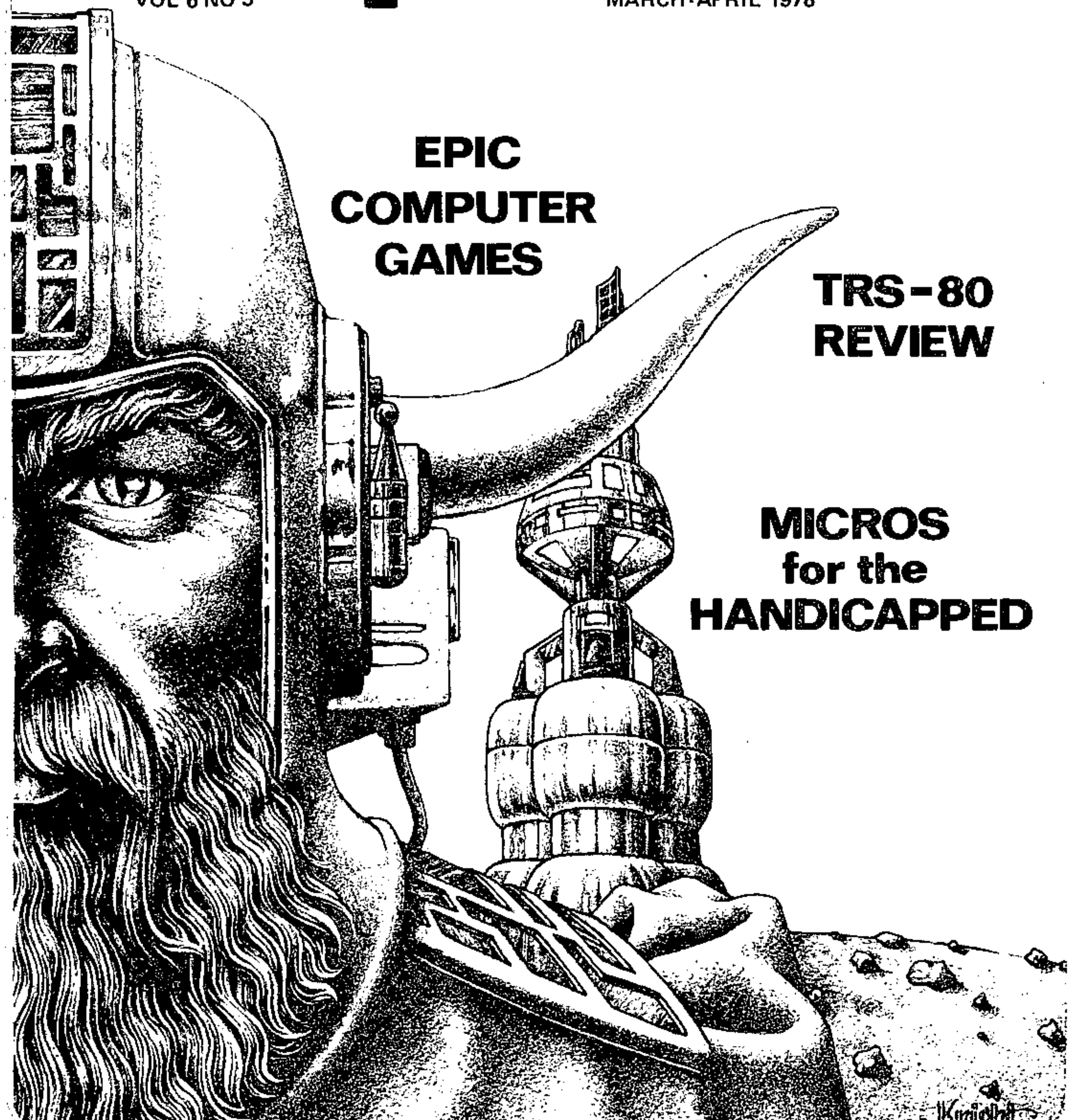
people's computers

$1.50

VOL 6 NO 5

MARCH-APRIL 1978



EPIC COMPUTER GAMES

TRS-80 REVIEW

MICROS for the HANDICAPPED

MICROCOMPUTER COMMUNICATION for the HANDICAPPED

BY TIM SCULLY

This article is more technical than many published in People's Computers, *but we believe that the general discussion will be interesting, informative, and thought provoking to all, even those who choose to skip the program listings and discussion.*

Tim Scully has been designing biofeedback equipment and doing biofeedback research for many years. Tim is a Research Fellow of the Humanistic Psychology Institute; he is now working towards his doctorate in psychology. His dissertation project involves researching and developing biofeedback systems and techniques for use in drug rehabilitation.

Tim is also teaching a computer class to fellow inmates at a Federal penitentiary. Although prison resources are scarce and he is not allowed to solicit donations, he is hopeful of somehow eventually acquiring a computer system for the prison.

The potential of microcomputers as tools for the handicapped is enormous and exciting: we encourage dissemination of such information. For this reason we are making copies of this article available. To receive a reprint, send a stamped, self-addressed envelope (24¢ for business size, 35¢ for 8½ by 11 inch) to People's Computers.

How would you communicate if you couldn't talk, didn't have the use of your hands, and could only somewhat control the movements of one knee? This is the problem which Robin, a young lady in her 20's has lived with all her life. She has cerebral palsy.

I met Robin in 1976, and this is the story of how a microcomputer communication system came to be built for Robin. The general concepts applied in the development of Robin's communication system may prove helpful in the development of microcomputer systems for other handicapped people.

When I first met Robin, her communication was accomplished by use of a word wheel. She could understand speech and she could read, but she needed help in 'talking'. Her word wheel was made from an electric clock motor and a bicycle spoke, with the bicycle spoke attached where the second hand of a clock would normally be mounted. A sheet of cardboard was mounted behind the spoke, with the letters of the alphabet on it, arranged in a circular pattern. The spoke pointed to the letters, one at a time, as it rotated. Robin could move her knee to one side and hit a kneeswitch mounted on her wheelchair, thus stopping the motor so that the spoke would freeze, pointing at the letter she had chosen.

The spoke rotated at one revolution per minute, so spelling proceeded at about one letter per minute! The person Robin was conversing with often had to write the letters down, to keep from forgetting them, as a message slowly built up. To speed up the communication process, a few words were written next to each letter of the alphabet, so that when the spoke stopped it would point at a group of words as well as a letter. The person with whom she was conversing would have to guess which of these Robin intended. It took considerable patience to hold a conversation with Robin, and not very many people took the time.

When I first saw Robin's communication system, I thought of replacing her word wheel with a microcomputer and video display, using a vocabulary of words stored in the computer's memory in place of the sheet of cardboard. A little over a year later, that system now exists and is being installed on Robin's wheelchair.

HOW IT WORKS

The present system is an expansion of the word wheel concept which uses a TV display with 16 lines of text. The top line is reserved for the display of a 'menu' of items (words, letters of the alphabet, punctuation symbols or control codes) from which Robin can choose. The second line is kept blank and the bottom 14 lines provide space for the display of a message of about 200 words.

As items are displayed on the menu, Robin can choose one by hitting the kneeswitch mounted on her wheelchair. In some modes of operation several items will appear on the menu at once, in which case the item at the left is the current item, the one which can be selected by hitting the kneeswitch.

On start-up, the system blanks the TV screen and then offers the SPELLING? mode by putting that word on the menu. This item remains on the menu for a time 'T1' (an adjustable time delay). If the kneeswitch is hit during that time, the SPELLING? mode is entered, otherwise the next menu item is displayed: PUNCTUATION?. If that item isn't chosen either, after another delay equal to T1, then the system will begin displaying the names of groups of words: A-BONE, BOOK-CROWN, CRY-FINGER, FINISH-HIDE, HIGH-LOT, LOUD-OUGHT, OUR-ROSE, ROSE ANN-STAY, SQUARE-TWENTY and TWO-YOURSELF, one group at a time. Each group of words contains about 120 words in alphabetical order. The name of each group is made up from the first and last words in the group.

If Robin doesn't pick any group of words, the computer then offers an ESCAPE? from the groups of words. If this isn't chosen, the names of the groups are offered again. If the ESCAPE? is chosen, the system returns to near the beginning of the program and offers SPELLING? again. This ESCAPE? to the beginning is offered from every mode of system operation.

If Robin does pick a group of words, HIGH-LOT for example, then the names of subgroups in that group begin being displayed, one at a time: HIGH-HONOR, HOPE-HUNT, HURRY-IMPORTANT, IN-INTERESTING, INTO-I'VE, JENNIFER-JUMP, JUST-KISS, KITCHEN-LAKE, LAND-LEAST, LEAVE-LIE, LIFE-LITTLE, LIVE-LOT and then ESCAPE?. If Robin picks a subgroup, such as LEAVE-LIE, then the words in that subgroup are displayed across the top line of the TV, with two spaces between each word:

LEAVE LED LEFT. . . LIBRARY LIE

If Robin hits the switch at this moment, LEAVE will be transferred down to the first available space in the message area of the TV screen and the menu will begin all over again by offering SPELLING?. If the first word, LEAVE, isn't chosen, then after the usual time delay T1, the list of words on the menu will shift one to the left, so that LED is on the extreme left and it becomes the current item. This process continues until a word is chosen or until the end of the subgroup, LIE. If LIE isn't chosen, ESCAPE? is offered, and if it isn't chosen, the complete list of 11 words in the subgroup is displayed across the menu and the cycle begins again.

By this system of groups of words, subgroups, and finally words, it is possible for Robin to look through a list of 1200 words in a short time, find the one she wants and add it to a message she is assembling on the TV screen. The computer automatically adds a space after each word chosen, so it isn't necessary for Robin to worry about spacing between words—she can just choose one word after another. All letters and words are upper case, so she doesn't have to shift.

When a sentence is complete, and when she wants punctuation symbols, Robin can select the PUNCTUATION? mode. The first item offered on entering this mode is CONTROL? and if that isn't chosen, then after the usual time delay, the punctuation symbols will be spread across the menu in much the same way that the words in a subgroup were displayed:

. ' ? ; : ! 0 1 2 . . . 9 # $ % & ( ) * + –

These items leave the screen at the left, one at a time, if they are not chosen. If one is chosen, the computer backspaces once (to undo the automatic spacing) and adds the chosen symbol to the message on the screen. Then the system starts over by offering SPELLING? again.

The CONTROL? mode offers Robin a few useful commands, one at a time, if it is chosen: BACKSPACE?, ERASE LAST WORD?, SPACE?, ERASE SCREEN?, and NEXT LINE?. These control codes operate immediately if selected. Then the system starts over by offering SPELLING? again.

The SPELLING? mode exists to allow Robin to spell words not found in the 1200 word vocabulary stored in the computer's memory. To speed up the process of spelling, letters of the alphabet are not offered in alphabetical order. Instead they are offered in the order of their probability of use in English. Except at the beginning of a word, the likelihood of a letter appearing in a word depends on the last letter chosen.† If we are in the middle of a word, and the last letter chosen was 'A', then the most likely next letter is 'E', the second most likely is 'B', etc.

Robin's system has 27 different alphabets stored in it. The first alphabet has the letters organized so that those most likely to appear at the beginning of a word will be displayed first. This is the alphabet which appears when the SPELLING? mode is first entered. The letters are spread out along the menu line as usual, with the first offering on the left. If no letter has been chosen by the time all of them have moved off the screen to the left, the usual ESCAPE? offering is made and the alphabet redisplays.

If a letter is chosen, it is added to the message area of the screen, and ESCAPE? is offered on the menu. If Robin decides to stay in the spelling mode, the computer then displays one of the 26 remaining alphabets—which one is determined by the letter she just chose. When she picks a letter from this new

† Mr A Ross Eckler suggested the bigram spelling scheme used in Robin's system. He supplied me with letter use frequency tables which he credited to F Pratt, *Secret and Urgent: The Story of Codes and Ciphers*, Blue Ribbon Books, 1942 pp 258-259.

alphabet, it is added to the message, immediately after the first letter (the system automatically backspaces to undo its automatic spacing). This process continues until she has completed spelling a word. Then she picks ESCAPE?, which returns her to the beginning of the program, which offers the SPELLING? mode, and a space is left after the word she has just completed.

This spelling scheme allows comparatively rapid spelling of words because Robin only has to wait for a few letters to display before the one she wants is likely, to become the current item. The automatic spacing also speeds up communication.

Now that we've looked at what Robin's system does, let's examine the hardware and software which do the work.

SYSTEM DESIGN

Robin's system was designed around the special limitations of her situation and my own situation. I met Robin through a United States Probation Officer, who was supervising me while I was temporarily free on appeal bond. I was waiting for the Court of Appeals to decide if it would uphold my conviction for conspiracy to manufacture LSD (back in 1968 and 1969). As it turned out, the Court did uphold my conviction, and I'm now serving a 10 year Federal prison term at McNeil Island Penitentiary in Washington.

My personal problems limited the system design to the use of a commercially available computer kit because of the difficulty of sending materials into prison. Robin's family had only a limited budget, and Robin's capabilities formed the remaining design limits.

In 1976, the budget we had (about $1,300) was just about enough to buy a computer kit with keyboard, cassette tape system, video monitor and 8K of memory, so this is the size system we planned on. The average word in English is about 5.5 characters long and we initially planned on a vocabulary of about 1,000 words, which uses up 5,500 bytes of memory. This left about 2,500 bytes for the program to control the system together with storage for spelling and punctuation symbols.

That's not enough memory for the use of a high level language such as BASIC, so the program had to be written in assembly language. Since my previous assembly language experience was with the 8080A, this was the CPU chosen for Robin's system.

We wanted the system to be expandable. In the future, Robin may want to add more memory, a printer, a speech synthesizer or other additional peripherals. For maximum flexibility in expansion, the S-100 bus structure was chosen because of the wide range of commercially available plug-in circuit cards. The computer also had to be small and light enough to mount under the seat of Robin's wheelchair. In order to modify the menu and message areas of the video display independently, the computer needed a memory-mapped video display. These constraints pointed us toward the Polymorphic Systems' Poly 88 System 4 kit.

The Poly 88 uses a 5 slot S-100 chassis, which makes it small and fairly light in weight. The Poly video card is memory mapped and displays 16 lines of 64 characters each—just right for Robin. The features of the Poly CPU card were also useful: it has 512 bytes of RAM together with a monitor program in ROM. A cassette tape interface card works together with tape loading software in the monitor ROM to handle program storage and loading.

The vocabulary for Robin's system is stored in RAM because we expect her vocabulary needs to change once she can communicate more freely. The problem with storing vocabulary in RAM is that RAM is volatile—the memory and thus the vocabulary are erased every time the computer is unplugged. So a battery back-up card was added to the system. This card keeps the program and vocabulary stored in RAM even though the computer may be unplugged for hours at a time while Robin's wheelchair is moved from place to place. Robin's computer uses the Seals Electronics BBUC card with NiCad batteries.

We had, at one point, considered battery powering the entire system, but ended up rejecting the idea. A large and heavy battery would have been required for reasonable life, and this would bring the total weight of the wheelchair and system up so high that Robin's mother wouldn't be able to lift it in and out of their family van for trips to school and other errands. As it is now designed, Robin's system has to be plugged into a wall outlet to operate, but the battery back-up card keeps memory alive while the system is unplugged so that it is instantly ready to start upon being plugged in.

HARDWARE MODIFICATIONS

A few additions and modifications were made to adapt the commercially available hardware to Robin's application. The Poly 88 chassis has only two controls: an on/off switch and a reset pushbutton. This is because it is designed to use a keyboard for functions which a control panel might perform. The reset pushbutton starts the ROM cassette tape loading program. I added a second pushbutton which activates a vectored interrupt and jumps to the beginning of Robin's program. This makes it possible to start up Robin's system without the keyboard. A schematic for this simple addition is shown in Figure 1.

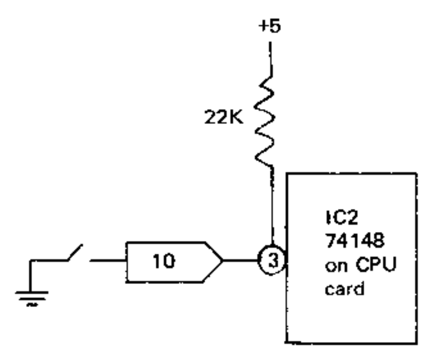


Figure 1

As a computer powers down, it can scramble data stored in memory by sending out false write commands. To eliminate this problem, the memory in Robin's system was partitioned so that an 8K block of RAM, containing the main program and stored vocabulary, could be write protected. This left only the 512 bytes of RAM on the CPU card unprotected (and the memory mapped video display, of course). The small CPU RAM area is used for all scratchpad functions and is one of the features of the Poly CPU card which encouraged its selection.

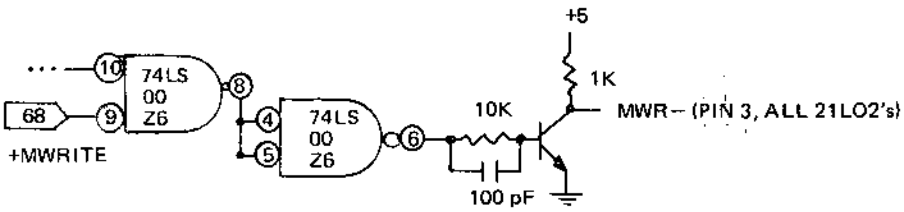


Figure 2

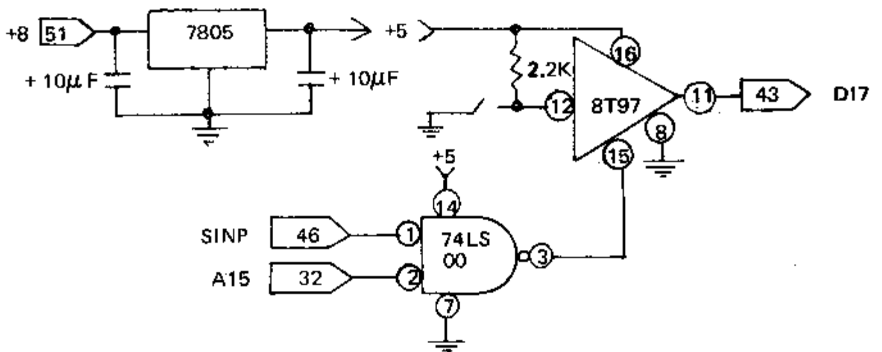


Figure 3

The RAM card used in Robin's system is an Industrial Microsystems IuS #000231 8K card which uses 21L02-4 chips. This card was modified slightly so that a toggle switch could be added to the computer's front panel which protects/unprotects the main 8K RAM. When loading new programs from cassette tape, RAM is unprotected. Otherwise it is protected. A schematic of this circuit is in Figure 2.

The final hardware modification for Robin's system was the addition of an input port for her kneeswitch. Figure 3 shows the schematic for this circuit, which was built on a small scrap of Vectorboard and mounted on the Poly 88 chassis.

SOFTWARE DESIGN

The program for Robin's system is listed, with comments, on the following pages. It was kept as brief and simple as possible to leave as much space in memory as possible for the storage of vocabulary. The vocabulary is stored as ASCII, with one character per byte of memory. ASCII doesn't use the eighth bit of an eight bit word, so I used the eighth bit as a 'beginning of word' flag. The first character of any character, word or phrase stored in memory has the eighth bit true, and all following characters (if any) have the eighth bit zero. This scheme allows the words in Robin's vocabulary to be packed tightly in memory. The only extra bytes of memory used are flags inserted at the end of each subgroup (FDH), group (FEH) and at the end of the vocabulary (FFH).

The main program uses one subroutine from the Poly 4.0 monitor ROM. That routine, WH1, outputs a character to the video display. It uses a location in the CPU board RAM, POS, to store the next position it will print into and it recognizes several control codes:

0DH = carriage return and line feed
0CH = erase screen and send cursor home (upper left corner of screen)
0BH = send cursor home without erasing screen
18H = erase current line

The starting address for the memory area mapped by the video display is F800H. Thus, if the control code 18H is in the A register when WH1 is called, it will stuff F800H into POS. WH1 saves all registers on entry and restores them on exit.

The monitor ROM on Robin's CPU board is a slightly modified version of the 4.0 monitor: at address 0008H a JMP 2000H has been inserted so that vectored interrupt VI6 jumps to the start of the main program. This allows a single pushbutton to start Robin's system.

Robin's software was hand assembled because I didn't have an assembler program to run on her system. The program listings were typed by hand and may contain a few errors.

TEXT AND EDITOR PROGRAMS

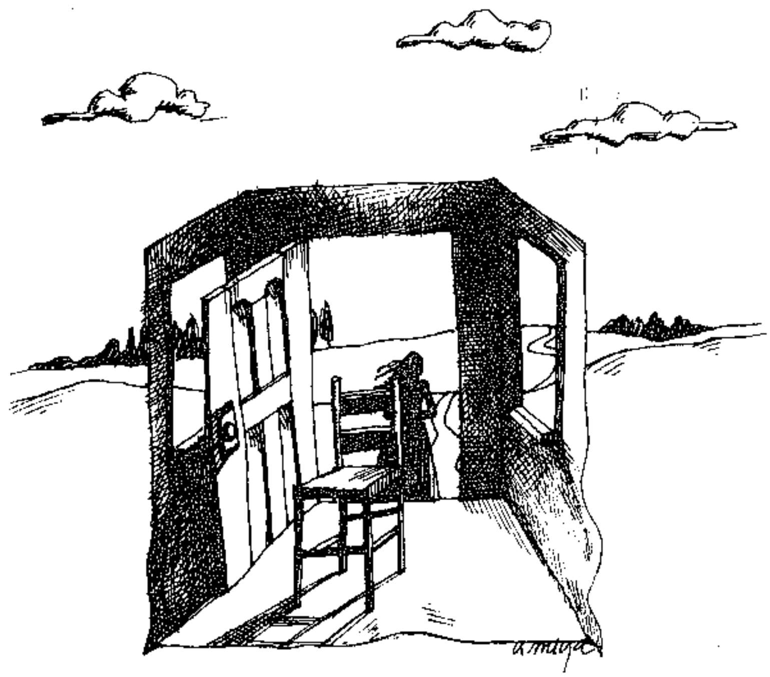
The TEXT and EDITOR programs written for Robin's system are both very short. TEXT was used to enter the messages, alphabets and vocabulary into her system's memory from the keyboard. EDITOR is used to modify her vocabulary and to add to it after the original entry. Here is what they do in detail.

TEXT is entered with a starting address in HL. The TV screen is erased and the system waits for text to be entered from the keyboard. Any unshifted letter is printed on the TV screen as a lower case letter but is stored in memory (beginning at the starting address in HL) as upper case ASCII with the eighth bit zero. The keyboard for Robin's system is a Teletype-like keyboard and does not have lower case letters, so the 'unshifted' letters are actually upper case, but TEXT translates them for display purposes.

Any letter of the alphabet typed while the CTRL key is held down (except Z) is printed on the TV screen as a capital letter and is entered into memory as upper case ASCII with the eighth bit turned on. This allows the first letter of any word or phrase to be identified. The Poly monitor program uses CTRL Z as a command to enter its front panel mode, so this is the one exception to the rule stated above. Shift O jumps to the EDITOR program, at the current address. Rubout erases the last character entered.

TEXT is also capable of inserting the control codes which identify the end of alphabets, subgroups and groups.
CTRL shift L = insert FBH
CTRL shift N = insert FDH
CTRL shift O = insert FEH

EDITOR is a somewhat longer and more complex program which allows the user to examine the text stored in the system's memory. It also allows modifications of



that text by insertions and deletions. If a deletion is made, all of the rest of the text (at addresses greater than the deleted address) is moved down one memory location to close the gap. If an insertion is made, all of the rest of the text is moved up one location to make room for the addition.

EDITOR is entered with a starting address in HL. Upon entry it will display a 'line' of text, beginning at that address. At the left end of the line, the current starting address will appear, in hex, followed by a space. Then the contents of memory are printed, up to and including the first 'control code' found. Any letters stored in memory with the eighth bit high will print on the TV as capitals, while those with the eighth bit low will print as lower case. The control codes will print as special symbols:

FBH = {   FDH = }   FEH = ~   FFH = ■

The EDITOR recognizes several commands, as listed below:

carriage return = display next line
line feed = display previous line
space = redisplay the current line, shifted one character to the left
NOTE: insertions made by EDITOR will go just in front of the first character on the display. The space is used to move along the current line so that insertions (or deletions) can be made in the middle of a line.
Shift O = jump to TEXT with HL equal to the starting address of current line.
CTRL shift L = insert capital L
CTRL shift M = insert capital M
CTRL shift O = insert FEH
CTRL shift N = insert FDH
rubout = delete first character of current line
any unshifted letter = insert that letter with eighth bit low
CTRL any letter except L, M, or Z = insert that letter with eighth bit high.

The EDITOR and TEXT programs use several more subroutines from the Poly 4.0 monitor ROM. Either program is entered with a starting memory address in HL. The monitor program allows register pairs to be pre-loaded from the keyboard while it is operating in the 'front panel' mode. For a detailed explanation of this procedure, see the Poly system manual Volume 2 pages 58-65. The other subroutines used are:

WHO = fetches a character from the keyboard and returns it in A. No other registers are affected.
DEOUT = print the two byte number in DE as a four character hex number.
MOVE = move -BC bytes from the area starting at (HL) to the area starting at (DE) – only works for moving to lower addresses.

Robin's main program turned out to be shorter than expected. Including the alphabets and punctuations symbols, it is 1250 bytes long. Even with 1,200 words of vocabulary in memory, there is still room for TEXT and EDITOR to remain in memory so that Robin's family can revise her vocabulary as needed.

A FEW WORDS ABOUT WORDS

It may be helpful to briefly mention how the initial vocabulary for Robin's system was chosen. The first 1,100 words were supplied by Robin's tutor, from lists of the first words taught in English. The remaining words were chosen by Robin and her family. These include the names of people, places, articles of clothing, foods and other objects which Robin comes in contact with.

As practical experience with the system is accumulated, revisions may be made in the initial vocabulary and possibly in the main program. For example, it may turn out that Robin will feel more comfortable spelling words than looking them up in the stored vocabulary. If this is the case, we may try adding a set of look-up tables for prefixes, roots and suffixes to speed up communication.

FUTURE DIRECTIONS

The basic system built for Robin can be expanded and modified to fit a wide range of possible situations. For example, the kneeswitch could easily be replaced by an electromyograph (EMG), an instrument which measures the electrical signals associated with muscle tension. An EMG can easily detect levels of muscle tension which are too weak to control a switch mechanically. This is a practical alternative to the kneeswitch for people who are only capable of very limited movement, such as an eyelid twitch. There are many hospitalized patients who experience extreme frustration because they are conscious but cut off from communication. Microcomputer communication systems of some kind may eventually become standard hospital equipment, and could help to make such patients' lives much more rich and meaningful.

There is a wide range of possible options for expanding Robin's system. It would be easy to add a printer, for example. She could assemble a message on the TV screen as usual, and then select a 'print' command which would cause the message area of the TV screen to be copied on paper. This would allow her to write an essay or a letter.

An S-100 compatible card is available from D. C. Hayes Associates (the 80-103A Data Communications Adaptor) which would allow her to select and dial a telephone number and send messages over the telephone to anyone having a computer terminal. A number of computer networks are now being used as communication networks, and it is reasonable to expect a network for handicapped people to develop in the near future.

Computers can be used to generate and control sounds. Several companies now offer S-100 compatible speech synthesis cards. It would be possible for Robin to learn to speak out loud, using one of these cards. Although the necessity of learning a new 'language' of phonemes would initially make this a slow communication process, the potential exists for this to be a very rapid communication mode.

Several companies now offer S-100 compatible circuit boards for music synthesis. It would be possible to write a program which would allow Robin to compose music and instruct the computer to perform it for her. Computer graphics are also possible. With a higher resolution video display, it would be possible for her to draw pictures with fine detail, and with a suitable printer, make 'hard copies' of these on paper.

There are several CMOS microprocessor CPU chips available now. Although CMOS memory and peripheral chips are still somewhat more expensive than TTL and NMOS chips used in Robin's system, it is already practical to build a microcomputer system similar to Robin's which would consume much less power. Such a system would be more expensive, but would be capable of battery operation, increasing portability.

S-100 compatible circuit cards are readily available which allow a computer to control electrically operated devices in its surroundings. It would be easy to expand a system like Robin's to allow her to turn on and off lights, appliances, etc. A system which can communicate can be a flexible control system too.

If you decide to try to build a microcomputer communications system for a handicapped person, I'd like to hear from you. I may be able to help with advice, and Robin might benefit from your ideas. My mailing address is:

Tim Scully
35267-136 CH
P O Box 1000
Steilacoom, WA 98388

NOTE: Thanks are due to the staff of McNeil Island Federal Penitentiary, whose cooperation made this project possible. The staff of Aquarius Electronics in Albion, California were also very helpful in tracking down parts for Robin's system. Robin's family provided the essential financial support, and Robin, her family and tutors all helped by contributing ideas and suggestions.

Tim Scully

McNeil Island December 1977 □

Programs

```
This is Robin's main program
2000  3E0C    START    MVI A,0CH
2002  CD240C           CALL WH1        erase screen
2005  CD4922           CALL PATCH
2008  22820C  ENTERL   SHLD CM         save current message address
200B  00               NOP
200C  215622  REENTRY  LXI H,SPELM-1
200F  CDFF20           CALL MESSAGE    offer spelling
2012  CA8F21           JZ SPELL
2015  215F22           LXI H,PUNM-1
2018  CDFF20           CALL MESSAGE    offer punctuation
201B  CAD321           JZ PUNCT
201E  217F25           LXI H,BEGIN     start of vocabulary
2021  11FE00           LXI D, 00FEH    set flags
2024  CD8120           CALL MENU       offer groups
2027  11FD00           LXI D,00FDH     set up flags
202A  2A840C           LHLD CI         fetch chosen item's address
202D  CD8120           CALL MENU       offer subgroups
2030  11FC01           LXI D, 01FCH    set up flags
2033  2A840C           LHLD CI         fetch chosen item's address
2036  CD8120           CALL MENU       offer words
2039  2A840C           LHLD CI         fetch chosen item's address
203C  7E      XMIT     MOV A,M         fetch character
203D  CD240C           CALL WH1        print it
2040  23               INX H           next
2041  7E               MOV A,M         fetch it
2042  E680             ANI 80H         check for start of next item
2044  CA3C20           JZ XMIT         if not next item, keep printing
2047  3E20             MVI A,' '
2049  CD240C           CALL WH1        space after completed item
204C  2A0E0C  ELOP     LHLD POS        fetch current message address
204F  C30820           JMP ENTERL      go back to offer spelling again
The following subroutine is used by MENU to save the 'flags' which start out in DE (the
flag in D is 0 unless MENU is asked to display words or individual characters, the flag in E
is the indicator telling MENU what to display, eg. FEH = groups, FDH = subgroups, etc.),
it also saves the data in HL as its starting address and saves the current value of POS in
the 'current message' storage.
2052  EB      ENTER    XCHG
2053  22860C           SHLD FLAGS
2056  EB               XCHG
2057  22800C  SENTER   SHLD CS
205A  2A0E0C           LHLD POS
205D  22820C           SHLD CM
2060  2A800C           LHLD CS
2063  C9               RET
The following subroutine erases the top two lines of the video display without disturbing
the message displayed on the bottom 14 lines. It ends with the cursor at 'home'.
2064  3E0B    NEW      MVI A, 0BH
2066  CD240C           CALL WH1        send cursor home
2069  3E18             MVI A,18H
206B  CD240C           CALL WH1        erase current line
206E  3E0D             MVI A,0DH
2070  CD240C           CALL WH1        go to next line
```

```
2073    3E18                MVI A,18H
2075    CD240C              CALL WH1        erase it too
2078    3E0B                MVI A,0BH
207A    CD240C              CALL WH1        send cursor back home
207D    C9                  RET
```

MAJOR SUBROUTINES: SMENU AND MENU

SMENU and MENU, which follow, are the major subroutines for displaying items on the menu (the top line of the video display). MENU is entered with flags in DE and a starting address in HL. The flags tell MENU to display groups, subgroups, words or individual characters. The starting address tells MENU where to find the first item to display. An exit from MENU is accomplished when an item is selected by use of the kneeswitch. Upon exit from MENU, the starting address of the chosen item will be in CI.

```
207E    11FB01    SMENU     LXI D,01FBH     set flags for spelling
2081    CD5220    MENU      CALL ENTER      save address & flags
2084    CD6420    ITEM      CALL NEW        erase menu
2087    22840C              SHLD CI         save current item address
208A    7E        DISPY     MOV A,M         fetch character from memory
208B    CD240C              CALL WH1        and display it
208E    23                  INX H           next
208F    7E                  MOV A,M
2090    E680                ANI 80H         check for msb=1
2092    CA8A20              JZ DISPY        if not, keep printing
2095    AF                  XRA A           are we finished with group or
2096    BA                  CMP D           are we printing with words or letters?
2097    C26421              JNZ WORD        if so, go on with words or end
209A    14                  INR D           otherwise, set flag
209B    3E2D                MVI A,'—'
209D    CD240C              CALL WH1        print '—'
20A0    2B                  DCX H
20A1    23        SEARCH    INX H           and look for end of group or
20A2    7E                  MOV A,M         subgroup
20A3    BB                  CMP E           by checking for a flag like E
20A4    DAA120              JC SEARCH       keep looking until found
20A7    2B        BACKUP    DCX H           then backup
20A8    7E                  MOV A,M         and print it
20A9    E680                ANI 80H
20AB    CAA720              JZ BACKUP
20AE    C38A20              JMP DISPY
```

The next four locations store the timing constants for two time delays: T1 and T2. T1 is the time each item on the menu is displayed and T2 is the minimum time the kneeswitch has to be closed before it is considered intentional (so that accidental twitches will be ignored).

```
20B1    5050                DW 5050H        T1 time constant
20B3    5050                DW 5050H        T2 time constant
```

SUBROUTINE: SWITCH

The subroutine SWITCH looks for a switch closure for time T1 and then returns with zero in D if the switch was never closed. If the switch closes, but not for at least T2, the routine just starts over, extending T1. If the switch closes for at least T2, then after the switch is released, it returns with one in D.

```
20B5    1600      SWITCH    MVI D,0         set up 'never closed flag'
20B7    E5                  PUSH H
20B8    2AB120              LHLD T1         fetch time constant
20BB    E5                  PUSH H
20BC    C1                  POP B           put it in BC
20BD    E1                  POP H
20BE    DB80      IN        IN 80H          look at switch
20C0    E680                ANI 80H         it's only one bit
20C2    CADA20              JZ CLOSED
20C5    22900C              SHLD 0C90H      waste time
20C8    2A900C              LHLD 0C90H      to make timing loop longer
20CB    22900C              SHLD 0C90H
20CE    2A900C              LHLD 0C90H
20D1    0D                  DCR C
20D2    C2BE20              JNZ IN          check switch every time
20D5    05                  DCR B
20D6    C2BE20              JNZ IN          keep timing
20D9    C9                  RET             time up, no contact
20DA    E5        CLOSED    PUSH H
20DB    2AB320              LHLD T2         fetch time constant
20DE    E5                  PUSH H
20DF    C1                  POP B           put it in BC
20E0    E1                  POP H
20E1    22900C    WAIT      SHLD 0C90H      waste time
20E4    2A900C              LHLD 0C90H
20E7    0D                  DCR C
20E8    C2E120              JNZ WAIT        keep timing
20EB    05                  DCR B
20EC    C2E120              JNZ WAIT        time up?
20EF    DB80                IN 80H          check switch
20F1    E680                ANI 80H         it's only one bit, the msb
20F3    C2B520              JNZ SWITCH      start over if not still closed
20F6    14                  INR D           set flag for contact
20F7    DB80      UP        IN 80H          check switch again
20F9    E680                ANI 80H
20FB    C0                  RNZ             wait until it is released
20FC    C3F720              JMP UP          meanwhile looping
```

SUBROUTINE: MESSAGE

The subroutine MESSAGE is used to display a number of short messages on the menu. Message is entered with an address in HL equal to one less than the starting address of the message to be displayed. It will display the message found, up to and including a terminating '?'. Upon exit from message, the zero flag in the PSW will be one if the offered item was chosen and zero if it was not chosen.

```
20FF    000000    MESSAGE   NOP NOP NOP     I deleted something here
2102    CD6420              CALL NEW        erase menu
2105    23                  INX H
2106    7E                  MOV A,M
2107    CD240C              CALL WH1        print
210A    FE3F                CPI '?'         check for end of message
210C    C20521              JNZ MESSAGE +6
210F    2A820C              LHLD CM
2112    220E0C              SHLD POS        restore POS
2115    CDB520              CALL SWITCH
2118    3E01                MVI A, 1
211A    BA                  CMP D
211B    C9                  RET
```

SUBROUTINE: COMP

The subroutine COMP is used by MENU to check the switch.

```
211C    CDB520    COMP      CALL SWITCH
211F    3E01                MVI A,1
```

```
2121   BA                  CMP D
2122   C22C21              JNZ NEXT       if no contact, offer next choice
2125   2A820C              LHLD CM
2128   220E0C              SHLD POS       restore main text POS
212B   C9
```

MORE ROUTINES USED BY MENU

The following chain of routines are used by MENU to find and display the next item, check for the last item in a list, offer ESCAPE? and recycle to the beginning of the list if nothing is chosen. The details of these operations vary depending on what items are being offered: groups, subgroups, words or characters.

```
212C   EB        NEXT      XCHG           save current address
212D   2A860C              LHLD FLAGS     while restoring flags
2130   EB                  XCHG
2131   7B                  MOV A,E
2132   FEFD                CPI FDH        are we displaying groups or subs?
2134   D24121              JNC CHECK      if so, check for end
2137   2A840C              LHLD CI
213A   23        FIN       INX H          skip current word or letter
213B   7E                  MOV A,M
213C   E680                ANI 80H        and keep skipping until the
213E   CA3A21              JZ FIN         start of the next, then check
2141   1C        CHECK     INR E          the last item will be followed
2142   7E                  MOV A,M        by a flag = to E + 1
2143   BB                  CMP E
2144   D25721              JNC LAST
2147   1D                  DCR E          restore flag in E
2148   FEFB                CPI FBH        if no control code found,
214A   DA8420              JC ITEM        keep displaying
214D   7B                  MOV A, D
214E   FEFD                CPI FD
2150   DA5721              JC LAST
2153   23                  INX H          skip control code
2154   C38420              JMP ITEM
2157   CD8221    LAST      CALL ESCAPE    if last item was displayed, offer
215A   2A860C              LHLD FLAGS     escape and then loop back
215D   EB                  XCHG
215E   2A800C              LHLD CS
2161   C38420              JMP ITEM       and start displaying over again
```

SUBROUTINE: WORD

WORD, the next subroutine, is used by MENU. If groups or subgroups are being offered, it is entered only after the complete offering has been printed and it jumps to COMP to check the switch. But if individual words or characters are being offered, WORD keeps printing words or characters across the menu space, with two spaces between each, until the end of the subgroup or until the end of the line.

```
2164   7B        WORD      MOV A, E       check flag
2165   FEFD                CPI FDH
2167   D21C21              JNC COMP       and split if groups or subs
216A   3A0E0C              LDA POS        check position in menu
216D   FE3C                CPI 3CH        if we are near the end of
216F   D21C21              JNC COMP       the line, stop printing &
2172   7E                  MOV A, M       split or if we are at the end
2173   BB                  CMP E          of the subgroup, split
2174   D21C21              JNC COMP
2177   3E20                MVI A, ' '     otherwise,
2179   CD240C              CALL WH1       print two spaces
217C   CD240C              CALL WH1
217F   C38A20              JMP DISPY      and add more to menu
```

SUBROUTINE: ESCAPE

The subroutine ESCAPE offers a return to the SPELLING mode and is used often.

```
2182   C5        ESCAPE    PUSH B
2183   214F22              LXI H, ESC-1   set up for message
2186   CDFF20              CALL MESSAGE
2189   C1                  POP B
218A   C0                  RNZ            return if no escape
218B   E1                  POP H          clean up stack
218C   C30C20              JMP REENTRY    and reenter SPELLING?
```

SUBROUTINES USED BY SPELLING MODE

The SPELLING mode uses this chain of subroutines. The first alphabet offered is different from the other 26, and the routine doesn't backspace before printing the first letter, so there is one routine for the first letter and another for all the others. ESCAPE? is offered after each letter is printed and before a new alphabet is offered. A look-up table is used to pick the right alphabet to offer after the first letter has been printed.

```
218F   211523    SPELL     LXI H, ASTART  address of initial alphabet
2192   CDB721              CALL FIRST     print first letter
2195   CD5720              CALL SENTER    to restore POS
2198   CD8221    TALE      CALL ESCAPE    offer escape
219B   21C722              LXI H, STAB    start of look-up table
219E   78                  MOV A, B       fetch last letter printed
219F   BE        LOOK      CMP M          and look for it in table
21A0   CAA921              JZ FOUND
21A3   23                  INX H          each table entry
21A4   23                  INX H          is three bytes
21A5   23                  INX H
21A6   C39F21              JMP LOOK       keep looking, you'll find it
21A9   23        FOUND     INX H          when you find it,
21AA   5E                  MOV E, M       get address from table
21AB   23                  INX H
21AC   56                  MOV D, M
21AD   EB                  XCHG           and put it in HL
21AE   CDC821              CALL SECOND    offer new alphabet
21B1   CDBA21              CALL OOP       print the chosen letter
21B4   C39821              JMP TALE       and loop back to do it again
21B7   CD7E20    FIRST     CALL SMENU     offer alphabet
21BA   2A840C    OOP       LHLD CI        fetch chosen item's address
21BD   7E                  MOV A, M
21BE   CD240C              CALL WH1       and print it
21C1   47                  MOV B, A       save it for look-up later
21C2   3E20                MVI A, ' '
21C4   CD240C              CALL WH1       and print a space
21C7   C9                  RET
21C8   CD7E20    SECOND    CALL SMENU     offer alphabet
21CB   2A0E0C    SECONDS   LHLD POS       get ready to backspace
21CE   2B                  DCX H          and
21CF   220E0C              SHLD POS       do it
21D2   C9                  RET
```

SUBROUTINE: PUNCT

The subroutine PUNCT handles offering the control codes (by calling another subroutine) and it offers the punctuation symbols. It uses one of the spelling subroutines to handle punctuation.

```
21D3   216B22   PUNCT    LXI H, CONTROLM-1
21D6   CDFF20            CALL MESSAGE   offer CONTROL?
21D9   CAEB21            JZ CONTROL
21DC   21AC22            LXI H, PSTART  starting address of punctuation
21DF   CDCB21            CALL SECOND    offer them
21E2   CDBA21            CALL OOP       print the chosen one
21E5   C34C20            JMP ELOP       go back to offer SPELLING?
21E8   000000            NOP NOP NOP    I took out something here
```

SUBROUTINE: CONTROL

CONTROL offers and executes the control commands.

```
21EB   217322   CONTROL  LXI H, BACKSPACE?-1
21EE   CDFF20            CALL MESSAGE   offer backspace
21F1   C2FA21            JNZ TWO
21F4   CDCB21            CALL SECONDS   backspace
21F7   C30820            JMP ENTERL     back to offer SPELLING?
21FA   217D22   TWO      LXI H, ERASE LAST WORD?-1
21FD   CDFF20            CALL MESSAGE
2200   C21222            JNZ THREE
2203   2A0E0C            LHLD POS
2206   2D                DCR L          back up
2207   2D       MORE     DCR L          back up
2208   3EA0              MVI A, ' '
220A   BE                CMP M          have we reached a space?
220B   C24022            JNZ RUB
220E   23                INX H          leave the space
220F   C30820            JMP ENTERL     and go offer SPELLING?
2212   218D22   THREE    LXI H, SPACE-1
2215   CDFF20            CALL MESSAGE
2218   C22322            JNZ FOUR
221B   3E20              MVI A, ' '
221D   CD240C   END      CALL WH1
2220   C34C20            JMP ELOP       back to offer SPELLING?
2223   219422   FOUR     LXI H, NEXT LINE?-1
2226   CDFF20            CALL MESSAGE
2229   C23122            JNZ FIVE
222C   3E0D              MVI A, 0DH
222E   C31D22            JMP END
2231   219E22   FIVE     LXI H, ERASE SCREEN?-1
2234   CDFF20            CALL MESSAGE
2237   CA0020            JZ START       start all over
223A   CD8221            CALL ESCAPE
223D   C3EB21            JMP CONTROL
224D   36A0     RUB      MVI M, A0H     put blank on screen
2242   C30722            JMP MORE
2245   00                NOP
2246   00                NOP
2247   00                NOP
2248   00                NOP
2249   2181F8   PATCH    LXI H, F881H   initialize text address
224C   220E0C            SHLD POS
224F   C9                RET
```

In the listing below I haven't typed the hex equivalents for the ASCII (this listing was hand-assembled).

```
2250                     ESCAPE?
2257                     SPELLING?
2260                     PUNCTUATION?
226C                     CONTROL?
2274                     BACKSPACE?
227E                     ERASE LAST WORD?
228E                     SPACE?
2295                     NEXT LINE?
229F                     ERASE SCREEN?
22AC                     .'?;:!0123456789#$"%&()*+-
22C6   FB                DB FBH         end flag
```

ALPHABET LOOK-UP TABLE

Here is the look-up table for the various alphabets, in non-standard form.

```
22C7   C13023            A 2330H
22CA   C24B23            B 234BH
22CD   C36323            C 2363H
22D0   C47723            D 2377H
22D3   C59123            E 2391H
22D6   C6AC23            F 23ACH
22D9   C7C223            G 23C2H
22DC   C8DA23            H 23DAH
22DF   C9F123            I 23F1H
22E2   CA0C24            J 240CH
22E5   CB1324            K 2413H
22E8   CC2B24            L 242BH
22EB   CD4624            M 2446H
22EE   CE6024            N 2460H
22F1   CF7B24            O 247BH
22F4   D09624            P 2496H
22F7   D1AD24            Q 24ADH
22FA   D2B124            R 24B1H
22FD   D3CC24            S 24CCH
2300   D4E624            T 24E6H
2303   D50025            U 2500H
2306   D61A25            V 251AH
2309   D72825            W 2528H
230C   D83F25            X 253FH
230F   D95725            Y 2557H
2312   DA7025            Z 2570H
```

THE ALPHABETS

And here are the alphabets, once again without the hex.

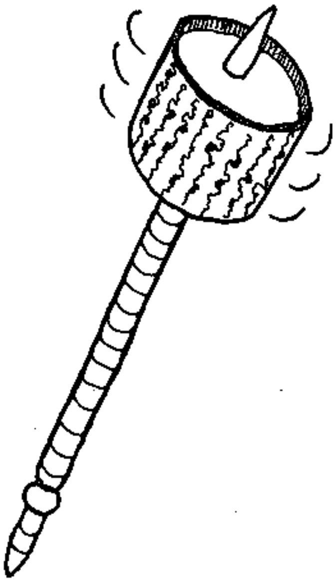
```
2315            ASTART   TAOSWIHCBFPMRELNDUGYJVQKZX
232F   FB                DB FBH         end of alphabet flag
2330                     NTSRLDCIGVMYPBKUFWOJXHZEQA
234A   FB                DB FBH
234B                     EAOUYRISLJTVMBDWCGHNPFK
2362   FB                DB FBH
2363                     OEHATKILURCYSONDZMW
2376   FB                DB FBH
2377                     EIUARSOLMDGYNVJQWHEFTPKBZ
2390   FB                DB FBH
2391                     RSNDALMCETVFPXIGYOWUHQKBJZ
23AB   FB                DB FBH
23AC                     ORIFEAULTSYWBMGCHNJPD
23C1   FB                DB FBH
23C2                     EHROAIGSLUTNYMFDBWZJKPC
23D9   FB                DB FBH
23DA                     EIAOTURYLNWDSMBHQFPCGK
23F0   FB                DB FBH
```

```
23F1                           NSTOCMLAREDVGPFBKXUZQIJLWY
240B    FB                     DB FBH
240C                           AEOUIJ
2412    FB                     DB FBH
2413                           EISANLYOGFWTURDPMKBJCHV
242A    FB                     DB FBH
242B                           EIALYODTSUFRMVWKPCBGNHJZXQ
2445    FB                     DB FBH
2446                           EAOIPMUYSBLFNTHCDRWGJKVCZ
245F    FB                     DB FBH
2460                           DTEGSCIAOYNLFVUKMJRQPHWXBZ
247A    FB                     DB FBH
247B                           NFRUMPLTOWSDCVIBEYAKHJGXZQ
2495    FB                     DB FBH
2496                           ROAELTSPIHMUYWFGKBNDCJ
24AC    FB                     DB FBH
24AD                           UIO  •
24B0    FB                     DB FBH
24B1                           EIOATSYDMNURCLVKGPWBFHXQJZ
24CB    FB                     DB FBH
24CC                           TEIOSHUCAPYKMWNLGQFBDRVJZ
24E5    FB                     DB FBH
24E6                           HEIOARSTUYLWCFMNBPDZGKVJQ
24FF    FB                     DB FBH
2500                           TSNRLCGPAEMDIFBOYZXUVKQJH
2519    FB                     DB FBH
251A                           EIAOYUSRVZKGM
2527    FB                     DB FBH
2528                           EAHIONRSLTDYKUPFBCMZWG
253E    FB                     DB FBH
253F                           EPTICAHUYOQLNWFSVGBKMRD
2556    FB                     DB FBH
2557                           EOSAITPMBLNWCRGDZHUFVXIK
256F    FB                     DB FBH
2570                           EAZOYIUKTVWHJB
257E    FB                     DB FBH      end of alphabets
257F    C1C1424C45C1424F       AAbleAbout  beginning of vocabulary storage
        5554
```

TEXT AND EDITOR

```
3F00    CD200C    TEXT     CALL WHO       keyboard input
3F03    FE7F               CPI 7FH        is it rubout?
3F05    CA263F             JZ RUB
3F08    FE5F               CPI 5FH        is it shift 0?
3F0A    CA383F             JZ EDITOR
3F0D    FE1C      CTL      CPI 1CH        is it a control character?
3F0F    DA213F             JC CONTROL
3F12    FE20               CPI 20H        is it a control code?
3F14    D2193F             JNC PRINT      if not, print it
3F17    C6DF               ADI DFH
3F19    77        PRINT    MOV M, A       store it in memory
3F1A    CDE53F             CALL LPRINT    put it on TV
3F1D    23                 INX H          next memory location
3F1E    C3003F             JMP TEXT       do it all over again
3F21    F6C0      CONTROL  ORI C0H        make eighth bit high
3F23    C3193F             JMP PRINT      for 'capital' letters
3F26    CDE53F    RUB      CALL LPRINT    rubout on TV
3F29    2B                 DCX H          back up in memory
3F2A    C3003F             JMP TEXT       go do it over
3F2D    2A800C    RETEXT   LHLD 0C80H     fetch starting address
3F30    3E0C               MVI A, 0CH     erase TV
3F32    CD240C             CALL WH1
3F35    C3003F             JMP TEXT
3F38    22800C    EDITOR   SHLD 0C80H     save start of current line
3F3B    3E0D               MVI A, 0DH
3F3D    CD240C             CALL WH1       start a new line
3F40    2A800C             LHLD 0C80H     fetch start of current line
3F43    EB                 XCHG
3F44    CDD103             CALL DEOUT     print address in hex
3F47    EB                 XCHG           restore address
3F48    3E20               MVI A, ' '
3F4A    CD240C             CALL WH1       print space
3F4D    7E        LOOP     MOV A, M       fetch character from memory
3F4E    CDE63F             CALL LPRINT    put it on TV
3F51    7E                 MOV A, M
3F52    23                 INX H
3F53    FEFB               CPI FBH        was it the end of a line?
3F55    DA4D3F             JC LOOP        if not, keep printing
3F58    CD200C    KEY      CALL WHO       wait until a key is pressed
3F5B    FE20               CPI ' '        is it a space?
3F5D    C2673F             JNZ M1         if not, keep checking
3F60    2A800C             LHLD 0C80H     fetch starting address
3F63    23                 INX H          space skips one character
3F64    C3383F             JMP EDITOR     and reprints line
3F67    FE7F      M1       CPI 7FH        is it rubout?
3F69    C2873F             JNZ M2
3F6C    2A800C             LHLD 0C80H     fetch starting address
3F6F    E5                 PUSH H         copy HL
3F70    D1                 POP D          into DE
3F71    3EFF               MVI A, FFH     end of vocabulary flag
3F73    010000             LXI B, 0       start counting at zero
3F76    2B                 DCX H
3F77    23        M3       INX H
3F78    0B                 DCX B          count one byte
3F79    BE                 CMP M          check for end flag
3F7A    C2773F             JNZ M3         keep counting if not the end
3F7D    2A800C             LHLD 0C80H     fetch starting address
3F80    23                 INX H          we are moving one space
3F81    CD0001             CALL MOVE
3F84    C3383F             JMP EDITOR     display edited line
3F87    FE0D      M2       CPI 0DH        is it carriage return?
3F89    CA383F             JZ EDITOR      then display next line
3F8C    FE0A               CPI 0AH        is it line feed?
3F8E    C2A03F             JNZ M4
3F91    2A800C             LHLD 0C80H     fetch starting address
3F94    2B                 DCX H          back up
3F95    2B        M5       DCX H          keep backing up
3F96    7E                 MOV A, M
3F97    FEFB               CPI FBH        look for control flag
3F99    DA953F             JC M5          and keep backing up until found
3F9C    23                 INX H          skip the flag
3F9D    C3383F             JMP EDITOR     and display previous line
3FA0    FE5F      M4       CPI 5FH        is it shift 0?
3FA2    CA2D3F             JZ RETEXT      if so, go to TEXT
3FA5    FE1C               CPI 1CH        is it a control character?
3FA7    DABE3F             JC M6          if so, it is upper case
3FAA    FE20               CPI 20H        could it be a control code?
```

Prayer Wheel Program



BY EDRID

When I finished building my computer, I wanted to do something far out with it to start off right. Having been a meditator for some time, I thought of a computer implementation of a Tibetan Prayer Wheel. I chose an ancient high mantra for the first thing my computer would do in its present incarnation.

We had the good fortune to meet Sonam Gyatso, a genuine Tibetan Lama. When told of my computer's 'recitations', he beamed brightly and said, characteristically, 'Oh my! Great Merit!'

As of January 27, 1978 the number of recitations by my computer was 22,199,184. We encourage the spread of this program, and would like to know of other implementations.

Edrid
c/o Dynabyte
4020 Fabian Way
Palo Alto, CA 94303

```
10 REM * * * * MANTRA * * * *
20 REM * * A PRAYER WHEEL PROGRAM * *
30 REM * * WRITTEN BY EDRID * * * *
40 REM * * IN NORTH STAR BASIC * * *
100 OPEN #0, "MANTRAF"
110 READ #0, N: CLOSE #0
120 CHR$(12)
130 PRINT: PRINT: PRINT: PRINT
140 PRINT "MUMBLE. . . MUMBLE. . ."
150 DIM M$(18)
160 FOR M=1 to 1007: READ M$
170 RESTORE: NEXT M
180 DATA "OM MANI PADME HUM!"
190 PRINT M$
200 N = N + 1008
210 PRINT "                ", N
220 OPEN #0, "MANTRAF"
230 WRITE #0, N: CLOSE #0
240 GOTO 160
```

100-110 gets the number of past recitations of the mantra from the disk.
120-140 clears the screen and prints a message to give a hint of what is going on.
150-170 prepares a space in memory for the mantra, then puts it in there over and over for 1007 times, one less than the number of petals in the Crown chakra.
180 is the mantra.
190 prints the 1008th.
200 adds 1008 to the number of recitations.
210 prints the total number of recitations of the mantra.
220-230 puts the new total onto the disk, with the thought that some of the power of the 1008 recitations is within the number.
240 goes back to the beginning to do the whole thing over again, endlessly.

```
3FAC  D2C03F          JNC INSERT    if not, insert it as is
3FAF  FE1E            CPI 1EH       be sure it is not M or L
3FB1  D2B93F          JNC M7        if it's not, then control code ok
3FB4  C6B0            ADI B0H       make into L or M
3FB6  C3C03F          JMP INSERT    and insert it
3FB9  C6DF    M7      ADI DFH       make into control code
3FBB  C3C03F          JMP INSERT
3FBE  F6C0    M6      ORI C0H       make into capital letter
3FC0  2A800C  INSERT  LHLD 0C80H    fetch starting address
3FC3  F5              PUSH PSW      save character on stack
3FC4  3EFF            MVI A, FFH    end of vocabulary flag
3FC6  010000          LXI B, 0      start counting at zero
3FC9  2B              DCX H
3FCA  23      M8      INX H
3FCB  0B              DCX B
3FCC  BE              CMP M
3FCD  C2CA3F          JNZ M8
3FD0  54      M9      MOV D, H      move forward
3FD1  5D              MOV E, L
3FD2  13              INX D
3FD3  7E              MOV A, M
3FD4  12              STAX D
3FD5  1B              DCX D
3FD6  2B              DCX H
3FD7  0C              INR C         count one space
3FD8  C2D03F          JNZ M9
3FDB  04              INR B
3FDC  C2D03F          JNZ M9
3FDF  F1              POP PSW       get back character
3FE0  12              STAX D        and insert it
3FE1  23              INX H
3FE2  23              INX H
3FE3  C3383F          JMP EDITOR
3FE6  FE60    LPRINT  CPI 60H       is it upper case?
3FE8  D2240C          JNC WH1       print as is
3FEB  FE41            CPI 41H       is it lower case?
3FED  DA240C          JC WH1        if not, print as is
3FF0  C620            ADI 20        make it lower case
3FF2  C3240C          JMP WH1       and print it
```