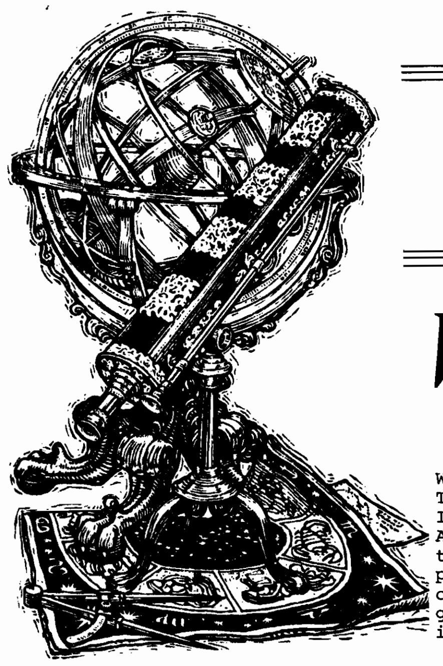
VOLUME I NUMBER 1



The COMPASS

Permission will be given users groups to reprint on request.

WHO ARE WE ??

Welcome!
This is the first issue of the International North Star Users Association newsletter. It's a totally independent newsletter published by a non-profit, completely independent, users' group. Independent means really independent. This newsletter is an open forum, a clearing house for the exchange of information between users of North Star Computers, or users of the North Star Disc Operating System or North Star Basic on any computer. The articles, opinions, and other data in these pages reflect the experience of the individual authors and not necessarily the opinions of INSUA or North Star Computers, Inc.

I expect this open policy may lead to controversy, but I hope it will also lead to the widest interchange of ideas, fixes, programs and techniques that will make your computer a more valuable tool.

INSUA has been formed to provide the North Star System user around the world with software support and practical knowledge that will assist in getting the most out of a very viable operating system. That is,I hope, our one and only bias. That bias is also our goal. We want to be able to provide any enhancement to the operating system that makes a good thing better.

You know that we will never be able to fully realize that goal since the nature of this dynamicaly expanding world of computer technology makes it a quest that requires continual striving. We'll try and with your help we hope to come close. We'll need your ideas, your criticisms, your program routines, and your debugging to share with other users. This is your forum.

The Association liasions with North Star Computers Inc to represent the needs and desires of users to the company. It also liasions with local computer groups, not as a competitor but rather to provide whatever support is wanted so that organizations can connect with others and individual users can connect with organizations in their oun geographical area.

continued page 3

INTERNATIONAL NORTH STAR USERS ASSOCIATION
PUBLISHERS OF THE COMPASS NEWSLETTER
POST OFFICE BOX 1318, ANTIOCH, CA. 94509

MOVABLE DOS BASIC

North Star has released a user relocatable BASIC and DOS. The new operating system, Release 5.2, can now run at any address. In fact, the standard factory DOUBLE DENSITY disc boots up at 100 hex. The factory disc is now priced at $25

✶ ✶ ✶ ✶

The operating system is still in 8080 code and thus compatible with all current users hardware. Only the RAM TEST utility decribed below is in Z80 code and configured particularly for the Horizon.

You should be able to make most older software compatible by changing the in/out references to DOS since the jump table and user I/O areas in DOS are retained at the same relative addresses, even though routines within the system have been altered.

The master disc includes a mover program in Basic and mirror image files of DOS, BASIC, and all the utilities. These files are merely copies of the all programs properly assembled to run at different addresses than the standard versions. The mover program compares the two versions so it can avoid changing the necessary fixed information while adjusting the rest of the code to run at your selected custom location. A very simple an clever implementation by North Star.

Everything can be moved, not only BASIC and DOS but also the various standard utilities such as Copy Disk, Copy File, Disc Test, etc. It is up to the user to decide where in memory he wants what to run.

This opens up the possibility of moving programs like Copy File out of the way so that their buffer areas don't overwrite Basic every time they are used.

For those who had reserved memory boards down below "normal" DOS for other software, the relocation capability can make a 8 kilobyte contribution to memory now available to North Star's Basic. You can just boot up a copy of the new factory Double Density disc and go.

The SINGLE DENSITY version of DOS still loads at 2000 HEX but is

continued page

SPECIAL OFFER! YOUR INSUA MEMBERSHIP CAN BE FREE.....

North Star Extended BASIC Disk Operating System

GET YOUR DISC
NORTH STAR VERSION 5.2
AT MEDIA COST + HANDLING

ONLY $10.00 FOR MEMBERS

REGULAR PRICE NOW $25.00

Save you the cost of one year's membership in your users association

INSUA has been authorised by North Star Computers Inc. to offer its members copies of the latest version 5.2 Relocatable Basic and Dos and all utilities.

FOR LIMITED TIME we will copy the factory disc for members for $10 to cover our costs. You MUST be a member to take advantage of this "discount price". The regular price for the factory disc is $25.00

Membership dues are $15 per year and include a subscription to the newsletter as well all other INSUA support. For a total of $25.00 ($15 + $10), the cost of version 5.2 Basic and Dos alone you'll recieve a copy of the factory disc and year's newsletter and all other benifits of membership in your independent users association.

NOTE: THESE DISCS ARE COPIES MADE BY INSUA. THEY ARE NOT FACTORY DISCS. North Star Computers Inc can take no responsibility for our copies, though INSUA will guarantee the media and replace defective media should it occur.

INTERNATIONAL
NORTH STAR USERS ASSOCIATION
POST OFFICE BOX 1318,
ANTIOCH, CA. 94509

MOVABLE DOS & BASIC

CONTINUED FROM PAGE ONE

unfortunately not relocatable as furnished on the factory disc, even though a Mover program is provided. A custom E900 prom would be needed to implement the relocated single density DOS because there are addresses referencing the standard DOS location in the standard single density prom. The price of custom North Star proms are now $150. A different prom would be needed for each relocated version of DOS.

Your users association suggested a short boot program to allow single density system owners to take advantage of any relocated version of single density DOS without the cost of new prom.

THEY SAID SINGLE DENSITY COULDN'T BE MOVED....BUT

Responding to this suggestion John Dvorak, 704 Solano Ave., Albany, CA 94706 is marketig a utility routine that attaches a boot program and modified software version of North Star's prom to DOS allowing the single density version to be moved.

When North Star's mover program moves the DOS it will also change the reference addresses in the software prom. His version of relocated DOS is 12 blocks long (1 block BOOT, 10 blocks DOS, and 1 block PROM).

Other Double or Quad users, who faithfully followed the old North Star configuaration of memory from 2000 hex to DFFF hex are not so lucky. They will have to first move the supplied DOS at 100 hex to a new location. Fortunately the mover program will function with the old 5.1 Basic.

All new North Star Computers will be shipped with memory at 0 to accomadate the new DOS according to the factory.

Relocation is not the only enhancement provided in Release 5.2. Here are some of the others:

READ/WRITE SPEED IMPROVEMENT
Double and Quad DOS has a speed improvement in the disc read and write routines which should be noticable during copy file and data file accesses.

NEW CK UTILITY
This uitility checks the factory master disc to verify that there have been no unauthorized changes in any of the code on the entire disc. Checksum for Horizon-S DOS=21516; Horizon-DQ DOS=58777; MDS-S DOS=54115; and MDS-DQ=38179.

INTERUPTS DISABLED DURING DISC ACTIVITY
DOS now disables interupts during disc transfers and re-enables them when disc activity finishes if the option has been selected.

DEVICE STATUS ROUTINES
DOS now includes device status check routines for input devices at 2041 and for output devices at 2044. Both return the Zero flag true if the periphials are ready. The user calls these routines by putting the device number in the accumulator and preserving all other registers.

RAM TEST
This new utility, for Horizon users only, does a more sophisticated memory test than the TM routines in the monitor. There are two versions to cover the computer address range, each with its own Horizon I/O so that DOS is not needed. These Ram tests not only read and write memory with an assortment of bytes but also run mini calculation routines and test the answers. A phase of the memory test checks on the board's ability to fetch code from different memory chips checking that the board deselects properly. The routines rely on the wait state generation provided in North Star's dual density board when E918 hex is called.

MEMORY FAILURE MESSAGE
Ram memory boards, which generate vectored interupt 5, can optionally print a memory failure message during any running program.

continued next page

CLIP TIPS

IDENTIFY BASIC VERSIONS

Various vintages of North Star Basic scattered in your disc file can be instantly identified using the LINE length setting and ERRSET statements in this program:

```
10 ERRSET 100,A,B
20 LINE 165
30   !  "THIS  IS  5.2  NORTH  STAR
BASIC"
35 END
40 LINE 132
50 ! "THIS IS 5.1 or 5.0 NORTH STAR
BASIC"
55 END
60 LINE 9
70 ! "THIS IS AN OLDER VERSION"
75 END
100 ERRSET 120,A,B
110 GOTO 40
120 ERRSET 140,A,B
130 GOTO 60
140 ! "BASIC HAS BEEN MODIFIED"
```

MOVABLE DOS & BASIC

CONTINUED FROM PAGE TWO **

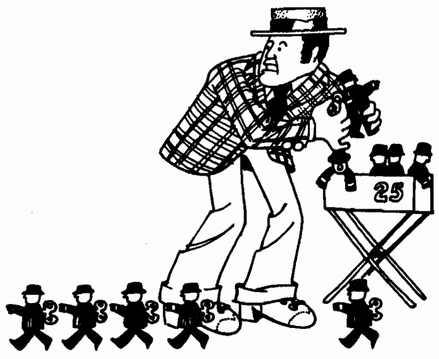
DLOOK ERROR CHECK
THE DLOOK routine now returns a zero in the accumulator if there is a syntax error in the file name.

BASIC's NEW LINE LENGTH
Basic in Release 5.2 can now support a 165 character line and can suppress carriage returns to the output device if an optional zero is added as an extra parameter to the LINE length statement.

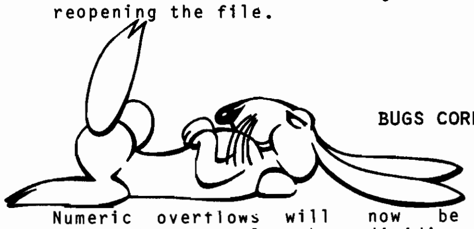
FILE SIZE FUNCTION
The size of a current open file, in-blocks,will be returned by a new FILESIZE (X) statement.

FILE POINTER POSITION
FILEPTR (X) will return the file pointer position in a current open file for possible use in random access read and write statements.

SETTING FILE POINTER
The Program statement WRITE #0 %P, NOENDMARK will move the filepointer to position (P) for the next sequential READ or WRITE statements. This opens the possibility of setting the pointer to zero without closing and reopening the file.



DOS ACCEPTS LOWER CASE
The DOS commands li or LI, JP or jp, GO or go, etc.are now accepted by the operating system. This change makes mini-floopy DOS compatable with North Star's hard disc operating system. It also could be useful to users of Electric Pencil I or other word processors who have occassion to switch in and out of DOS and lower case text inputing. Basic, however is still limited to upper case commands.



BUGS CORRECTED IN VERSION 5.2 OPERATING SYSTEM

Numeric overflows will now be detected properly when dividing large numbers by very small numbers; Use of user DEF functions in IF/THEN statements will nolonger preven use of the ELSE clause; DEF statements can now be followed by additional statements on the same line; An "obscure bug" in the CReate statement will no longer affect other statements.

SINGLE DENSITY: The PRESS RETURN TO CONTINUE bug in single density DOS as been corrected (See simple fix for 5.1 DOS in this issue). Other bugs corrections in single density include the posibility of setting the auto start option without using a fresh copy of DOS.

DOUBLE DENSITY: No more failures to boot up properly regardless of previous contents of memory at the DOS load address; No more systems hang up on listing a poorly seated disc; Directory listing buffer will no longerdamage programs overhanging 2A00 HEX; OFTEN now called during seeks on drives with fast access; Copy File will now copy small files into large files of 512 blocks or more.

BOTH VERSIONS OF DOS: Hard disc error messages no routed to Device #0; Read after right check operational in disc INitialization mode; CReate command fails if given illegal file name.

COMPACT: The CO command will not proceed if there are overlapping files on the disc.

DOUBLE DENSITY: No more failures to boot up properly regardless of previous contents of memory at the DOS load address; No more systems hang up on listing a poorly seated disc; Directory listing buffer will no longer damage programs overhanging 2A00 HEX; OFTEN now called during seeks on drives with fast access; Copy File will now copy small files into large files of 512 blocks or more.

BOTH VERSIONS OF DOS: Hard disc error messages no routed to Device #0; Read after right check operational in disc INitialization

INTEGRATED BUSINESS SOFTWARE
—NORTH STAR BASIC—

•Inventory System	: Complete inventory control system with transactions and reports.
•Name/Address	: Complete with reports and automatic letter generation.
•Accounts Receivable	: With customer list and general ledger tie-ins.
•Accounts Payable	: With vendor list and general ledger tie-ins.
•Payroll	: With state and federal reports and general ledger tie-ins.
•General Ledger	: With optional depreciation schedules, last year and budget planning.
•Leasing Package	: With tie-ins to general ledger and accounting packages.
•Sales Management	: With tie-ins to Name/Address.
•Order Entry and Invoicing	: With tie-ins to all other programs.
•Bill of Materials	: With reports and tie-in to Inventory.

All packages are $250 each with complete documentation to run on North Star, Helios, Thinker, Dynabyte, Cromemco or hard disk drives.

TECHNOLOGY SYSTEMS
P.O. Box 137 © Bethel, CT 06801
(203) 748-6856

WHO ARE WE

CONTINUED FROM PAGE ONE

Your user's association has been organized as a non-profit group and is currently filing with the State of California to be chartered unter terms of the law as non-profit corporation. It's been oranized by small group of peopl (See Tony Severa's letter) who met for the first time at last year's West Coast Computer Fair. We hope we won't remain a small group. You can change that.

You can make your association what you want it to be. Your ideas, criticisms, letters and help are more than welcome, they are needed to mold the users' association into a group refecting the needs of most of the users.

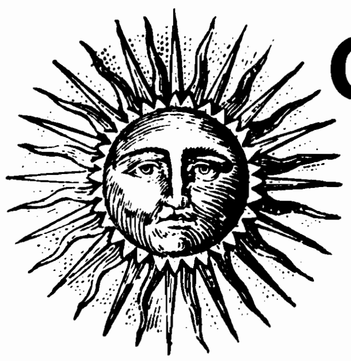
We hope this issue of COMPASS provides you with some useful information, routines and programs tailored to make your North Star system more useful. If you find these contributions helpful, please reciprocate by making your own tips, fixes and programs available to share with other members. Please take a look at the guide for contributers in this issue and send your material now for the next issue of COMPASS.

Please bear in mind that though "INSUA can arrange projects such as making North Star's newest version of relocatable Basic and DOS available to the user at cost, it took a member's boot routine to make factory version of the relocatable DOS usable to the single density drive owner. Your programs can do similar things for your fellow user.

Thanks....Clyde Steiner, Newletter Editor

COMPASS NEWSLETTER COPYRIGHT 1980 by INTERNATIONAL NORTH STAR USERS ASSOCIATION

**The International North Star Users' Association is a group independent of North Star Computers, Inc. North Star does not have any control over or responsibility for any of the material contained herein. While every attempt is has been made to insure the accuracy of the articles INSUA likewise can bear no responsibility nor liability for the contents of this newsletter.



COMMAND

CONSOLE COMMAND ROUTINE

EDITOR'S NOTE: This Basic program is available in condensed machine code as part of Pavel Breder's DOS-POWER. This program gives the North Star user the ability to create a sub routine type program which he can save on disc, load any place in memory, call as wanted to execute a series of consecutive commands as though he had entered them individually from the keyboard.

It gives North Star the a powerful feature, similar to CP/M's SUBMIT command, but with considerable enhancement. You can not only chain DOS commands for auto execution, you can combine them with Basic programs. For example, assuming you are in Basic, you call the subroutine previously created by BUILDER and loaded into memory at your selected address. The routine would run sequentially and automaticly:

```
BYE (exit basic)
DE TEST,2
LI2
JP 2D00 (return to basic)
5 DIM A$(20)
10 IF A$="END" THEN END
15 INPUT "Name",A$
20 !#2 A$
30 GOTO 10
RUN (actualy executes)
NSAVE TEST,2
CAT
BYE (return to DOS)
```

The program is user configurable to match your hardware. It will even accept "control"characters if desired.

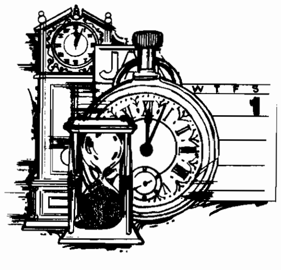
In addition to providing a simple management proceedure for batch processing, it opens the way for the user to write his basic programs using his word processor.

```
10 REM BUILDER1
20 REM
30 REM
40 REM BY PAVEL BREDER COPYRIGHT 1980
50 REM (415) 756-4263
60 REM
70 REM
80 REM
90 DIM F$(100) : F$=""
100 REM *******************************************************
110 Y=0          : REM GO ADDRESS                                *
120 X=1          : REM BLOCKS USED (1 BLOCK=256 BYTES)           *
130 D=8192       : REM DOS ORG                                   *
140 D1=16        : REM DOS CINP JUMP (RELATIVE)                  *
150 D2=22        : REM DOS CONTC JUMP (RELATIVE)                 *
160 B=11520      : REM BASIC ORG                                 *
170 B0=15        : REM BASIC AUTO START FLAG (RELATIVE)          *
180 B1=24        : REM BASIC CONTC FLAG (RELATIVE)               *
190 B2=23        : REM BASIC BACKSPACE  (RELATIVE)               *
200 B2$-CHR$(0)  : REM BACKSPACE CHR$ (8,95,127)                 *
210                REM IF BACKSPACE=CHR$(0) THEN PROGRAM WILL    *
220                REM DEFAULT TO BASIC'S OWN BACKSPACE AT       *
230                REM LOCATION 'BASIC+23'                       *
240 E$=CHR$(10)  : REM END INPUT CHARACTER (CHR$(10)=LINE FEED)*
250 E=1          : REM DATA IN MEMORY/DISK  0-READ 1-DON'T READ*
260 C$=CHR$(26)  : REM CLEAR SCREEN CHARACTER  (11? 12? 26?)   *
270 W=13         : REM IF NO RETURN DESIRED AFTER LAST COMMAND *
280                REM THEN CHANGE W=32 (SPACE) ELSE W=13 (CR) *
290 X5=27        : REM DURING BUILDER EXECUTION, PRESSING THIS *
300                REM KEY WILL COMPLETE THE CURRENT COMMAND    *
310                REM AND THEN EXIT FROM BUILDER (27=ESCAPE)  *
320 T1=6         : REM TAB INPUT                                 *
330 T2=57        : REM TAB FREE MEMORY                           *
340 Z1=0         : REM 1=IGNORE CTRL CHAR$  0=CTR CHAR$ O.K.    *
350 E8=0         : REM lower case TO UPPER CASE   =NO 1=YES    *
360 E9=0         : REM ENDMARK
370 R=0          : REM READ/WRITE TO DISK FILE 0=NO 1=YES        *
380 N$="BLD3!"   : REM NAME OF FILE                              *
390 REM *******************************************************
400 DEF FNA$(A$)
410 IF A$="" THEN A$=" "
420 IF A$(1,1)>="a" THEN IF A$(1,1)<="z" THEN A$(1,1)=CHR$(ASC(A$)-32)
430 RETURN A$
440 FNEND
450 INPUT "Instructions (Y/N or S-skip set up) : ",A$
```

CLIP TIPS

DATA FILE: CLOCK

This subroutine can be called from Basic or other programs to provide timing sequences for games,user response, or any other "-lock" need by utilizing your CPU's clock frequency. Timing is adjusted by changing the two hex numbers following the LXI B instruction. The XTHL's (E3's) are used as slow down loops. Add more E3's as needed by your hardware. This example provides one second clock units runing on the 8080 CPU Sol Computer.



The routine runs at zero. To load elseware change the two C2 02 00 instructions to C2 (your load address + 2).

```
ADDR. INST.  HEX CODE
0000  PUSH B  C5
0001  LXI B   3A00 time delay
0002  XTHL    E3
0003  XTHL    E3
0004  XTHL    E3
0005  XTHL    E3
0006  DCR C   0D
0007  JNZ     C2 0200 first XTHL
0008  DCR B   05
0009  JNZ     C2 0200 first XTHL
0010  POP B   C1
0011  RET     C9
```

EXAMPLE: This simple program prints a count down timer on screen.

```
10 Z=11:REM CLEAR SCREEN CODE
20 !"TIME CLOCK IN SECONDS"
30 INPUT"TOTAL SECONDS WANTED",T
50 ! CHR$(Z)
60 FOR I=0 TO T
70 !:!:!
80 X=CALL(0(
90 ! I,"OF",T," SECONDS SET"
100 ! T-I," REMAINING"
110 NEXT I
```

COMMAND
PROGRAM
CONTINUED
NEXT
PAGE

COMMAND PROGRAM BY PAVEL BREDER CONTINUED

```
460 A$=FNA$(A$)
470 IF A$="Y" THEN GOSUB 3010
480 IF A$="S" THEN 850
490 !
500 ERRSET
510 !C$,
520 !"THIS PROGRAM IS SET UP WITH THE FOLLOWING DATA:"
530 !"   (FOR PERNAMENT CHANGES USE YOUR EDITOR)"
540 !%#6I
550 IF X=0 THEN X=1
560 !"(0) BLOCKS USED  :",X
570 !"(1) GO ADDRESS   :",Y," DEC",FNH$(Y)
580 !"(2) BASIC ORIGIN :",B," DEC",FNH$(B)
590 !"(3) DOS   ORIGIN :",D," DEC",FNH$(D)
600 !"(4) CURRENT DATA :",E,"       (0-READ   1-IGNORE)"
610 !"(5) CONTROL CHR$ :",Z1,"       (0-ACCEPT 1-IGNORE)"
620 !"(6) WRITE TO FILE:",R,"       (0-NO     1-YES)"
630 !"(7) NAME OF FILE : ",N$
640 !"(8) END INP CHR$ : ",
650 IF ASC(E$)>32 THEN !E$ ELSE !"CTR/",CHR$(ASC(E$)+64)
660 !%#
670 INPUT "enter CODE to change or RET to start : ",A$
680 !
690 IF A$="" THEN 850
700 ERRSET 490,0,0
710 S1=VAL(A$)+1
720 ON S1 GOTO 730,740,750,760,770,780,790,800,810
730 INPUT "(0) BLOCKS USED  : ",X  GOTO 490
740 INPUT "(1) GO ADDRESS   : ",Y  GOTO 490
750 INPUT "(2) BASIC ORIGIN : ",B  GOTO 490
760 INPUT "(3) DOS ORIGIN   : ",D  GOTO 490
770 INPUT "(4) CURRENT DATA : ",E  GOTO 490
780 INPUT "(5) CONTROL CHR$ : ",Z1 GOTO 490
790 INPUT "(6) WRITE TO FILE: ",R  GOTO 490
800 INPUT "(7) NAME OF FILE : ",N$ GOTO 490
810 !"(8) END INP CHR$ : ",
820 E$=INCHAR$(0)
830 IF ASC(E$)>32 THEN !E$ ELSE !"CTR/",CHR$(ASC(E$)+64)
840 GOTO 490
850 D1=D1+D+1   REM DOS CINP
860 D2=D2+D     REM DOS CONTC
870 B0=B0+B     REM BASIC AUTO START FLAG
880 B1=B1+B     REM BASIC CONTC FLAG
890 B2=B2+B     REM BASIC BACKSPACE LOCATION
900 IF B2$=CHR$(0) THEN B2$=CHR$(EXAM(B2))
910 H=EXAM(B+17)+EXAM(B+18)*256
920 !
930 !C$,TAB(22),"STAND BY..."
940 IF R=0 THEN 1190
950 F1=FILE(N$)
960 IF F1>-1 THEN 1120
970 !
980 !"FILE ",N$," NOT FOUND...."
990 !
1000 INPUT "DO YOU WANT ME TO CREATE THE FILE (Y/N) ",A$
1010 A$=FNA$(A$)
1020 IF A$="Y" THEN 1050
1030 FILL B0,0
1040 A=CALL(B)
1050 ERRSET 1100,0,0
1060 CREATE N$,X,0
1070 ERRSET
1080 A3=1
1090 GOTO 950
1100 !"CAN'T DO IT..."
1110 STOP
1120 OPEN#0%F1,N$,R1
1130 IF A3 THEN A1=256*256 ELSE 1140  GOTO 1190
1140 READ#0%4,&A,&A1
1150 A1=A+A1*256-54
1160 IF X<=R1 THEN 1190
1170 !"CURRENT FILE IS",R1," BLOCK(S) - ALOCATED SPACE REDUCED"
1180 X=R1
1190 FOR Z=Y TO Y+57
1200   READ J
1210   FILL Z,J
1220 NEXT Z
1230 IF E THEN FILL Z,E9
1240 J=FNA(1,D1-Y)  J=FNA(16,D1-Y)  J=FNA(46,D1-Y)
1250 J=FNA(21,D2-Y)
1260 J=FNA(4,54)   J=FNA(7,58)   J=FNA(10,56)  J=FNA(13,19)
1270 J=FNA(24,40)  J=FNA(29,40)  J=FNA(32,56)  J=FNA(38,48)
1280 J=FNA(43,54)  J=FNA(50,56)  FILL Y+27,X5  FILL Y+41,W
1290 FILL Y+36,E9
1300 DEF FNA(K,L)
1310 FILL Y+K,Y+L  FILL Y+K+1,(Y+L)/256
1320 RETURN J
```

CONTINUED ON NEXT PAGE

CLIP TIPS

PROGRAM CONVERSION

Have you have been puting off adopting programs published in the commnercial computer press because they always seem to be written in a version of the ever-present Microsoft Basic? Does the line on line drugery of changing all those MID$, etc. get you down? Thereis a slick solution. The N-BUS utility program reviewed elsewhere in this issue does makes program syntax conversion to North Star a snap with its global change feature, not only for two character variables but for any string.

You might also take a look at the article ON TIME AND SPACE by Dr David Yates, University of Queensland, Australia in the August issue of Kilabaud Microcomputing (page 76). His 137 line North Star Basic program "CHANGE" will run through a disc file and make changes for you if you enter the Microsoft based properly first.

the electric abacus

congratulates

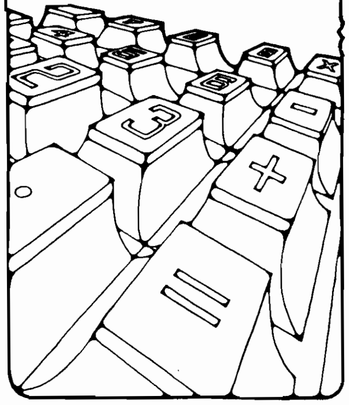
INSUA

on its premier issue.

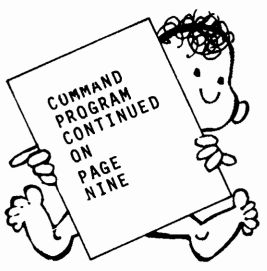
the electric abacus
19 mayfield road
regina, saskatchewan
s4v 0b7 canada

producers of:

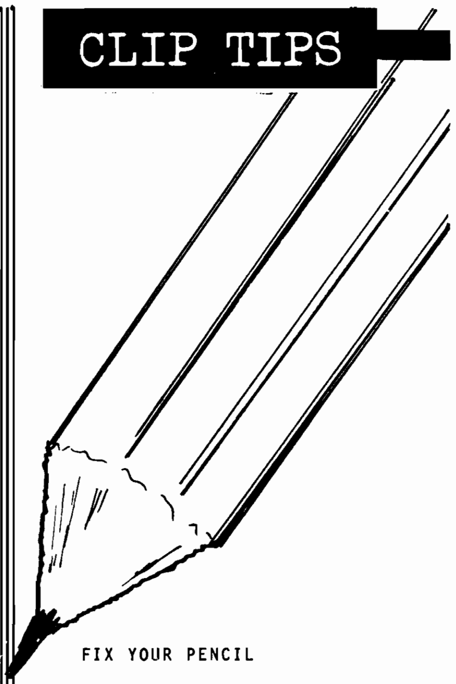
REILEY
Real Estate Index & Locator



```
1330 FNEND
1340 DATA 42,0,0,34,0,0,33,0,0,34,0,0,33,0,0,34,0,0,201
1350 DATA 229,205,0,0,202,0,0,254,0,202,0,0,42,0,0,126,254,255
1360 DATA 194,0,0,62,13,42,0,0,34,0,0,35,34,0,0,225,201,0,0,0,0
1370 X=(X-1)*256+198
1380 DIM B$(X+1)
1390 A$="D"
1400 U=1
1410 IF E THEN 2140
1420 FOR I=Z TO Z+X
1430  M=M+1
1440  IF R AND A3=0 THEN READ#0%I-Y,&A ELSE A=EXAM(I)
1450  IF A<>E9 THEN 1480
1460  B$(M,M)=CHR$(13)
1470  EXIT 2140
1480  B$(M,M)=CHR$(A)
1490  IF A=13 THEN U=U+1
1500 NEXT I
1510 M=0
1520 U=I-1
1530 GOTO 2140
1540 !C$,
1550 M=0
1560 U=1
1570 !"enter the command(s) (press ",
1580 IF ASC(E$)>32 THEN !E$, ELSE !"CTR/",CHR$(ASC(E$)+64),
1590 !" to return to command mode)"
1600 GOSUB 2410
1610 IF X-M<2 THEN 1940
1620 IF M=0 THEN 1650
1630 IF B$(M,M)<>CHR$(13) THEN 1660
1640 T=0
1650 !%3I,U,TAB(T1),
1660 FILL B1,1
1670 A$=INCHAR$(0)
1680 IF A$=E$ THEN 1960
1690 IF A$=B2$ THEN 1740
1700 IF Z1 THEN IF A$<" " THEN IF A$<>CHR$(13) THEN 1660
1710 IF A$>CHR$(31) THEN 1850 ELSE IF A$=CHR$(13) THEN 1850
1720 !"^",
1730 GOTO 1850
1740 IF T=0 OR M=0 THEN 1660
1750 X1=(ASC(B$(M,M))<32)
1760 M=M-1
1770 T=T-1
1780 IF E8 THEN A$=FNA$(A$)
1790 !A$,
1800 FILL H,EXAM(H)-2
1810 IF X1 THEN 1830
1820 GOTO 1660
1830 X1=0
1840 GOTO 1790
1850 M=M+1
1860 T=T+1
1870 B$(M,M)=A$
1880 IF A$<>CHR$(13) THEN IF A$<" " THEN A$=CHR$(ASC(A$)+64)
1890 IF A$<>CHR$(13) THEN !A$, ELSE 1910
1900 GOTO 1610
1910 !TAB(T2),%6I,X-M-1
1920 U=U+1
1930 GOTO 1610
1940 !
1950 !"              --->OUT OF SPACE<---",
1960 !
1970 GOSUB 2440
1980 !
1990 FILL B1,0
2000 INPUT"E-fix, Cn-change, D-display, A-add, RET-exit :",A$
2010 !
2020 IF A$="" THEN 2930
2030 A$=FNA$(A$)
2040 IF A$="E" THEN 2480
2050 IF A$(1,1)<>"C" THEN 2130
2060 ERRSET 2110,O,O
2070 U1=1
2080 U1=VAL(A$(2))
2090 IF U1=0 THEN U1=1
2100 GOTO 2150
2110 ERRSET
2120 GOTO 1540
2130 IF A$="A" THEN 2150
2140 IF A$<>"D" THEN 2790
2150 U=1
2160 A$=A$(1,1)
2170 IF A$="C" THEN IF U1=1 THEN 1540
2180 GOSUB 2400
2190 IF M<2 THEN 2340
2200 !%3I,U,TAB(T1),
2210 FOR A=1 TO M
2220   IF B$(A,A)<>CHR$(13) THEN 2290
```



CLIP TIPS



FIX YOUR PENCIL

If you are running one of the Electric Pencil I North Star disc versions you'll have cursed two bugs.

The first occurs if you leave Pencil for any reason or inadvertantly CLEAR ALL TEXT AFTER CURSOR without positioning the cursor properly (or just CRASH) before saving your laborously created text. The text is still there in memory but the program has lost its connecting pointers and you have lost your words. Re-entering Pencil at its normal zero address strikes the death blow as the program immediately zeroes all available memory and wipes the text slate clean.

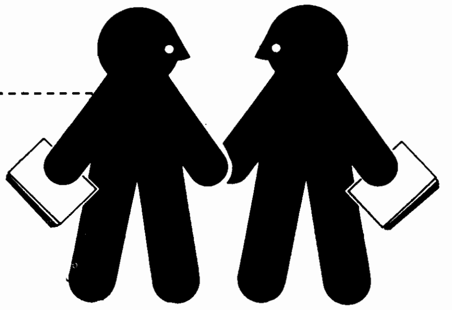
There is a simple way to protect yourself from disaster. Each time you start up, type a nonprinting line feed on the first text line, hit Control K to bring up the command menu, and then go back to the text entry mode. The Control K procedure sets your re-entry connection, should you need it later. You can now exit Pencil at will, re-enter at the normal zero address and the text will still be there.

The second bug affects those who use sheet feed printers like the Selectric. If you have tired of reaching for the panic stop near the end of every page, this fix may work for you. Change two bytes in the CALL and JMP addresses located at or near 0B27 and 0B2A repectively. Using your monitor search for the code CD 09 41 and change the 41 to 38 then search for C3 0B 26 and change the 26 to 22.

This will stop the printer every 53 lines to give you time to change paper. Hit return to start printing on the newly inserted sheet.

N* GO CPM = CPM GO N*

By Bob Stek
19 Mayfield Rd.
Regina Seshatchewan,S4V 0B7
Canada



*****EDITOR'S NOTE*****
THE WORD AROUND HERE IS THAT N* IS PLANNING TO GO THE CPM ROUTE. IN FACT, OF LATE THE WISPERING VOICES ARE GETTING STRONGER-THAT THEY HAVE ALREADY DECIDED TO JOIN THE BANDWAGON AND MAKE CPM THEIR OPERATING SYSTEM TOO. WE ARE JUST WAITING FOR THEM TO MAKE AN OFFICIAL ANNOUNCEMENT OF WHAT EVERYBODY ALREADY KNOWS.

NORTHSTAR BASIC ON CP/M:
TWO APPROACHES:

First, let me loudly proclaim: I love you, NorthStar BASIC! You were my first love, and you'll be my last love....I'll be true to you...(with apologies to the Shirelles(?)). Your multi-line functions, BCD arithmetic, and ease of random access file handling put that 'other' BASIC to shame. But let's face it, what is a nice BASIC like you doing hanging around a DOS like that?

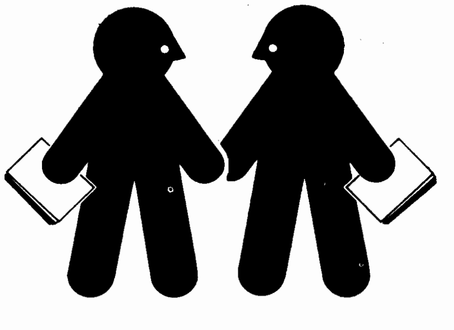
Actually, N* DOS isn't all that bad. For its time, it provided all any hobbyist needed - sort of like a fast cassette with random access capability. And Pavel Breder has made miraculous extensions with his add-on DOS-POWER modules. And, with release 5.2, we can (finally) move DOS down to 0HEX where it belongs and reclaim that otherwise awkward 8k chunk between 0H and 1FFFH. But, let's face it, probably 90% or more of N* owners use CP/M as well.

NorthStar, in their corporate wisdom, realizes that they have a bit of a vested interest in sticking to their own operating system. And you'll have to admit you do get a great BASIC interpreter FREE when you buy their hardware. You'll also have to admit their updates are dirt cheap compared to other systems.

But CP/M is the software bus for micros. So, why not make a version of N* BASIC which can run under CP/M? (SEE EDITOR'S NOTE).After all, one of the main advantages of CP/M is dynamic file allocation. That is, you don't have to specify how big a file is when it is first created. Under DOS you must manually create and 'type' a data file of a specific size. However, Xyzzx's Corollary to Murphy's Law says that no matter how big you made it, it's too small. So eventually you go through the rigamarole of creating a new, bigger file, then use the CF and CO utilities to get back to where you started from. With CP/M as your operating system, you don't even concern yourself with file size; it is handled automatically (after all, computers are supposed to be pretty good at that kind of stuff).

Another (potential) advantage of having N* BASIC under CP/M is that you could edit your programs with your expensive screen-oriented editor. How many times have you ached for a global search-and-replace function so you could change all those ' xxx PRINT ...' statements to ' xxx PRINT #1, ...' format?

Well, several very clever people have found the answer to your prayers. And if you don't mind paying along with praying, you too can have N* BASIC running under CP/M. If you live in California and know the right people, there is a hobbyist/consultant who will give/trade/sell? you a patched version he has done non-commercially - I don't have his permission to use his name. The two commercial firms who offer a patch are: The SoHo Group, 140 Thompson St., Suite 4-B, New York, N.Y. 10012 (theirs is called the Matchmaker), and InfoSoft Systems, Inc., 25 Sylvan Road South, Westport, Connecticut 06880 (NSBASIC (TM) interface).



THE MATCHMAKER

This is one of those clever Why-didn't-I-think-of-that-myself!! type patches (with all that implies!). The SoHo Group figured that BASIC works fine, just where it is (2A00 or 2D00, as the case may be); all that really needs changing is BASIC's links to the operating system to read and write to the disk. So their original patch (for 32k systems) loads at 100H, relocates itself above BASIC, and leaves the low end of memory from 100H free for BASIC programs. That 256 byte area is the I/O patch area from your DOS. This approach gives you about 11K of program area. Got a bigger program? - chop it into pieces and use 'CHAIN'!

But the SoHo Group did not sit on their collective ascii. Soon came the BIG Matchmaker for 48K and larger CP/M systems. This version also leaves BASIC where it is but extends program memory up to the base of the CP/M CCP area, giving up to 27K free in a 56K CP/M system.

BUT, you ask, what do I lose in this deal (remember, TANSTAAFL)? Well, obviously you _still_ lose that 8K or so from 100H to around 2900H if you are using the BIG Matchmaker, and you lose the ability to have more than an 11K program if you use the smaller version. You are limited to data files 64K in length or smaller. No more than five files _in total_ may be opened at any time during the execution of a given program, even if some are subsequently closed. CREATE, NSAVE, and DESTROY (though generally not needed since Matchmaker automatically creates a file when a SAVE command is issued) all give unpredictable and probably undesireable results. File names are limited to 6 characters plus a '.' followed by a file type - must be a 'B' for programs, anything else for data. And files are saved in the standard N* packed form so you can't use your standard editor on them. The only other thing you lose is $90 for the patch.

Want to run existing N* BASIC programs? Boot up with N* DOS, LF filename 100 (as long as the file is less than 31 blocks long); then insert your CP/M disk, boot up, and SAVE xx filename where xx is the size in blocks. If your program (or data file) is bigger than 31 blocks, you must break it up, save it in pieces, transfer it in pieces, and then put it together again under CP/M.

Want to run an MBASIC or CBASIC or xyzBASIC program which is already saved as an ASCII file under CP/M? Tough. Since N* BASIC saves and stores programs in a packed form (i.e., it uses one-byte 'tokens' for reserved words, etc.), it cannot read an ASCII file directly into program memory. Fortunately, says the SoHo Group, it is simple to patch the CIN routine under standard DOS and BASIC so that it can get characters from sequential memory locations starting with the beginning of said file previously loaded in free RAM. Unfortunately, "This comment is meant only as a

CONTINUED ON NEXT PAGE

N* GO CPM = CPM GO N*

CONTINUED FROM LAST PAGE

general guide. The SoHo Group cannot answer questions about, or provide programs for this procedure." Just what we need, another 'and this is left as an exercise for the student'. If it's so simple, why didn't they provide it?

NORTH STAR BASIC INTERFACE

Although this patch has been around longer, it still has only the potential for being head and shoulders above the Matchmaker. To wit: as of October 2nd, 1980 the InfoSoft patch is only useful for the old single-density N* controller board and BASIC. After it has been patched, however, it will work under double- (or quad-) density CP/M, but many of the utility programs which make it shine will not work without the single density controller board. A new version is promised for delivery in November, however. If it works as well with the newer controller board as the old system did, this becomes THE version to have. (InfoSoft talks of November delivery; if their past record with me is any indication, this means availability sometime in February.)

As far as I am concerned, InfoSoft (in particular, Rich Roth) has made the patch in a very professional way. Actually, it is more than a patch, it is a very well thought out _system_. First, BASIC is actually relocated down to begin at 100H, the TPA area of CP/M; result - you just got back that ol' 8K chunk. Second, and even more importantly, InfoSoft provides all the utility programs one finally realizes that you _need_ to make working with this bastardized BASIC tolerable if not enjoyable. For example, they provide:

NSDIR.COM - to list N* DOS directory under CP/M
NSDOS.COM - N* extended DOS with extra commands to allow copying to a CP/M disk
NSLOAD.COM - to load a file from a N* DOS disk to a CP/M disk
NSLIST.COM - to create a source file from ASCII to standard packed format
NSENTER.COM - to convert from an ASCII BASIC source format back to the standard packed format

If you have a two-drive system, these utilities make it a snap to do all the things you want to. After booting up CP/M, you can effectively turn drive B: into a N* DOS drive 2. That is, drive B: now accepts standard N* DOS disks. Either NSLOAD or NSDOS allows you to transfer old files to CP/M format with little more than a one-line command reminiscent of a PIP transfer. Even a one-drive system is tolerable, though you can't use NSDOS (what am I saying? NO one-drive CP/M system is tolerable!). NSLIST and NSENTER allow you to use your fancy screen editor to make those find-and-replace changes and allow you to edit those xyzBASIC programs to run under N* BASIC.

Once started, BASIC automatically sizes up memory and uses as much as it can (but a minimum 32k system is required). Control-P and control-S work as they do under CP/M. Logging in a new disk under BASIC may be accomplished with either a CALL (5) or a new command, RESET; either of these methods also allow selecting a new current disk with a CALL (5, disk#) or RESET diskletter:. N* devices are pre-assigned to standard CP/M devices:

0 - console
1 - list device
2 - reader/punch

Within BASIC a standard CP/M 8 character name (with optional disk specifier) is used to refer to program or data files (e.g., SAVE TESTPROG or OPEN #0,"B:TESTDATA"). Under CP/M proper, there are four standard file types:

N* type	CP/M type	
0	.NDT	N* data
1	.ABS	GO file
2	.NBP	N* BASIC program
3	.NDT	N* data file

CAT SUPPORTS AMBIGUOUS FILE NAMES NOW........

CAT now allows ambiguous filenames; e.g., CAT B:*.NBP lists all BASIC programs on drive B:. CREATE works fine, though you must specify a dummy file length if you also include the file type. DESTROY allows ambiguous filenames. SAVE automatically creates a new file if non-existent; NSAVE is no longer a part of BASIC. OPEN will automatically create a type 3 data file if it does not exist. InfoSoft has also modified user defined subroutines to accept 2 arguments and provides several standard CALLs.

If you get the impression that I favor the InfoSoft package, you are an astute observer. HOWEVER , I do not know if the planned revision of NSBASIC will follow exactly the above description of the old (although InfoSoft would have to be plenty stupid if it didn't). As far as I know, the old single-density version is still available for $75. The new version will cost $100, but it comes with the InfoSoft I/OS operating system; this is an act-alike CP/M compatible operating system which usually sells for $150 separately. Though this sounds like a fantastic deal for the N* user who doesn't have CP/M, I notice that their ad is directed toward Horizon owners, suggesting that if you don't have a Horizon, don't expect to just boot up their disk on your N* disk + ABC Computer + XYZ terminal. Caveat emptor.

Will either of these patches work for those rich speed freaks among us with the N* floating point boards? I don't know (but then again, I didn't ask). Or will owners of modified N* BASICs (e.g., Ashley's modified HDS BASIC or the excellent N*BUS BASIC with built-in super editor) be able to CP/M-ify their versions with either of these two packages? Again, I respond with a resounding "I don't know!". Will the honchos at N* in Berkeley ever realize that they have the best of both worlds and officially embrace CP/M as their second standard operating system and therefore come up with an official N* BASIC to run under CP/M? Would the Pope elope with a nice Jewish girl from Long Island and raise their children as Moonies?

EDITOR'S NOTE: Seems likely!!

CLIP TIPS

BUG....BUG....BUG....BUG....BUG

If you are reading or writing long data files beware that there is a bug in VERSION 4 and VERSION 5 Basic and Dos that does not permit access to a record that happens to cross the block boundry line on the disc between blocks 256 and 257 when the data is encoded as a string. Later versions of Dos and Basic have eliminated the bug.

COMMAND CONTINUED FROM PAGE SIX

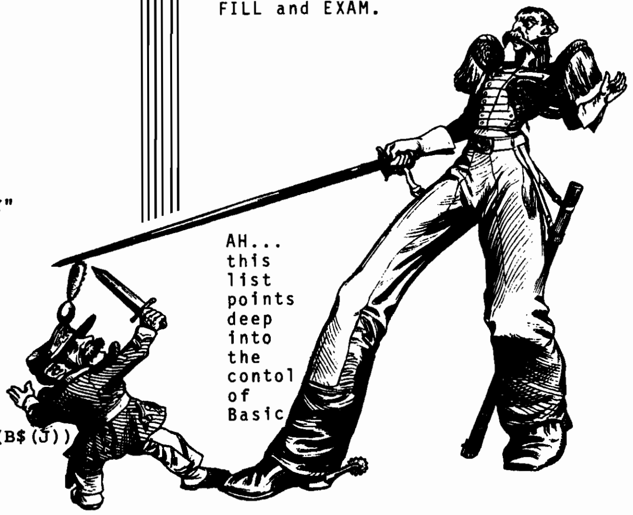
```
2230   !TAB(T2),%6I,X-A-1
2240   U=U+1
2250   IF A=M THEN 2330
2260   !%3I,U,TAB(T1),
2270   IF A$<>"C" THEN 2330
2280   IF U=U1 OR U>U1 THEN EXIT 2360 ELSE 2330
2290   IF B$(A,A)<" " THEN !"∧", ELSE 2320
2300   !CHR$(ASC(B$(A,A))+64),
2310   GOTO 2330
2320   !B$(A,A),
2330 NEXT A
2340 IF A$="C" THEN A$="A"
2350 GOTO 2390
2360 M=A
2370 T=0
2380 GOTO 1660
2390 IF A$="A" THEN 1610 ELSE 1970
2400 !C$,
2410 !
2420 !"No.",TAB(T1),"Command (Statement)",TAB(T2),"MEMORY"
2430 GOTO 2460
2440 IF A$="A" THEN IF T AND M>1 THEN !
2450 IF A$="D" THEN IF T AND M>1 THEN !
2460 !"---",TAB(T2),"------"
2470 RETURN
2480 IF R=0 THEN 2520
2490 FOR I=Y TO Y+57
2500   WRITE #0%I-Y,&EXAM(I)
2510 NEXT I
2520 J=1
2530 FOR I=Z TO Z+M-1
2540   IF R THEN WRITE#0%I-Y,&ASC(B$(J)) ELSE FILL I,ASC(B$(J))
2550   J=J+1
2560 NEXT I
2570 IF I=Z THEN FILL I<,E9 ELSE FILL I-1,E9
2580 IF R=0 THEN 2930
2590 IF I=Z THEN WRITE#0%I-Y,&E9 ELSE WRITE#0%I-Y-1,&E9,NOENDMARK
2600 CLOSE#0
2610 INPUT "TEST RUN (Y/N) :",A$
2620 A$=FNA$(A$)
2630 !
2640 X$=FNH$(Y) : X$=X$(3,7)
2650 Y$=FNH$(B+4) : Y$=Y$(3,7)
2660 F$="BYE"+CHR$(13)
2670 IF A1=Y AND F1=1 THEN 2690
2680 F$=F$+"TY "+N$+" 1"+X$+CHR$(13)
2690 IF A$<>"Y" THEN IF A1=Y AND F1=1 THEN 2940
2700 IF A$="Y" THEN 2730
2710 F$=F$+"JP"+Y$+CHR$(E9)
2720 GOTO 2740
2730 F$=F$+"GO "+N$+CHR$(E9)
2740 J=1
2750 FOR I=Z TO Z+LEN(F$)-1
2760   FILL I,ASC(F$(J))
2770   J=J+1
2780 NEXT I
2790 !C$,
2800 A=CALL(Y)
2810 END
2820 DEF FNH$(D)
2830 H1$="   "
2840 H=3
2850 FOR I=H TO 0 STEP -1
2860  H1=INT(D/(16∧I))
2870  IF H1>=10 THEN H1$=H1$+CHR$(ASC("A")+H1-10)
2880  IF H1< 10 THEN H1$=H1$+CHR$(ASC("0")+H1)
2890  D=D-(H1*(16∧I))
2900  NEXT I
2910 RETURN H1$+" HEX"
2920 FNEND
2930 !
2940 IF M<2 OR R THEN END
2950 IF A$<>"E" THEN END
2960 INPUT "TEST RUN (Y/N) :",A$
2970 A$=FNA$(A$)
2980 IF A$="Y" THEN A=CALL(Y) ELSE END
2990 !C$,
3000 END
3010 !C$,
3020 !TAB (20),"BUILDER1"
3030 !
3040 !"CAUTION: IF 'LI,CAT or LIST' USED THEN PAGING MUST BE OFF!!!"
3050 !
3060 !"PROGRAM SETUP:"
3070 !
```

CONTINUED ON PAGE FOURTEEN

CLIP TIPS

NORTH STAR BASIC TOKEN CODES

Basic reserved words are encoded in memory as token numbers to save space. Here are their DECIMAL equivalents manipulatible through FILL and EXAM.



128	LET	129	FOR
132	IF	133	READ
136	GOTO	137	GOSUB
140	STOP	141	END
144	FN	145	DEF
148	OUT	149	FILL
152	CLOSE	153	WRITE
157	DESTROY	158	CREATE
161	LIST	162	MEMSET
165	LOAD	166	CONT
169	NSAVE	170	SAVE
173	DEL	174	PSIZE
177	TO	178	THEN
180	ELSE	181	CHR$
184	STR$	185	NOENDMARK
190	FILEPTR	187	FILE
226	*	227	+
236	AND	237	OR
241	<>	239	=>
245	=	246	>
198	INT	204	LEN
202	SGN	203	SIN
216	FREE	217	INP
220	COS	221	LOG
130	PRINT	131	NEXT
134	INPUT	135	DATA
138	RETURN	139	DIM
142	RESTORE	143	REM
146	!	147	ON
150	EXIT	151	OPEN
155	CHAIN	156	LINE
159	ERRSET	160	RUN
163	SCR	164	AUTO
167	APPEND	168	REN
171	BYE	172	EDIT
175	CAT	176	STEP
188	SIZE	179	TAB
182	ASC	183	VAL
186	INCHAR$	189	FILESIZE
224	(	224	(
229	-	231	/
239	>=	240	<=
240	=<	244	<
247	NOT	225	
205	CALL	206	RND
196	SQRT	210	ATN
218	EXAM	219	ABS
222	EXP	223	TYP

ESP

ESP - TEST

The combination of microcomputers and parapsychology may seem to be an unlikely one, but in actuality they complement each other quite well. Many people experience unusual "coincidence" or play "hunches". However, trying to evaluate whether or not ESP played any part in a "lucky guess" is usually very difficult. Figuring out probabilities and their statistical significance can be quite tedious. But with a microcomputer, the mathematical calculations are a snap. In addition, the computer can generate random targets, give you immediate feedback about your parformance, and help you learn to develop your own ESP abilities. That is exactly what ESP - TEST does on your micro.

Parapsychology has existed as a branch of science for less than 100 years (and has been officially recognized in the scientific community for less than 10 years). Much of the early work was done by Dr. J. B. Rhine and his wife Louisa at Duke University. Beginning in the 1930's. They pioneered the use of standardized procedures and elaborate safeguards against cheating, accidental or otherwise. Perhaps the best known technique used by the Rhines is the use of Zener cards. There are five symbols: 0, +, , , and *, and a complete deck consists of five of each symbol for a total of 25 cards. If a deck is well shuffled so that its order is random, and if you guess at the order of cards, it is likely that you will be correct about five times by chance, alone. But suppose you repeat this experiment many times, and your average number of "hits" is 8. Is this an indication of ESP? If all other factors can be ruled out (that is, you weren't cheating!), then it is possible to evaluate the statistical significance of 8 hits instead of the expected 5. If the significance reaches a certain level, typically a probability of 1 in 20 or better that your score was by chance alone, then it is inferred that some other factor - ESP - was operating.

ESP - TEST is a computerized version of a deck of Zener cards, using the numbers 1 to 5 instead of the symbols. The program allows for testing of two types of ESP - clairvoyance and precognition. In the clairvoyance mode, you attempt to perceive the target which your computer has already randomly selected and has stored in memory. In the precognition mode, you

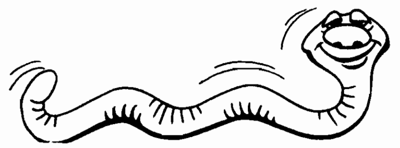
```
1000 REM  ******************************************************
1010 REM
1020 REM            ESP-TEST PROGRAM
1030 REM
1040 REM    WRITTEN BY ROBERT STEK,  PH.D.   MAY, 1978
1050 REM    REVISED FOR INSUA NEWSLETTER          1980
1060 REM    THIS PROGRAM SIMULATES A DECK OF ZENER CARDS USED
1070 REM    IN TESTING FOR ESP.
1080 REM
1090 REM  ******************************************************
1100 REM  C1$ = CLEAR SCREEN CHARACTER
1110 C1$ = CHR$ (11)
1120 REM  W = WIDTH OF VIDEO SCREEN
1130 W = 64
1140 DIM M1$(W / 2), Z$(W), N$(15)
1150 REM  FNM$ IS A FUNCTION TO CENTER MESSAGES ON THE SCREEN
1160 DEF FN M$(Z$) = M1$(LEN (Z$) / 2 + 3) + Z$
1170 PRINT C1$
1180 PRINT   PRINT
1190 PRINT FN M$("E S P  -  T E S T")
1200 PRINT   PRINT   PRINT   PRINT
1210 PRINT FN M$("Test your psychic powers!")
1220 PRINT   PRINT   PRINT   PRINT
1230 FOR I = 1 TO 50                  REM    WAIT FOR TITLE
1240    PRINT TAB (W - 1), CHR$ (13),  REM    TO BE READ
1250 NEXT I
1260 PRINT C1$
1270 INPUT "Hi! What is your first name?  ", N$
1280 REM  LINES 1290 - 1350 MAKE SURE THAT THE PLAYER'S NAME,
1290 REM  IN N$,  DOES NOT REMAIN AS ALL CAPITAL LETTERS
1300 FOR I = 2 TO LEN (N$)
1310    IF ASC (N$(I, I)) > 96 THEN 1350
1320    N$(I, I) = CHR$ (ASC (N$(I, I)) + 32)
1330    IF N$(I, I) <> CHR$ (64) THEN 1350
1340    N$(I, I) = CHR$ (32)  I = I + 1
1350 NEXT I
1360 PRINT   PRINT
1370 PRINT "Do you wish instructions, ", N$, "?  (Y or N) ",
1380 A$ = INCHAR$ (0)
1390 IF A$ = "" THEN 1380
1400 IF A$ = "N" THEN 1510
1410 IF A$ <> "Y" THEN 1380
1420 PRINT C1$
1430 PRINT "ESP-TEST was designed to test your Extra Sensory Perception"
1440 PRINT "under two conditions:"
1450 PRINT
1460 PRINT "1) Clairvoyance - in which you attempt to perceive a number"
1470 PRINT "which I have already generated and have stored in memory,  or"
1480 PRINT
1490 PRINT "2) Precognition - in which I will generate the number after"
1500 PRINT "you make your guess."
1510 PRINT   PRINT
1520 PRINT "So, ", N$, ", do you wish to test your Clairvoyance or your"
1530 PRINT "Precognition?  (C or P) ",
1540 T$ = INCHAR$ (0)
1550 IF T$ = "" THEN 1540
1560 IF T$ = "P" THEN 1710
1570 IF T$ <> "C" THEN 1540
1580 PRINT C1$
1590 PRINT FN M$("C L A I R V O Y A N C E")
1600 PRINT
1610 PRINT "   I will choose a number between 1 and 5.  Type in your ESP"
1620 PRINT "impression of what my number is,  then I will tell you what"
1630 PRINT "number I had chosen."
1640 PRINT "   If you wish to skip a trial because you are not sure of your"
1650 PRINT "answer,  then enter a '0' (zero).  That guess will not be"
1660 PRINT "counted."
1670 PRINT "   We will have 25 trials per run.  A minimum of 10 runs is"
1680 PRINT "suggested for statistical purposes."
1690 PRINT   PRINT "Good luck, ", N$, "!"
1700 GOTO 1840
1710 PRINT C1$
1720 PRINT FN M$("P R E C O G N I T I O N")
1730 PRINT
1740 PRINT "   First,  choose a number between 1 and 5.  This should be a"
1750 PRINT "number which you feel that I will pick.  After you choose your"
1760 PRINT "number,  I will make my random selection and tell you what it"
1770 PRINT "was."
1780 PRINT "   If you wish to skip a trial because you are not sure of"
1790 PRINT "your answer,  then enter a '0' (zero).  That guess will not"
1800 PRINT "be counted."
1810 PRINT "   We will have 25 trials per run.  A minimum of 10 runs is"
1820 PRINT "suggested for statistical purposes."
1830 PRINT   PRINT "Good luck, ", N$, "!"
1840 PRINT   PRINT
1850 R = RND ( - 1)  REM  INITIALIZE RANDOM NUMBER GENERATOR
1860 INPUT "How many runs (of 25 trials each) do you wish to make? ", R1
1870 R1 = INT (R1)
1880 IF R1 < 1 OR R1 > 50 THEN 1860
```

```
1890 FOR K = 1 TO R1
1900    H = 0  REM  INITIALIZE # OF CORRECT HITS TO 0
1910    FOR I = 1 TO 26
1920       IF I = 15 THEN I = I + 1  REM  LET'S NOT USE THE LETTER 'O'
1930       PRINT C1$  PRINT   PRINT
1940       PRINT FN M$("Trial "), CHR$ (I + 64)
1950       IF T$ = "P" THEN 1970
1960       R = INT (5 * RND (0) + 1)  REM  GENERATE TARGET #
1970       PRINT FN M$("Your impression ?")
1980       PRINT
1990       PRINT FN M$(" "),
2000       G$ = INCHAR$ (0)
2010       IF G$ = "" THEN 2000
2020       REM  MAKE SURE THAT GUESS IS IN THE RANGE 0 - 5
2030       IF ASC (G$) < 48 OR ASC (G$) > 53 THEN 2000
2040       PRINT G$  G = VAL (G$)
2050       IF T$ = "P" THEN R = INT (5 * RND (0) + 1)  REM  FOR PRECOGNITION
2060       IF G = 0 THEN 1930
2070       PRINT FN M$(""), R
2080       PRINT FN M$(" was the target number.")
2090       PRINT
2100       IF G = R THEN 2110 ELSE 2230
2110       H = H + 1  REM  WE GOT A HIT! INCREMENT HIT COUNTER
2120       REM  GIVE DIFFERENT FEEDBACK DEPENDING ON # OF HITS
2130       IF H >= 9 THEN 2210
2140       ON H GOTO 2150, 2150, 2150, 2150, 2150, 2170, 2170, 2190
2150       PRINT FN M$("  Correct!")
2160       GOTO 2230
2170       PRINT FN M$("Good,  "), N$, "!"
2180       GOTO 2230
2190       PRINT FN M$("Very good,  "), N$, "!!"
2200       GOTO 2230
2210       PRINT FN M$("Fan-n-n-tastic,  "), N$, " !!!"
2220       PRINT FN M$("You show real promise!")
2230       PRINT   PRINT   PRINT   PRINT
2240       INPUT "Press RETURN to continue", A$
2250    NEXT I
2260    REM  GIVE SUMMARY FEEDBACK ABOUT THIS RUN
2270    PRINT C1$
2280    PRINT   PRINT "You made", H, " hits in 25 trials."
2290    PRINT
2300    I = I - 2
2310    GOSUB 2500   REM  COMPUTE PROBABILITY OF THIS SCORE
2320    GOSUB 2660   REM  SHOW PROBABILITY LEVEL IF < .10
2330    PRINT   PRINT
2340    H1 = H1 + H  REM  INCREMENT HIT COUNTER OVER R1 RUNS
2350    IF K = R1 THEN 2380
2360    PRINT "Press RETURN to begin Run #", K + 1,
2370    INPUT " ", A$
2380 NEXT K
2390 H = H1
2400 I = I * R1
2410 PRINT   PRINT   PRINT
2420 INPUT "Press RETURN to continue", A$
2430 PRINT C1$
2440 PRINT "You made a total of", H, " hits in", R1, " runs."
2450 GOSUB 2500
2460 GOSUB 2660
2470 PRINT   PRINT   PRINT   PRINT   PRINT
2480 END
2481 REM*******************************************************************************************
2490 REM  SUBROUTINE TO COMPUTE PROBABILITY OF SCORE
2500 M = I * .2
2510 D = ABS (H - M)
2520 IF I = 25 THEN C1 = 0 ELSE C1 = .5  REM  CORRECTION FACTOR
2530 C = (D - C1) / SQRT (I * .2 * .8)  REM  FOR JUST 1 RUN
2540 T1 = 1 / (1 + .2316419 * C)
2550 P1 = 3.1415926535
2560 E1 = 2.7182818285
2570 B1 = .31938153
2580 B2 =  - .356563782
2590 B3 = 1.781477937
2600 B4 =  - 1.821255978
2610 B5 = 1.330274429
2620 F1 = (1 / SQRT (2 * P1)) * (1 / (E1 ∧ ((C * C) / 2)))
2630 P=F1*(B1*T1+B2*T1*T1+B3*T1*T1*T1+B4*T1∧4+B5*T1∧5)*2
2640 RETURN
2641 REM****************************************************
2650 REM  SUBROUTINE TO PRINT PROBABILITY LEVEL IF < .10
2660 IF P < .10 THEN 2700
2670 PRINT "This is not statistically significant, ", N$, "."
2680 PRINT "But keep on trying!"
2690 RETURN
2700 PRINT "This is statistically significant at the ",  %8F7, P
2710 PRINT "level of probability.  In other words, ", N$, ","
2720 PRINT "you could expect that result only once in", INT (1 / P), " times"
2730 PRINT "by chance alone.  Your ESP seems to be working pretty well!"
2740 RETURN
```

attempt to predict the target your computer will generate after you make your guess. After simulating a run of 25 trials, ESP - TEST computes the statistical significance of your results. If the results are probably due to chance, you will be told that. If, however, your results are not likely to be due to chance, you will be told the odds against your particular results.

In addition, ESP - TEST gives you immediate feedback to help you to learn to use your ESP. In the traditional use of Zener cards, the "percipient" doesn't know how well he is doing until all 25 cards have been used. He is then told how many "hits" he made out of 25. Recently, parapsychologist Charles Tart has pointed out that this procedure inhibits the learning of ESP. Psychologists have studied the process of learning for years and know that learning occurs best with immediate feedback given so that the person learns to discriminate correct from incorrect responses. Tart states that his understanding of ESP is that of

"listening to a still, small voice within, " trying to pick up an infrequent and subtle quality of your mental processes that conveys psi information. This difficult task must be done against the background of the incessant mental chatter of our ordinary state of consciousness. When you do not get any feedback until results are scored at the end of, say, twentyfive calls, you can't reliably recall whether a certain subtle feeling you had occured on call eleven, which you now know was a hit, or on call twelve which was a miss, etc.



With ESP - TEST you are given immediate feedback about whether your response was correct or incorrect. If ESP is operating, the feedback should help you to increase your ESP scores as you learn to trust certain feelings that accompany correct responses. It should be noted that learning occurs only if you have some ability to begin with. That is, if you continually score at chance levels, then the feedback only reinforces chance occurences and not ESP abilities. If however, you are able to score consistently above chance - even just 6 or 7 hits per 25 - then eventually you should learn to improve your scores. (An interesting project would be to write a program to keep track of your scores over a period of time and generate a learning curve.)

BETTER HORIZON I/O

```
*
*  WRITTEN BY:    JOE MAGUIRE              198Ø
*                 PO BOX 3742 DT
*                 ANCHORAGE, AK 9951Ø
*
*****************************************************
* THIS IS AN INPUT/OUTPUT ROUTINE FOR THE NORTHSTAR
* MICRODISK DOS.  THIS EXAMPLE IS TAILORED FOR THE
* HORIZON COMPUTER BUT BY ALTERING THE PORTS/STATUS
* BITS IT COULD BE USED WITH ANY SYSTEM.
*         IT HAS THE FOLLOWING FEATURES:
*
* SPEED CONTROL FOR SCROLLING ON A VIDEO TERMINAL.
* PRINT DEVICE SELECTION SIMILAR TO THAT OF CP/M.
*  SPACE BAR HALT TO FREEZE DISPLAY FOR VIEWING.
*         IMPROVED DELETE KEY OPERATION.
*
* NOTES: 1. THE IMPROVED DELETE FUNCTION TRAPS
* THE UNDERLINE CHARACTER (5FH) WHICH WAS OUTPUT
* ORIGINALLY BY THE DOS AND SUBSTITUTES A BACK-
* SPACE (Ø8H) INSTEAD. THIS PREVENTS AN UNDERLINE
* FROM BEING PRINTED FROM BASIC. IF AN UNDERLINE
* CHARACTER IS WANTED, PRECEED THE BASIC PRINT
* STATEMENT WITH THE FOLLOWING LINE:
*      FILL 1Ø557,255
* THIS DISABLES THE TEST FOR 5FH AT LOCATION
* 293DH IN THIS ROUTINE. TO RESTORE, USE:
*      FILL 1Ø557,95
* 2. IF MEMORY BOARDS WITH PARITY OPTION ARE USED,
* THEY SHOULD BE INITIALIZED WITH A SEPARATE PROGRAM
* AFTER THE DOS HAS LOADED. SEE THE N* MANUAL FOR
* THE PROPER CODE.  3. INPUT FROM DEVICES OTHER THAN
* A TERMINAL CONNECTED TO THE LEFT SERIAL PORT ARE NOT
* SUPPORTED BY THIS I/O ROUTINE. IF YOU NEED THIS
* FEATURE, AGAIN, SEE THE N* MANUAL FOR AN EXAMPLE.
*****************************************************
*
* * HORIZON PORTS AND STATUS BITS *
*
* PARALLEL PORTS -
*  Ø = I/O
*  1 = I/O
* 1ST SERIAL (TERMINAL) PORT -
*  2 = I/O
*  3 = STATUS/CONTROL
* 2ND SERIAL (PRINTER) PORT -
*  4 = I/O
*  5 = STATUS/CONTROL
* SERIAL STATUS BITS -
*  Ø = OUTPUT (CTS PIN OF USART)
*  1 = INPUT (CHAR IN INPUT BUFFER)
*  7 = CONTROL (DSR PIN OF USART)
* USART COMMAND CODES -
*  REFER TO 8251 DATA SHEET
* MOTHERBOARD PORTS -
*  6 = CONTROL/STATUS
*  7 = CONTROL/STATUS
* MOTHERBOARD STATUS BITS -
*  Ø = PARALLEL OUTPUT FLAG
*  1 = PARALLEL INPUT FLAG
* MOTHERBOARD COMMAND CODES -
*  2Ø (HEX) RESET PO FLAG
*  3Ø (HEX) RESET PI FLAG
*  6Ø (HEX) SET PO FLAG
*
```

```
2ØØØ              ORG 2ØØØH
                 *
2ØØØ =           BEGIN: EQU $
                 *
2ØØD             JUMPS: ORG BEGIN+ØDH
2ØØD C32529       JMP COUT
2Ø1Ø C3ØØ29       JMP CIN
2Ø13 C3CD29       JMP TINIT
2Ø16 C3C129       JMP CONTC
                 *
29ØØ             START: ORG BEGIN+9ØØH
                 *
                 * * CHARACTER INPUT ROUTINE *
                 *
                 CIN:
29ØØ CD8E29       CALL INPUT      ;1ST SERIAL IN (TERMINAL)
29Ø3 FE1Ø         CPI 1ØH         ;PRINT ON? (CONTROL/P <DEVICE>)
29Ø5 CAØE29       JZ PRINT
29Ø8 FE7F         CPI 7FH         ;WAS IT A DELETE?
29ØA CØ           RNZ             ;RETURN IF NOT
29ØB 3E5F         MVI A,5FH       ;RETURN WITH N* DELETE CHAR
29ØD C9           RET
                 *
29ØE CD8E29      PRINT CALL INPUT
2911 FE34         CPI 34H  ;THIS AND THE CPI 3ØH BELOW SET THE
                           ;LIMITS OF THE DEVICE #.  IT IS
                           ;CURRENTLY SET FROM Ø TO 3.
2913 D22129       JNC P1
2916 FE3Ø         CPI 3ØH
2918 DA2129       JC P1     ;IF THE DEVICE # IS OUT OF RANGE, THEN
                           ;PRINTING WILL BE TURNED OFF AND THE
                           ;CHARACTER SENT TO THE TERMINAL.
291B 32FF29       STA STORE
291E C3ØØ29       JMP CIN
2921 32FF29      P1 STA STORE
2924 C9           RET
                 *
                 * * CHARACTER OUTPUT ROUTINE *
                 *
                 * THE A REGISTER IS CHECKED FOR THE DEVICE
                 * NUMBER AND THE OUTPUT IS SENT ACCORD-
                 * INGLY.  EITHER THE CONTROL/P <DEVICE>
                 * OR THE PRINT#- AND LIST#- COMMANDS OF
                 * NORTHSTAR BASIC MAY BE USED TO FORCE
                 * PRINTING. HOWEVER, PRINT#- COMMANDS
                 * FROM BASIC (OR THE MONITOR) WILL NOT
                 * ECHO TO THE TERMINAL
                 *
                 COUT:
2925 FEØ1         CPI 1       ; 1 = 2ND SERIAL
2927 CA9A29       JZ SER2O
292A FEØ2         CPI 2
292C CAA829       JZ PARAO    ; 2 = PARALLEL OUTPUT
292F FEØ3         CPI 3
2931 ØØ           NOP         ; 3 = USER ROUTINE
2932 ØØ           NOP         ;SPACE FOR YOUR JUMP
2933 ØØ           NOP
                 *
2934 DBØ3        TERM IN 3        ;DEVICE #Ø (TERMINAL)
2936 E6Ø1         ANI 1
2938 CA3429       JZ TERM
293B 78           MOV A,B         ;GET CHAR TO BE OUTPUT
293C FE5F         CPI 5FH         ;IS IT DELETE?
293E C25Ø29       JNZ T1
2941 Ø6Ø8         MVI B,8         ;MOVE IN BACKSPACE
2943 CD3429       CALL TERM       ;SEND IT OUT
2946 Ø62Ø         MVI B,2ØH       ;NOW A BLANK
2948 CD3429       CALL TERM
294B Ø6Ø8         MVI B,8         ;BACKUP AGAIN
294D C33429       JMP TERM
                 *
                 * VIDEO SCROLL SPEED ROUTINE *
                 *
295Ø E5          T1 PUSH H        ;SAVE HL
2951 2AFD29       LHLD SPEED      ;LOAD SPEED BYTE
2954 2C           INR L           ;MAKE SURE L NON ZERO
2955 AF           XRA A
2956 2B          TIMER DCX H      ;START DELAY LOOP
2957 BC           CMP H
2958 C25629       JNZ TIMER
295B E1           POP H           ;RESTORE HL
295C 78           MOV A,B         ;GET CHAR TO OUTPUT
295D D3Ø2         OUT 2           ;SEND IT OUT
295F DBØ3         IN 3            ;SETTING SPEED?
2961 E6Ø2         ANI 2
2963 CA7A29       JZ T2           ;ZERO SAYS NO
2966 CD9529       CALL IN1
2969 FE3A         CPI 3AH         ;HIT ANY NUMBER KEY DURING
                                  ;OUTPUT TO SET SCROLL SPEED.
                                  ;Ø=FASTEST     9=SLOWEST
```

```
296B D27A29      JNC T2
296E FE30        CPI 30H
2970 DA7A29      JC T2
2973 E60F        ANI 0FH
2975 17          RAL
2976 17          RAL
2977 32FE29      STA SPEED+1
                *
297A 3AFF29     T2 LDA STORE       ;PRINT ON?
297D FE31        CPI 31H           ;PRINTER
297F CA9A29      JZ SER2O
2982 FE32        CPI 32H           ;PARALLEL
2984 CAA829      JZ PARAO
2987 FE33        CPI 33H           ;USER
2989 00          NOP
298A 00          NOP
298B 00          NOP
298C 78          MOV A,B
298D C9          RET
                *
298E DB03       INPUT IN 3
2990 E602        ANI 2
2992 CA8E29      JZ INPUT
2995 DB02       IN1 IN 2
2997 E67F        ANI 7FH
2999 C9          RET
                *
299A DB05       SER2O IN 5         ;2ND SERIAL PORT
299C E601        ANI 1
299E CA9A29      JZ SER2O
29A1 78          MOV A,B
29A2 FE7F        CPI 7FH           ;DONT SEND DELETE TO PRINTER
29A4 C8          RZ                ;IT GOOFS IT UP
29A5 D304        OUT 4
29A7 C9          RET
                *
29A8 DB06       PARAO IN 6   ;PARALLEL PORT
29AA E601        ANI 1
29AC CAA829      JZ PARAO
29AF 3E20        MVI A,20H   ;RESET PO FLAG
29B1 D306        OUT 6
29B3 78          MOV A,B
29B4 F680        ORI 80H     ;SET STROBE FALSE
29B6 D300        OUT 0       ;SEND CHARACTER
29B8 EE80        XRI 80H     ;TOGGLE STROBE
29BA D300        OUT 0
29BC D300        OUT 0       ;AGAIN
29BE E67F        ANI 7FH     ;RESTORE ASCII
29C0 C9          RET
                *
                * CONTROL C TEST *
                *
                CONTC:
29C1 CD9529      CALL IN1
29C4 FE03        CPI 3
29C6 C8          RZ        ;RETURN IF CTRL/C
29C7 FE20        CPI 20H   ;SPACE BAR HALT?
                           ;SPACE BAR GIVES ONE LINE. ANY
                           ;OTHER KEY KEEPS GOING.
29C9 CC8E29      CZ INPUT
29CC C9          RET
                *
                * HORIZON INITIALIZATION *
                *
                TINIT:
29CD AF         INIT XRA A
29CE D306        OUT 6
29D0 D306        OUT 6             ;REQUIRED 3 TIMES TO ARM USART
29D2 D306        OUT 6
29D4 3ECE        MVI A,0CEH        ;SEND USARTS COMMANDS
29D6 D303        OUT 3
29D8 D305        OUT 5
29DA 3E37        MVI A,37H
29DC D303        OUT 3
29DE D305        OUT 5
29E0 DB02        IN 2
29E2 DB04        IN 4
29E4 3E20        MVI A,20H         ;RESET FLAGS
29E6 D306        OUT 6
29E8 3E30        MVI A,30H
29EA D306        OUT 6
29EC C9          RET
                *
29FD             ORG START+0FDH
                *
29FD 0000       SPEED DW 0
29FF 00         STORE DB 0
                *
2A00             END
                *
```

ESP continued from page 11

Suggestions for using ESP - TEST

1. ESP - TEST generates what is known as an open deck. That is, there is not necessarily an equal distribution of each number in a run of 25 trials. With a "closed deck" each number occurs exactly 5 times:

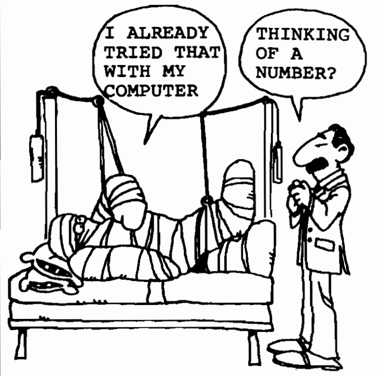
#	frequency
1	5
2	5
3	5
4	5
5	5

On trial #22 a person's impression may be that he should guess a "3"; however, he may realize that he has guessed a "3" five times already and could erroneously conclude that he should change his guess. With an open deck each number is equallv likely to appear; over many runs, each number will have appeared one-fifth of the time on the average. In a single run of 25 trials, however, you may get a distribution such as:

#	freauency
1	7
2	3
3	6
4	5
5	4

This way, a person may feel free to guess a particular number as many times as he wishes without having to worry about how often he does so.

2. A person's belief in ESP can affect his scores. Gertrude Schmeidler conducted a well known experiment with "sheep" (participants who beleived in ESP) and "goats" (those who did not believe in ESP). The sheep tended to score above chance level, thus confirming their beliefs. The goats tended to score below chance level, thus thinking they were confirming their negative beliefs. However, scores which are significantly below chance level do as much to show the existence of ESP as scores which are significantly above chance level. This is because a person cannot score significantly below chance expectation unless he is using ESP on an unconscious level to identify some targets correctly and then unconsciously but deliberately changing his guess in order to



CONTINUED ON PAGE 14

HELP HELP

YOUR COMPUTER TIME WANTED
TO HELP REUNITE FAMILIES

Vietnamese families separated during the "boat people" exodus are being reunited through a network of 1000's of micro computers. You can join the network to assist the UN and other agencies in locating relatives of the boat people now scattered around the world.

Raw data is available in a printout or via modem.

The data base creation and sorting programs are run on independent computers through the coordination of Family Reunification Services, 7203 Huntercrest Rd. N.W., Calgary, Alberta T2K 4J9 Canada (403)274-3894. Contact them for details.

ESP CONTINUED FROM PAGE 13

confirm his beliefs. ESP - TEST will also identify people who score consistently lower than would be expected by chance. You might compare scores of friends who are "sheep" and "goats".

3. Much current research suggests that a state of relaxed alertness may be conducive to ESP. If you are trying to pay attention to what may be very subtle cues in your awareness, it may be easier if you are not preoccupied with other thoughts. Some researchers suggest that hypnosis, meditation or deep muscular relaxation may promote ESP. You could rate yourself at various times as "more relaxed", "average", or "less relaxed". Then compare your scores under these conditions to see if they differ.

4. Automated recordkeeping could be built into ESP - TEST. You might add a section that creates a data file based on a person's name. This section would automatically keep a record of a person's scores, and would update the file each time that person used ESP - TEST. This way, you would be less likely to "forget" about the low or non-significant scores and not just remember the high scores. You could also keep notes about conditions during the trials, e.g. "Bob felt very relaxed, but was not confident. He is a 'goat'."

5. Another project would be to modify ESP - TEST to keep track of "confidence calls". People may have hunches about when they are right. Some studies have shown that when percepients are asked to indicate whether they have particular confidence in a call, the proportion of correct calls is significantly higher than on those calls for which the person did not indicate confidence.

COMMAND CONTINUED FROM PAGE NINE

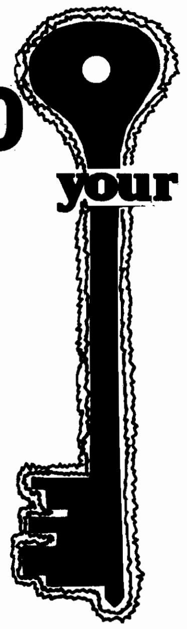
```
3080 !"(0) BLOCKS USED: MAXIMUM NUMBER OF 256 BYTE BLOCKS THAT"
3090 !"                  WILL BE USED TO CREATE THE FILE."
3100 !"                  IF THE FILE ALREADY EXISTS AND THE"
3110 !"                  BLOCKS USED IS CHANGED TO MORE THAN"
3120 !"                  PRESENT FILES LENGHT,THEN BLOCKS USED"
3130 !"                  WILL BE DECREASED BY THIS PROGRAM TO"
3140 !"                  EQUAL CURRENT FILE LENGHT."
3150-!"                  IF FILE ACCESS IS DISABLED THEN THE"
3160 !"                  ABOVE DOES NOT APPLY."
3170 GOSUB 3810
3180 !"GO ADDRESS     : ADDRESS IN MEMORY WHERE FINISHED"
3190 !"                 PROGRAM WILL EXECUTE USING 'GO','JP',"
3200 !"                 'CALL' OR 'EXecute'"
3210 !
3220 !"BASIC ORIGIN   : ORIGIN ADDRESS OF BASIC"
3230 !
3240 !"DOS   ORIGIN   : ORIGIN ADDRESS OF DOS"
3250 !
3260 !"CLEAR SCREEN   : CLEAR SCREEN CHARACTER"
3270 !
3280 !"CURRENT DATA   : IF 0 THEN READ CURRENT COMMANDS"
3290 !"                 FROM MEMORY (OR DISK)."
3300 !"                 IF 1 THEN IGNORE THIS DATA."
3310 GOSUB 3810
3320 !"CONTROL CHR$   : IF 0 THEN ALL CONTROL CHARACTERS WILL"
3330 !"                 BE ACCEPTED BY THIS PROGRAM DURING"
3340 !"                 INPUTS OF COMMANDS FOR THE USE OF"
3350 !"                 FINISHED FILE. IN ORDER TO INDICATE"
3360 !"                 THAT CONTROL CHARACTER WAS PRESSED"
3370 !"                 '^' WILL BE PRINTED (CTR/C - ^C)"
3380 !"                 IF 1 THEN ALL,EXEPT 'RETURN' WILL BE"
3390 !"                 IGNORED (CTR/C IS DISABLED DURING"
3400 !"                 COMMAND TO FILE INPUTS)."
3410 !
3420 !"WRITE TO FILE : IF 0 THEN ALL READING/WRITING WILL BE"
3430 !"                 DONE IN MEMORY ONLY"
3440 !"                 IF 1 THEN ALL READING/WRITING WILL BE"
3450 !"                 DONE IN MEMORY AND TO/FROM FILE"
3460 GOSUB 3810
3470 !"NAME OF FILE   : NAME OF FILE ON DISK TO BE USED."
3480 !"                 IF FILE DOES NOT EXITS, THEN REQUEST"
3490 !"                 TO CREATE NAMED FILE WILL BE MADE.
3500 !"                 IF ANSWER IS 'Y' OR YES
3510 !"                 THEN NAMED FILE WILL BE CREATED AS"
3520 !"                 THE TYPE 1;  OTHERWISE YOU WILL BE"
3530 !"                 RETURNED TO THE BEGINING OF PROGRAM."
3540 !
3550 !"END INP CHR$   : PRESSING THIS CHARACTER DURING INPUTS"
3560 !"                 OF COMMANDS TO THE FILE WILL TERMINATE"
3570 !"                 THE INPUTS AND RETURN TO COMMAND MODE"
3580 !"                 WILL BE MADE."
3590 GOSUB 3810
3600 !"COMMAND MODE:"
3610 !
3620 !"E.....ENTER COMMANDS, IF ANY, TO THE FILE"
3630 !"      IF NO COMMANDS PRESENT, THEN END PROGRAM"
3640 !"      IF COMMAND(S) PRESENT THEN REQUEY_T A TEST RUN"
3650 !
3660 !"C.....ERASE PREVIOUS COMMANDS AND GET NEW COMMANDS"
3670 !"Cn....ENTER NEW COMMANDS COMMENCING WITH NUMBER 'n'"
3680 !
3690 !"A.....ADD NEW COMMANDS TO THE PRESENT COMMANDS"
```

CLIP TIPS

BASIC ENTRY POINTS AND FLAGS

	SINGLE DENSITY		DOUBLE DENSITY		STANDARD VALUE
ENDBAS	2A06-7	(10758-9)	2D06-7	(11526-7)	
HIGHMEM	2A09-A	(10761-2	2D09-A	(11529-30)	
LINE	2A0E	(10766	2D0E	(11534)	80
AUTOS	2A0F	(10767)	2D0F	(11535)	1
BOOTPROM	2A10	(10768)	2D10	(11536)	224 E8H
PAGES	2A13	(10771)	2D13	(11539)	24
DELECHO	2A17	(10775)	2D17	(11543)	95 08
CONTROL C	2A18	(10776)	2D18	(11544)	0
COLDSTART	2A00	(10752)	2D00	(11520)	
WARMSTART	2A04	(10756)	2D04	(11524)	
CONTINUE	2A14	(10772)	2D14	(11540)	
PRINTHDTABLE	2A11-2	(10768-70)	2D11	(11537-8)	

THE KEY TO your Minifloppy data



DCOM INTERFACE FOR BASIC
By R. Beaver

Summary

This article describes a method for interfacing North Star Basic programs to the DOS disk command (DCOM) function. This permits the programmer to access blocks of disk data from a Basic program without requiring a file to exist on the disk.

The approach used here may be used to assist interfacing Basic to other machine routines, including other DOS commands, permitting the user to take better advantage of the speed of machine programs without sacrificing the programming advantages of Basic.

Introduction

North Star Basic provides a means of accessing disk files using OPEN, CLOSE, READ, and WRITE statements. However, these statements require that a file exists, and that its name is known.

For certain applications where a filename is not known or a file does not exist, these statements are not the right "set" to use, and the programmer is faced with the problem of trying to get from here to there without the right tools. For these applications the programmer may find the CALL function provided by Basic can help extend Basic capabilities to fill specific needs. Lets see how this might provide a necessary Connection between your basic program and the DOS disk command function.

First, the process is not as simple as including a program statement such as CALL(DCOM,PARAMETER), since the CALL function passes one parameter, and the DCOM routine requires five parameters.

The way to fill this information gap is to write a machine routine which "passes the parameters" and correctly executes the DCOM routine, as demonstrated by the 8080 routine which follows.

Naturally, the sneaky way to get this routine into memory is to have Basic write it using FILL statements, as is shown in the Results section of this article.

8080 ASSEMBLY ROUTINE FOR INTERFACING BASIC'S CALL FUNCTION TO NORTH STAR'S DOS DISK COMMAND (DCOM) ROUTINE. SEE NORTH STAR SOFT-DOC REVISION 2.1 (pg F4).

```
Instruction            Remarks

MVI  A,BLOCKS       :BLOCKS FOR DCOM
LXI  B,COMUNIT      :BC=COMMAND & UNIT
LXI  H,RAM          :RAM ADDRESS
XCHG                :EXCHANGE DE & HL
CALL DCOM           :ISSUE DISK COMMAND
RNC                 :RETURN IF NO ERROR
JMP  BASIC          :ON ERROR JUMP BASIC
```

This routine has advantages of simplicity and flexability. Note that this routine is relocatable without modification.

Calling parameters are: B=Blocks, C=Command (0=write, 1=read, 2=verify), U=Disk Unit (add 128D for double density), R=Ram Address for read/write/verify activity, M=Ram Address for the routine, and A=Disk Address (blocks). A jump to Basic (2A04H) is made if DCOM fails because of illegal arguments.

Note that M+15 should be changed from 42 to 45 if Basic origin is 2D00H. Note also that this routine assumes standard origin DOS at 2000H.

Results

The object of the Basic function shown below is to write a machine routine into memory at a location specified by the user. The routine is executed by a CALL instruction which issues the disk command using the DOS DCOM routine.

```
200 DEF FNC(B,C,U,R,M)
205 FILL M,62
206 FILL M+1,B
210 FILL M+2,1
211 FILL M+3,U
212 FILL M+4,C
215 FILL M+5,33
216 FILL M+6,INT(R-256*INT(R/256))
220 FILL M+7,INT(R/256)
225 FILL M+8,235
230 FILL M+9,205
231 FILL M+10,34
232 FILL M+11,32
235 FILL M+12,208
240 FILL M+13,195
241 FILL M+14,4
242 FILL M+15,42
245 RETURN M
250 FNEND
```

Example

Assuming program lines 200-250 are present, and RAM is available in 0-8K region, the following Basic statement will read a double density directory from drive #1 (Blocks=8, Unit=1+128, Disk Address=0, Command=1) using RAM address 4096-4112 for the DCOM interface routine, and place the directory data in RAM beginning at address 4200. The variable Q has no significance except to provide proper syntax for the statement. Remember to modify line 242 for Basic with origin 2D00H.

```
Q=CALL(FNC(8,1,129,4200,4096),0)
```

Conclusion

Interfacing Basic to machine routines can be fairly straight-forward, and can provide an extension of Basic capabilities to meet specific needs. The DCOM interface presented here provides a simple example. As it turns out, this simple example has some rather interesting practical applications, which are described in accompaning articles.

INTERNATIONAL NORTH STAR USERS ASSOCIATION MEMBERS CAN RECEIVE THE NEW DOS AND BASIC (VERS 5.2) FOR ONLY $10 INSTEAD OF $25 -- SELECT EITHER SINGLE OR DOUBLE DENSITY -- ▶▶▶ SEE PAGE TWO

FIND IT FILING EFFICIENCY made easy

LIBRARY INDEX UTILITY PROGRAM

The following program operates on the SUPER INDEX database described in 'AN INDEX FOR YOUR SOFTWARE LIBRARY'. It provides programming for DISPlaying any directory file on the index disk, SCANning of all directory files for a given program, and LISTing the entire index on a user defined output device. To run this program you must have one North Star drive and 24K RAM allocated 8-32K, and have a completed SUPER INDEX. The program uses the 'DCOM INTERFACE FOR BASIC' for all disk activity.

a)In the listing,the program line separator appears as ¢.

b)The program may be auto-started using procedures described in your North Star documentation. In addition, the program may be customized by deleting lines 100-160 (providing Basic auto-starts with memset 28600),by deleting lines 310 and 315 and 325 (substitute these with appropriate assignment statements for variables D and B0), and by removing line 350.

c)Take note of lines 295 and 300 for customization with another memset and for columns shown on LIST or DISPlay activity.

d)The program assumes that all directory files begin with ascii '*' (2A hex or 42 decimal). If another designator is used, then appropriate changes may be made to lines 640 and 760.

```
*******************************************************************
                              LIBUTIL
                             R. Beaver
*******************************************************************

100 !¢!¢!¢!¢!¢!¢!"          LIBRARY INDEX UTILITY"¢!
110 !"*********************************************"
120 !"**** REQUIRES 24K SYSTEM ALLOCATED 8-32K ****"
130 !"****          AND MEMSET 28600          ****"
140 !"****      TYPE 'RUN 200' AFTER MEMSET   ****"
150 !"*********************************************"¢!
160 STOP
200 REM DCOM INTERFACE FOR BASIC
210 DEF FNC(A,B,C,D,M)
220 FILL M,62 ¢FILL M+1,A ¢FILL M+2,1¢FILL M+3,C¢FILL M+4,B
230 H=INT(D/256)¢L=INT(D-256*H)
240 FILL M+5,33 ¢FILL M+6,L¢FILL M+7,H ¢FILL M+8,235
250 FILL M+9,205 ¢FILL M+10,34 ¢FILL M+11,32 ¢FILL M+12,208
260 REM JUMP TO BASIC (2A04H) ON ERROR
270 FILL M+13,195¢FILL M+14,4 ¢FILL M+15,B0
280 RETURN M¢FNEND
290 REM *INITIALIZE*
295 REM P9=COLUMNS/DISPLAY  P8=COLUMNS/LIST  M=MAX RAM-4168
300 M=28600¢P9=5¢P8=6
305 M1=M+1¢M2=M+20¢D1=M+50¢D2=M+2100
310 !"INDEX DISKETTE DIRECTORY DENSITY"
315 INPUT"(1) SINGLE DENSITY  OR (2) DOUBLE DENSITY ?",D
320 D=INT(D)¢IFD<1ORD>2THEN315
325 !¢!"BASIC ORIGIN"¢INPUT"(1)=2A00H  (2)=2D00H  :",B0
330 IFB0=2THENB0=45ELSEB0=42
335 IFD=2THENU1=129ELSEU1=1
340 IFD=2THENA1=8ELSEA1=4
345 IFD=2THENN1=127ELSEN1=63
350 INPUT"PLACE INDEX DISK IN DRIVE #1 AND PRESS RETURN ",A$
360 P1=0¢Q=CALL(FNC(A1,1,U1,D1,M1),P1)
365 REM *MENU*
370 !¢!"LIBRARY UTILITY MENU."
380 !" EXIT-EXIT TO BASIC."
390 !" DISP-DISPLAY A GIVEN DIRECTORY."
400 !" SCAN-SCAN INDEX FOR A GIVEN PROGRAM."
410 !" LIST-PRINT THE ENTIRE INDEX."
420 !¢INPUT" <UTIL>: ",A$¢IFLEN(A$)<2ORLEN(A$)>4THEN370
430 IFA$(1,2)="EX"THEN900¢IFA$(1,2)="DI"THEN450
440 IFA$(1,2)="SC"THEN620¢IFA$(1,2)="LI"THEN750¢GOTO370
445 REM *DISPLAY*
450 INPUT"DISPLAY FILENAME ",F$¢!
460 L=LEN(F$)¢IFL<1THEN420
470 FORI=0TON1¢L1=D1+16*I
480 FORJ=0TOL-1¢IFCHR$(EXAM(L1+J))<>F$(J+1,J+1)THENEXIT500¢NEXTJ
490 EXIT520
500 NEXTI¢!"FILE NOT FOUND."¢!¢GOTO450
510 REM NOW GET THE FILE DIRECTORY AND DISPLAY IT
520 IFEXAM(L1+10)=8OREXAM(L1+10)=4THEN530
525 !"FILE LENGTH ERROR.."¢GOTO420
530 GOSUB950¢S=0¢N=0¢P=0
550 FORI=0TON2¢L2=D2+16*I¢IFEXAM(L2)=32THEN590¢P=P+1¢N=N+1
560 !%4I,EXAM(L2+12)," ",¢FORJ=0TO7¢!CHR$(EXAM(L2+J)),¢NEXTJ
570 IFP<P9THEN580¢P=0¢!
580 S=S+256*EXAM(L2+11)+EXAM(L2+10)
590 NEXTI¢IFP>0THEN!
600 !"FILES= ",%3I,N,"     STORAGE= ",%3I,S," BLOCKS"
610 !¢!¢GOTO450
```

Continued Next Page

AN INDEX FOR YOUR SOFTWARE LIBRARY
By R. Beaver

Summary

This article describes a simple method of creating and maintaining a disk file index. The indexing system described here provides directory backup for your disks, can be set up and maintained using DOS functions already available, and can be used to compliment your existing disk library system.

Introduction

Looking for that program that you know you stored somewhere in that pile of disks? Tired of typing endless names and descriptions into your homebrew librarian? Well, here's an idea that might help you get your library better organized.

Someone has already figured out a way to keep track of filename, disk address, length, filetype, and type dependent information. Also available is the software necessary to update this information as it is changed.

Where can you get this wonderful system? Look no farther than your North Star DOS, and any disk formatted using the DOS.

The directory file on your disk begins at disk address 0 and is 1024 bytes long (single density) or 2048 bytes long (double density). It contains a simple data base which permits the DOS to find files on the disk (so you can think about other things). Most importantly, it is standard. Every North Star User has this data base and is using it.

Suppose that you had all the directories in your disk library (or as many directories as would fit) on a single diskette. Why, with all that information at your fingertips you could "rule the world" (or at least a small chunk of it). Naturally, this SUPER INDEX would have to be organized to indicate which directory and disk belonged to each other, information which is not contained in the disk directory.

Identifying Disks

Take a stack of tested and initialized diskettes and number these sequencially by sticking a small label on the outside. Preceed each number with a * or other special character which is not used to preceed filenames other than directory filenames. I recommend a

Continued Next Page

starting disk number of *101 so that all disks will have the same name length.

On each diskette, create a file with the corresponding disk name. These files can be created using DOS, must begin at disk address 0 and be of appropriate length to include the entire disk directory. The appropriate DOS commands for this are:

```
*CR *101 4 0 (single density direc)
*CR *101 8 0 (double density direc)
```

The directory file is not adversly affected by this process, except to include the new file entry, even when there are other files present. Thus, you can name your old (full) diskettes without reinitializing them, providing there is still room for another directory file entry.

For those of us who are using both sides of our diskettes without the benefit of quad density systems, it is necessary to distinguish between the two sides. The simple solution to this is to tack an A or B to the directory file name to identify front vs. back (ie *101A vs. *101B).

Finally, each of the directory files is copied to a corresponding disk file on an index diskette (ie *LIB). Use the DOS file copy routine (CF) to accomplish this.

Voila! The SUPER INDEX containing the directories of all your diskettes. If you glitch a directory, you can copy it back from the index. As disk directories are changed, copy the new directory to the index diskette.

Conclusion

The disk library index system described here can be set up and maintained using North Star DOS functions already available. In addition, it also provides a <u>standard</u> database for software file searches and other functions useful to the user. Finally, it can be used to compliment your existing library system by providing up to date information on program location and disk storage availability.

```
615 REM *SCAN*
620 INPUT"SCAN FOR FILENAME : ",S$¢!
630 L0=LEN(S$)¢IFL0<1ORL0>8THEN420
640 FORI=1TON1¢L1=D1+16*I¢IFEXAM(L1)<>42THEN730¢D$=""
650 FORJ=0TO7¢D$=D$+CHR$(EXAM(L1+J))¢NEXTJ
660 IFEXAM(L1+10)=8OREXAM(L1+10)=4THEN670
665 !"FILE LENGTH ERROR FOR DIRECTORY  ",D$¢GOTO730
670 GOSUB950
680 FORJ=0TON2¢L2=D2+16*J¢K=1
690 IFEXAM(L2+K-1)<>ASC(S$(K,K))THEN720¢K=K+1¢IFK<L0+1THEN690
700 FORK=1TO8¢F$(K,K)=CHR$(EXAM(L2+K-1))¢NEXTK
710 !#V,"   ",D$," : ",F$
720 NEXTJ
730 NEXTI
740 !#V¢!#V,"SEARCH FOR ",S$," COMPLETED."¢GOTO620
745 REM *LIST*
750 S=0¢D=0¢F=0¢INPUT"OUTPUT DEVICE # ",V¢IFV<0ORV>7THEN750
760 FORI=0TON1¢L1=D1+16*I¢IFEXAM(L1)<>42THEN870¢D$=""
770 FORK=0TO7¢D$=D$+CHR$(EXAM(L1+K))¢NEXTK
780 IFEXAM(L1+10)=8OREXAM(L1+10)=4THEN790
785 !#V,"FILE LENGTH ERROR FOR DIRECTORY  ",D$¢GOTO870
790 GOSUB950¢F0=0¢S0=0¢P=0
800 FORJ=0TON2¢L2=D2+16*J¢IFEXAM(L2)=32THEN840¢P=P+1¢F0=F0+1
810 !#V,%3I,EXAM(L2+12)," ",
815 FORK=0TO7¢!#V,CHR$(EXAM(L2+K)),¢NEXTK
820 IFP<P8THEN830¢P=0¢!#V
830 S0=S0+256*EXAM(L2+11)+EXAM(L2+10)
840 NEXTJ¢IFP>0THEN!#V
850 !#V,"DISK: ",D$,"   FILES:",%3I,F0,"   BLOCKS: ",%3I,S0¢!#V¢!#V
860 S=S+S0¢F=F+F0¢D=D+1
870 NEXTI¢!#V
880 !#V,"SUMMARY-  DISKS: ",%2I,D,"  FILES:",%4I,F,"  BLOCKS:",%5I,S
890 GOTO420
895 REM *EXIT*
900 !¢!¢!"**** BYE BYE...DON'T FORGET TO RESET MEMSET."
910 END
945 REM *SUBROUTINE*
950 REM GET A DIRECTORY POINTED TO BY L1
955 A2=EXAM(L1+10)¢IFEXAM(L1+12)>127THENU2=129ELSEU2=1
960 P2=256*EXAM(L1+9)+EXAM(L1+8)¢Q=CALL(FNC(A2,1,U2,D2,M2),P2)
965 N2=16*A2-1¢RETURN
```

CLIP TIPS

DOS LIBRARY ROUTINES AND FLAGS

DCOM	2022H	(8226D)	A = number of blocks
			B = command(0=write, 1=read, 2=verify -1=single, -2 =double)
			C = unit number(bit 7=double density)
			DE = starting ram address
			HL = starting disk address

DLOOK	210CH	(8460D)	A=1, HL=address of filename in ram
DWRIT	2105H	(8223D)	(must follow DLOOK)
LIST	2025H	(8229D)	A=drive number, L=device number
DOS	2028H	(8232D)	DOS warm start
RWCHK	202BH	(8235D)	1=read after write, 0=no check
DOSERR	202CH	(8236D)	branched to on hard disk errors
DEN	202FH	(8239D)	contains density flag after DLOOK 0=S 80H=D
START	200AH	(8202D)	bootstrap entry point
AUTOS	2030H	(8240D)	0=turnkey, 1=no turnkey
BUFF	2031H	(8241D)	location of DOS input buffer
OFTEN	2007H	(8199D)	called frequently by DCOM
PAGES	2033H	(8243D)	terminal page length for LIST, 0=no paging
COUT	200DH	(8205D)	character output routine
CIN	2010H	(8208D)	character input routine
TINIT	2013H	(8211D)	terminal initialization routine
CONTC	2016H	(8214D)	control C detect routine

NEED EPROMS BLASTED?

2708 (1Kx8) and 2716 (2kx8) EPROMS will be be custom programmed to your data requirements for only $5 each plus the prom. It's a real and serious proposition. Fr Thomas McGahee, Don Bosco Technical High School, Paterson, NJ 07502 provides the service. We can highly reccommend this meaticulous computer teacher's service. He'll even try for two day turn around, and he means it.

Requirements are: 1) Send the data on CUTS format (1200 baud) tape. 2) Use clean cassette and record near beginning of tape. 3) Send the Eprom you want programed (old ones that have to be erased are OK, just inclosed $1 extra to cover erasing). 4) Include a return address label. 5) Enclose sufficient postage for return of Eprom and tape. 6) Write name of program and its start and end address.

Fr McGahee will also copy from Eprom to Eprom for $5. He'll verify each Eprom against the tape too!

DOS POWER

REVIEW

ADDING POWER TO DOS
By R. Beaver

DOSPOWER, written by Pavel Breder, is a machine routine which provides an extention of North Star DOS capabilities. When connected to DOS 4S, 5.1S, 5.0 DD, or 5.1 QD, it provides on-line enhancements that significantly add to the users ability to view directory information, to copy files, recover deleted files, lock/unlock/rename files, and display/enter hex data. Additionally, DOSPOWER provides for execution of these functions from a BASIC program, and for programmed batch processing of any sequence of DOS+DOSPOWER+BASIC commands.

The table below shows DOSPOWER commands, which are extentions of the DOS LI command. The LIC (copy files) command, for example, allows the user to copy selected programs from one diskette to another. The way Mr. Breder has implimented this command, however, shows both his ingenuity and awareness of what makes life easy for the user. The disk directory is displayed (formatted as specified by the user in 1,2,3...columns) with a number beside each file. A series of file numbers (optionally a range of file numbers) may then be specified, and the copy process begins. Each file is appended to the destination diskette after checking to see if the destination diskette is full or if the file already exists. Files that already exist are not overwritten, and files that cannot be transfered because of their length are reported as LONG. The LIC command has several other options which combine to make it the most powerful copying command available to North Star users. Several of the other commands show comparable power and utility.

DOSPOWER is 4K, virtually self-configuring, and comes with a well written manual and diskette with user relocatable programs. The software vendor for this useful program is John Dvorac, 704 Solano Ave, Albany CA. 94706. Priced about $60, the program is highly recommended by many users. (DOS 5.2 versions now supplied-ed)

```
                       DOS-POWER COMMAND SUMMARY

LIC(#N) (DRIVE#X <DRIVE#Y>) (DENSITY)...COPY FILES WITH NO STOP
LIQ(#N) (DRIVE#X <DRIVE#Y>) (DENSITY)...COPY FILES WITH REQUEST
LI↑     (DRIVE#X <DRIVE#Y>)          ...COPY DISK
LIX(#N) (DRIVE#X <TYPE>)             ...LIST DIRECTORY
LIS(#N) (DRIVE#X <TYPE>)             ...LIST DIRECTORY & STATUS
LIF     (DRIVE#X)                    ...FIND DELETED DIR NAME
LIR     OLDNAME(,DRIVE) NEWNAME      ...RENAME FILE
LIL     FILENAME(,DRIVE)             ...LOCK FILE
LIU     FILENAME(,DRIVE)             ...UNLOCK FILE
LIM(#N) ADDRESS                      ...MONITOR DISP/SUB HEX
LIZ     XXXX YYYY ZZ                 ...FILL MEMORY (8080)
LIT     XXXX YYYY ZZZZ               ...MOVE MEMORY (8080)
LID(#N) DECIMAL NUMBER               ...CONVERT TO HEX
LIH(#N) HEX                          ...CONVERT TO DECIMAL
LIH(#N) HEX (+ OR -) HEX             ...ADD/SUBTRACT HEX
LIWn                                 ...SET COLUMNS
LIOn                                 ...SET OPTIONS
LI?(#N)                              ...SYSTEM INFORMATION
LII                                  ...DISCONNECT

SPECIALS  LIE/LIA/LIP/LIG
LIE...ENTER TEXT   LIA...ADD TEXT  LIP...PRINT TEXT  LIG...EXEC:
 (#N) =INT BUFFER   (#N)-XXXX=EXT BUFFER   (#N)$=AUTO BUFFER

Z-80 ENHANCEMENTS TO LIM COMMAND.
LIM(#N)I XXXX  ...DUMP HEX
LIM(#N)$I XXXX ...DUMP ASCII
LIM(#N)$ XXXX  ...DISP/SUBSTITUTE ASCII
LIMF XXXX YYYY ZZ ...FILL MEMORY
LIMM XXXX YYYY ZZZZ..MOVE MEMORY
LIMC XXXX YYYY ZZZZ..COMPARE MEMORY
```

REVIEW

COMPLETE EDITOR

A LOOK AT A NEW EDITOR

BY JIM LIND

We have all read the ads telling us that a new software package is the best available, but how can we be sure we aren't paying for what might as well be a blank disc? User comments should be unequaled in evaluating the worth of a new product - unfortunately we all have a tendancy to tell about the new gidget that does what we think is a good job but are reluctant to tell about the imperfect purchase. At this point you may think you are about to read of my latest "imperfect purchase". Quite the contrary, this is about a very worthwhile Utility Set for those who are using North Star Basic.

Software Systems of Altadena, California has produced a Utility Set for North Star Basic called N*BUS. There are four Utilities produced:

1. Global Line Editor.
2. Program which will pack a Basic Program by eliminating unnecessary spaces and remarks (REMs).
3. Utility that lists simple and dimensioned variables, functions, and subroutines functions, and subroutines, cross referenced to program line numbers.
4. Utility to change the name of a file.

The Editor is the main attraction. For those of us who have used the North Star Monitor to locate all occurrences of a particular subroutine or function in a long program because a change is necessary, this Editor is the answer. In addition to locating the first occurrence or all occurrences of a string, the Editor is also able to change the first occurrence or all occurrences of string 1 to string 2.

Some of the other features include the ability to (a) Copy one or more lines without changing the original line(s), (b) Move one or more lines from one location to another, (c) View the program

CONTINUED NEXT PAGE

EDITOR continued from page 18

one page at a time to better view a particular region of a program and (d) append to current, insert characters, delete, etc., the same as the North Star Editor but in slightly different form.

The Editor is 14 blocks long and can be positioned after Basic (in Tandem) or there is a version which can be located outside the normal space allocated for Basic (Romable). The Tandem version reduces the space available for Basic programs but this version of the Editor is loaded at the same time Basic is loaded. The Romable version can be located outside the normal Basic memory area but it must be loaded with a separate "GO" command. The diskette provided includes a Basic program designed to personalize the Editor to match the user's system. The user is also provided a good 64 page documentation package.

Before this sounds too much like a commercial, let's examine some of the limitations of N*BUS. According to the documentation, the Editor will only work with North Star Basic, 4.0 or later. Unfortunately, the other three Utilities (Pack Program, List Variables, and Rename File) will not function with DOS 4.0 since they depend "on the argument passing capability inherent in the DOS "GO" command starting with release 5". These three Utilities also require a standard origin DOS.

Initially the editing process is slower because it takes time to learn the new commands but before long they are "second nature". If you are a "creature of habit" and you intend to use both the North Star Editor and the N*BUS (which you can do) there is one word of caution. N*BUS seeks a line when the user simply types the line number; when using Basic, typing only the line number will delete that line. If you use both editors and suddenly find there are some missing lines, this is probably what happened.

This list of N*BUS Editor commands will give you an insight to its use.

>	IS THE EDITOR PROMPT
A	APPEND TO CURRENT LINE (Same as CTL-G)
B	POSITION LINE POINTER TO BOTTOM OF PROGRAM
BASIC	RETURN CONTROL TO BASIC
BYE	RETURN CONTROL TO DOS
C ;F1;F2;	CHANGE FIRST F1 TO F2 ON CURRENT LINE
C,30 ;F1;F2;	AFTER COLUMN 30 ON CURRENT LINE CHANGE FIRST F1 TO F2
C ;F1=0\;; 3	ELIMINATE FIRST F1=0 ON CURRENT & NEXT 2 LINES
C,20	AFTER COLUMN 20 ELIMINATE FIRST F1=0 ON CURRENT & NEXT TWO LINES. DEFAULT STRING CONTAINS ";F1=0\;; 3" FROM PREVIOUS EXAMPLE.
CL ;F1:F2:	CHANCE ALL OCCURRENCES OF F1 TO F2 ON CURRENT LINE.
COPY	
1000 DATA 1,2,3	EDITOR DISPLAYS CURRENT LINE.
N>2000	COPY line 1000 TO 2000
1010 DATA 4,5,6	EDITOR DISPLAYS NEXT LINE
N>2010	COPY LINE 1010 TO 2010
1020 DATA 7,8	EDITOR DISPLAYS NEXT LINE
N>CR	CARRIAGE RETURN ENDS COPY
*	CAN BE USED TO REPRESENT MAXIMUM NUMBER
D 25	DELETES CURRENT LINE AND NEXT 24 LINES
D*	DELETES CURRENT LINE AND NEXT 65535 LINES (up to EOF)
D	DELETES CURRENT LINE
DEV 1	LISTINGS FROM EDITOR TO DEVICE #1
DEV 0	LISTINGS FROM EDITOR TO DEVICE #0
EC 25,30	ERASE COLUMN 25 THRU 30
EC 25,25	ERASE COLUMN 25
EC 25,*	ERASE COLUMN 25 TO END OF LINE
GO 1120	POSITIONS LINE POINTER TO LINE 1120 (GO is optional)
I 18	PRINTS CURRENT LINE THRU COLUMN 18 THEN USER INSERTS DATA. CARRIAGE RETURN TERMINATES INSERT -USER TYPES A SINGLE CHARACTER OR CARRIAGE RETURN - EDITOR PRINTS CURRENT LINE TO SINGLE CHARACTER, READY FOR INSERT, OR TERMINATES (CR)
L NOENDMARK	LOCATE NEXT LINE (after current line) THAT CONTAINS NOENDMARK.
L,30 FOR	LOCATE NEXT LINE " WITH "FOR" (start search at column 30).
L,30	SAME AS ABOVE - AGAIN, BY DEFAULT STRING CONTAINS "FOR".
L	LOCATE NEXT LINE WITH "FOR" (default) START SEARCH AT COL. 30.
LA FOR	IF STARTED AT THE TOP, LOCATES ALL FOR-NEXT LOOPS.
MOVE	
3670 !#1 "WHERE"	EDITOR LISTS CURRENT LINE.
N>4120	DELETES 3670 AND MOVE TO 4120
3680 RETURN	EDITOR LISTS NEXT LINE
N>CR	CARRIAGE RETURN ENDS MOVE
N 20 or +20	ADVANCES FROM CURRENT LINE BY 20.
N-15 or -15	BACKWARD MOVEMENT OF 15 FROM CURRENT LINE.
P 15	PRINT 15 LINES STARTING AT CURRENT LINE.
P *	PRINT FROM CURRENT LINE TO END OF PROGRAM.
P	PRINT CURRENT LINE.
PROMPT	PROMPT FLAG IS TOGGLED "ON" IF "OFF" OR "OFF" IF "ON".
S	A SCALE OF COLUMNS IS PRINTED.
T	PROGRAM LINE POINTER POSITIONED TO THE TOP OF THE PROGRAM.
TAB 10	EDITOR'S TABULATOR STOP VALUE SET 10.
TCHAR #	"WILD CARD" CHARACTER BECOMES "#"
V	PRINTS ONE VIDEO PAGE - WILL QUICKLY SCROLL, PAGE AT A TIME.
W	OPENS A "WINDOW" AROUND THE CURRENT LINE - TO VIEW BEFORE AND AFTER CURRENT LINE.
WN	NEXT WINDOW
WP	PREVIOUS WINDOW
GO BpRT(,unit #)	(PROGRAM(,unit #)) (OPTION 1, 2, OR3) (#dev)
GO BPAK(,unit #)	(OLDPROG(,unit #) (NEWPROG(unit #))
GO RE(,unit #)	OLDFILE(,unit #) NEWFILE

54 4F20 43 4F 4E 54 49 4E 55 45

Hard Disk Drive!

by Joe Maguire, P.O. Box 3742 DT, Anchorage AK, 99510

This report on Northstar's new HD-18 hard disk storage system consists of some good news and some bad news. First the good news: My overall opinion of the system is good with excellent software support. Now the bad news: The system will not work with an 8080 CPU computer.

Northstar lists the following requirements for the HD-18:

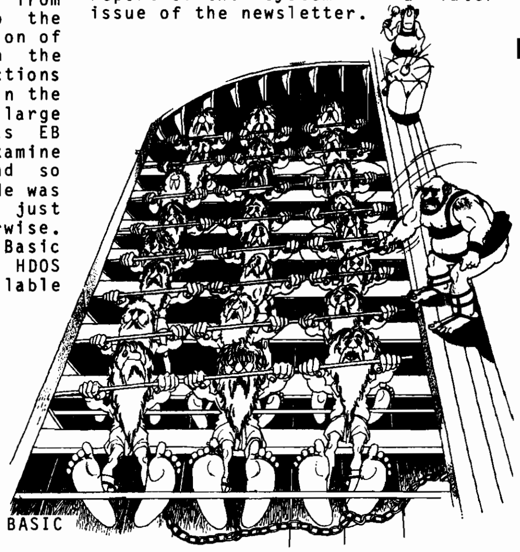
1. A Z80 CPU. (the HDOS is written in Z80 code)
2. Processor speed of 4 mHz. (required for data transfer rate)
3. 48K of contiguous memory with 56K recommended.
4. Parallel port interface.
5. Double density minidisk(s). (required for coldstart and backup)

As can be seen from requirements 1 and 2 this rules out the Sol or any 8080 CPU based system. The HDOS (hard disk operating system) occupies the first 18k of memory beginning from address 100 hex. The HDOS contains all the necessary code to permit data transfer operations from memory to the hard disk or to the minidisks of to any combination of the three. Also included in the HDOS are all the Monitor functions which are in a separate file in the non-hard disk system plus a large number of new commands such as EB (examine byte), EM (examine memory), EP (examine port) and so on. It is obvious why Z80 code was used to write the HDOS, it just wouldn't fit in 18K otherwise. (Application programs such as Basic can overlay part of the HDOS incresing the remaining available memory.)

I got the distinct impression from the HD manual that NS was trying to push the use of their Horizon computer with the system but I also feel that the software could be rewritten to accommodate the many 8080 based users of Northstar products. P Perhaps after the initial rush of orders is over NS will get around +o providing a version of the HDOS for them.

In addition to my Sol, I have a Horizon so I was able to make use of the system. The drive is a Century Data Marksman which is contained in a sealed unit. The capacity of 18 megabytes allowed me to transfer my 50 minidisks of programs an data over to the HD with still some more room to go. This new found convenience is not without its hazards however. A disk crash can wipe out years of work! NS -as considered this possibility and has included a well thought out series of backup and recovery programs. It takes only a few minutes a day to accomplish using the minidisks which are expected to be available with the system. This is in marked contrast to the 30 to-40 minute tape backup I observed with another system.

I will have a more detailed report of the system in a later issue of the newsletter.



CLIP TIPS

CAT COMMAND FROM WITHIN BASIC PROGRAM

N* Basic doesn't permit listing the contents of a disc from a running basic program - very useful if you want to use the disc directory as menu to select parts of a longer segmented program. This short subroutine calls the DOS LI command. It can be located anywhere in protected memory. Use MEMSET.

SOURCE	HEX	DEC	REMARKS
MOV A,E	7B	123	drive unit
MOV L,D	6A	106	output dev
MVI H,0	26	38	H = 0
	00	0	
JMP LI	C3	195	jmp to DOS
	25	37	list
	20	32	routine

SUB ROUTINE from Basic:

```
10 M=XXXX
     REM XXXX=protected memory
20 FILL M,123
     FILL M+1,106  FILL M+2,38
     FILL M+3,00
30 FILL M+4,195  FILL M+5,37
     FILL M+6,32
40 D=   REM disc drive unit
50 L=   REM output device for CAT
60 X=L*256+D
70 C=CALL(M,X)
```

Note that D and L must be defined in lines 40 and 50 before use. Also, note that the routine assumes DOS origin at 2000 HEX. For DOS at other locations, change 32 in line 30 to reflect the new page address for DOS.

CLIP TIPS

FIX FOR DOS 5.1s LISTING DISPLAY

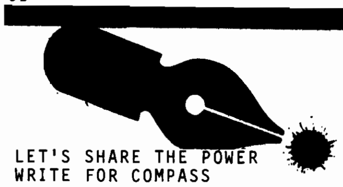
Listing a disc directory with 5.1 DOS (single density) presents a minor but annoying eye strain producing bug. The PRESS RETURN TO CONTINUE message is followed on the same line by the next entry without the grace of even a space. North Star neglected the LF/CR in the code. Its been corrected in the rewitten 5.2 single density DOS. Here's a quick and dirty fix if you are running 5.1.

The ASCII code for PRESS RETURN TO CONTINUE begins at 219C and runs to 21B3. Use the monitor to enter 20 Hex (spaces) starting at 21A9 and you will eliminate the word CONTINUE. This gives you room at 21B2 to enter carriage return (OD HEX at 21B2) and line feed (OA HEX at 21B3).

You may prefer to skip the OA Hex and just enter a OD Hex at 21B2 to save the screen display space. The short PRESS RETURN line and the fact thatthe cursor jumps back on itself makes the break between screen takes very visable.

Here's the code starting at address 21A9

```
21A9 :20 20 20 20 20 20  20  20  20
OD
```



LET'S SHARE THE POWER
WRITE FOR COMPASS

In computers, knowledge is power just as it is in the real world. You can help by sending information to COMPASS. We'll help by publishing the material you wish to share with others.

We'll try to publish everything as space permits. The only delimiter is made in the interest of efficiency and accuracy...PLEASE SEND YOUR CONTRIBUTION ON MAGNETIC MEDIA. We'll guarantee to copy and return your disc within one working week.

The reasons for requesting magnetic media are twofold. It elimiates accidental errors in retyping of programs and listings, errors that could be disasterous to the other users. It holds down the drain on your association dues by elimating the cost of paying a typist to retype copy for printing reproduction.

At present we can handle material in the following formats:

Electric Pencil I or II
WordStar
Assembler text editor
(line length not more than 35 characters).
North Star Basic files

DISCS CAN BE FORMATED IN SINGLE OR DOUBLE DENSITY

COMMUNICATION NETWORKS

MODEM COMMUNICATIONS CONTACTS

This listing of micro computer bulletin board systems across the nation is made available to INSUA members by the Peripheral People, Box 524, Mercer Island, WA 98040. The CBBS systems operate under CPM, ABBS under Apple and FORUM 80 are TRS-80, but all may be accessed with the Telestar (North Star Dos) modem and file management program soon to be available from INUSA's software library.

NorthStar LINK

HOT LINE

North Star Computer's HOT LINE for technical assistance is available 24 hours a day through their Proxima modem data link. This user service is in addition to the normal business hour voice communications HOT LINE system on (415) 524 9202

To access the data link you will need 110 or 300 baud modem. The data link phone number is (415) 527 0400.

The Proxima system operates under Ward Christianson's Chicago CBBS system and uses the normal CBBS protocal, i.e. you issue 2 or more carriage returns to establish your communications baud rate on initial connection, use the CP/M control keys for terminal management, etc.

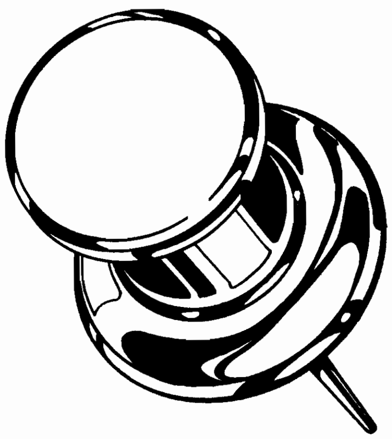
The system runs on a 32K Horizon with a Micro Magic Modem. It is menu driven and the first time user can get a detailed printout of all commands by sending a "?" when initially queried as to desired functions.

The system covers three broad operating areas:
Enter user messages
Recall message summaries
Recall full messages

Queries and messages are not limited to requests for technical help regarding North Star product implimentation. They also cover the full range of electronic mail provided by other micro computer network links. They deal with buying and selling of equipment, trading advice, and notices on software use and modification.

CITY	SERVICE	AREA CODE	TELEPHONE
BOUNDBROOK NJ	SJ ELECT MAIL	(201)	457-0893
UNION NJ	FORUM 80	(201)	688-7117
POMPTON PLNS NJ	ABBS	(201)	835-7228
WYCKOFF NJ	ABBS	(201)	891-7441
PISCATAWAY NJ	ABBS	(201)	968-1074
WASHINGTON DC	PROGRAM STORE	(202)	337-4694
BIRMINGHAM AL	ABBS	(205)	945-1489
SEATTLE WA	ABBS	(206)	244-5438
ELMA WA	ABBS	(206)	482-5134
SEATTLE WA	ABBS	(206)	524 0203
SEATTLE WA	FORUM 80 (PER. PEOPLE)	(206)	723-DATA
FRESNO CA	LIMITED ACCESS.	(209)	638-6392
STATEN IS NY	ABBS	(212)	448-6576
LOS ANGELES CA	ABBS	(213)	276-4276
TORRANCE CA	ABBS	(213)	316-5706
TORRENCE CA	PROG SALES	(213)	329-3715
CANOGA PK CA	ABBS	(213)	340-0135
LOS ANGELES CA	ABBS	(213)	349-5728
LOS ANGELES CA	ABBS	(213)	360-6332
SANTA MONICA CA	ABBS	(213)	394-1505
BRENTWOOD CA	ABBS	(213)	395-1592
SIGNAL HL CA	ABBS	(213)	424-3506
LONG BCH CA	NA	(213)	428-4718
PAC. PALISADES CA	ABBS	(213)	459-3177
HAWTHORNE CA	ABBS	(213)	675-8803
PASADENA CA	CBBS	(213)	795-3788
PASADENA CA	ABBS	(213)	799-1632
SO. PASADENA CA	ABBS	(213)	799-6514
LOS ANGELES CA	ABBS	(213)	826-0325
SANA MONICA CA	ABBS	(213)	828-3400
SAN FNANDO CA	NA	(213)	843-5390
DALLAS TX	FORUM 80	(214)	288-4859
DALLAS TX	ABBS	(214)	634-2668
DALLAS TX	NA	(214)	634-2775
DALLAS TX	CBBS	(214)	641-8759
AKRON OH	CBBS	(216)	745-7855
FT LAUDERDALE FL	ABBS	(305)	566-0805
WEST PALM BCH FL	ABBS	(305)	689-3234
FT LAUDERDALE FL	FORUM 80	(305)	772-4444
MIAMI FL	ABBS	(305)	821-7401
HOLLYWOOD FL	ABBS	(305)	989-9647
ARLINGTON HTS IL	ABBS	(312)	255-6489
CHICAGO IL	FORUM 80	(312)	269-8083
CHICAGO IL	ABBS	(312)	337-6631
NAPERVILLE IL	ABBS	(312)	420-7995
CHICAGO IL	CBBS	(312)	528-7141
CHICAGO IL	NA	(312)	622-9609
CHICAGO IL	CMMS	(312)	767-0202
CHICAGO IL	CBBS	(312)	782-9751
DOWNERS GROVE IL	ABBS	(312)	964-7768
DETROIT MI	?	(313)	288-0335
YPSILANTI MI	PET BBS	(313)	484-0732
DETROIT MI	ABBS	(313)	569-2063
ST LOUIS MO	FORUM 80	(314)	838-7784
WICHITA KS	FORUM 80	(316)	746-2078
IOWA CTY IA	ABBS	(319)	353-6528
DUBUQUE IA	ABBS	(319)	557-9618
ATLANTA GA	CBBS	(404)	394-4220
AUGUSTA GA	ABBS	(404)	733-3461
ATLANTA GA	NORTHSTAR	(404)	939-1520
ATLANTA GA	ABBS	(404)	939-8429
SANTA CLARA CA	CBBS	(408)	241-1956
SAN FRAN CA	FORUM 80	(415)	348-2139
PALO ALTO CA	ABBS	(415)	493-7691
SAN FRANCISCO CA	ABBS	(415)	561-0705
FREMONT CA	ABBS	(415)	792-8406
LOS ALTOS CA	ABBS	(415)	948-1474
SPRINGFLD MO	?	(417)	862-7852
LOUISVILLE KY	ABBS	(502)	245-8288
PORTLAND OR	CBBS	(503)	646-5510
SAN ANTONIO TX	ABBS	(512)	657-0779
NEW YORK NY	CBBS	(516)	938-9043
PHOENIX AZ	ABBS	(602)	866-0258
PHOENIX AZ	ABBS	(602)	955-1486
PHOENIX AZ	ABBS	(602)	957-4428
PHOENIX AZ	ABBS	(602)	957-9282
VANCOUVER BC	CBBS	(604)	687-2640
MARLTON NJ	ABBS	(609)	983-5970
MINNEAPOLIS MN	ABBS	(612)	929-8966
COLUMBUS OH	CBBS	(614)	272-2759

CONTINUED NEXT PAGE



SOFTWARE SERVICES

SOFTWARE LIBRARY

NORTH STAR RELEASE 5.2 SOFTWARE is being offered to INSUA members through the software library. To obtain your copy, send $10. check or money order and the following information to the INSUA MAILBOX at P O BOX 1318, Antioch CA 94509.

Member Number (see address label)
Single or Double Density Version?

These requests will recieve high priority response. A disk and 16 page documentaton update will be sent to your member address. Send a self sticking address label if another address is desired.

TELSTAR communication programs for modems have been donated to the INSUA library by the author (and INSUA member) Leonard Garcia. This is an impressive and important collection of communication programs for use with the North Star operating system. The INSUA library will be making an offering of these programs to members in the immediate future.

COMMUNICATION NETWORKS

NASHVILLE TN	CBBS	(615)	254-9193
BOSTON MA	ABBS	(617)	354-4682
WELLESLEY MA	FORUM 80	(617)	431-1699
DUNSTABLE MA	FORUM 80	(617)	649-7097
CAMBRIDGE MA	CBBS	(617)	864-3819
MAYNARD MA	CBBS	(617)	897-0346
BOSTON MA	CBBS	(617)	963-8310
LAS VEGAS NV	FORUM 80	(702)	873-9491
WASHINTN DC	AMRAD	(703)	281-2125
FLS CHURCH VA	CBBS	(703)	734-1387
FAIRFAX VA	GENEALOGY	(703)	978-7561
HOUSTON TX	ABBS	(713)	654-0759
COLLEGE STA. TX	ABBS	(713)	693-8080
HOUSTON TX	ABBS	(713)	977-7019
SAN DIEGO CA	ABBS	(714)	449-5689
LEMON GROVE CA	ABBS	(714)	463-0461
FULLERTON CA	COMM 80 (OCTUG)	(714)	526-3687
ORANGE CNTY CA	FORUM 80	(714)	537-7913
GARDEN GROVE CA	DATA EXCHANGE	(714)	537-7913
SAN DIEGO CA	NA	(714)	565-0961
SAN DIEGO CA	ABBS	(714)	582-9557
SANTE FE SPGS CA	ABBS	(714)	739-0711
IRVINE CA	ABBS	(714)	751-1422
ANAHEIM CA	ABBS	(714)	772-8868
WESTMNSTR CA	ABBS	(714)	898-1984
HUNTIN'TN BCH CA	ABBS	(714)	962-7979
LOGAN UT	ABBS	(801)	753-6800
XXXXX VT	SJBBS	(802)	748-9089
AUGUSTA GA	FORUM 80	(803)	270-5392
COLUMBIA SC	NORTHSTAR	(803)	771-0922
VENTURA CA	FORUM 80	(805)	484-9904
TAMPA FL	FORUM 80	(813)	223-7688
KANSAS CTY MO	FORUM 80	(816)	861-7040
KANSAS CTY MO	COMMODITIES	(816)	931-3135
WICHITA FLS TX	FORUM 80	(817)	855-3914
FT WORTH TX	FORUM 80	(817)	923-0009
MEMPHIS TN	FORUM 80	(901)	276-8196
MEMPHIS TN	HOBBIEST 80	(901)	362-2222
MEMPHIS TN	ABBS	(901)	761-4743
DESTIN FL	ABBS	(904)	243-1257
FT WALTON BCH FL	NA	(904)	243-8565
KETCHIKAN AK	CBBS	(907)	225-6789
OLATHE KS	ENGINEER 80	(913)	764-1520
OLATHE KS	AVIONICS	(913)	782-5115
EL PASO TX	CBBS	(915)	584-5393

MARKETPLACE

THESE BUY, SELL, OR SWAP ADS ARE FREE TO MEMBERS OF INSUA. YOU MAY FIND THIS SECTION A CONVENIENT WAY TO TURN YOU EXTRA OR UNUSED EQUIPMENT INTO CASH. PLEASE NOTE THAT INSUA MAY HAVE TO LIMIT THE AMOUNT OF "MARKET PLACE" SPACE AVAILABLE AND ADS WILL BE PRINTED ON A FIRST COME FIRST SERVED BASIS. NO RESPONSIBLITY CAN BE TAKEN FOR THE CLAIMS OF BUYERS OR SELLERS. ALL ADS MUST BE "NON-COMMERCIAL". BUSINESS VENDORS OF SOFTWARE OR HARDWARE MAY ONLY ADVERTISE VIA DISPLAY ADVERTISING. PLEASE WRITE FOR ADVERTISING RATE SCHEDULE.

NORTH STAR 16K Memory Board mint condition with parity option included but not installed $210. Call 215- 4598895 Don Butterworth, RD 4, Box 11, Glen Mills, PA.

WANTED INFORMATION on structural or civil engineering software compatable with SOL-20, North Star or CPM. LA Duerksen, 1804 Miles Ave. Muscatine, IA 52761.

NORTH STAR Single Density and controller $325 with software and 20 disks. Bill Giffen, 1713 Hearthstone, Plano, TX 75023. 214- 8670114.

16 K DYNAMIC MEMORY (S100) PTC with all its problems. Works but...AS IS $50. Steiner, 2519 Greenwich, San Francisco, CA 94123. 415-3460625.

WANTED SOFTWARE, games, editors, languages, business packages, anything. Please send description, requirements and price or trade with software i currently have. Victor Chinn, 1618 Milva St., Berkeley, CA 94709.

NORTH STAR S100 disc controller board $200. Cromemco dazzler graphics board $200. Richard Katz, 701 Marichopa Dr., Prescott. AZ 86301.

WANTED NORTH STAR disc controller and drive unit to extend my ability to provide software utilities as I now do for SOL users. Please advise lowest asking price in first letter. Fr. Thomas McGahee, Don Bosco Technical High School, Paterson, NJ 07502.

16 K STATIC MEMORY board (S100) Type Z16 by TDL. Perfect. Bank address to any 4 k. Battery back up possible. Socketed. Write protect switches for any 4k or all memory. Extra standby replacement memory chip (never used). $160. Steiner, 2519 Greenwich St., San Francisco, CA 94123. 415-3460625.

24k GODBOUT STATC RAM board. Like new with all support chips but no ram chips. Uses 5257's or 4044's. Two 8K blocks and two 4K blocks addressed anywhere. Socketed. $120 Steiner, 2519 Greenwich, San Francisco, CA 94123. 415 346- 0625.

NORTHSTAR HORIZON 1, 32K Ram double density and 1 parallel port, 6 edge connectors installed, power supply for second drive installed. Original shipping container. $1995 or make offer. Jim Larson, 10501 Landmark Ct., Boise, Idaho 83704 Evenings phone 208- 377 2041.

we've been fighting for you

&

ALL NorthStar systems users:

Your association offers a continuing source of information for novice as well as advanced users of North Star disc systems.

Your association publishes a quarterly newsletter of practical techniques and application programs and fixes designed specificly for the North Star system expanding the options in the use of your computer.

Your association is a non profit, independent, users representative providing feedback to North Star Computers Inc.

Your association provides the liason link between local users groups to promote the exchange of information useful to all North Star systems users.

Your association maintains a software library of programs and utilities for members using North Star Basic or North Star Dos.

Attention

LET'S ALL HANG TOGETHER !!

SEND YOUR $15 ANNUAL DUES NOW FOR ALL MEMBERSHIP PRIVILEGES OF BEING PART OF THE INTERNATIONAL NORTHSTAR USER'S ASSOCIATION

(See special OFFER on page two)

Name______________________________
Street______________________________
City______________________________
State________________Zip________
Phone______________________________
Local Club__________________________
Street______________________________
City______________________________
State________________Zip________

SYSTEM SURVEY

Computer	( )Horizon		( )Other
Memory	( ),32K		( ).32K
Disk	( )one	( )two	( )three
Disk	( )5"	( )8"	( )hard
Density	( )single	( )dbl	( )quad
Use	( )Business	( )Hobby	( )Both

MAIL TODAY TO:
International North Star Users Association, PO Box 1318, Antioch, CA 94509

NORTHSTAR is a registered trade mark of North Star Computers, Inc.

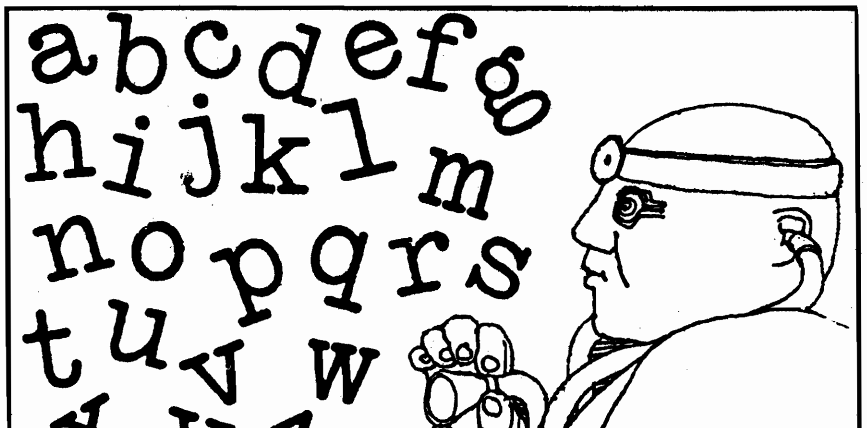
POSTMASTER: ADDRESS CORRECTION REQUESTED
Please enter key number on form 3579

good news for you!

N

INTERNATIONAL NORTH STAR USERS ASSOCIATION
PUBLISHERS OF THE COMPASS NEWSLETTER
POST OFFICE BOX 1318, ANTIOCH, CA. 94509

BULK RATE
U.S. POSTAGE
PAID
ANTIOCH, CA
PERMIT NO. 265



HELP WANTED

SOFTWARE LIBRARIAN NEEDED.....

INSUA and INSUA's members need a software librarian to plan and operate the association's software library exchange. The position offers a large amount of satisfaction and a very small amount of compensation.

The librarian will have full discretion in establishing policy to ensure the widest distribution of selected quality programs among North Star users.

If you want to help and receive the satisfaction spreading good programs among INSUA's worldwide membership, please contact Dr. William Banaghan, Chairman INSUA board of directors, 1261 Lancashire Dr, Concord, CA. 94518 (415) 689-4337

IN THIS ISSUE OF The Compass:

VOLUME I, NUMBER 2

The COMPASS

IN THIS ISSUE:

NS BASIC COMPILER!

BY Bob Stek
19 Mayfield Rd
Regina, Seshatchewan S4V OB7
Canada

Finally! A true compiler for NorthStar BASIC! Allen and David Ashley have just made available COMSTAR, a compiler system which will take a type 2 BASIC file and turn it into a type 1 machine language GO file (and provide you with the assembly language source besides). It really works!

This is just a short review; I have only had COMSTAR for a few weeks and have not yet had time to explore all aspects of the system. Indeed, I am one of several "guinea pigs" for the pre-release version which was initially offered to purchasers of Ashley's Compiler Development System (which compiled only an integer subset of N* BASIC) before being offered for general sale.

First, the facts. Cost: $400. Ouch! But writing a compiler is no small investment of time. What you get:

COMPILE	- the compiler
MAKRO	- disk based macro assembler
EDIT	- text editor
PROCOM	- console command processor
CLINK/LINKC	- linking loaders plus library modules

You may order COMSTAR to run at either E00 or 2D00, though it will compile programs to run at both locations. COMSTAR is available only for double-density controller boards (you need it, as the system uses large amounts of disk space).

FUNCTIONALITY : Excellent. COMSTAR is virtually identical in design to the NorthStar interpreter. There are a few restrictions, however. Variables

CONTINUED ON PAGE 2

NS CP/M

NORTH STAR IS ON THE CP/M BANWAGON

BY Clyde Steiner

North Star has finally adopted CP/M as an "officially" recognized operating system.

The company is selling their ready to boot up Horizon version to users.

Unlike other manufacturers who offer their hardware compatible CP/M at a substantial discount, North Star is offering theirs at a substantial increase in price over the de facto micro industry standard from Lifeboat Associates. Is it worth the extra bucks?

Its the new release 2.2 version of CP/M but like other CP/M's from hardware maufacturers, North Star's version of the "unirversal" operating system is not compatible with previous Lifeboat/North Star CP/M's. They have added their own extra jump to the standard CP/M User I/O jump table so you can't just drop in your own I/O from and earlier CP/M and go. You have to rewrite.

Ofcourse if you own a Horizon the advantage of their CP/M is that it is ready off the shelf. Just boot up and run. They have also provided a neat combination MOVCPM and SYSGEN utility program that will customize the system size to your

CONTINUED ON PAGE 4

NS BASIC COMPILER

CONTINUED FROM PAGE 1

are not allowed in DIM statements; e.g., DIM A(100) is okay, but N=100 DIM A(N) is not. Disk file numbers may not variables or expressions; they must be a decimal constant within the range 0-7. This applies to READ, WRITE, OPEN, CLOSE, TYP, and FILEPTR. COMSTAR requires that a single NEXT be used for each FOR. These restrictions are relatively minor and should not cause substantial changes to be made in most programs.

I constucted several benchmark programs and compared their execution speeds under interpreted BASIC, interpreted BASIC with the floating point board, compiled BASIC, and compiled BASIC with the floating point board. The results:

EASE OF USE: Excellent Ashley offers PROCOM to automate the compiling sequence so as to make minimal demands upon the user. A typical sequence would look like:

```
  +GO PROCOM - execute the batch
               processor
    COMMAND FILE?
    COMPCOM  - a  file of compiler
          commands created by EDIT
    PARAMETER LIST?
    YRPROG,2 - the program to be
                          compiled
    The last program invoked by
    COMPCOM is CLINK which prompts:
    SAVE  xx  BLOCKS  AT  (3400)
    PROGRAM RUNS AT (E00)
3400 & E00 are dependent upon the
COMSTAR address
```

Once you go through the familiar procedure of CR, TY, and SF, you may then GO filename.

SUMMARY: While the price of COMSTAR may seem high for the average hobbyist, for systems houses, dealers, and software entrepreneurs, it can be easily justified. In contrast to Microsoft's policy, Ashley asks for no royalties on sales of compiled programs; all he asks is that a notice be embedded in the program that it was compiled with COMSTAR.

The advantages of being able to develop programs interactively with NorthStar BASIC and then compile them with COMSTAR for speed and source code protection are obvious. COMSTAR is an excellent product and is highly recommended.

	Int	FPInt	Comp	FPComp	Description
PRIME1	5:28	3:31	2:52	0:50	Inefficient prime # gen.
PRIME2	1:21	0:42	0:45	0:11	Better prime # generator
MATINV	1:15	1:02	0:40	0:23	15x15 matrix inversion
SORT-W	1:45	1:37	0:28	0:22	Wood sort - 500 elements
SORT-S	2:24	2:16	0:47	0:37	Shell sort - 500 elements
SORT-B	3:02	2:57	1:18	1:08	Bubble sort - 200 elements
STRSORT	2:05	1:59	0:32	0:23	String sort - 26 13-char.

For comparison, PRIME1 under the Microsoft MBASIC interpreter took 3:47 while it compiled counterpart took 1:02. These results should be taken with a grain of salt since COMSTAR's performance is variable and is very dependent upon what operations are being performed by the program. It really shines with sorting programs; the compiled versin of SORT-W ran nearly 4 times faster than the interpreted version. (Should COMSTAR therefore be described as a compiler of sorts?) Transcendental functions, however, only run about 30% faster than their interpreted counterparts, according to Ashley.

I found that there appeared to be some bugs in file handling. Initializing files under COMSTAR have taken longer or given error messages where the interpreted version ran just fine. Ashley has already tracked down some of these problems and has told me that the file handling routines have been corrected.

DOCUMENTATIONN: Fair to good. For my tastes, documentation for all of Ashley's products has tended to be sparse, or more oriented to the advanced programmer who already knows the general concepts and just needs the details of this particular implememtation. Documentation for COMSTAR is no exception. The sections on EDIT, MAKRO, and CLINK are practically with his previous documentation for his Program Development System. The section on the COMPILER program itself is just 7 pages; it provides the information you need to use it, but is not tutorial. Source code is not provided.

As mentioned, COMSTAR requires double-density and preferably two drives; quad density is even better.

ERROR HANDLING : Excellent. Errors detected during compilation are displayed on the console as well as in the disk source code file; compilation or assembly errors will abort the generation sequence. Execution errors are give the same error codes as those found in the NorthStar BASIC manual; these error numbers and the line numbers in which they occurred are displayed by the run-time error routine.

SUPPORT : Excellent. Ashley prefers to give support by telephone, but he will respond to letters as well. He has always been extremely helpful whenever I have called, and he seems to be available at any reasonable hour of the day or evening. He has constantly upgraded his previous products whenever NorthStar has released a new version of DOS or BASIC. I am now in the process of returning my COMSTAR disk for an update. Ashley mentioned that he has made over a dozen fixes and/or improvements in just the first month it has been available (one enhancement is the addition of the ability to generate in-line code, especially useful for loops using integers for counters instead of floating point numbers). Ashley is justifiably proud of his products and appears to be more than willing to improve them on a practically continuous basis.

LATE NEWS

INSUA SOFTWARE LIBRARY

The INSUA software library is off to a great start with a selection for members of the classics:

DOS and BASIC, version 5.2, single density, order # 1001, price - $10

DOS and BASIC, version 5.2, double density, order # 1002, price - $10

MODEM COMMUNICATIONS PROGRAMS, for both single and double density, order # 1003, price - $15

THE COMPASS, Number 1 PROGRAMS, single density disk, includes all Basic programs and Assembly Source Codes as published in the Newsletter, order # 1004, price - $15

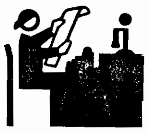
THE COMPASS, Number 2 PROGRAMS, single density disk, includes all Basic programs and Assembly Source Codes as published in the Newsletter, order # 1005, price - $15

QUICK NORTH STAR BASIC SHORTHAND CODER, MACHINE LANGUAGE PROGRAM AND SOURCE CODE, and....LANCE ROSE'S DISASSEMBLER, for single and double density, order # 1006, price - $15

!!!All INSUA Software Library selections are FULLY DOCUMENTED!!!

Send your check and order to Sid Owens, Librarian, INSUA, PO Box 1318, Antioch, CA 94509

MY LIST MICRO DATA BASE



By R. Beaver

This program will help you create, maintain, and use lists of ascii strings for many simple tasks that you may be doing by hand. for example, shopping lists, job lists, phone lists, address lists, calendar lists, memo lists, drop dead lists, birthday lists, book of lists, program lists, ham lists... all those little bits & bytes of information that keep getting misplaced.

NOW you can misplace important information in your computer! In this way, 'computer error' can provide an impressive excuse for foul-ups.
But hold, before you can join the elite (but growing) group of list users & losers, you need to learn how the program works.

DATA FILES are created by MYLIST after the user specifies filename, record size (bytes), and maximum number of records. An offset for the first record (bytes) is also needed to provide room for record 0 which keeps these four numbers.

Records 1 through the maximum are placed in the data file in random access fashion using the formula:

```
  P= I*(R+3)+R1 WHERE:
  P= file pointer for record I
  R= record length in bytes
  R1= offset for first record
```

The filesize is determined from the formula:

```
 B=int((M*(R+3)+R1)/256)+1  where;
 B=filesize in blocks
 M=maximum records allowed
```

COMMANDS are entered by the user to direct program activities. These are shown in lines 200-300 of the program. Because BASIC does not allow redimentioning of variables, it is necessary to exit the program and 'RUN' for each new file. When a file is opened, dimentioning is done.

APPLICATION NOTE: Last Issue, I described a simple to use library index system for disks. Imagine, if you will, a program which gets a disk directory (a la LIBUTIL in the last COMPASS), searches a list of program name/descriptions (a la MYLIST), and prints disk number, program name, and description. "Alternatively, a sequential search through name/description with a search through all directories to give all disk numbers which contain a given program. More fun for the next issue. Well, until then, have try MYLIST. I hope you will find it useful.

```
100 REM PROGRAM MYLIST
105 REM WRITTEN FOR 'THE COMPASS' NEWSLETTER  MARCH 1981
110 REM BY R BEAVER
115 !"PROGRAM MYLIST: LISTKEEPING PROGRAM FOR N* SYSTEMS."
120 !"CONSULT 'THE COMPASS', INSUA NEWSLETTER VOL.1 NO.2"
125 !"FOR COMMENTS ON THE USE OF THIS PROGRAM."
150 L=15:REM L=PAGING CONTROL FOR LISTS (IN RECORDS).
195 REM: MENU SECTION
200 !
205 !"COMMANDS ARE: EN -ENTER NEW RECORD."
210 !"              DE -DELETE A RECORD."
215 !"              LI -LIST RECORDS."
220 !"              CR -CREATE A FILE AND OPEN IT."
225 !"              OP -OPEN A FILE."
230 !"              SE -SEARCH & LIST RECORDS."
235 !"              EX -EXIT"
240 !
245 INPUT"<MYLIST>",C$:IFLEN(C$)<2THEN200:C$=C$(1,2)
250 IFC$="EN"THEN500:IFC$="DE"THEN600:IFC$="LI"THEN700
255 IFC$="CR"THEN800:IFC$="OP"THEN850:IFC$="SE"THEN900
260 IFC$="EX"THEN300
290 !"??":GOTO200
295 REM: EXIT LOGIC
300 IFF=0THENEND:WRITE#0%0,N,M,R,R1,NOENDMARK:CLOSE#0
305 END
400 REM SUBROUTINE SECTION
405 REM  READ THE K TH RECORD
410 READ#0%K*(R+3)+R1,A$:RETURN
415 REM WRITE THE K TH RECORD
420 WRITE#0%K*(R+3)+R1,A$,NOENDMARK:RETURN
490 REM END SUBROUTINES
495 REM  ENTER RECORD TO FILE
500 IFF=0THEN590:IFN>=MTHEN585
505 !"RECORD #",%4I,N+1:!" ",:FORI=1TOINT(R/5):!"    '",:NEXTI
510!: INPUT">",A$:IFA$=""THEN240
520 N=N+1:K=N:GOSUB420:GOTO500
585 !"FILE IS FULL.":GOTO240
590 !"NO FILE HAS BEEN OPENED.":GOTO240
595 REM DELETE ENTRY
600 IFF=0THEN590:IFN<1THEN690
605 INPUT"DELETE RECORD # ",D:IFD<1ORD>NTHEN240
610 K=D:GOSUB410:!A$:INPUT"DELETE??(Y/N):",C$:IFC$=""THEN600
615 IFC$(1,1)<>"Y"THEN600
620 K=N:GOSUB410:K=D:GOSUB420:N=N-1:!"DELETED.":GOTO600
690 !"FILE IS EMPTY.":GOTO240
695 REM LIST ENTRIES
700 IFF=0THEN590:IFN<1THEN690:!%4I,N," RECORDS."
705 INPUT"ENTER START AND END RECORD NUMBERS FOR LIST :",S,E
710 IFS<1ORS>ETHEN240:IFE>NTHEN240
715 INPUT"ENTER OUTPUT DEVICE NUMBER:",V:IFV<0ORV>7THEN240
720 C=0
725 FORK=STOE:GOSUB410
730 !#V,%4I,K,"> ",A$
735 C=C+1:IFC<LTHEN750:C=0:INPUT"(RET TO CONTINUE)",C$
750 NEXTK:GOTO240
795 REM  CREATE FILE
800 IFF=1THEN890
805 INPUT"CREATE NEW FILE NAME:",N$:IFN$=""THEN240
810 INPUT"ENTER RECORD SIZE (BYTES), MAXIMUM RECORDS:",R,M
815 IFR<1ORM<1THEN810
820 INPUT"OFFSET (BYTES) FOR FIRST RECORD (MIN 25):",R1
825 IFR1<25THEN820
830 B=INT((M*(R+3)+R1)/256)+1
835 CREATE N$,B:OPEN#0,N$:WRITE#0%0,0,M,R,R1,NOENDMARK:CLOSE#0
840 !"FILE ",N$," CREATED WITH LENGTH ",%4I,B," BLOCKS.":GOTO860
845 REM  OPEN FILE
850 IFF=1THEN890
855 INPUT"OPEN FILENAME :",N$:IFN$=""THEN240
860 OPEN#0,N$:READ#0%0,N,M,R,R1:F=1
865 DIM A$(R),S$(R)
870 !"FILE ",N$," IS NOW ACTIVE."
875 !"RECORDS: ",%4I,N,"      MAX RECORDS: ",%4I,M
880 !"RECORD SIZE :",%4I,R," OFFSET :",%4I,R1
885 !:GOTO240
890 !"FILE ",N$," ALREADY IN USE.EXIT AND RE-RUN PGM.":GOTO240
895 REM SEARCH AND LIST RECORDS
900 IFF=0THEN590:IFN<1THEN690
905 INPUT"ENTER SEARCH STRING: ",S$:IFS$=""THEN240
910 INPUT"ENTER BEGINNING BYTE NUMBER IN RECORD:",B1
915 IFB1<1ORB1>R-LEN(S$)THEN990:B2=B1+LEN(S$)-1
920 INPUT"ENTER OUTPUT DEVICE NUMBER:",V:IFV<0ORV>7THEN240
925 C=0
930 FORK=1TON:GOSUB410:IFLEN(A$)<B2THEN950
935 IFA$(B1,B2)<>S$THEN950
940 !#V,%4I,K,"> ",A$
945 C=C+1:IFC<LTHEN950:C=0:INPUT"(RET TO CONTINUE)",C$
950 NEXTK:GOTO240
990 !"RECORDS ARE ",%4I,R," BYTES. REQUEST IS OUT OF BOUNDS."
995 GOTO905
999 REM END OF PROGRAM MYLIST
```

NS CP/M

CONTINUED FROM PAGE 1

memory requirements and write it to a new disk. But again it only works for a Horizon.

If you are a North Star user who bought their disk controller to run on an alien machine, you are out of luck. North Star does not provide the normal MOVCPM as separate program for some reason, though the normal SYSGEN is provied.

Disk organization is different too. The user area is at disk address 8 instead of Lifeboat's address 28...but then Lifeboat has changed to address 19 in their version 2.2. So much for standardization.

Further enhancements to North Star's CP/M include the ability of DOS LI command to list the location of BDOS, User area, CCP, etc. In addition North Star provides a DIRDUMP utility (similar to one from the CP/M users group) that lists every location for the program segments that CPM scatters around on the disk. Lifeboat doesn't provide this but their latest version 2.2A does list the system information from DOS and in addition booting up CPM from DOS with a GO CPM command.

North Star provides thier own version of Softdoc for CPM in a 48 page manual that strips down the unwieldly 5 volumes of CPM documentation that everybody complains about.

North Star has also cleaned up the CPM error message system, replacing the non-infomation messages with DOS' standard error message code system. Their COPY program has added a check-sum command to verify disks and they use the North Star convention of Drive 1 and 2 rather than Drive A: and B:. But to keep the user alert, they have switched back to CPM's A: and B: for PIP and other general systems commands.

All the stanard CPM utilities: ED,ASM,DDT,DUMP,etc are furnished by North Star and are compatible as is the system's ability to read and execute programs from Lifeboat's disks. It all works fine, running Microsoft Basic on North Star's official CPM as long as you keep the alien disks out of Drive A:.
No uitities are provided for transferring old North Star files and programs to CPM, but you can make the transfers (see below).
The company says the idea of running their great BASIC on CPM has occured to them and they are thinking about it but nothing is official. If you are thinking about it too, see the description of North Star BASIC under CPM by Bob Stek in the Vol.1 No.1 issue of the COMPASS. Info-Soft's updated patches for 5.2 BASIC are now available as are utilites to transfer North Star disk programs and files to CPM. They all seem to work fine.

The impelementation of North Star's CPM as a turnkey system for the Horizon probably is designed open that vast store of "professional" business software to the prospective hardware buyer. Business buyers are where the action is...according to a North Star spokesman. He has managed to turn the company around from selling 60 percent of their computers to individuals and only 40 percent to business users a year ago to 60 percent business, 40 personal this year. Magic Figures!!!!

PENCIL POINTS

by Sid Owen
(415) 321 7979

Pencil Modifications

I am a devotee of the Electric Pencil text editor and use it extensively for personal and business purposes. Since I used it for nearly a year before modifying it, it must be good!

On some of my discs, the computer boots up with the Electric Pencil running. I realized the full value of this arrangement only after I had modified the Pencil presets to the values that I normally use. Here is how you can make similar modifications to your Pencil, but before you start, please put a write protect sticker on your personalized and running Pencil, copy it on a new disc and make all changes on the copy.

Since there are many versions of the Pencil to accommodate various equipments, you will need to find the addresses of the presets applicable to your version. Do this as follows:

1. Load your personalized Pencil normally with a "Go Pencil" command.

2. Jump to DOS.

3. Load the North Star Monitor

4. Load your personalized Pencil at a second location at least 8900 (decimal) higher than the first, using the DOS LF command.

5. Jump back to the first Pencil and change the various settable values, (e.g. margin, page, etc.) Use different values for each and keep track of new values so that you can recognize them.

6. Back to DOS and jump to Monitor. Now compare the files with the CM command; i.e. compare the altered Pencil with the unaltered one higher up (decimal file length should be at least 8900). Most of the differences will be due to the changes you made. You can now read where in memory the changes were made and also what the old and new values are.

Surprise! The interesting changes are outside the basic Pencil file. My Pencil for example starts at

CONTINUED ON PAGE 5

COMPASS NEWSLETTER
COPYRIGHT 1981
by the
INTERNATIONAL
NORTH STAR
USERS' ASSOCIATION,
INCORPORATED
Post Office Box 1318
Antioch, CA 94509

permission to reprint will be given on request to users' groups

The International North Star Users' Association is a group independent of North Star Computers, Inc. North Star does not have any control over, nor responsibility for, any of the material contained herein. While every attempt has been made to insure the accuracy of the articles, INSUA, likewise, can bear no responsibility nor liability for the contents of this newsletter.

WHO'S WHO

Dr. William Banaghan
Chairperson

Mr. Robert Beaver
Membership Chairperson
Treasurer

Ms. Sarah Wassermann
Secretary

Mr. Sid Owen
Software Librarian

Mr. Pavel Breder
Club Coordinator

Mr. Clyde Steiner
Compass Editor

PENCIL POINTS

CONTINUED FROM PAGE 4

2A00 and is recorded on 24 disc sectors (single density). The recorded file therefore ends at 4200H. Changing the settable values however (step 5 above) resulted in changes out in the 4C90H area!

It turns out that the preset values are written out there by the very first lines of the program when it is loaded and run. To see this, use the Monitor DH command from 2A20 to 2A3F. Starting at 2A28 we see 3E 0C, 32 A1 4C; that is Move Immediate (MVI) into the accumulator the value 0CH, then Store the accumulator (STA) in 4CA1.

Since we have already identified 4CA1 as the Page Spacing value, obviously we can change the preset value by changing 2A28 using the DS Monitor command. Before you make any changes however, reload Pencil freash from disc. Since I use loose 8 1/2 X 11 sheets in my printer I chose the maximum permissible Page Spacing value of 20H so that there is no chance of starting to print the second page at the bottom of the first. The locations of presets on my Pencil are: (SEE FIGURE ONE)

The next preset in the program is Page Length at 2A2E. Its value was 36H and I changed it to 48H since I ~~pre~~fer to manually determine when to end a page and dont't want Pencil to end it prematurly. Finally, Line Length is specified as 3EH at 2A33. However, if we simply change this to our favorite value, we won't be able to make any other preset changes. That is, there are no provisions or room in the program to change any other presets since they are all zero in the original Pencil and consequently they don't need to be moved or written out in the 4CXX area. There is no room at the start of the program to write more commands. The problem is, where is there room in the program?

Use your Monitor DA command to find the message "THE ELECTRIC PENCIL". It starts at 366A in my version. This is the message that appears each time you run the Pencil. Replace the first three bytes with 47, 4F, 00. The message is now "GO"; plus many spare bytes ("00" ends the written message). Now, back at 2A33 put in your favorite line length (I use 40H = 64D) and add a CALL to the newly available space after "GO", e.g. starting at 2A33, 40 CD 6D 36. At 366D we put the STA command 32 9E 4C, and we

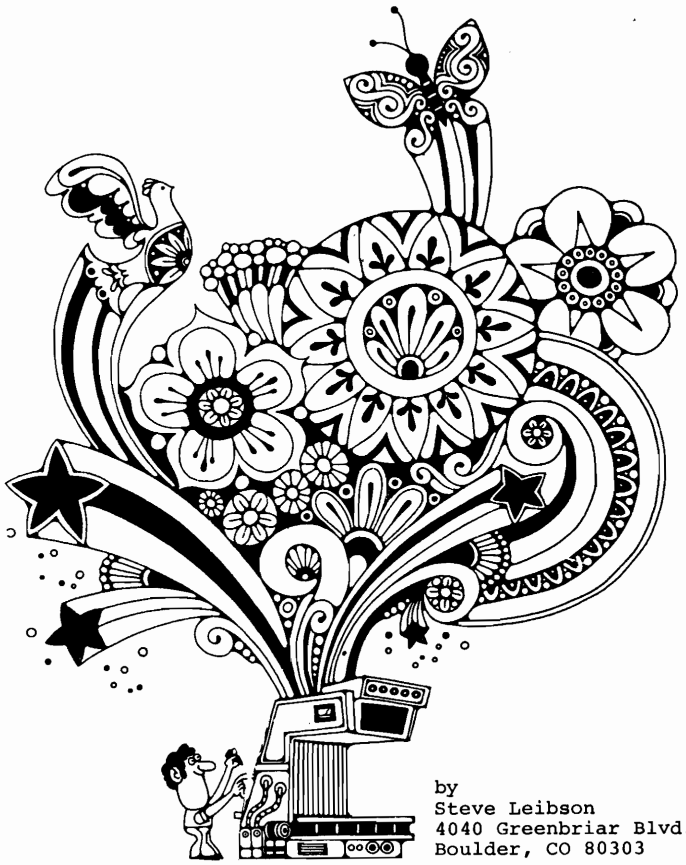
PRESET LOCATIONS

PRESET	LOCATION	VALUE	NEW VALUE
Scroll Speed	4C97	0	48H = NA
Line Length	4C9E	3EH = 62	40H = 64
Line Spacing	4C9F	0	
Left Margin	4CA0	0	8H = 8D
Page Spacing	4CA1	0CH = 12	20H = 32
Right Justify	4CA8	0	
Page Length	4CBD	36H = 54	48H = 72
Page Number	4CBF	0	
Print Length	4CC1	0	48H = 72

PENCIL POINTS - FIGURE ONE

SORTLIST

```
100 REM PROGRAM 'SORTLIST'. R BEAVER  MARCH 1981.
105 REM WRITTEN FOR THE COMPASS NEWSLETTER V1.N2.
110  DEF FNX(X)=(X-1)*G7+1
115  DEF FNY(Y)=Y*G7
120 !"ALPHA SORT PROGRAM FOR 'MYLIST' DATA FILES."
125 !"PLACE DATA DISK IN DESIRED DRIVE."
130 INPUT"ENTER DATA FILENAME: ",N$:IFN$=""THEN130
135 OPEN#0,N$:READ#0%0,N,M,R0,R1
140 !:!"FILE  ",N$," OPEN FOR SORT."
145 !"RECORDS: ",%4I,N,"   MAX RECORDS: ",%4I,M
150 !"BYTES/RECORD: ",%4I,R0
155 !"OFFSET FOR FIRST RECORD (BYTES): ",%4I,R1
160 IFN>1THEN165:!"LESS THAN TWO RECORDS. END PGM.":END
165 !:INPUT"ENTER START AND END BYTE NUMBERS FOR SORT:",S,E
170 IFS<1ORS>ETHEN145:IFE>R1THEN145
175 INPUT"ENTER OUTPUT DEVICE NUMBER FOR PRINT (8=NO PRINT):",V
180 IFV<0ORV>8THEN175
185 INPUT"ENTER DESTINATION FILE NAME (RET=NONE):",M$
190 IFM$=""THEN205:!"PLACE DESTINATION DISK IN DESIRED DRIVE."
195 INPUT"NEW ? (Y/N):",C$:IFC$=""THEN205:IFC$(1,1)<>"Y"THEN205
200 B=INT((M*(R0+3)+R1)/256)+1:CREATE M$,B
205 G7=E-S
210 DIM A$(R0)
215 DIM R$(G7),K$(G7),R1$(G7*N),U8(INT(LOG(N)/LOG(2)+.5),2),O(N)
220 FORI=1TON:O(I)=I
225 READ#0%I*(R0+3)+R1,A$
230 IFE>LEN(A$)THEN240
235 R1$(FNX(I),FNY(I))=A$(S,E)
240 NEXTI
245 IFN*G7<200THEN265
250 !"THIS WILL TAKE A BIT OF TIME."
255 IFN*G7>1000THEN!"WHY NOT TAKE A SHORT BREAK WHILE I WORK."
260 IFN*G7>5000THEN!"CHECK BACK IN ABOUT 15 MINUTES."
265 GOSUB1000
270 IFM$=""THEN280
275 OPEN#1,M$:WRITE#1,N,M,R0,R1,NOENDMARK
280 FORI=1TON:K=O(I):READ#0%K*(R0+3)+R1,A$
285 IFV<8THEN!#V,A$
290 IFM$>""THENWRITE#1%I*(R0+3)+R1,A$,NOENDMARK
295 NEXTI
300 CLOSE#1:CLOSE#0
305 END
1000 REM QUICKSORT STRING R1$ USING FNX AND FNY AS POINTERS
1005 REM G7=STRING LENGTH ,R1$=SUPERSTRING ,O(I) RETURNS ORDER.
1010 N8=1:U8(1,1)=1:U8(1,2)=N
1020 L=U8(N8,1):R=U8(N8,2):N8=N8-1
1030 I=L:J=R:Z8=INT((R-L)*RND(0)+0.5)+L:!"*",
1035 K$=R1$(FNX(Z8),FNY(Z8))
1040 IFR1$(FNX(I),FNY(I))>=K$THEN1050:I=I+1:GOTO1040
1050 IFK$>=R1$(FNX(J),FNY(J))THEN1060:J=J-1:GOTO1050
1060 IFI>JTHEN1090
1070 R$=R1$(FNX(I),FNY(I))
1071 R1$(FNX(I),FNY(I))=R1$(FNX(J),FNY(J))
1080 R1$(FNX(J),FNY(J))=R$
1085 O(0)=O(I):O(I)=O(J):O(J)=O(0)
1086 I=I+1:J=J-1
1090 IFI<=JTHEN1040
1110 IFJ-L>=R-ITHEN1150
1120 IFI>=RTHEN1140
1130 N8=N8+1:U8(N8,1)=I:U8(N8,2)=R
1140 R=J:GOTO1180
1150 IFL>=JTHEN1170
1160 N8=N8+1:U8(N8,1)=L:U8(N8,2)=J
1170 L=I
1180 IFL<RTHEN1030
1190 IFN8>0THEN1020
1199 !:RETURN
```



by
Steve Leibson
4040 Greenbriar Blvd.
Boulder, CO 80303

THE I/O FARM

Interfacing computers to other devices has fascinated me for years. One of the reasons for this is the rush of satisfaction you get when a system finally works. Usually, you are confronted with little or no documentation from either the computer or peripheral manufacturer.

Take North Star for example. There is an inch thick book on the DOS, Monitor and Basic. The software is relatively well documented as to what it can do.

On the other hand, there are two serial I/O (Input/Output) ports, two parallel ports (one input and one output) and a real-time clock in the Horizon computer. Total documentation for these features consists of a schematic and a listing of the Horizon I/O routines. This is hardly documentation to rave about.

Still, it is very possible to make use of these features. You can even add I/O boards to a North Star system with very good results.

This column is being written to help you do just that. We will be covering the basics of I/O, North Star's I/O hardware and I/O boards from other manufacturers. If you have any questions that you think might make a good column, why not write me?

This issue, we will look at a most common problem: how do you make a serial port work well with a printer?

A Handshake Instead of a Kiss

Recently, I read a letter from a dissatisfied North Star user. This letter appeared in several of the microcomputer magazines' letters columns. The problem he was having was getting a printer connected to the North Star to print at full speed.

In order to do this, the printer, with a "standard" RS-232 serial interface needed to handshake with the computer to tell it when to stop sending. This would allow the printer time to catch up. The user was unhappy with North Star because "North Star DOS does not test for handshaking signals!"

Handshaking over RS-232 interfaces is a problem that will plague computer users forever. Though RS-232 is indeed a standard, I have yet to see a completely standard implementation and I have seen a lot of RS-232 devices.

Let's get one thing straight right at the start. There is no such thing as a "standard" RS-232 handshake, as required by a direct computer-peripheral link. Several manufacturers of peripherals have implemented RS-232 interfaces with handshakes but they are all non-standard.

The Connector, DCE's and DTE's

Figure 1 is a picture of the RS-232C connector as defined by the standard. It has 25 pins, most of which are usually left unconnected. The most commonly used signals are the "basic eight". These are pins 1 through 7 and pin 20. We will consider only those pins in this article.

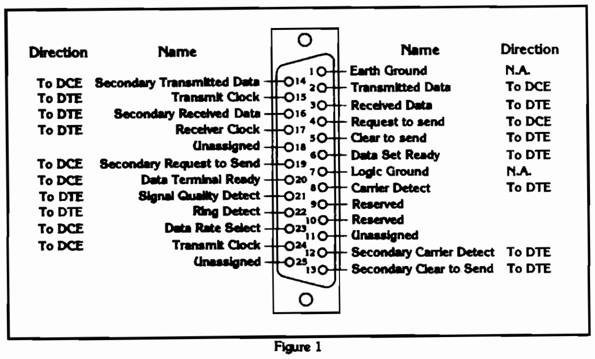


Figure 1

The first things to note in figure 1 are the direction columns. Some of the signals are marked "To DCE" and others are marked "To DTE". DCE stands for Data Communications Equipment and DTE stands for Data Terminal Equipment.

That's modems and terminals to you. You see, the RS-232 standard was designed to connect terminals to modems for the telephone company. When timesharing of computers became popular, several users wanted to use the phone lines to connect their terminals to a central computer. After all, the telephone lines were already in place, why not use them?

The phone company would not allow terminals to be connected directly to telephone lines. They required that the terminal be connected to a device that would encode the digital signal into tones compatible with the phone system. This device is called a modem.

Just Data

In the simplest form, this terminal-modem connection uses only three wires: Transmitted Data, Received Data and a Logic Ground. This is shown in figure 2. The three signals are found on pins 2, 3 and 7 of figure 1. Signal names are always from the perspective of the terminal.

These three wires are sufficient to allow data transfer between a terminal and a modem. The modem is also connected to the telephone line. At the other end of the phone line, there is another modem connected to a computer masquerading as a terminal.

Now, let's say we are going to connect a printer to a computer. This poses an immediate problem. Which device is a DTE and which is a DCE? More specifically, which is going to transmit on pin 2 of the

THE I/O FARM

CONTINUED FROM PAGE 6

connector and which on pin 3?

Is this question important? You bet it is! If both devices think they are DTE's, they will both attempt to transmit data on pin 2 and receive on pin 3. This won't work because one device must listen to the other. One device must transmit on pin 2 and receive on pin 3, the other must receive on pin 2 and transmit on pin 3.

The confusion of deciding which device is going to look like a DTE and which will be a DCE usually means that 50% of the time, you will be faced with the problem of connecting two DTE's or two DCE's together.

And Now, Handshaking

Now that we have the data lines set up right, let's look at that infamous handshake. We are trying to find a pair of lines that look like they might be used to control the flow of data. Aha! Pins 4 and 5 are marked Request to Send and Clear to Send. These look like prime candidates for a pair of handshake lines. Couldn't we have the printer control Clear to Send so that it could tell the computer when to stop for a while?

Unfortunately, that's the mistake several printer manufacturers make. The strict definition of the Request to Send and Clear to Send signals do not allow this type of use.

When the transmitting DTE wishes to send data to the DCE, it is supposed to assert Request to Send. This should cause the DCE to establish a link over the phone line to the remote modem. When this link is established, the DCE asserts Clear to Send and the DTE may start transmitting. So far, so good.

The hook is that the DCE is not allowed to negate Clear to Send until the DTE negates Request to Send. Therefore, Clear to Send cannot be used to stop data transmission arbitrarily.

Similarly, Data Set Ready and Data Terminal Ready on pins 6 and 20 are to be used as power-on indicators. They are not to be used as a handshake either.

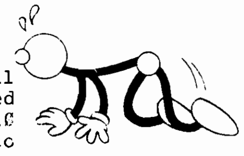
So what? We could go ahead and use pin 5, Clear to Send as a handshake line. If the printer drops this line, the computer had better stop transmitting. Won't this work?

CONTINUED ON PAGE 8

WILD CARD

Wild Card option for Memory Search routine of North Star Monitor 5.2

written by: Joe Maguire



Patching the Monitor

The Monitor occupies the full eight blocks in which it is stored on the disk. A new file of 10 blocks (DD) should be created to hold the original file and the patches.

```
                ;
                ;
                ;
                ;
                ;
                ; Syntax:   SM addr Byte,*,Byte,Byte,*,...etc.
                ;           The asterisk character will match
                ;           any byte value.
                ;
                ; Example:  To find all JMPs to the EØØH addr
                ;           area, type:  SM addr C3,*,ØE
                ;
                ; Note:     ,*, and ,"*", are not equivalent.
                ;           The first is the wild card.  The
                ;           second will substitute the Hex value
                ;           2A in the byte list.
                ;
ØEØØ            ENTRY:  ORG   ØEØØH  ;For Monitor at ØEØØH as supplied
                ;                     on factory master disk.  You can
                ;                     change this as required.
                ;
1Ø34            SMCOM:  ORG      ENTRY+234H  ;Change entry in command
1Ø34 8Ø16               DW       ZBUF        ;table to jump to this
                ;                            ;routine
                ;
11BC =          SMEM:   EQU      ENTRY+3BCH  ;Original SM address
                ;
11CE            PACH2:  ORG      ENTRY+3CEH  ;Jump out to check for
11CE 8F16               DW       MATCH       ;wild card.
                ;
11D6 =          NMACH:  EQU      ENTRY+3D6H  ;Return here if no match
11DØ =          YMACH:  EQU      ENTRY+3DØH  ;Return here if a match
                ;
138F            PACH3:  ORG      ENTRY+58FH  ;Jump out to store wild
138F 9B16               DW       PUTWC       ;card in buffer.
                ;
1393 =          BFILL:  EQU      ENTRY+593H  ;Original buffer fill
13A1 =          QUOTE:  EQU      ENTRY+5A1H  ;Check for quote mark
                ;
168Ø            ZBUF:   ORG      ENTRY+88ØH  ;Start of Wild Card routine
                ;
168Ø 21AF16             LXI     H,WCBUF ;Zero the Wild Card buffer
1683 Ø61Ø               MVI     B,16    ;Buffer length
1685 36ØØ       ZB1:    MVI     M,Ø     ;Move in the zero
1687 23                 INX     H       ;Bump pointer
1688 Ø5                 DCR     B       ;Decrement counter
1689 C28516             JNZ     ZB1     ;Keep looping if not done
168C C3BC11             JMP     SMEM    ;Return to SM routine
                ;
168F D5         MATCH:  PUSH    D       ;Save original pointer
169Ø 14                 INR     D       ;Bump it up 256 bytes
1691 1A                 LDAX    D       ;Look in WC buffer
1692 FE2A               CPI     '*'     ;See if it's a WC
1694 D1                 POP     D       ;Get back pointer
1695 C2D611             JNZ     NMACH   ;No WC, return to routine
1698 C3DØ11             JMP     YMACH   ;Yes.  Bypass routine
                ;
169B FE2A       PUTWC:  CPI     '*'     ;See if WC in byte list
169D C2A113             JNZ     QUOTE   ;No WC, return to CPI '"'
16AØ E5                 PUSH    H       ;This is part of
16A1 D5                 PUSH    D       ;original code
16A2 E5                 PUSH    H       ;Save original pointer
16A3 24                 INR     H       ;Bump it up 256 bytes
16A4 362A               MVI     M,'*'   ;Move in WC flag
16A6 E1                 POP     H       ;Get back pointer
16A7 C39313             JMP     BFILL   ;Continue orig routine
                ;
                ; Note:  The Wild Card buffer lies 256 bytes above
                ;        the regular byte list buffer in order
                ;        that a simple INR can be used to adjust
                ;        the required pointers.
                ;
16AF            WCBUF:  ORG     ENTRY+8AFH
                ;
16AF                    DS      16      ;Length is 16 bytes
                ;
16BF                    END
```

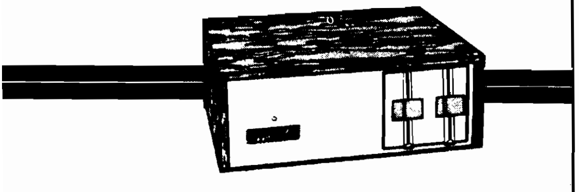
THE I/O FARM

CONTINUED FROM PAGE 7

It might, depending on what pin 5 is connected to in the computer. On some computers, pin 5 isn't connected to anything and use of pin 5 as a handshake line is useless. Other computers have pin 5 connected to the integrated circuit sending the data.

If this integrated circuit stops as soon as Clear to Send is dropped, the character being transmitted at that instant will be ruined. That's because serial communication transmits characters serially (what else?). There is a precise timing relationship between each bit. If the transmission is interrupted in mid- character, this timing relationship is destroyed and the character is ruined.

Some integrated circuits used for serial transmission will not stop until they finish transmitting the current character. Computers using these parts have a better chance of handshaking with a printer. This only succeeds however if the printer doesn't wait until its buffer is full to negate Clear to Send. Most printers give themselves room to spare in their buffers and this scheme succeeds.



The Horizon

The North Star Horizon serial ports are brought out to female connectors. Normally, a female connector is associated with the DCE. This should make it ideal for connecting terminals which are supposed to use male connectors. Serial communications in the North Star are handled by an 8251 USART (Universal Synchronous/Asynchronous Receiver/Transmitter). This chip does have a Clear to Send input. An Intel 8251 will stop transmitting in the middle of a character while Intel 8251A's will complete the current character and then stop. My North Star has National 8251's in it and I don't know which way they work. As you will see, it won't matter.

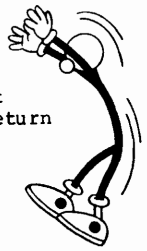
The serial dip (dual inline package) configuration header allows you to change which signals internal to the North Star are

CONTINUED ON PAGE 9

HEX MATH

Hex Math routine which will give you the sum and difference of any two numbers.

```
                 ; Hex Math routine for NorthStar Monitor 5.2
                 ;
                 ;     written by:   Joe Maguire
                 ;
                 ; Syntax:  HM  value1  value2
                 ;          Values can be anything from 0 to FFFFH
                 ;          or 0 to 65535T (decimal).  Overflow or
                 ;          underflow is ignored.  Both values must
                 ;          be present.
                 ;
0E00             ENTRY:  ORG  0E00H  ;For Monitor at 0E00H as supplied
                 ;                    on factory master disk.  You can
                 ;                    change this as required.
                 ;
105E             HMCOM:  ORG  ENTRY+25EH  ;Command table entry
                 ;
105E 484D                DB      'HM'     ;Command syntax
1060 0016                DW      HMATH
1062 00                  DB      0        ;End of table mark
                 ;
                 ; Subroutine addresses within Monitor 5.2
                 ;
1181 =           VOUT:  EQU  ENTRY+381H  ;Output a 16 bit value from HL
12D7 =           COUT:  EQU  ENTRY+4D7H  ;Output a character from A
12E7 =           SOUT:  EQU  ENTRY+4E7H  ;Output a string pointed to by HL
12F1 =           CRLF:  EQU  ENTRY+4F1H  ;Output C/R and L/F
147D =           GETV:  EQU  ENTRY+67DH  ;Get a 16 bit value to HL
                 ;
1600             HMATH:  ORG  ENTRY+800H  ;Hex Math routine address
                 ;
1600 CD7D14              CALL    GETV     ;Get value1
1603 E5                  PUSH    H        ;Save it
1604 CD7D14              CALL    GETV     ;Get value2
1607 E5                  PUSH    H        ;Save it
1608 212916              LXI     H,MSG    ;Point to format string
160B CDE712              CALL    SOUT     ;Send it out
160E D1                  POP     D        ;Get value2 to DE
160F E1                  POP     H        ;Get value1 to HL
1610 E5                  PUSH    H        ;Save it again
1611 19                  DAD     D        ;Add DE to HL
1612 CD8111              CALL    VOUT     ;Send out the result
1615 E1                  POP     H        ;Get value1 back to HL
                 ;
1616 7A                  MOV     A,D      ;Now get two's complement
1617 2F                  CMA              ;of DE to find difference
1618 57                  MOV     D,A
1619 7B                  MOV     A,E
161A 2F                  CMA
161B 5F                  MOV     E,A
161C 13                  INX     D

161D 19                  DAD     D        ;Add it to HL
161E 3E2D                MVI     A,'-'    ;Send out minus sign
1620 CDD712              CALL    COUT
1623 CD8111              CALL    VOUT     ;Send out minus result
1626 C3F112              JMP     CRLF     ;Give a C/R & L/F & return
                 ;
1629 20203D      MSG:    DB      '  ='
162C 20202B              DB      '  +'
162F 00                  DB      0        ;End of message mark
                 ;
1630                     END
```



CLIP TIPS

```
10REM Latest revision 2-23-81, by Steven Inness
20REM 3200 Marlene Ave., Redding, CA  96002. (916) 221-1795.
30!"A short long division program..."
40INPUT"Dividend (numerator), 8 digits max? ",N
50INPUT"Divisor (denominator), 7 digits max? ",D
60Q=INT(N/D):!Q,".",
70N=(N-Q*D)*10:Q=INT(N/D):!%1I,Q,:GOTO70
```

THE I/O FARM

CONTINUED FROM PAGE 8

brought out to the various pins on the serial connector. Figure 3 shows the header as shipped by North Star.

Pin 2 of the RS-232 connector is linked through the header to the Horizon's data receiver and pin 3 is connected to the data transmitter. Thus the standard configuration is to receive on the Transmitted Data line and transmit on the Received Data line. This is consistent with a DCE.

Now things get complicated. Pin 4, Request to Send is connected to the Data Set Ready input on the USART. Clear to Send, pin 5 is connected to the Data Terminal Ready output of the USART. Carrier Detect, pin 8 is connected to a signal which appears in the North Star motherboard status byte and Data Terminal Ready, pin 20 connects to the Clear to Send input of the USART.

Just what is going on here? Data Terminal Ready is the signal that indicates that the DTE is powered on. This is a good choice to control the USART's Clear to Send input since the signal will be asserted when power is turned on and then stay on.

The Data Terminal Ready input of the USART is simply connected to a register in the 8251 that the processor can read. This is good because the 8251 cannot be relied on to stop properly when the DTE drops Clear to Send unless it is an 8251A. If the processor checks the state of the USART Data Terminal Ready bit before sending each character, the handshake will work.

The Data Set Ready input of the USART can also be read by the processor so it serves well connected to Request to Send. The Request to Send output of the USART can be used to control the state of the Data Set Ready line on pin 6 of the connector.

This may all seem quite complicated, and it is. Putting a handshake on RS-232 isn't easy.

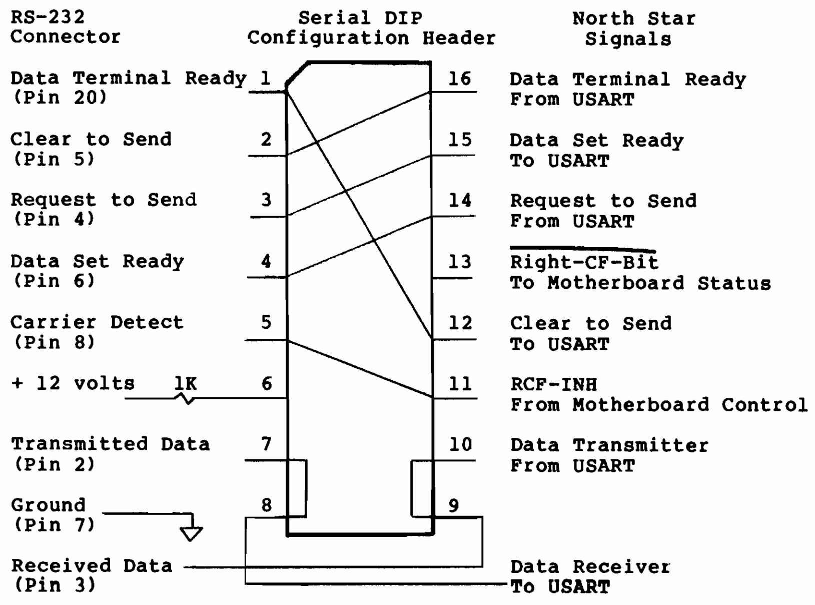
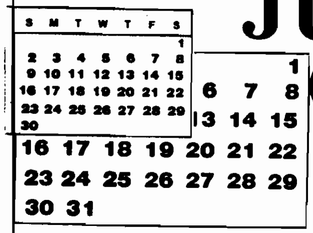


Figure 3
Standard Header Configuration



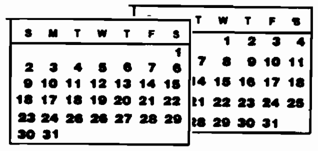
JULIAN CODE

The Julian period is a universal cycle used in chronology, invented by French chronologist Joseph Scaliger in 1582.

It should be noted that the Julian period has nothing to do with the Julian Calendar or Julius Caesar. A Julian period is composed of 7,980 years, and is a multiple of the solar cycle (28 years), the Metonic cycle (19 years), and the Roman Indiction (15 years).

The year and day of the beginning of the julian period, as established by Scaliger, are January 1, 4713 B.C., since the three cycles all began together on that date. The cycle is complete on A.D. 3267.

To find the number of a year in the Julian period, add the year to 4713; for example, the year 1980 is 6693 of the Julian period. Individual days in the period are indicated by the days elapsed since the beginning of the period, and every day has its own Julian-day number. Tables of Julian-day numbers are given in government ephemerides to facilitate turning any date into its Julian-day number, for astronomical calculations involving large time intervals.



There are definite advantages if Julian Day numbers are used to record dates. Normally the space required to store a date in file is reduced over that which is required for a string such as "1-15-81". In addition, the number of days between two Julian Day numbers is calculated by a simple subtraction and the day of the week for a given Julian Day number is easily calculated.

User defined functions are quite effective in converting to and from Julian Day numbers. Assuming the date is in string form, i.e., J$="2-25-81", the following user defined function will perform the conversion to Julian Day number:

```
10 DEFFNJ(J$)
20 J=LEN(J$)
30 J1=VAL(J$)
40 J2=ABS(VAL(J$(J-4)))
50 J3=1900+VAL(J$(J-1))
60 IF J1<3 THEN J3=J3-1
70 IF J1>2 THENJ1=J1+1 ELSEJ1=J1+13
80 J=INT(365.25*J3)+INT(30.6001*
                        J1)+J2+1720982
90 RETURNJ
100 FNEND
```

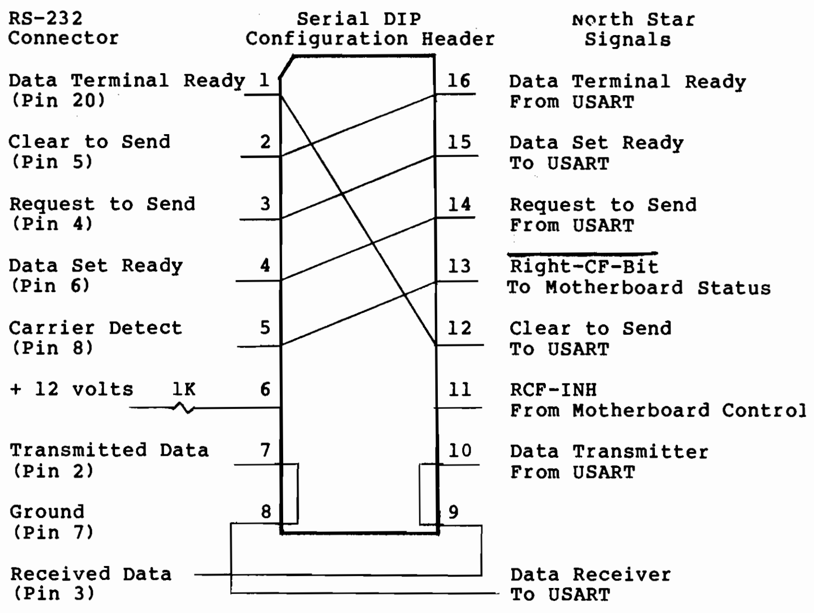


Figure 4
Modified Header Configuration

THE I/O FARM

CONTINUED FROM PAGE 9

When I connected my printer to my North Star, I rewired the configuration header to make the software for implementing a handshake easier.

Making the Connection

Figure 4 shows how I rewired my configuration header. Two changes have been made. Carrier Detect, pin 8 is now connected to the motherboard status port. This port is easier to read than the USART register and is used for handshaking. Data Terminal Ready, pin 20 is connected to the USART Clear to Send input.

My printer, a NEC Spinwriter is connected to the computer as shown in figure 5. Remember, the printer is emulating a terminal (DTE). Pins 2,3 and 7 are brought straight over. Pins 4 and 5 on the printer are connected to each other so that when the printer asserts Request to Send, it immediately gets back a Clear to Send signal. Pins 6, 8 and 20 are connected together so that when the printer asserts Data Terminal Ready it gets back Carrier Detect and Data Set Ready. Both of these signals must be asserted before the printer will print.

The handshake for the printer is on pin 19. This is Secondary Request to Send, a signal we have not discussed. NEC avoided violating the standard by using a little used pin for the handshake. Pin 19 is connected to the North Star's pin 8 which is Carrier Detect. The state of this pin can be read by the processor in the Horizon's motherboard status register.

CONTINUED ON PAGE 11

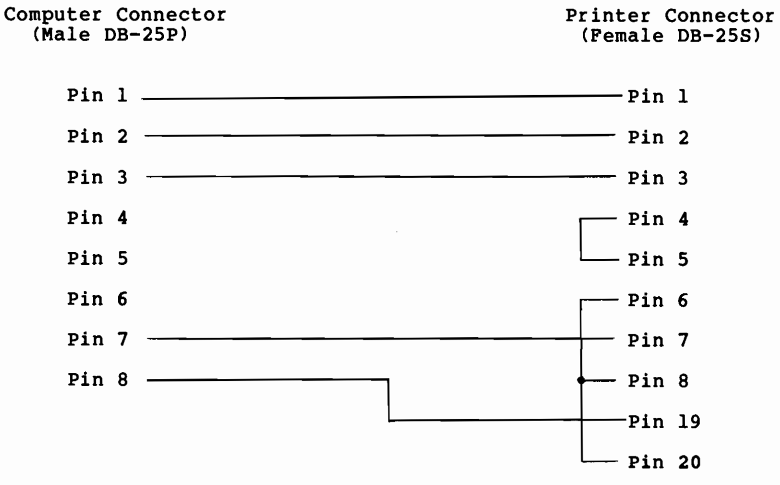
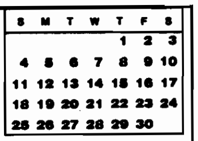


Figure 5
Computer to Printer Cable



JULIAN CODE

The following user defined function will convert a Julian Day number to a date string, J$:

```
110 DEFFNJ$(J)
120 J2=J-1720982
130 J3= INT((J2-122.1)/365.25)
140 J1=INT((J2-INT(365.25*J3))
                       /30.6001)
150 J2=J2-INT(365.25*J3)-INT
                    (30.6001*J1)
160 IF J1<14 THEN J1=J1-1 ELSE
                         J1=J1-13
170 IF J1<3 THEN J3=J3+1
180 J2$=STR$(J2)
190 J1$=STR$(J1)
200 J3$=STR$(J3-1900)
210 J$=J1$(2)+"-"+J2$(2)+"-"+J3$(2)
220 RETURNJ$
230 FNEND
```

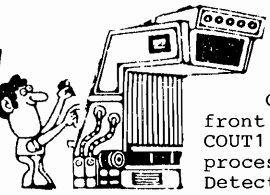
If it is desirable to determine the day of the week when a Julian Day number is known, the following user defined function is useful:

```
240 DEFFNJ4(J)=INT(7*((J-1720977)
        /7-INT((J-1720977)/7))+.5)
```

J4 will be the day of the week where 0 is Sunday, 1 is Monday, etc. through 6 which is Saturday. The following subroutine will return with F$ containing the day of the week for a Julian Day number, J:

```
250 J4=FNJ4(J)
260 RESTORE 270
270
DATA"SUNDAY","MONDAY","TUESDAY","WE
DNESDAY","THURSDAY",
"FRIDAY","SATURDAY"
280 FOR K=0 TO J4
290 READ F$
300 NEXT
310 RETURN
```

THE I/O FARM



CONTINUED FROM PAGE 10

And Now the Software

Now that everything is hooked up, all we have to do is tell the processor how the handshake works and let the system fly. If you have dabbled in assembly level software before, you may stop reading and write the routines as an exercise.

For those of you who may not be too familiar with assembly code or North Star I/O drivers, I'll show you what I did. The printer in the North Star system is device number 1. The console is device zero. Two different software routines are listed below. The first is for North Star DOS, releases 5.1 and earlier. Release 5.2 needs a different routine because of the status routines added to the DOS. The release 5.1 routine would work in release 5.2 but the status routines wouldn't be quite right.

COUT10 has been added to the front of the standard North Star COUT1 routine. First, the processor will wait for the Carrier Detect bit to go ready, then it will wait for the USART transmitter to go ready also.

Again, the first part of the routine has been added. It is still mostly North Star's code. Notice that the status routine does not actually send the character to the printer. OST is called by the character output routine. OST1 is specifically for device 1, the printer port. The status routine is supposed to set the Z or Zero status bit in the processor if the device is ready and then return to the calling routine.

The character output routine continuously calls the OST routine until it returns a ready indication. It then transmits the character.

If this article has been somewhat confusing, chances are you haven't done too much assembly language coding. I am writing this series to acquaint you with some of the basics of interfacing and assembly programming for interfacing. If you have questions that you would like to see covered or special topics that interest you, why not write me? I want to know what you would like to see in "The Compass" relating to I/O.

Release 5.1 Printer Handshake Routine

```
COUT10      IN      A,(6)       GET THE MOTHERBOARD STATUS BYTE
            AND     10H         MASK OUT ALL BITS BUT CARRIER DET.
            JP      Z,COUT10    IF ZERO, WAIT FOR READY
COUT1       IN      A,(5)       GET USART STATUS BYTE
            AND     1           MASK OUT ALL BITS BUT XMIT READY
            JP      Z,COUT1     IF ZERO, WAIT FOR READY
            LD      A,B         GET CHARACTER TO BE OUTPUT
            OUT     (4)         SEND IT
            RET
```

Release 5.2 Output Status Routine

```
OST1      IN      A,6           GET THE MOTHERBOARD STATUS BYTE
          CPL                   INVERT THE BIT FOR PROPER LOGIC
          AND     10H           MASK OUT ALL BITS BUT CARRIER DET.
          PUSH    BC            SAVE THE B REGISTER
          LD      B,A           SAVE CARRIER DET. BIT IN B REG.
          IN      A,5           GET USART STATUS BYTE
          CPL                   INVERT THE BIT FOR PROPER LOGIC
          AND     1             MASK OUT ALL BUT XMIT READY BIT
          OR       B             ADD IN CARRIER DET. BIT, SET Z
FLAG
          LD      A,1           RETURN A 1 FOR DEVICE NUMBER
          POP     BC            RESTORE B REGISTER
          RET
```

CLIP TIPS

CONVERT DECIMAL INTEGERS TO BINARY FORMAT
BY W.P.MASON
(408) 998-1060

```
100 INPUT "NUMBER TO BE CONVERTED TO BINARY ",A
110 IF A=0 THEN END
120 !
130 !"DECIMAL",TAB(22),"BINARY"
140 !:! A,
150 GOSUB 190
160 !
170 GOTO 90
180 END
190 L=8388608
200 IF A>=L THEN 270
210 !TAB(12),"0",
220 L=L/2
230 IF L<1 THEN 260
240 GOTO 200
250 !
260 RETURN
270 !TAB(12),"1",
280 A=A-L
290 GOTO 220
```

MOVE SOLOS

For Processor Tech Sol owners

Moving SOLOS to F000 Hex

by Joe Maguire

Having turned to CP/M out of desperation to find a macro assembler, the next problem I ran into was not enough memory in my Sol 20. Not that I didn't have it available, I did. But the SOLOS operating system, at C000 hex, sat smack in the middle (or so it seemed) rendering the top 16K unusable. CP/M, unlike PTDOS, requires a contiguous memory block. I decided something had to give. I am not about to be bested by a box full of ICs (even if it is my beloved Sol) so with apologies to Lee Felsenstein, (I considered his modification kit too expensive) I dug into the circuit diagrams to see how I could relocate SOLOS, its RAM and the video RAM to a higher address. The total memory required for SOLOS and the VDM is 4K. That made the highest permissible address F000 hex.

The hardware modification turned out to be absurdly simple; one jumper! The software relocation may be a stumbling block for some but it is not out of reach of the well equipped hobbiest or as a computer club project. What is needed are a pair of 2708 EPROMs, or a single 2716, with SOLOS reassembled to address F000 hex and burned into them. Then they must be installed in the personality module in place of the original 9216 masked ROM. As luck would have it, there are a few unused contacts on the personality module connector so, with just one additional jumper, the switch from SOLOS at C000 to F000 and back again can be accomplished entirely by just exchanging modules. Spare module boards (for 2708s) have been advertized from time to time in various publications including the Proteus Newsletter and Kilobaud/Microcomputing.

I should emphasize that the relocation moves everything up to F000. The 4K block from C000 to CFFF is free for other use. This can add up to 12K additional memory for use by CP/M. (C000-EFFF) Some readers are no doubt wondering about software compatibility but with the quick change capability, this is not really a problem.

The source code for SOLOS is available from the CP/M or Proteus libraries. You can also copy it from the Sol manual if you don't mind all that typing but the original source code contains several absolute address references which can cause problems unless they are changed. See table 1. for corrections.

CONTINUED ON PAGE 12

HEX MATH

```
10000REM Latest revision 2-23-81, by Steven Inness
10010REM 3200 Marlene Ave., Redding, CA  96002. (916) 221-1795.
10020REM I hope these hex-dec/dec-hex routines are the best in
10030REM the land. These are numbered in the 10000's so they
10040REM can be directly appended onto existing programs.
10050DIMH$(53)
10060INPUT"Hex number? ",H$:IFH$=""THEN10070:H=FND(H$):IFH>=0THEN!H:GOTO10060
10070INPUT"Decimal number? ",H:IFH=0THEN10060
10080IFH<>INT(H)THEN!"WARNING: Non-integer, truncated"
10090!FNH$(H,0):GOTO10070
10100REM**********DEFINE DECIMAL TO HEX CONVERSION FUNCTION**********
10110REM If the number is too large to print with the digit
10120REM specification given, the first character will be non-hex.
10130REM If the digit specification is zero the number will be
10140REM printed with exactly the number of digits it needs.
10150DEFFNH$(H,D):REM FNH$(<Dec. Number>, <Digits>). USES H$, I0, N0.
10160H$="":IFH<0THEN!"-",:H=ABS(INT(H))+.4
10170FORN0=D+(INT(LOG(H)/LOG(16))+1+(H<1))*(D=0)-1TO0STEP-1
10180I0=INT(H/16∧N0):H$=H$+CHR$(I0+48+7*(I0>9)):H=H-I0*16∧N0:NEXT
10190RETURNH$
10200FNEND
10210REM**********DEFINE HEX TO DECIMAL CONVERSION FUNCTION**********
10220REM Accurately converts numbers up to 5F5E0CFH.
10230DEFFND(H$):REM USES A, H$, P, S, V.
10240IFH$=""THEN10310
10250S=0
10260FORA=1TOLEN(H$)
10270P=ASC(H$(A)):IFP>47ANDP<58THEN10280:IFP<65ORP>70THENEXIT10310:P=P-7
10280S=S*16+P-48
10290NEXTA
10300RETURNS
10310!"Not hex!":RETURN-1
10320FNEND
```

MOVE SOLOS

CONTINUED FROM PAGE 11

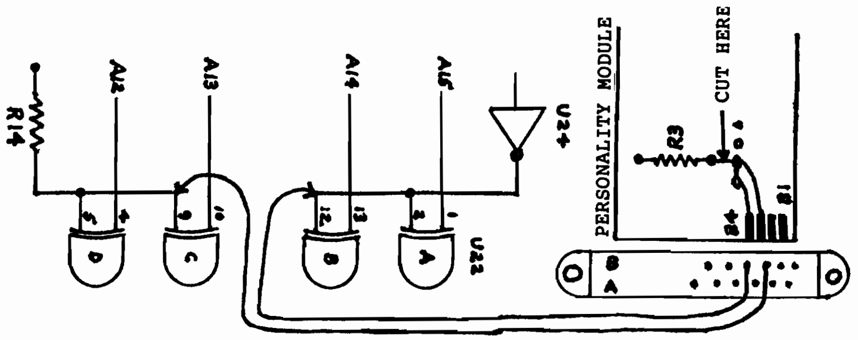
EDITOR'S NOTE
Joe McGuire advises that he has located some personality module boards for the SOL and is willing to put together relocated versions for interesed users at cost. A labor of love he says. He figures it will run about $25. If your interested write him now, so he can commit himself to getting these boards in his hands before they disappear. You can reach Joe McGuire at PO Box 3742 DT, Anchorage, AK. 99510

As you may know, a commercially available relocated module that moves SOLOS and VDM to F000 can be ordered from Metron Computerware Inc.,552 West 114 St, New York,NY 10015 Tel (212) 666 2400. The price is $129 plus $3 postage.

Robert Hogg the designer of the North Star compatible controller reviewed in this issue is also selling a relocated SOLOS at F000. His however is enhanced or altered, depending how you look at it. He has eliminated the tape reading and writing routines and substituted a full monitor that includes: Display memory in ASCII, Input ASCII string, Fill memory with a value, Hex to Decimal conversion, Move memory block, Compare memory block, Search memory for hex, Search memory for ASCII, Test memory, and command jumps to 2A00, 2A04, 2A14, 0000, and 2028. His price is around $100

Original source code		Modified code		
CØDF FE DØ	CPI ØDØH	FE ØØ	CPI Ø	End of Screen
CØE1 DA DB CØ	JC ERAS1	C2 DB FØ	JNZ ERAS1	Keep blanking
VDMEM EQU	ØCCØØH	VDMEM EQU	START+ØCØØH	Screen Adr

Table 1. Modifications to SOLOS software before reassembly to FØØØH.



PM SOCKET XRAY VIEW

1. For permanent change of SOLOS to FØØØØH, jumper pin 9 to pin 12 U22.

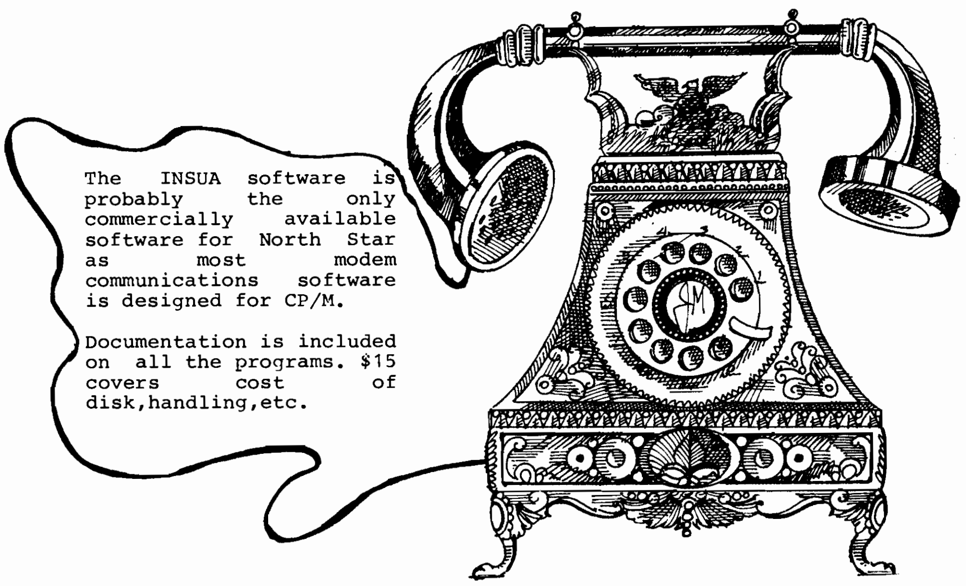
2. For change by plug in of PM; on solder side of Sol PC, run jumpers as shown from pins 9 and 12 of U22 to pins B3 and B4 on PM socket.

3. For FØØØH PM; on component side of PM cut trace as shown next to R3.

4. For CØØØH PM; on component side of PM cut trace as shown next to R3 and on solder side of PM cut connection between B3 and B4.

SOLOS modification to FØØØH

MODEM



The INSUA software is probably the only commercially available software for North Star as most modem communications software is designed for CP/M.

Documentation is included on all the programs. $15 covers cost of disk,handling,etc.

Radio Shack is selling a cheap terminal for the CompuServe network. Atari now has a promotional tele comunications tie up. The Readers Digest will sell Terminals from France for its SOURCE operation....and INSUA has a complete comunications software operating package for the North Star disc operating system.

The INSUA disk includes software marketed by Leonard Garcia and Pavel Breder. It has been donated to INSUA and is available to members for $15,,the cost of media, duplicating, postage and handling,

It's a complete tele communications management package with the following programs:

MODEM5--Pavel Breder's streamlined modem program that allows you to send files direct from your disk or recieve files from other computers and save them to disk. All automatic.

You can communicate with others operating under CPM or any time-share data network like the Source. The program is furnished in two versions like North Star's 5.2 DOS and is similarly relocatable to run anywhere in memory. It uses 6 blocks on the disk and works with double or single density systems.

It can be configured to operate with D.C.Hayes or PMMI S100 boards as well as acustic and direct connect modems

TELESTAR & TELESTAR2--Leonard Garcia's communications programs operating at 2A00 and 2D00 (files must however be in single density). Separate routines permit accessing computer networks and saving text recieved or transfering files to and from disk.

There are patches for D.C.Hayes and PMMI though it boots up running for serial port modems.

TELEREAD--A basic program to read ASCII files from modem transmission.

TELEWRIT--A stripped down editor to create ASCII files on your North Star discs. Machine language program loads at 2D00 and runs with single or double density.

TELESHARE--Machine language program to take control of another computer via modem and provide for full operation of the remote device. A number of versions of the program are furnished, to use with computers of various memory sizes.

ASCII--A machine language program to convert North Star's packed basic programs into ASCII files ala Microsoft.

we've been fighting for you

&

ALL NorthStar systems users:

JOIN
YOUR USERS
ASSOCIATION
TODAY

1.A source of information for new as well as advanced users of North Star disk systems

2.A source of programs and utilites from a central software exchange library

3.A quarterly newsletter of practical techinques and application programs designed specificly for your North Star system

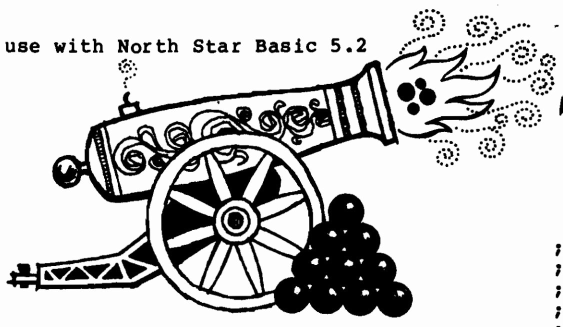
4.A liason between local users groups to promote the exchange of information usefull to all North Star users

5.An independent non-profit users representive providing feedback to North Star Computers inc.

INTERNATIONAL NORTH STAR USERS ASSOCIATION
PUBLISHERS OF THE COMPASS NEWSLETTER
POST OFFICE BOX 1318, ANTIOCH, CA. 94509

DUES ARE $20 PER YEAR

A patch for use with North Star Basic 5.2



SUPER BASIC

written by:

Stephen Maguire
P.O. Box 3742 DT
Anchorage, AK 9951Ø

February, 1981

Souping up North Star Software

by Stephen and Joe Maguire
P.O. Box 3742 DT
Anchorage, AK 9951Ø

North Star BASIC

It's a rare piece of software that can't stand improvement and North Star's Basic is no exception. It was written almost five years ago when North Star was very small, (Kentucky Fried Computers, in fact!) and the micro computer field consisted of only a few hardy pioneers. Now, new, sophisticated hardware has grown up around the software and North Star's products are used worldwide.

Some of that new hardware is video terminals and printers capable of graphics and extended character sets. Those early writers could hardly have imagined some of the uses to which Basic is now being put. What follows is a listing of some patches to Basic which will allow it to be used with graphics and other characters that use the eighth bit of the character set plus some other enhancements which are described below.

High Bit Capability

Basic could print a character which used the high bit but could not list it. The list routine converted them all to keywords. This conversion also occured in Remark and Data statements and during editing functions. The patch takes care of this problem. Remember: if you want to enter a character into the program which has the high bit set (reverse video, for example) you must be certain that the ANI 7FH line of code in the DOS Input routine is deleted. (It strips off the high bit)

Standard Syntax

I think North Star is the last holdout against using PEEK and POKE. You needn't be however. Merely by changing the keywords in the syntax table you can have them. The internal token (the byte value to which the keyword is converted) is not changed however. This means that you can use PEEK and POKE all you like in your programs but if you give a copy to your friend who is using an unmodified Basic, he will see EXAM and FILL just as always. Along the same line is the use of the backslash as the statement separator. The standard among other Basics is the colon.

CONTINUED ON PAGE 15

```
                  ;
                  ; 1.)  All characters on the line following
                  ;      a REMARK statement now remain unchanged.
                  ;      Characters with the high bit set are no
                  ;      longer converted to keywords.
                  ;
                  ; 2.)  The PRINT statement has been fixed so
                  ;      that characters with the high bit set
                  ;      are no longer converted to keywords
                  ;      when the program is LISTed.
                  ;
                  ; 3.)  The statement separator "\" has been
                  ;      replaced by a colon ":".  The internal
                  ;      token is not changed.
                  ;
                  ; 4.)  An additional formatting character is
                  ;      now allowed for printing numeric
                  ;      values.
                  ;
                  ; 5.)  Line numbers are now LISTed in a right
                  ;      justified format.  The internal pro-
                  ;      gram storage remains unchanged.
                  ;
                  ; 6.)  FOR NEXT loops are indented for better
                  ;      appearance and readability.
                  ;
                  ; 7.)  EXAM and FILL are changed to PEEK and
                  ;      POKE which have now become standard
                  ;      Basic syntax.
                  ;
                  ; Note:  These changes, with the exception of the
                  ; special formatting character, are all local to
                  ; the patched copy of Basic.  Programs created
                  ; using these patches will all produce normal
                  ; output and run correctly on an unpatched copy.
                  ; If a non standard format character is used in
                  ; the PRINT list, it will flag as a SYNTAX ERROR
                  ; on an unpatched copy of Basic.
                  ;
ØEØØ =            ENTRY:  EQU     ØEØØH  ;For Basic 5.2 as supplied
                  ;                       on factory master disk.
                  ;                       You may change this as
                  ;                       required.
                  ;
ØEØ6              ENDBAS: ORG     ENTRY+6  ;Reset new start of
ØEØ6 3Ø43                 DW      PSTART   ;program
                  ;
ØE19                      ORG     ENTRY+19H
                  ;
ØE19 3Ø43         ATN:    DW      PSTART  ;Disallow deletion of
ØE1B 3Ø43         SIN:    DW      PSTART  ;ATN, SIN, LOG, etc.
ØE1D 3Ø43         LOG:    DW      PSTART  ;This patch will
ØE1F 3Ø43         EXP:    DW      PSTART  ;follow those.
                  ;
1274                      ORG     ENTRY+474H  ;REM input change
1274 C21D42               JNZ     REMIN
                  ;
12C3                      ORG     ENTRY+4C3H  ;Line reformat
12C3 C36242               JMP     LINEF
                  ;
12D8                      ORG     ENTRY+4D8H  ;Statement separator
12D8 C33A42               JMP     CHGSS
                  ;
12FØ                      ORG     ENTRY+4FØH  ;REM LISTing
12FØ C32F42               JMP     REMOT
                  ;
151C                      ORG     ENTRY+71CH  ;EDIT reformat
151C C3F842               JMP     EDIT
                  ;
163D                      ORG     ENTRY+83DH  ;LIST reformat
163D C3ØB43               JMP     LIST
                  ;
1DØF                      ORG     ENTRY+ØFØFH ;Reset default
1DØF CA2643               JZ      FSTOR       ;format to $
                  ;
1D16                      ORG     ENTRY+ØF16H ;Numeric formatter
1D16 C21D43               JNZ     FORMAT
```

SUPER BASIC

CONTINUED FROM PAGE 14

REMark Statement

The Remark statement is a do-nothing statement. All computer languages have them. But North Star Basic proceeds to convert all "illegal" characters in the line anyway. What a waste and annoyance. Try printing a colon after a Remark - you can't. The patch takes care of that.

Output Editing

When North Star Basic was first written memory was expensive. About all most hobbiests could afford was 8K. If you had 16K you were really loaded! As a result, Basic was written to conserve memory. You could type in everything all squashed together and Basic would accept it. Just to prove to you that Basic was really trying help you along in this regard, it printed out everything all squashed together just as you typed it in. Processor Tech's Basic had a very nice output formatting routine which would unsquash the line for you and didn't use any additional memory. Our patch puts that routine into NS Basic for you. Take a look at this example:

```
OLD:      1ØFOR I=1 TO 1Ø
          5ØØPRINT I
          1333ØNEXT I

NEW:        1Ø FOR I=1 TO 1Ø
           5ØØ    PRINT I
         1333Ø  NEXT I
```

Now, which do you prefer?

Numeric Formatter

Have you ever used the dollar sign in front of a monetry value? Of course you have. Now imagine yourself living in England or Japan or anyplace else but the USA and you want to print a monetary value preceeded by the local currency sign. With NS Basic you can't. They flat forgot about the rest of the world! Ah, but maybe you are thinking that on printers made for other countries, the dollar sign is replaced by the local sign. It isn't. The US$ is used throughout the world and every printer has it assigned to code 24H. In Japan the Yen sign is assigned 5CH and in England the Pound is assigned 23H. Our patch lets you print anything you want in front of a numeric value. One thing our patch doesn't do yet and that is let you print multiple characters. West Germany, for example, uses DM for their monetary sign. Maybe North Star will get around to fixing this in a future release CONTINUED ON PAGE 16

```
                        ;
                        ;  Change EXAM and FILL to PEEK and POKE.
                        ;  Only the syntax is changed.  The internal
                        ;  tokens are not changed.
                        ;
2ØB5                    FILL:   ORG     ENTRY+12B5H  ;Keyword entry
2ØB5 5Ø4F4B45                   DB      'POKE'
                        ;
21EF                    EXAM:   ORG     ENTRY+13EFH  ;Keyword entry
21EF 5Ø45454B                   DB      'PEEK'
                        ;
                        ;  Fix the REM statement.  It changes the
                        ;  ":","[","]" and ";" when it shouldn't.
                        ;
421D                    PATCH:  ORG     ENTRY+341DH  ;Start here
                        ;
11D9 =                  CONVT:  EQU     ENTRY+3D9H   ;Convert routine
                        ;
421D FE8F               REMIN:  CPI     8FH        ;Token for REMARK
421F C2D911                     JNZ     CONVT      ;If not REM, go back
                        ;
4222 7E                 RM1:    MOV     A,M        ;Get character in line
4223 FEØD                       CPI     ØDH        ;Is it a C/R?
4225 CAD911                     JZ      CONVT      ;If so we're done
4228 12                         STAX    D          ;Store the char
4229 13                         INX     D          ;Bump pointers
422A ØC                         INR     C
422B 23                         INX     H
422C C32242                     JMP     RM1        ;Loop to end of line
                        ;
                        ;  Fix the REM so that when LISTed it does
                        ;  not convert bytes with the high bit set
                        ;  to keywords.  This will allow graphic
                        ;  symbols, special characters, etc. to be
                        ;  LISTed correctly.
                        ;
12C8 =                  CVNUM:  EQU     ENTRY+4C8H  ;Numeric conversion
                        ;
422F FE9Ø               REMOT:  CPI     9ØH        ;Check REM token+1
4231 C2C812                     JNZ     CVNUM      ;Go check for number
4234 C5                         PUSH    B
4235 ØEØD                       MVI     C,ØDH      ;Set up line to
4237 C34F42                     JMP     PR1        ;bypass conversion
                        ;
423A FE5C               CHGSS:  CPI     5CH        ;Is it a "\"?
423C C24742                     JNZ     PRINT      ;If not, check next
423F 1B                         DCX     D          ;Back up pointer
424Ø 3E3A                       MVI     A,':'      ;Move in colon
4242 12                         STAX    D          ;Store it
4243 13                         INX     D          ;Bump pointer
4244 C3C812                     JMP     CVNUM      ;Go to numeric
                        ;
4247 FE22               PRINT:  CPI     22H        ;No conversion if quote
4249 C2C812                     JNZ     CVNUM
424C C5                         PUSH    B
424D ØE22                       MVI     C,22H      ;Set quote as end char
424F 23                 PR1:    INX     H          ;Bump pointer
425Ø 7E                         MOV     A,M        ;Get character
4251 12                         STAX    D          ;Store it
4252 FEØD                       CPI     ØDH        ;See if end of line
4254 CA6Ø42                     JZ      PR2
4257 13                         INX     D          ;Bump pointer
4258 B9                         CMP     C          ;Check for end char
4259 C24F42                     JNZ     PR1        ;Keep looping if not
425C C1                         POP     B          ;Clean up
425D C3C812                     JMP     CVNUM      ;Go to numeric
                        ;
426Ø C1                 PR2:    POP     B          ;Clean up
4261 C9                         RET                ;Exit
                        ;
                        ;  The following routine edits the line
                        ;  during output to improve appearance
                        ;  and readability.  It requires no
                        ;  additional memory for program storage.
                        ;
12CA =                  CKNUM:  EQU     ENTRY+4CAH  ;Check for numeric
12BF =                  LNCVT:  EQU     ENTRY+4BFH  ;Convert line
151F =                  PEDIT:  EQU     ENTRY+71FH  ;Edit routine
164Ø =                  SCVAL:  EQU     ENTRY+84ØH  ;Get scroll value
298D =                  SCBLK:  EQU     ENTRY+1B8DH  ;Scan off blanks
2DØ6 =                  ASCLN:  EQU     ENTRY+1FØ6H  ;Convert line nr.
2FØ7 =                  PLIST:  EQU     ENTRY+21Ø7H  ;List routine
                        ;
4262 AF                 LINEF:  XRA     A          ;Set up counters
4263 322C43                     STA     FORS       ;Store initial value
4266 D5                         PUSH    D          ;Save pointer
4267 CDØ62D                     CALL    ASCLN      ;Convert line nr.
426A 3A2F43                     LDA     VALUE      ;Get count
426D BB                         CMP     E          ;Compare with
426E CA7A42                     JZ      LF1        ;buffer pointer
```

SUPER BASIC

CONTINUED FROM PAGE 15

Patching Basic

These patches go in the memory at the end of the present Basic code. The pointer ENDBAS is adjusted upward so that user programs will not overwrite them. There is enough space left in the standard disk allocation of 54 blocks (DD) to hold Basic and the patch. If you want to patch FPBasic or Basic of more than the standard precision, you must find the correct addresses for the various routines. The ones given in the listing are only correct for 8 digit precision Basic, release 5.2.

```
4271 D1                     POP     D         ;Restore pointer
4272 3E2Ø                   MVI     A,' '     ;Insert a space
4274 12                     STAX    D
4275 13                     INX     D         ;Bump pointer
4276 E1                     POP     H         ;Get back HL
4277 C3BF12                 JMP     LNCVT     ;Convert rest of line
                    ;
427A F1             LF1:    POP     PSW       ;Adjust stack
427B 3E2Ø                   MVI     A,' '     ;Insert space
427D 12                     STAX    D
427E 13                     INX     D         ;Bump pointer
427F E1                     POP     H         ;Get back pointer
428Ø E5                     PUSH    H         ;Save it again
                    ;
4281 23             LF2:    INX     H         ;Bump it up
4282 23                     INX     H
                    ;
4283 7E             LF3:    MOV     A,M       ;Get character
4284 23                     INX     H         ;Bump pointer
4285 FE9A                   CPI     9AH       ;Is it numeric?
4287 CA8142                 JZ      LF2       ;No format if so
428A FEØD                   CPI     ØDH       ;C/R?
428C CAD342                 JZ      LF7       ;Go format
428F FE8F                   CPI     8FH       ;REM gets it too
4291 CAD342                 JZ      LF7
4294 FE81                   CPI     81H       ;FOR statement?
4296 C2A742                 JNZ     LF4
                    ;
4299 3A2E43                 LDA     NXTIND    ;This starts FOR
429C 3C                     INR     A         ;NEXT loop format
429D 322E43                 STA     NXTIND
42AØ 3A2C43                 LDA     FORS      ;Increment count
42A3 3C                     INR     A
42A4 322C43                 STA     FORS
                    ;
42A7 FE83           LF4:    CPI     83H       ;NEXT statement?
42A9 C28342                 JNZ     LF3
42AC 3A2C43                 LDA     FORS
42AF B7                     ORA     A
42BØ CABA42                 JZ      LF5
42B3 3D                     DCR     A         ;This cancels
42B4 322C43                 STA     FORS      ;indentation
42B7 C3C542                 JMP     LF6
                    ;
42BA 3A2D43         LF5:    LDA     LPIND     ;All this checking
42BD B7                     ORA     A         ;allows for
42BE CAC542                 JZ      LF6       ;nested loops
42C1 3D                     DCR     A
42C2 322D43                 STA     LPIND
                    ;
42C5 3A2E43         LF6:    LDA     NXTIND
42C8 B7                     ORA     A
42C9 CA8342                 JZ      LF3
42CC 3D                     DCR     A
42CD 322E43                 STA     NXTIND
42DØ C38342                 JMP     LF3
                    ;
42D3 3A2D43         LF7:    LDA     LPIND
42D6 B7             LF8:    ORA     A
42D7 CAE642                 JZ      LF9
42DA F5                     PUSH    PSW       ;Time to insert
42DB 3E2Ø                   MVI     A,' '     ;some spaces
42DD 12                     STAX    D
42DE 13                     INX     D
42DF 12                     STAX    D         ;Each loop gets
42EØ 13                     INX     D         ;two spaces
42E1 F1                     POP     PSW
42E2 3D                     DCR     A
42E3 C3D642                 JMP     LF8
                    ;
42E6 3A2E43         LF9:    LDA     NXTIND
42E9 322D43                 STA     LPIND
42EC E1                     POP     H         ;Have to adjust
42ED 23                     INX     H         ;buffer pointer
42EE 23             LF1Ø:   INX     H
42EF 7E                     MOV     A,M
42FØ FE2Ø                   CPI     ' '
42F2 CAEE42                 JZ      LF1Ø
42F5 C3CA12                 JMP     CKNUM     ;All done
```

CONTINUED ON PAGE 17

CLIP TIPS

Lance E. Rose
P.O. Box 1082
Kalispell, MT 59901

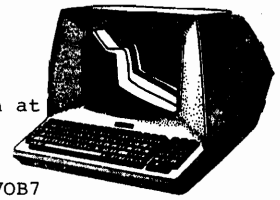
BASIC KEYWORDS

For those who want to, it is possible to change the list of BASIC keywords to conform more to other BASICS. Some candidates for this are EXAM(=PEEK), FILL(=POKE) and SQRT(=SQR). All that is required is to alter the list of tokens and keywords found in the BASIC interpreter. It is found at different addresses in different versions of BASIC but is easily recognizable by the ASCII keywords, each of which follows the corresponding token. The beginning of the list looks like

80H,'LET',81H,'FOR',........

and the end of the list is marked by a single 255 byte (OFFH). Within the list, BASIC doesn't care what the actual keywords are as long as they are matched up with the correct tokens. One of the monitors can be used to locate and then alter this list to customize one's keywords. The only limitations are that the new list cannot exceed the length of the old one, and no gaps are permitted in the list, i.e. the list must be token, keyword, token, keyword, Otherwise any ASCII characters may be used to make up the keywords. For example, the single character "'" can be used in place of the REM keyword. Even the mathematical operators can be altered to use different symbols. Just watch for confusion if you start dabbling in this area. Good luck.

CLIP TIPS



HARDWARE NOTE: Run Your Horizon at 19,200 baud

BY Bob Stek
19 Mayfield Rd
Regina, Seshatchewan S4VOB7

Here is a way to operate your Horizon serial ports at 19.2 Kb instead of its current 9600 baud. The basic idea is very straight forward; merely double the clock speed going to the USARTS. A stock Horizon uses a 2 Mhz clock to feed the USARTS. But the 4 Mhz clock used by the CPU is available if you are willing to make a minor, easily reversible change. This tip was sent to me by Dr. Jim Byram of Massachusetts.

The timing signal will come from the CPU board itself; it could come from the motherboard, but that would require removing the motherboard from the chassis. This alternative is much simpler to do and undo. The CPU card should be placed in the last bus slot. Standing in front of the Horizon, look over the top of the CPU card at its solder side. Directly below IC 6G (74LS175) are three unused plated-through holes. They are aligned between pins 3 and 4, 5 and 6, and 7 and 8 of IC 3G. Note that a trace runs from the rightmost of these holes to pin 2 of IC 3G (74LS08). Solder a molex pin socket into this hole on the solder side of the CPU card for convenience.

Take a 14 pin dip header (like those used for configuring the I/O interfaces or the real time clock) and solder a 14 pin socket on top of the header. Solder all socket pins to the top of the header your CPU card. Remove IC 7C (74LS14) from the motherboard, plug your socket-dip header sandwich into the socket 7C, and plug the 74LS14 into the socket on the sandwich. That's it. The baud rates will now be double those given in the Horizon manual when the baud rate header is configured.

The essence of the modification seems to be that pin 11 of the LS14 now gets a 4 Mhz clock signal (conveniently lifted from the CPU card) rather than the 2 Mhz signal it received previously.

When the baud rate header is set to 9600 baud for serial port #1 (standard configuration - header pins 3, 4, and 16 connected), the rate will actually be 19,200. If you are running a printer at 1200 baud on serial port #2, the header must now be set for 600 (header pins 5, 6, and 12 connected).

To undo this modification, simply disconnect the jumper from the CPU card, remove the socket-header sandwich, and plug the LS14 back into its socket (7C) (and redo the baud rate header if necessary).

SUPER BASIC

CONTINUED FROM PAGE 16

```
                    ;
                    ;  The line to be LISTed, or EDITed,
                    ;  is reformatted in the respective
                    ;  buffer.  The byte stored in VALUE
                    ;  is the low byte address of the
                    ;  buffer pointer where the reformat-
                    ;  ted line will start.  This is for
                    ;  right justification of the line number.
                    ;
3C00 =              EDBUF:  EQU     ENTRY+2E00H  ;Edit buffer
3CA8 =              LIBUF:  EQU     ENTRY+2EA8H  ;List buffer
                    ;
42F8 AF             EDIT:   XRA     A         ;No indent here
42F9 322D43                 STA     LPIND
42FC 322E43                 STA     NXTIND
42FF 3E07                   MVI     A,EDBUF+7 AND 0FFH
4301 322F43                 STA     VALUE
4304 F1                     POP     PSW
4305 CD8D29                 CALL    SCBLK     ;Scan past blanks
4308 C31F15                 JMP     PEDIT     ;Return to EDIT
                    ;
430B AF             LIST:   XRA     A         ;No indent yet
430C 322D43                 STA     LPIND
430F 322E43                 STA     NXTIND
4312 3EAF                   MVI     A,LIBUF+7 AND 0FFH
4314 322F43                 STA     VALUE
4317 CD072F                 CALL    PLIST     ;List the line
431A C34016                 JMP     SCVAL     ;Do the next
                    ;
                    ;  This routine will print any character
                    ;  of your choice preceeding a numeric
                    ;  value.  It checks for the flag character
                    ;  in the print list and substitutes your
                    ;  format sign in place of the dollar sign.
                    ;  The syntax of the formatted PRINT
                    ;  statement should agree with the rules
                    ;  given in the North Star Software manual.
                    ;  The $ sign is still available for use.
                    ;
                    ;  Enter your choice of flag character after
                    ;  the FCHAR  EQU, and the format sign
                    ;  after the FSIGN  EQU.
                    ;
                    ;  In this example, the Japanese Yen sign
                    ;  is printed after detecting a "Y" in the
                    ;  print list.
                    ;
0059 =              FCHAR:  EQU     'Y'       ;This will be a Yen sign
005C =              FSIGN:  EQU     5CH       ;on Japanese printers
                    ;
1D60 =              NPRIN:  EQU     ENTRY+0F60H     ;Numeric print
1D19 =              FPRIN:  EQU     ENTRY+0F19H     ;Format print
33D5 =              DCHAR:  EQU     ENTRY+25D5H     ;$ stored here
                    ;
431D FE59           FORMAT: CPI     FCHAR     ;Is it a special format?
431F C2601D                 JNZ     NPRIN     ;No, go print the number
4322 3E5C                   MVI     A,FSIGN   ;Move in format sign
4324 0601                   MVI     B,1
4326 32D533         FSTOR:  STA     DCHAR     ;Put in place of $
4329 C3191D                 JMP     FPRIN     ;Go to format print
                    ;
432C 00             FORS:   DB      0         ;FOR/NEXT space count
432D 00             LPIND:  DB      0         ;Line ptr. indent
432E 00             NXTIND: DB      0         ;NEXT indent value
432F 00             VALUE:  DB      0         ;New buffer addr
                    ;
4330                PSTART: END     ;Program start addr
```

DISKLIST

BY: Leonard Morgenstern

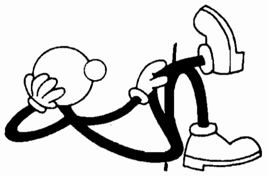
DISKLIST is a file utility that will catalog your North Star disk files.

Diskettes are inserted one at a time. The program reads the disk directory into a buffer. Each entry is converted into a 12-byte string ("word") that includes the file name, type, and name of disk. These words are concatenated into a single long string that is sorted prior to each listing.

Listing is done in three columns. Each directory is listed in the first column as it is entered. Then you reposition the paper to the top, and the program lists the entries, sorted alphabetically in the second column. This is repeated for a third listing by file type in the third column.

REQUIREMENTS
The program assumes double density North Star DOS at 2000 Hex. The number of entries that can be handled by the program is proportional to the amount of free memory available.

The program sets up a buffer 811 bytes long (hex) for holding disk directory and subroutines. This can be anywhere in memory. If it can be located in an area not normally used by BASIC (for example, at 0), the maximum limit of directory entries is increased. Otherwise, the buffer must be put at the end of the BASIC work area by using MEMSET, as explained below, in the section on INITIALIZATION.



UNUSUAL FEATURES
The program executes the DOS commands DCOM and LIST by means of function calls. In each case, the function sets up a subroutine in memory. It accomplishes this by READing DATA statements, inserting bytes into memory at a location L, then CALLing L. In the case of the DCOM call, the return value is 0 if the call was successful, and 1 if not.

The program uses an alphabetical sort that I originally obtained from NSUG. I have placed it at the front of the program to improve speed.

INITIALIZATION
If your computer has memory at 0, the program is ready to run. If not, the program will calculate the correct address to use. To accomplish this, change statement 1050 from DATA 0 to DATA -1, and run the program. The program will end and print a suggested memory limit on the screen. Say it is 47086. Then change statement 1050 and set memory limits as follows

```
40 !"THIS PROGRAM WILL COMBINE N* DISK DIRECTORIES INTO"
50 !"ONE DIRECTORY, SORT AND LIST 3 WAYS"
51 !"WRITTEN FOR DOUBLE DENSITY N* DOS AT 2000H"
52!"BY LEONARD MORGENSTERN 1980"
60 !                                   PRINT OUTPUT DEVICE
70 INPUT "PRINTER OPTION (0=SCREEN,3=PRINTER)",P1
80 IF P1=0  THEN 130
90 !:!"CHECK PRINTER. INSERT LOTS OF PAPER. THERE WILL BE"
100 !"THREE PASSES. PAPER SHOULD BE RESET TO BEGINNING AT"
110!"START OF EACH PASS"
120 !:INPUT "TODAY IS ",D : !#P1,"TODAY IS ",D : !
130 !"LOAD DISKETTES ONE AT A TIME. ON REQUEST, TYPE"
140 !"NAME OF DISKETTE (3 CHARACTERS MAXIMUM)"
150 DIM N$(12000),A$(21),B$(15)
160 L1=2048 : REM L1=BUFFER FOR DIRECTORIES. 2048=800H
161 L=0 : REM ADDRESS TO SET UP SHORT 8080 ROUTINES
170 N0=1 : REM N0=POINTER FOR N$ STRING
180 !:!(N0-1)/12," ENTRIES SO FAR. DO NOT EXCEED 1000"
190 !TAB(10),"CR ONLY TO START SORTING"
200 INPUT "DISKETTE NAME (MAX 3 CHARS) :",D$
210 IF LEN(D$)=0 THEN 480
220 A$="": IF P1<>3 THEN 240
230 INPUT "DESCRIPTION OF DISKETTE (MAX=20 CHAR)",A$
240 !:!"DISKETTE ",D$,"    ",A$
250 !"IS THAT CORRECT? HAVE YOU CHECKED DISK DRIVE?"
260 !"Y=YES"
270 B$=INCHAR$(0): IF B$<>"Y" THEN 180
280 IF P1<>3 THEN 310
290 !#P1:!#P1,"DISKETTE: ",D$
300 !#P1,A$:!#P1
310 ERRSET 2070,E1,E2
320 I=FNL(1,P1)
330 ERRSET 2080,E1,E2
340 I=FNC(0,4,0,L1,1,1,1)
350 ERRSET
360 IF I=0 THEN 380
370 !"DISK ERROR ",D$ : GOTO 180
380 FOR I=L1 TO L1+16*128-1 STEP 16
390 IF EXAM(I)=32 THEN 460
400 A$=""
410 FOR J=0 TO 7 : A$=A$+CHR$(EXAM(I+J)) : NEXT
420 I1=EXAM(I+12): IF I1>=128 THEN I1=I1-128
430 A$=A$+CHR$(I1)
440 D$=D$+"    ": A$=A$+D$(1,3)
450 N$(N0)=A$ : N0=N0+12
460 NEXT
470 GOTO 180
480 !"PASS 1 COMPLETE. RESTORE PAPER TO BEGINNING"
490 O=1 : GOSUB 900:REM SORT
500 !"PASS 2 COMPLETE. RESTORE PAPER TO BEGINNING"
510 FOR I=1 TO N0-1 STEP 12
520 A$=N$(I+8,I+8)+N$(I,I+7)+N$(I+9,I+11)
530 N$(I)=A$
540 NEXT
550 O=2 : GOSUB 900:REM SORT
560 !"JOB COMPLETE"
570 END
580 REM THIS SUBROUTINE WILL SET UP A DCOM COMMAND
590 REM AT LOCATION L
600 REM
610 DEFFNC(L,S,D,M,A,D1,U)
620 REM L=LOCATION,S=SECTORS,D=DISK ADDR,M=MEM ADDR,
630 REM A=ACTION TO TAKE (0=WRITE,1=READ),D1=DENSITY(1=DD),
640 REM U=DISK UNIT)
650 REM RETURNS 0=OK, 1=ERROR
660 REM
670 RESTORE 770
680 FOR I1=0 TO 16: READ I2 : FILL L+I1,I2 : NEXT
690 FILL L+1,S
700 FILL L+4,A
710 FILL L+3,128*D1+U
720 I1=FNM(D,256):I2=INT(D/256)
730 FILL L+6,I1
740 FILL L+7,I2
750 RETURN CALL(L,M)
760 FNEND
770 DATA 62,0 : REM MVI A,0
780 DATA 1,0,0 :REM LXI B,0
790 DATA 33,0,0 : REM LXI H,0
800 DATA 205,34,32 : REM CALL 2022
810 DATA 33,0,0 : REM LXI H,0
820 DATA 208,35,201 :   INX H RET
830 REM  (MODULUS FUNCTION)
840 DEFFNM(X,M)=X-INT(X/M)*M
```

CONTINUED ON PAGE 19

DISKLIST

```
1050 DATA 47087
MEMSET 47086
```

Note that the value in the DATA statement is 1 plus the memory limit. In this way, the buffer is put beyond the area used by BASIC. The new memory limit is permanent,in the sense that it remains in effect until reset by another MEMSET command or BASIC is reloaded.

If your machine only has memory from 2000 to 7FFF, the program will not fit (sorry about that).

Your BASIC must have LOG.

I have tested the program on a SOL (8080) computer. I believe it will work on a Z80, but I have not tested it.

CONTINUED FROM PAGE 18

```
850 REM ******
860 REM FAST SORT, MINIMUM COMPARISON, MIN. STEPS
870 REM VARIATION OF WOODRUM SORT BY RICHARD HART
880 REM PUBLISHED IN CREATIVE COMPUTING JAN/FEB 1978
890 REM NEXT LINE SHOULD SET N=NUMBER OF WORDS,W=LENGTH
900 N=(N0-1)/12:W=12
910 Z=P1 : REM P1 IS USED DURING SORT
920 W1=W-1
930!"SORTING..."
940 REM ENTRY
950 IF O<>1 THEN 980
960 L9=INT(LOG(N)/LOG(2)+.9) : REM LINKS=N+LOG2(N)+2 ELEMENTS
970 DIM L(N+L9+2) : REM AUTO LINK ARRAY
980 FOR I=1 TO N+L9+2 : L(I)=0 : NEXT
990 I=0:K1=0:M1=0:T2=0:T4=0
1000 REM K1,I,M1,T2,T4=0
1010 LET J=N+1 : REM HEAD OF SEQ 1
1020 L(1)=1 : L(J)=1 : K2=1
1030 IF N<=1 THEN 1760 : REM EXIT, NOTHING TO SORT
1040 S1=N : REM NUMBER OF LEAVES
1050 REM CLIMB TREE
1060 IF S1 < 4 THEN 1120 : REM LOW ORDER TWIG VALUE
1070 K2=K2*2 : REM TOTAL # OF TWIGS
1080 B2=S1/2
1090 S1=INT(B2)
1100 T4=T4+(B2-S1)*K2
1110 GOTO 1050
1120 REM INITIAL CALCULATIONS
1130 T4=K2-T4 : REM # OF LO-ORDER TWIGS
1140 B2=K2/2 : REM HI-BIT VAL OF BIN CTR
1150 REM NEXT TWIG
1160 IF K1=K2 THEN 1760 : REM EXIT SORT COMPLETE
1170 T1=K1+1 : K1=K1+1 : REM TWIG #
1180 B1=B2 :REM HI BIT VALUE
1190 T3=T2 : REM PREV REFLECTED TWIG #
1200 REM ADD 1 TO REFL BIN CTR & CARRY
1210 T1=T1/2
1220 IF INT(T1)<T1 THEN 1270 :REM NO MORE CARRIES
1230 M1=M1+1 : REM # OF MERGES & CARRIES
1240 T2=T2-B1
1250 B1=B1/2 : REM NEXT BIT VALUE
1260 GOTO 1200 : REM CARRY 1
1270 REM TWIG CALCULATIONS
1280 T2=T2+B1 : REM REFLECTED TWIG #
1290 IF S1=2 THEN 1350 : REM 2-TWIGS & 3-TWIGS
1300 REM 3-TWIGS & 4-TWIGS
1310 IF T3<T4 THEN 1360 : REM LO-ORDER TWIG (3-TWIG)
1320 REM 4-TWIG
1330 M1=-M1 : REM DISENGAGE # OF MERGES
1340 GOTO 1430
1350 IF T3<T4 THEN 1410 : REM LO-ORDER TWIG (3-TWIG)
1360 REM 3-TWIG
1370 M1=M1+1 : REM # OF MERGES
1380 I=I+1 : REM NEXT LEAF
1390 L(I)=I : L(J)=I : REM GENERATE A LEAF
1400 J=J+1 : REM NEXT SEQUENCE HEAD
1410 REM 2-TWIG
1420 M1=M1+1 : REM # OF MERGES
1430 I=I+1 : REM NEXT LEAF
1440 L1=I : L(I)=I :L(J)=I : REM GENERATE A LEAF
1450 L0=J : REM HEAD OF OLDER LEAF (LAST LINE)
1460 J=J+1 : REM HEAD OF LATEST LEAF (NEXT 2 LINES)
1470 I=I+1 : REM NEXT LEAF
1480 L2=I : L(I)=I : L(J)=I : REM GENERATE A LEAF
1490 GOTO 1550 : REM MERGE LEAVES
1500 REM MERGE TWIGS & BRANCHES
1510 J=J-1 : REM HEAD OF LATEST BRANCH OR TWIG
1520 L0=J-1 : REM HEAD OF OLDER BRANCH OR TWIG
1530 L1=L(L0) : REM HEAD OF SEQUENCE 1
1540 L2=L(J) : REM HEAD OF SEQUENCE 2
1550 P1=1+(L1-1)*W : P2=1+(L2-1)*W : REM WORD POINTERS
1560 IF N$(P1,P1+W1) <= N$(P2,P2+W1) THEN 1640   : REM STAY IN 1
1570 L(L0)=L2 : REM SWITCH TO SEQ 2
1580 L0=L2 : REM TOP LEAF IN SEQ 2
1590 L2=L(L0) : REM NEXT LEAF IN SEQ 2
1600 IF L2=L0 THEN 1690 : REM END OF SEQ 2
1610 P1=1+(L1-1)*W : P2=1+(L2-1)*W : REM WORD POINTERS
1620 IF N$(P1,P1+W1) > N$(P2,P2+W1) THEN 1580 : REM STAY IN 2
1630 L(L0)=L1 : REM SWITCH TO SEQ 1
1640 L0=L1 : REM TOP LEAF IN SEQ 1
1650 L1=L(L0) :REM NEXT LEAF IN SEQ 1
1660 IF L1<>L0 THEN 1550 :REM NOT END OF SEQ 1
1670 L(L0)=L2 : REM SWITCH TO SEQ 2
1680 GOTO 1700
1690 L(L0)=L1 : REM SWITCH TO SEQ 1
1700 M1=M1-1 : REM # OF MERGES
1710 IF M1>0 THEN 1500
1720 IF M1=0 THEN 1150
1730 REM GENERATE 2ND HALF OF A 4-TWIG
1740 M1=1-M1 : REM RE-ENGAGE # OF MERGES
1750 GOTO 1430
1760 REM EXIT
1770 L0=N+1 : REM FIRST LINK IN SEQ
1780 FOR I= 1 TO N
1790 L0=L(L0) : REM FOLLOW LINKS
1800 P=1+(L0-1)*W : REM WORD POINTER
1810 GOSUB 1870 : REM PRINT WORD
1820 NEXT I
1830 P1=Z
1840 RETURN
1850 REM
1860 REM THIS SUB PRINTS THE ENTRY
1870 ON O GOTO 1880,1910
1880 IF Z=3 THEN !#Z,TAB(35),
1890 !#Z,N$(P,P+7),%3I,ASC(N$(P+8))," ",N$(P+9,P+11)
1900 GOTO 1930
1910 IF Z=3 THEN !#Z,TAB(60),
1920 !#Z,%3I,ASC(N$(P))," ",N$(P+1,P+8)," ",N$(P+9,P+11)
1930 RETURN
1940 REM
1950 REM THIS WILL LIST A DISK DIRECTORY
1960 REM D=DRIVE, P=PORT
1970 REM
1980 DEFFNL(D,P)
1990 RESTORE 2000
2000 DATA 235,124,195,37,32
2010 FOR I1=0 TO 4
2020 READ I2 : FILL I1,I2
2030 NEXT
2040 RETURN CALL (0,256*D+P1)
2050 FNEND
2060 REM DISK ERROR ROUTINES
2070 GOSUB 2110 : !"CHECK DISK!" : GOTO 280
2080 GOSUB 2110 : !"PROBABLE FAULTY DISK"
2090 !"TYPE COMMAND (1=CONTINUE,2=DO NOT LIST THIS DISK)"
2100 IF INCHAR$(0)<>"2"THEN 350 ELSE 180
2110 !"***DISK ERROR ",E1,E2 : RETURN
2120 REM
2130 REM THIS SUBROUTINE LISTS N$
2140 !"NUMBER OF CHARS=",N0
2150 FOR I1=1 TO N0-1
2160 A$=N$(I1,I1)
2170 IF ASC(A$)<32 THEN !"",ASC(A$),"", ELSE !A$,
2180 NEXT
2190 !" "
2200 RETURN
```

HEAR NORTHSTAR'S GOT A NEW CPU CHIP
THAT'S WHAT THEY SAID IN MY YOUTH TOO

QUICKKEY

* ROUTINE TO WRITE NS BASIC

* PROGRAMS IN SHORTHAND BY

* USING PRE-STORED TEXT AND

* STATEMENTS IN HIGH MEMORY.

BY Leonard Garcia

QUICK-KEY is an 8080 assembly language program that generates North Star BASIC comands and statements from pre-stored text in memory by using a two key code.

The program also allows a user to enter and define any text that will be sent to the BASIC interpreter (or of any other system program) upon hitting a special sequence of keys.

The text is stored in blocks of 32 characters (except 64 characters for the version assembled at 0000H), and the statements or text are accessed by hitting the sequence:

Esc x

where Esc is the "ESCAPE" keyboard key and x is a letter key in the range A thru Z.

Once the QUICK-KEY program is executed in high memory, it will locate the address of the user's Character-In routine in the DOS I/O area, patch it into it's own, re-route the Character-In calls to itself and then returns to the Disk Operating System.

The program will then scan all keyboard entries for the Esc x code to send a text to whatever program called the Character-In routines for keyboard data.

Thus, the sequence: Esc A: will send the statement: ABS to the BASIC interpreter, and display on the screen as it is being echoed by the system. A second Esc will cause the first entry, ABS, to be rubbed out and the next statement of the "A" series to be sent:APPEND. Continuing to hit the Esc key will send the next command or statement that starts with "A", deleting the last that was sent. Once a space or another character other than Esc is entered, the key code is then voided and another Esc x sequence may be started to generate a different text.

The following keys do not have any corresponding BASIC statement or command and are available to the user for any custom text: H,J,Q,U,X,Y,Z.

The custom text may exceed the normal block size allowed, but the code for the next block of text will be overlaid.

The custom text may also be entered using semi-colons (;) as delimiters causing the text to be sent a word or phrase at a time as above, ABS;APPEND;etc. Several versions of the QUICK-KEY program assembled at different memory locations are included on INSUA's library disk, and the user should run the one that matches his highest memory location.

OPERATION

Once the North Star DOS is loaded, start the program with: GO QUICKxxK where xx is the total amount of memory starting at 2000H. The message: * QUICK-KEY * NORTH STAR BASIC STATEMENT GENERATOR CUSTOMIZE (C) OR EXECUTE (E) PROGRAM? will print If customizing of the text is desired, type "C" and refer to the customizing instructions below , else type "E" to execute the program or hit the CTRL-C key to return to the DOS.

Once the "E" command is entered, the message: I/O PATCHED. "MEMSET" BASIC NO HIGHER THAN xxxxx RETURNING TO DOS.. will display. The customizing routines reside in the first (lower) part of the program so it may be overlaid as required. However, since NS BASIC places temporary data at the highest address for which it has

CONTINUED ON PAGE 21

```
*  EQUATES:
*
DOS EQU 2028H  DOS RE-ENTRY POINT
CIN EQU 2010H  DOS CHAR-IN JUMP ADDR
COUT EQU 200DH  DOS CHAR-OUT JUMP
CR EQU 0DH  CARRIAGE RETURN
LF EQU 0AH  LINE FEED
ESC EQU 01BH  ESCAPE
ETX EQU 01H  END-OF-TEXT
QUICK ORG 9A00H
*
 CALL MSSG0  PRINT HERALD
START CALL MSSG1  "CUST OR EXECUTE?"
 XRA A  CLEAR A
 CALL CIN  GET REPLY
 CPI 'C'  CUSTOMIZE?
 JZ CUST
 CPI 'E'  EXECUTE?
 JZ PATCH
 CPI 03H  CTRL-C?
 JZ DOS
 JMP START  LOOP
*
* THIS ROUTINE ACCEPTS CUST TEXT
*
CUST CALL MSSG3  "FOR KEY (A-Z)?"
 LXI H,0000H
 LXI D,TEXT
 XRA A
 CALL CIN  GET ANSWER
 MOV B,A
 CALL CHOUT  ECHO
 CPI 041H  LESS THAN 'A'?
 JC CUST
 CPI 05BH  LESS THAN '('?
 JNC CUST
 SUI 40H  MULTIPLIER
 MOV L,A  LSB
 DAD H
 DAD H
 DAD H
 DAD H
 DAD H
 DAD D  ADDR+OFFSET
 SHLD ADDR  BASE ADDR
CUST2 CALL MSSG4  "ENTER TEXT"
 LHLD ADDR
ACPT XRA A
 CALL CIN
 CPI 03H  CTRL-C?
 JZ THRU
 CPI 5FH  "RUB"?
 JNZ CONT
 MVI B,08H  BACKSPC ROUTINE
 CALL CHOUT
 MVI B,020H
 CALL CHOUT
 MVI B,08H
 CALL CHOUT
 DCX H  DECR ADDR
 JMP ACPT  GET NEW CHR
* SAVE CHARACTER TO MEMORY
CONT MOV M,A  TO RAM
 MOV B,A
 CALL CHOUT  ECHO IT
 CPI CR   CR?
 JNZ CONT2
 MVI B,LF  LINE FEED
 CALL CHOUT
CONT2 INX H  INCR ADDR
 JMP ACPT  GET NEXT CHR
* TERMINATE ENTRY WITH ETX CODE
THRU MVI A,ETX
 MOV M,A
 CALL CRLF
 JMP START  TO BEGINNING
*
* THIS ROUTINE PATCHES THE DOS CHARACTER-IN
* TO CALL THIS PROGRAM'S I/O ROUTINES.
*
PATCH CALL MSSG2  "I/O PATCHED"
 LHLD CIN+1  USER'S CIN ADDR
 SHLD CHIN+2  PGM'S CHIN ADDR
 LXI H,INPUT  NEW INPUT ROUTINE ADDR
 SHLD CIN+1  PATCH IN JUMP TABLE
 MVI A,00H  SET FLAGS FALSE
 STA NEXC  FOR MORE CHARACTERS
 STA RUB  AND RUBBING FLAG
 STA COUNT  AND CHR COUNTER
 JMP DOS  PGM PATCH,RET TO DOS
```

QUICKKEY

been set, it must be "MEMSET" after it has been loaded to an address no higher than given by "xxxxx" in decimal to prevent it from overlaying the functional part of the QUICK-KEY program.

The version at 0000H, of course, needs no such procedure. After the QUICK-KEY program has returned to the DOS and the BASIC interpreter has been loaded & "MEMSET", the program will ignore all keyboard entries until an ESC x sequence. To enter a statement or text the following example may be used: The statement

100 IF A1 = 0 THEN GOTO 300 ELSE PRINT "TEST"
may be generated with the sequence:
100 Escl A1 = 0 EscT EscGEsc 300 EscE EscP "TEST"

Note that EscG will generate the text "GOSUB". Since the statement we wanted was GOTO, another Esc was entered to rub out the GOSUB and send the GOTO instead. ALSO note that in the portion "ELSE PRINT" the EscE was followed by a space BEFORE entering the EscP. If the ESC key was hit again after the EscE without an intervening character it would have rubbed out "ELSE" and sent the next statement in the "E" series, "EXIT".

Should the user wish to use spaces to indent a program, the sequence Esc (Space Bar) will send 5 blank spaces to the BASIC interpreter. Another immediate ESC will rub the first five and send 10, and a third ESC will rub the last 10 and send 15 ASCII spaces. The ESC (Space Bar) will in fact act as a tab to enter a formatted program listing.

CUSTOMIZING

To customize the QUICK-KEY program to send text, hit the "C" key in the command mode. The message: FOR KEY LETTER (A THRU Z)?will display. Enter the letter key that you wish to define the text by. Note that any text entered in this mode will overlay the text that is already under that key code, if any. After the key code is entered, the format:

------------31 CHAR------------
will display above the cursor as a guideline for the number of characters to be entered. As mentioned before, more than these characters can be entered under a key code, but since each key code letter (A THRU Z) only 31 characters + an End-of-Text mark, the text starting at the 33rd character will reside in the area allocated for the next letter code. For example: If the text :
NOW IS THE TIME FOR ALL GOOD MEN TC WRITE PGMS(CTRL-C) is entered under key code "X", then hitting EscX will send the entire text to the BASIC interpreter; while hitting EscY will only send the portion "TO WRITE PGMS" since it starts at the 33rd character.

This may be used to send a name & address to a program, or address only, etc.

Another method of entering text is by use of delimiters. The semicolon (;) is used to signify to the program the end of characters. Thus an entry formatted as: NOW IS THE;TIME FOR ALL;GOOD MEN(CTRL-C) under a key code letter "Z", will send text as follows:
EscZ = NOW IS THE Esc = TIME FOR ALL (Rubbing out "NOW IS THE")
Esc = GOOD MEN (Rubbing out "TIME FOR ALL")

When entering the text, the last character must be a CTRL-C. This will indicate to the program the end of the text.

More that one line of text is allowed, with the carriage return being sent along with the text to the BASIC interpreter. Thus the custom text:

```
1000 REM---COMMON USED ROUTINE---
1010 FOR A = 1 to 10
1029 PRINT "     "
1030 NEXT A
1040 RETURN
(CTRL-C)
```

may be stored in memory under any desired key (A-Z).

NOTES:
When customizing long lines of text, the text should be defined under a key that will allow for that amount of storage. The storage area reserved for the last key, "Z", is 32 characters ending at the top of memory for a given version, that is, program "QUICK32K" for example has it's end at "Z" area + 32 characters or 9FFF. The version of the program assembled at address 0000H, "QUICK0", has the same text as the remaining versions, but stored in blocks of 64 charac- ters to allow for longer lines of text. Since North Star BASIC does not use the area of memory between 0000H and 1FFFH, longer text may be stored under this version and saved to disk for future use.

After all the text has been entered in the customizing routine, and then ended with a CTRL-C, the program will return to the command mode. Once the program is executed with the "E" command, it will re-route the I/O and cannot be jumped to and executed or customized again unless the I/O has been restored. This is done by the sequence ESC0. This command will restore the I/O to it's original configuration and the program will be inoperative.
If the customizing routines have been in fact overlaid, then the program must be re-executed from the disk. As a rule, all the customizing should be done after the program has been loaded from disk to avoid locking up the system.

The user must ALWAYS restore the I/O routines with the EXC0 command before loading another program that will overlay the QUICK-KEY program. Please remember that this program sends data IN to the DOS, BASIC or any other pgms that uses the DOS I/O routines. COMMANDS as LOAD, DESTROY, NSAVE, etc. are sent just as if they had been typed in at the keyboard & must be deleted manually if incorrect! manually if incorrect!

CONTINUED FROM PAGE 20

```
1020 *
1030 * THIS IS THE SYSTEM MESSAGES ROUTINE
1040 *
1050 MSSG0 LXI H,TEXT0
1060  JMP PRINT
1070 MSSG1 LXI H,EXT1
1080  JMP PRINT
1090 MSSG2 LXI H,TEXT2
1100  JMP PRINT
1110 MSSG3 LXI H,TEXT3
1120  JMP PRINT
1130 MSSG4 LXI H,TEXT4
  40 PRINT CALL CRLF
1150 PRIN2 MOV B,M
1160  CALL CHOUT  SEND A CHAR
1170  INX H
1180  CPI ETX  END OF TEXT?
1190  JNZ PRIN2  NO?,LOOP
1200 CRLF MVI B,CR
1210  CALL CHOUT
1220  MVI B,LF
1230  CALL CHOUT
1240  RET  RETURN
1250 CHOUT XRA A  CLEAR ACCUM
1260  CALL COUT  CALL DOS COUT
1270  RET  RETURN
1280 *
1290 * THESE ARE THE SYSTEM MESSAGES
1300 *
1310 TEXT0 DB '    * QUICK-KEY *'
1320  DB CR
1330  DB LF
1340  DB 'NS BASIC STATEMENT GENERATOR'
1350  DB LF
1360  DB ETX
1370 TEXT1 DB 'CUSTOMIZE (C) OR EXECUTE (E) PROGRAM?'
1380  DB ETX
1390 TEXT2 DB 'I/O PATCHED."MEMSET" BASIC NO HIGHER THAN XXXXX'
1400  DB CR
1410  DB LF
1420  DB 'RETURNING TO DOS...'
1430  DB ETX
1440 TEXT3 DB 'FOR KEY LETTER (A THRU Z)?:'
1450  DB ETX
1460 TEXT4 DB '       ENTER TEXT BELOW:'
1470  DB CR
1480  DB LF
1490  DB '------------31 CHAR------------'
```

QUICKKEY

CONTINUED FROM PAGE 21

```
1500  DB ETX
1510 *
1520 * THIS IS THE FUNCTIONAL PART OF THE PGM
1530 *     (DOS CHIN CALLS NOW JUMP HERE)
1540 *
1550 INPUT PUSH H  SAVE USER'S REGISTERS
1560  PUSH D
1570  PUSH B
1580  PUSH PSW
1590  LDA NEXC  CHK IF MORE CHAR
1600  CPI 0FFH  MORE CHAR TO PRINT?
1610  JZ SEND  YES,KEEP SENDING
1620  LDA RUB  CHK IF STILL ERASING
1630  CPI 0FFH  MORE CHAR TO "RUB"?
1640  JZ NEXT  YES,GO TO IT
1650 INCHR CALL CHIN  NO,GET CHR
1660  MOV B,A  SAVE THE CHARACTER
1670  CPI ESC  IS IT "ESCAPE"?
1680  JZ PROC  YES,SKIP
1690 NPROC MVI A,00H  FLAG FALSE
1700  STA KEY  STORE IT
1710  STA COUNT  ZERO CHAR COUNTER
1720  JMP RSTOR  NO,GO RETURN
1730 *
1740 * THIS ROUTINE PROCESSES THE ESC
1750 *                            SEQUENCE
1760 PROC LDA KEY  CHK KEY FLAG
1770  CPI 0FFH  KEY ALREADY HIT?
1780  JZ NEXT  YES,MUST WANT NEXT
1790  LXI H,0000H
1800  LXI D,TEXT
1810  CALL CHIN  NO,GET KEY CHAR
1820  CPI '0'  ASCII 0?
1830  JZ RPATC  YES,GO REPATCH DOS
1840  CPI 020H  SPACE BAR?
1850  JZ SPCS
1860  CPI 041H  LESS THAN 'A'?
1870  JC INCHR
1880  CPI 05BH  LESS THAN '('?
1890  JNC INCHR
1900 LETTR SUI 040H  MULTIPLIER
1910  MOV L,A  LSB
1920  DAD H
1930  DAD H
1940  DAD H
1950  DAD H
1960  DAD H
1970 SPCS DAD D  ADDR+OFFSET
1980  JMP SEND0
1990 *
2000 * THIS ROUTINE RUBS OUT STATEMENT,SENDS
2010 *                                  NEXT
2020 NEXT MVI A,0FFH  SET RUB FLAG
2030  STA RUB  TRUE,ERASING
2040  LDA COUNT  LOAD COUNT
2050  DCR A  DECREMENT
2060  STA COUNT  RE-STORE
2070  JNZ NEXT2
2080  MVI A,00H  SET FLAG FALSE
2090  STA RUB  ALL DONE
2100  JMP SEND  NEXT STATEMENT
2110 NEXT2 MVI B,05FH  ASCII "RUB"
2120  JMP RSTOR  RETURN W/RUB
2130 *
2140 * THIS ROUTINE OUTPUTS THE TEXT
2150 *
2160 SEND0 SHLD ADDR  BASE ADDR
2170  SHLD POINT  BASE POINTER
2180  MVI A,0FFH  FLAG TRUE
2190  STA KEY  FOR KEY &
2200 SEND MVI A,0FFH  TRUE
2210  STA NEXC  FOR MORE CHARACTERS
2220  LXI H,COUNT  CHAR COUNTER
2230  MOV C,M  COUNT TO C
2240  LHLD POINT  CURRENT ADDR POINTER
2250  MOV B,M  CHAR TO B
2260  INX H  CHK NEXT CHR
2270  MOV A,M  TO A
2280 NEXCK CPI ';'  DELIMITER?
2290  JNZ ENDCK  NO?,CHK FOR END
2300  MVI A,00H  ZERO FLAG
2310  STA NEXC  NEXT CHAR FLAG
2320  INX H  SKIP OVER THE ';'
2330  INR C  INCR COUNT
2340  JMP RSET
2350 ENDCK CPI ETX  END OF STATEMENTS?
2360  JNZ RSET  NO?,SKIP
2370  LHLD ADDR  RE-LOAD BASE ADDR
2380  MVI A,00H  ZERO FLAG
2390  STA NEXC  NO NEXT CHAR
2400  INR C  INCR COUNT
2410 RSET SHLD POINT  NEW ADDR
2420  INR C
2430  MOV A,C
2440  STA COUNT
2450 *
2460 * THIS ROUTINE RETURNS W/THE CHARACTER
2470 *
2480 RSTOR POP PSW  RESTORE REGISTER
2490  MOV A,B  CHARACTER TO A
2500  POP B
2510  POP D
2520  POP H
2530  RET  RETURN FROM CHIN CALL
2540 *
2550 * THIS ROUTINE RE-PATCHES THE DOS I/O
2560 *
2570 RPATC LHLD CHIN+2  USER'S CHIN ADDR
2580  SHLD CIN+1  BACK IN DOS JUMP TABLE
2590  POP PSW
2600  POP B  RESTORE USER'S REG
2610  POP D
2620  POP H
2630  JMP CIN  RETURN TO DOS CALL
2640 *
2650 * THIS IS THE NEW CHIN ROUTINE
2660 *
2670 CHIN XRA A
2680  CALL 0000H  DUMMY ADDRESS
2690  RET
2700 *
2710 * THIS IS THE TEMPORARY DATA AREA
2720 *
2730 NEXC DS 1  MORE CHARACTERS FLAG
2740 RUB DS 1  MORE CHAR TO RUB FLAG
2750 KEY DS 1  KEY LETTER HIT
2760 ADDR DS 2  BASE TEXT ADDRESS
2770 POINT DS 2  POINTER ADDRESS
2780 COUNT DS 1  NO.OF CHAR PRINTED
2790 * THIS IS THE TEXT AREA
2800 TEXT DB '      ;          ;
2810  DB ETX
2820 TEXTA ORG TEXT+32
2830  DB 'ABS;APPEND;ASC;ATN;AUTO'
2840  DB ETX
2850 TEXTB ORG TEXTA+32
2860  DB 'BYE'
2870  DB ETX
2880 TEXTC ORG TEXTB+32
2890  DB 'CALL;CHAIN;CHR$;CLOSE;CONT;COS'
2900  DB ETX
2910 TEXTD ORG TEXTC+32
2920  DB 'DATA;DEF;DEL;DESTROY;DIM'
2930  DB ETX
2940 TEXTE ORG TEXTD+32
2950  DB 'ELSE;EXIT;EXAM;EXP;ERRSET;END'
2960  DB ETX
2970 TEXTF ORG TEXTE+32
2980  DB 'FOR;FILE;FNEND;FILL;FREE'
2990  DB ETX
3000 TEXTG ORG TEXTF+32
3010  DB 'GOSUB;GOTO'
3020  DB ETX
3030 TEXTH ORG TEXTG+32
3040  DB 'H IS BLANK, ERASE THIS LINE'
3050  DB ETX
3060 TEXTI ORG TEXTH+32
3070  DB 'IF;INPUT;INT;INP;INCHAR$'
3080  DB ETX
3090 TEXTJ ORG TEXTI+32
3100  DB 'J IS BLANK, ERASE THIS LINE'
3110  DB ETX
3120 TEXTK ORG TEXTJ+32
3130  DB 'K IS BLANK, ERASE THIS LINE'
3140  DB ETX
3150 TEXTL ORG TEXTK+32
3160  DB 'LIST;LOAD;LET;LINE;LOG'
3170  DB ETX
3180 TEXTM ORG TEXTL+32
3190  DB 'MEMSET'
3200  DB ETX
3210 TEXTN ORG TEXTM+32
3220  DB 'NEXT,NSAVE;NOENDMARK'
3230  DB ETX
3240 TEXTO ORG TEXTN+32
3250  DB 'ON;OPEN;OUT'
3260  DB ETX
3270 TEXTP ORG TEXTO+32
3280  DB 'PRINT;PSIZE'
3290  DB ETX
3300 TEXTQ ORG TEXTP+32
3310  DB 'Q IS BLANK,ERASE THIS LINE'
3320  DB ETX
3330 TEXTR ORG TEXTQ+32
3340  DB 'REM;RETURN;RESTORE;READ;RND'
3350  DB ETX
3360 TEXTS ORG TEXTR+32
3370  DB 'SAVE;SQRT;STOP;SIN;SGN;STR$'
3380  DB ETX
3390 TEXTT ORG TEXTS+32
3400  DB 'THEN;TAB;TYP'
3410  DB ETX
3420 TEXTU ORG TEXTT+32
3430  DB 'U IS BLANK,ERASE THIS LINE'
3440  DB ETX
3450 TEXTV ORG TEXTU+32
3460  DB 'VAL'
3470  DB ETX
3480 TEXTW ORG TEXTV+32
3490  DB 'WRITE#'
3500  DB ETX
3510 TEXTX ORG TEXTW+32
3520  DB 'X IS BLANK,ERASE THIS LINE'
3530  DB ETX
3540 TEXTY ORG TEXTX+32
3550  DB 'Y IS BLANK,ERASE THIS LINE'
3560  DB ETX
3570 TEXTZ ORG TEXTY+32
3580  DB 'Z IS BLANK,ERASE THIS LINE'
3590  DB ETX
3600 *
3610  END
```

CURSOR

```
10 !" DEMONSTATION OF CURSOR CONTROL FOR SOPHISTICATED"
20 !" INTERACTION WITH USER IN DATA INPUT"
30 !" ............BASED ON A PROGRAM BY KATHY RANDAZZO"
40!"                    **********************"
45!"USE EDIT TO TRANSFER LINE NUMBERS AND KILL ! & QUOTES"
46!"                    **********************"
50!"FOR ADM3A=ENTER THESE AT LINES 140 & 145:"
51!"    140 DEFFNC$(X$,Y$)=CHR$(27)+CHR$(61)+CHR$(R+31)+CHR$(C+31)"
60!"    145L=23:W=80:!CHR$(26)"
70!"FOR HAZELTINE ENTER THESE AT LINES 140 & 145:"
71!"    140 DEFFNC$(X$,Y$)=CHR$(126)+CHR$(17)+CHR$(C-1)+CHR$(R-1)"
72!"    145L=23:W=80:!CHR$(126)+CHR$(28)"
75!"FOR SOROC ENTER THESE AT LINES 140 & 145"
76!"    140 DEFFNC(X$,Y$)=CHR$(27)+CHR$(61)+CHR$(R+31)+CHR$(C+31)"
77!"    145L=23:W=80:!CHR$(27)+CHR$(42)"
80 !"FOR SOL OR VDM USE LINES 140 AND 145 AS ENTERED"
85 !:!"NSAVE ROUTINE FROM LINE 130 OR RUN 130":!:!:STOP
90 REM L=SCREEN LENGTH - W=SCREEN WIDTH - CLEAR SCREEN FOLLOWS
95 REM X$ & Y$ DUMMIES FOR CURSOR CONTROL DEFINED FUNCTION
96 REM B$=TAKES DATA INPUT - A$=STORES DATA FOR LATER USE
100 REM C & C$ ARE COLUMN MARKERS - R & R$ ARE ROW MARKERS
110!"                     ******************"
120 REM THIS IS THE SUBROUTINE THAT DOES IT
130 DIM X$(1),Y$(1),A$(200):A$="":C$="0":R$="0"
140 DEFFNC$(X$,Y$)=CHR$(27)+CHR$(2)+CHR$(R-1)+CHR$(27)+CHR$(1)+CHR$(C-1)
145 L=15:W=64:!CHR$(11)
160 C=1:R=1 :REM SETS TOP PRINT POSTION
170 !FNC$(X$,Y$),:INPUT"COLUMN:",C$ :GOSUB220
180 C=W-35:R=1 :REM SETS TOP PRINT POSITION
190!FNC$(X$,Y$),"(# TO QUIT)                 ",:INPUT"ROW:",R$:GOSUB220
200 C=VAL(C$):R=VAL(R$)
210 !FNC$(X$,Y$),:INPUT"DATA?",B$:GOSUB220:A$=A$+B$:GOTO160
220 IF B$="#"THEN 230:IF C$="#"THEN230:IF R$="#"THEN 230
221IF VAL(C$)>=W-5ORVAL(C$)<1THEN160:IF VAL(R$)>=LORVAL(R$)<2THEN180:RETURN
230 C=1:R=1
240 !FNC$(X$,Y$),"YOU ENTERED ",A$:C=20:R=L-5:!FNC$(X$,Y$),
250 ! "VIS-CALC ANYONE?"
```

PENCIL

can continue loading and storing presets. I store the same value (40H) at 4C97 to slow down the preset scroll speed.

Continue loading and storing with the machine code format of 3E XX 32 ZZ 4C; where XX is the new preset value and ZZ is the low address byte for that preset. Write a C9 (RETURN) when you are done, and of course don't go past the "00" of the original written message.

Remember to save your new Pencil. My modified program looks like this:

```
DH 2A28-2A36
2A28  3E 20 32 A1  4C 3E 48 32   BD
4C 3E 40  CD 6D 36
>

DA 366A-367D
366A  47 4F 00 32  9E 4C 32 97   4C
3E 48 32  C1 4C 3E 08
      G  O     2  -   L  2 -     L
>  H  2  -A  L  >
367A  32 A0 4C C9
      2 -   L -I
>
```

With these modifications I very frequently boot, write, save, and print. That's about as handy as a text editor can be!

96 TRACKS

819 KILOBYTES With
NEW CONTROLLER FOR NORTH STAR

BY Clyde Steiner

A new relocatable (double density) controller board has emerged from southern California. It is compatible with North Star's hardware and software and then some. They call it PHASE LOCK II.

The prom is switch RELOCATABLE anywhere in memory by the user. The controller works with the new 48 track and also the 96 track drives from Tandon. The 96 track double sided unit gives you 819k per drive. The controller also runs North Star's standard 35 track drives either double or single sided.

According to the manufacturer, the board can be modified to operate with a 64k memory board. We have not tested that claim.

There is a new DOS to go with the PHASE LOCK II which uses all the standard North Star command conventions and user jump addresses to be compatible with your old North Star software. The DOS has been rewritten internally for obvious copyright reasons but it also offers some small enhancements.

The GO is no longer necessary, just type in the program name ala CPM. In addition there is space in the command table for the user's custom commands and their jumps to special routines like print drivers, etc. Interestingly the controller permits you to boot up from drive 1 or 2 or 3 or 4.

This new double density DOS, called McDOS, only occupies the space up to 2A00 permitting older programs that loaded at 2A00 to run without modification. This is accomplished by both loading the McDOS boot program in low memory and using the area below 2000 for the operating system's buffer areas. You therefore have to have memory below 2000. A new relocatable McDOS is promissed.

The commented source code for the McDOS is available on disk for $40. It includes a special version for the SOL Computer

All this at a price considerablly under North Star's. Nothing like a little competition to sweeten the user's pie.

Unfortunately there there is no dynamic file handling in this independently produced DOS, but then neither is that facility forthcoming from North Star.

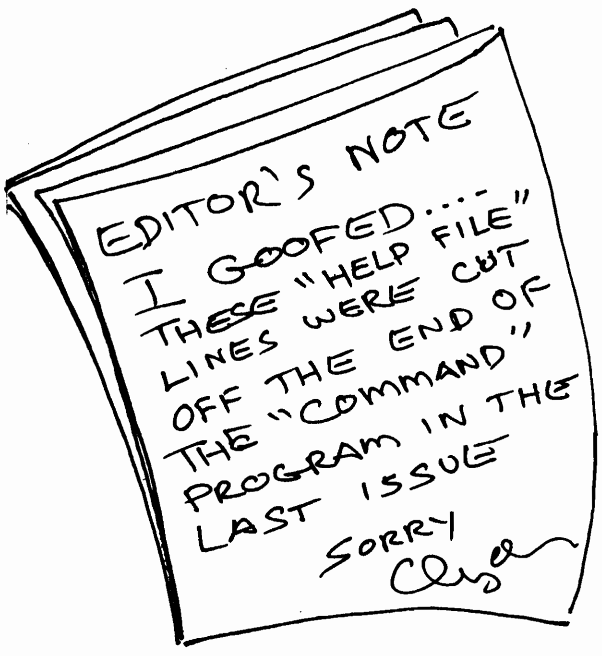
Preliminary testing here for two weeks has shown that the controller operates reliably and as claimed by the designer, Robert Lee Hogg. Wish we could say the same for the Tandon drives, one of which broke down in a couple of days. It's to the credit of Hogg's outfit, Micro Complex, 25651 Minos Street, Mission Viejo, CA. 92691 Tel (714)770 2168 that they agreed to a no hassel repair on the Tandon unit.

NORTH STAR COMPUTERS, INC., has moved from Berkeley to:

NORTH STAR COMPUTERS, INC.
14440 Catalina Street
San Leandro, CA 94577

Communications telephone numbers:
Business (415) 357-8500
Telex (910) 366-7001
Proxima (Modem Bulletin Board)
(415) 357 1130
Hotline (10:00 am - 04:00 pm PST)
(415) 357-8500

```
3600 !"COMMAND MODE:"
3610 !
3620 !"E.....ENTER COMMANDS, IF ANY, TO THE FILE"
3630 !"       IF NO COMMANDS PRESENT, THEN END PROGRAM"
3640 !"       IF COMMAND(S) PRESENT THEN REQUEST A TEST RUN"
3650 !
3660 !"C.....ERASE PREVIOUS COMMANDS AND GET NEW COMMANDS"
3670 !"Cn....ENTER NEW COMMANDS COMMENCING WITH NUMBER 'n'"
3680 !
3690 !"A.....ADD NEW COMMANDS TO THE PRESENT COMMANDS"
3700 !
3710 !"D.....DISPLAY CURRENT COMMAND ON FILE"
3720 !
3730 !"RET...EXIT PROGRAM WITH OUT ALTERING PREVIOUS DATA"
3740 GOSUB 3810
3750 !"DURING ACTUAL RUN OF THE FINNISHED FILE A CHECK WILL BE"
3760 !"MADE FOR CTR/C;IF PRESSED THEN THE RUNN WILL TERMINATE."
3770 !"ALSO CHECK WILL BE MADE WHETHER A ESCAPE KEY (MAY BE"
3780 !"CHANGED TO OTHER KEY) WAS PRESSED. IF IT WAS THEN"
3790 !"CURRENT COMMAND WILL BE COMPLETED AND THEN THE RUN WILL"
3800 !"TERMINATE."
3810 !
3820 INPUT "press RETURN to continue",A$
3830 !C$,
3840 RETURN
```



EDITOR'S NOTE
I GOOFED....
~~THESE~~ "HELP FILE"
LINES WERE CUT
OFF THE END OF
THE "COMMAND"
PROGRAM IN THE
LAST ISSUE*
SORRY
[illegible]

* ITS ALL ON OUR SOFTWARE LIBRARY DISK #1004

INTERNATIONAL NORTH STAR USERS ASSOCIATION
PUBLISHERS OF THE COMPASS NEWSLETTER
POST OFFICE BOX 1318, ANTIOCH, CA. 94509

BULK RATE
U.S. POSTAGE
PAID
ANTIOCH, CA
PERMIT NO. 265

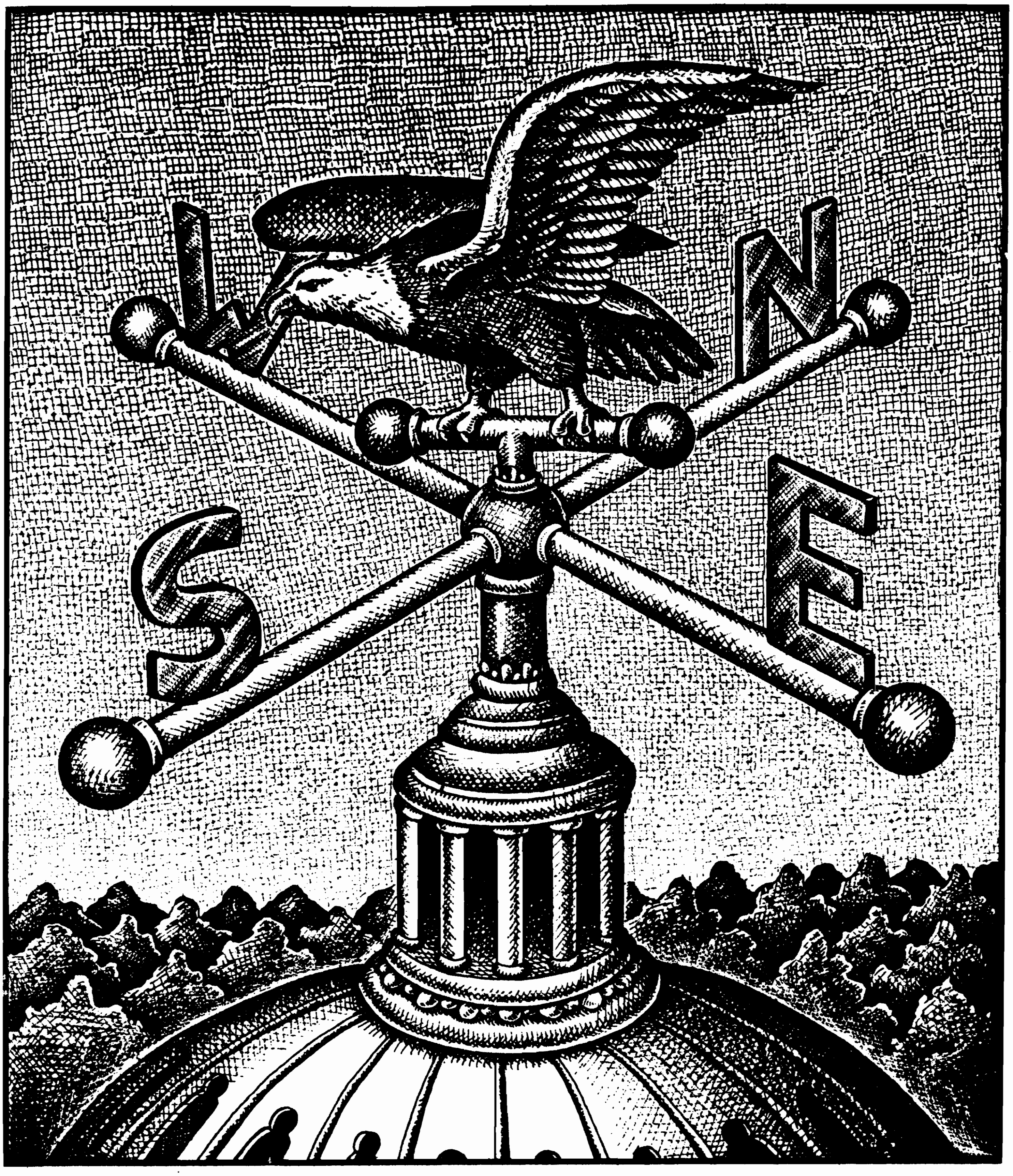
good news for you!

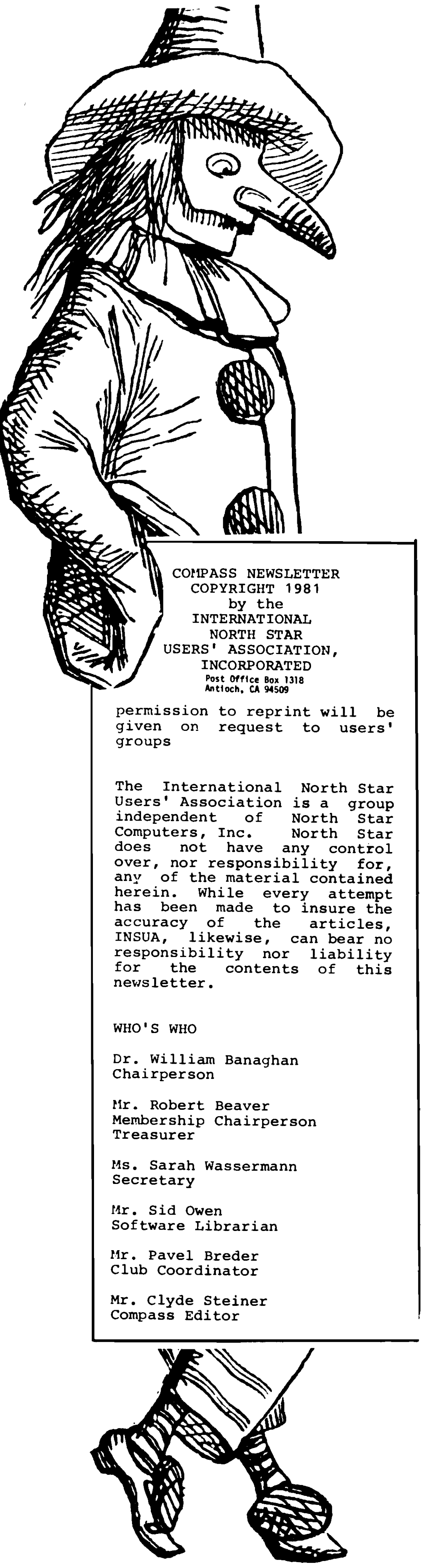
POSTMASTER: ADDRESS CORRECTION REQUESTED
Please enter key number on form 3579

THE COMPASS NEWSLETTER

VOLUME 1 NUMBER 3

INTERNATIONAL NORTH STAR USERS ASSOCIATION





COMPASS NEWSLETTER
COPYRIGHT 1981
by the
INTERNATIONAL
NORTH STAR
USERS' ASSOCIATION,
INCORPORATED
Post Office Box 1318
Antioch, CA 94509

permission to reprint will be given on request to users' groups

The International North Star Users' Association is a group independent of North Star Computers, Inc. North Star does not have any control over, nor responsibility for, any of the material contained herein. While every attempt has been made to insure the accuracy of the articles, INSUA, likewise, can bear no responsibility nor liability for the contents of this newsletter.

WHO'S WHO

Dr. William Banaghan
Chairperson

Mr. Robert Beaver
Membership Chairperson
Treasurer

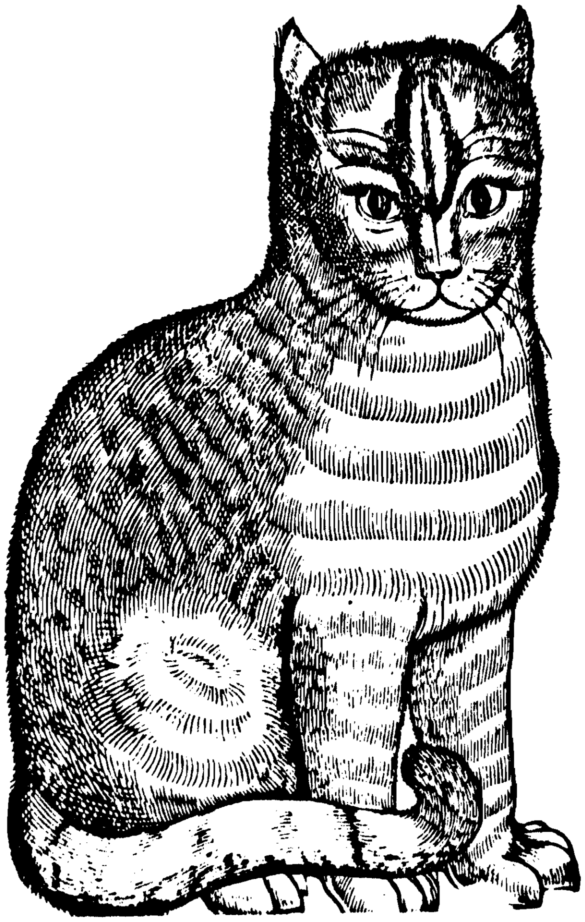
Ms. Sarah Wassermann
Secretary

Mr. Sid Owen
Software Librarian

Mr. Pavel Breder
Club Coordinator

Mr. Clyde Steiner
Compass Editor

IN THIS ISSUE:



BASEX COMPILER

Exploring Alternatives :
The BASEX Compiler
BY BOB BEAVER

Modem software not avilable to do what you want it to? Those waits for searches & sorts of your special data base got you staring wistfully at your screen? Need a fast tape handler for a special 'dedicated' computer application, but dreading the assembly language detail. Ready to design your own word processor, cause you want it done right for a change (ie your way)?

It's faster than a speeding bullet, more compact than a K-Car, what ho! it's BASEX, a compiler for 8080 systems. This might be just what you've been seeking for those applications that require high speed I/O, limited memory, and the programming convienence of a high level language.

Quick Feature index
(for those who want to know if they should read on.)
-Works in 8080/Z80/8085 systems
-Fast. 7-20times faster than Basic.
 (Very fast I/O & MEM operations)
-Compact. Compiled code save about 6K.
 (2K runtime package)
-Documented. 136 pages.
 (includes annotated sourcelisting)
-High Level Language.
 (similar to N* Basic)
-Relocatable programs.
 (relocator is provided)
-Optional extentions for N* DOS.

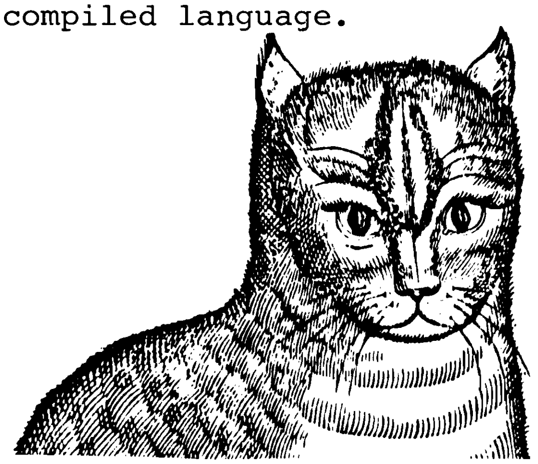
If all of that looks pretty good to you, it is likely you have needed something like this and have been faced with the choice of (a) writing your application in machine code (yetch) or (b) writing the equivalent of Basex, then writing your application program (double yetch).

Actually, as each machine code hacker knows, there is associated

CONTINUED

with each program an 'overhead' of command processors, hex ascii decoders, I/O protocols, etc, etc, etc....all reinvented for each program. The experts make a substantial investment in learing the tricks and assembling routines to handle these tasks. Paul Warme, the inventor of Basex, has made available a set of 'overhead' routines which are combined in ways that make a powerful high level language, while retaining speed and memory advantages of machine code.

Let's find out more about the advantages (and limitations) of this compiled language.



How Does Basex Work?

Basex programs use a set of execution routines which are located in low memory (0000-0863H, about 2K). These routines form the heart of the Basex system, much like DOS is the heart of the North Star operating system. Compiled Basex programs consist of a series of CALLs to the execution routines, resulting in compact and fast exececuting programs.

The minimum system consists of (a) the execution routines, 2K (b) User supplied I/O ?K (c) The compiled Basex program, ?K (d) Data areas required by the program, ?K. Say, 2.5K and up.

Disk file handling routines are also available for DOs 3,4, and 5.0. The minimum requirements are increased to include resident DOS (+2.5K) and disk handlers for Basex (+0.5K), about 5.5K and up. These can be optionally included or excluded as needed.

The Execution Routines

These are used by all Basex programs, and must be present during run time. These are self-modifying, and therefore are not ROMable. For most practical purposes, these are not relocatable, although there appears to be no theoretical reason why they could not be relocated. Relocation would, however, require extensive address modification to the execution routines, the Basex Compiler and Loader (I haven't mentioned these yet), and to any program dependent on these including Basex programs and Disk handlers. As I said, this could be a chore for the first one to tackle it. the rest of us will cheer encouragement from the rear (those in back cried "FORWARD", while those's in front cried "BACK!").

Annotated source listings of the execution routines are provided, including instructions on how to modify these for special applications. There are about 80D bytes within the 0000-0864H area for user modification, and it is easy to 'tack-on' additional routines to the end of the execution routine area.
It should also be noted that the execution routines preserve the interrupt processing area (0000-0063H) except for 0000H which is used to restart your Basex program.

The Compiler

An interactive compiler is provided, written in Basex, which allows the user to write, edit, and compile Basex programs. Since the compiler is itself a Basex program, it can be used to modify a version of itself (makes me dizzy to think about it), allowing for extensions which are of special interest to the user.

Annotated listings of the compiler, including instructions on how to modify this, are provided.

The Basex Loader

An interactive relocator, called the Loader, is provided to allow easy relocation of any Basex program (including the compiler) to convienent places in memory. All Basex programs continue to require

CONTINUED

the execution routines at 0000-0863H, however. Since the Loader is itself a Basex program, it can be used to relocate a version of itself (I'm getting dizzy again). The Basex Compiler can also be used to modify the Loader. Between these two programs alone, one could have a lot of fun.

The Loader also has the facility of compressing Basex programs to remove non-executable statements, such as remarks, and to remove unused space between program and symbol tables. Compressed programs are not modifyable by the compiler, or relocatable by the loader.

Annotated listings of the Loader, and instructions on how to modify this are provided.

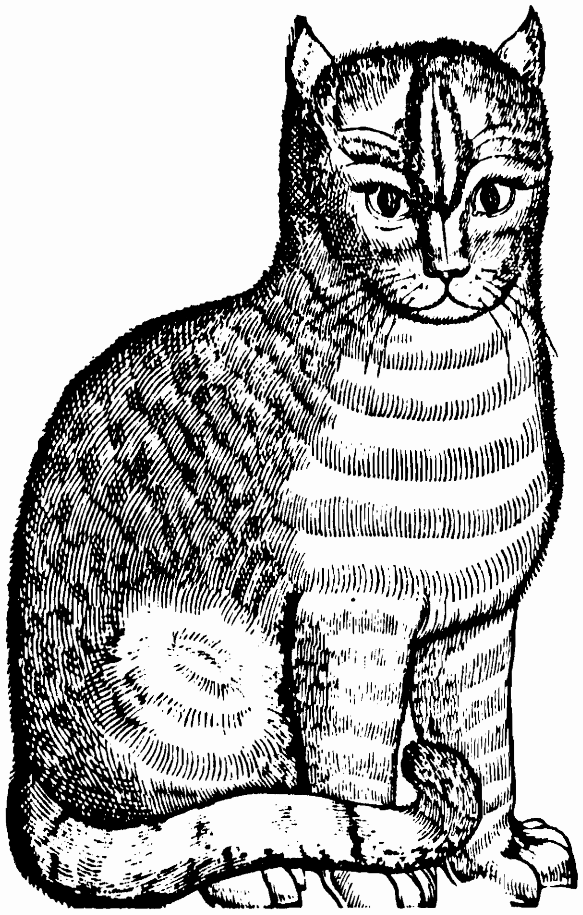
DOS Extentions

Extentions are available for DOS release 2 through 5. These are written in 8080 code which is assembled to execute at 2A00H (DOS 2,3,4) or 2D00H (DOS 5.0) assuming DOS at 2000H. These are connected to the compiler by the user, if desired, and provide a fairly complete set of file access functions (load, save, read, write, verify, open, close, create, delete, list). Sequential and random access functions are supported. Relocation of these routines to another address, and modification to run with DOS at other locations than 2000H, should be relatively straight-forward for an intermediate user (familier with assembly language/machine code), since annotated source listings are provided.

User Notes

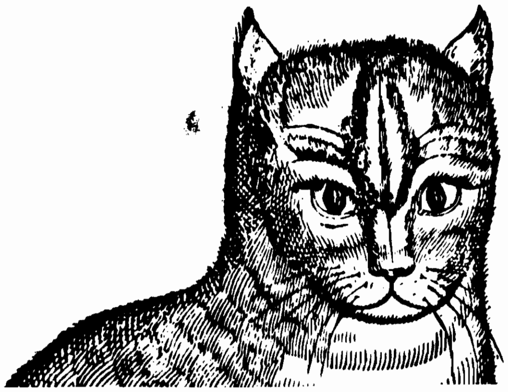
Some additional tidbits should be added to help you evaluate the Basex system. These include items which might be troublesome to the uninitiated, and some observations.

-Connecting Basex to your I/O. Simple, use your DOS I/O as is. However, the accumulator does not contain the I/O device number as is the recent N* convention and may need to be zeroed prior to entering your I/O routines.



-Getting started. Easy if you have a DOS at 2000h. Actually, the Basex system comes pretty much 'set-up' for n* users if the Disk extentions are purchased.
-upgrades for future DOSes. In order to save space, several undocumented routines in dos are used by the disk handlers. As DOS changes, upgrades may be requires.(ie DOS 5.2 handlers are not yet available).
-CP/M Use. Well, theoretically yes. Requires relocation of the execution routines etc.(see previous comments).
-String handling. Very good, similar to N* Basic except strings are limited to a maximum of 256 characters.
-Arithmetic. Fast 16 bit integer arithmetic. No floating point or math functions provided.
-I/O & Memory operations. Excellent, fast, more comprehensive than N* Basic. Fill, search, examine, move, input, output, wait/status ready.
-Interrupts. Fully provided for, including enable and disable interrupts commands.

CONTINUED



Learning More About Basex

Basex was written by INSUA member Paul Warme of Interactive Microware, Inc. 116 s. Pugh St., State College, PA 16801 (814) 238-7711. His review of Basex appears in DR Dobb's Journal No.30 pg 26-39.

The Basex manual (without DOS extentions) was published by BYTE BOOKS, a division of BYTE Publications,INC. 70 Main Street, Peterbourough, NH 03458 (603) 924-7217 as a PAPERBYTE book, selling for $8. All source listings are included.

Interactive Microware is offering the 97 page Basex Manual for $8, and a North Star single density disk containing Basex, Disk handlers and 39 page Disk (and Meca Tape system) guide for $35. Previous Basex purchasers may upgrade to the Disk handlers for $15 and return of the origional disk.

Paul Warme has also written a new book entitled "My Computer Speaks Basex", to be published by Haden Books this year (july?). This is a primer on Basex, including examples and explanations.

Bottom Line

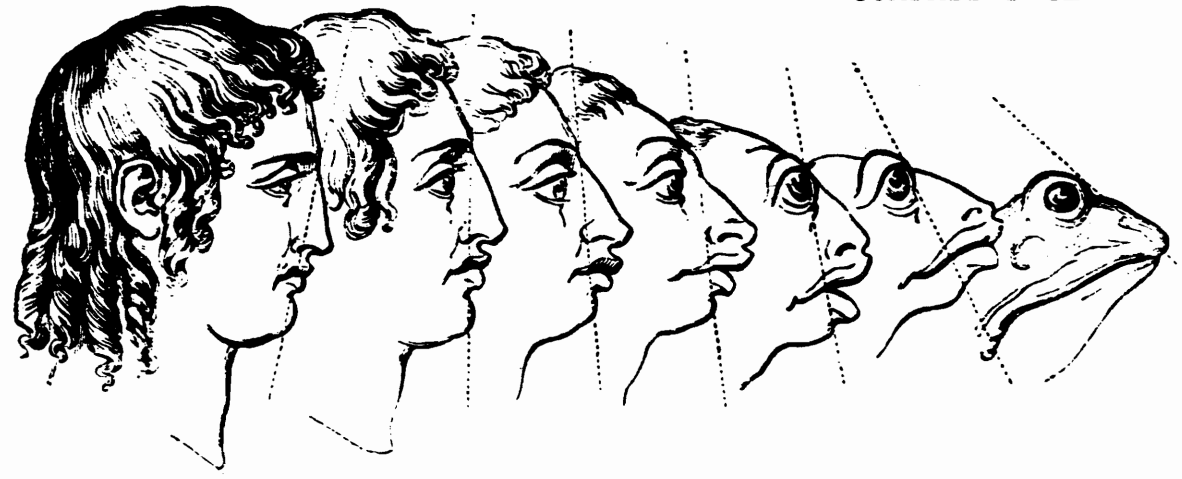
Job well done. Availability of source is a tremendous advantage to users (when used with discretion folks), and the system is available at very reasonable cost. Basex provides an intermediate between assembly language programming and Basic programming, which retains many of the advantages of each. The North Star User may find these to be of benefit in the creation of GO files for the North Star operating system, or in the development of utilities for CP/M.

EXAMPLE BASEX PROGRAM

The following program is a memory test in Basex, which performs 25,600 write, read, and compare operations to test 100 bytes of memory. It runs on my 8080 (1MHz) in 29 seconds and requires about 3K of memory.
Also shown is a N* Basic program wich performs the same operations, but runs in 400 seconds and requires 16 K. Note the striking similarity of the code. Note also that Basex uses lables instead of line numbers for line referencing. This is inherently an unfair comparison of the two languages, since N* Basic has floating point arithmetic, and is in other ways a superior language. It does point out that, in some situations, a significant advantage is obtained by using Basex.

```
        BASEX
        -----
11000 ?  REM !MEMORY TEST IN BASEX!
11006 ? PRT "FIRST AND LAST BYTES"
11012 ? INP FIRST LAST
11020 ? FOR I=0
11028 ? FOR J=FIRST
11036 ? BRT J=I
11044 ? BRD J
11050 ? SBT A-I
11058 ? JMP EQ OK
11066 ? BRD J
11072 ? PRT"BYTE"J"WAS"A";SHOULD BE" I
11088 ? CHR 13
11094 ? *** OK
11100 ? TIL J+1>LASB
11112 ? TIL I+1>255
11124 ? PRT "DONE"
11130 ? CHR 13
```

```
        BASIC
        -----
100 RER MEMORY TEST IN BASIC
110 INPQT"FIRST AND LAST BYTES:",F,L
120 FOR I=0 TO 255
130 FOR J=F TO L
140 FILL J,I
150 IF EXAM(Y)=I THBN 170
160 !"BYTE",J," WAS ",EXAM(J)," ;
                        "SHOULD BE ",I
170 NEXT J
180 NEXT I
190 !"DONE"
```



ASSEMBLE THE CODE

INSUA ASSEMBLER / EDITOR
By BRUCE KENDALL

This is an adaptation of the SCS assembler to the North Star Disk System for INSUA members providing both assembly of source listings as well a number of monitor functions to manipulate computor memory.

This is a specially made version of the self contained system (assembler/editor) written by Microtec and originally released thru Processor Yechnology (as SYS-8). It is configured to operate with a North Star dos at 2000H.

A second relocated version, shifted up by 300h to be fully compatable with the double density North Star dos is also furnished.

There are two segments:

1.Romable part: 2a00h-39ffh

2.Ram tables and stack: 3a00h-4300h

System entry points and dos links:

2A00h - system initialization and start up

2A04h - system restart entry (will not initialize pointers)

2A07h - jump to dos 'cout' routine

2A0Ah - jump to dos 'cin' routine

2A0Dh - jump to dos 'contrl-c' routine

2A - jump to dos entry point (usually 2028h)

Notes:

1. stack size is 39 bytes.

2. the format of the data bytes in the command, symbol,and assembler table uses 8080 standard, (low address byte first).

3. the assembler allows the use of star (*) or semi-colon (;) to indicate comments. It also tolerates a trailing colon (:) on labels to remain compatable with the intel format.

Special characters:

Control X or @ sign: delete current input line and restart input sequence.

Rubout, delete, or backspace: backspace one character and echo backspace (05Fh) to terminal.

Control-R: issue carriage return/ line feed and dump the input buffer to the terminal. Then continue input sequence from the last character in the buffer.This useful if your terminal does not backspace the cursor or print head when you delete a character.

CONTINUED

Control-C: halt assembly, listing, or other automatic output processes and restart (go back to command level).

There are 20 commands. Three commands are left as empty positions in the command table (NOPs at the end).

Summary of the valid commands :

Note: all commands consist of a 4-letter command name followed by one or more parameters (separated by spaces). In most cases the parameters are hex addresses or decimal line numbers.

DUMP <add add2>

This will give a hex dump of memory from add add2 (inclusive)

DUMPA <add add2>

This gives an ascii dump of memory.

EXEC <add

This jumps into the code starting at address: add

ENTR <add

This allows entering data (in hex bytes) into memory starting at address: add the bytes are entered in hex separated by spaces and terminated by a carriage return to continue entry on a new line or by a slash (/) and carriage return to finish the entry and exit the entry sequence.

LIST <line #>

This will begin listing the current text file beginning with the specified line # or from the beginning of the file if no line # is specified.

DELT <line line2>

This will delete text lines (in the current file) from line1 to line2 (inclusive). If only the first number is specified it will only delete one line. Note, unlike some basics, the editor will allow a line with only a line number.

FILE

List the current file name and dimensions

FILES

List the dimensions of all open files (up to six)

FILE /fname/ add

Open (or create) a new text file in memory with name: fname (5 letters or less) starting at add.

FILE /fname/ 0

Delete file with name: fname. Note, this means that all text files must start above address: 00h.

FILE /fname/

Get file: fname (one of the already open text files) and make it the current file for editor and assembler operations.

ASSM add1 add2 Begin assembly of the current text file with the program counter starting at add1
On the second pass of the assembly process this will start loading the machine code in memory starting at add1 unless the second address is specified add2 in which case the code will actually be loaded in memory starting at add2
During pass 2 a listing of the assembly process will be sent to the terminal along with possible error messages:

o	opcode error
l	label error
d	duplicate label error
m	missing label error
v	value error
u	undefined symbol
s	syntax error
r	register error
a	argumemt error

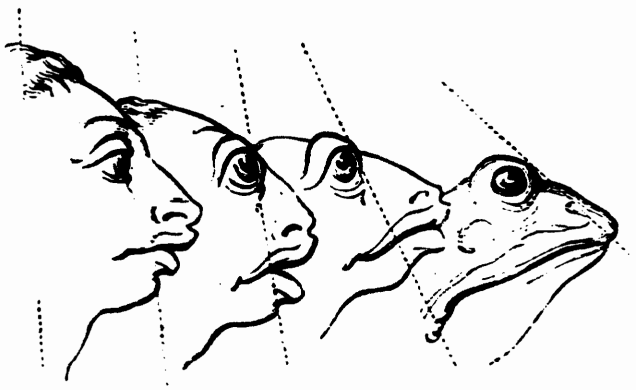
CONTINUED

ASSME add1 add2 similar to above command but only give terminal listing of lines containing errors

CUST branch to user supplied custom command at 4a00h

NDOS issue carriage return line feed and branch to 2A10H, which will take you back to dos

FILL add1 add2 byte fill memory from add1 to add2 inclusive with the specified byte. if byte not given default to 00



MOVE add1 add2 add3 copy code from add1 thru add2 to memory starting at add3 this will even work correctly if the source and destination areas overlap

COMP add1 add2 add3 compare memory from add1 thru add2 with memory starting at add3 and report the addresses and values of compare errors

FIND add1 add2 addx search memory from add1 thru add2 for the occurance of addx. if addx is given as 1 2 hex characters the search will be for a single byte,otherwise the search will be for a reference to a two byte word in intel reverse byte order such as a call or jump to addx

SYMB list the assembly symbol table after you have just assembled a text file

LEDT line dump the specified text line to the terminal and then position the cursor at the end of the line to either add characters to the line or backspace into it to change it

SRCH search the current text file for the occurance of the specified input string and list all text lines in which a match occures. after giving the command the computer prompts you for the input string with a colon

FCHK recover text file from memory at the current file position set by the file fname add1 command. if the file has errors in it it will attempt to recover as much as possible

CONTINUED

CLIP TIPS

Did you ever use the BASIC statement DESTROY to kill a file, only to realize too late that you have killed the wrong file? There is a painless way to salvage the situation, as long as you haven't written onto the disk or compacted it in the meantime. The key to this recovery is the fact that DESTROY acts only on the DIRECTORY ENTRY, not on the file itself.

In other words, the file is still in its original form on the disk, but the directory entry has been wiped out.

Suppose that you have a file called MISTAKE stored on disk. The directory shows the following information:

```
MISTAKE   54  12 D   2
```

If you erroneously DESTROY 'MISTAKE', you can recover it by using these two DOS operations:

```
CR MISTAKE 12 54 D
 TY MISTAKE 2
```

Notice that the length and disk starting address are reversed in the CR statement. In practice, you will seldom be blessed with all the necessary specifications, and you may have to do a bit of detective work to discover the length of the original file and its actual position on the disk.

PWJonas, 4/7/81

INSUA ASSEMBLER DISASSEMBLER

ON DISK FOR MEMBERS

ORDER LBRARY DISK NO.1007
ENCLOSE $15 TO COVER
MEDIA AND HANDLING

DISK INCLUDES BOTH
ASSEMBLER AND
DISASSEMBLER
IN BOTH SINGLE AND
DOUBLE DENSITY

Notes on the use of text files:
To start typing a text file, open or create a file with the file FNAME add1 command. type in text lines terminated by carriage returns. Note each valid text line must start with a 4 digit decimal line number followed by a space the next character must either be a comment character star or semi colon, the first letter of a label, or a space

All labels must start in this position

If you now want to save the file first determine its size with the FILE command then exit to dos with the NDOS command and create and save the file with the dos CR and SF commands
You can then re enter scs21 with a JP 2A04 command

To re-store a saved file from disc first open a text file with the file FNAME add1 command. then exit back to dos and load the text file at add1 with the LF command you can now recover the file pointers by re entering SCS21 with a JP 2A04 and issuing a FCHK command

If you are still awake, try loading the example file on the INSUA disk. Recover it, list it, assemble it, and run it

Good Luck

Editor's Note: All references in the above are to the single density version of the assembler-editor. add 300 hex for the double density version.

CONTINUED

DISASSEMBLE THE CODE

OPERATION GUIDE FOR NORTH STAR COMPATIBLE DISASSEMBLER

Distributed exclusively to INSUA members

Written by Lance E. Rose
P.O. Box 1082
Kalispell, MT 59901

Two versions of the disassembler are furnished, one running at 2A00 for single density systems and one at 2D00 for double desity.

The disassembler is executed by typing GO DSSM after loading the DOS. It will print the prompting character '>' and wait for a command. All commands are entered in the format:

(Letter) (Hex number up to 4 digits long) (Carriage return).

Typing delete erases the last character typed, and control-X cancels the entire line and allows it to be entered again. Omitting the hex number is the same as typing in a value of 0. If an invalid command is given or a hex number is entered which is too large or contains illegal characters, an error message (?) is printed and the command must be reentered. The list of commands available is as follows:

D =
Return control to the DOS.

C =
Clear the symbol table. When first executed the table is automatically cleared.

S =
Print the symbol table. It is printed on the command device which is assumed to have a line width of at least 64 characters.

P =
Set the program counter to the hex number typed in. When first executed, the program counter corresponds to the location in memory that the dissassembled program would normally be executed at.

O =
Set the offset to the number typed in. This is the difference between where the program to be dissassembled is currently located in memory and where it normally executes. For example, if a program normally executes at 2A00, but, due to the presence of the disassembler, must be loaded at 5000, the offset would be 2600. Note that this number must be given module FFF+1, so that if the same

CONTINUED

program were loaded at 0, the offset would be D600.

M =
Set the mode of dissassembly. The mode must be set to a number from 0 to 3 or an error message will result. The modes of disassembly are:

(0) disassemble but do not use the symbol table for any purpose;

(1) enter the 2-byte quantity in the W command or the 2nd and 3rd bytes in the L command into the symbol table as a label;

2) reference the symbol table and if the program counter corresponds to a label, print the label and all the locations which refer to it;

(3) perform the same operation as (2), but print the output on a hard copy device (assumed to be device #1 in the North Star I/O routines where the command device, such as a CRT, is device #0). This device is assumed to have a line width of at least 128 characters. If it is an 80 character device, location 2EFE in the disassembler must be changed from 10H to 06H.

L =
Disassemble starting at the present program counter and continuing up to but not including any instructions which start at the location of the hex number entered with the command. Opcodes are printed.

H =
Perform the same operation as in the L command, but do not enter the operands of 3-byte instructions into the symbol table even if mode (1) is in effect. This command is most useful for decoding LSI instructions where the operand is not a label.

W =
Disassemble 2-byte quantities as DW espressions. If mode (1) is in effect, they are entered into the symbol table.

B =
Disassemble 1-byte quantities as DB expressions.

A =
Disassemble as in the B command, but if the byte is in the range 20H to 7EH, print it as an ASCII character in the listing.

The best way to familiarize yourself with DSSM is to use it a few times to disassemble programs with which you are familiar, preferably which have source listings to compare the output with.

The disassembler in its normal version resides at 2A00 to 4FFF and requires 38 blocks of storage on disk. However, this includes a symbol table capable of holding approximately 800 labels and their cross-references. Therefore, once a symbol table is built up, it can be saved simply by saving the entire disassembler in a temporary file on disk. When called back from that file, it will still contain the symbol table for further processing. The symbol table is only cleared the first time the original disassembler is executed.

Some helpful hints:
When printing out long listings or symbol tables, the command may be aborted by typing a control-C. This will return control to the command level and print the prompting character. At this time, any variables may be changed and

CONTINUED

D CUSTOM DSSM

```
10 !"HARDWARE CUSTOMIZATION FOR INSUA DISASSEMBLER":!:!
20 !"MEMSET BASIC TO 28671 (HEX 6FFF)FOR DOS AT 2000"
30 !"EXIT TO DOS AND LOAD DSSM AT MEMSET+1,i.e 7000"
40 !"RETURN TO BASIC AND RUN PROGRAM AT 50":STOP
50 !"TYPE IN DECIMAL ASCII FOR CODES TO BE USED BY PROGRAM":!
60 INPUT"PRINTER PORT NOW #1 -CODE FOR NEW PORT (CR SKIPS) ",P$
70 IFP$<>""THEN P=VAL(P$)ELSEP=1
80 INPUT "DEC.FOR CHARACTER TO REPLACE TAB (NOW 9H=9D) ",Y
90 INPUT"DEC. CODE TO REPLACE BACKSPACE (NOW 7FH=127D) ",B
100 X=28672 :RE] DSSM NOW LOADED AT 7000HEX
110 !:!:!"SEARCHING AND CHANGING CODE"
120 FILL X+133,254:FILL X+134,P:FILL X+138,0:REM PRINT DEV.#
130 FILL X+852,Y :REM NEW TAB MARKER
140 FILL X+40,B :FILL X+180,B:REM NEW BACKSPACE CODE
150 X=X+1318:REM NEW SEARCH START
160 C=0:Y=32:GOTO180:REM ZERO OP CODE CHAR COUNTER & ADD SPACE
170 X=X+1:IFX=28672+3413THEN210:REM END SEARCH
180 E=EXAM(X):IFE>=65ANDE<=90THENC=C+1
190 IFC>=2ANDEXAM(X+1)=9THEN200ELSE170
200 !"*",:FILLX+1,Y:C=0:X=X+1:GOTO170
210 !:!"CHANGE COMPLETED---EXTI TO DOS":!"CR CUSTDSSM 14"
220 !"TY CUSTDSSM 1 (GO ADDRESS)":!"SF CUSTDSSM 7000"
```

CONTINUED

MARKETPLACE

THESE BUY, SELL, OR SWAP ADS ARE FREE TO MEMBERS OF INSUA. YOU MAY FIND THIS SECTION A CONVENIENT WAY TO TURN YOU EXTRA OR UNUSED EQUIPMENT INTO CASH. PLEASE NOTE THAT INSUA MAY HAVE TO LIMIT THE AMOUNT OF "MARKET PLACE" SPACE AVAILABLE AND ADS WILL BE PRINTED ON A FIRST COME FIRST SERVED BASIS. NO RESPONSIBLITY CAN BE TAKEN FOR THE CLAIMS OF BUYERS OR SELLERS. ALL ADS MUST BE "NON-COMMERCIAL". BUSINESS VENDORS OF SOFTWARE OR HARDWARE MAY ONLY ADVERTISE VIA DISPLAY ADVERTSING. PLEASE WRITE FOR ADVERTISING RATE SCHEDULE.

FOR SALE TASA KEYBOARD with base and socket -$50; Dynabyte 57 naked terminal board - $225.00; John Evans, 618 "N" St. #4, Anchorage AK 99501 (907) 272-7789

CUSTOM NS BASIC new commands, SIGI KLUGER, 1750 Lee Trevino #24H, El Paso, TX 79936 (915) 593-8013

WANTED Single Density Controller Carl Stewart, 6570 Covington Way, Goleta CA 93117 (805) 967-3001

WANTED Astrology, Numerology, Metaphysical programs wanted for North Star Horizon. Will share/trade what I have. Clark Johnson, Box 715, Tempe, AZ 85281 (602) 253-3338

WANTED Used computer science books in good condition. One title of each for personal use: Programming languages & techniques, Knuth Vol.2, Systems programming, Computer architecture, Databases. Martin Berman, 494 Forest Ave, Teaneck, N.J. 07666

PLOTTER FOR SALE Houston Instruments DMP-2 Hi-Plot digital plotter, mint condition $750. Conrad Romberg, Box 548, Rowlett, TX 75088 (214) 475-3913

WANTED Single or Double Density Controller Board, N* compatible. P.G.Springer, 208 W. John, Champaign, IL 61820, (217) 351-0943 the listing continued at the same value of the program counter. One can easily modify the control-C routine in the DOS to include the ability to stop and start displaying again. This is useful for displaying long listings on a CRT device.

An important characteristic of this disassembler is the necessity for any output device used with it to be able to tab. If your CRT or printer cannot do this, you will have to write some software into the output driver routine to keep track of the cursor or print head position and cause it to space to the proper column when a tab

****

EDITOR'S NOTE:

Readers will see from Lance Rose's description of the INSUA DISASSEMBLER, that this is indeed a versitile easy to use debugging or program investigation tool. I think you will find the cross referenced listing of all the calls and jumps to each lable a great enhancement to the normal symbol table produced by most other disassemblers.

Please note that the program is furnished to work with a standard printer on the serial (North S-ar #1 port). The short basic program listed here will convert printed output to #2 port and or remove all the imbedded tabs on print output if your hardware requires that.

We hope to add two futher enhancements to the INSUA DISASSEMBLER. The ability to write the source code created direct to disc as well as currently implemented CTR or printer output would be welcome. A mini editor to enable changing the symbal table references to meaningful names is, I think, a second useful enhancement. If any members wish to take on these challenges, please let me know. I'll provide you with the disassembler's source code.

****

SAMPLE DISASSEMBLY

5.2(DQ) Basic's keyboard control character decoding table.

Note that Basic was LF to 5D00 to avoid conflict with DSSM and then Offset (O command set to 3000). Program counter set to NORMAL address of start of area to disassemle (P command set to 4C9C). Disassembly set to run upto 4CCD (L command set to 4CCD). Lables are listed with L prefix in righthand column.

```
>P4C9C

>O3000

>L4CCD
4C9C  FE01    CPI       01H
4C9E  CA0E4D  JZ        L4D0E
4CA1  FE07    CPI       07H
4CA3  CA174D  JZ        L4D17
4CA6  FE11    CPI       11H
4CA8  CA584D  JZ        L4D58
4CAB  FE08    CPI       08H
4CAD  CA534D  JZ        L4D53
4CB0  FE1A    CPI       1AH
4CB2  CA474D  JZ        L4D47
4CB5  FE5F    CPI       5FH
4CB7  CA584D  JZ        L4D58
4CBA  FE7F    CPI       7FH
4CBC  CA584D  JZ        L4D58
4CBF  FE04    CPI       04H
4CC1  CA244D  JZ        L4D24
4CC4  FE19    CPI       19H
4CC6  CA784D  JZ        L4D78
4CC9  FE0E    CPI       0EH
4CCB  CA8A4D  JZ        L4D8A
```

CLIP TIPS

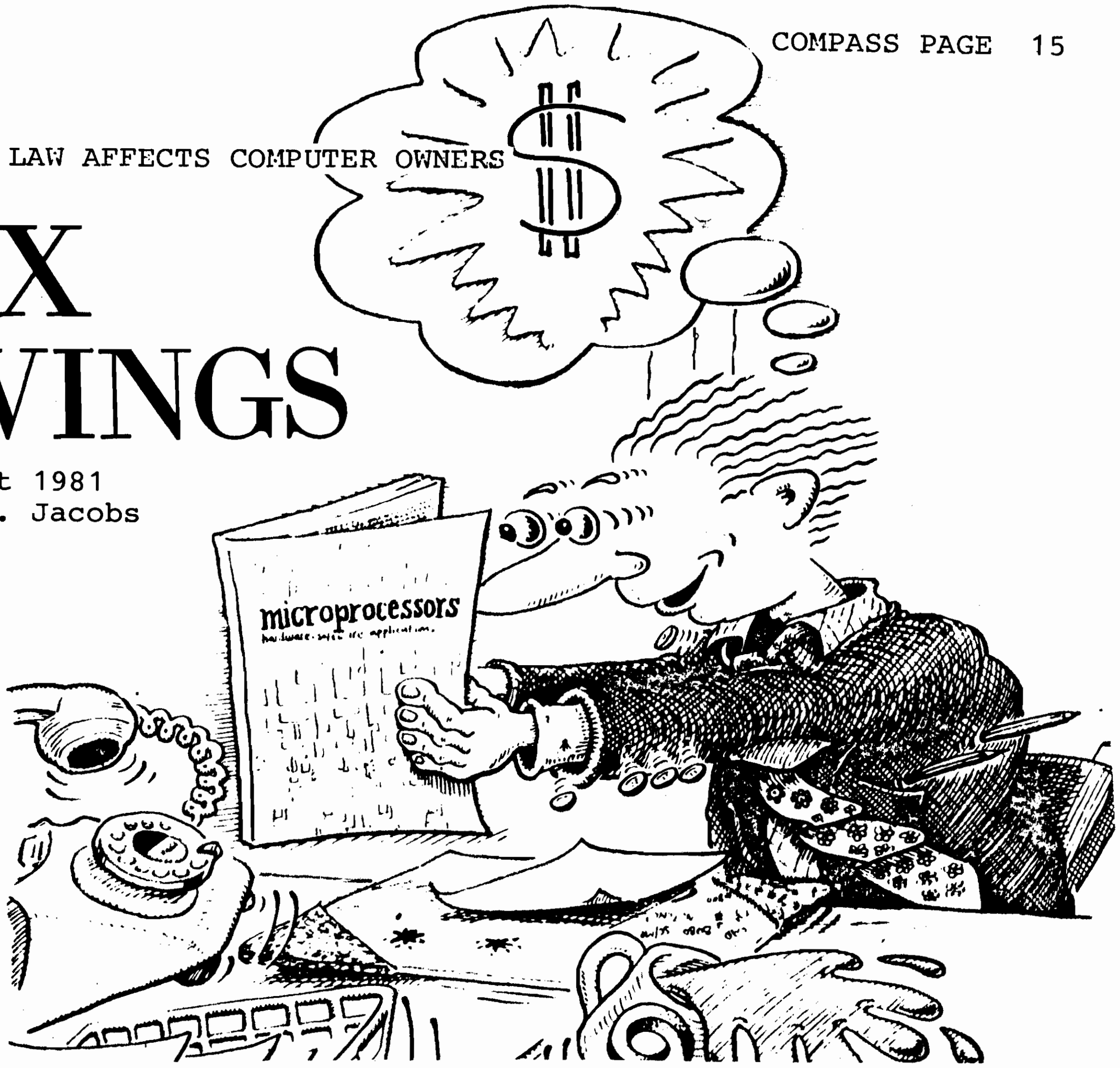
FIX FOR 5.1DQ DOS TO BOOT PROPERLY

```
10 OPEN#0 %0,"DOS,1"
20 WRITE#0 %59,&0, NOENDMARK
30 WRITE#0 %6,&1, NOENDMARK
40 CLOSE#0
```

HOW NEW TAX LAW AFFECTS COMPUTER OWNERS

TAX SAVINGS

Copyright 1981
Vernon K. Jacobs



Should you buy your first (or next) computer this year or next? Should you lease or buy? Are there any other areas of the new tax law that might affect those who own or are thinking of owning a computer? There are at least 109 specific provisions in the "Economic Recovery Tax Act of 1981", and it will be months (perhaps years) before the impact of all the provisions is evaluated. It's almost certain that we will have another tax bill early next year to correct the inevitable technical errors and flaws in this hastily drafted and complex set of tax law changes. Nonetheless, here is a brief summary of some of the provisions of the new tax law that should be of specific concern to computer owners and lessees.

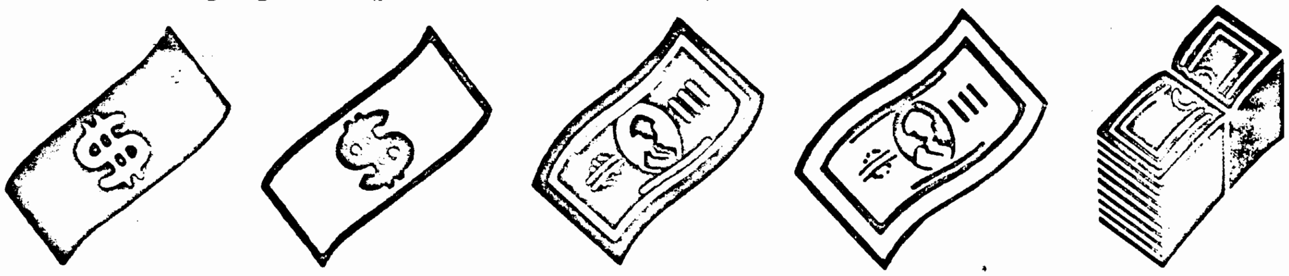
Full Write Off For Small Computers: One of the provisions of he new tax law will permit businesses to deduct the first $5,000 of business equipment acquired in 1982 and 1983, the first $7,500 of purchases in 1984 and 1985 and the first $10,000 of purchases after 1985. This means that many small desktop computers could be fully expensed in the year acquired. No investment credit would be allowed on such purchases but the immediate write off would usually be better. If the cost of the computer exceeds the deductible amount, the excess would be eligible for the new depreciation method. This full write off provision is not available for investors. It's only available if the equipment is to be used in a trade or business.

CONTINUED

TAX SAVINGS

New Depreciation Rules: If you purchased a computer in 1981, the 100% write off won't be available, but the new method of depreciation (called the "Asset Cost Recovery System") does apply to 1981 equipment purchased. Under the new method, computers will be depreciated over a five year period using specified rates for each of the five years. (If computers can be classed as research and development equipment they can be depreciated over a three year period.) For five year class equipment purchased in 1981 through 1984, the first year's depreciation will be 15% of the cost. The second years depreciation will be 22% of the cost and the rate will be 21% in each of the next three years. The entire cost will be deducted over the five year period.

By contrast, the prior law permitted a computer owner to write off up to 40% of the cost in the first year if the equipment was placed in service before July first. An additional 24% of the cost would be written off in the second year, 14.4% in the third year and 10.8% in the fourth and fifth years. This assumes a five year life, which has been typical for computer owners. Consequently, owners of larger and more expensive computers won't fare as well under the new law as under the old, but owners of desktop size computers will be better off - assuming no other equipment purchases in the year.



If the tax deductions won't be available because of other tax deductions of business losses, computer owners will be able to elect to write the equipment off over a 12 year or a 25 year period using a straight line method of depreciation. However, the election to use the slower method is mandatory for each years purchases - meaning you can't change your mind after a year or two. The main reason to use a slow method of depreciation is to avoid the possible loss of deductions during a prolonged start up period due to the existing time limit on offsetting losses of one year against profits of future years. The new law provides substantial relief in this area - which may make the slow depreciation method unnecessary. Previously, business losses could be carried forward for seven years, but the new law extends this to 15 years - retroactive to 1976.

Changes In Rules For Investment Tax Credit: Computer buyers will realize a small increase in the amount of available investment tax credit for purchasing a computer. Under current law, equipment with a five year useful life is eligible for 2/3 of the full 10% tax credit. Equipment with a five year life will now be qualified to claim the full 10% tax credit for equipment that is depreciated over a period of five or more years. If the

CONTINUED

equipment will have a three year useful life (autos, trucks and certain R & D equipment), the tax credit will be 6% of the cost of the property rather than 10%. These new tax credit rules take effect in 1981 - including property that was acquired before the law was passed. (It was signed on 8/13/81.)

There was no specific change relative to claiming the tax credit on the full cost of a system that included both hardware and software. However, if the tax credit is claimed on the software because the price is combined with the hardware, then the buyer must depreciate the software with the hardware. If the software is purchased separately, and is licensed rather than purchased, then the full software cost can be deducted in the year of acquisition.

New Rules For Defining Leases: Taxpayers and the IRS have been arguing for years about whether a lease is really a lease or just a method of financing an equipment purchase. The new tax law attempts to simplify some of the complex rules that have cropped up in this area of controversy. Basically, the parties must clearly agree that the transaction is a lease and the lessee must not acquire ownership of the property at any time during the lease. The lessor must be a corporation and must have an investment of at least 10% that is "at risk" in the investment. Generally, the property must be new property.

●●

About the author: Vernon K. Jacobs is a CPA who practices as a tax consultant in the Kansas City area. He is the Editor of Tax Angles, a nationally circulated tax planning newsletter, and he is the author/publisher of the Financial Systems Report - a monthly six page report about computer systems for financial specialists. In his spare time he develops tax oriented computer programs for desktop computers. A copy of either of his newsletters can be ordered from Research Press, Inc., P.O. Box 8137-P, Prairie Village, KS. 66208. Send $3 for each sample to cover postage and handling costs.

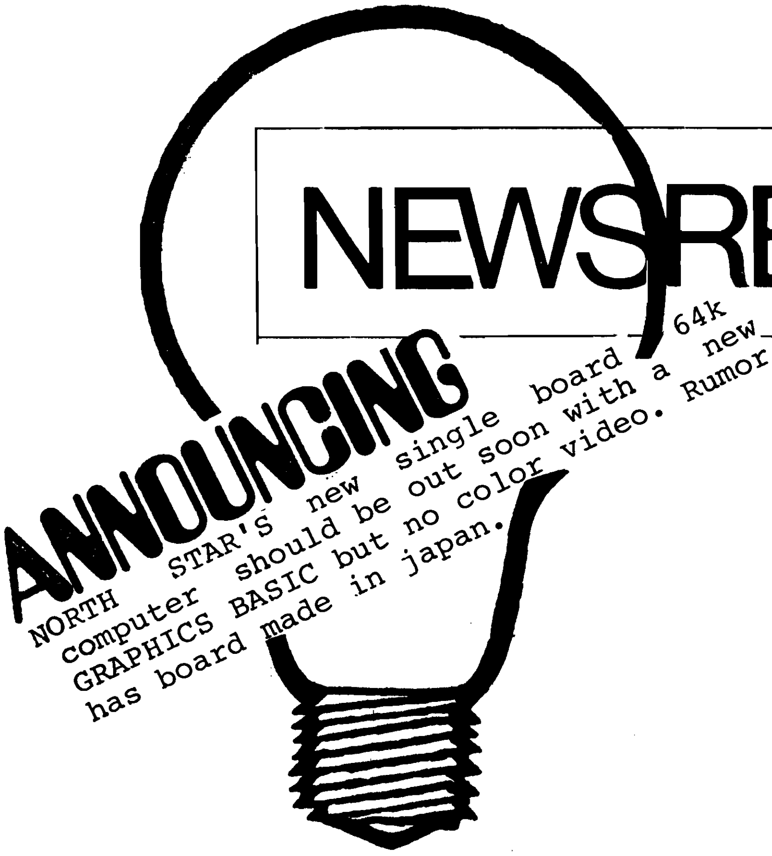
CLIP TIPS

19200 BAUD CORRECTION
(SEE Page 17 COMPASS ISSUE No TWO

EDITOR'S NOTE: Missing Information on 19200 Baud Conversion

Take a 14 pin dip header (like those used for configuring the I/O interfaces or the real time clock) and solder a 14 pin socket on top of the header. Solder all socket pins to the top of the header <u>except for pin 11 of the socket which should be bent out at a right angle. Connect a jumper between this pin and the molex socket on your CPU card.</u> Remove IC 7C (74LS14) from the motherboard, plug your socket-dip header sandwich into the socket 7C, and plug the 74LS14 into the socket on the sandwich. That's it. The baud rates will now be double those given in the Horizon manual when the baud rate header is configured.

NEWS REPORT



NORTH STAR'S new single board 64k computer should be out soon with a new GRAPHICS BASIC but no color video. Rumor has board made in japan.

NEW NORTH STAR DISC SYSTEM
By Bill Banaghan

We have been hearing a rumor, which we have been unable to substantiate with North Star Inc., that North Star already has an "octal" density drive, as compared to the present Quad density.

The capacity of the new disc is said to be 819 K or about four fifths of a megabite, as compared to the 360 K now available on the Quad. Not only is the density itself doubled, but there is an extension of the area of the disc used. Although this results in somewhat better than a doubling of storage, it will be necessary to use discs of a quality which can handle the added density.

A significant difference exists between the Quad and the octal in the formatting of the files on the disc. No longer does a file go from one end of the disc to the other end and then switch to the other side as it does on the Quad. Rather, on the new octal, there is a switching back and forth from a block on one side of the disc to a matching block on the other side of the disc as the file progresses throughout the disc.

Because of the increase in data per area, it has also been necessary to diminish the size of the head which will read the data.

The user of the octal unit will be able to read double density, but it will not be able to write octal density on the discs unless the discs are initialized octal and capable of handling octal density. Therefore, it will not be possible to write octal over double density data. The disc must be formatted for octal before writing.

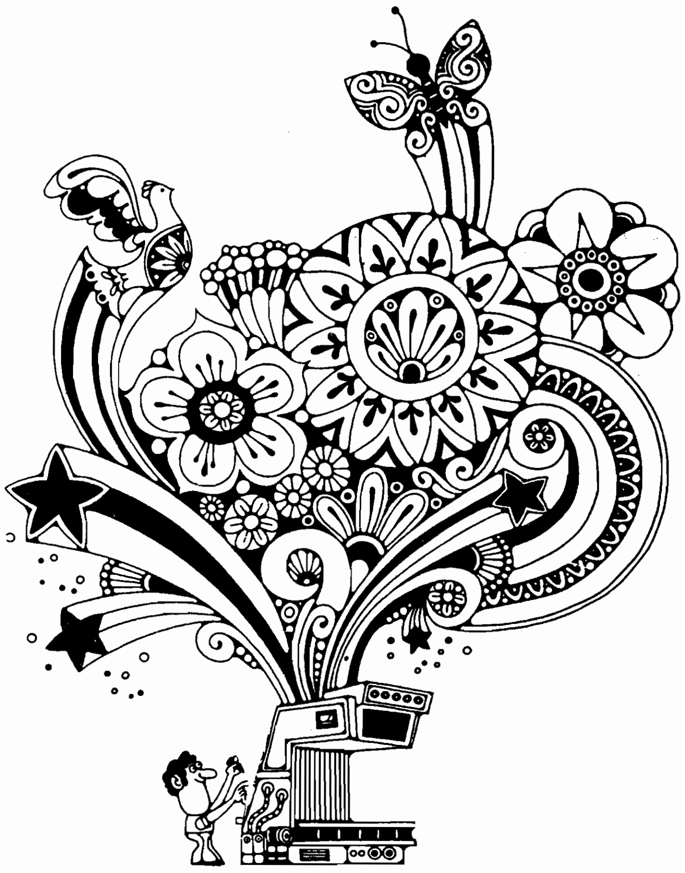
North Star's new CP/M 2.2b is capable of operating octal and capable of operating in an environment that has the controller located within the CP/M. That is, with the controller at E800H, the CP/M 2.2B might be set up to use memory before and after the memory ascribed to the controller.

The new CP/M will make it possible to use an additional 4 K that is not accessable through the present CP/M.

To the best of our knowledge, the North Star D.O.S. is not yet adapted to handle the octal.

North Star is said to be using the Tandem drive for octal use.

We have no information on price at this time.



we use to produce sounds that others understand. But the hardware of speech is not sufficient for shared understanding - a common language is also required.

Wanted: A Standard Code

Computers also need a language for their internal communication. There are several methods of representing data internally in a computer and it would be advantageous if there were some standard language that computers could use for communicating with other equipment.

In addition, it is important that such a language be compatible with human communications since some of the devices the computer will be talking to will in turn talk to

THE I/O FARM

Steve Leibson
4040 Greenbriar Blvd.
Boulder, CO. 80303

Last issue, I started this series on my favorite soapbox, RS-232 connections. Though not introductory, I hope I got your attention. This time, I want to do a little tutorial so we all are using the same jargon. In this episode, we will cover character codes, a topic that is at the very heart of the I/O farm. (Ol Steve Leibson had a farm, EIE I/O)

Language is quite possibly the most powerful component of civilization. People could not purposefully organize without a shared language. Furthermore, the roots of all major human languages are verbal rather than visual.

Speech, our verbal use of language, would not be possible without the evolutionary heritage all people share that has produced our marvelously complex vocal tract, with lips, teeth, tongue, larynx and the other organs people. Printers and CRT terminals are good examples of this type of equipment.

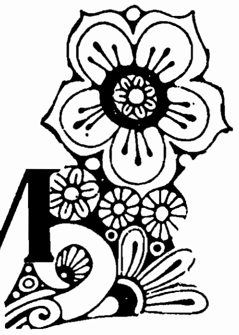
History of Codes

The problem of creating a code, or computer language, that corresponds to an alphabet existed before there were electronic computers. Even before electricity was harnessed for communications, man-made devices such as flags and semaphores were being used to send messages over distance.

Samuel Morse perfected the first code for electrical data transmission, the Morse code. This set of dots and dashes is capable of representing the English alphabet and Arabic numerals so that intelligent messages may be sent between remote stations. Early in this century, interest developed in replacing human telegraph operators

CONTINUED

THE I/O FARM



with machines. Morse code was not used due to the difficulty in machine decoding the varying length of the characters.

The idea of a standardized code was retained however. The dots and dashes evolved into bits. A bit may be either a zero or one, and may be represented by either the presence or absence of an electrical signal.

The first code to use bits for machine communications that was widely used required five bits to encode the alphabet. Five bits allowed thirty-two characters to be represented. Since English uses twenty-six characters and at least ten numerals are needed, it would seem that there are not enough character codes to go around. Add in punctuation marks and the problem gets worse.

Shift Codes

The problem was solved by designating two of the thirty-two codes as shift codes. One was called Letters and the other Figures . These shift codes do not represent printable characters but cause one of two sets of thirty-two characters to be activated. Receipt of a Letters code causes all following codes to be interpreted as letters of the alphabet. Receipt of the Figures code causes all following codes to be interpreted as numerals and punctuation marks. The Letters and Figures codes are common to both character sets.

Such special codes are called shift codes because they cause the receiver to shift between different sets of characters. There were two five-bit codes in wide use, called the Murray and Baudot character codes.

These codes were very similar in concept. They vary only in some of the code assignments to characters. The existence of two competing codes led naturally to the first I/O incompatibility problems.

CONTINUED

GOOFS

Corrections/additions to programs by R.Beaver in the Compass

LIBUTIL("FIND IT") Vol.1 No.1

```
Line 340 change to: 340 A1=4
Line 695 new line:  695 F$="        ";REM 8 SPACES
Line 780 change to: 780 IF EXAM(L1+10)=2 OR EXAM(L1+10)=4 THEN
790
LINE 965 change to: 965 N2=16*A2-1; IF EXAM(L1+12)>127 THEN
N2=32*A2-1; RETURN
```

DOUBLE DENSITY SYSTEMS REQUIRE 32K (8-40K) AND MEMSET 36600.
Also change LINE 300 to: 300 M=36600:P9=5:P8=6

MYLIST Vol.1 No.2

```
Line 915 change to: 915 IF B1<1 OR B1>R-LEN(S$)+1 THEN 990;
B2=B1+LEN(S$)-1
```

THE I/O FARM

A Flaw

Character codes that rely on shift characters for proper operation can be troublesome because proper interpretation of incoming codes relies on the previous history of the message. Unless the receiving device knows which character set to use, there is a 50% chance of erroneous decoding.

Clearly five bits were not enough. A code capable of represeting all printable characters and which did not rely on previous transmissions for proper decoding was needed. In addition, some sort of expandability was needed to prevent another dead-end system.

Modern Codes

By the time the need for this new code was felt, the technology capable of supporting more complex codes as available. Many manufacturers were building equipment that could potentially use a new character code. Whenever the need arises for standardization, there are two ways such standards come into existence.

A single manufacturer can simply go and invent a solution, expecting the rest of the industry to follow. This is what IBM did. They created the EBCDIC (Extended Binary Coded Decimal Interchange Code) character code. EBCDIC is an eight-bit code allowing 256 characters to be represented. Since there aren't that many printable characters, several codes in EBCDIC aren't used.

No other manufacturer chose to follow IBM's lead. Maybe no one could pronounce the name of the new code.

ASCII

The other method of obtaining a standard is through compromise in a committee. Other manufacturers did meet and produced a national standard called ASCII (American Standard Code for Information Interchange) and pronounced "Ass-Key".

ASCII is a seven-bit code. It is formally known as ANSI standard X3.4-1977. One hundred and twenty-eight characters are represented in ASCII. Not all of them are printable. Included in the character set are all the letters of the alphabet, both upper and lower case, the ten numerals and punctuation marks such as the period and question mark.

The non-printable codes are control codes , so called because they control the operation of the receiving device. Carriage return and line feed are control codes familiar to anyone who has used a typewriter. Other control codes include form feed, bell and horizontal and vertical tabs. These control codes were clearly set up for printing or display devices although some manufacturers have pressed the control codes into service for all manner of special functions.

Finally, as in the five-bit codes, there are codes which control how a receiving device will interpret subsequent codes. There are two shift characters called **shift in** and **shift out** . These are used to shift between ASCII and some other character set (English letters aren't the only kind you know). ANSI standards

CONTINUED

THE I/O FARM

X3.41-1974 and X3.64-1979 expand the definition of the escape control code for even greater flexibility.

There are also control codes that delimit text such as start of text and end of text . These codes are used primarily in block transmissions.

ASCII has been a very successful character code. Thousands of instruments and computer related products use this code for interfacing. Even IBM now offers equipment that understands ASCII.

Almost any hardware interface can handle ASCII transmissions. The exception is the BCD interface designed to handle four-bit BCD codes.

Planning the Escape

The planners of ASCII tried to forsee as many different applications as possible. That is the reason for including the various control codes. They recognized that technological advances could not be well predicted and they therefore gave themselves an escape clause.

As was mentioned above, one of the ASCII characters is the escape character. It designates that codes which follow have special meaning.

The intent in creating the escape sequence was to extend the range of the character set by allowing selection from a range of available sets. Graphic characters, nationalized character sets and special applications sets have been developed. Escape character sequences allow for a much richer variety of printable symbols than the simple shift in/shift out scheme of the five-bit codes.

The now common CRT terminal has provided the escape sequence its widest application however. The

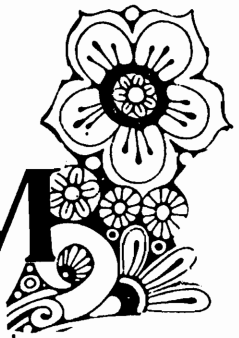
CONTINUED

Bits				b7 b6 b5	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b4	b3	b2	b1	COLUMN / ROW	0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[	k	{
1	1	0	0	12	FF	FS	,	<	L	\	l	|
1	1	0	1	13	CR	GS	−	=	M	]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	_	o	DEL

Mnemonic and Meaning[1]		Mnemonic and Meaning[1]	
NUL	Null	DLE	Data Link Escape (CC)
SOH	Start of Heading (CC)	DC1	Device Control 1
STX	Start of Text (CC)	DC2	Device Control 2
ETX	End of Text (CC)	DC3	Device Control 3
EOT	End of Transmission (CC)	DC4	Device Control 4
ENQ	Enquiry (CC)	NAK	Negative Acknowledge (CC)
ACK	Acknowledge (CC)	SYN	Synchronous Idle (CC)
BEL	Bell	ETB	End of Transmission Block (CC)
BS	Backspace (FE)	CAN	Cancel
HT	Horizontal Tabulation (FE)	EM	End of Medium
LF	Line Feed (FE)	SUB	Substitute
VT	Vertical Tabulation (FE)	ESC	Escape
FF	Form Feed (FE)	FS	File Separator (IS)
CR	Carriage Return (FE)	GS	Group Separator (IS)
SO	Shift Out	RS	Record Separator (IS)
SI	Shift In	US	Unit Separator (IS)
		DEL	Delete

In this table of ASCII characters, the most significant three bits are shown at the head of each column, in both binary and hexidecimal. The least significant four bits are shown for each row. To determine the ASCII code for an upper case H, use the most significant three bits of column four, 100; and the least significant four bits from row eight, 1000; to form the binary code 1001000. Columns 0 and 1 are non-printing control characters. The rest are printable except for the last character, DEL, which is the delete character.

THE I/O FARM



inclusion of microprocessors in terminal design has greatly augmented CRT capabilities. The serial communications link to these terminals has not changed in years. One data channel to the host is all that is available.

Ordinarily, any characters received via this channel are displayed on the terminal screen. But capabilities for character and line deletion, display enhancements such as inverse video, underlining and blinking, and even control of tape drives built into the terminal were not foreseen in the ASCII standard. The escape sequence allows manufacturers to compatibly add these system functions.

Creativity

Manufacturers of CRT terminals are now adding increased performance to their products with escape sequences. Unfortunately, there was little standardization of these sequences until the X3.64 standard came out in 1979. Without standardization, designers felt free to exercise their creativity. They created their own standards.

For example, one major feature now found on most CRT terminals is absolute cursor positioning. The computer can send a command to the terminal which will place the cursor anywhere on the screen. This capability is important for many types of form- filling operations. There are about as many escape sequences to do this task as there are terminal manufacturers. The Digital Equipment Corporation (DEC) has several, depending on the vintage of the terminal. The new DEC terminal, the VT100 (along with a host of VT100 imitators) conforms to the X3.64-1979 standard and is one of the few terminals available today that does.

How does an escape sequence such as cursor positioning work? The host computer starts by sending the escape character. This is followed by one or more characters that indicate the type of escape sequence. Two or more characters then follow giving the desired X and Y coordinates for the cursor. Some escape sequences self-terminate. When the receiver has received the right number of characters, it exits the escape sequence mode and returns to normal mode.

Self terminating escape sequences don't meet ANSI standards. The X3.41 and X3.64 standards are quite explicit about what kinds of characters can appear in escape sequences. All legal escape sequences have proper terminating characters. This allows for variable length escape sequences.

Note that characters received in an escape sequence are not interpreted as printing characters but as control information. The escape character has the effect of making all character codes available for control of a device.

Code Conversion

The vast majority of computer equipment available today uses the ASCII character set. Unfortunately some older equipment does not.

Interfacing these older devices may require that the ASCII characters the computer would like to send be converted to the character codes the peripheral would like to see. Here we assume that the hardware interfacing requirements have already been met. In addition, some modern devices

CONTINUED

GROUPS

LOCAL NORTH STAR USER GROUPS
::::::::::::::::::::::::::::

Organizations interesed in listings their meeting locations, etc to attract new members, should drop a line to the editor of the COMPASS for publication here. The COMPASS currently reaches about 800 INSUA members. Publication is free ofcourse.
::::::::::::::::::::::::::::

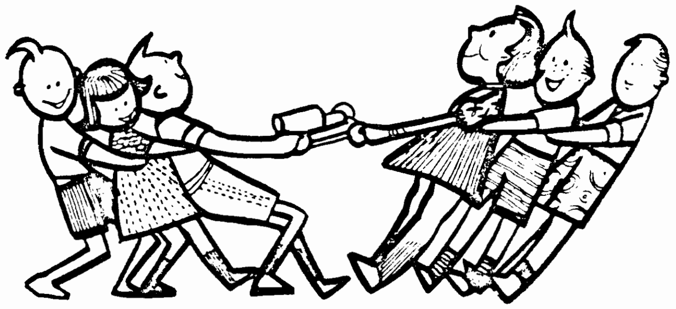
AUSTRAILIA, 80AT, Bill Bolton, PO Box 80, Newport Beach, NSW,2106 Australia

::::::::::::::::::::::::::::

MINNEAPOLIS-SAINT PAUL North Star Users, 2399 Bourne Ave, Saint Paul, MN 55108 David Duggan (612) 646-1192

::::::::::::::::::::::::::::

BOSTON COMPUTER SOCIETY, North Star Group, Gary Saxton, (617) 877-9497, Three Center Plaza, Boston, MA 02108 -Software Librarian B.J.Thompson % Sperry Reseach Center, 100 North Rd. Sudbury, MA 01776



LOS ANGELES I need to find a North Star User group in the LA airport area. Can you help?
Rod Davis, 2209 Voorhees, Redondo Beach, CA 90278

LOS ANGELES I'm interested in joining and participating in a local group (Los Angeles area). I am retired and doing Diesel Eng. Consulting.
A.J.Horner (213) 780-9378

::::::::::::::::::::::::::::

KINSTON, NC I am interested in setting up a local N* users and would greatly appreciate information on other groups
Bradford Rice Jr.
2805 Murray Hill Rd
Kingston, N.C. 28501

THE I/O FARM

may have odd requirements that can only be met through code conversion. An example is a printer that automatically inserts a line feed after receiving a carriage return. Unless the application calls for double spaced printing, the printout will not be as desired since most computers send both carriage return and line feed when desiring a new line to print on.

One solution to this problem is to have the computer convert all line feeds to non-printing characters, such as nulls . Nulls are supposed to be ignored by receiving devices. They serve as time wasters to help in synchronizing fast computers and slow peripherals. A null is ASCII character code zero.

Character codes are yet another source of incompatibility in the world of I/O. Just as differences in language can create communications problems for people, character code incompatibility can render an otherwise operable interface useless.

Peter Jonas, RR#1, Box 814
Munising, Michigan 49862

64k FOR $100

Well, not quite, but I couldn't fit a whole paragraph of description into one line. Anyway, it makes a catchy title, doesn't it? What I HAVE done is convert a N* 16K RAM board (type RAM 16-A3) into a 64K board for just under $101. The breakdown on that cost is $96 for the new RAMS and $5 for miscellaneous parts.

To accomplish this, I have removed the old 4027 RAMS (4K x 1 with 200 ns. access time) and substituted 200 ns. 4116 RAMS (16K x 1). The cost of the latter has dropped steadily in the past year because of their widespread use in TRS-80 and Apple computers. Several suppliers are currently offering them at $3 per chip, or $96 for the set of 32 needed for 64K of RAM. Add an extra $12 for the 4 parity RAMs, if desired (but see discussion below on parity).

Virtually all characteristics of the 4027 and the 4116 are either identical or quite comparable, with the following three exceptions:

1. Since the 4116 is 16K x 1, it holds 4 times as much information as the 4027;
2. Pin 13 of the 4116 must receive useful address information rather than just being a chip enable;
3. The outputs of the 4116 are NOT latched.

A FREEBEE: If you are NOT interested in parity, a 4116 chip can be directly substituted for a 4027 in the RAM board under discussion. The prices of the two chips are roughly the same as of this writing.

The new RAMs are inserted with each pin 13 bent upwards. The network of pins 13 is connected by means of Molex pins soldered to PC foil. Prying out the old RAMs, bending all the pins 13 upwards on the new RAMs, and inserting them into the PC board is a straightforward set of tasks, but it requires a reasonable amount of time and patience.

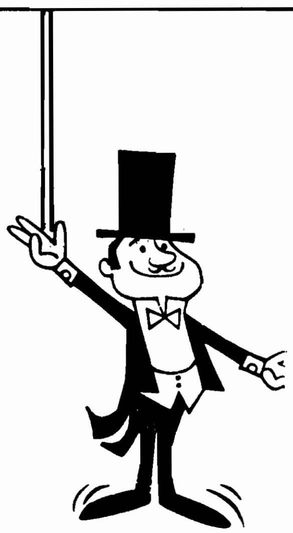
In addition, there is a plug-in module which is inserted into 3 sockets on the existing board. The module consists of mostly DIP sockets and headers, wires and jumpers. Two new IC's are used; a small PC board of the garden variety is used for one of these. Total cost for the module's components is roughly $5. Construction time for the module should run from 1 to 2 hours. Again, patience is the keyword.

CONTINUED

LIST CPM DISKS

By Pavel Breder

This short machine language routine permits you to read CPM disk directories while operating under standard DOS. It can be assembled to run anywhere in protewcted memory.



```
DOS EQU 2000H
COUT EQU DOS+0DH
CINP EQU DOS+10H
CONC EQU DOS+16H
DLOK EQU DOS+1CH
DCOM EQU DOS+22H
DENS EQU DOS+2FH
CR EQU 0DH
LF EQU 0AH
 MVI A,0FFH
 STA COLM
START LXI H,MSGW
PRINT MOV A,M
 ORA A
 JZ PRIN1
 CALL OUTPT
 INX H
 JMP PRINT
PRIN1 CALL CINP
 CPI 3
 JZ CRLF
 MOV C,A
 ANI 0FH
 MOV B,A
 CPI 5
 JNC TEST
 CPI 1
 JNC FINE
TEST LDA COLM
 INR A
 JZ PRIN1
FINE MOV A,B
 PUSH PSW
 MOV A,C
 CALL OUTPT
 POP PSW
 STA DRIVE
 STA COLM
 LXI H,BLANK
 CALL DLOK
 CALL CRLF
 LDA DENS
 CPI 80H
 JZ ISDOB
 XRA A
ISDOB STA DENST
 ORA A
 JNZ DOBLE
 MVI B,1
 LDA DRIVE
 MOV C,A
 MVI A,8
 LXI H,30
 LXI D,BUFFR
 CALL DCOM
 JMP SINGL
DOBLE LXI H,20
 LXI D,BUFFR
 CALL GETIT
 LXI H,25
 LXI D,BUFFR+200H
 CALL GETIT
 LXI H,21
 LXI D,BUFFR+400H
 CALL GETIT
 LXI H,26
 LXI D,BUFFR+600H
 CALL GETIT
SINGL MVI A,65
 STA TOFIL
 JMP COMEH
GETIT MVI B,1
 LDA DRIVE
 MOV C,A
 LDA DENST
 ADD C
 MOV C,A
 MVI A,1
 JM DCOM
 INR A
 JMP DCOM
COMEH LXI H,BUFFR
 JMP MES3
NEXRC CALL CONC
 JZ START
 LDA TOFIL
 DCR A
 JZ START
 STA TOFIL   ;UPDATE FILE COUNT
 MOV A,M
 CPI 0E5H
 JZ SKIPT   ;YES
 INX H
 MVI C,11   ;SET FILE NAME LENGHT COUNT
GETNM MOV A,M   ;GET LETTER IN B
 CALL OUTPT   ;PRINT THE CHARACTER
 INX H  ;INCREMENT CHARACTER POINTER
 MOV A,C
 CPI 4
 JNZ GETX
 CALL SPACE
GETX DCR C
 JNZ GETNM
 MOV A,M
 ORA A
 JNZ GONX1
 CALL SPACE
 CALL SPACE
 JMP GONX6
GONX1 PUSH PSW
 MVI A,'+'
 CALL OUTPT
 POP PSW
 ADI 48
 CALL OUTPT
GONX6 LXI D,20
 DAD D
 LDA COLM
 DCR A
 STA COLM
 JZ MES3
 CALL SPACE
 CALL SPACE
 JMP NEXRC
MES3 CALL CRLF
 MVI A,4
 STA COLM
 JMP NEXRC
SKIPT LXI D,32
 DAD D
 JMP NEXRC   ; NEXT FILE
CRLF MVI A,CR
 CALL OUTPT
 MVI A,LF
 JMP OUTPT
SPACE MVI A,20H
OUTPT MOV B,A
 XRA A
 JMP COUT
MSGW DB CR,LF,CR,LF,'CP/M DIRECTORY BY PAVEL BREDER
 DB CR,LF,CR,LF,'CP/M DRIVE (CTR/C TO END) :',0
BLANK DB 20H,CR
DRIVE DB 0  ;CP/M DRIVE
TOFIL DB 0  ;FILE COUNT
DENST DB 0  ;DISK DENSITY
COLM DB 0  ;COLUM COUNT
BUFFR DS 2048
```

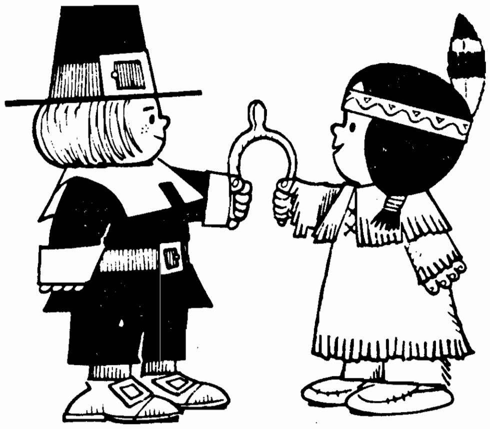
64k FOR $100

The only connection to the PC board itself is made at the 'PH' point for PHANTOM. THERE ARE NO TRACES CUT ON THE PC BOARD, NOR IS IT MARRED OR DAMAGED OR DEFACED IN ANY WAY.

The GOOD NEWS:

I have preserved all signals present on the original board, such as OCCLUDE, PORT-MATCH, and PARITY ARM. I have added EXCLUDE to allow for operation at less than 64K. I am currently running my own board at 60K, EXCLUDing the region at E000-EFFF; this neatly bypasses the Disk Controller ROM at E800, my auxiliary SSM-VB1 memory-mapped video board at EC00, and the PROM option on the HORIZON's Z80 board, which is activated at E000 in my setup. (Many variations of EXCLUDE are possible, allowing for operation at 60K, 56K, 48K, etc.)

All 60K have been tested repeatedly using the N* TM


PLEASE MAKE IT WORK THE FIRST TIME

memory test from the MONITOR, version 5.0; to date, there hasn't been even a single recorded error, which has me somewhat ecstatic. When running BASIC programs, all appears

When running BASIC programs, all appears normal; again, no news is VERY good news. (All material that I have submitted to this INSUA issue was prepared using the new 64K arrangement.)

The BAD NEWS:

Although I have preserved all the mechanics of the parity test, this particular feature of the original design makes use of the LATCHED nature of the 4027 outputs and consequently gives erroneous results when the UNLATCHED outputs of the 4116 are encountered. In particular, the LED glows cherry-red at all times. The signal PARITY ARM, however, gives an accurate reading. For my own purposes, I can live without the parity feature; indeed, I did so for 2 years. Besides, any other way of upgrading my system to 64K would set me back at least $500.

I haven't given up on the parity test, just relegated it to the back burner. Perhaps some ideas could be borrowed from TRS-80 or Apple. Please allow a little leeway in the above discussion in case I have a misconception of the actual problem. The PONDERABLES: I am not a hardware expert, so I cannot vouch for some of the obscene glitch-type possibilities that could crop up. However, I am 100% satisfied myself, to date.

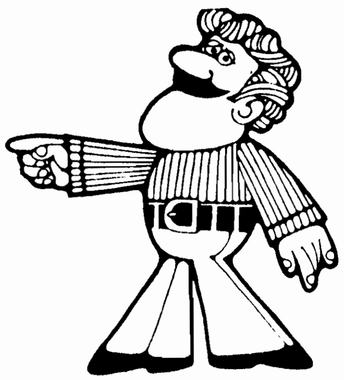
The modification uses 2 new IC's, which might be straining the capacities of the existing power supplies; I just don't know. I am running my board with all chips for the parity option in place, and have encountered no problems. Anyway, there are several ways to adjust if power supply problems should crop up.

The BOTTOM LINE:

No traces are cut on the original board, nor are any changes made on the PC board

CONTINUED

MOVE IT



MOVE YOUR DISK CONTROLLER FROM E800

Feel straight-jacketed by your Double Density North Star controller sitting at E800, limiting what could be a 64k machine to only 56k? Well....MOVE IT!!

The controller can be moved up to FC00 at the top of memory by simply plugging in two new prom chips. That will give you 62k of memory space for the largest business programs.

Bob Hogg, designer of the relocatable...and N* compatible disk controller board will make the necessary hardware available to INSUA members for $35.

After substituting the two new chips, you'll have to modify your DOS with N*'s mover program (on INSUA 5.2 DOS disk) to recognize the new controller address. Bob will,hovever, supply INSUA members with his own N* compatible disk operating system on disk....FREE with the proms. His DOS automaticaly searches for the disk controller whatever its address on boot-up and patches itself. Also included are a complete set of disk utilities.

Lifeboat's CPM also automatically searches for the relocated prom and patches itself.

To get the proms and free disk send your $35 check directly to Bob at MICRO COMPUTER DEVICES, 25651 Minos St., Mission Viejo, CA 92691 (tel:714-770 2168)

NOTE: The proms are plug compatible with N*'s and set up the board at FC00 as furnished. If you want to move the board around at will to different addresses, a third chip is required. Bob will supply that one for an extra $15.

64k $100

itself. If you don't like the results, you have only to remove the 'PH' wire, put the original RAMs and IC's back in place, and try to peddle the 4116's to your TRS-80 buddies.

I am preparing a set of wiring instructions plus theory that I'll send to interested persons for $10 plus a self-addressed, stamped envelope. As long as my supply holds out, I'll also include the single-IC PC board as well as the small section of PC foil that is needed. Anyone attempting the conversion might well recover most of the cost by selling the set of 4027 RAMs that is removed from the original board. I know that my modification works on the board marked RAM 16-A3.

I would speculate that the same treatment should work on versions RAM 16-A2 and/or RAM 16-A, but I just won't know until somebody tries it. Converting the N* 32K RAM board to 64K (without parity) appears to be even easier, since the proper signals are already present at pin 13 of the RAMs; this conversion is also unverified at present.

DISK LIB

LIBRARY NEWS

The INSUA Library now offers seven discs which should be useful to most members and which are available at reasonable cost.

The ASSEMBLER disc includes both the single density and the double density adaptation of the self contained system (assembler/editor) originally released through Processor Technology and now adapted for NORTHSTAR.

The #1001-DOS holds the 5.2 single density relocatable NORTHSTAR DOS and #1002-DOS DD holds the double density version. This makes it possible to move the DOS to other more appropriate locations.

Each of the three COMPASS discs contains those programs printed in one COMPASS newsletter.

TELSTAR is an assembly language program written by Leonard E. Garcia for transferring named disc files, either single or double density, thru the telephone via a modem. ccording to Garcia, an additional program included on the disc, TELSHARE, is designed to run in high memory and allows a remote user to access and share all inputs and outputs of any program running under DOS with the local user. Included on the disc is a program by Pavel Breder, MODEM, which makes it possible through the use of a modem to tie into Bulletin Boards and other informational networks as well as interact with other users.

The #1001 disc, DOS and the #1002 DOS DD are each $10. The others, #1003-TELSTAR, #1004-COMPASS1, #1005-COMPASS2, #1006-ASSEMBLER, and #1007-COMPASS3 are each $15.

Please indicate by number which you want and send check or money order to INSUA, P.O.Box 1318, Antioch, CA 94509. Allow six weeks delivery.

FREED FROM 35 TRACKS

77 TRACK DRIVES & NORTHSTAR
80 TRACK DRIVES & NORTHSTAR
40 TRACK DRIVES & NORTHSTAR
OR.....JUST 35 TRACK DRIVES

By: Joe Maguire
2321 Foxhall Dr.
Anchorage, AK 99504

Many of you have no doubt seen the advertizements for the newer "octal" disk drives. These are drives capable of 96 tracks per inch for from 77 to 80 useable data tracks on each side of a 5.25 inch diskette.

You probably thought you were losing out as NorthStar's DOS does not support this new technology. Worry no more! With a simple patch to DOS 5.2DQ you can take full advantage of the increased track capability.

I recently obtained a pair of 40 track Wangco drives at a flea market and decided to see what could be done with the DOS to make use of the extra disk capacity. A short session with a disassembler and a debugger gave me the answers. It's a shame that NorthStar doesn't provide the source for their DOS. There are hundreds of users, I'm sure, who could provide all sorts of innovations (free of charge!) to make NS software more desirable. The DOS is so hardware dependent that no one could possibly use it for any other purpose than with a NS controller. Oh well, paranoia is easy to come by when you think everyone out there is trying to steal your software.

To make the patch to DOS, enter the code given in listing 1. If you don't have DOS 5.2, (available from INSUA) you are going to have to search around for the correct addresses.

With a NorthStar double density controller and a double sided 77 track drive, you will now have almost 790K bytes of storage per disk. Who needs a hard disk?

These patches were tested using my 40 track drives and by a friend who has a pair of 80 track Teac drives. One limitation is that the Teac (and other 96 TPI drives) cannot read correctly a 35 track disk written with the standard DOS. This is because the controller only knows to issue a certain number of step pulses. It doesn't know the head has only moved half as far across the disk surface. This could be compensated for in software which is left as an excercise for the user. It is possible to transfer files from a 35 track disk to a high density disk by having at least one 35 track drive in the system.

The patch was tried in the BIOS of a Lifeboat version of CP/M with mixed results. CP/M seemed to recognize the 40 track patch OK but became confused when patched for the 80 track Teac. This is no doubt a result of Lifeboat's adaptation of the DOS into CP/M and should be correctable through software.

CONTINUED

35 TRACKS

As an interesting sidelight, I had always wondered how NorthStar compensated for different processor speeds when performing time dependent disk functions. They do it in a clever way. Beginning at 771H bytes from the DOS origin, (DOS 5.2) is a routine which reads the controller status. The routine enters a loop until the status is available. Each time around the loop a counter is incremented. When the status finally arrives from the controller, the loop count is directly proportional to the speed of the processor. (The time for the status to arrive is a function of the clock speed of the controller, not the host computer) This count is then stored for use in functions like controlling the time for stepping the head from track to track. Neat!

For those of you who are using drives with track-to-track stepping speeds other than the 40 ms (slow) or 5 ms (fast) supported by DOS, the step delay loop begins at DOS Origin + 182H. The value (38H) at DOS Org + 186H can be "trimmed" to provide intermediate step rates.

EDITOR'S NOTE:
If you envy North Star users who have new fast stepping drives on their quad systems...you need envy them no longer.

Using Joe's delay loop adjusting idea even works on old 35 track drives. First, set the config byte in double density, DOS ORG+34 to make DOS think you have fast drives (see N* documentation furnished with INSUA 5.2 DOS disk.) Then change byte at DOS ORG+186. We used B0 Hex and got much faster disk activity. Note, however, if you set this byte too low for the condition and age of your hardware,DOS will lose track of the head and bomb out so that you will have to reboot before you can try further adjustments. Make sure you have the read after write option at ORG+2B on during your tests.

It is also posible to make your good old Shugart 35 track drives think they are 40 track machines as outline in Joe's source code listing. BUT THIS CAN BE DANGEROUS.

It's simple enough to put in the patches to your DOS so that the drive head writes out to disk address 399 on the disk. This gives you about 15 percent more storage per disk side.

The problem arrises with BASIC and other programs which check the config byte and will not CREATE or NSAVE files beyond the regular limit unless they think you have the wrap around feature of a double headed (QUAD) drive. Fool Basic into thinking you have QUAD's by setting the high bits of the config byte as per North Star's documentation and it will happily write to your "new" 40 tracks....if the file is longer than 399 blocks it will also happily wrap around to nowhere and contine to write to disaster.

Fortunately, Basic will read the longer files without checking the high bits of the config byte so you can safely leave them "0" and use the extra space if you preCReate your data files under DOS.

CONTINUED

35 TRACKS

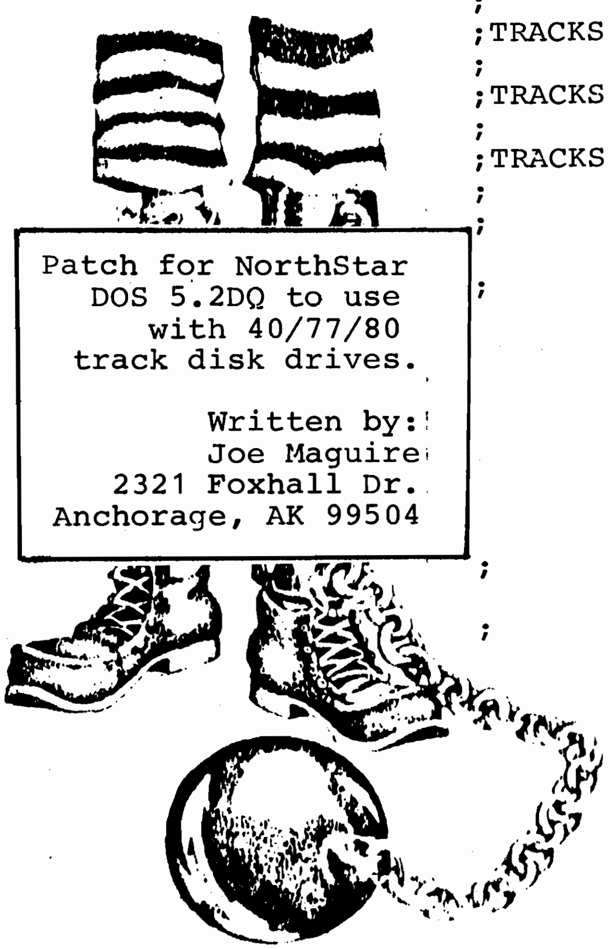
Adjusting CPM requires the same two patches but it is a bit more complicated. Lifeboat's adaptation of CPM has a DOS very similar to North Star's but the code is unfortunately scattered in different locations in diferent versions of their CPM 2x. It's easiest to use their SETDRIVE Utility to make CPM think you have fast stepping hardware. This sets the config byte according to the North Star convention above. You must then use your monitor to seach the code for the fast step timing loop. Look for:

address1	16
address2	38
address3	15
address4	C2
address5	ADDRESS3

Change the 38 at address2 to your hardware dependent delay. We use BF. (In CPM2.22 the config byte is at Warm boot address + 6F5 and delay byte at Warm boot + 64B).

EDITORS NOTE: See Bill Banaghan article on North Star's new official 80 track drives. They furnish their compatable version of CP/M for these drives. Any member out there interested publishing the patches for Lifeboat's North Star drivers in the COMPASS?

You'll want to read Peter Stark's "Firm Up Your Floppy with 800K" in August 1981 Kilobaud Microcomputing for a excellent explanation of disk hardware including explanations of why single density drives work with double density and why "single" density disk media that will not work on double will not work on single density drives.
C.S.



Patch for NorthStar
DOS 5.2DQ to use
with 40/77/80
track disk drives.

Written by:
Joe Maguire
2321 Foxhall Dr.
Anchorage, AK 99504

```
;
DOS:     ORG     0100H             ;Standard Origin 5.2
;
;TRACKS: EQU     80                ;For 80 track drives
;                                  ;
;TRACKS: EQU      77               ;For 77 track drives
;                                  ;
;TRACKS: EQU     40                ;For 40 track drives
;                                  ;
;
         ORG     DOS+366H
;
         PUSH    PSW               ;original code
         PUSH    H                 ;        "
         PUSH    D                 ;        "
         PUSH    B                 ;        "
         MVI     A,TRACKS-1        ;<==== patch
         SUB     B                 ;original code
         JNC     DOS+377H          ;        "
         ADI     TRACKS            ;<==== patch
;
         ORG     DOS+44AH
;
         DCR     D                 ;original code
         RLC                       ;        "
         JNZ     DOS+44AH          ;        "
         MVI     A,TRACKS*2        ;<==== patch
         LXI     D,-TRACKS*20      ;<==== patch
         RC
         RRC
         LXI     D,-TRACKS*10      ;<==== patch
         RET
```

LETTERS

WANTED: A SCREEN 'CATCH' SUBROUTINE

To Anyone out there

It would be quite useful for several applications of mine to be able to save the contents of the CRT screen in individual ASCII files on diskette. This seems quite simple in theory, since my Intertube II terminal has a screen dump command which routes all 1920 characters on the CRT screen through the auxiliary serial port. In addition, the HORION has a second serial port which could be used to 'catch' or receive the transmission. However, what seems simple in theory has so far evaded me in practice. One major stumbling block is that the Intertube does not provide any handshaking signals. Here is the setup: The HORION is connected to the Intertube via the main port for each machine. A BASIC program that is running on the HORION is generating output on the CRT screen. The BASIC program then jumps to a screen catch subroutine and awaits further developments. The screen dump command is given manually from the Intertube's keyboard. The 1920 ASCII characters on the CRT screen are sent from the Intertube's auxiliary port to the HORION's second port, without handshaking, i.e., in a single burst. The subroutine stores the transmission in RAM and then SAVEs it in a pre-named disk file. The subroutine then returns control to the main BASIC program. If anyone has any suggestions or knows of an inexpensive solution to this problem, I would appreciate hearing from you.

Peter Jonas
RR#1, Box 814
Munising, Michigan 49862

INSUA
Has Anyone done any machine code or basic subroutines to utilize the smart video terminal cursor movements and other controls? I have talked to many dealers and none know of any programs that will give a person a clue on how to write one for positioning or other functions. I'm sure many members write masking programs but I haven't seen any in any magazines.

Raymond C. King
915 El Rancho
Pocatello, ID 83201

PS: Your publication was much, much needed. I look forward to your issues.

******

EDITORS NOTE: A cursor positioning subroutine for four different terminals was illustrated in the COMPASS Vol.1 No.2 (CURSOR, Page 23) and could be adopted to most other terminals by changing the control codes as needed by your particular terminal.
Masking of particular keyboard keys to perform special functions including cursor control can be added to your DOS user I/O area (See COMPASS Vol.1 No.1 BETTER HORIZON IO, Page 12) for ideas. Since this sort of thing is usually tailored to individual requirements, you will probablly have to write it yourself or have it written for you. Pavel Breder (415-756 8060) programs custom DOS I/O for North Star users around the country. His fees are usually about $50 depending on the needs of the client.

MORE .....

LETTERS

INSUA

Two issues into my membership, and I think I've already got my moneys worth. Keep it coming.

I have a problem (question) your staff or a reader may be able to help with.

I've got my N* working with a Scitronics controller to handle the house lighting, heat, security, etc. Although I infrequently have power problems, I'd like to restart and run the control program when the power comes back on...that is I want an automatic RESET, GO BASIC, LOAD and RUN.

The 79 N* manual has info on modifying DOS to GO BASIC (auto start). It also shows how to modify Basic for a turnkey operation. I have done both successfuly, but cannot get DOS to go right to a turnkey program sucessfully.

Robert Bucklew
707 Homestead Ave
Havertown, Pa 19083
215-649-2514

ED NOTE: Since you apparently have Basic come up ok from DOS auto start the problem may be in the Basic side of the auto start sequence. See your SOFTDOC and you'll note that you have to SaveFile your modifed copy of Basic TOGETHER with the program it is to run into file that has had its size increased to the total number of blocks for both Basic and the subject program.

The COMPASS has featured a program in issue No. 1 called COMMAND which enabled you to create machine language routines that Dos could load autoimaticly and execute either Dos commands or B-sic Programs without keyboard entry. The program really substitues

INSUA,
Bravo for the I/O Farm!! I have heard that it is possible to send files back and forth between two CP/M equipped computers by conecting their RS-232 ports together and using the PIP function.
How is this done?
This would make a very useful article for the I/O Farm since it would permit software that is available only on 8" soft sectored discs to be transfered to our trusty N*s.
Keep up the good work.

Live Long & Prosper,
Durk Pearson
Spectrum Technology Service
Box 942
Palos Verdes Estates,CA 90274

pre-defind keyboard entry for the normal on line entry. This software allows you to preprogram the keyboard entries that the running program may require during its operation as well.

That may be your problem....does your basic sit there waiting for keyboard entries when it starts?

As to power failure...The guick cheap answer is not to let your computer go down in the first place. Motorcycle or cheap car batteries can be hooked up to supplement your power supply (before the voltage regulation point) to keep things running when the line power goes out. A bleeder resistor in the battery line will keep the batteries under trickle charge when the computer operates normally.

Its a simple hook up that any radio-TV repair shop could do for you.

MORE....

LETTERS

Bob Stek
19 Mayfield Road

Regina, Saskatchewan S4V 0B7
Canada

Dear Editor:

Every so often, I stop and think about what hardware or software features that it would be nice to have on my micro system. Many times, these features are ways of overcoming inherent design problems or are extensions to expand or improve performance of my Horizon. I am a fairly good applications programmer, but I am not very hardware oriented, nor am I a systems programmer. I am sure that many other NorthStar users would appreciate solutions to these problems as well. So I have constructed a "wish list" to share with other NorthStar users who may have already solved a particular prpblem and would be willing to share their solutions through the COMPASS or Dr. Dobb's.

PROBLEM #1 - NorthStar's 32K board is really only a 24K board if you are using it in the top half of memory. That is, you must deselect the upper 8K (from E000 to FFFF) due to the bootstrap PROM at E800 on the disk controller board. This wastes 6K of memory. Has anyone figured out a hardware modification for the memory board to deselect just 2K starting at E800 (or F800, see below)?

PROBLEM #2 - NorthStar's disk controller's PROM at E800 limits you to a maximum CP/M size of 56K. Sure, that's enough for most purposes, but all you need is one program which requires a little bit more, and you soon become envious of people with a full 64K CP/M system.

Possible Solution A : Move the bootstrap PROM to F800. Sure, but that still won't give you a full 64K. And I believe NorthStar now charges $150 for a relocated PROM. How about a "home brew" solution, perhaps with a switch selectable address? (Don't forget a patch for DOS and CP/M to take account of the new address.)

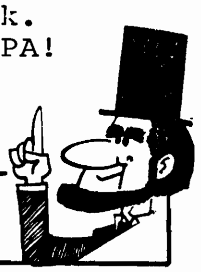
Possible Solution B : Modify CP/M to "go around" the PROM. Perhaps the bulk of CP/M could reside from F000 up, with the TPA ending at DFFF (or E7FF).

Possible Solution C : "Phantom out" the PROM or put it in the second bank of memory (and again patch DOS and CP/M to account for this).

Possible Solution D : Leave the PROM at E800 and put all user memory in the second bank. Possibly you could also have other PROM boards, your floating point board, or memory mapped video displays (like a VDM) in the first bank. This seems like the easiest to implement since it will only require software modifications (assuming you have bank switching memory boards). This also suggests a generalized solution of placing all operating system software in the first bank with user programs going into the second bank. Imagine CP/M with a 64K TPA!

Continued Next Page

CLIP TIPS



HOT LINE USED COMPUTER EXCHANGE

Bruce W. Lnch has established a phone hotline to bring buyers and sellers together for used computer equipment. For a $5 to $10 service charge, The Used Computer Exchange, 2329 Hunters Woods Plaza, Reston, VA 11091 will give buyers current quotes on micro equipment listed with the organization. Sellers pay 5% of their sales price when they sell their equipment. The quotes are available phone toll free: 800-327-9191 Ext61. Lynch can be contacted at 703-471-0044.

MORE .....

LETTERS

Bob Stek Letter
(continued)

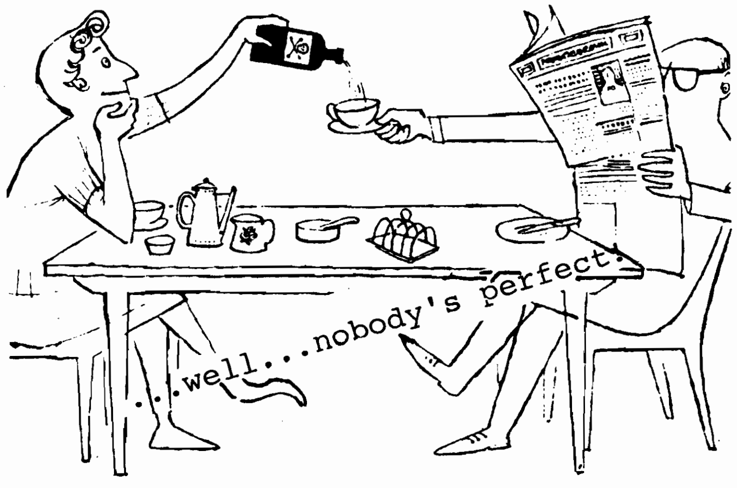
PROBLEM #3 - Most CP/M floppy based systems are disk I/O bound. CP/M could be made faster if larger disk buffers were available. Ithaca Intersystems has the ultimate solution with their "Cache CBIOS" which dedicates the entire second bank of 64K to disk buffers. This improves performance by a factor of 3 or 4.

PROBLEM #4 - The Horizon CPU operates at "only" 4 Mhz. With 6 Mhz. Z-80 chips available, can we speed up our CPU boards with a few modifications?

PROBLEM #5 - NorthStar CP/M allocates 1K blocks if you are using single sided disks, but uses 2K blocks if you specify quad density drives (and therefore wastes a lot more disk space). First, why? What is the trade off? And second, how can you patch it to use 1K blocks?

I hope this gives some of you "hackers" some incentive to share some solutions or to tackle an interesting new project. Just let the rest of us know how to do it too!

...Bob



INSUA

If you can give me information on getting past and subsequent issues of the Northstar (Company) Newsletter, there might be information of help to me, and to get on their mailiong list if they still have one. I would also like to get a version of North Star Basic for the Z-80 if they have ever produced one.

Frank E. Laughlin
3 Dundee Circle
Quincy, IL 63301

ED NOTE: Not much hope on Northstar Inc. newletter, I'm afraid. We have not seen one in ages. That's part of the reason we are here.

There is a Z-80 Northstar like Basic available from Micro Mikes, 905 S. Buchanan, Amarillo, TX 79101 (806) 372-3633. Its available from your dealer for $150 and runs under DOS or CP/M. Its call "baZic" since it is written in Z-80 code. The claim of 25-60% speed advantage is made. It also has expanded the N* basic commands to include terminal control commands like clear screen and cursor positioning saving you the DEF code (see CURSOR in Compass No. 2).

INSUA
I am currently engaged in modifying Tiny Pascal to access disk files. I have written a source file editor that is similar to the kinds of editors that I use at work.

I have started on the design and coding of a link-loader to make the Tiny Pascal project easier.

I would appreciate the following programs in source format:

Disk I/O routines
Floating point routines (N*)
Bootstrap loader routines

Richard Omer
12 Greenview St, #209
Framingham, MA 01701

BASIC SORT PATCH

NORTH STAR BASIC SORT STATEMENT

By Jim Lind

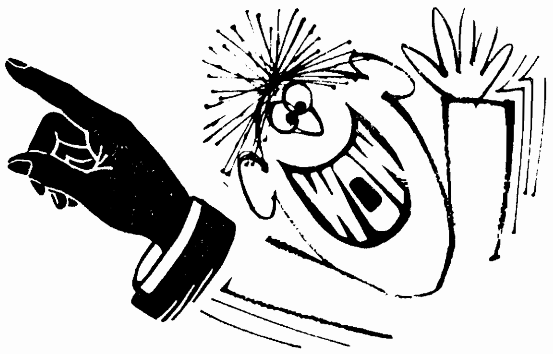
The author of N*BUS, the Global Line Editor for North Star Basic, has released a machine code patch to North Star Basic that allows the programmer to sort one and two-dimension arrays or string data with a single Basic statement. The "LET" statement, which is seldom used, is replaced with "SRT" and the time consuming operations of sort programming are greatly reduced.

N*SORT, as it is called, implements in machine code the Shell-Metzner ex-change sort algorithm. The documentation provides sorting time comparions and states: "N*SORT represents a vast improvement in sorting time over all other techniques implemented in BASIC, and approximately 18 times faster than an equivalent Shell-Metzner sort." Although tests were not performed to verify the performance comparisons provided in the documentation, the following unscientific data is relevant:

A. A relatively large program had a series of sort routines which required an average time of 9 minutes 22 seconds to complete using QUICKSORT. This time also includes reading and writing to disc and number crunching. This same program was modified by (1) changing constants to variables where possible, (2) changing division to multiplication, i.e., multiply times A (A = 0.01) instead of divide by A (A = 100), and (3) moving the QUICKSORT routine from near the end of the program to the top of the program. This reduced the run time to an average of 7 minutes 10 seconds.

B. N*SORT was installed and the same routine required an average of 4 minutes 20 seconds.

C. When QUICKSORT was used, it was apparent that the sort was taking a good deal of time based on



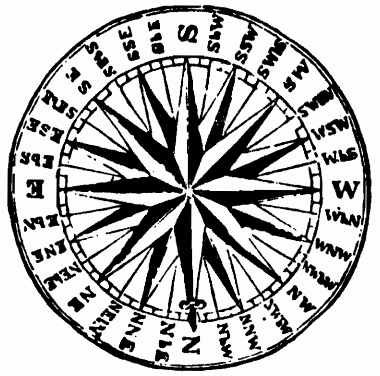
the amount of time the disc drive was idle because data was being read from disc to an array, was sorted and the sorted data was written to file. When N*SORT was used, the disc drive was not idle; therefore, markers were inserted in the program to indicate on the console the completion of various steps. It was found that the total sort time was averaging 10 seconds which meant the disc activity and number crunching required 4 minutes 10 seconds. Since the non-sort activity was the same in all cases, N*SORT appears to have reduced this sort time from 3 minutes to 10 seconds.

N*SORT uses approximately 850 bytes of storage and, as with N*BUS, N*SORT comes in two different versions, TANDEM or ROMABLE. The personalization program, SORTGEN, provided with N*SORT is designed to operate under control of any version of DOS and BASIC, release 4 or later. SORTGEN is independent of DOS and BASIC starting addresses, whether BASIC uses hardware or software floating point routines, the arithmetic precision of BASIC or whether DOS is single or double density. If sorts using BASIC are taking too long, this package appears to be the answer. N*SORT is distributed by SZ Software Systems, 1269 Rubio Vista Road, Altadena, Ca. 91001.

The COMPASS

INTERNATIONAL NORTH STAR USERS ASSOCIATION BOX 1318, ANTIOCH, CA. 94509

BULK RATE
U.S. POSTAGE
PAID
ANTIOCH, CA
PERMIT NO. 265



POSTMASTER: ADDRESS CORRECTION REQUESTED
Please enter key number on form 3579

HELP WANTED

COMMENT FROM THE CHAIRMAN

The various activities and services provided by INSUA are based on the efforts of a number of individuals who give freely of their time for the good of the organization. Because of the geographical spread of the membership, the problem of meeting together on tasks, and the problems of communicating by mail or telephone, it has been difficult for many of the members to become involved in INSUA activities.

Similarly, with the passage of time it becomes increasingly difficult for some of the presently active workers to continue their committment of effort to INSUA in spite of their desire to do so and the rewards of such experience. Therefore, if INSUA is to continue over the years, it must have an orderly transition of responsibilities and work effort from a current group of members to a succeeding group. Presently there is need for help and/or leadership roles with the newsletter, with membership development, with organizing the annual meeting and with other activities.

I would therefore like to encourage those individuals with an inclination to work with INSUA to speak up and let us know how you might be more closely involved and helpful in the providing of services and/or the running of INSUA, your organization.

Bill Banaghan Chairman

9/21/81

VOLUME ONE NUMBER FOUR

The COMPASS

INTERNATIONAL NORTH STAR USERS ASSOCIATION
PUBLISHERS OF THE COMPASS NEWSLETTER
POST OFFICE BOX 1318, ANTIOCH, CA. 94509

FREE

FREE

FREE

DISK IN THIS ISSUE

SEE NEXT PAGE

YOUR 1981 MEMBERSHIP EXPIRES WITH THIS ISSUE TOO..... SEE LAST PAGE TO RENEW

HERE IS YOUR FREE MEMBER'S BONUS

A COMPLETE PROFESSIONAL ACCOUNTING SOFTWARE PACKAGE !!!!

The International North Star Users Association has secured the rights for its members to use the MICRO-COUNT II financial accounting software package. This is a comprehensive commercial software system normally valued at over $600.

The disk is enclosed with this newsletter. Documentation is on the disk and can be printed out by loading basic and runing the DOC-TYPE program.

The disk is furnished to members under an limited licensing agreement with the author T.I.Warshauer. It is NOT public domain software and can NOT be traded or copied for others under terms of the agreement. It is EXCLUSIVELY for use of INSUA members.

Your bonus disk marks the end of your first year's subscription to The Compass newsletter. Elsewhere in this issue you'll find a renewal form for your 1982 membership dues. To assure that you receive the next four issues of the newsletter please remember to send your $20 renewal check NOW!!!!!.

We hope you find the software helpful and hope you find the newsletter, disk library and idea exchange worthwile. Let us have your comments.

****

MICRO-COUNT II

This software set was designed and coded by the author to provide the corner-stone for a series of small business applications packages. The specifications required a set of software that had the following attributes:

1. Menu driven user interface.
2. Flexible chart of accounts format for ease of installation
3. Chart of accounts controlled statement formatting.
4. Complete audit trail, ie. detailed general ledger report.
5. On-line verification of transaction account numbers.
6. The ability to handle multiple cash accounts with unique transaction journals for receipts to, and disbursements from, each account.
7. Simple method of interfacing supporting subsystems.
8. Minimum use of file sorts due to slow sort times on micros.
9. Compact program modules to minimize memory requirements and general overhead.
10. Avoid the use of custom or complex data file access or indexing methods.
11. The ability to handle several companies on the same system.

The programs included in this package represent one approach to reaching these goals. The general design has been carried over into later software releases operating on both North Star DOS and CP/M based systems.

REPORTS PRODUCED:

Income Statement - Provides income and expense information for the current period and year to date. Format controlled by COA type coding.

Income Statement With %'s - Requires 96 column printer

Balance Sheet - Produced for year to date balances in a format controlled by COA type coding.

Supporting Schedules - Reports sub-account detail for accounts with type codes HP or for SP accounts with SX accounts following.

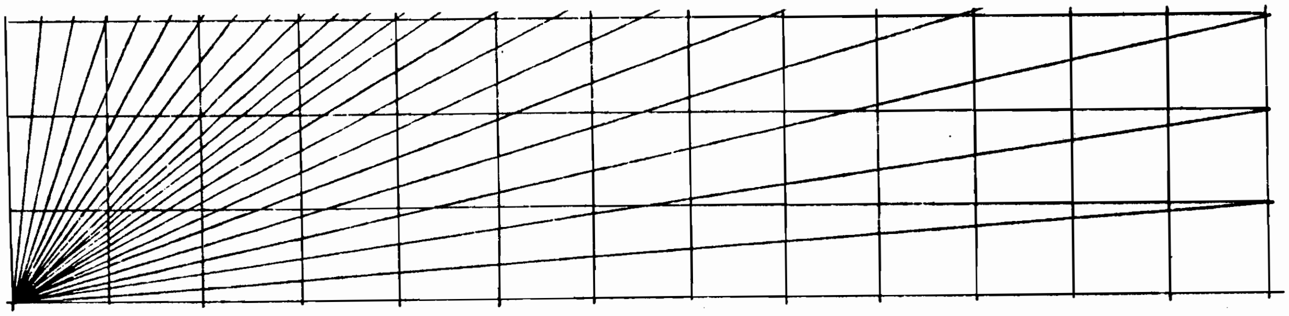
Chart of Accounts - User can select listing of all accounts or for posting type accounts (SP & SX) only. Will display or print as requested.

Journal Edit Reports - Can be produced for each defined cash receipt, cash disbursement, and general journal.

Trial Balance Report - May be run at any time for summary of all transactions processed by GLPOST. Reports both current period activity and year to date balance for all accounts.

General Ledger Report - Provides detailed report of all account activity during current period as well as beginning and endinr balances. The GLPREP program must be run prior to requesting this report.

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

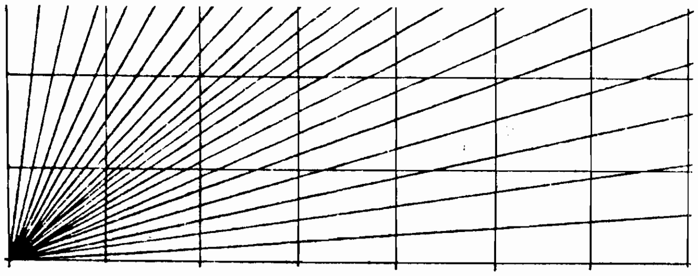


NO NUMBERS

BY Alan H. Nelson & Chris Nelson

Northstar BASIC will accept most keyboard characters: therefore virtually any text can be typed and SAVEd in a BASIC file. If the file is to be RUN, blocks of arbitrary text are normally protected by REM statements, or, if they are to be printed out, are enclosed in quotation marks. Under many circumstances, however, the user intends his file to be LISTed rather than RUN: the LIST-IT option is familiar to members of the INSUA. The problem with LISTing a text program is that line numbers are normally included in the text. One way around this problem is to use a text formatting system like Wordstar or Northword for text and to use BASIC or other sophisticated languages for programs which manipulate variables. Wordprocessing software is expensive, however, and may not be held in common by users who wish to exchange text files. We have devised an extremely simple program which will allow BASIC files to act as text files. In essence, the program will allow the user to LIST a BASIC file to a printer without line numbers. The numbers which occur at the beginning of each line are suppressed. A flag set upon detection of any non-numeric character turns off the stripper until a LINE-FEED is detected: in this way, numbers which occur in the text itself will not be suppressed. Our program was formatted in assembly language using the PDS software devised by Allen Ashley. It occupies 31 bytes in the 2900 page of DOS, specifically from 29C0 to 29DE. The NONUMBER option is turned on with the command FILL 10551,192, and turned off with the command FILL 10551,73. The decimal number 10551 is hex 2937, the low byte of the jump to the CHOUT1 routine (second serial port). After testing a flag in the back-up D register, the program either tests the character and returns for a new character if it detects a number (flag reset), or prints the character if it detects a nonnumber, then sets the flag to print all ensuing characters until it detects a LINE-FEED character, whereupon it resets the flag. Chris has translated the assembly language routine into a BASIC program which can be RUN. We have also provided a DOS with the changes saved so that it is possible to boot up with the NONUMBER program already in place. Efficient text typing in BASIC requires the use of the AUTO command. Once this command is invoked, the keyboard can be used almost like any other typewriter. Hitting CR at the end of each line will feed the next line number and position the cursor at the first available character position. Here the first variation on normal BASIC usage occurs. Whereas convention (including INSUA convention) sometimes requires a space between the line number and the text, the space is not really necessary, even for increased legibility, if the numbers are eventually to disappear; moreover, typing the space at the beginning of each new line interrupts the flow of work, since such a space is not part of

NO NUMBERS



may be EDITed with ease using the BASIC EDIT commands. The EDIT routines in Northstar BASIC are fairly sophisticated, and are very quick once they are mastered. NONUMBER will not of course alter the format of the text, except that it will keep a straight left margin even when line numbers increase from two to three or from three to four digits. NONUMBER will, however, allow Northstar to function as a simple typewriter, and for this reason may be of use to beginners and advanced users alike. NONUMBER will also allow documentation for Northstar programs to be printed out in an esthetically more pleasing format.

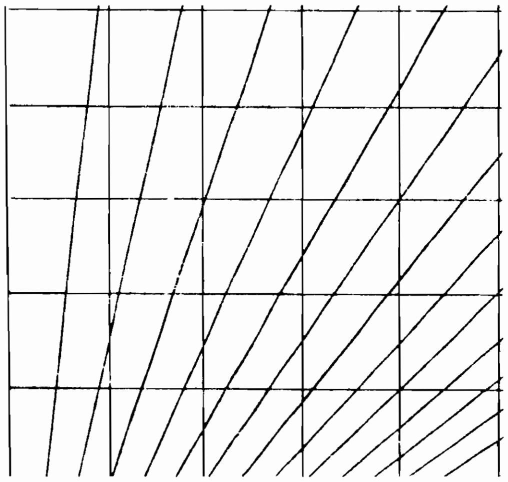
INSTRUCTIONS

If you have a copy of the NONUMBER diskette, boot-up with this diskette, which contains NONUMBER program in DOS. GO BASIC on this diskette, loading in a BASIC which has been changed to preserve the characters ;:(Load NNUM.DOC to test whether the charters are in fact preserved. List to a printer using LIST#1: line numbers should be printed along with text. Give the FILL 10551,192 command and list again to printer. The numbers should have been stripped from the text: notice that the left margin is straight. Give the FILL 10551,73 command and test to be certain that the numbers are once again listed. If you do not have a copy of the NONUMBER diskette, create NONUMBER BASIC program as given above. Use the FILL commands to protect the special characters Then follow the instructions given in the preceding paragraph. NOTE: If you intend to SAVE a text file, be certain that you have LOADed it with a BASIC altered to protect the open Oparenthesis: otherwise it will be changed into an open square bracket.

NONUMBER program by Alan H. Nelson, 2600 Buena Vista, Berkeley CA 94708
(415) 848-1992

```
29C0                 0008 ;NONUMBER PROGRAM
29C0                 0009        ;
29C0 F5              0010        PUSH  PSW     ;SAVE PSW
29C1 78              0011        MOV   A,B     ;CPY B TO A FOR TESTS
29C2 D9              0012        EXX           ;SAVE REGISTERS
29C3 CB 42           0013        BIT   0,D     ;TEST FLAG
29C5 28 0D           0014        JRZ   TSTNL   ;JUMP ON FLAG RESET
29C7 FE 30           0015        CPI   30H     ;TEST A FOR NUMBER
29C9 38 07           0016        JRC   RESET   ;JUMP ON NONNUMBER
29CB FE 40           0017        CPI   40H     ;TEST A FOR NUMBER
29CD 30 03           0018        JRNC  RESET   ;JUMP ON NONNUMBER
29CF D9              0019        EXX           ;INITIATE RET ON NMBR
29D0 F1              0020        POP   PSW
29D1 C9              0021        RET           ;RET FOR NEW CHAR
29D2                 0022        ;
29D2 CB 82           0023 RESET  RES   0,D     ;RESET FLAG ON NONNMBR
29D4 FE 0A           0024 TSTNL  CPI   0AH     ;TEST A FOR NL
29D6 20 02           0025        JRNZ  CHOUT   ;JUMP ON NO NL
29D8 CB C2           0026        STB   0,D     ;ON NL SET FLAG
29DA D9              0027 CHOUT  EXX           ;INITIATE COUT1
29DB F1              0028        POP   PSW
29DC C3 49 29        0029        JMP   2949H   ;JUMP TO COUT1
```

NO NUMBERS



normal typing practice. Several features of Northstar BASIC interfere with normal text entry. 1) BASIC will not allow a line which is entirely blank except for the line number: hence, any line which is meant to remain blank must be "protected" by typing in at least one space after the line number. 2) BASIC interprets any number typed immediately after a line number as part of that line number (resulting in an unintentionally large line number): hence it is most important that the user never begin a line of text with a number. Any other character, including a space, will "protect" the number. (If numbers occur frequently within a given text, it may be wise to type in a space at the beginning of every line of a BASIC text file.) 3) Northstar BASIC has the annoying (for us at least) habit of turning certain characters into other characters. The routine was devised to facilitate the translation of other BASIC programs into Northstar BASIC. In brief, ";" becomes "," ....":"becomes "(" and ")" respectively.

Either the characters on the left side of these equations must be avoided, or BASIC must be altered to protect the problem characters. The following FILL commands will protect the four characters and will allow them to be saved to the diskette. (The open parenthesis however will be changed to a square bracket if LOADed with an unaltered BASIC.)

```
FILL 12529,93
FILL 12536,59
FILL 16418,91
FILL 16420,28
```

NOTE: These FILL statements may be given as direct commands, written into a program, or RUN as they stand here. Text may be SAVEd in regular BASIC files, and

```
10 REM  NONUMBER, a line-number stripper for Northstar DOS 5.2
20 REM
30 REM   devised by Alan H. Nelson, BASIC program by Chris Nelson.
40 REM
50 REM  First RUN program, then type FILL commands to turn option on
60 REM  or off.  When the option is on, any file sent to a printer
70 REM  or to any other device on the second port will have all initial
80 REM  numbers suppressed.  NONUMBER will allow BASIC text files to be
90 REM  printed in a more legible and esthetically more pleasing form.
100 FOR I=10688 TO 10718
110 READ X
120 FILL I,X
130 NEXT
140 GOTO 180
150 DATA 245,120,217,203,66,40,13,254,48,56
160 DATA 7,254,64,48,3,217,241,201,203,130
170 DATA 254,10,32,2,203,194,217,241,195,73,41
180 !"NONUMBER option enabled.  In BASIC, affects -#1 commands only."
190 !
200 !"To turn line numbering off, type FILL 10551,192"
210 !"To turn line numbering on,  type FILL 10551,73"
220 !
```

SMART PRINTER DRIVER

By Joe Maguire

There I sat, fidgeting, while my printer alternately remained motionless and then sprang to life, only to print a single character. It reminded me of a myopic one-fingered typist who had misplaced his glasses. What it was actually doing was trying to print a large biorhythm plot from a Basic program. Each line of print was almost entirely filled with blanks. The printer was waiting for all those spaces to arrive from the serial port before moving the carriage to the print column. In addition, after printing a character in the far right column of the 132 column sheet, the carriage would return to the left margin only to dash over to the far right again to print the character on the next line! I decided that there had to be a better way.

What I needed was a bi-directional, logic seeking printer driver which could be used with my NEC Spinwriter or other printers, such as the Xerox Diablo series, which are capable of bi-directional printing but do not contain their own "smart" logic for doing so.

The accompanying assembly language program is the result of my effort. The listing as shown is tailored for the Spinwriter serial printer but by changing the TAB codes and the COUT routine it could be used with the Diablo or others.

The driver receives a character to be printed by intercepting it from the "device #1" routine of the North Star DOS. This is the second serial port on the Horizon computer but can be changed to anything desired. The character, once received, is stored in a buffer behind the driver until a carriage return character (ØD) is received. Then the buffer is "flushed" to the printer.

Before flushing, the driver strips all unnecessary blanks from the line. TAB codes are then substituted to position the print head properly. This significantly speeds up printing, particularly when using low baud rates for communication.

Each sucessive line is scanned for length and position on the page and the driver tabs the printhead so that printing is always accomplished in the most efficient manner. This is known as "logic seeking." The driver also detects when a series of linefeeds or carriage returns is received and performs only vertical paper movement.

After writing this program I tried it out on the biorhythm plot. Printer throughput increased by over 6Ø%.

CLIP TIPS

FREE CPM SOFTWARE

CP/NET operated by Kelly Smith, 3055 Waco St. Simi Valley, CA 93063 has 20 megabytes of CPM Users Group software on line for modem conection transfer at no charge but your phone time. (805) 527-9321. Typing DISKMENU.DOC will tell you whats available. Type HOWTO.USE for instructions. Kelly can be reached at 527-0518

SMART PRINTER DRIVER

A "smart" printer driver for use
with North Star software.

Featuring Bi-directional and Logic
Seeking printing. Fully compatable
with all North Star software
including CP/M.

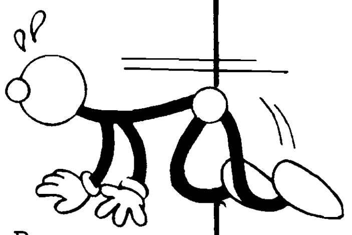
Written by: Joe Maguire June 1981
P.O. Box 3742 DT
Anchorage, AK 99510

This program requires about 512 bytes of memory including the code and the necessary buffer space. It has been tailored for the NEC Spinwriter serial printer and the Horizon computer's "device #1" output but by changing the command codes and the COUT routine it could be used with any system.

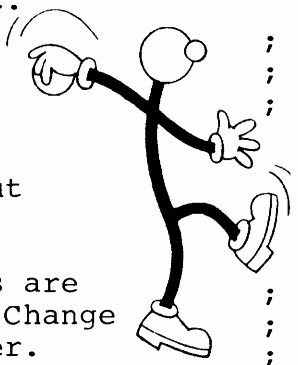
INSTALLATION:

Change the necessary ORGs and EQUates to agree with your DOS and I/O routine. Assemble this program and save the object code in a file on your N* disk. TYpe the file a type 1 with a GO addr of the program ORG below. To activate the driver, type GO <file>. The INSTAL routine below will modify your DOS I/O to jump to this driver when you select printer output. Sit back and enjoy. To use with CP/M, change the DEVSEL EQUate to point to the jump to the IST device in your CP/M BIOS.

```
        ORG     0F000H  ;A free memory
;                       ;area in my system.
;
;  DEVSEL is the location in the DOS I/O
;  block where the jump to this routine
;  must be placed.
;
DOS:    EQU     0100H   ;DOS 5.2 Origin
;
DEVSEL: EQU     DOS+978H  ;For Horizon I/O
;                       ;release 5.2
;
;DEVSEL: EQU    DOS+949H  ;For Horizon I/O
;                       ;release 5.1
;
;DEVSEL: EQU    DOS+99AH  ;For Horizon I/O
;                       ;given in Compass
;                       ;Vol. 1, #1.
;
INSTAL: MVI     A,0C3H  ;Store JMP code
        STA     DEVSEL  ;for device #1
        LXI     H,START ;Patch DOS to jump
        SHLD    DEVSEL+1  ;to this routine
        RET             ;Return to DOS
;
START:  MOV     A,B     ;Store B in A
        PUSH    PSW     ;Save everything
        PUSH    B
        PUSH    D
        PUSH    H
;
;  Check for special characters
;
        MOV     A,B     ;Char arrives in B
        CPI     0DH     ;A Carriage Return?
        JZ      PRINT   ;If so
        CPI     0AH     ;Linefeed?
        JZ      LINEF   ;Do the LF
        CPI     20H     ;Any CTRL chars?
        JC      STFWD   ;Set forward print
```

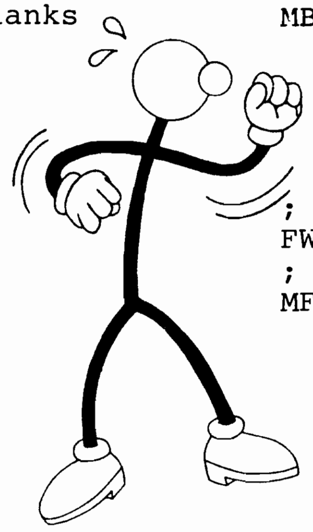


```
        PUSH    PSW     ;Save char
        LDA     EOLINE  ;Get End of Line
        INR     A       ;Bump it
        STA     EOLINE  ;Save it
        POP     PSW     ;Get char back
W1:     LHLD    TXTPT   ;Get text pointer
        MOV     M,A     ;Put char in buffer
        INX     H       ;Bump text pointer
        MVI     M,0     ;Text end mark
        SHLD    TXTPT   ;Save text pointer
;
EXIT:   POP     H       ;Time to go
        POP     D
        POP     B
        POP     PSW
;
;**************** NOTE! *******************
; If you are not using the Horizon I/O of
; DOS 5.2, delete the next two instructions.
;
        INX     SP      ;Get rid of RET
        INX     SP      ;in DOS 5.2.
;
        RET
;
;  Process the linefeed
;
LINEF:  CALL    COUT    ;Send it out
        JMP     EXIT
;
;  On the Spinwriter, command codes are
;  preceeded by an ESC char (1B).  Change
;  this as required for your printer.
;  Command codes must be picked out of the
;  buffer in the correct order. Therefore,
;  after sending a command, we must force
;  forward printing. (The buffer is read in
;  the same direction as printing occurs.)
;  NOTE: The driver sends TAB codes and
;  DIRECTION commands between reads of the
;  buffer so, setting forward print is not
;  required for them.
;
```



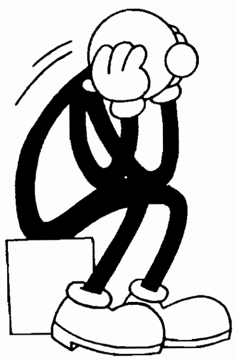
```
;
STFWD:  STA     DRCTN
        JMP     W1
;
;  This routine determines the direction,
;  the column and the mode in which the
;  print head will move.
;
COMND:  EQU     1BH     ;ESCape code
;
PRINT:  LDA     DRCTN   ;Get direction flag
        ORA     A       ;If not zero,
        JNZ     MFWRD   ;set FWD print
;
        LXI     H,BUFFER+1  ;Strip leading blanks
        MVI     D,0     ;# of blanks stripped
        MVI     A,' '   ;will go in D
P1:     CMP     M
        JNZ     P2
        INR     D
        INX     H
        JMP     P1
;
;  The following 3 lines of code check for no
;  characters in the buffer.  That means just
   a carriage return was received.  The C/R
   is ignored and only the linefeed is sent to
   the printer.  This keeps the print head from
   returning to the left margin with multiple
   line spacing.  If your software does not
;  send a C/R L/F sequence, this part of the
;  driver must be modified.  (N* uses CRLF)
;
P2:     MOV     A,M     ;Check 1st buff char
        ORA     A       ;If zero,
        JZ      CLRBUF  ;then exit.
;
        DCX     H       ;Move ptr to buff start
        MVI     M,0     ;Store marker
        MOV     A,D     ;Store beginning
        STA     BOLINE  ;of line count
```

```
'
        LHLD    TXTPT   ;Strip lagging blanks
        LDA     EOLINE  ;Get EOL
        MOV     D,A     ;Move count to D
        MVI     A,' '
P3:     DCX     H
        DCR     D       ;Subtract 1 from
        CMP     M       ;for each blank
        JNZ     P4
        JMP     P3
P4:     INX     H       ;We found EOL
        MVI     M,0     ;Store end mark
        MOV     A,D     ;Store new
        STA     EOLINE  ;end of line
        SHLD    TXTPT   ;Store text pntr
;
;  Do we print forward or backward?
;
        LDA     HDPOS   ;Get print head position
        PUSH    PSW     ;Save it
        LXI     H,BOLINE  ;Get BOL
        SUB     M       ;Subtract HDPOS
        CM      ABSVAL  ;Remove sign
        MOV     B,A     ;Save it
        INX     H       ;Point to EOL
        POP     PSW     ;Get HDPOS back
        SUB     M       ;Subtract it
        CM      ABSVAL  ;Remove sign
        SUB     B       ;Compare BOL to EOL
        JP      MFWRD   ;If +, FWD is faster
;
;  Print right to left
;
BAKCHR: EQU     '<'     ;Print BKWD code
;
MBWRD:  LDA     EOLINE  ;Get EOL
        CALL    TAB     ;Tab head to it
        MVI     A,COMND ;Send command
        CALL    COUT
        MVI     A,BAKCHR  ;Set BKWD print
        CALL    COUT
        LDA     BOLINE  ;Get BOL
        ORA     A       ;See if left margin
        JZ      MB1     ;Can't print left
        DCR     A       ;of margin.
```



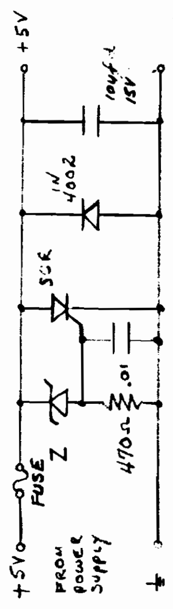
```
MB1:    STA     HDPOS   ;Store new HDPOS
        LXI     D,-1    ;Set up indexer
        LHLD    TXTPT   ;Get text pntr
        JMP     PRLINE  ;Send out line

  Print left to right
;
FWDCHR: EQU     '>'     ;Print FWRD
;
MFWRD:  LDA     BOLINE  ;Get BOL
        MVI     D,0     ;Save in DE
        MOV     E,A
        CALL    TAB     ;Tab head to it
        MVI     A,COMND ;Send command
        CALL    COUT
        MVI     A,FWDCHR  ;Set FWRD print
        CALL    COUT
        LDA     EOLINE  ;Calculate and
        INR     A       ;store new
        STA     HDPOS   ;head position
        LXI     H,BUFFER  ;Point to 1st
        DAD     D       ;character
        LXI     D,1     ;Set up indexer
;
;  Now print the buffer
;
PRLINE: DAD     D       ;Bump text pointer
        MOV     A,M     ;Get character
        ORA     A       ;End of buffer?
        JZ      CLRBUF  ;Exit if so
        CALL    COUT    ;Send out char
        JMP     PRLINE  ;Over and over
;
;
;  Clear the buffer and reset pointers
;
CLRBUF: LXI     H,BUFFER+1
        SHLD    TXTPT
        XRA     A
        STA     EOLINE
        STA     BOLINE
        STA     DRCTN
        STA     BUFFER+1
        JMP     EXIT
```

```
;
;  Put the head at proper column position.
;
TAB:    PUSH    PSW     ;Save column count
        MVI     A,COMND ;Send command
        CALL    COUT
        POP     PSW     ;Get back count
        PUSH    PSW     ;Save it again
;
;  The following is the column addressing
;  calculation for the Spinwriter.  Change
;  this if you are using another printer.
;
        ANI     0E0H    ;Convert to
        RLC             ;Spinwriter code
        RLC
        RLC
        ADI     'P'
        CALL    COUT
        POP     PSW
        ANI     1FH
        ADI     '@'
        CALL    COUT
        RET
;
ABSVAL: CMA             ;Absolute value
        INR     A
        RET
;
;  This is the character output routine.
;  This version is for a Horizon computer.
;
COUT:   PUSH    PSW
        IN      5
        ANI     1
        JZ      COUT+1
        POP     PSW
        OUT     4
        RET
```



```
;
;  The following must reside in RAM.
;
DRCTN:  DB      0       ;Direction flag
HDPOS:  DB      0       ;Print head position
BOLINE: DB      0       ;Beginning of line
EOLINE: DB      0       ;End of line
TXTPT:  DW      BUFFER+1  ;Text pointer
BUFFER: DW      0       ;Buffer starts here
;
        END
```

CLIP TIPS



(Reprinted from the CSRA Computer Club's Newsletter, July 1979)

Q. Is there any easy (cheap) way to add overvoltage protection to any power supply?

A. Yes! And you are wise to wonder about overvoltage protection. Many hobbyist home-brew power supplies are built using three-terminal voltage regulators like the LM340K-5 or the 78H12. In fact, these IC's make excellent regulators until one goes bad. Then you find that, like all semi-conductors, a voltage regulator IC is as likely to short as it is to open. And, if it does short, it will supply maximum power supply voltage to whatever it was connected to. When this happens you stand a good chance of burning out many other IC's in your circuit -- they simply can't stand doubled power supply voltage. If you are building a power supply, or if your present equipment does not have overvoltage protection, try adding the following circuit to the output of your supply.

A couple of the parts values must be changed according to the power supply voltage you are using. The zener diode (Z) should be rated about 0.6 volts greater than the normal output voltage of the power supply -- 5.6 volts for a 5 volt supply, and 12.6 volts for a 12 volt supply, etc. The silicon controlled rectifier should be able to handle several times the maximum power supply current. Additionally, the SCR's gate turn-on current must be low, 50 milliamperes or less. Finally use a fuse rated at to 10% above the maximum power supply current. This circuit works very simply. If your 5 volt regulator melts down and tries to apply 9 or 10 volts to your circuit, the zener diode applies the overvoltage to the SCR gate. Next the SCR turns on and becomes a dead short across the power supply which, of course, blows the fuse. In other words, if the 5 volt power supply tries to go above 5.6 volts, the fuse blows and all power is shut off.

NS CP/M

TO CP/M AND BACK AGAIN

Ann Hernández
3111 Erla Way
Richmond, CA 94806

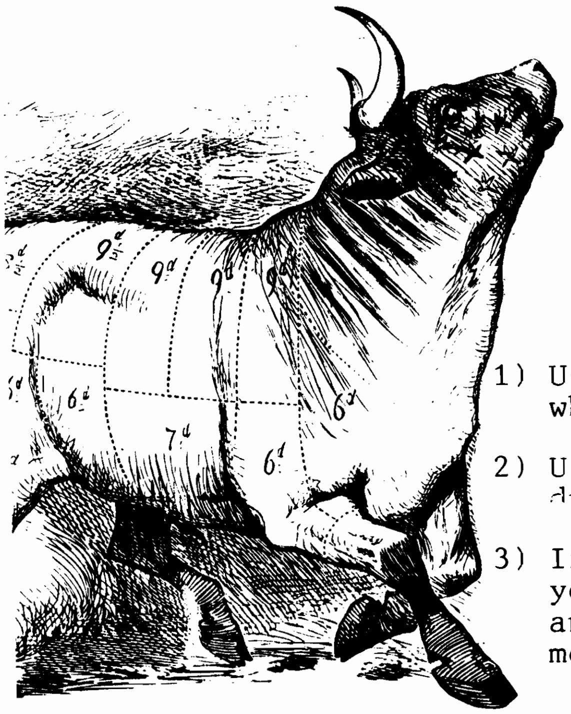
When I wanted to find out how to run existing N*BASIC programs under MicroMike's baZic for CP/M, I turned to Bob Strek's article on page 6 of the first Compass (Vol. I, no. 1) and found that files fewer than 31 blocks long could be loaded as follows:

```
Boot up in N*DOS
+LF OLDNAME 0100
Boot up in CP/M
A) SAVE NEWNAME.002 30
READY
LOAD NEWNAME
RUN
```

NOTE: For files shorter than 30 blocks, substitute the correct size in blocks

Not hard at all, thanks to Mr. Strek, but what about loading in the other direction, from CP/M baZic back to the N* format?

This is a fairly easy way to send files back to N*BASIC, where they can be massaged with N*BUS and/or other NorthStar utilities we have learned to love.



```
Boot up in CP/M
A) DDT OLDNAME.002
Boot up in N*DOS
+CR NEWNAME 4 120
+WR 120 0100 4
+TY NEWNAME 2
+GO BASIC
READY
LOAD NEWNAME
RUN
```

(Where 120 is the disk address; 4 is the size in blocks; 0100 is where CP/M files are loaded in; 2 is the filetype)

1) Use the direct command PSIZE in BASIC to tell what size file to reserve.

2) Use B:DDT if your DDT.COM file is on a second drive.

3) If you forgot to get the PSIZE of your program, you can guess-timate from DDT using the -d option and then moving the program up to a safer place in memory (eg. -m0100,0150,6000) before rebooting.

SWAP I/Os

Leonard Morgenstern

As everyone knows, the NorthStar DOS has a reserved segment of 256 bytes for I/O routines, and a jump table for accessing it, more than enough to support a keyboard, a screen, and a printer. However, there is a problem if you need casette, serial, etc. etc. And why not have an I/O routine do simple formatting instead of running the text through an editor?

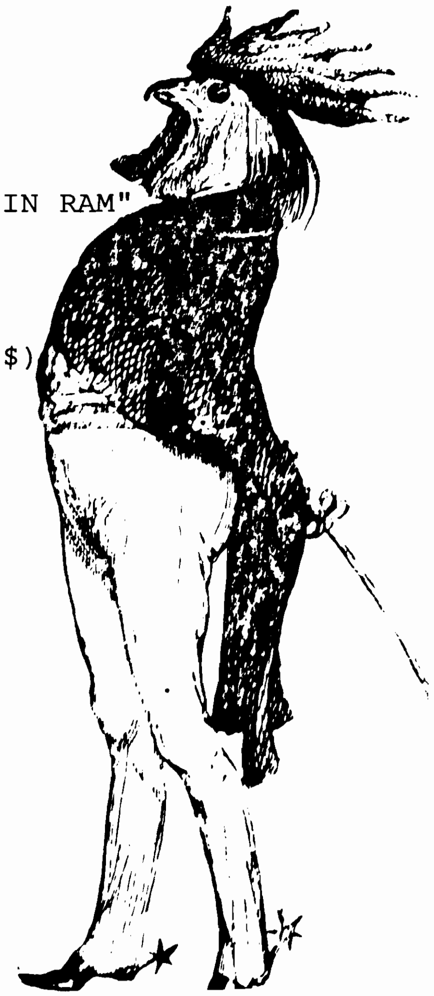
A solution is to have a quick way of changing the I/O segment.

The program CRDOSIO, written in North Star Basic, will create a type 1 ('GO') file that will load the I/O segment of the DOS and set the pointers in the DOS jump table.

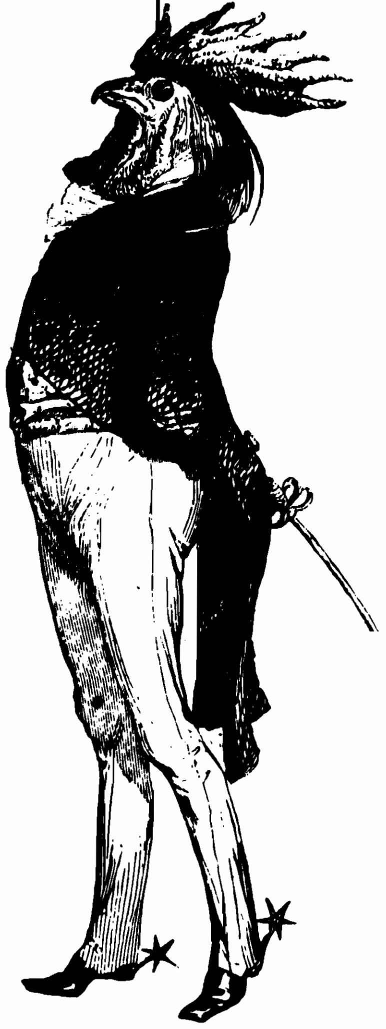
To make such a file, load DOS, BASIC, and CRDOSIO, and run. The program is essentially self-documenting. For double density DOS, the object program will have its origin at DOS+700H, which is the DOS input buffer; hence no special area needs to be allocated. For single density systems you will need to select a 512-byte block of memory in which the program can run.

A warning. CRDOSIO is not a relocator. It extracts data from the DOS that is currently in RAM. The object program must be used on a DOS at the same location and density as the program from which it was created.

```
                       LDOSIO"
         QUICKCHANGE ARTISTRY FOR THE DOS"

 10 REM CRDOSIO
 20 REM
 30 !"CREATE  LOADER FOR I/O PORTION OF DOS NOW RESIDENT IN RAM"
 40 !
 50 !"DO YOU NEED INSTRUCTIONS ",
 60 ON FNY(0) GOTO 70,80
 70 GOSUB 820
 80 INPUT "ADDRESS OF DOS (standard=2000) ",D$ : D=FNH1(D$)
 90 INPUT "DENSITY (D=double, S=single) ",D1$
 100 IF D1$<>"S" THEN 140
 110 O=8*256 : REM OFFSET OF I/O FOR SINGLE DENSITY
 120 INPUT "RUN ADDRESS ",R$
 130 R=FNH1(R$) : GOTO 150
 140 R=D+1792 : O=9*256 : R$=FNH$(R)
 150 !:INPUT "NAME OF 'GO' FILE TO BE CREATED ",F$
 160 IF FILE(F$)<0 THEN 190
 170 INPUT "File already exists. Is that OK? (Y/N)?",I1$
 180 IF I1$(1,1)<>"Y" THEN 150 ELSE 200
 190 IF FILE(F$)<0 THEN CREATE F$,2
 200 OPEN#0,F$
 210 WRITE #0,&33:X=D+13:GOSUB600
 220 WRITE#0,&17:X=R+32:GOSUB600:WRITE#0,&6,&12,&205
 230 X=R+23:GOSUB600
 240 WRITE #0,&33:X=D+O:GOSUB600
 250 WRITE#0,&17:X=R+44:GOSUB600:WRITE#0,&6,&0,&205
 260 X=R+23:GOSUB600
```



```
270 WRITE#0,&201
280 WRITE#0,&26,&119,&19,&35,&5,&194
290 X=R+23: GOSUB600
300 WRITE#0,&201
310 REM WRITE DOS JUMP TABLE
320 FOR I=D+13 TO D+24 : WRITE#0,&EXAM(I) : NEXT
330 REM WRITE I/O SEGMENT
340 FOR I=D+O TO D+O+255 : WRITE#0,&EXAM(I) : NEXT
350 FOR I=1 TO 63 : !"*", : NEXT : !
360 !:!"FILE ",F$," HAS BEEN CREATED"
370 !"IT SHOULD BE SET TO TYPE 1, AS FOLLOWS "
380 !TAB(10),"BYE"
390 !TAB(10),"+TY ",F$," 1 ",FNH$(R)
400 FOR I=1 TO 63 : !"*", : NEXT : !
410 END
420 REM
430 REM
440 REM convert hex to decimal, return -1 on error
450 DEFFNH1(I1$)
460 I1=LEN(I1$)
470 IF I1=0 THEN RETURN -1
480 I0=0
490 FOR I2=1 TO I1
500 I3=ASC(I1$(I2,I2))
510 IF I3<48 OR I3>70 THEN EXIT 560
520 IF I3>57 AND I3<65 THEN EXIT 560
530 I0=I0*16+I3-48-(I3>=65)*7
540 NEXT
550 RETURN I0
560 RETURN -1
570 FNEND
580 REM
590 REM WRITE TWO BYTES TO FILE#0 FOR A RAM ADDRESS
600 I1=INT(X/256)
610 WRITE#0,&(X-I1*256),&I1 : RETURN
620 REM FUNCTION TO INPUT Y/N . RETURN 1 IF Y, 2 IF NOT Y
630 DEFFNY(I)
640 INPUT "(Y/N)? ",I1$
650 IF I1$="" THEN RETURN 2
660 I1$=I1$(1,1) : IF I1$="Y" THEN RETURN 1 ELSE RETURN 2
670 FNEND
680 REM
690 REM
700 DECIMAL TO HEX CONVERSION
710 DEFFNH$(X)
720 I1$="" : I3=X
730 FOR I1=1TO 4
740 I2=INT(I3/16) : I1$=FNH1$(I3-I2*16)+I1$
750 I3=I2
760 NEXT
770 RETURN I1$
780 FNEND
790 REM
800 REM CONVERT VALUE TO SINGLE HEX CHAR
810 DEFFNH1$(Y)=CHR$(48+Y+(Y>9)*7)
```



EASY SWITCH

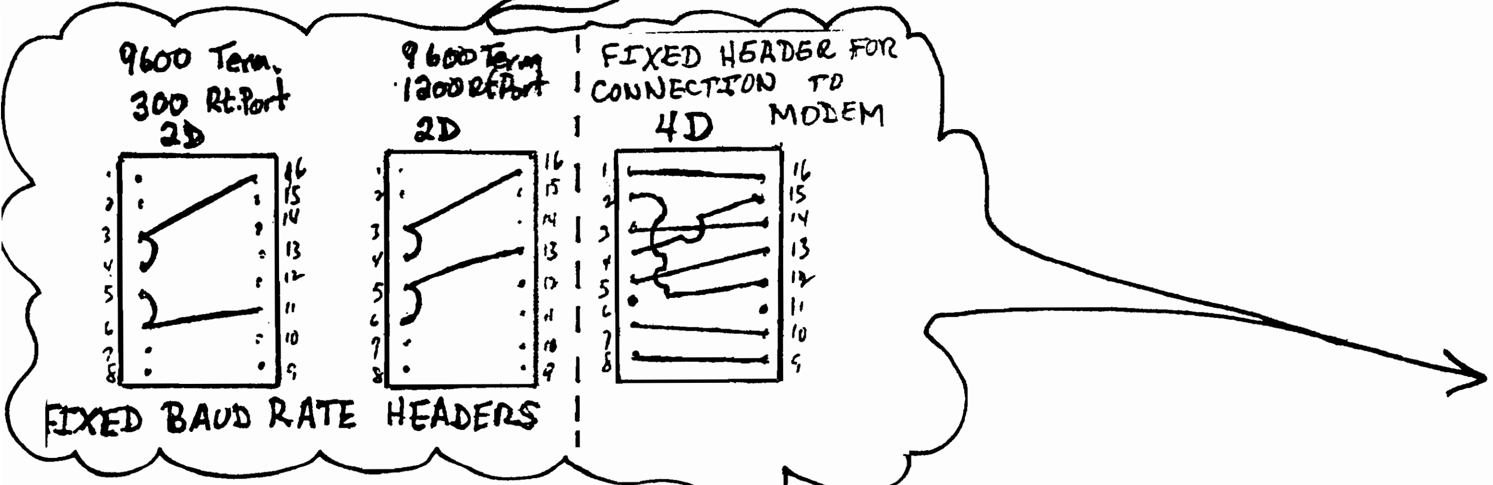
by Alan H. Nelson

Header-switches for the Horizon

Northstar Horizon presents a somewhat deceptive face to the world. Behind that smooth, clean facade lives a computer of astonishing power and versatility. The clean facade also represents a slightly misleading conceptual idea of the internal workings of the machine. Because the front panel is bare, one is led to believe that the Horizon needs no manual switches. But two switches, on and reset, are placed on the back panel, a situation convenient enough for orangutangs, but somewhat awkward for humans. Additional switches are actually placed inside the machine, where they can be got at only by lifting off the entire cover. I refer in fact not to literal switches, but to the headers, which are more versatile than switches, but are much more awkward to manipulate.

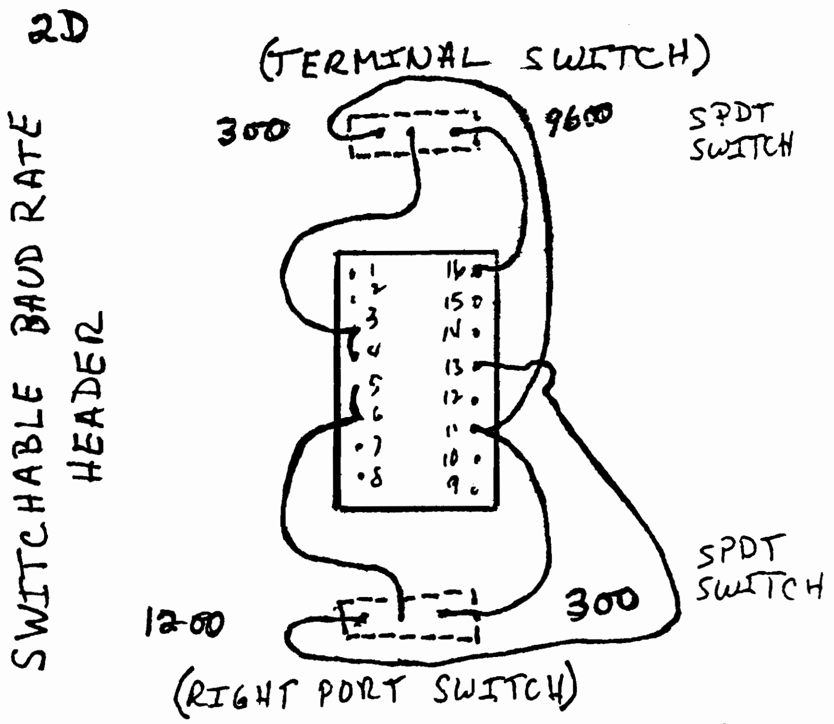
Many users will be able to install permanently configured headers and will never have to touch them again. Others, like myself, use the Horizon in several different configurations, and must manipulate headers with some frequency. A good example is offered by the TELESTAR program. One header is needed when the right serial port is connected to a modem for communicating with a time-sharing system, while another is needed for output to a printer. Similarly, the user who has a 1200 baud printer but a 300 baud modem must be able to change the baud rate header.

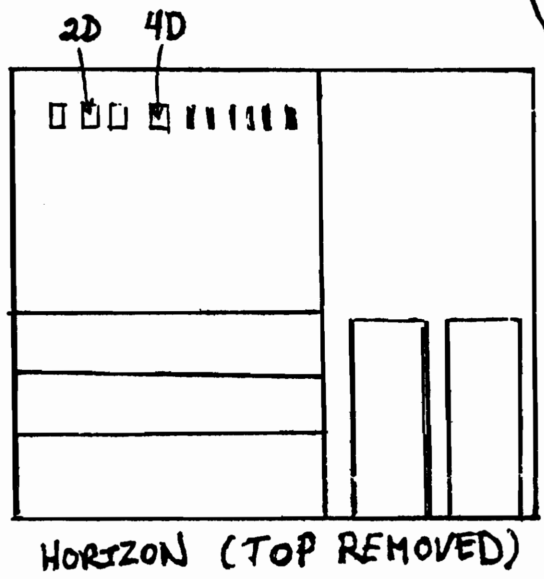
One solution to the problem of manipulating headers is to install switches wired to the headers. The accompanying diagrams should furnish most of the information needed to accomplish this task, but a few additional words may still be in order.


9600 Term. 300 Rt.Port 2D
9600 Term 1200 Rt Port 2D
FIXED HEADER FOR CONNECTION TO MODEM 4D
FIXED BAUD RATE HEADERS

EASY SWITCH

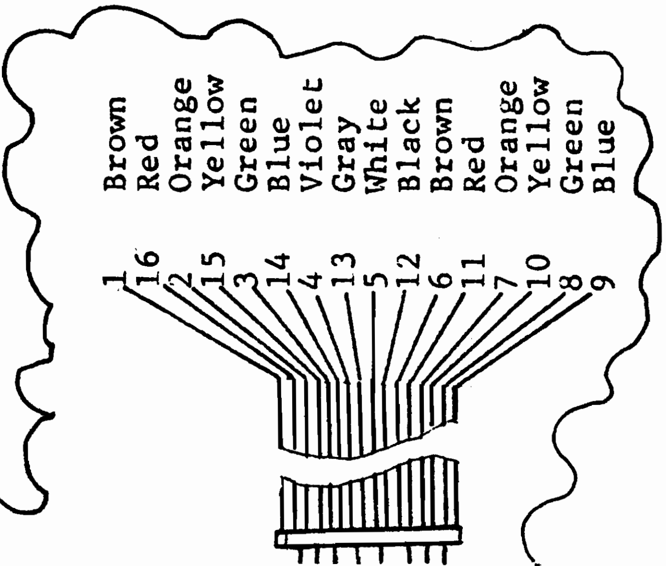
First, concerning the baud-rate header. The Northstar Manual explains which pins should be wired to which, but a diagram makes the matter much simpler to understand. Pins three and four, wired together, control the baud rate of the terminal. For most applications the terminal should run at 9600 baud, and thus these pins should be wired to pin 16. Pins five and six, wired together, control the right serial port. Many peripherals, including modems and printers, work satisfactorily at 300 baud, so these pins may be wired to pin 11. Thus a fixed header may suffice. Other applications, however, may require a changeable baud rate. A simple SPDT switch will allow the terminal to be switched between (say) 9600 and 300 baud; another SPDT switch will allow the right port to be switched between (say) 300 and 1200 baud. The holes already drilled in the Horizon back panel accommodate switches with 3/8" shanks. One or two switches as needed may be wired to standard DIP plug .



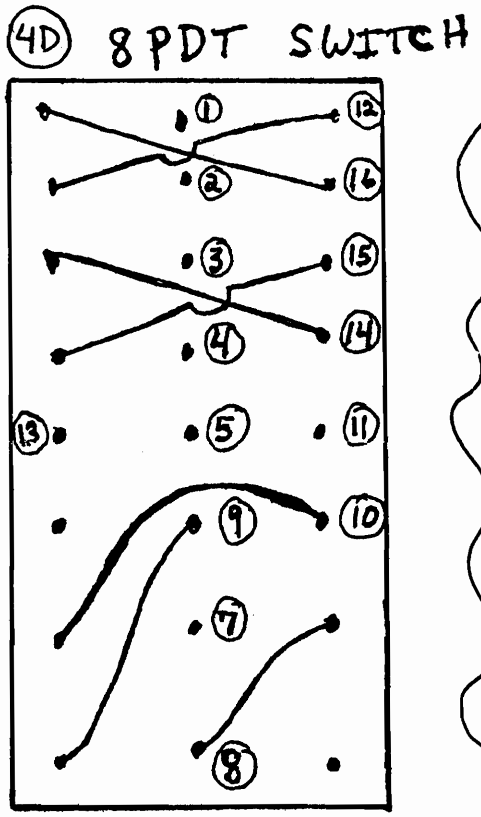


A more complicated device is required to switch from printer to modem configuration. Since fifteen of the sixteen pins

EASY SWITCH



16-wire ribbon with DIP plug



(Connect switch lug to indicated pin on header.)

SWITCH FOR RIGHT PORT CONNECTION TO MODEM or PRINTER

are involved, a 16-wire ribbon cable with a DIP plug already attached to one end is a virtual necessity. The other end of the wires should be attached to the lugs of an 8PDT switch as shown. The easiest approach is to attach the shunt wires to the switch first, then solder on the wires from the cable. The greatest problems in making this improvement to Horizon will probably be finding an 8PDT switch, figuring out the gang arrangement, and mounting the switch to the back of the Horizon. A good electronics supply house and some ingenuity should suffice.

My own Horizon, which has a fancy back-panel, also has a switch leading from the 32K ram board so that I can change dip-switch settings from the outside. The ultimate refinement and convenience would be to mount all the switches on the front panel of the Horizon.

dBASE™ II

BY BOB STEK

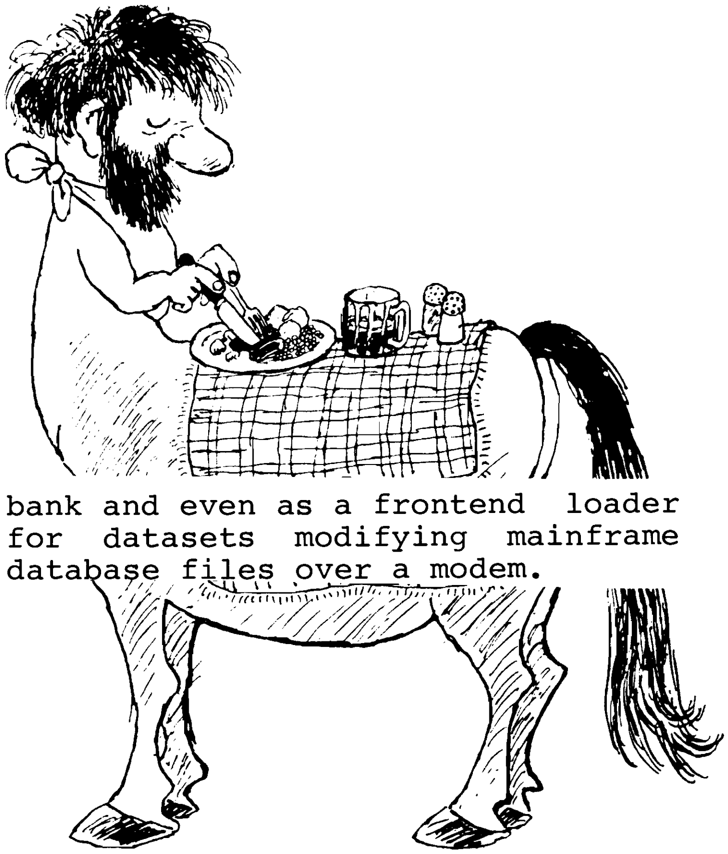
Good, user friendly database managers for microcomputers are not the most common commodity to find, so it was a pleasant surprise to receive a superior piece of software from Ashton-Tate, dBASE.

dBASE is described as a relational database manager, that is, the user defines his information in sets of tables (or files) which are linked together by relationships. This is not a unique aspect of file management in that programmers often develop systems which use multiple files which necessarily contain keys for linking purposes. For example, a personal library inventory would only want to carry a publisher code with each book entry, and have a separate table or file for identifying the demographic information relating to the publishers. These two tables or files would be linked by some logical or relational connector, such as a publisher code. The feature which makes dBASE truely unique is that the management of the indicies is performed for the user by the system. System index files are created easily and maintained automatically as the files are altered.

This is different, and simpler than using other database approaches which maintain hierarchical structures linking the various items of information.

USEFULNESS

The manual suggests several applications for which dBASE would be an appropriate tool, however I found as I used it that the potential applications became extremely broad. I have used it for system management, i.e., keeping track of the flow of systems used and datasets created, personal library records maintenance, education test item bank and even as a frontend loader for datasets modifying mainframe database files over a modem.



It appears to have sufficient power and speed to be an excellent general purpose processor for any microcomputer system. I have found it able to interface easily to other applications systems such as word processors and spelling programs.

dBASE behaves somewhat like an interpreter, and indeed can be visualized as being like a BASIC interpreter with additional built-in file handling functions for appending, editing and indexing. Never have I encountered a micro-computer system which matches dBASE's capabilities and ease of data entry and editing. Once a file is defined, the user may begin entry of data, record by record. While a record is available for data entry, full cursor positioning is available. The record is displayed vertically on the screen with each field in the record identified by name, and the field length indicated by matched colons. Data editing is performed by simply typing EDIT <record number>, and the entire record is presented, again utilizing full cursor control.

dBASE™ II

The system allows the user to work interactively with the files in use, or alternatively to process files through the creation and execution of command files. A command file is simply a list of dBASE instructions which can be executed at the users request. This dual capability extends the power of the processor and assists in the development of applications software.

A major deficiency in the current version is the lack of a resident editor for the creation of command files. I currently use my word processor for this task, and have found that the capability of dBASE to leave itself, execute another program, and return to dBASE when done does speed up the process somewhat. A further improvement which would assist a systems developer and data entry operator would be the inclusion of a range-checking feature for data entry routines. Currently, the processor checks data types, however range checking procedures must be developed by the analyst.

dBASE provides the user with the powerful extension of macro applications. This means that a variable can be defined which in fact may contain an executable statement. The macro can be developed while the program is executing and can cause the program to alter its functions in a user defined direction.

DOCUMENTATION

The documentation comes in two manuals, one which is a typical systems manual in that it attempts to describe the syntax of the language and indicate the various options available to the user. The second manual was written by a user, who takes you through the various commands providing examples and hints as you progress.

The unfortunate aspect to the second manual is that it appears to be geared to the application which is given at the end, and doesn't answer all of the questions which users will have, particularly as new applications are attempted. I felt that it would have been better to present it as a tutorial which would lead the new user step by step through various applications aspects. The application listed at the end of the manual would have been suitable for this purpose, and would probably facilitate new users becoming familiar with the system quickly.

The other manual consists of statements and examples of syntax, and is a handy ready reference when needed. Syntax is easily found and the examples given usually provide sufficient information for the user to procede.

The greatest weakness I found was in what was not said. For example, there is a system function for substring manipulation. This function is passed two parameters, the first character, and the length of the substring. Nowhere did I find a note indicating that only numeric values or macros could be used as parameters to be passed, simple variables could not. Aside from making the processor somewhat more difficult to use, it also added excitement, and frustration to the learning process.

EASE OF USE

In general, once a user has mastered a few of the basic statements, the processor becomes an easy tool for system development. Its greatest asset is the built-in file handling functions. You define the structure of your file or files in an easy and straightforward fashion, and you no longer need to be concerned with the characteristics of the file. If the structure of the file needs to be modified, a built-in function allows for easy modification, i.e., addition, deletion or editing of file fields. Of equal importance is the fact that command files which do not use the new fields need not be altered. The only possible disadvantage I could identify is that the datafile is lost and must be copied to a different name before any modifications are made. The system warns the user of this matter, and requests direction before proceding, so if you lose your file, you have only yourself to blame.

ERROR HANDLING

Error handling can only be described as superior. The messages provided to the user when runtime errors are encountered allow the user to make corrections and continue from where the interruption occurred. The user must know the syntax well enough to be able to identify and correct the problem.

SUMMARY

I am pleased to have a processor which is easy to use for applications purposes, allows the user to work interactively or in a batch submit environment, encourages structure in system development and removes the laborious tasks associated with system development and maintenance of files. Overall I feel it is an excellent piece of software, well worth the price charged.

Available from: Ashton-Tate
Suite 1500
3600 Wilshire Blvd.
Los Angeles, California 90010
Price: $700

CLIP TIPS

The HORION hardware manual HR-D-DOC contains a discussion called 'USING THE PARALLEL I/O INTERFACE' which gives elaborate details on how to implement both the parallel input and the parallel output ports. What is NOT clear is that the main thrust of the discussion is aimed at using the input port for keyboard input and the output port as a printer port bearing the N* designation of device 2. The latter would be used in BASIC statements such as 100 PRINT 2, 'THIS COMES FROM THE PARALLEL OUTPUT PORT.'

In my setup, I am using the two SERIAL ports for keyboard and printer I/O, and simply want to use the PARALLEL ports to control external devices and monitor them. The implementation of such a setup is extremely simple.

1. Forget about those machine language subroutines --- they are unnecessary for this type of application.
2. Connect pin 1 to pin 14 on the DIP header at location 9C. This enables the data output latch. The data input latch is already enabled. NO OTHER JUMPERS ARE NECESSARY ON THE DIP HEADER.
3. Forget about the N* designation as device 2, as it does not apply to this application.
4. Instead, become acquainted with the BASIC statements INP and OUT which require a port number in standard 80 notation. Both ports in question have been hard-wired as 80 port 0; see the discussion 'MOTHERBOARD PORT I/O ADDRESSING' in HR-D-DOC for details.
5. To monitor external data using the parallel input port, use a BASIC statement such as 100 = INP(0)
6. To control external devices via the parallel output port, use a BASIC statement such as 200 OUT 0,A where A is an integer from 0 to 255.

PWJonas

WordStar

by Steve Leibson
4040 Greenbriar Bl
Boulder, CO 80303

I can't type. This may seem a strange way to start a review of a word processing software product but it really isn't. Those of us who never learned to type need a word processor even more than those who do.

Typists do not make nearly as many mistakes as non-typists. It's the mistakes that take all the time. When I made an error on the typewriter, I had to roll the page up, paint on an opaque white solution, wait for it to dry and finally roll the page back down to where I stopped.

Many times, I wouldn't let the paint dry or would miss the place where I had stopped typing. Thus one mistake could easily be compounded by staggered lines or a large white splotch in the middle of my text.

Revolution

The WordStar word processing package has revolutionized my communication skills. Errors are simply eradicated with the press of a delete key. This is a minute fraction of the power placed at my disposal however.

WordStar is a screen editor. The computer screen replaces the typed page for composing. WordStar shows you exactly what will be printed when you finish. This includes showing you where the page will end and how the characters will fall on the line. Many word processors do not show page breaks and final copies can end up with dangling sentences at the end of a page or tables split in half by a page break.

Automatic word wrap is another feature that aids the typist and non-typist alike. You need not be concerned when nearing the end of a line. WordStar will split the sentence between words and carry the next word to the start of the next line. No carriage returns are needed except to end paragraphs.

WordStar will left and right justify text. Normally, I use both left and right justification for articles but turn off right justification for personal letters. It makes the letter look more "hand done".

These are just a few of the features that make WordStar easier for me to use than a typewriter. I have been able to improve both the quantity and quality of my reports and articles because of this amazing piece of software.

Choosing a Product

I had a NorthStar Horizon computer for seven months before purchasing WordStar. When I started my publishing house Data Press, I wrote my first book using a text editor that was part of an assembly language development system. Text editors, as opposed to word processors, are character oriented instead of word oriented. They don't know about words or sentences or paragraphs. They count characters or lines. A line is a string of characters that ends with a carriage return.

Though I managed to type my book into the computer using the text editor, I vowed to never do it again. This started my search for a good word processor and printer.

There were several requirements that had to be met in choosing a word processor. First, it had to be compatible with my computer, the North Star. This didn't eliminate many software packages because the North Star can run several operating systems, notably the NorthStar Disk Operating System (DOS) and CP/M.

Second, the software should be able to take advantage of the rest of my hardware. This includes a memory-mapped video display and a keyboard with extra, user-definable function keys. Also, I wanted a printer that would produce high quality text that could be used as camera-ready copy. Since my system did not yet have a printer, the word processor would have to handle as many different types of printers as possible.

WordStar fills those requirements very well. As it turned out, I bought both the word processor and the printer at the same time. These products were purchased at the Computerland of Denver. The only way I was able to decide on WordStar was to take the operating manual home for an evening. A thorough reading proved to me that WordStar would do what I wanted.

Bringing the System Up

When WordStar arrived from MicroPro, the manufacturer, there was an immediate problem. I had based my decision on version 1.8 but ordered version 2.0 which was brand new and had extra features. One extra feature added was a copying lock. A special program on the disk had to be run to configure WordStar to my system. Once configured, the configuration program destroyed itself. If my system ever changed in the future, I would have to send my disk back to MicroPro to have the installation program replaced.

This was simply not acceptable. My system is constantly changing as I improve my video and keyboard software. This sort of anti-pirating protection might be marginally acceptable for a business that never planned to upgrade its system but even there, I have my doubts.

Fortunately, MicroPro agreed that this was a problem and replaced WordStar 2.0 with version 2.1. I have been extremely happy with it ever since. Incidentally, many writers have written that version 2.0 was not copyable after being configured. This was never true. Backup copies were possible using the same configuration, after the installation program had been run.

Installation of WordStar is a breeze. You must have CP/M running reliably on your computer before trying to start up WordStar. A balky operating system will destroy text files and possibly ruin your program.

All parameters that WordStar requires for operation are entered once using the INSTALL program supplied. This program steps you through a series of menus. A selection from each menu is made and when INSTALL has finished, WordStar has a complete description of your system.

The first menu is the Terminal Menu. The menu is displayed as:

AMAZING! WordStar

```
TERMINAL MENU #1  *****

    A  Lear-Siegler      ADM-3A
C Lear-Siegler ADM-31
    D    Hazeltine         1500
E Microterm ACT-IV
    F  Beehive 150/Cromemco 3100
G IMSAI VIO
    H  Hewlett-Packard 2621 A/P
I Infoton I-100
    J  Processor  Tech SOL / VDM
K Soroc IQ-120
    L  Perkin-Elmer 550 (Bantam)
Z None of the above
    2 Terminal Menu #2
```

Terminal Menu number 2 looks like this:

```
           *****        WordStar
TERMINAL MENU #2 *****

    M      Microterm       ACT-V
N Televideo 912
    O         Visual         200
P Flashwriter I
    Q      Flashwriter        II
R SWTPC CT-82
    S       Compucolor    8001G
V TEC Model 571
    1        original       menu
Z none of the above
```

By entering the letter corresponding to your terminal, INSTALL places the necessary software routines in your version of WordStar to make that terminal work. Cursor addressing is used extensively in WordStar to update the text display and menus. That is why the program has to know what type of terminal you have. Each terminal requires a different sequence of characters, called an Escape sequence, to position the cursor on the terminal screen and to turn on inverse video for highlighting.

I have added some software to my memory-mapped video to make it appear as a Soroc IQ-120. Though WordStar worked with this setup, I found a much better configuration. WordStar knows about memory-mapped video displays.

My video board, a Vector Graphics Flashwriter II is on Terminal Menu number 2. Unfortunately, the standard Vector Graphic's configuration is for the Flashwriter to be addressed at D000 hex in the computer memory. My Flashwriter is at E000 hex so I could not use the standard INSTALL selection.

MicroPro has done an excellent job of documenting the terminal and printer drivers so I had no trouble patching in my board. That is because I can program in assembly language and understand much of the inner workings of my computer. It is not a job for the novice.

The computer store you buy your software from should be able to help you if you have problems. It might be best to assure yourself that they have the expertise and willingness to help before you buy from them. If you buy your software by mail order to get a better price, be sure you either buy the software configured for your system (with a guarantee of operation or money back), have someone standing by who can help you or be ready for an ulcer.

MicroPro has made the configuration as painless as possible but there is no substitute for an experienced computer programmer who can find the way around inside the guts of your computer, just in case.

The next menu presented is the printer menu. Printers supported are: generic "Teletype-like" printer, printer with a backspace capability, Diablo 1610/1620, Diablo 1640/1650, Qume Sprint 5, NEC Spinwriter 5510/5520, printer with "half-line-feed" capability or a printer driven by the MicroPro I/O Master interface board. Just as with terminals, MicroPro has provided you with a large group from which to choose. I can't think of any printer that isn't included in this list.

Except mine. I bought a NEC Spinwriter 5525. This printer appears to the computer as a Diablo 1610/1620 but it has some extra features. After carefully reading the printer documentation, I decided to configure WordStar for the Diablo 1610/1620. As mentioned for terminals, if you have a printer listed in the menu you will probably have no problem with INSTALL. Otherwise, have someone knowledgeable standing by.

The next menu is the "Communications Protocol Menu". Printers that connect to computers over a serial link commonly encounter a problem. Though characters can be transferred to these printers at up to 960 characters per second, most printers print the characters at a slower rate. Somehow, the computer has to be told to stop sending for a while, until the printer can catch up.

A mechanism called handshaking is usually used to do this signaling but the "standard RS-232C" serial interface doesn't really have handshaking. Many printer manufacturers violate the standard by using some of the RS-232 signal lines for a handshake but use in the printer doesn't guarantee that the computer will also violate the RS-232 standard in the same manner. The violation is not standardized!

A more complex handshake is performed when the printer sends a character to the computer requesting a pause in the transmission of characters. Later, when the printer catches up, it sends another character to the computer to start transmission again. Two sets of characters are commonly used for this purpose. They are ETX/ACK and XON/XOFF. Neither of these handshakes is standard.

WordStar can understand either of these character handshakes. You can also tell INSTALL that the handshake is to be done by other software or hardware and to not be concerned with handshaking. I had already implemented a hardware handshake in my computer so I told INSTALL to forget about handshaking.

Good Impressions

This brings us to the last INSTALL menu, the Driver Menu. This lets WordStar know how to send characters to your printer. Options are: as the standard CP/M "List" device, to a parallel output port somewhere in the computer or using user-installed subroutines.

The easy way out is to use the CP/M List device if your printer is already connected and working with your computer. Otherwise, you'll need the aid of a software specialist again.

Finally, INSTALL provides a "patcher" that allows you to directly access the code byte-by-byte. Thus you can put in your own machine language routines or set up other options that are explained in the back of the manual. This procedure definitely requires the services of a programmer if you need special options set up.

As you can see, there are quite a few things to tell WordStar about your computer. I think MicroPro has done an excellent job creating a way for one piece of software to be used on virtually any system.

I think you can see why I did not feel comfortable with version 2.0. I have now INSTALLed WordStar several times and each time I get a slightly better, easier-to-use system.

Working with WordStar

Since reading the WordStar manual before purchasing the software and installing it, I have not read it again. The only reason I can do this is because of the extensive use of menus displayed on the screen. The upper half of the computer display is devoted to a list of all the commands available.

Commands are issued by using control characters. This is done by holding the "control" key down on the keyboard while pressing a letter key simultaneously.

WordStar

Normally, you don't need commands, you can just type.

When WordStar is started, it provides a very clear menu of the different operations that can be performed. The menu looks like this:

```
     D=create or edit a Document
file        H=set Help level
     N=create or edit a Non-
document file   X=eXit to system
     M=Merge-print    a    file
P=Print a file
     F=File directory off (ON)
Y=delete a file
     L=change Logged disk drive
O=cOpy a file
     R=Run       a       program
E=rEname a file
```

Beneath this menu, a file directory is listed, if the directory is turned on. The F command has an alternating action that turns the directory display off if it is on and on if it is off. Usually, I press "D" which starts the document editing process.

WordStar asks for the name of a file to edit in response to a "D" command. If you give a name of a file that already exists, WordStar assumes you want to edit that file. A backup file is automatically created so if you make a severe error, only the edits entered during the current editing session are at risk. The backup file holds the text as it appeared before you started to edit.

Restoring the file is as simple as renaming the backup file. All backup files end with the characters ".BAK". WordStar will refuse to edit any file ending in ".BAK" thus protecting you from yourself.

If the file name given to WordStar for editing doesn't exist, a file with that name will automatically be created. This is how you start new files.

I have found this system to be nearly foolproof. In eight months, I have never lost a file due to the program or my own clumsiness and I have been pretty clumsy at times.

Control Codes

Though you can use WordStar by just typing normally, the real power of the program is in the use of control codes. They are represented on the screen as an up-arrow followed by a letter such as "^A". Though there are two characters shown, a control character is a single character.

The cursor can be moved around on the screen with "^S","^D","^E","^X" representing left, right, up and down respectively. Also "^A" and "^F" stand for move left and right by one word. These control characters allow you to walk the cursor around the screen to wherever you need to edit. Text will automatically scroll as you try to walk the cursor off the screen.

This way, you never need to be concerned with how to get the portion of the file you wish to edit to appear on the screen. If you move the cursor in the proper direction, the text of interest will eventually appear.

Screen editing in this manner is very natural. I frequently think of something I should have written in a previous paragraph, walk the cursor up to that point, add in the extra words and return to where I was. Even though I can't type I can almost get the words into the computer as fast as I think them!

The control characters "^C" and "^R" cause a full screen to scroll by, up or down. This allows quick movement through the text. A "^G" is a backspace which is the same character as a backspace on most keyboards. The same is true for "^I" which is a tab. A ruler just below the menu and above the text shows where the tab stops are. WordStar has absolute tab stops as opposed to relative tab stops. This is how typewriters have their tabs.

Normally, WordStar starts up in the insert mode. Characters are always being inserted into the file wherever the cursor appears. A "^V" will turn insert off. Then any charcters typed will replace characters already on the screen. A box in the upper right of the screen tells you whether insert is on or off. The "^V" will change the status from off to on and on to off. I usually use insert mode.

Prefixes

Five control characters are prefixes. They allow you to do some really fancy things to your text. Whenever a prefix control character is entered, the main menu disappears and a submenu takes its place. This submenu shows what operations are possible through that prefix. If none of the operations are desired, a space will cancel any prefix.

A "^P" (control P) is the print prefix. The next letter typed will determine a print control that will be inserted into the text. Some of these are:

S underScore toggle: this is entered before and after text that is to be underlined.

B Boldface toggle: this is entered before and after text that is to be printed in **BOLD** characters.

D Double strike toggle: similar to boldface but the characters aren't as dark.

X Strikeout Toggle: indicates that the characters between two strikeout toggles will have -'s printed over them. This is useful to indicate deleted text in a revised document.

V Subscript toggle.
T Superscript toggle.
Y Ribbon color toggle.

When a print control character is inserted into a text file, it appears as a control character. Thus **BOLD** looks like ^BBold^B in the text. This is necessary because most video displays are not capable of displaying the wealth of print enhancements possible with WordStar. Naturally, if a printer is not capable of printing an enhancement, WordStar will not be able to either.

The "^O" prefix is called the format prefix. With it you can set margins, tab stops, center text automatically and control how the display appears on your screen. The tab stop ruler can be turned on or off. The directory can be called up or deleted from the display and word-wrap. Justification and the page-break display can be turned on and off.

The "^Q" prefix is for commands. There are commands to take the cursor to the beginning of the file, the end or to any of ten markers that can be placed in the file.

These markers can be placed with the "^K" prefix, followed by the number of the marker (zero through 9). This prefix also allows you to do block operations such as move or delete marked blocks of text.

Finally, the "^J" prefix makes several help aids available. Explanations of most of the WordStar control characters are stored on disk. The "^J" prefix can call these onto the screen at any time without disturbing the text. "^J" can also be used to eliminate the menu or submenus for experienced users who prefer to see more text on the screen.

Fine Tuning

Since the control character commands are always on the menu, memorization of the special control keys is not required. I like to have my system running as smoothly as possible however and decided that the special function keys on my keyboard would be quite useful for replacing the control keys.

There is a block of memory in WordStar allocated to user-supplied machine code routines. I put a special keyboard routine here that intercepts the keycodes. If the key pressed is a normal key, it is passed on to WordStar. If it is a special-function key however, the code is transformed into a control key and then passed on to WordStar.

This way, I can turn off the main menu and still not have to memorize control keys. This also allowed me to tie in my cursor control keys for moving the WordStar cursor around on the screen.

And Dots Not All

Another type of text enhancement is possible with dot commands. WordStar, as well as several other word processors, takes advantage of the fact that periods are never used at the beginning of a line. If a period does appear at the beginning of a line, it is interpreted as a dot command. There are commands for setting line height (.LH), paper length (.PL), top and bottom margins (.MT and .MB) as well as for controlling page numbering and text headings or footings.

A special feature of WordStar is real-time recognition of dot commands. The far right column of the video display is usually blank. Text normally only uses the first 65 columns. When entering a dot command, WordStar will recognize the period as the start of a dot command an place a question mark in the far right column. This question mark will not disappear until a recognizable dot command has been typed in. This syntax checking is very useful. You don't have to wait until the text is typed to see if the command is a proper one. Some word processors will type an error message during the printing of your file if an improper dot command is found. Unfortunately, by then it's too late to fix it for the current printout.

The dot commands are the only WordStar features not explained in the submenus or the "^J" command. There is a list of these commands in the back of the manual however. That's the only part of the manual I use now.

Using WordStar

I have used WordStar for business and personal letters, addressing envelopes and writing articles such as this one. Letters are now fun because I can pull an old letter in from the disk, edit it and send it out. I haven't had to type my return address for months!

Envelopes are a little tricky because they won't look good unless the spacing is just right. I needed to waste about three envelopes before I got a format that pleased me. Now, I call in an old envelope file, edit it and print it.

Articles are the most fun to do on WordStar. Even though I outline what I intend to write, I almost always think of extra things to write about. It is very important for me to be able to go back and add or refine my writing. In addition, I never worry anymore about making mistakes because of the ease of editing.

One particularly shining example of the great benefits WordStar can provide is an article series which I am writing. I originally wrote a series of twelve articles on computer interfacing over the span of two years. After typing these into WordStar, it took me only about three hours to read in four articles, condense them, add in some linking paragraphs and thus transform a series of twelve articles into a group of three. There is no way I could have done this without a word processor. I simply haven't the time.

In addition, I have used WordStar to create assembly language source files. It is much easier to use WordStar, a word processor, than ED the text editor that comes with CP/M. I use the N option in the first menu (edit a non-document) for this purpose.

WordStar

Performance

WordStar lives in the CP/M environment on my NorthStar computer. It is thus constrained by the speed of the disk drives and software for file access. I have not found that writing has been hampered too much by delays caused by this system. A large file can require several seconds in traveling from end to beginning but I think it would take longer with a typewriter.

Fortunately, I have Quad-density disk drives. Text requires vast amounts of disk space and double- or quad-density 5.25" minifloppies or 8" floppies are a must. The new quad-drives on the NorthStar are also faster than the older minifloppy drives which is nice.

The amount of memory in your computer will determine WordStar's performance. You must have at least 45K bytes of memory to run WordStar. If you have more memory than that minimum level, WordStar will not have to swap text between memory and RAM as often. Swapping takes time and is really the only relatively slow process in the system.

WordStar's interaction with the memory-mapped video is astounding! There is almost never a delay between typing and display, even when submenus are brought up or large portions of text are changed. The Soroc configuration was definitely slower. I think that a real word processing system ought to have a memory-mapped display for performance. The only times I have to wait are when text is being transferred between disk and computer memory.

WordStar will drive my Spinwriter at full speed. The program allows the printer to print both forward and backward so time is not wasted with carriage returns. Almost all of the features available on the Spinwriter are accessible through WordStar. It seems like a very good marriage.

One feature of WordStar that I never use is the ability to edit one file while printing another. This requires more memory and destroys system performance. Both editing and printing require the use of the disk drives and printing gets priority. Thus the pauses during text paging off the disk become much longer. I find it less irritating if I ignore this feature entirely.

So far, I have experienced no program crashes due to software bugs. The only times problems have arisen were due to a determined effort on my part to kill the program or occasional static electricity discharges. Static is just plain bad for computers and MicroPro could hardly fight this problem.

There are always Bugs

I have found WordStar to operate flawlessly. It does not hide text as some programs I have used did. It does not confuse me with cryptic error messages. Best of all, it does not lose files or turn them into garbage.

The only problem I have had was in trying to use special print controls that allow my printer to print special symbols. The manual says WordStar will support this if you patch a few bytes with INSTALL's patcher.

The print controls do work, but not if they are the first characters of a line. This is a bug which MicroPro confirmed about a month after I sent them a letter of inquiry through my dealer.

Secrtary a word processor

EDGAR F. COUDAL
627 S. CRESCENT AVE.
PARK RIDGE, IL 60068
(312) 823-3834

The last thing I wanted was a secretary, which is why I got one...

Many other North Star Horizon owners are using the same secretary, as a matter of fact...and she's a beauty. Some may have known her in her younger days as "Maryelln," but she's a full-fledged professional Secretary now, grown out of her ganky early years.

As a free-lance writer and editor of 20 years standing, I bought a North Star Horizon, Diablo 1610, and Soroc 120 terminal when I decided to switch from typewriter to word processing. After testing various other word processors Issettled on the Secrtary program, which came highly recommended by another writer, but which is virtually invisible in the ads and reviews. I found Secrtary in an ad being run by American Square Computers of North Carolina, touting it at less than $100 as "the best!"

I can't speak to the "best" claim--I haven't used them all--but I do know that it's a high performance, powerful and flexible tool that has added the equivalent of two to three hours to my daily production time. And that's what I was looking for in a word processor in the first place.

Secretary was written by Gary Young of North Hollywood, CA, and has been in development for more than four years. Secretary is a line-oriented word processor using the editing commands of the North Star line editor, and supplementing them with direct and easy to understand English words.

Young began writing Secretary ("SECRTARY" in the 8-letter DOS directory) for himself, simply because there were no word processing systems available for North Star at the time.

"As others saw Secrtary, they suggested changes and features. The commercial possibilities became obvious," Young said. "My goal was to get something out that was inexpensive, and easy to learn and use. The commands are as similar to BASIC as possible.The idea was to develop a system that someone who was used to programming in North

WordStar

MicroPro's response was that version 2.25 would eliminate this bug. I was told that I could have my dealer send back my copy of the program for updating.

Another irritation, though not a bug is MicroPro's WordStar Customization Notes. These notes tell you many extra things about customizing WordStar. If you are a programmer, you are tantalized with mentions in the installation chapter of the WordStar manual of default locations for help levels, timeouts and how to add special-function keys.

Unfortunately, these notes cost about $100. With the already high price of WordStar, I have not felt justified in paying more for these notes. I have either accomodated myself to what MicroPro set up or found other methods for making changes.

Recommendation

I believe that MicroPro's WordStar is an excellent word processor and a well written program. It shows very little programmer orientation as opposed to user orientation, and provides a lot of features for the non-computer type user.

There are so many features in WordStar that I have covered less than half in this article. WordStar can do just about anything with text except do the writing for you. If you are looking for a word processor, you should definitely consider WordStar.

WordStar Addendum

After this article was written, MicroPro introduced WordStar 3.0 with new features and SpellStar, a spelling checker. I have found both of the new WordStar 3.0 features most useful. SpellStar, although very nice in concept, still has some rough edges.

WordStar's New Features

The two new WordStar features are horizontal scrolling and column-mode operation. The menus have been reorganized for easier reading. Horizontal scrolling works just like the vertical scrolling of earlier WordStars. If the line is longer than the video screen, it extends past the right edge. To see the rightmost portion of the line, the entire page is scrolled to the left. Thus the screen is a window to a text file that is as long as your disk is large and as wide as 255 characters.

I use horizontal scrolling for working with documents to be printed at 12 pitch (characters per inch) where I get 90 characters per line. That doesn't fit on my 80 column screen but with horizontal scrolling, it is quite easy to manipulate the wide text. Previously, WordStar would wrap the line around to the next, so that "What you see is what you get" was not really true. Now it is.

Column-mode operation is very useful. The intended purpose is to allow you to manipulate columns of text when making lists and it does work well for this. I have found an even better use however. In newsletter work, the 90 characters per line were hard to read. I now set margins for 40 characters per line, type in the text, reset the margins for 90 characters per line and then use the column-mode mover to create pages with two text columns, each with 40 characters per line. This electronic cut-and-paste works extremely well.

The utility of the two new WordStar features lead me to recommend MicroPro's text editor even more strongly than before. It is an excellent package made better.

SpellStar

Many people don't spell well and spelling checkers are becoming quite popular. MicroPro built SpellStar to coexist with WordStar as a program overlay. It is called from WordStar's main menu.

I have had problems with SpellStar. First off, you must have 56K of memory to run it. This requirement is poorly documented if at all. Also, the SpellStar overlay takes 30K of disk space and the 20,000 word dictionary takes 98K. This essentially makes WordStar a two-disk applications program with WordStar, WordStar overlays, MailMerge overlays, SpellStar overlays and the SpellStar dictionary taking over 200K of disk space. WordStar has never worked well running both disk drives on my Horizon. The second drive always seems to access slower than the primary drive.

SpellStar is also expensive,retailing at $250. You must also buy or upgrade to WordStar 3.0 to run SpellStar. If you need a spelling checker, you might possibly want to wait for MicroPro to get the bugs out of SpellStar or look into one of the other checkers being marketed.

Star BASIC could pick up and run with almost immediately, without memorizing new control characters or a new command language."

Further, Young said, "At the time I began developing this software, I did not have memory-mapped I/O, so it had to function without cursor movement used by screen-oriented systems. Secretary will run on a teletype or any kind of single line printing device."

He added: "It's really a word processor designed for a minimum system."

While Secretary has been known to North Star users, and those operating other mainframes running under North Star DOS, it is now available to a much broader audience of users because Young is introducing a software package that will run under the ubiquitous CP/M.

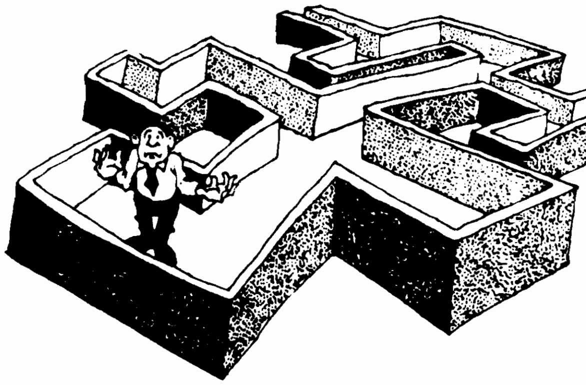
One feature of both Secretaries is the STATUS command. It displays every possible user-changeable command which would be used in inputting information, or formatting it for printing or display. The program allows you to see the final formatted output on the console, before sending it over to the hard copy printer. Among those changeable commands are:

Entry line nulls, offset from left margin, page size, number of lines to a page and number of lines to start from the top of the page, margins, output line length, up to 10 tab stop sets, spacing between lines, repetitive title for successive pages, choice of output device, repeat copies, automatic page numbering. and line fill and right margin justification. The screen display of the final format shows the right justification, which is something a lot of $10,000 stand-alone word processors are not capable o doing.

In addition, the "CONFIGURE" command allows the user to further change the program for his system, offering a choice of such features as screen size with both variable display column width and number of lines, selection of one of a variety of backspace control key devices, the option to stop at the bottom of a page, or print through to the end of the file, choice of disk density, and others. A particularly appealing feature for the fast typist is the line wraparound without carriage return. While one is hammering along, the program decides if a word will fit entirely at the end of a line. If not, it automatically moves it to the beginning of the next line.

It seems almost superfluous to list the basic word processing capabilities, since any such program would not be worth reviewing if it did not contain certain broad capabilities. Secretary features:

--Global search and replace all, or selected, occurences of the word sought.



--Insertion of character, word, line or block.

--Deletion of character, word, line or block

--Block movements, automatically deleting the old line positions.

--Block copying, without deleting the old copied block and position.

--Appending of other files

--Merging of data into letters written in Secrtary format, where the data comes from a BASIC program.

--Justify and fill lines

--Single,double or multiple spacing

--Format print, including margins, length, headers, and numbering

--Centering and underscoring

--Up to 10 tabsets for columnar material

--Forcing pages to end on a paragraph

--Operator pause and prompt, to change print wheels or paper.

And a special HELP command that presents all Secrtary commands in tidy, crossword-like block. That command is HELP. The commands are English words, such as RECOVER, but Secretary gets the drift of what you want to do after 4 letters, like RECO, DELE, CHAI and so on. The keyword list is automatically displayed on the console when an invalid command is entered.

Unlike some word processors, Secretary takes certain steps to protect the user against himself. For instance, the "DESTROY" command is followed immediately by "ARE YOU SURE (Y/N)", which gives you pause to think about what you're really doing to that innocent disk.

A second safeguard is in the "RECOVER" command, which will bring the entire memory file back into being after a SCRATCH (and before anything else has been entered), configure, JP to address, or hardware or memory error. How often have you SCRATCHed something before saving it? With Secretary, you get it back.

The BLOAD and BSAVE commands format programs written in BASIC, or which are to be saved in BASIC, so that they can be edited with Secretary. This is done with a token table, which automatically changes certain command symbols common to other BASICs to command symbols which its North Star BASIC heritage recognizes: i.e., a instead of : and , instead of ; and others. Young explains that the token table looks up the hex character, then converts it to the comparable ACSII character or word.

One drawback, Young said, is the fact that while it will fill and justify, it will not support proportional printing. It fills lines by pushing the last full word on the line to the right margin by inserting blanks after, first, every instance of two consecutive blanks in the line, then going back and adding a blank after every other single blank. On a short output line, with a lot of space to fill, this occasionally produces an unacceptably airy effect, which needs remedying through word editing.

Secretary also will take a BASIC data file, and with its MERGE command and the up arrow, use that data file for such purposes as mailing lists and form letters.

The 30-page manual covering use of the system is clear, direct, and filled with examples, as well as anticipating user questions.Young said: "I know the frustration of poor documentation and tried to take the user through each step."

Though the system itself, written in assembly language, does take up 13.5K of memory, one can use the CHAIN command to link file after file, making in effect its free memory the entire contents of a disk.

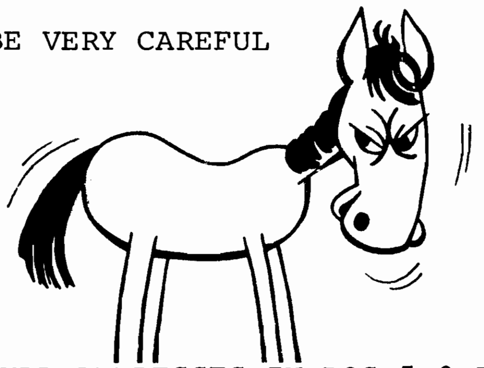
In all, the Secretary word processor does almost everything under CP/M and North Star DOS that the high-priced software does...for less than $100.

It's available by mail from Edgar F. Coudal, representing Young, at Coudal & Associates, 627 S. Crescent ave., Park Ridge, IL 60068, for $99 dollars plus California tax of 6% if ordered by someone in that state. The price covers the system disk and manual. On the disk, in addition to the SECRTARY program, is an excellent demo and tutorial file, the mailing list generator and printing programs, and other useful file.

############

2 COL LIST

```
 10  REM WRITTEN BY M.L. MADDOX NOV. 1980
 20  REM (408) 738 4339 OR (415) 855 5711
 30 PRINT "LIST THIS PROGRAM BEFORE USE!! TYPE CONT TO CONTINUE"
 40 PRINT "THIS PROGRAM USES THE FIRST 37 BYTES OF THE USER AREA"
 50 PRINT "OF DOS 5.0.  CHANGE THE ADDRESSES IF YOU WANT IT ELSEWHERE."
 60 STOP
 70 REM THIS PROGRAM CHANGES DOS TO PERMIT THE LISTING
 80 REM OF THE DIRECTORY IN TWO COLUMNS.
 90 REM THIS VERSION HAS BEEN TESTED ON 5.0 DOS AND NO OTHERS
 100 REM SINCE THE ADDRESSES IN OTHER VERSIONS ARE DIFFERENT
 110 REM
 120 REM        BE VERY CAREFUL
 130 REM
```



```
140 REM SOME USEFULL ADDRESSES IN DOS 5.0 ARE
150 REM 25FDH OUTPUT A CARRAGE RETURN LINE FEED
160 REM 25F5H OUTPUT A BLANK
170 REM 2611H OUTPUT TO THE CONSOLE (PORT 0) THE CONTENTS OF THE A REGISTER
180 REM 2616H OUTPUT TO PORT 0 THE STRING POINTED TO BY THE DE PAIR
190 REM       THIS STRING IS TERMINATED BY " (022H)
200REM 26EAH IS THE PORT NUMBER TO WHICH OUTPUT IS SENT
```

```
220REM THIS ROUTINE PRINTS 8 SPACES BETWEEN THE COLUMNS
230REM IF YOU WANT MORE OR LESS THEN MAKE THE CHANGES INDICATED
240REM ASSEMBLY LISTING USING TDL MNEMONICS FOLLOWS
250REM 29BC  00       FLAG   .BYTE   0    ,SWITCH TO OUTPUT TAB(8) OR CRLF
260REM 29BD  3A 29BC  DOUB   LDA    FLAG ,ENTRY POINT GET FLAG
270REM 29C0  FE01             CPI    1    ,IF FLAG=1 THEN CRLF
280REM 29C2  280E             JRZ    D2   ,ELSE TAB(8)
290REM 29C4  3E01             MVI    A,01 ,SET FLAG TO 1
300REM 29C6  32 29BC          STA    FLAG ,
310REM 29C9  0E08             MVI    C,08 ,SET UP SPACE COUNTER
320REM 29CB  CD 25F5  D1     CALL   BLANK,OUTPUT SPACE
330REM 29CE  0D               DCR    C    ,DECREMENT COUNTER
340REM 29CF  20FA             JRNZ   D1   ,AND LOOP UNTIL FINISHED
350REM 29D1  C9               RET         ,RETURN FOR NEXT ENTRY
360REM 29D2  AF       D2     XRA    A    ,ZERO A REGISTER
370REM 29D3  32 29BC          STA    FLAG ,MAKE FLAG ZERO
380REM 29D6  C3 25FD          JMP    CRLF ,AND RETURN FOR NEXT ENTRY
390REM 29D9  0E05     D3     MVI    C,05 ,SET UP FOR 5 SPACES TO SKIP
400REM 29DB  CD 29CB          CALL   D1   ,OVER PGM STARTING ADDRESS
410REM 29DE  C3 2589          JMP    2589H,WHEN NOT PRESENT
420REM
```

DISKS ON N★ I/O PORT

Converting the North Star Disk Controller to I/O Port Addressing

BY LANCE ROSE

For those of you who have wanted in the past to remove the disk controller ROM and disk control registers from the address space of your computer, here is a method you can use.

There are only two prerequisites for accomplishing this task but they are important ones and cannot be gotten around. The first of these is that you MUST have a Z80 CPU. An 8080 or 8085 will not work for this application. Secondly, you must have a ROM board or ROM option on your CPU board that can hold the code for booting up the disk. This is necessary because when the disk controller disappears from the memory space of the computer the on-board bootstrap ROM goes with it.

If you have a Horizon, you may already have the EPROM option on it and can use that as a place for the disk boot unless it is already taken up with a ROM monitor or other necessary software. If you have the North Star CPU but don't have the EPROM option, it can easily be added and is not very expensive.

Some other CPUs also have an on- board EPROM option and, what's even better, on some, the EPROMS can be made transparent in their addressing thus allowing the full 64K of RAM to be used by the system without even the ROM getting in the way.

If yours is a system that can't do this, at least you can locate a 2708 (or other) EPROM up in high memory such as at 0FC00H and thus use all but 1K of the memory space for RAM.

This can be desirable if you're running programs that require a large amount of RAM such as COBOL, PASCAL, or large BASIC data bases, etc.

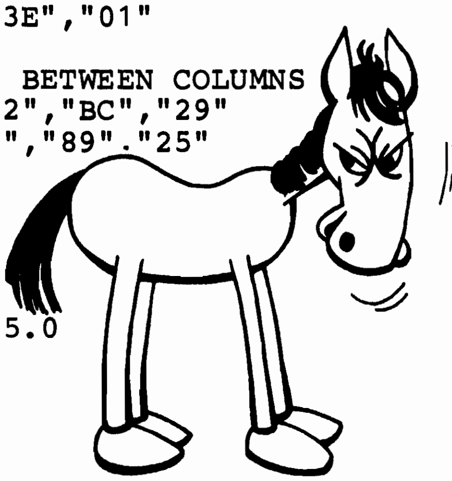
To see, first of all, why a Z80 is required for this, it is necessary to look briefly at the difference between a memory read instruction and a port input instruction. When the CPU executes a memory read, it first places the 16-bit memory address on the address lines and sets the SMEMR line high to indicate to the memory board that this is a memory read opera- tion. When the data is ready (wait states can occur), the CPU reads it in from the 8 data lines by asserting the PDBIN signal.

For an I/O read, the basic timing is the same except that some different control lines are used. In this case the I/O port number is placed on the lower 8 address lines and the SINP (as opposed to SMEMR) line is brought high, indicating to any I/O boards that this is an input (as opposed to memory read) instruction. Finally, PDBIN is pulsed to read in the actual data from the data lines.

The similarity between these two procedures makes it quite easy to convert between one process and another. To see how this is so, it is also necessary to review a procedure called

2 COL LIST

```
430REM THREE CHANGES MUST BE MADE WITHIN DOS 5.0
440REM 2033                     .BYTE  30    ,NUMBER OF LINES ON CRT
450REM 2589  CD 29BD            CALL   DOUB  ,ENABLE THE TWO COLUMN LISTING
460REM 257B  C2 29D9            JNZ    D3    ,NO STARTING ADD, SKIP
470REM  NOTE THAT BASIC CHANGES COLONS TO\
480REM  AND SEMICOLONS TO ,
490 FILL 8243,48 REM CHANGE NUMBER OF ENTRIES TO 48 FOR 24 LINE CRTS
500 DIM H$(2)
510 DEF FND(H$)
520 T=0
530 FOR E=2 TO 1 STEP -1
540   C=ASC(H$(E,E))
550   IF (C>=ASC("0")) AND (C<=ASC("9")) THEN C=C-48 ELSE C=C-55
560   T=T+C*(16∧(2-E))
570 NEXT
580 RETURN T
590 FNEND
600 DATA "D9","29","BD","29"
610 DATA  "00","3A","BC","29","FE","01","28","0E","3E","01"
620 DATA "32","BC","29","0E","08"
630 REM THIS LAST BYTE "08" IS THE NUMBER OF SPACES BETWEEN COLUMNS
640 DATA "CD","F5","25","0D","20","FA","C9","AF","32","BC","29"
650 DATA"C3","FD","25","0E","05","CD","CB","29","C3","89","25"
660 READ H$
670 T=FND(H$)FILL 9596,T
680 READ H$T=FND(H$)FILL 9597,T
690 READ H$T=FND(H$)FILL 9610,T
700 READ H$T=FND(H$)FILL 9611,T
710 M=10684REM 29BCH START OF THE USER AREA OF DOS 5.0
720 FOR I=1 TO 37
730   READ H$
740   T=FND(H$)
750   FILL M,T
760   M=M+1
770 NEXT I
780 !"FINISHED HOPE YOU LIKE THE RESULTS"
790 RESTORE 600
800 STOP
```



GOOFS

Leonard Morgenstern

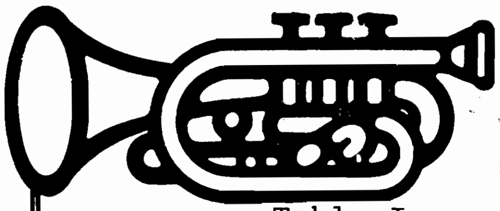
Is my face red! When I looked at the listing of my DISKLIST program in the COMPASS, May, 1981, I was horrified to find that I had sent you a preliminary version. And further, that the final version apparently is lost forever.

Below is a short program that will list the changes needed to make DISKLIST run properly for computers without memory at 0.

It is desirable that the parameter P in line 1980 be changed to P1 by all users.

```
0010 !"CORRECTIONS FOR DISKLIST PROGRAM ",
0020 !"PUBLISHED IN COMPASS V1 #2 MAY 1981"
0030!"NO MODIFICATION  NECESSARY FOR ",
0040 !"MEMORY AT 0"
0050 !"PROGRAM ASSUMES DOUBLE DENSITY.",
0055!"BASIC MUST HAVE LOG "
0060 S=11529 : L9=EXAM(S)+EXAM(S+1)*256
0070 !"YOUR MEMORY IS NOW SET TO ",L9
0080 !"CHANGES TO 'DISKLIST':"
0090 !"MEMSET ",L9-2100
0100 !"160 L1=",L9-2080
0110 !"161 L= ",L9-2099
0120 !"340 I=FNC(L,4,0,L1,1,1,1)"
0130 !"1980 DEFFNL(D,P1)"
0140 !"2010 FOR I1=L TO L+4"
0150 !"2040 RETURN CALL (L,256*D+P1)"
```

N I/O PORT



address mirroring. This refers to the fact that when a port address is placed on the lower 8 address lines, the upper 8 lines are not needed to further define the port number (which lies in the range of 0-255). W-en the 8080 was designed, it implemented address mirroring which duplicated the lower 8 bits of the address lines on the upper 8. Anyone with a front panel and single step switch can demonstrate this quite easily.

What happened was that this eventually led to problems when the Z80 microprocessor began to get popular because it does not use this procedure and some old I/O boards which decoded the upper lines to get the port address were unusable unless modified to decode the lower 8 lines. Address mirroring is not actually needed to perform the conversion but it is necessary to understand that the only reason we can change to I/O port mapping is because of the upper 8 bits of the address bus not being needed to define the I/O port number.

The Z80 uses the high byte of the address in a couple of different ways during I/O operations. In the multiple byte inputs and outputs, it serves as a byte counter that can be decoded by an I/O device to control the data transfer. For single byte I/O transfers (which is what we will be using here) the high byte of the address contains the contents of the accumulator at the time of the operation. This can be the basis for simulating a 16-bit address to the disk controller.

Ordinarily, the disk is accessed by reading from a memory location in the address range of the controller. For the standard version, this is E800-EBFF. This area also includes the bootstrap PROM. Since, on an input instruction, the port address appears on the lower byte of the address bus, we need to shift it to the upper byte to form the most significant byte of the total address. This lets us place a data byte to be written to the disk in the accumulator and, with the

Table I.
Connections for single density boards:

IC			
5G, pin	13R	to	Bus line 83
6G	5R		82
6G	11R		29
6G	13R		30
6G	3R		31
6G	1R		81
6G	9R		80
5G	1R		79
5G	11R		32
5G	9R		86
4G	11R		85
4G	13R		33
5G	5R		87
4G	1R		37
4G	3R		34
4G	5R		84
8G	9R		46

Table II.

Connections for double density boards:

IC			
10D, pin	11R	to	Bus line 83
11D	15R		82
11D	17R		29
10D	13R		30
11D	11R		31
11D	13R		81
10D	15R		80
10D	17R		79
7D	17R		32
7D	15R		86
7D	13R		85
7D	11R		33
7D	8R		87
7D	6R		37
7D	4R		34
7D	2R		84
5D	5R		46

(R means remove from socket when making the connection.

The last connection in each table substitutes the SINP signal for SMEMR.

N★ I/O PORT

upper/lower byte address lines reversed, it will appear as what was formerly the least significant address of the address itself.

Without worrying about the software patches at this point, I will describe what must be done hardware-wise to the disk controller to make it possible to read and write to it on an I/O basis.

The address lines A0-A15 are buffered as they reach the board by either Schmidt trigger gates (single density version) or tri-states (double density version). The outputs from these gates then go to the on-board PROM (high lines) and other decoding circuitry including the write shift register (low lines).

The neatest way to reverse these and which doesn't require any trace-cutting is to take the input pins of the buffer chips and bend them out from the socket and then attach jumper wires to the pins.

Ordinary wire-wrap wire works qiute well as jumpers. A very fine point pair of needle nose pliers seems to serve better than any of the wire-wrapping tools for making the actual connections. Table I lists the connections that have to be made for the single density board and Table II the double density board.

The solder connections can be made to the contact fingers of the PC board. Just be sure you make them far enough from the board edge to prevent interfering with the motherboard connector when the controller board is inserted into its socket.

This whole procedure shouldn't consume more than an hour or two and the results can be well worth it. Aside from removing the disk controller from the address space, I noticed in my own board that a problem I had with the motors turning on occasionally for no reason vanished when the board was converted to I/O addressing.

In an upcoming second part to this article, I plan to discuss the software changes needed to go with this modification and plan to make available the patches to Release 5.2 DOS and boot PROM that are necessary to get it operational. To date I have been using my own single density board with Release 4 DOS and Release 5.1 BASIC and utilities and it functions just as before.

If you need more memory space, this is certainly one way

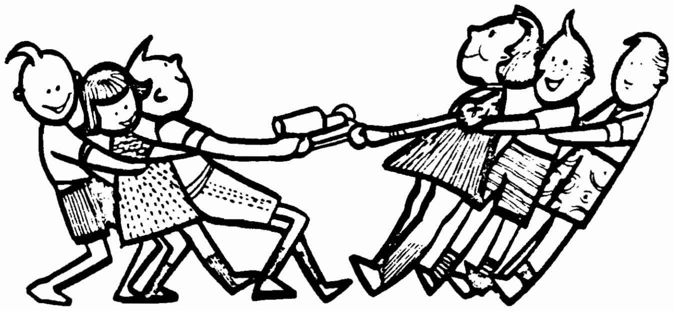
CLIP TIPS

Removing S-100 boards from the HORIZON motherboard can be a real hassle. I have discovered an easy and inexpensive remedy. I retrieved a discarded 12-inch plastic knitting needle from my junk box and converted it into a long, thin J-hook. A few seconds in the steam from a tea kettle was all that was needed to heat the plastic to the point where it could be easily bent (using pliers). The J-hook is easy to insert under the ends of the S-100 boards. A few tugs on each end is all that is necessary to remove even the most stubborn board. And no more cussin'.
PWJonas

GROUPS

LOCAL NORTH STAR USER GROUPS

Organizations interested in listings their meeting locations, etc to attract new members, should drop a line to the editor of the COMPASS for publication here. The COMPASS currently reaches about 800 INSUA members. Publication is free ofcourse.



CHICAGO AREA NORTH STAR USER GROUP

By Edgar F. Coudal

In response to Bill Banaghan's plea in the last issue of Compass, the Chicago Area North Star Users Group (CANSUG) will contribute a regular column. Following is the first, which is really a collection of excerpts from recent issues of the Chicago Area Computer Hobbyists Exchange (CACHE) bulletin.

CANSUG formed its user library in July; it is made up of common domain software (i.e., anything not under license, such as Wordstar) and is available to all members for copying and personal use. Group members are active in developing their own software, especially in such areas as business applications and utilities. Don Budsberg, the Technical Wizard of Lillipute Computers, has volunteered to act as librarian, along with group leader Steve Bogolub.

At the November meeting, Budsberg reported that the library contained 73 NDOS/BASIC 5 1/4" disks and 91 8" CP/M disks, made up of a great variety of games, utilities, business and educational programs. All disks are cataloged by Directory listing, and descriptive files will be added to each, ala ZOSO, eventually.

The subject of first technical meeting was modems and remote terminals, and their use and application. Also displayed on the two machines present were new word processing software packages: Lettergo, from Datek Systems, Inc., of Arlington, VA., which does everything Wordstar will do, except under N* DOS rather than CP/M; Benchmark, another powerful cursor-oriented WP for the N* DOS, and Secretary, a cheap ($99!) line-oriented processor that does things even the big ones don't, such as showing right margin justification on the screen, but which may be most useful as a super BASIC program editor.

At the following CANSUG general meeting, it was noted that Sony is now offering a tiny dictating machine, which uses a 3" floppy disk, and which, it is rumored, will be available with an RS232 interface. This would make the device an ideal traveling companion.

One other rumor note: Diablo is said to be readying a letter-quality printer, aimed at the home computer market, which will retail at around $800. A Diablo guy who claims to have seen it says that it's light and small, but with excellent print characteristics. However, it's not designed for the pounding an office or business environment would generate. "Early in 1982" is the offering date, he sayeth.

One of the members' continuing and expensive mail-order hassle with Infosoft over its mock CP/M-North Star utility is a horror story all too common. As long as we're on the subject, a West Coast outfit offered a remarkable catalog of used software...at dirt-cheap prices. The range of software...stuff not offered anywhere else, big IBM stuff scaled down to run on the N*, and so on...was too good to be true. It was. It leads one to believe that you're only safe dealing with the most widely advertised and proven software suppliers. On that score, Lifeboat and Dynacomp, to mention just two, get high marks.

Another feature at the September meeting was a demonstration of Supercalc, the Visicalc lookalike that runs on machines other than the Wormy. Jake Farber, President of Lillipute, was good enough to bring along the hardware and demo Supercalc at length for all interested. Seems to do all the things that have made Visicalc the most important single cosntributor to Apple's success.

--Steve Keith is putting together the Compuprism board, which claims to bring full graphics to the Horizon. He's also fooling with the same company's hardware that claims to let any North Star user run TRS-80 software.

--For North Star users who have written salable programs, check with Dynacomp (ads in Byte and other magazines). Charlie Walsh there is desperate for new programs, reports back quickly,and offers a royalty-payment compensation arrangement. He doesn't want games, unless they're very special. He does want business-oriented stuff.

--Joe Alonso has launched Omni, a software publishing house aimed primarily at business and accounting users. Joe has perhaps the broadest range of North Star DOS based software available in the business area and his prices are competitive. Especially of interest is his 1040 income tax preparation package. Joe is an accountant, and all software has been proven in his own operations. In November, it was announced that eight more disks--loaded with financial applications programs--have been added to the first 23 in the catalog of OMNI Software Systems, Inc., 146 Broad St., Griffith, IN., which further strengthens OMNI's claim to being the biggest publisher of such software in the country.

A recent column in Micro-Systems (Sol Libes' old S-100 magazine) indicated that North Star has put 22,000 computers into the field, with another 12,000 or so running the North Star operating system. That's big business. Of course, North Star is one of the grandaddies of the micro industry, now being all of five years old.

Word has it that the North Star Horizon will be offered by a Chicago area original equipment manufacturer, a large office machine company getting into micros for the first time, under that company's own name.

The December technical meeting involved a vivisection of the Horizon, and explanation of how it all works, along with a demo of Diagnostics II, the powerful set of hardware testing utilities from Supersoft, Inc., of Champaign.

While the above report is pretty fragmented, it should serve to give the national group some idea of what we're up to here in the Midwest--the "Silicon Prairie."

FIND-CHANGE VARIABLES

A NEW BASIC PROGRAM TECHNIQUE

OVERVIEW

An existing business program,which has been 'production' for about two years, contains a large number similar and repititious statements. These statements are used to implement the adding of data to various assigned variables.

This data is brought in from disk as random access files. The data is read and depending on the account variable involved, is assigned to a line number via an ON GOTO command. This assigned variable ranges from A(X) TO Z(X) Where X can be any number from 1 to 18. Thus when adding, a line typically could take on the form:

```
 1660 A(X)=A(X)+T1
```

T1 being some numerical value to be added and stored to the value of A(X).

The above is the straight forward/conventional method of BASIC programming for this type of task.

What is proposed is not conventional. The idea is to use a single general statement and let it be altered by the computer prior to using it as a Subroutine.

As usual,this new idea is not completely new. FILL statements have been used many times and in many ways.

What is new is the assigning and the replacing a specific variable in place of a general variable of a general statement and not,repeat not, using a series of statements pointed to by a GOTO command. This is close to giving your computer a bit of intelligence. It reads the incoming data, writes the variables necessary in the general statement and then GOSUB to solve the equation with data on hand. At any given moment, during this part of your program, you wont know which particular equation is being solved.

The time for the computer to execute this method vs the original way has not been measured, although no significant change in the execution time for the overall program has been noticed. The block size required was reduced from 57 to 54 blocks or approximately 750 bytes net reduction for this one application.

One of the main goals of this write up is to hopefully inspire others to apply this idea to whatever. Who knows where it might lead.

IMPLEMENTATION

We are going to make extensive use of the early part of the BASIC program. So RENUMBER your present program to start at line 100.

Add a new first line,say,Line 5.

```
5 GOTO 100
```

Add line 10 to contain the general statement you wish to solve. In my business program I used:

```
10 A(X)=A(X)+T1
```

Since this to be a Subroutine:

```
 15 RETURN
```

```
1535 REM******READ FILE AND LOAD ACCT VARIABLES*********
1540 GOSUB 2270\REM IS PRINTER TO BE ON OR OFF?
1560 FOR J= 0 TO N-1
1570 READ #0%J*S1+5,E$
1580 X$=E$(1,5)
1590 U1$=E$(6,15)
1600 X=VAL(X$(2,3))
1610 T1=VAL(U1$)\REM T1=AMT OF DOLLARS AND CENTS (REAL)
1620 IF 72-ASC(X$)>=0 THEN 1630 ELSE IF 81-ASC(X$)>=0 THEN 1640 ELSE 1650
1630 ON ASC(X$)-64 GOTO 1660,1670,1680,1690,1700,1710,1720,1730
1640 ON ASC(X$)-73 GOTO 1740,1750,1760,1770,1780,1790,1800,1810
1650 ON ASC(X$)-81 GOTO 1820,1830,1840,1850,1860,1870,1880,1890,1900
1660 A(X)=A(X)+T1\GOTO 1910
1670 B(X)=B(X)+T1\GOTO 1910
1680 C(X)=C(X)+T1\GOTO 1910
1690 D(X)=D(X)+T1\GOTO 1910
1700 E(X)=E(X)+T1\GOTO 1910
1710 F(X)=F(X)+T1\GOTO 1910
1720 G(X)=G(X)+T1\GOTO 1910
1730 H(X)=H(X)+T1\GOTO 1910
1740 J(X)=J(X)+T1\GOTO 1910
1750 K(X)=K(X)+T1\GOTO 1910
1760 L(X)=L(X)+T1\GOTO 1910
1770 M(X)=M(X)+T1\GOTO 1910
1780 N(X)=N(X)+T1\GOTO 1910
1790 O(X)=O(X)+T1\GOTO 1910
1800 P(X)=P(X)+T1\GOTO 1910
1810 Q(X)=Q(X)+T1\GOTO 1910
1820 R(X)=R(X)+T1\GOTO 1910
1830 S(X)=S(X)+T1\GOTO 1910
1840 T(X)=T(X)+T1\GOTO 1910
1850 U(X)=U(X)+T1\GOTO 1910
1860 V(X)=V(X)+T1\GOTO 1910
1870 W(X)=W(X)+T1\GOTO 1910
1880 X(X)=X(X)+T1\GOTO 1910
1890 Y(X)=Y(X)+T1\GOTO 1910
1900 Z(X)=Z(X)+T1\GOTO 1910
1910 NEXT J
1920 GOTO 2250
```

FIGURE TWO

VARIABLES

The next thing we must do is to assign a value to a pointer let's call it 'S'. Since the program statements begin immediately after the end of the BASIC interpreter in memory we will start there with EXAM statements in a loop subroutine until we have located the first 'A' and the token for '(' in line 10.

At this point, it might be well to become familiar with the BASIC STATEMENT STORAGE FORMAT, see page 3 of INSUA Reference Data Guide and the N* TOKEN CODES found in the same reference and in Vol.1 Number 1 of COMPASS under CLIP TIPS.

Assume BASIC at 2D00, then 2D06 and 2D07 in reverse order becomes 5F9E HEX or 24478 DEC. (ENDBAS)

We will use that address as the beginning of our 'S' locator loop.

```
20 FOR U=24478 TO 25000
25 IF EXAM(U)<>65 THEN NEXT U
30 IF EXAM(U)+1=224 THEN EXIT 50
40 NEXT U
50 S=URETURN
```

The 25000 number was chosen as one that would be large enough to encompass the byte locations desired. Also please note that the programming technique to find the pointer 'S' differ from the one given here and the example to follow. Not to worry, just showing two different ways at arriving at the same place. Also note, that the number of EXAM lines in the actual program equals 4 while the example shows only 2. Actually, this could be as little as one or much greater than the four tests we have shown.

It is really a decision of how much confidence you may want that you have arrived at the one and only place designated.

Figure one is an excerpt from the business program that was changed to incorporate this idea.

Figure two is an excerpt from the same program before the change was made.

the computer is NorthStar business

Making it work harder is ours

JOIN
YOUR USERS
ASSOCIATION
TODAY

1.A source of information for new as well as advanced users of North Star disk systems

2.A source of programs and utilites from a central software exchange library

3.A quarterly newsletter of practical techinques and application programs designed specificly for your North Star system

4.A liason between local users groups to promote the exchange of information usefull to all North Star users

5.An independent non-profit users representive providing feedback to North Star Computers inc.

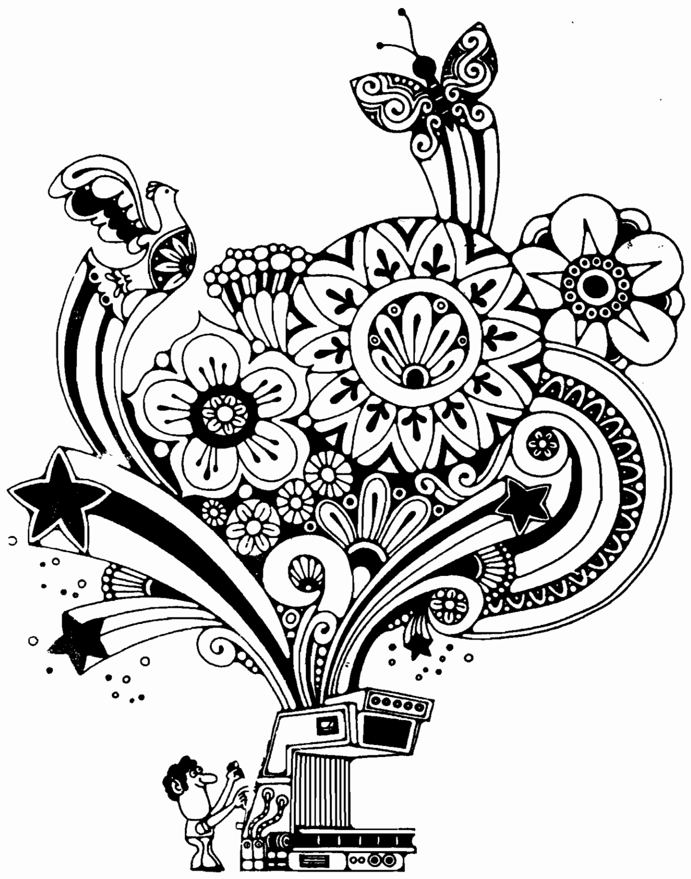
DUES ARE $20 PER YEAR

INTERNATIONAL NORTH STAR USERS ASSOCIATION
PUBLISHERS OF THE COMPASS NEWSLETTER
POST OFFICE BOX 1318, ANTIOCH, CA. 94509

VARIABLES

```
5 GOTO 160
7 A(X)=A(X)+T1
8 RETURN
20 FOR U=20507 TO 49000
22 IF EXAM(U)=65 THEN 24 ELSE 30\REM LOOKING FOR ASCII 'A'
24 IF EXAM(U+1)=224THEN 26 ELSE 30\REM LOOKING FOR THE TOKEN OF'('
26 IF EXAM(U+2)=88THEN 28 ELSE 30\REM LOOKING FOR ASCII 'X'
28 IF EXAM(U+3)=41THEN EXIT 32\NEXT U\REM LOOKING FOR ASCII ')'
30 NEXT U
32 S=U\REM SETS 'S' POINTER VALUE
34 RETURN
160 DIM A(38),B(38),C(38),D(38),E(38),F(38),G(38),H(38),J(38),K(38)


1535 REM******READ FILE AND LOAD ACCT VARIABLES**********
1540 GOSUB 2270\REM IS PRINTER TO BE ON OR OFF?
1550 GOSUB 20\REM FIND START OF TEXT OF LINE 7
1560 FOR J= 0 TO N-1
1570 READ #0%J*S1+5,E$
1580 X$=E$(1,5)
1590 U1$=E$(6,15)
1600 X=VAL(X$(2,3))\X9=ASC(X$(1,1))
1605 REM X=INDEX # OF A SUBSCRIPTED VARIABLE
1606 REM X9=ASC VALUE(DEC) OF THE ALPHA NAME OF THE VARIABLE
1610 T1=VAL(U1$)\REM T1=AMT OF DOLLARS AND CENTS (REAL)
1620 FILL S,X9\FILLS+5,X9\REM THIS CHANGES LINE 7'S VARIABLE NAMES
1630 GOSUB 7
1650 NEXT J
1920 GOTO 2250
```

FIGURE ONE



THE I/O FARM

by
Steve Leibson
4040 Greenbriar Blvd.
Boulder, CO 80303

In this newsletter, we will be returning to the North Star Horizon I/O. We will be discussing the parallel output port in the Horizon. This is usually used to attach a parallel-interfaced printer to the computer.

Parallel Ports

A parallel I/O port transmits bits of information in parallel. That is about the only general statement that can be made about them. There are no standards to follow for connectors, number of bits transmitted at one time, handshake signals, signal levels or signal timings. In other words, you are mostly on your own.

Out of this seeming chaos, printers have managed to acquire what is known as a "standard Centronics parallel interface". Unfortunately, this phrase is as misleading as the "standard RS-232" interface discussed in the first I/O Farm. I am going to explain the Centronics parallel interface, and will then tell you how printers and computers try to implement this interface.

Interfacing Centronics Style

The Centronics parallel interface is a seven-bit parallel interface. As we discussed in the last newsletter, the ASCII character set requires seven bits to represent each character. Each of these seven bits is sent on a separate wire from the computer to the printer. That is why this is called a parallel interface. The seven bits are sent in parallel.

Most parallel interfaces have handshake lines. These signals serve to regulate the flow of data over the interface; no faster than the sender can transmit, no faster than the receiver can accept. It takes two signals to have a handshake. One signal, controlled by the sender, informs the receiver that a new piece of data is on the data lines. The other signal, controlled by the receiver, informs the sender that the data has been accepted.

I can offer no explanation as to why Centronics created a three-wire handshake for their printers. No other printer or computer manufacturer has yet bothered to fully implement this handshake. They simply cannot figure out what to do with that third wire!

Figure 1 shows the sequence of events in the Centronics handshake. First, the computer is supposed to check both the ACKNOWLEDGE* (note that a star following the signal name means that signal is asserted when it is at its low voltage level and negated when at its high voltage level.) and BUSY to make sure that the printer is ready to receive data. ACKNOWLEDGE* should be high (negated) and Busy should be low (also negated). Both ACKNOWLEDGE* and BUSY are signals controlled by the printer.

When the computer decides that the printer is ready for a data transfer, it places the data on the data lines and waits. Nothing happens during this wait time, it is a settling period to allow the signal levels at the other end of the printer cable to settle. After at least one microsecond, the computer asserts DATA STROBE* by driving it low for at least one microsecond but not more than 500 microseconds. The computer then negates DATA STROBE* by driving the signal line high. After negating DATA STROBE* and waiting at least one more microsecond, the computer is allowed to remove the data from the data lines.

Some time after the negation of the DATA STROBE* signal by the computer, the printer acknowledges the initiation of a data transfer by asserting BUSY, driving it high. Some undetermined time after this the printer negates BUSY. After another indeterminate time the printer asserts ACKNOWLEDGE* and then later negates it. This low-true pulse on ACKNOWLEDGE* completes the data transfer cycle.

The above explanation is based on the way some real Centronics printers (models 101, 102, 103, 104, 301 and 306) work. Printers not made by Centronics but claiming to have "Centronics-compatible" interfaces don't really work this way.

Some printers only generate the BUSY signal, others only the ACKNOWLEDGE* signal. The MP1886 printer can be strapped to generate either BUSY or its complement, BUSY*. Some standard.

What Can Go Wrong?

Now, let's consider what problems we might encounter in connecting a "Centronics-compatible" printer to a computer. First, let's say the computer will be running the handshake from software. It looks at both BUSY and ACKNOWLEDGE* and decides the printer is ready for data. The computer places the data on the data lines and pulses DATA STROBE*.

The computer is now supposed to wait for BUSY to pulse high and then ACKNOWLEDGE* to pulse low. If we write our software dumb enough, we will only look for BUSY to be low and ACKNOWLEDGE* to be high. Note the problem here. This condition is met both before and after the printer has acknowledged the data transfer. Thus we can't use this approach because the program cannot tell the difference between a printer which is processing data but has not yet told the computer, and a printer which has completed the data transfer.

If we write the software so that the computer looks for the pulses, it may never complete the first handshake. Why? Assume that our software is looking for the ACKNOWLEDGE* pulse. This pulse can be as short as 2.5 microseconds. A software loop on our Z80 which reads in the status of the ACKNOWLEDGE* line, masks off the extra seven bits, compares it to a zero and loops until it is a zero will take considerably longer than 2.5 microseconds. The ACKNOWLEDGE* pulse could occur while our software is looking at an old piece of data and be gone before the processor can read the state of the ACKNOWLEDGE* signal again. The same holds true if we were to look at the BUSY signal instead.

THE I/O FARM

These problems are due to the Centronics' use of a non-interlocked handshake. Signals may come and go as they please, their timings determined solely by one device or the other. A fully interlocked handshake uses a signal sequence where a signal is never negated until acknowledged by the device receiving that signal. Thus a software routine has as long as it needs to complete the handshake.

Finally, North Star's Interface

The engineers at North Star apparently understood the Centronics interface because the Horizon parallel port has hardware to capture the ACKNOWLEDGE* pulse. Information about the Horizon parallel output port is contained in the North Star Horizon Computer System manual (HRZ-D-DOC) on pages 76-78. The schematic for the parallel output port section of the Horizon motherboard is on page 91 of this manual.

My Horizon's parallel ports came unconfigured. If you are going to use this port, you must be sure that the parallel port control header at motherboard location 9C is wired properly. The instructions on pages 77 and 78 clearly tell you how to wire this header. The manual does not tell you in detail why these particular connections are being made. I will.

A. Connect pin 1 to pin 14 of the header to permanently enable the data output latch. Explanation: Pin 1 on the control header is ground. Pin 14 is connected to pin 1 of the 74LS373 data latch at location 8A. This pin is an output enable. Unless this pin is at a low voltage (grounded), the latch outputs are disabled and cannot drive the data lines on the parallel output port.

B. Connect pin 2 to pin 13 to cause a positive-going edge of the acknowledge signal to set the output flag flip-flop. Explanation: Note again the ACKNOWLEDGE* signal in figure 1. The last transition that this signal makes in a data transfer is from a low-voltage level to a high-voltage level. This is called a positive-going edge.

Note also that this transition is absolutely the last event in the data transfer cycle. This edge is used to set a flip-flop, which remembers that the edge occurred, signifying the end of the transfer. The flip-flop save this information so that the slow Z80, executing a handshake program can read the state of the flip flop and see that the ACKNOWLEDGE* signal has pulsed low and then returned to a high signal level.

C. Connect pin 4 to pin 12 to signal the printer of each new data byte to be output. Explanation: This step connects the motherboard signal LD-PO to the output signal marked SPARE* through an inverter. LD-PO is a signal that pulses high when the Z80 places a data byte in the parallel output latch. This high-true pulse is transformed into a low-true pulse by the inverter. This low-true pulse almost meets the requirement for the DATA STROBE* signal of the Centronics parallel interface.

The reason for the almost qualifier is because of the one microsecond delays which surround the DATA STROBE* pulse. The data is supposed to be on the data lines at least one microsecond before the pulse. The Z80 timing driving the Horizon parallel port does not provide this "set-up" time.

Fortunately most printers don't require a long delay and the North Star implementation can work. Unfortunately, the software designed for this hardware will not work as we shall shortly see.

D. Step D tells you to wire your printer to the parallel port. To do this you need to know what kind of connector your printer has and what signals are connected to which pins. "Centronics-compatible" doesn't always mean that the printer uses a Centronics-type connector. If your printer does use a Centronics-type connector with the Centronics signal pinout, here is what it is supposed to look like:

Signal Name	Pin Number
DATA STROBE*	1
DATA 1	2
DATA 2	3
DATA 3	4
DATA 4	5
DATA 5	6
DATA 6	7
DATA 7	8
DATA 8	9
ACKNOWLEDGE*	10
BUSY	11

Pins 19-30 are logic ground wires. These must also be connected to provides a signal leve reference and signal current return path. The connector is a 36-pin Amphenol "Blue Ribbon" connector.

The North Star manual gives the following routine for outputting data on the parallel port:

```
POUT    IN   6        READ MOTHERBOARD STATUS
        ANI  1        MASK TO GET PO FLAG
        JZ   POUT     PRINTER NOT YET READY
        MOV  A,B      RESTORE DATA TO A-REGISTER
        OUT  0        OUTPUT DATA TO PRINTER
        MVI  A,20H    LOAD COMMAND BYTE TO A REGISTER
        OUT  6        RESET PO FLAG
        RET
```

This routine will not always work. Note that the first instruction, IN 6 reads the state of the PO flag. Recall that this flag indicates whether the ACKNOWLEDGE* signal has pulsed. The ANI 1 masks out all bits of port 6, the motherboard status byte, but the least significant. This bit is the parallel output (PO) flag. After the ANI, if the PO flag was zero, the A register will be zero and if the PO flag was a one, the A register will be a one.

Next, a JZ POUT is performed. This causes the processor to repeat the input from port 6 until the PO flag is set. The PO flag can only be set if the ACKNOWLEDGE* signal pulses. This explains why the Horizon power-up initialization routine unconditionally outputs a character to the printer. It is forcing the printer to pulse ACKNOWLEDGE* so that the first time a print is attempted, the PO flag will have been set.

Consider the consequences of not sending out a character unconditionally in the initialization routine. The PO flag will not have been set by the ACKNOWLEDGE* signal because the printer hasn't been sent anything. The first time a print is attempted, the POUT routine will loop forever waiting for the printer to set the PO flag with a pulse on ACKNOWLEDGE*. This will never happen because the printer never received a pulse on DATA STROBE* from the computer. The result is that the computer locks up and has to be reset. Warning, if you have a parallel interfaced printer on your Horizon, don't mess with the JMP TIN1 at the end of the initialization routine or you will lock the computer up as soon as you try to print.

Even if the computer makes it past the loop, it probably won't get the second character out to the printer. After the loop, the MOV A,B places the byte to be output in the A register and the OUT 0 sends it to the printer. This will automatically toggle the DATA STROBE* signal, which won't meet data set-up time requirements. Then the MVI A,20H and OUT 6 instructions clear the PO flag so that the printer can respond to this latest transfer.

THE I/O FARM

What if the printer responds to the transfer by pulsing ACKNOWLEDGE* before the OUT 6 instruction is performed? The computer will lock up on the next character to be output because it has lost the PO flag again.

Fixing the North Star Parallel Interface

North Star recognized the shortcomings of their parallel interface and changed the software in the Errata & Additions to the System Software Manual dated 7/29/79. The new software looks like this:

```
COUT2:  IN   6        ;READ MOTHERBOARD STATUS
        ANI  1        ;MASK TO GET THE PO FLAG
        JZ   COUT2    ;PRINTER NOT READY YET
        MVI  A,20H    ;RESET PO FLAG
        OUT  6
        MOV  A,B      ;CHARACTER TO ACC
TIN1:   ORI  80H      ;SET STROBE FALSE
        OUT  0        ;SEND CHARACTER
        XRI  80H      ;TOGGLE STROBE
        OUT  0
        XRI  80H      ;TOGGLE STROBE
        OUT  0
        ANI  177Q     ;MASK DOWN TO ASCII CHAR
        RET
```

Now what has happened? It took me a while to puzzle this out from just the software but here is the story. The routine between COUT2: and TIN1: eliminates the problem of clearing the PO flag after the character has been sent. Now the flag is cleared before sending the character so that even if the printer issues the ACKNOWLEDGE* pulse instantly, the PO flip-flop will catch it.

Even more important, it seems that North Star is not using the SPARE signal line as the DATA STROBE* any more. Remember, I said that the Centronics parallel interface really used 7 data lines. The eighth is not needed for ASCII. North Star has taken advantage of this by using the eighth data line for DATA STROBE*.

At TIN1:, the DATA STROBE* (most significant bit in the data byte) is set high by the ORI 80H instruction. This makes sure that DATA STROBE* is negated when the data is first output. Then the XRI 80H flips the most significant bit to a zero and the subsequent OUT 0 causes DATA STROBE* to be asserted. Another XRI 80H flips the eighth bit back to a one and the OUT 0 negates DATA STROBE* once again.

Now, there is a long set-up period between the first OUT 0 and the second. This new implementation does meet all the needs of the Centronics parallel interface and should work with "Centronics-compatible" printers if you connect them correctly.

How do you connect the North Star's parallel port?

North Star Parallel Output Port

North Star Name	Pin Number on 15-pin connector	Centronics Name
Data (bit 0)	5	DATA 1
Data (bit 1)	12	DATA 2
Data (bit 2)	4	DATA 3
Data (bit 3)	11	DATA 4
Data (bit 4)	10	DATA 5
Data (bit 5)	2	DATA 6
Data (bit 6)	9	DATA 7
Data (bit 7)	1	DATA STROBE*
ACK*	7	ACKNOWLEDGE*
Grounds	3,13,14,15	Grounds

These connections will only work if you use the parallel port driver software in the second listing of this article. Also, your printer must generate a negative ACKNOWLEDGE* pulse.

Some printers generate the BUSY signal instead the ACKNOWLEDGE*. If your printer is in this group, you must change one connection. Go back to the North Star Horizon Computer Manual, to page 77. Step 8 has a parenthetical note about wiring the control header that says "Connect pin 3 to pin 13 if a negative-going edge should set the flag flip-flop."

Look once more at the timing diagram in figure 1. If a printer used BUSY alone for the handshake, the last event of a data transfer is the high-to-low transition of BUSY. This is a negative edge. By making a single wire modification to the control header, you can accomodate printers which generate either handshake signal of the Centronics parallel interface.

This was a thorough coverage of the North Star parallel output port used as a Centronics printer interface. This is not the only use for this port. Almost any simple parallel scheme can be implemented using appropriate connections on the control header and some modification to the driver software.

Unfortunately, I cannot give you a thorough explanation of your printer because they vary so much. If you have successfully connected a printer to this port, why not write it up for The **Compass**? It could help out a lot of readers.

In the next issue, I will be discussing "Warp Drive for WordStar" which will tell you how to modify your WordStar so that it can recognize any special keys on your keyboard. On my system, I have 24 extra keys. They are now labeled for WordStar and I have never learned the cursed control codes.

I/O Farm Quickie Project Number 1

by
Steve Leibson
4040 Greenbriar Blvd.
Boulder, CO 80303

Bit Rate Switches for the Horizon

The North Star Horizon has powerful serial interfacing capability in it's two serial ports. There is a difficulty in using that capability. Though most of the parameters of the serial ports are programmable, the bit rates that the ports operate at are not. To change the bit rates, you must open the Horizon up, remove a header, resolder some connections, replace the header and replace the computer's cover. All this work is most unnecessary.

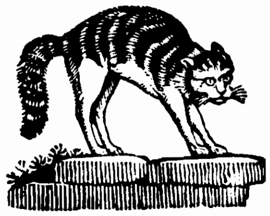
This article will show you how to bring the signals that control the serial bit rates to the back panel where they may easily be changed. A later article will show you how to make the bit rates programmable.

First, you will need a few parts. All of these may be obtained at your local Radio Shack. Feel free to substitute any parts you have laying around, nothing is critical.

Parts List for I/O Farm Quickie Project #1

Quantity	Description	Radio Shack Number
3	16-pin Wire Wrap socket	276-1994
1	Prototype circuit board	276-158
1	16-pin DIP jumper	276-1976
2	8-position DIP switch	275-1301

These parts will cost you $12.70 at the Radio Shack. In addition, you will need a Wire-Wrapping tool. If you don't have one, Radio Shack will sell you one for $5.95. You will also need some Wire Wrap wire. This project may also be wired using soldering techniques. If you do solder it together, substitute regular sockets (Radio Shack 276-1998) for the Wire Wrap sockets.



MUMPS FOR NORTH STAR

The latest version of 8080 MUMPS is now available in NorthStar format disks running under CP/M (56K minimum). The MicroMUMPS User's Group (c/o Dr. Richard Walters, Dept. of Community Health, U. of California, Davis, CA. 95616) has recently announced the availability of version 2.50 in both standard 8" single density and NorthStar format.

MUMPS (Massachusetts General ospital Utility MultiProgramming System) was developed in the mid 60's as a general purpose interpretive language. It contains a powerful set of string manipulation capabilities, a pattern matching facility, and timing functions, all of which facilitate the creation of conversational programs. File management is transparent to the user as tree structured files ("globals") are accessed as virtual arrays. MUMPS has been approved by the American National Standards Institute as a national standard language, joining FORTRAN, COBOL, and PL/1 in that category.

The MicroMUMPS User's Group offers a service which provides for automatic updates of 8080 MUMPS object code together with new applications and news releases on new development.

This service is available for an annual subscription rate of $93.00; every three months, the subscriber will automatically receive a new disk with updated object code and new or revised applications and a release notice with news and other information of general interest. The subscription will be renewable on an annual basis.

In addition, the current version of the system is also available on a one-time basis for $33.00 per copy; the source code is available on a $33.00 per copy basis as well.

Contact Dr. Walters at UC Davis.

I/O Farm Quickie Project Number 1

by

Steve Leibson
4040 Greenbriar Blvd.
Boulder, CO 80303

Bit Rate Switches for the Horizon

The first step is to place the 16-pin sockets on the board. Do this so that two of the sockets peek through two of the large punched holes in the back panel of the Horizon. These sockets will hold the DIP switches and allow you to select different bit rates for the two serial ports. Place the third socket anywhere else on the board. Once the sockets are in position, solder them in place, if the circuit board you are using has copper donuts around the socket pins.

Next, wrap the connections shown in figure 1. Be sure to note that the bottom socket is for the right serial port and the top socket is for the left serial port. This is not critical but it helps if you are consistent with the documentation.

Once the sockets are wired, you will need to drill or punch the circuit board for the mounting holes. You should find some 4-40 nuts and bolts for mounting the board, along with some stand-offs to allow room for the DIP plug and cable you have purchased.

After punching the holes, plug the two DIP switches into the switch sockets. Make sure the switch labeled 1 is plugged into pin 1 of the socket. Also plug the DIP plug and cable into the cable socket, making sure pin 1 is plugged into the socket's pin 1. Then mount the completed board to the back panel, inside the Horizon.

Remove the DIP header at 2D on the North Star motherboard. Plug the free end of the DIP plug and cable into the emptied socket at 2D. This completes the project.

You now have access to all available bit rates, independently for each serial ports. Make sure only one DIP switch is closed or "ON" for each port. The bit rates will be as follows:

ON Switch Number	Selected Bit Rate
1	9600
2	4800
3	2400
4	1200
5	600
6	300
7	110
8	150 or 75 (jumper selectable at J4)

These bit rates are obtained when the "bit rate factor" in the 8251 USART has been programmed for 16X. You may also diddle with the "bit rate factor" to get other bit rates.

I have found this arrangement to be most convenient, not only for ease of changing bit rates but also to quickly see where the rates have been set. I hope you enjoyed this quick and inexpensive project to add some functionality to your North Star. Remember, it was home-grown on the I/O Farm.

DISK LIB

LIBRARY NEWS

The INSUA Library now offers ~~seven~~ NINE! discs which should be useful to most members and which are available at reasonable cost.

The ASSEMBLER disc includes both the single density and the double density adaptation of the self contained system (assembler/editor) originally released through Processor Technology and now adapted for NORTHSTAR.

The #1001-DOS holds the 5.2 single density relocatable NORTHSTAR DOS and #1002-DOS DD holds the double density version. This makes it possible to move the DOS to other more appropriate locations.

Each of the three COMPASS discs contains those programs printed in one COMPASS newsletter.

TELSTAR is an assembly language program written by Leonard E. Garcia for transferring named disc files, either single or double density, thru the telephone via a modem. ccording to Garcia, an additional program included on the disc, TELSHARE, is designed to run in high memory and allows a remote user to access and share all inputs and outputs of any program running under DOS with the local user. Included on the disc is a program by Pavel Breder, MODEM, which makes it possible through the use of a modem to tie into Bulletin Boards and other informational networks as well as interact with other users.

The #1001 disc, DOS and the #1002 DOS DD are each $10. The others, #1003-TELSTAR, #1004-COMPASS1, #1005-COMPASS2, #1006-ASSEMBLER, #1007-COMPASS 3, #1008-COMPASS 4 #1009-Micro Count II all at $15 each Please indicate by number which you want and send check or money order to INSUA, P.O.Box 1318, Antioch, CA 94509. Allow six weeks delivery.