6800 Software May 1978

Compiled by: Julian E. Jetzer Sheboygan, WI

This month we will present part II of the software offerings submitted by Stephen Heinecke of Allenton, WI.

The program presented is another version of much discussed and presented program "LIFE". However, you 6800 fans will find it unique in that few are presented in assembly language.

LIFE has a variable array size from 3 X 3 to 15 X 15. The height and width don't have to be the same size. While keying in your pattern any character may be used to represent the asterisk but only the space key will do for a space. This allows you to index your patterns. Control X is used for the backspace here elsewhere it is the up arrow. To end a line you may type a carriage return. To end the pattern, type in the escape key (hex 1B). The maximum limit of generations is 999 and the maximum limit skipped is 127. Automatic control lets it keep cycling until a stop is encountered. Manual control will stop after each pattern after it homes up. Some of the messages may not work since they use my own backspace and line up.

This is due to the fact that I have modified my cursor control board so that I can use the backspace and line up. Control X (Hex 18) performs a backspace. Control Y (Hex 19) performs a line up. All other functions remain as listed in the SWTPC manuals.

The listing shows Hex 5F as a backspace. This will be correct for SWTPC terminals.

For more information on "LIFE" and what it is all about I suggest you read over some of the listings in 101 Computer Games and similar publications.

Have fun with "LIFE". Our next offering will be a MULTI-GAME program I'm sure you will have a lot of fun with. It will appear in a subsequent issue of the newsletter.

Hope you all have a nice Summer!!!!!!

Julian

Please send 6800 programs and information to:

Julian E. Jetzer
6400 Hawthorn Road
Sheboygan, WI 53081

This is a sample listing of "LIFE"
Typed in characters are underlined.

```
*L
*G
LIFE

A PATTERN GENERATING GAME

PLEASE, ENTER HEIGHT? 3cr

PLEASE, ENTER WIDTH? 3cr

PLEASE, ENTER PATTERN.
          1 2 cr
          3 esc
THANK YOU
HOW MANY GENERATIONS? 5cr

HOW MANY SKIPPED? 0cr

AUTOMATIC OR MANULA CONTROL? A

          * *
          *

 GENERATION 0 ,POPULATION 3

          * *
          * *

 GENERATION 1, POPULATION 4

          * *
          * *

 GENERATION 2, POPULATION 4
 STABLE PATTERN.
```

When the system was first run noise would occasionally reset the Tarbell Disc Interface. This was traced to the EXTERNAL CLEAR line on the S-100 bus (pin 54). This line is not connected to anything in the SOL-20, and has no pull-up resistor on the Tarbell interface. A pull-up resistor was installed on the Tarbell Disc Interface from pin 10 to U28 to +5 volts which solved the problem.

The only problem found in using the DZ80 in the SOL-20 is that the cassette tape could not be read until a .002 uf capacitor was put on the chip select line of the SOL-20 1K RAM memory (pin 13 of U9). This solution is not understood; however, it works and appears to have no side effects.

In the course of the WAIT STATE battle, the Tarbell Disc READ and WRITE software loops were shortened by replacing the loop exit JP instruction with a RP. This shortens these loops by about 4 u sec..

FAST READ LOOP

```
RLOOP:    CALL  FASTLOOP
RDONE:    IN    DISCSTATS
          ANI   9DH
          RET
FASTLOOP  IN    WAIT
          ORA   A
          RP
          IN    DISCDATA
          MOV   M,A
          INX   H
          JMP   FASTLOOP
```

FAST WRITE LOOP

```
WLOOP:    CALL  FASTWRITE
WDONE:    IN    DISCSTATS
          ANI   OFDH
          RET
FASTWRITE: IN   WAIT
          ORA   A
          RP
          MOV   A,M
          OUT   DISCDATA
          INX   H
          JMP   FASTWRITE
```

The SOL-20 CP/M MACHINE with DZ80 SUPER-CHARGER

By

Tom Nunamaker & Don Stevens

Our goal was to build a CP/M system around the compact and flexible SOL-20 with its five S-100 card slots plus video, and cassette hardware using a Tarbell Floppy Disc Interface. The Tarbell Disc Interface was attractive because of its flexibility in choice of Disc drives, plus it's low cost and the availability of a CP/M software driver. A DZ80 addition allows one to run the growing body of software being generated for the Z80 CPU. Last but not least, we wanted a soft sectored IBM 3740 type system because of the widespread use of this format.

The modified CP/M I/O supplied by Tarbell is designed to use I/O addresses from F8H to FCH which is incompatible with the SOL-20; so these addresses were changed to E8h to ECH. This change also makes the cold start ROM supplied with the Tarbell Disc Interface useless; however, one can program a new ROM. We have been using a cold start loader written from cassette tape, which is not really that inconvenient as the cold start loader is only needed when the system is first powered up.

When the system was first brought up, it would only function with INTEL 8080A chips! This problem was traced to the interaction of the WAIT state circuitry on the SOL-20 and the Tarbell Disc Interface. First off, the SOL-20 adds a WAIT state to all normal system IN and OUT instructions. This is probably done for the sake of the slow UARTS used on the SOL-20. In any case, these extra WAIT states cut the timing margins of the Disc READ and WRITE software loops to bare bone. In addition to the above WAIT state situation, there is a potential Hang-Up condition associated with the WAIT state gating on the Tarbell Disc Interface when used with some manufacturers 8080 CPU chips. What happens is that when executing a STATUS IN from the Tarbell Disc Interface, circuitry on the Tarbell Disc Interface connects PWAIT (S-100 pin 27) to PRDY (S-100 pin 72), the consequence of this is that if a WAIT state is ever entered the system will Hang-Up for ever!! This signal path was broken by cutting the trace at pin 13 on U57 on the Tarbell Disc Interface board. After breaking this loop, all manufacturers 8080 chips we tried worked including the DZ80 CPU.

In an effort to kill the IN and OUT additional WAIT states in the SOL-20, the lines on pin 5 and 6 of U53 were disconnected. However with normal IN and OUT timing, tape I/O became marginal because of the slow UART. A compromise was reached where by the SOL-20 was modified to only add the WAIT state on SOL-20 IN instructions. This may be accomplished by connecting a wire from pin 6 of U83 to pins 5 and 6 of U53 on the SOL-20 CPU board.

```
Locn   B1 B2 B3

0280   DF 2A       STRING  STX D     SAVE      String imput processer
0282   7F 04 FF    STR2    CLR E     04ff      Clear the byte before the string
0285   8D 4E               BSR       STLOCN    Go get the string location
0287   5C                  INCB                add one to counter
0288   BD E1 AC    STINP   JSR E     INEEE     Go get a character
028B   81 0D               CMPA I    ASCIIcr   If it's a carriage return
028D   27 2E               BEQ       STEND     then end the string
028F   81 5F               CMPA I    ASCII _   If it's not the backspace flag
0291   26 13               BNE       STSTR     then go check if it should be stored
0293   8D 0A               BSR       STBCK2    Go print a backspace and clear this byte
0295   C1 04               CMPB I    04        If this is the beginning of the string
0297   27 EF               BEQ       STINP     don't print any more backspaces
0299   8D 02               BSR       STBACK    If not print another backspace
029B   20 EB               BRA       STINP
029D   09          STBACK  DEX                 Adjust pointer and counter
029E   5C                  INCB
029F   6F 00        STBCK2  CLR X     X+00      Clear byte pointed to
02A1   86 18               LDAA I    18        P int a backspace
02A3   7E E1 D1     STOUT   JMP E     OUTEEE
02A6   80 30               SUBA I              If character is not a number
02A8   25 DE               BCS       STINP     then go get another chatacter
02AA   81 0A               CMPA      0A
02AC   24 DA               BCC       STINP
02AE   A7 00               STAA X    X+00      otherwise store it after masking of
02B0   08                  INX                 top four bytes
02B1   5A                  DECB                Increment pointer and decrement counter
02B2   26 D4               BNE       STINP     If counter ≠ 0 go get another byte
02B4   BD 00 22     ERROR   JSR E     CRLF2     Otherwise ERROR by to many bytes
02B7   86 3F               LDAA I    ASCII ?   P int "crlf?"
02B9   8D E8               BSR       STOUT
02BB   20 C5               BRA       STR2      and get another string
02BD   5A           STEND   DECB                If the bytes were entered
02BE   27 15               BEQ       STLOCN    then get location and return
02C0   5A                  DECB                If two bytes were entered
02C1   27 05               BEQ       SHIFT     Then shift up once
02C3   5A                  DECB                If no bytes were entered
02C4   26 EE               BNE       ERROR     Then go do error routine
02C6   8D 00               BSR       SHIFT     Other shift up twice
02C8   CE 05 02     SHIFT   LDX I     0502      When there are less than 3 bytes
02CB   C6 04               LDAB I    04        entered then then bytes must be moved
02CD   A6 00        SHFT2   LDAA X    X+00      so that the ones are in the ones place
02CF   A7 01               STAA X    X+01      and the tens are in the tens place.
02D1   09                  DEX
02D2   5A                  DECB
02D3   26 F8               BNE       SHFT2
02D5   CE 05 00     STLOCN  LDX I     0500      Here is location of string and
02D8   C6 03               LDAB I    03        the count of bytes
02DA   39                  RTS
```

```
Locn   B1 B2 B3

01FE   34          START   DES             Relocate the stack
01FF   34                  DES
0200   CE 05 00    NWLIFE  LDX I   0500    Clera out the areas for the field,
0203   8D BB              BSR     CLRARY    population count, generation count
0205   DE F0              LDX D   FIELD     and their respective compliments
0207   8D B7              BSR     CLRARY
0209   CE 03 00           LDX I   0300      Print "homeeofLIFEcrlflfHEIGHT ? "
020C   8D E0              BSR     PRTIII    get the height and
020E   8D D0              BSR     SIZE      get the height and print "WIDTH ? "
0210   D7 35              STAB D  LINE1     Store the height (# of lines)
0212   8D CC              BSR     SIZE      Get the width and print"homeeofPATTERNcr
0214   D7 44              STAB D  CELS      Store the width (# of cells)
0216   86 10              LDAA I  10        Compute the tab for center
0218   10                 SBA
0219   97 3F              STAA D  TAB       and store it
021B   4A                 DECA    BORDER    Compute the border adjustment
021C   97 5C              STAA D  BORDER    and store it
021E   86 80              LDAA I  80        Set the mode to accetp
0220   97 46              STAA D  MODE      the inputed pattern
0222   B7 05 60           STAA E  Locn 0560 Prevent final display
0225   BD 00 2F           JSR E   LOOP      Go get the pattern
0228   86 30              LDAA I  ASCII 0   Set the generations to zero
022A   B7 04 62           STAA E  Locn 0462
022D   7F 00 46    MORGEN  CLR E   MODE      Set the mode to print the pattern
0230   CE 03 80           LDX I   0380      Print "home eof GENERATION LIMIT? "
0233   8D B9              BSR     PRTIII
0235   8D 49              BSR     STRIN     Get the generation limit
0237   6F 60       GL1    CLR X   X+60      Clear the final display prevention
0239   A6 00       GL2    LDAA X  X+00      Set up the limit so that it will stop the
023B   26 0A              BNE     GL4       loop when it has been reached.
023D   08                 INX               All leading zeroz must be cleared
023E   5A                 DECB
023F   26 F8              BNE     GL2
0241   8D 71              BSR     ERROR     The number zero is not allowed
0243   20 F2              BRA     GL1
0245   A6 00       GL3    LDAA X  X+00      Limit must be positioned 256 bytes
0247   8B 30       GL4    ADDA I  ASCII 0   above the generation count
0249   A7 60              STAA X  X+60      And digits must be in proper ASCII
024B   08                 INX               number format
024C   5A                 DECB
024D   26 F6               BNE     GL3
024F   8D 9A              BSR     CRPRT     Print "crlfENTER SKIP FACTOR ? "
0251   8D 1D              BSR     ENRY2     Get the skip factor
0253   97 C5              STAA D  SKIP      and store it
0255   8D 94              BSR     CRPRT     Print "crlfAUTO/MANUAL CONTROL ? "
0257   BD E1 AC           JSR E   INEEE     Get the answer
025A   80 41              SUBA I  ASCII A   A = automatic control
025C   97 BA              STAA D  WAIT
025E   08                 INX               Print "homeeof"
025F   8D 8D              BSR     PRTIII
0261   BD 00 2F           JSR E   LOOP      Print out pattern(s)
0264   20 9A              BRA     NWLIFE
```

```
Locn   B1 B2 B3

Left over

01E0   BD 02 E0   SIZE     JSR E     BINARY    Go get a number and convetr it to binary
01E2   81 03      SZ2      CMPA I    03        3 is the smallest size a lowed
01E5   25 0A               BCS       WRNGSZ    if smaller go get another size
01E7   81 10               CMPA I    10        15 is the largest size allowed
01E9   24 06               BCC       WRNGSZ    if larger go get another size
01EB   BD 00 22            JSR E     CRLF2     Go print "crlf"
01EE   7E E0 7E            JSR E     PDATA1    And then print next message
01F1   BD 02 E9   WRNGSZ   JSR E     BNERR     Get another size and then
01F4   20 ED               BRA       SZ2       check it again

0270   20 6E      BNRY2    BRA       BINARY    Get a number and convert it to binary

02E0   8D 9E      BINARY   BSR       STRING    Get a number
02E2   A6 00      BN2      LDAA X    X+00      Get the hundreds byte
02E4   27 09               BEQ       BN3       If no hundreds go to tens and ones
02E6   4A                  DECA
02E7   27 04               BEQ       BNHND     If one hundeed add it
02E9   8D C9      BNERR    BSR       ERROR     Otherwise it an error
02EB   20 F5               BRA       BN2
02ED   86 64      BNHND    LDAA I    64        Add one hundred
02EF   AB 02      BN3      ADDA X    X+02      Add the ones
02F1   E6 01               LDAB X    X+01      Get the tens
02F3   27 05               BEQ       BNEND     If no tens then return
02F5   8B 0A      BNTENS   ADDA I    0A        Otherwise add ten
02F7   5A                  DECB                and decrement count of tens
02F8   26 FB               BNE       BNTENS    untill all tens have been added
02FA   4D         BNEND    TSTA                Check to make sure that we
02FB   2B EC               BMI       BNERR     havent entered a negative number
02FD   39                  RTS
```

```
Locn   B1 B2 B3

018F   31          ENDPAT  INS          TO end the pattern input cycle
0190   31                  INS          you simply increment the stack twice
0191   39                  RTS          and return to main control

0192   08          NL2     INX          To end a line the pointer and cell
0193   5A          NEWLIN  DECB         counter must be set to the values
0194   26 FC               BNE     NL2  they'd be if it ended normally
0196   5C                  INCB
0197   39                  RTS

0198   09          NOBACK  DEX          you cannot back beyond the beginning
0199   5C                  INCB         of a line so pointer and counter must
019A   20 0E               BRA     SPACE bet set and a space is printed

01A0   8D 08       SPCR    BSR     SPACE Print a set of spaces as counted
01A2   5A                  DECB         by B to tab over and center pattern
01A3   26 FB               BNE     SPCR
01A5   39                  RTS

01A8   8D 00       TWOSPC  BSR     SPACE Print two spaces
01AA   7E E0 CC    SPACE   JSREE   OUTS  Print one space

01B0   D1 44       BACK    CMPB D  CELS  If count equal limit then you're at
01B2   27 E4               BEQ     NOBACK the beginning and cant back further
01B4   09                  DEX          Otherwise do a backspace adjusting
01B5   6F 00               CLR X   X+00  the pointer and counter and
01B7   09                  DEX          clearing the present memory cell.
01B8   5C                  INCB
01B9   5C                  INCB
01BA   86 18               LDAA I  18    Print another backspace just to keep
01BC   7E E1 D1            JMP E   OUTEE thing lined up

01C0   C6 00       CLRARY  LDAA I  00    Clear the arrays
01C2   8D 0C               BSR     CLEAR
01C4   DE F2               LDX D   GEN   Clear the generation counter
01C6   8D 02               BSR     CLR3
01C8   DE F4       CLRPOP  LDX D   POP   Clear the population counter
01CA   C6 03       CLR3    LDAB I  03
01CC   01                  NOP           I thought it would be nice to let
01CD   01                  NOP           the processer rest for a while.
01CE   01                  NOP
01CF   01                  NOP
01D0   6F 00       CLEAR   CLR X   X+00  Clear an area as defined by the
01D2   08                  INX           index and B
01D3   5A                  DECB
01D4   26 FA               BNE     CLEAR
01D6   39                  RTS
```

```
Locn   B1 B2 B3

012C   DF F8      INCPOP  STX D      SAVEBB    Increment population.
012E   DE F4              LDX D      POP
0130   A6 02      INC2    LDAA X     X+02      Get a digit
0132   81 39              CMPA I     ASCII 9   if it is less than 9
0134   27 08              BEQ        INC3
0136   8A 30              ORAA I     ASCII 0
0138   4C                 INCA                 increment  it and
0139   A7 02              STAA X     X+02      store it back where you found it
013B   DE F8              LDX D      SAVEBB
013D   39                 RTS                  and return.
013E   86 30      INC3    LDAA I     ASCII 0   Otherwise set that digit to zer0
0140   A7 02              STAA X     X+02      and store that where it came from
0142   09                 DEX                  and set pointer to nest digit up
0143   20 EB              BRA        INC2      and increment that digit

0150   2B 24      MDCTRL  BMI        INPAT     If mode if negative goto pattern input
0152   26 0B              BNE        PRCSS     if ≠ 0 skip the printing section
0154   8D 09              BSR        PRCSS     .o process the cell
0156   27 50              BEQ        TWOSPC    if dead print two spaces
0158   86 2A              LDAA I     ASCII *   other wise print an "*"
015A   BD E1 D1           JSR E      OUTEEE
015D   20 4B              BRA        SPACE     and a space
015F   A6 00      PRCSS   LDAA X     X+00      Get this cell
0161   27 0B              BEQ        RTSCTL    If allready dead then leave (quietly)
0163   81 03              CMPA I     03        If this is a birth
0165   27 08              BEQ        MARKR
0167   44                 LSRA                 or a surviver
0168   81 41              CMPA I     41
016A   27 03              BEQ        MARKR     then set the marker
016C   6F 00              CLR X      X+00      Otherwise kill off any residuals
016E   39          RTSCTL  RTS
016F   86 80      MARKR   LDAA I     80        Get the marker
0171   A7 00      STMRK   STAA X     X+00      Store the marker
0173   7E 01 2C           JMP E      INCPOP    count one living cell
0176   BD E1 AC   INPAR   JSR E      INEEE     Gen a character
0179   81 20              CMPA I     ASCIIspc  If it's a space
017B   27 2D              BEQ        SPACE     go print another space
017D   81 18              CMPA I     18        If it's my backspace
017F   27 2F              BEQ        BACK      go do backspace routine
0181   81 0D              CMPA I     ASCIIcr   If it's a carriage return
0183   27 0E              BEQ        NEWLIN    go start a new line
0185   81 1B              CMPA I     ASCIIesc  If it's an escape character
0187   27 06              BEQ        ENDPAT    go finish the pattern
0189   86 03              LDAA I     03        Otherwise it's a baby (see line 0163)
018B   8D E4              BSR        STMRK     Lets make it comfortable
018F   20 1B              BRA        SPACE     and leave some space for it
```

```
Locn   B1 B2 B3

00D0   96 46       FNLDSP   LDAA D    MODE      Final display
00D2   26 02                BNE       FD2       Test skip mode
00D4   8D 14                BSR       PRG+P     Print " gen + pop
00D6   CE 04 B0    FD2      LDX I     04B0      Print "FINAL DISPLAY"
00D9   8D 12                BSR       PRTAAA
00DB   8D D0                BSR       INAAAA    Get an input
00DD   81 1B                CMPA I    ASCIIesc  is input an escape
00DF   27 07                BEQ       FD3       if so return to main control
00E1   8D 1D                BSR       NWCYCL    if not recycle the field
00E3   31                   INS                 and find out how long to go on
00E4   31                   INS
00E5   7E 02 2D             JMP E
00E8   39          FD3      RTS

00EA   CE 04 54    PRG+P    LDX I     0454      Print " GENERATION ???,POPULATION ???
00ED   7E E0 7E    PRTAAA   JMP E     PDATA1          homehome"

00F0   06 00       FIELD
00F2   04 60       GEN
00F4   04 70       POP

0100   DE F0       NWCYCL   LDX D     FIELD     Move this pattern 256 bytes
0102   5F                  CLRB                up for the next old pattern
0103   A6 00       MOVE     LDAA X    X+00
0105   08                  INX
0106   A7 FF               STAA X    X+FF      NOTE: I used the same trick here to
0108   5C                  INCB                      move it up 256 bytes
0109   C1 F0               CMPA I    F0
010B   26 F6               BNE       MOVE
010D   CE 05 EF             LDX I     05EF      Set up the next generation field
0110   A6 11       NEWGEN   LDAA X    X+11      Get a cell
0112   2A 10                BPL       NG2       if that cell is negative
0114   6C 00                INC X     X+00      then increment all the cells
0116   6C 01                INC X     X+01      around it
0118   6C 02                INC X     X+02
011A   6C 10                INC X     X+10
011C   6C 12                INC X     X+12
011E   6C 20                INC X     X+20
0120   6C 21                INC X     X+21
0122   6C 22                INC X     X+22
0124   08          NG2      INX                 increment pointer
0125   5A                   DECB                decrement count
0126   26 E8                BNE       NEWGEN    if ≠ 0 do it agin
0128   DE F2                LDX D     GEN       Increment the generation
012A   20 04                BRA       INC2
```

```
Loca   B1 B2 B3

0073   DE F0          LDX D    FIELD   Compare this pattern with the last one
0075   C6 F0          LDAB I   F0
0077   8D 12          BSR      CMP2
0079   27 1D          BEQ      STBL    If they're equal then do stable routine
007B   DE F4          LDX D    POP     Test population count
007D   8D 0A          BSR      CMP1    if
007F   27 23          BEQ      ZROPOP  if zero go report no more life
0081   DE F2          LDX D            Test generation count
0083   8D 04          BSR      CMP1
0085   27 49          BEQ      FNLDSP  if equal limit go report final display
0087   20 27          BRA      NEXT    otherwise end this loop and prepare next
0089   C6 03    CMP1  LDAB I   03      Compare three bytes
008B   A6 00    CMP2  LDAA X   X+00    Compare loop
008D   08            INX              note; the bytes compared are 256 bytes
008E   A1 FF          CMPA X   X+FF           apart
0090   26 03          BNE      CMPEND  if inequality detected return
0092   5A            DECB
0093   26 F6          BNE      CMP2
0095   39      CMPEND RTS

0098   96 46    STBL  LDAA D   MODE    Was this a print or skip loop
009A   26 2C          BNE      FRCPRT  if it was skipped go print out pattern
009C   8D 4C          BSR      PRG+P   Else print "GENERATION ???,POPULATION ???
009E   08            INX              home       homeeofSTABLE PATTERN"
00A0   8D 4C          BSR      PRTAAA
00A2   20 0A          BRA      INAAAA  Go wait for any input then return

00A4   86 30    zropop LDAA I  ASCII 0 Store a 0 after the population.
00A6   CE 04 40        LDX I    0440    Print "homeeof NO MORE LIFE crlf
00A9   A7 32           STAA X   X+32          GENERATION  ???,POPULATION 0"
00AB   8D 40           BSR      PRTAAA        home home"
00AD   7E E1 AC INAAAA JSE E    INEEE   Wait for any input then return

00B0   D6 46    NEXT  LDAB D   MODE    Is this the input mode
00B2   2A 01          BPL      NXT2    if so then return to
00B4   39             RTS
00B5   26 0C    NXT2  BNE      NXT4    If this is a skip then skip print sectior
00B7   8D 31          BSR      PRG+P   Go print " GENERATION ?,POPULATION?homeo
00B9   86 XX          LDAA I   ??      If this wait flag is cleared then
00BB   27 02          BEQ      NXT3    do not wait for an input,
00BD   8D EE          BSR      INAAAA  if not wait for any input
00BF   D1 C5    NXT3  CMPB D   SKIP    Test for skip on not skip
00C1   27 08          BEQ      NXT6    if not skip dont change mode
00C3   5C       NXT4  INCB             increment mode
00C4   C1 XX          CMPB I   00-7F   is this the last skip
00C6   26 01          BNE      NXT5    if not store mode as is
00C8   5F       FRCPRT CLRB            Force the loop to print by clearing mode
00C9   D7 46    NXT5  STAB     MODE    store mode
00CB   8D 33    NXT6  BSR      NWCYCL  Set up for new cycle
00CD   7E 00 2F       JMP E    LOOP    Go to the next loop
```

Locn	B1 B2 B3	Label	Operator	Operand	Comments
0020	DF 2A	CRLF1	STX D	SAVE	Save index register
0022	16	CRLF2	TAB		Save A in B
0023	CE 04 E0		LDX I	04E0	Print "crlf"
0026	BD E0 7E		JSR E	PDATA1	
0029	CE 00 00		LDX I	XXXX	Restore index register
002C	08		INX		and add one to it
002D	39		RTS		
002F	BD 01 C8	LOOP	JSR E	CLRPOP	Clear population count
0032	DE F0		LDX D		
0034	C6 XX		LDAB I	03-0F	Line count 3-15
0036	D7 67	LP2	STAB D	LINE2	
0038	D6 46		LDAB D	MODE	If this is a skip loop dont print
003A	27 02		BEQ	LPTAB	spaces to center display
003C	2A 05		BPL	NOTAB	
003E	C6 XX	LPTAB	LDAB I	01-0D	Tab count 1-13
0040	BD 01 A0		JSR E	SPCR	Center pattern
0043	C6 XX	NOTAB	LDAB I	03-0F	Space count 3-15
0045	86 XX	LP3	LDAA I	00-80	Mode control 0=print,80=input,1-7F=skip
0047	BD 01 50		JSR E	MDCTRL	
004A	08		INX		
004B	5A		DECB		
004C	26 F7		BNE	LP3	
004E	96 46		LDAA D	MODE	If this is a skip loop dont print
0050	2B 02		BMI	LPCR	a carriage return and line feed
0052	26 03		BNE	NOCR	
0054	8D CA	LPCR	BSR	CRLF1	Go print "crlf"
0056	09		DEX		
0057	DF 2A	NOCR	STX D	SAVE	Store index
0059	6F 00		CLR X	X+00	Clear right border space
005B	86 XX		LDAA I	00-0C	Compute left border
005D	9B 2B		ADDA D	SAVE2	
005F	97 2B		STAA D	SAVE2	
0061	DE 2A		LDAAD	SAVE	Get left border location
0063	6F 00		CLR X	X+00	CLEAR LEFT border space
0065	08		INX		
0066	C6 XX		LDAB I	01-0F	Get line count
0068	5A		DECB		Decrement it
0069	26 CB		BNE	LP2	if ≠ 0 do another line
006B	6F 00	LPCLR	CLR X	X+00	Clear bottom border
006D	08		INX		
006E	8C 06 F8		CPX I	06F8	
0071	26 F8		BNE	LPCLR	

FOR SALE

Special price on "INNOTRONICS" Full Sized Floppy Disc Unit when purchased with TARBELL Floppy Disc Interface Board. If interested, contact Don Stevens

****

Contact DON SENZIG if you are interested in demonstrating your system at the Science Fiction Club Show which will be held the first part of June. Don's phone is 263-7792.

****

SPECIAL PRICE to Clubmembers on DYSAN Soft Sectored Floppy Discs - Both full sized and mini-floppy discs available. If interested, contact Don Stevens

****

In forthcoming issues, I hope to publish a column which will feature systems of clubmembers. If you are interested in your system being featured in this column, please contact me for particulars.

****

If you have anything to SELL or TRADE and wish to have it published (No Charge), please contact the Editor.

JOHN EVERS
6372 N. 89TH STREET
MILWAUKEE, WISC. 53225

****
* LETTER MAIL *
* FIRST CLASS *
****

WISCONSIN COMPUTER SOCIETY
P.O. Box 159
Sheboygan Falls, WI 53085

WISCONSIN COMPUTER SOCIETY

NEWSLETTER

****

VOLUME 3, Issue #5 May 1978 Don Stevens, Editor

****

MEETING NOTICE

The monthly meeting of the Wisconsin Computer Society will be held this coming Saturday, MAY 6, 1978. It will be held at the WAUKESHA TECHNICAL INSTITUTE. The room location has been changed - SCIENCE BLDG., Rooms S206 & S207.

PROGRAM AGENDA

CP/M DISK OPERATING SYSTEM will be discussed. Members are asked to bring along to the meeting any items that would be of interest to other club members.

NEWSLETTER

The Newsletter Editor is still looking for articlers of interest to publish. Surely with all of that vast talent out there, someone has something to submit!!