ANTIC FORTH#

'Screens of Disk Two'

```
SCR # F
  0 (  ERROR MESSAGES  )   135  159
  1  9  8  7  10  ;S
  2 empty stack
  3
  4 dictionary full
  5
  6 has incorrect address mode
  7
  8 isn't unique
  9
  A
  B
  C disc range ??
  D
  E full stack !
  F
 10 disc error !
 11
 12
 13
 14
 15
 16
 17
 18
 19
 1A
 1B
 1C THIS IS IT
 1D
 1E HELP ME!
 1F

SCR # 10
  0 ( USER INDEX SCR ... ....... ;S
  1  PHYSICAL OFFSET: SCREEN # 1A
  2 SCR #|       Contents
  3 00/00| INDEX
  4 01/01| COVER SCREEN
  5 02/06| ERROR MESSAGES
  6 07/0A| BOOT MAKER - call SYS
  7      |
  8      |
  9 0B/0B| disk RPM checker
  A 0C/0E| FREE
  B 0F/0F| low level DECOMP info
  C  DO NOT MOVE THESE SCREENS !!!
  D 17/18| REV.G COMMENTS
  E 10/23| TUTORIAL
  F 24/2A| COMMAND SUMMARY/REV NOTES
 10 30/32| DUMB TERMINAL V.1.0
 11 33/37| Formatted LIST program
 12 38/3F| FREE
 13      |
```

```
 14      |
 15      |
 16      |
 17      |
 18      |
 19      |
 1A      |
 1B TO CONTINUE TO THE DOCUMENTATION
 1C PART OF THE DISK TYPE
 1D
 1E   10 UE     and press <RETURN>
 1F

SCR # 11
  0 ******  fig-FORTH  MODEL  ******
  1
  2
  3      Through the courtesy of
  4
  5
  6       FORTH INTEREST GROUP
  7          P. O. BOX 1105
  8       SAN CARLOS, CA. 94070
  9
  A        Implemented on the
  B          ATARI  800/400
  C                by
  D           Steve Calfee
  E              1/26/81
  F              4/01/82
 10    PETER LIPSON/ROBIN ZIEGLER
 11              4/10/82
 12           HARALD  STRIEPE
 13          5/5/82 - 10/16/82
 14  XL Mods - John Stanley 18Jun85
 15      RELEASE 1.4S     REV.H
 16      WITH COMPILER SECURITY
 17      VARIABLE  LENGTH NAMES
 18  SWITCHABLE TOP OF STACK DISPLAY
 19      DECOMPILER/DISSASSEMBLER
 1A   ENHANCED SCREEN EDITOR & FAST
 1B  EDIT WORDS, BASE BORDER DISPLAY
 1C  ENHANCED SYSTEM SET UP/BOOTMKR
 1D        DRIVE 2 LINK/UNLINK
 1E      Further distribution must
 1F      include the above notice.

SCR # 12
  0 BREAK Abort.
  1
  2 IOCB already open.
  3
  4 Non-existant device.
  5
  6 IOCB is write-only.
  7
  8 Invalid command (for this device
  9 )
  A Device or file not open.
```

```
  B
  C Bad IOCB #
  D
  E IOCB is read-only
  F
 10 End Of File
 11
 12 Truncated Record
 13
 14 Device Timeout
 15
 16 Device NAK (Negative AcKnowledge
 17 )
 18 Serial Bus input framing error
 19
 1A Cursor out of range
 1B
 1C Serial Bus data-frame overrun
 1D
 1E Serial Bus data-frame checksum e
 1F rror.

SCR # 13
  0 Device-done error
  1
  2 Read-after-write compare error
  3
  4 Function not implemented in hand
  5 ler
  6 Insufficient RAM
  7
  8
  9
  A
  B
  C
  D
  E
  F
 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 1A
 1B
 1C
 1D
 1E
 1F

SCR # 14
  0 (  ERROR MESSAGES  )   135  159
  1  9  8  7  10  ;S
```

```
  2 empty stack
  3
  4 dictionary full
  5
  6 has incorrect address mode
  7
  8 isn't unique
  9
  A
  B
  C disc range ??
  D
  E full stack !
  F
 10 disc error !
 11
 12
 13
 14
 15
 16
 17
 18
 19
 1A
 1B
 1C THIS IS IT
 1D
 1E HELP ME!
 1F

SCR # 15
  0 (  ERROR MESSAGES  )
  1
  2 compilation only, use in definit
  3 ion
  4 execution only
  5
  6 conditionals not paired
  7
  8 definition not finished
  9
  A in protected dictionary
  B
  C use only when loading
  D
  E off current editing screen
  F
 10 declare vocabulary
 11
 12 outside allocated file space
 13
 14 writing off current line
 15
 16
 17
 18
 19
 1A string stack empty !!
```

```
 1B
 1C
 1D
 1E
 1F

SCR # 16
  0 (  TARGET COMPILER ERROR MESSAGE
  1 S                 WFR-79JUN02 )
  2
  3
  4 below lower bound of virtual mem
  5 ory
  6 disc compiler assembly error in
  7 mode of
  8 can't find in TARGET
  9
  A target redef.
  B
  C T: error, is it paired with T;
  D ?
  E above virtual memory bounds
  F
 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 1A
 1B
 1C
 1D
 1E
 1F

SCR # 17
  0 ( SYS/BOOTMAKER        1/4 )
  1    FORTH DEFINITIONS   HEX
  2    SAVENFAs
  3  HERE 1C +ORIGIN !  ( FENCE )
  4
  5  HERE 1E +ORIGIN !  ( DP )
  6
  7  HERE DUP FENCE  ! 0 +ORIGIN -
  8
  9  80 / 1+  CONSTANT  #SECT
  A
  B   CODE CALLDK XSAVE STX, E453
  C    JSR, TYA, PHA, ( STATUS )
  D   XSAVE LDX, PUSH JMP, C;
  E
  F
 10 : DKIO 301 ! ( CMD, DRIVE # )
 11   30A ! ( SECT. # ) 304 !
```

```
 12   ( RAM BUFFER ) CALLDK ( DKHND)
 13   DUP 0< IF ." ERROR " 0FF AND
 14   BASE @ SWAP DECIMAL
 15   . BASE ! QUIT ENDIF DROP ;
 16 : WTSEC SWAP 304 ! 130 300 !
 17 ( verif $57->) 50 302 C! SECIO ;
 18  : RDSEC SWAP 304 ! 130 300 !
 19          52 302 C! SECIO   ;
 1A : FORMAT ." FORMAT DRIVE " DUP .
 1B   ." -ARE YOU SURE?" 0 PAD ! PAD
 1C  1      EXPECT PAD C@ 59 ( Y) =
 1D   IF 2100 OR PAD 0 ROT DKIO ELSE
 1E         DROP THEN ;
 1F  0 VARIABLE BOOT ( ->CODE ) -->

SCR # 18
  0 ( SYS SET UP/BOOTMKR   2/4 )
  1 : MAKEBOOT FLUSH  EMPTY-BUFFERS
  2  ." INSERT NEW DISK, TYPE Y" CR
  3 0 PAD ! ( DEFAULT ) PAD 3 EXPECT
  4  PAD C@ 59 = IF 1  52  C! CR
  5  ." Writing sectors:" CR CR BOOT
  6  @   1 DUP .  WTSEC #SECT 0 DO I
  7  80 * +ORIGIN I 2 + WTSEC I 2 +
  8  . LOOP 0 52 C! CR ." BOOT COMPL
  9 ETED" CR THEN ; ( BOOT CODE:)
  A  HERE BOOT ! ( PT TO US )
  B  ASSEMBLER 1FF , 480 , ' V1.4S ,
  C  #SECT # LDA, 0= IF, 0 +ORIGIN ,
  D  1 , ENDIF, N STA,    2C8 C@
  E  # LDA, 2C8 STA,  D01A STA,
  F     2C6  C@   # LDA, 2C6 STA,
 10   D018 STA,
 11  52 # LDA, 302 STA, 48C LDA, 30A
 12  STA, 48D LDA, 30B STA, ( SCT1 )
 13  1 # LDA, 301 STA, 48A LDA, 304
 14 STA, 48B LDA, 305 STA, ( ORIGIN)
 15  BEGIN, 30A INC, 0= IF, 30B INC,
 16        ENDIF, E453 JSR, 303 LDA,
 17  .A ASL, CS IF, RTS, ( FRETURN )
 18        ENDIF, 304 LDA, 80 # EOR,
 19  304 STA, 0< NOT IF, 305 INC,
 1A  ENDIF, ( BUMP PTR.) N DEC, 0=
 1B UNTIL, 48A LDA, 0A STA, 48B LDA,
 1C  0B STA,  E C@ # LDA, 2E7 STA,
 1D  F C@ # LDA, 2E8 STA, CLC, RTS,
 1E    FORTH
 1F   -->

SCR # 19
  0 ( BACKUP HES 82AUG15   3/4 ) (
  1 35F ARRAY BUCD BLK @ BLOCK A0 +
  2 BUCD 35F CMOVE CODE bg E474 JMP,
  3  C; : BACKUP  BUCD 480 35F CMOVE
  4  480 C ! bg  ;         )   -->
 11 fig-FORTH 1.4S FAST
 12  BACKUP Vers.1.2   BY H.E.STRIEP
 13 E 1982
 14   START   - commence I/O
```

```
 15   SELECT  - write with verify
 16   OPTION  - REBOOT
 17 Insert source disk and press STA
 18 RT,   or select OPTION to REBOOT
 19
 1A Reading SOURCE disk...
 1B Insert destination disk, press S
 1C TART, or SELECT
 1D Writing DESTINATION disk...
 1E }****** DUPLICATION SUCCESSFUL *
 1F *****

SCR # 1A
  0 ( SYS SET UP/BOOTMKR   4/4 )
  1  : DoFORget  ( Forgets below   )
  2      ' TEXT NFA FENCE ! ( fence)
  3      0 FORGET TEXT  ;
  4
  5    HEX
  6    LMARGN @ 2700 LMARGN !
  7 ." }fig-FORTH 1.4S SYSTEM SET-
  8 UP Vers.1.1 " CR CR
  9 ." DoFORget WORD  forgets below
  A FENCE." CR CR
  B ." n SETPHYS      permanently ch
  C anges the                 OFFSET
  D  of screen #0." CR CR
  E ." RESPHYS        resets the OFF
  F SET to its                origin
 10 al value; use" CR
 11 ."                n SETPHYS twic
 12 e to set                 RESPHY
 13 S to a new value" CR CR
 14 ." n FORMAT       formats disk i
 15 n drive n. " CR CR
 16
 17 ." MAKEBOOT       writes out com
 18 piled boot
 19        sectors." CR
 1A ." SETSYS         sets booton pa
 1B rameters:" CR ."
 1C screen margins, colors" CR CR (
 1D ." BACKUP         fast single dr
 1E ive utility " CR  )
 1F  LMARGN ! EMPTY-BUFFERS SP!  ;S

SCR # 1B
  0 ( DISK RPM CHECKER     1/1 )
  1 CODE ZIO XSAVE STX, BOT LDA,
  2          E459 JSR, XSAVE LDX,
  3          BOT STY, BOT 1+ STA,
  4      NEXT JMP, C;
  5 246 CONSTANT DSKTIM
  6 : DSIO ( DISK HNDLR VIA SIO )
  7 ( BADDR AUXS UNIT-CMD DATFLG )
  8    303 C! ( SET DATA-FLAG )
  9    301 ! ( DUNIT,CMD) 31 300 C!
  A    ( DEVICE) 30A ! ( AUXES )
  B    304 ! ( BUFER-ADDR ) 0 ZIO ;
```

```
  C 1 VARIABLE DR#
  D  : BIO DSKTIM @ 306 C! 80 308 !
  E          ( BUFLEN) DSIO ;
  F  : RDSEC 5200 OR 40 BIO ;
 10 ( DISK SPEED CHECKER )
 11
 12   14 CONSTANT RTCLOCK
 13  2F2 CONSTANT KBCHAR
 14   FF CONSTANT EPTY
 15 : READ-SEC PAD 1 DR# C@ RDSEC ;
 16 : CLR-KBRD   EPTY KBCHAR C! ;
 17 : RPM DECIMAL CLR-KBRD 1 2F0 C!
 18   DR# C! 0 17 POS. ." RPM=    "
 19   BEGIN KBCHAR C@  EPTY = WHILE
 1A   READ-SEC DROP 0 RTCLOCK C!
 1B  10 1 DO READ-SEC DROP LOOP D2F0
 1C   0 RTCLOCK C@ U/ SWAP DROP
 1D   4 17 POS. . REPEAT CLR-KBRD
 1E   0 2F0 C! HEX ;  ;S
 1F ( NEEDS DRIVE # ON THE STACK )

SCR # 1C
  0 ( DISK RPM CHECKER     1/1 )
  1 CODE ZIO XSAVE STX, BOT LDA,
  2          E459 JSR, XSAVE LDX,
  3          BOT STY, BOT 1+ STA,
  4      NEXT JMP, C;
  5 246 CONSTANT DSKTIM
  6 : DSIO ( DISK HNDLR VIA SIO )
  7 ( BADDR AUXS UNIT-CMD DATFLG )
  8    303 C! ( SET DATA-FLAG )
  9    301 ! ( DUNIT,CMD) 31 300 C!
  A    ( DEVICE) 30A ! ( AUXES )
  B    304 ! ( BUFER-ADDR ) 0 ZIO ;
  C 1 VARIABLE DR#
  D  : BIO DSKTIM @ 306 C! 80 308 !
  E          ( BUFLEN) DSIO ;
  F  : RDSEC 5200 OR 40 BIO ;
 10 ( DISK SPEED CHECKER )
 11
 12   14 CONSTANT RTCLOCK
 13  2F2 CONSTANT KBCHAR
 14   FF CONSTANT EPTY
 15 : READ-SEC PAD 1 DR# C@ RDSEC ;
 17 : RPM DECIMAL CLR-KBRD 1 2F0 C!
 18   DR# C! 0 17 POS. ." RPM=    "
 19   BEGIN KBCHAR C@  EPTY = WHILE
 1A   READ-SEC DROP 0 RTCLOCK C!
 1B  10 1 DO READ-SEC DROP LOOP D2F0
 1C   0 RTCLOCK C@ U/ SWAP DROP
 1D   4 17 POS. . REPEAT CLR-KBRD
 1E   0 2F0 C! HEX ;  ;S
 1F ( NEEDS DRIVE # ON THE STACK )

SCR # 1D
  0 ( EMPTY BLOCK D  ) ;S
  1
  2
  3
```

```
  4
  5
  6
  7
  8
  9
  A
  B
  C
  D
  E
  F
 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 1A
 1B
 1C
 1D
 1E
 1F

SCR # 1E
  0 \ scr# E   empty block       1/1
  1  ;S
  2
  3
  4
  5
  6
  7
  8
  9
  A
  B
  C
  D
  E
  F
 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 1A
 1B
 1C
```

```
 1D
 1E
 1F

SCR # 1F
 10 ???ADCANDASLBCCBCSBEQBITBMIBNEBP
 11 LBRKBVCBVSCLCCLDCLICLVCMPCPXCPYD
 12 ECDEXDEYEORINCINXINYJMPJSRLDALDX
 13 LDYLSRNOPORAPHAPHPPLAPLPROLROR
 14 RTIRTSSBCSECSEDSEISTASTXSTYTAXTA
 15 YTSXTXATXSTYA
 16
 17
 18    #X))Y,X,YN).A
 19
 1A
 1B
 1C
 1D >> DECOMP DISASSEMBLER STUFF <<
 1E  DO NOT MOVE FROM THIS SCREEN !
 1F

SCR # 20
  0 ( Greetings from  Team Forth  )
  1           24 LIST = crib sheet >
  2 Team Forth welcomes you to the
  3 world of Atari fig-Forth. We are
  4 trying to make FORTH easy for
  5 you to learn and use. Since we
  6 are learning too, we would be
  7 pleased to get any feed back you
  8 may have on this version of fig-
  9 FORTH, and the included documen-
  A tation. Please send any comments
  B to Team Atari, 4029 Payne Ave.,
  C San Jose, Ca., 95117 or leave
  D E-mail on Compuserve for ~[70525,
  E 434] We'll try to answer all
  F correspondence. (TYPE LL) Rev.H+
 10 This disk is an attempt to write
 11 a self-tutorial on using this
 12 version of fig-FORTH. Just read
 13 the text and try the examples by
 14 placing the cursor on the line
 15 with the example and pressing
 16 the <RETURN> key. Have fun.
 17     NOTE 0 LIST is a catalogue.
 18 Let's try out a couple of words.
 19 ( UPPER LIST & LOWER LIST )
 1A
 1B Ok To begin, type in the lower
 1C screen area. First type UL <RE
 1D TURN> then to return here type
 1E LL <RETURN>, then to go to the
 1F next screen by pressing <N>

SCR # 21
  0 ( PREVIOUS, NEXT, TOGGLE       )
  1 Now type P and then use the T
```

```
  2 command to get back here. OK,
  3 N = next, P = previous and  T =
  4 toggle; FORTH stores the last
  5 two screens you LOADed or LISTed
  6 or EDITed in two buffers which
  7 you can toggle by using <T>.
  8 This is especially handy to
  9 compare screens from different
  A areas of a disk, or from two
  B different disks.
  C
  D The next handy-dandy command to
  E learn is the INDEX word.
  F       (TYPE  LL TO CONT.)
 10 INDEX is the word you use to see
 11 what is on one of your FORTH
 12 source disks. It shows you the
 13 first line of each screen on the
 14 disk. ( perhaps I should mention
 15 that FORTH organizes the disk in
 16 screens, the units you have been
 17 looking at, i.e. that which will
 18 fill up a TV screen. A good rule
 19 to follow is to make the first
 1A line of any screen you write a
 1B short description of what is on
 1C the screen. Then when you use
 1D INDEX you get a quick list of
 1E the contents. By printing this
 1F you have a nice (TYPE N)

SCR # 22
  0 ( ."/LPINDEX/SHOWLP/LPOPEN/    )
  1 listing of the contents of the
  2 disk. To try this out type in
  3 the following line down below:
  4 0 GR. 0 4D INDEX  or you might
  5 try useing <SHIFT> <CLEAR> and
  6 then typing 0 4D INDEX.
  7       ( Must be in caps )
  8 <."> is a FORTH word pronounced
  9 DOT QUOTE that works with <">
  A pronounced QUOTE to print the
  B text included between them.
  C  Make a note...
  D  TO RETURN HERE TYPE   12 UE
  E   (for page 12, uppper edit)
  F   then type LL to continue.
 10 Now to print the Index, you need
 11 to get your printer all ready,
 12 type LPOPEN,  and then type-
 13 0 4D LPINDEX. This will print it
 14 out. To print any single screen
 15 you type xx LISTLP (xx being the
 16 screen number). Two screens may
 17 be shown side by side on a page
 18 from the printer by typing -
 19 xx SHOWLP. This will print scrn.
 1A xx and the next screen after xx.
```

```
 1B
 1C
 1D REMEMBER TO RETURN TYPE  12 LE
 1E
 1F

SCR # 23
  0 ( BACKUP/CSECTS/WARN/GS/WS/N/+ )
  1 By now you may have wondered how
  2 to copy this disk. You can't do
  3 it using standard DOS as there
  4 is no Directory on a FORTH disk.
  5 You can use 'SUPERDUP', ARCHIVE,
  6 or any other sector copier.
  7 The word SYS will load several
  8 useful words like FORMAT and
  9 MAKEBOOT. MAKEBOOT makes a boot-
  A able disk with all the words you
  B have defined included.    SYS
  C will load a menu screen giving
  D instructions. NOTE: DO NOT MOVE
  E any of the screens 1 through B
  F and F !!!             (type LL)
 10 To turn on the english error
 11 message system, type WARNON. To
 12 turn them off and get only #
 13 msgs type WARNOFF.
 14
 15 The options available now for
 16 screen color are GS=green scrn,
 17 WS=white screen, NS=normal scrn,
 18 BS=black screen. Give them a try
 19 and see which ones you like.
 1A
 1B Now try HX, DX. The border color
 1C will change to remind you which
 1D number base you are in. NOTE
 1E HEX & DEC or DECIMAL are good
 1F too.          (type N to cont.)

SCR # 24
  0 ( GENERAL INFO.               )
  1 Now that we have some basic
  2 tools with which to find our
  3 way around the disk, a little
  4 general information on how the
  5 goodies are stored here. This
  6 disk contains most of the words
  7 of this version of fig-FORTH in
  8 the compiled form, i.e. they are
  9 LOADed and ready to use. You got
  A the list of words available to
  B you if you printed out the
  C VLIST. The second disk in this
  D package contains a very small #
  E of words in compiled form and
  F      (type LL to cont.)
 10 the rest are on source screens.
 11 This allows you to keep your
```

```
 12 kernel (the compiled words) of
 13 FORTH as small as possible, thus
 14 saving as much memory space in
 15 RAM as possible. The more words
 16 compiled in RAM the less room
 17 for you to work with in your own
 18 applications. In order to add a
 19 section of words to the kernel
 1A just LOAD the screen they are on
 1B by typing xx LOAD. (xx is the
 1C screen number). xx L& turns of
 1D the screen for faster compila-
 1E tion, but use it only on tested
 1F screens.      (type N to cont.)

SCR # 25
  0 ( LIST/L./DOIT/WIPE            )
  1 The format you have been using
  2 so far is the EDITOR format.
  3 You can look at screens in two
  4 different formats. By typing L#
  5 you will get line numbers on the
  6 editing screen to facilitate use
  7 of line edit commands. If you
  8 want to get rid of them later
  9 just type NL# and they will go.
  A
  B The other major format for look-
  C ing at screens is the LIST or L.
  D format. Type xx L. to see this
  E one.
  F               (type LL to cont.)
 10 In the LIST format you have to
 11 use the <CTRL><1> key to stop
 12 and start the scrolling of the
 13 screen.
 14
 15 The next screen is blank so you
 16 can try some screen editing.
 17 Type N to go there and then type
 18 on the screen. When done use the
 19 cursor arrow keys to move down
 1A to the line with "DOIT" on it &
 1B press the <RETURN> key. Then
 1C type FH or FLUSH to save it to
 1D the disk. To erase it type
 1E 16 WIPE, or W while in the
 1F editor (TYPE 17 UE TO SKIP THAT)

SCR # 26
  0 \ scr# 16  empty block       1/1
  1 HELLO
  2
  3
  4
  5
  6
  7
  8
```

```
  9
  A
  B
  C
  D
  E
  F
 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 1A
 1B
 1C
 1D
 1E
 1F

SCR # 27
  0 ( scr# 17  REV.G COMMENTS  ) ;S
  1 ( HES   82SEP27            )
  2 REV.G has a number of bug fixes
  3 for REV.A OS. It also intercepts
  4 the DOSINI vector during BOOT,
  5 and at WARMSTART (RESET). This
  6 makes it possible to set screen
  7 colors and margins at boot up
  8 and reset.
  9 Simply set your margins and
  A screen colors before calling SYS
  B Select STACKON or STACKOFF. Then
  C command SETSYS,  and MAKEBOOT.
  D Your new disk will now boot in
  E your colors,  and have your mar-
  F gins.       <LL>
 10 You can also hook in an ML SUB-
 11 ROUT.at this time. It must have
 12 been defined prior calling SYS.
 13 After calling SYS, command
 14 HOOK word (<your assembler word)
 15 UNHOOK reverses the process, if
 16 you think you made a mistake.
 17  NOTE:  Your word is called as
 18 a machine subroutine, and must
 19 end with RTS,.
 1A   The screen editor now has a
 1B KEYBOARD intercept,  so you can
 1C not accidentally hit CR while
 1D editing the upper screen. To
 1E override CR on the upper portion
 1F ( while editing) hit CTRL CR (N)

SCR # 28
```

```
  0 ( scr# 18  REV.G contd.    ) ;S
  1   UNLINK  (DEFAULT) disconnects
  2 Dr2 (810 only) from PHYSOFF,
  3 your second disk normally does
  4 not contain the BOOT,  e.g. SCR
  5 # 0 starts at sector 1. LINK
  6 reverses UNLINK.
  7 Note that a lot of screens have
  8 been changed from the previous
  9 REV., so full compilation
  A requires staying with one group
  B of source screens. The enhanced
  C SYS commands should not be used
  D with KERNELS of previous REV.s.
  E CHECK SCR#'s $24 for EDIT comnds
  F  ETC.. #$38 has REV H comments
 10 If you have questions, contact
 11 me (at reasonable hours, please)
 12     at <408>356-3921,
 13    or you can leave MESSAGES at
 14  the following BBS's:
 15    TABBS   <408> ???-????
 16    IBBS    <408> 298-6930
 17  (The latter is the preferred
 18   communications technique.)
 19 Since DAVE FLORY is distributing
 1A this package, naturally he can
 1B also be contacted.
 1C
 1D May fig-FORTH be with you...
 1E H.E.STRIEPE
 1F NOTES by John A. Peters 239-5393

SCR # 29
  0 ( PLAYER/MISSILE DEMO          )
  1 Now let's take a look at the
  2 player/missile demo. To do this
  3 please type 1A 1B SHOWLP. If you
  4 have not previously used the
  5 printer words type LPOPEN first.
  6 Then you will have the two scrns
  7 printed out and can compare them
  8 as you read the rest of this.
  9
  A The B/H word is used to convert
  B a binary shape table to hex and
  C decimal for use in Players and
  D missiles. See screen 1C for more
  E info.
  F         ( type 1C UE to cont.)
 10
 11
 12
 13
 14
 15
 16
 17
 18
```

```
 19
 1A
 1B
 1C
 1D
 1E
 1F

SCR # 2A
  0 ( PLAYER/MISS.STUFF-RZ 1/1 )
  1  0 VARIABLE 0VP 64 VARIABLE 0HP
  2  0 VARIABLE 0VPOLD
  3 : SPB  HIMEM @ 1+  F800 AND
  4     800 - DUP Qbase ! 17F +
  5               HIMEM ! ;
  6 : GETPS  0VP  ! ROT BLOCK ROT +
  7     Qbase @ 400 + 0VP  @ + ROT
  8     CMOVE ;
  9 : SPLAY  0 0 HPOS! 7 GR. SPB
  A   Qbase 1+ C@  PMBASE C!
  B   2A 0 COLPM!
  C   1 0 SIZE!   3E D400 C!
  D   3E DMACTL C! 3 GRACTL C!
  E   1C 20 8 64 GETPS  ;
  F  : CLRPM  Qbase @ 800 ERASE ;
 10 : MOVEH   0 STICK F XOR C AND
 11    DUP IF 2 / 3 - ENDIF 0HP @
 12    + DUP 0HP ! 0 HPOS!  ;
 13 : VPOS!  0VPOLD @ 9C00 + DUP
 14   9800 8 CMOVE  8 ERASE  9C00 +
 15   9800 SWAP 8 CMOVE ;
 16 : MOVEV        0 STICK F XOR
 17   3 AND   DUP IF 2 * 3 - ENDIF
 18   -DUP IF 0VP @ DUP 0VPOLD ! +
 19   DUP 0VP !  VPOS! ENDIF ;
 1A : RUNIT BEGIN   MOVEH MOVEV
 1B     2FC C@ FF = NOT END ;
 1C : BN BINARY ; : OCT OCTAL ;
 1D : B/H DUP HEX ." H
 1E EX =" . DX  ." DEC.=" . BN QUIT
 1F ;      6 GPRIOR C! ;S

SCR # 2B
  0 ;S ( P/M COMMENT & INSTRUCTIONS
  1    by Dave Flory/ Bay Area ATARI
  2 User Group.
  3 0VP = P/0 vert. position
  4 0HP = P/0 hor. position
  5 0VPOLD = P/0 old vert. position
  6 GETPS = get player shape & place
  7 in P/M memory space.
  8 SPLAY = show player. The 2A in
  9 front of the COLPM! is the color
  A of player in HEX. <setcolor colo
  B r * 16 + luminance>  The first
  C no. in front of SIZE! is the P/M
  D width no., 2nd is no. of P/M.
  E The no.s for GETPS in SPLAY are:
  F 1. the no. of screen holding P/M
```

```
 10 2. the # of bytes into the scrn.
 11 where the Player data starts.
 12 3. the # of bytes of player data
 13 4. the hor. pos. to show player
 14
 15 You must be careful to place the
 16 data correctly on the data scrn.
 17 Count in carefully and enter the
 18 data in characters. This is done
 19 by using BDUMP to enter the HEX
 1A numbers into RAM buffer position
 1B corresponding to the screen. See
 1C BDUMP comments & comments on the
 1D next screen which holds sample
 1E player data.   )
 1F             (TYPE N TO CONT.)

SCR # 2C
  0 ( PLAYER/MISSILE SHAPE TABLE )
  1 Z<   ~$<
  2
  3
  4
  5
  6
  7 The characters on the top of the
  8 screen were entered into the scr
  9 buffer usinG BDUMP after getting
  A the correct HEX numbers using
  B the B/H word. It is already
  C available in this version. (When
  D working with the source disk
  E Load the P/M screen.)
  F          (Type LL to cont.)
 10 This will give the numbers for a
 11 small flat box shape:
 12
 13 BN
 14 11111111 B/H HEX =FF DEC.=FF
 15 10000001 B/H HEX =81 DEC.=81
 16 10000001 B/H HEX =81 DEC.=81
 17 11111111 B/H HEX =FF DEC.=FF
 18
 19 To use B/H you must be in BN or
 1A binary base, then you enter the
 1B binary shape line and type B/H
 1C and press <RETURN>, FORTH will
 1D type the HEX and DEC. no.s for
 1E you.
 1F       (Type N to cont.)

SCR # 2D
  0 ( BDUMP/ EXAMPLE            )
  1 To put these numbers onto the
  2 disk without having to look up
  3 Atari control keys in the table,
  4 you use the BDUMP word. In this
  5 case it works like this. You
  6 type the following: 1C BLOCK
```

```
  7 DUP 2A + BDUMP    This gives you
  8 the top 2A character positions
  9 of screen 1C displayed on the
  A monitor. You then count in the
  B number of spaces you want the
  C data to be and enter it on the
  D screen using the cursor and
  E pressing <RETURN> on the same
  F line.  (Type LL to cont. )
 10 It looks like this:
 11
 12 1C BLOCK DUP 2A + BDUMP
 13 0500 28 20 50 4C 41 59 45 52
 14 0508 2F 4D 49 53 53 49 4C 45
 15 0510 20 53 48 41 50 45 20 54
 16 0518 41 42 4C 45 20 29 20 20
 17 0520 18 FF 24 DB 24 5A 99 81
 18 0528 20 20 20 20 20 20 20 20
 19  oK
 1A The place that the player is on
 1B now is displayed on line 520.
 1C By putting the cursor on that
 1D line and typing in the no.s you
 1E see how I entered the player
 1F shape.   (type N to cont.)

SCR # 2E
  0 ( SPLAY/RUNIT/GRAPHICS WORDS )
  1 Enter 6 GPRIOR C!
  2 To see the shape and run the
  3 player/missile demo type SPLAY
  4 and RUNIT. This will display the
  5 shape under the control of joy-
  6 stick 0. You will notice that
  7 the shape "hides" behind the
  8 text window and and the stack
  9 display (if the latter is on).
  A Later you can draw some play-
  B field stuff and experiment with
  C making him go in front of some
  D and behind others by putting a
  E different # in GPRIOR.
  F             ( LL )
 10 The graphics commands in this
 11 FORTH for the Atari are very
 12 like those you use in BASIc
 13 except that as you have noted
 14 here the numbers come first not
 15 after the commands.
 16
 17 x C. = COLOR x
 18 x GR. = GRAPHICS x
 19 x y PL. = PLOT x y
 1A x y DR. = DRAWTO x y
 1B similarly POS. is POSITION, SE =
 1C SETCOLOR, etc. The fill command
 1D is XIO18 and it requires you to
 1E put the fill color on the stack
 1F first.            (N)
```

```
SCR # 2F
  0 ( DEMO OF P/M PRIORITY         )
  1 ( For a demo load this screen.
  2 It will draw a couple of bars
  3 of vertical colors for the bug
  4 to play in. After typing SPLAY
  5 type 1F LOAD then type RUNIT. )
  6
  7 DX  1 C. 90 90 PL. 90 10 DR. 80
  8 10  DR. 80 90 POS. 1 XIO18 ( FIR
  9 ST BAR ) 2 C. 70 90 PL. 70 15
  A DR. 50 15 DR. 50 90 POS. 2 XIO18
  B  ( 2ND BAR ) 3 C. 30 40 PL.
  C 30 20 DR. 10 15 DR. 10 80 POS.
  D 3 XIO18 ;S 3rd bar fill runs
  E over to the right this time
  F            (LL)
 10 as the POS. statement is lower
 11 than the initial PL. statement
 12 for this bar.
 13
 14 After looking at this one play
 15 around a little. It will take
 16 some work to make this display
 17 four players at the same time,
 18 but its fun to try. You may want
 19 to go back and redefine some of
 1A the words to expect the player
 1B number on the stack to tell them
 1C which player to work on, or you
 1D can just define a word for each
 1E player and call them by name.
 1F Hope you have fun.  (N)

SCR # 30
  0 ( STACK DISPLAY                )
  1 One of the nice features of this
  2 system is the stack display. To
  3 use it just type STACKON. Then
  4 type a few numbers and press
  5 <RETURN>. You will see them show
  6 on the stack display at the top
  7 of your monitor screen. To turn
  8 it off when unwanted STACKOFF.
  9 You will find the STACK display
  A very useful to you as you start
  B defining your own words as you
  C can go through each step in the
  D word singly and watch what it
  E does to the stack. The stack is
  F the single most difficult  (LL)
 10 thing for most begiinning FORTH
 11 nuts to understand, and this
 12 display is used by even the
 13 experienced programmers I know.
 14
 15 Put some numbers on the stack
 16 and try some of the Math words
```

```
 17 like + and /, and *. Use SWAP,
 18 and ROT, and . and DROP, etc.
 19 and observe their stack effects.
 1A
 1B NOTE THIS Forth is based on the
 1C HEX number system ( 1 2 3 4 5 6
 1D 7 8 9 A B C D 10 11 12 etc.)
 1E
 1F

SCR # 31
  0 ( DECOMPILER/DCP/ZZ            )
  1 Now we come to one of the most
  2 useful and powerful words in the
  3 vocabulary of this FORTH version
  4 the decompiler word. To use it
  5 just type DC xxxx, with xxxx the
  6 word you want decompiled or
  7 taken apart. This word works on
  8 even the primitive words in the
  9 fig-FORTH kernel. Try a few of
  A the simple words first, like
  B GS, or NS. When you do you will
  C find that the numbers above 0,1,
  D and 2 all have LIT after them.
  E The first three are used so much
  F that they are FORTH words. (LL)
 10 ( screen 21 )
 11 The LIT tells FORTH that the
 12 They are to be taken as literal
 13 values. To give you an idea how
 14 this works try typing in this
 15 definition and then DC AS.
 16
 17 DX : AS 26 709 C! 18 710 C!
 18 16 712 C! ;
 19
 1A You now have another word avail.
 1B to you AS or amber screen. To
 1C make this a permanent part of
 1D your disk you can use MAKEBOOT
 1E to write out all the compiled
 1F words you have in RAM. (N)

SCR # 32
  0 ( TEXTS ON FORTH               )
  1 This should give you enough to
  2 think about for quite a while if
  3 you are new to FORTH. We haven't
  4 tried to make you a programmer,
  5 but just to introduce you to the
  6 features of this version of the
  7 language and help you over a few
  8 of the initial rough spots.
  9
  A There are several good books for
  B new people in FORTH. The one I
  C personally found most helpful
  D was DISCOVER FORTH published by
```

```
  E McGraw Hill. The best next book
  F and probably the best    (LL)
 10 if you can only afford to buy
 11 one is STARTING FORTH by Leo
 12 Brodie, and published by FORTH
 13 Inc.
 14
 15 I wish you the best of luck and
 16 hope that you will keep us up to
 17 date on your extensions to this
 18 language. We will try to act as
 19 a clearing house for ATARI FORTH
 1A people. AS we come up with more
 1B features we will add them to the
 1C package. we will try to upload
 1D the new stuff onto Compuserve
 1E ACCESS area so that you can get
 1F              (N)

SCR # 33
  0 ( CONCLUSION                    )
  1 updates without a lot of mailing
  2 disks and paying postage. To
  3 this end I suggest that you keep
  4 the original source disk without
  5 change so that you can recompile
  6 your working kernel as you get
  7 updates. If someone writes us
  8 a terminal prgram written in
  9 Atari FORTH we will make it
  A available immediately so that we
  B can all communicate FORTH stuff
  C more easily, without typing in
  D the screens on the keyboard.
  E
  F             (LL)
 10 That's all, keep on SWAPing,
 11
 12     Dave Flory Presently pres.
 13
 14   BAY AREA ATARI USER GROUP
 15      4029 PAYNE AVENUE
 16       SAN JOSE, CALIF.
 17           95117
 18
 19       408) 244-7181
 1A    COMPUSERVE ~[70424,434]
 1B
 1C       02:26 27 June, 1982
 1D
 1E PLEASE COPY AND PASS THESE DISKS
 1F AROUNDTO EVERY ONE YOU KNOW <N>

SCR # 34
  0 \ scr# 24  empty block       1/1
  1  ;S
  2
  3
  4 L,N,P      - List current,next
```

```
 5              or previous screens
 6 LL         - List lower 1/2 of
 7              current edit screen
 8 UL         - List upper 1/2 of
 9              current edit screen
 A DOIT       - Take top 16 lines
 B              of screen and place
 C              them into the top
 D              or bottom 1/2 ( LL
 E              or UL ) of the edit
 F              screen.
10 x y COPY   - Copy block x to
11              block y. No change
12              to block x
13   n LIST   - Set SCR to n and
14              list the block
15 x y SHOW   - List blocks x to y
16              inclusive
17 x y INDEX  - List first line of
18              blocks x thru y
19    FLUSH   - Return to FORTH voc
1A              and write out all
1B              updated blocks
1C   UPDATE   - Mark block (SCR) as
1D              updated
1E
1F fig-FORTH 1.4S comnds next block

SCR # 35
 0       FAST EDIT  COMMANDS
 1
 2   EDT   - same as EDITOR
 3   FORTH - exits EDITOR without
 4           action
 5   n UE  - same as n EDIT UL
 6   n LE  - same as n EDIT LL
 7     N   - edit next upper scrn
 8     N.  - edit next lower scrn
 9     P   - edit prev upper scrn
 A     p.  - edit prev lower scrn
 B     T   - edit other upper scrn
 C           in buffer
 D     T.  - edit other lower scrn
 E           in buffer
 F     FH  - same as FLUSH
10   WIPE   - clr scrn to be edited
11     W    - WIPE,will respond
12            with question, RETURN
13            or Y will execute
14   LOAD   - FLUSH scrn edited,and
15            LOAD
16   n LOAD - will flush and load n
17   L#OFF  - off LINE # display
18   L#ON   - on  LINE # display
19   SOUNDOFF - remove tone cue
1A            (the beep heard for
1B             various edit comnds)
1C   SOUNDON  - reset tone cue
1D
```

```
 1E
 1F   LINE EDITOR COMMNANDS NXT SCRN

SCR # 36
  0   EDITOR LINE EDITING COMMANDS
  1
  2   n TL  - Type line n    >(PAD)
  3   n HL  -      line n    >(PAD)
  4   n DL  - Delete line n  >(PAD)
  5   n IL  - Insert (PAD) after n
  6   n RL  - Replace n with (PAD)
  7   n SL  - Spread at line n
  8   n BL  - Blank line n
  9
  A   n $_  - Text following $_ will
  B           replace line n and go
  C           to PAD
  D   n %_  - Text following %_ will
  E           be inserted after line
  F           n and go to PAD
 10 s n CL  - Move line n of block s
 11           to PAD
 12
 13
 14 *** NOTE ***
 15
 16  >(PAD) : Means that line n is
 17           also moved to PAD
 18
 19
 1A
 1B
 1C
 1D  ( fig-FORTH 1.4S COMNDS NXT )
 1E
 1F

SCR # 37
  0     fig-FORTH 1.4S COMMANDS
  1
  2 n1 n2 n3 COPIES
  3         - move scrns n1 to n2
  4           to location starting
  5           at scrn n3 (n1-->n3,
  6           n1+1-->n3+1 etc.)
  7 n1 n2 DUPLICATE
  8         - will duplicate scrns
  9           on a single drive
  A SYS     - loads bootmaker and
  B           related words on
  C           SCREEN # $7
  D STACKON - turn stackdisplay on
  E STACKOFF- turn stackdisplay off
  F  NOTE:  BORDER REFLECTS # BASE
 10  STON   - STACKON
 11  STOF   - STACKOFF
 12  WARNON - display warning txt
 13  WARNOFF- just display error #
 14  DRAIN  - EMPTY-BUFFERS
```

```
 15  n1 n2 LZERO
 16          - CLEARS scrns n1 to n2
 17  n SETPHYS
 18          - set PHYSOFF to n
 19  RESPHYS- reset PHYSOFF to orgnl
 1A           value, use n STEPHYS
 1B           twice to chnge RESPHYS
 1C  DCP     - DECOMP
 1D  ZZ      - DECOMP
 1E  x y CDUMP - dump char. x to y,
 1F   USE LIKE BDUMP-  cont nxt scrn

SCR # 38
  0      fig-FORTH 1.4S COMMANDS
  1 BY BOB GONSALVES/ANTIC MAGAZINE
  2 n ><      - swap bytes
  3 n MSBYTE - leave MSB of n
  4 n LSBYTE - leave LSB of n
  5   VAR     - TO variable
  6             n TO <varname> will !
  7             n <varname> will @
  8 n MSB     - shortform MSBYTE
  9 n LSB     - shortform LSBYTE
  A
  B BY R.MANSFIELD /COMPUTE! & HES
  C x y FIND text - search for text
  D  starting with scr# x to y, will
  E search all scr of DR1 as default
  F abort by pressing START button
 10   NS,GS,WS,BS
 11           - change screen colors
 12   n U.    - n . unsigned
 13   VERIFY & NOVERIFY
 14           - change diskhandler
 15             write command
 16   n SNDOFF - turn off channel n
 17             (e.g. n 0 0 0 SOUND)
 18   THERE   - returns adr below
 19             display
 1A   FREE    - THERE HERE -
 1B   OCTAL   - change BASE to 8
 1C   BINARY - change BASE to 2
 1D   HX,DX,BX - shortforms
 1E   PON now does not require POFF
 1F   in same expression.   (contd.)

SCR # 39
  0 ( fig-FORTH 1.4S COMMANDS  ) ;S
  1
  2 n CHR  selects between three
  3          character sets, 0 is
  4          the normal ATARI vers
  5
  6 n .CHRSET   prints out the
  7             desired set
  8
  9
  A read the TUTORIAL on the use of
  B  BDUMP and CDUMP ( note that
```

```
  C     CDUMP has a bug, and cannot
  D     handle  's )
  E
  F
 10 There are several ways to access
 11 the second drive in a two drive
 12 system. Screens starting at $800
 13 will access the second drive. If
 14 you are in the LINKed mode, SCR#
 15 $0 will start with the same
 16 PHYSOFF as dr 1. UNLINK starts
 17 drive two on the first sector.
 18 You can also use the standard
 19 Forth procedure and call DR0 for
 1A dr 1,  and DR1 for dr 2. Each in
 1B this case have their own PHYSOFF
 1C , which can be set with XXYY SET
 1D PHYS,  where YY is the PHYSOFF
 1E for dr1, and XX for dr2.
 1F            < N >

SCR # 3A
  0 \ scr# 2A  REV H NOTES HES
  1
  2  Note that a 2 SETPHYS will set
  3 the PHYSOFF for dr2 to 0.
  4  NEWDATE lets you enter your ini
  5 tials and date that will be used
  6 in the EDITOR by DATE to date th
  7 e comment line. In FORTH, DATE
  8 requires a screen range on the s
  9 tack. ( Works well with LZERO ).
  A PNS will convert a PNS screen th
  B at you might be editing. Just do
  C  the normal n UE (n is scr#),  t
  D han command PNS.  The \ comment
  E line is now also acceptable to f
  F ig-Forth.
 10 PICK and ROLL also have been add
 11 ed. n PICK will take the nth ite
 12 m on the stack, and duplicate it
 13 on top,  n ROLL just rolls it to
 14  the top.
 15 SETSYS, HOOK/UNHOOK are now part
 16 of the std. vocabulary.  JMP and
 17 JSR permit ML experiments,  just
 18 leave the address of your routin
 19 e on TOS.
 1A DR2 adds $800 to scr# (makes it
 1B easier when using DECIMAL mode).
 1C
 1D
 1E
 1F

SCR # 3B
  0 ( scr# 2B  empty block 1/1 ) ;S
  1
  2
```

```
  3
  4
  5
  6
  7
  8
  9
  A
  B
  C
  D
  E
  F
 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 1A
 1B
 1C
 1D
 1E
 1F

SCR # 3C
  0 ( scr# 2C  empty block 1/1 ) ;S
  1
  2
  3
  4
  5
  6
  7
  8
  9
  A
  B
  C
  D
  E
  F
 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 1A
 1B
```

```
 1C
 1D
 1E
 1F

SCR # 3D
  0 ( scr# 2D  empty block 1/1 ) ;S
  1
  2
  3
  4
  5
  6
  7
  8
  9
  A
  B
  C
  D
  E
  F
 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 1A
 1B
 1C
 1D
 1E
 1F

SCR # 3E
  0 ( scr# 2E  empty block 1/1 ) ;S
  1
  2
  3
  4
  5
  6
  7
  8
  9
  A
  B
  C
  D
  E
  F
 10
 11
 12
```

```
 13
 14
 15
 16
 17
 18
 19
 1A
 1B
 1C
 1D
 1E
 1F

SCR # 3F
  0 ( scr# 2F  empty block 1/1 ) ;S
  1
  2
  3
  4
  5
  6
  7
  8
  9
  A
  B
  C
  D
  E
  F
 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 1A
 1B
 1C
 1D
 1E
 1F

SCR # 40
  0 ( FORTH TERMINAL HES   1/3 )
  1 ( V.1.0    HES 82AUG24     )
  2 ( USES IOCB #2/R1:,HANDLER )
  3 ( already booted           )
  4   FORTH DEFINITIONS   HEX
  5
  6 ( OPEN #2,R1: INPUT/OUTPUT )
  7  : OPNR:  2 IOCB ( IOCB #2 )
  8     0 D R1: OPEN ( INP/OUT )
  9   ( ICSTA CH? ) ; ( STATUS )
```

```
  A
  B ( CONFIG.#2,NO TRANSL,ATASCII )
  C  : CNFGR:  2 IOCB ( IOCB #2 )
  D      26 ICCOM C! R1: ICBAL !
  E      20 I1CAX ! CIO
  F   (  ICSTA CH? ) ; ( STATUS )
 10  0 VARIABLE STR:
 11 ( COMMAND n ---        )
 12  : STATR:   2 IOCB
 13      D  ICCOM C! ( COMMAND )
 14      CIO       ; ( STATUS  )
 15 ( START CONCURRENT IO )
 16  : STRTR:   2 IOCB 28 ICCOM C!
 17      R1: ICBAL ! 0 I1CAX C!
 18      CIO  ( ICSTA CH? ) ;
 19 ( CLOSE R: )
 1A  : CLSR:   2 IOCB CLOSE  ;
 1B ( CHECK   R:BUFFER )
 1C  : ?RBUF
 1D      STATR: 2EB @ ;
 1E   -->
 1F

SCR # 41
  0 ( FORTHTERM V.1.0      2/3 )
  1 ( CHECK KEYBOARD
  2  : ?CHLEAVE CH C@ FF = NOT IF
  3      FF CH C! ." Leaving MODE"
  4      CR LEAVE THEN ; )
  5 ( GET R: ROUTINE )
  6  : GETR    2 IOCB GET ;
  7
  8 ( PUT R: ROUTINE )
  9  : PUTR    2 IOCB PUT ;
  A
  B ( K: routines use IOCB #3  )
  C  : K: " K: " DROP ;
  D
  E ( Open for INPUT )
  F  : OPNK:  3 IOCB ( IOCB #3   )
 10     0 4 K: OPEN ( INPUT ONLY )
 11    ( ICSTA CH? ) ; ( STATUS  )
 12
 13 ( CLOSE )
 14  : CLSK:  3 IOCB  CLOSE  ;
 15
 16 ( GET INPUT  --- n )
 17  : GETK   3 IOCB  GET ;
 18
 19
 1A   -->
 1B
 1C
 1D
 1E
 1F

SCR # 42
  0 ( FORTHTERM V.1.0      3/3 )
```

```
 1
 2 : TERMINAL  CR CR
 3 ." fig-FORTH TERMINAL V.1.0"
 4 CR CR ."     START  --> exit"
 5 CR CR
 6 ."  ATASCII mode, full DUPLEX
 7 " CR CR 0 2BE C! ( lower case )
 8  OPNK: OPNR:   CNFGR:   STRTR:
 9
 A
 B  BEGIN
 C   ?RBUF IF       ( BUFFER ?? )
 D      GETR EMIT THEN ( GETIT  )
 E   CH C@ FF = NOT ( KEY PRESS )
 F   IF GETK PUTR THEN ( SEND IT)
10   ?CONSOL 7 AND  UNTIL ( EXIT )
11   CLSK: CLSR: CR
12
13    CR ." Leaving terminal mode."
14    40 2BE C!  CR CR ;
15
16
17
18
19  ;S
1A
1B
1C
1D
1E
1F

SCR # 43
 0 ( FORMATTED LIST PROG. 1/5 )
 1
 2  VOCABULARY FORMY   IMMEDIATE
 3  FORMY   DEFINITIONS
 4   BASE @ OCTAL 40 CN SPACBYT 54
 5 CN COMCHR : IARRAY 0 VARIABLE -2
 6  ALLOT ; : 0> DUP 0= IF DROP  0
 7 ELSE  0< 0= THEN ;
 8 0 VARIABLE INDENT 106 CN FCONS
 9 111 CN ICONS 0 VARIABLE TLFLG
 A 0 VARIABLE KERKNT 100 CN MAXLIN
 B : NXSPACE >R 1+ >R 0   R> R> DO
 C SPACBYT I C@ = IF DROP I LEAVE
 D THEN LOOP ; : NXNSPACE >R 1+ >R
 E 0 R> R> DO SPACBYT I C@ = 0= IF
 F
10  DROP I LEAVE THEN LOOP ; : GTNX
11 WD DUP IF + OVER SWAP NXSPACE
12 ELSE DROP THEN DUP IF OVER SWAP
13 NXNSPACE DUP IF OVER OVER
14 NXSPACE DUP IF OVER - ELSE DROP
15 OVER OVER - 1+ THEN ELSE DUP
16 THEN ELSE DUP THEN ; : TORLCR TL
17 FLG @ IF CRLP ELSE CR THEN KERKN
18 T 0SET ; : TORLY DUP 1+ KERKNT +
19 ! TLFLG @ IF LYPE ELSE TYPE SPAC
```

```
 1A E THEN ; : DOIND INDENT @ 0> IF
 1B INDENT @ 0 DO 0 0 TORLY LOOP THE
 1C N ; : PRWORD DUP 1+ KERKNT @ + M
 1D AXLIN > IF TORLCR THEN KERKNT @
 1E 0= IF DOIND THEN OVER OVER TORLY
 1F  ; : 1SET 1 SWAP ! ;  -->

SCR # 44
  0 ( FORMATTED LIST PROG. 2/5 )
  1 : ( 51 WORD 6 ALLOT ;
  2 : IA IARRAY ; IA L1G 10  , ( :)
  3 ( CODE) ( ,CODE) ( SUBROUTINE)
  4  ( IA) ( IARRAY) ( LABEL) ( TBL)
  5  IA L2G 2 , ( ;) ( C;)
  6  IA L3G 2 , ( NXT,) ( NEXT,) IA
  7 L4G 6 , ( IF) ( DO) ( IF,)
  8   ( CASE) ( BEGIN) ( BEGIN,) IA
  9 L5G 3 , ( ELSE,) ( ELSE)
  A   ( WHILE) IA L6G 16 , ( THEN,)
  B ( THEN) ( END,) ( END) ( SOB,)
  C ( BACK) ( UNTIL) ( AGAIN) ( REPE
  D AT) ( ENDIF,)
  E   ( UNTIL,) ( LOOP) ( +LOOP) ( E
  F NDIF) IA L7G 7 , ( CONSTANT)
 10 ( IR)   ( VARIABLE)       ( CN)
 11 ( ARRAY) ( INTEGER) ( ORCON)
 12    IA L8G 1 , ( () IA L9G 3 , (
 13 LD,) ( ST,) ( LOAD)
 14 IA LAG 1 , ( ;CODE)
 15
 16 : CMPWORD DUP >R C@ OVER = R>
 17 SWAP IF >R OVER
 18   R> SWAP OVER DUP C@ DUP 4 > IF
 19  DROP 4 THEN 0
 1A   DO I OVER + 1+ C@ >R OVER R>
 1B SWAP I + C@
 1C   = 0= IF 0 LEAVE THEN LOOP
 1D
 1E   0= IF DROP DROP 0 THEN ELSE 0
 1F THEN ;   -->

SCR # 45
  0 ( FORMATTED LIST PROG. 3/5 )
  1 : GSCAN DUP @ SWAP 2+ SWAP 0 DO
  2 CMPWORD IF LEAVE
  3   0 ELSE 6 + THEN LOOP IF 0 ELSE
  4  DROP 1 THEN ;
  5 : NEWCR KERKNT @ IF TORLCR THEN
  6 ;
  7 : DUPBC OVER >R >R OVER R> SWAP
  8 R> ;
  9 : FINDCHAR SWAP >R SWAP 1+ R>
  A DO DUP I C@ =
  B   IF DROP I LEAVE 0 THEN LOOP IF
  C  0 THEN ;
  D : PRNEWL PRWORD TORLCR ;
  E : >= OVER OVER = IF DROP DROP
  F 1 ELSE > THEN ;   -->
 10
```

```
 11
 12
 13
 14
 15
 16
 17
 18
 19
 1A
 1B
 1C
 1D
 1E
 1F

SCR # 46
  0 ( FORMATTED LIST PROG. 4/5 )
  1 : EL1G NEWCR INDENT 0SET PRWORD
  2 GTNXWD PRNEWL
  3   10 INDENT ! ;
  4 : EL2G NEWCR PRNEWL INDENT 0SET
  5 ;
  6 : EL3G NEWCR PRNEWL ;
  7
  8 : EL4G NEWCR PRNEWL 2 INDENT +!
  9 ;
  A : EL5G NEWCR -2 INDENT +! PRNEWL
  B  2 INDENT +! ;
  C : EL6G NEWCR -2 INDENT +! PRNEWL
  D  ;
  E : EL7G PRWORD GTNXWD PRNEWL INDE
  F NT 0SET ;
 10 : EL8G DUPBC 51 FINDCHAR DUP
 11
 12   IF SWAP DROP OVER - 1+ PRNEWL
 13 ELSE DROP PRWORD THEN ;
 14 : EL9G PRNEWL ;
 15
 16 : ELAG NEWCR 10 INDENT ! PRNEWL
 17 ;
 18 : ASSWRD DUP 4 >= IF OVER OVER +
 19  1- C@ COMCHR = IF
 1A    OVER DUP C@ ICONS =  SWAP 1+
 1B C@ FCONS = AND
 1C   IF 2 ELSE 1 THEN ELSE 0 THEN E
 1D LSE 0 THEN ;
 1E -->
 1F

SCR # 47
  0 ( FORMATTED LIST PROG. 5/5 )
  1 : PRCWRD L1G GSCAN IF EL1G ELSE
  2 L2G GSCAN IF EL2G ELSE
  3   L3G GSCAN IF EL3G ELSE L4G GSC
  4 AN IF EL4G ELSE L5G GSCAN
  5   IF EL5G ELSE L6G GSCAN IF EL6G
  6  ELSE L7G GSCAN IF EL7G
  7   ELSE L8G GSCAN IF EL8G ELSE L9
```

```
 8 G GSCAN IF EL9G ELSE
 9   LAG GSCAN IF ELAG ELSE ASSWRD
 A IF ASSWRD 2 =
 B   IF EL4G ELSE PRNEWL THEN ELSE
 C PRWORD THEN THEN THEN THEN THEN
 D THEN THEN THEN THEN THEN THEN ;
 E : FORLST TORLCR DUP TLFLG @ IF L
 F ISTLP ELSE
10  TORLCR LIST THEN TORLCR TORLCR
11 DUP BLK !
12   BLOCK DUP 1777 + SWAP KERKNT 0
13 SET INDENT 0SET 0 BEGIN GTNXWD
14   DUP IF PRCWRD THEN DUP 0= END
15 DROP DROP DROP BLK 0SET ;
16 : ASTER TORLCR 40 0 DO 52 SP@ 1
17 TORLY DROP LOOP TORLCR ;
18 : FORSHW 1+ OVER DO ASTER I FORL
19 ST TORLCR LOOP DROP ;
1A FORTH DEFINITIONS : FLST FORMY T
1B LFLG 0SET FORLST ; : FLSTLP FORM
1C Y TLFLG 1SET FORLST FFLP ; : FSH
1D W FORMY TLFLG 0SET FORSHW ; : FS
1E HWLP FORMY  TLFLG 1SET FORSHW
1F FFLP  ;  LPOPEN ;S
```