

```

0 ( GAME-OV ----- OVERLAY PREFIX -----
1 ( GAME OPTIONS - CONSTANTS
2 ( GAME OPTIONS - VARIABLES
3 ( GAME OPTIONS - VARIABLES
4 ( GAME OPTIONS - !$ #CPL #LPS
5 ( GAME OPTIONS - ?BL<-
6 ( GAME OPTIONS - ##LINES
7 ( GAME OPTIONS - #YBLT #XBLT .SUB$
8 ( GAME OPTIONS - .#LINES
9 ( GAME OPTIONS - .FORMAT BORDERMARGINS
10
11
12 ( GAME OPTIONS - ERASEWIND ERASELINE .PAGE
13 ( GAME OPTIONS - BORDER
14 ( GAME OPTIONS - PRESSANY .GAMEOPS
15 ( GAME OPTIONS - .LOADGAME .SAVEGAME
16
17 ( GAME OPTIONS - .ERROR1 BLINK
18 ( GAME OPTIONS - ?GAMEDSK !TIMESTAMP TME>DISK
19 ( GAME OPTIONS - .INSERTSAVE REMOVEGAME
20 ( GAME OPTIONS - .ERROR2 LOADSAVE
21 ( GAME OPTIONS - .INSERTLOAD
22 ( GAME OPTIONS - .CKSUMERROR SAVECHK
23 ( GAME OPTIONS - LOADGAME
24 ( GAME OPTIONS - .INSERTGAME
25 ( GAME OPTIONS - .KEYTORESUME
26 ( GAME OPTIONS - PRESERVEDISPLAY RESTOREDISPLAY
27 ( GAME OPTIONS - ESCDISABLE ESCENABLE OFFCACHE ONC
28 ( GAME OPTIONS - CHECKSUM
29
30 ( GAME OPTIONS - .LOADING .LOADED
31 ( GAME OPTIONS - .SAVING DONE
32 ( GAME OPTIONS - PRMSAV PRMLD
33 ( GAME OPTIONS - RETSAV RETSAV
34 ( GAME OPTIONS - .RS testing words
35 ( GAME OPTIONS - !DISK
36 ( GAME OPTIONS - <DISK >DISK
37 ( GAME OPTIONS - ARR>DISK DISK>ARR
38 ( GAME OPTIONS - DSP>DISK DISK>DSP
39 ( GAME OPTIONS - DISK>KRN KRN>DISK
40 ( GAME OPTIONS - RTN>DISK DISK>RTN PRM>DISK
41 ( GAME OPTIONS - DATASAVE
42 ( GAME OPTIONS - SETSAVE CLRSAVE
43 ( GAME OPTIONS - DATALOAD
44 ( GAME OPTIONS - LOADGAME
45 ( GAME OPTIONS - SAVEGAME
46 ( GAME OPTIONS - .CONFIGHEAD MONITORTYPE
47 ( GAME OPTIONS - MEMORYSIZE
48 ( GAME OPTIONS - CONFIGGAME
49
50 ( SAVEGAME - GET-OPTION#
51 ( SAVEGAME - GAMEOPCASES RESUMEGAME
52 ( GAME OPTIONS - >GAMEOPTIONS
53 ( GAME OPTIONS - GAMEOPTIONS
54 ( GAME OPTIONS - <GAMEOPTIONS
55 ( GAME OPTIONS - GAMEOPS

```

```

55 ( GAME OPTIONS - GAMEOPS
56
57 ( GAME-OV ----- OVERLAY SUFFIX -----

```

0

3

```

0 ( GAME-OV ----- OVERLAY PREFIX ----- 7-11-85) ( GAME OPTIONS - VARIABLES 9-26-85)
1
2 VOCABULARY GAME IMMEDIATE
3 120 OPEN-OVERLAY
4 GAME DEFINITIONS
5
6 2500 TRANS-ALLOT ( allot space for heads)
7
8 NEWT-DP
9
10
11
12
13
14
15

```

V: LMARG (left margin)
 V: RMARG (right margin)
 V: TMARG (top margin)
 V: BMARG (bottom margin)
 V: CHARHEIGHT (height of character)
 V: CHARWIDTH (width of character)
 V: LINESPACING (pixels between lines)
 V: CPL (characters/line)
 V: LPS (lines/screen)
 V: \$ADDR (string address)
 V: \$LEN (length of string)
 V: #LINES (number of lines string occupies)
 V: SUB\$ADDR (substring address)
 V: SUB\$LEN (substring length)

1

4

```

0 ( GAME OPTIONS - CONSTANTS 9-26-85) ( GAME OPTIONS - !$ #CPL #LPS 9-26-85)
1 0 C: KRNBLK ( base block for kernal image)
2 64 C: DSPBLK ( base block for display image)
3 80 C: RTNBLK ( base block for return stack image)
4 81 C: PRMBLK ( base block for parameter stack)
5 82 C: ARRBLK ( base block for array data)
6 4 C: RDROP ( # bytes to drop from the return stack)
7 0 C: SDROP ( # bytes to drop from the parm stack)
8
9 BACKUP 5:
10 YBLT, XBLT, XORMODE, LBLT, WBLT, ABIT, BLTSEG
11 COLOR TO COLOR
12
13
14
15

```

HEAD: !\$ (addr cnt -- \ set current string)
 T: \$LEN ! \$ADDR ! T;

 HEAD: #CPL (-- \ compute characters/line)
 T: RMARG @ LMARG @ - CHARWIDTH @ / CPL ! T;

 HEAD: #LPS (-- \ compute lines/screen)
 T: TMARG @ BMARG @ - CHARHEIGHT @ LINESPACING @ + / LPS ! T;

2

5

```

0 ( GAME OPTIONS - VARIABLES 9-26-85) ( GAME OPTIONS - ?BL<- 9-26-85)
1 \ V= ESC-EN ( ESC key enable flag)
2 \ V= ESC-PFA ( pfa of current escape invoked routine)
3 \ V= [#CACHE] ( # of cache buffers possible)
4 V: [BCACHE] ( start of cache space address)
5 V: RESUME ( resume play flag)
6 V: TIMESTAMP ( save version identifier)
7 V: RTNTEMP 80 ALLOT ( temp return stack space)
8 V: NSYSK ( new memory size in k bytes)
9 V: CKSUM ( check sum accumulator)
10 V: TYPE-BAK ( TYPE VECTOR BACKUP)
11 V: EMIT-BAK ( EMIT VECTOR BACKUP)
12 V: BUFSeg-BAK ( DISPLAY BUFFER SEG BACKUP)
13 V: BUFCNT-BAK ( " " COUNT " ")
14 V: YTAB-BAK ( " " YTAGL " ")
15

```

HEAD: ?BL<- (addr -- addr' \ given address, end if space else)
 T: (decrement until a blank space is encountered)
 BEGIN
 DUP C@ BL = NOT
 WHILE
 1-
 REPEAT T;

6

9

```

0 ( GAME OPTIONS - ##LINES                                9-26-85) ( GAME OPTIONS - .FORMAT BORDERMARGINS 9-26-85)
1
2 HEAD: ##LINES ( -- \ compute number of lines will be occupied) HEAD: .FORMAT ( addr count -- \ plot center formatted string)
3 T: ( by the current string)                                T: !$ #CPL #LPS
4 $ADDR @ SUB$ADDR !                                         .#LINES T;
5 1 #LINES !          ( count initialized)
6 BEGIN
7   SUB$ADDR @ CPL @ + DUP
8   $ADDR @ $LEN @ + U<
9   WHILE
10    ?BL<-          ( address.of.linebreak.blank --)
11    1+ SUB$ADDR !
12    1 #LINES +!
13 REPEAT DROP T;
14
15

```

7

10

```

0 ( GAME OPTIONS - #YBLT #XBLT .SUB$                        9-26-85)
1
2 HEAD: #YBLT ( -- \ compute initial line location for current )
3 T: ( text message so that the message is centered)
4 ##LINES
5 LPS @ #LINES @ - 2/ CHARHEIGHT @ LINESPACING @ + * NEGATE
6 TMARG @ + YBLT ! T;
7
8 HEAD: #XBLT ( -- \ compute x location for current substring)
9 T: ( such that the current substring is centered)
10 YBLT @ 900 YBLT ! 0 XBLT ! SUB$ADDR @ SUB$LEN @ TYPE
11 YBLT ! RMARG @ LMARG @ + 2/
12 XBLT @ 2/ NEGATE + XBLT ! T;
13
14 HEAD: .SUB$ ( -- \ plot centered substring)
15 T: #XBLT SUB$ADDR @ SUB$LEN @ TYPE T;

```

8

11

```

0 ( GAME OPTIONS - .#LINES                                9-26-85)
1 HEAD: .#LINES ( -- \ plot current string as center formatted)
2 T: #YBLT
3 $ADDR @ SUB$ADDR !
4 BEGIN
5   SUB$ADDR @ CPL @ + DUP
6   $ADDR @ $LEN @ + U<
7   WHILE
8    ?BL<-          ( address.of.linebreak.blank --)
9    SUB$ADDR @ -
10   SUB$LEN ! .SUB$
11   CHARHEIGHT @ LINESPACING @ + NEGATE YBLT +!
12   SUB$LEN @ 1+ SUB$ADDR +!
13 REPEAT DROP
14 $ADDR @ $LEN @ + SUB$ADDR @ - SUB$LEN ! .SUB$ T;
15

```

12

15

```

0 ( GAME OPTIONS - ERASEWIND ERASELINE .PAGE          9-26-85) ( GAME OPTIONS - .LOADGAME .SAVEGAME          9-26-85)
1
2 HEAD: ERASEWIND ( -- \ erase text in display window)      HEAD: .LOADGAME ( -- \ display load game disk message)
3 T: COLOR @ 150 20 40 140 BLACK POLY-WINDOW-FILL !COLOR T;  T: >3FONT 53 171 POS. ." STARFLIGHT"
4                                " PLEASE INSERT A COPY OF THE GAME DISK INTO DRIVE A."
5 HEAD: ERASELINE ( -- \ erase text below display window)    .PAGE PRESSANY T;
6 T: COLOR @ 17 20 5 159 BLACK POLY-WINDOW-FILL !COLOR T;
7
8 HEAD: .PAGE
9 T: BORDERMARGINS ERASEWIND >2FONT .FORMAT T;
10
11
12
13
14
15

```

13

16

```

0 ( GAME OPTIONS - BORDER          9-26-85)
1
2 HEAD: BORDER ( -- \ display standard interface border with )
3 T: ( Interstel logo)
4 WHITE !COLOR
5 5 155 10 160 LLINE 10 160 150 160 LLINE
6 150 160 155 155 LLINE 155 155 155 25 LLINE
7 155 25 150 20 LLINE 150 20 20 20 LLINE
8 20 20 17 23 LLINE 17 23 17 30 LLINE
9 17 30 14 33 LLINE 14 33 8 33 LLINE
10 8 33 5 36 LLINE 5 36 5 155 LLINE
11 5 29 COLOR @ .!LOGO T;
12
13
14
15

```

14

17

```

0 ( GAME OPTIONS - PRESSANY .GAMEOPS          9-26-85) ( GAME OPTIONS - .ERROR1 BLINK          9-26-85)
1
2 HEAD: PRESSANY ( -- \ display message below border)      HEAD: .ERROR1 ( -- \ display load game disk message)
3 T: >2FONT 20 17 POS. ERASELINE      T: " PLEASE REMOVE THE GAME DISK." .PAGE T;
4 ." PRESS ANY KEY TO CONTINUE" >0FONT KEY DROP T;
5
6 HEAD: .GAMEOPS ( -- \ display message below border)
7 T: >DISPLAY DARK BORDER
8 >3FONT 46 171 POS. ." GAME" 10 XBLT +! ." OPTIONS"
9 >2FONT 39 17 POS. ." SELECT AN OPTION"
10 34 120 POS. ." 1. SAVE GAME"
11 34 100 POS. ." 2. RESUME GAME"
12 34 80 POS. ." 3. CONFIGURE SYSTEM" >0FONT T;
13
14
15

```


18

```

0 ( GAME OPTIONS - ?GAMEDSK !TIMESTAMP THE>DISK
1
2 HEAD: ?TIMEDISK ( -- t \ does loaded disk match timestamp?)
3 T: EMPTY-BUFFERS 359 BLOCK 1012 + @ TIMESTAMP @ = T;
4
5 HEAD: ?GAMEDSK ( -- t \ is the current game disk loaded?)
6 T: ?TIMEDISK
7 359 BLOCK 1008 + @ 23040 = AND T;
8 HEAD: !TIMESTAMP
9 T: ( -- \ take the 10 word of the real time clock)
10 ( and store it into the variable TIMESTAMP; 0 not allowed)
11 TIME 2+ @ ?DUP 0= IF 2 THEN TIMESTAMP ! T;
12
13 HEAD: THE>DISK ( -- \ write the timestamp to the disk)
14 T: ( NOTE: virgin play disk has 0 in timestamp field)
15 TIMESTAMP @ 359 BLOCK 1012 + ! SAVE-BUFFERS T;

```

19

```

0 ( GAME OPTIONS - .INSERTSAVE REMOVEGAME
1
2 HEAD: .INSERTBLANK ( -- \ display remove game disk message) T:
3 " PLEASE REMOVE THE GAME DISK AND INSERT A BLANK FORMATTED DIS
4 K FOR DATA STORAGE INTO DRIVE A." .PAGE T;
5
6 HEAD: REMOVEGAME ( -- \ make sure game disk is removed)
7 T: BEGIN
8 ?GAMEDSK ( check for removal of game disk)
9 WHILE
10 BEEP ' .INSERTBLANK 1 BLINK ( if game disk still in, beep)
11 PRESSANY
12 REPEAT T;
13
14 HEAD: INSERTBLANK ( -- \ insert a blank disk for game storage)
15 T: .INSERTBLANK PRESSANY REMOVEGAME T;

```

20

```

0 ( GAME OPTIONS - .ERROR2 LOADSAVE
1
2 HEAD: .ERROR2 ( -- \ feedback if correct save disk not loaded)
3 T: " INSERT THE CORRECT SAVE DISK INTO DRIVE A." .PAGE T;
4
5 HEAD: LOADSAVE ( -- \ make sure right save disk is loaded)
6 T: BEGIN
7 ?TIMEDISK NOT ( check for matching save disk)
8 WHILE
9 BEEP ' .ERROR2 1 BLINK ( if save disk not in, beep)
10 PRESSANY
11 REPEAT T;
12
13
14
15

```

21

```

9-26-85) ( GAME OPTIONS - .INSERTLOAD
HEAD: .INSERTLOAD ( -- \ display load save disk message) T:
" PLEASE REMOVE THE GAME DISK AND INSERT THE SAVE DISK INTO DR
IVE A." .PAGE T;
HEAD: REMOVEGAME2 ( -- \ make sure game disk is removed)
T: BEGIN
?GAMEDSK ( check for removal of game disk)
WHILE
BEEP ' .INSERTLOAD 1 BLINK ( if game disk still in, beep)
PRESSANY
REPEAT T;

HEAD: INSERTLOAD ( -- \ insert the save disk for loading)
T: .INSERTLOAD PRESSANY
REMOVEGAME2 ( check for removal of game disk)
LOADSAVE T; ( check for matching save disk)

```

22

```

9-26-85) ( GAME OPTIONS - .CKSUMERROR SAVECHK
HEAD: .CKSUMERROR ( -- \ warn user of checksum error)
T: ERASELINE BEEP
" SAVE DISK CONTAINS BAD DATA. PREPARE TO EXIT." .PAGE
PRESSANY PAGE BYE T;

HEAD: SAVECHK ( -- \ warn user game not saved)
T: 359 BLOCK 1016 + @ NOT
IF ERASELINE BEEP
" GAME IN PROGRESS NOT SAVED. PREPARE TO EXIT" .PAGE
PRESSANY PAGE BYE
THEN T;

```

23

```

9-26-85) ( GAME OPTIONS - LOADGAME
HEAD: .ERROR3 ( -- \ display re-load game disk message)
T: " RE-INSERT THE GAME DISK INTO DRIVE A." .PAGE T;

HEAD: LOADGAMEDSK ( -- \ make sure game disk is loaded)
T: BEGIN
?GAMEDSK NOT ( check for removal of game disk)
WHILE
BEEP ' .ERROR3 1 BLINK ( if game disk not in, beep)
KEY DROP
REPEAT T;

```

24

27

```

0 ( GAME OPTIONS - .INSERTGAME                      9-26-85) ( GAME OPTIONS - ESCDISABLE ESCENABLE OFFCACHE ONCACHE 9-26-85)
1
2 HEAD: .INSERTGAME ( -- \ display load game disk message) T:      HEAD: ESCDISABLE
3 " PLEASE REMOVE THE SAVE DISK AND RE-INSERT THE GAME DISK INTO T: ( -- \ turn off escape key path to game options)
4 DRIVE A." .PAGE                                         ESC-EN OFF T;
5 PRESSANY                                               HEAD: ESCENABLE
6 LOADGAMEDSK T; ( check for game disk reload)          T: ( -- \ turn on  escape key path to game options)
7                                                         ESC-EN ON T;
8
9                                                         ( install number of cache blocks in variable [#cache] during)
10                                                        ( configure-system)
11 HEAD: OFFCACHE ( -- \ turn cache memory off)
12 T: #CACHE OFF CACHE-BEGIN DUP @ [BCACHE] ! OFF CACHE-HEAD OFF T;
13
14 HEAD: ONCACHE ( -- \ turn cache memory back on)
15 T: [#CACHE] @ #CACHE ! AUTO-CACHE EMPTY-CACHE T;

```

25

28

```

0 ( GAME OPTIONS - .KEYTORESUME                      9-26-85) ( GAME OPTIONS - CHECKSUM                      9-26-85)
1
2 HEAD: .KEYTORESUME ( -- \ display press key to resume message) HEAD: CHECKSUM ( addr count -- \ accum a check sum given)
3 T: " PLEASE PRESS ANY KEY TO RESUME THE GAME IN PROGRESS." T: ( memory start and byte count)
4 .PAGE PRESSANY T;                                         0 00 DUP I + C@ CKSUM +! 1 /LOOP DROP T;
5
6
7
8
9
10
11
12
13
14
15

```

26

29

```

0 ( GAME OPTIONS - PRESERVEDISPLAY RESTOREDISPLAY  9-26-85)
1
2 HEAD: PRESERVEDISPLAY ( preserve current display)
3 T: DBUF-SEG @ 0 [BCACHE] @ 0 16384 LCMOVE T;
4
5 HEAD: RESTOREDISPLAY ( move picture from cache to display)
6 T: [BCACHE] @ 0 DBUF-SEG @ 0 16384 LCMOVE T;
7
8
9
10
11
12
13
14
15

```

1 EMIT @ MC + 3

30

```

0 ( GAME OPTIONS - .LOADING .LOADED
1
2 HEAD: .LOADING ( -- \ display message to user)
3 T: ERASELINE
4 " LOADING GAME..." .PAGE
5 55 17 POS. ." PLEASE WAIT" >OFONT T;
6
7 HEAD: .LOADED
8 T: ( -- \ game save complete, wait for key then exit)
9 ERASELINE
10 " ...GAME LOAD COMPLETE." .PAGE
11 PRESSANY T;
12
13
14
15

```

31

```

0 ( GAME OPTIONS - .SAVING DONE
1
2 HEAD: .SAVING ( -- \ display message to user)
3 T: ERASELINE
4 " SAVING GAME..." .PAGE
5 55 17 POS. ." PLEASE WAIT" >OFONT T;
6
7 HEAD: DONE ( -- \ game save complete, wait for key then exit)
8 T: ERASELINE
9 " GAME SAVE COMPLETE. REMOVE THE SAVE DISK AND PREPARE TO EXIT
10 ." .PAGE
11 PRESSANY DARK BYE T; ( exit to system)
12
13
14
15

```

32

```

0 ( GAME OPTIONS - PRMSAV PRMLD
1 CREATE PRMSAV ( addr -- \ save parm stack image at addr)
2 ( with the parameter stack pointer in the format:)
3 ( [sp] [-----1000 bytes-----] (-sp0)
4 HERE DUP 2- ! HEX 5B C, 89 C, 27 C, 43 C, 43 C, 56 C, 57 C,
5 8B C, FB C, 8B C, 36 C, SP0 , 81 C, EE C, EB C, 03 C, B9 C,
6 EB C, 03 C, F3 C, A4 C, 5F C, 5E C, AD C, 8B C, D8 C, FF C,
7 27 C,
8 CREATE PRMLD ( addr -- \ given addr of parm stack image)
9 ( move it to the stack area and load the parm stack pointer)
10 HERE DUP 2- ! 5B C, 8B C, 27 C, 43 C, 43 C, 89 C, 76 C, FE C,
11 89 C, 7E C, FC C, 8B C, F3 C, FA C, 1E C, 07 C,
12 8B C, 3E C, SP0 , 81 C, EF C,
13 EB C, 03 C, B9 C, EB C, 03 C, F3 C, A4 C, 03 C, 26 C, ' SDRDP ,
14 8B C, 76 C, FE C, 8B C, 7E C, FC C, FB C, AD C, 8B C, D8 C,
15 FF C, 27 C, DECIMAL

```

33

```

9-26-85) ( GAME OPTIONS - RETSAV RETSAV
8-28-85)
HEX
CREATE RETSAV ( addr -- \ save return stack image at addr)
( with the current return stack pointer in the format:)
( [rp] [-----80 bytes-----] (-r0) HERE DUP 2- !
5B C, 89 C, 2F C, 43 C, 43 C, 56 C, 57 C, 8B C, FB C, 8B C,
36 C, R0 , 83 C, EE C, 50 C, B9 C, 50 C, 00 C, F3 C, A4 C,
5F C, 5E C, AD C, 8B C, D8 C, FF C, 27 C,
CREATE RETJMP ( addr -- \ given addr of return stack image)
( move it to the return stack and jump to next word)
HERE DUP 2- ! FA C, 1E C, 07 C,
5B C, 8B C, 2F C, 43 C, 43 C, 8B C, F3 C, 57 C, 8B C, 3E C,
R0 , 83 C, EF C, 50 C, B9 C, 50 C, 00 C, F3 C, A4 C, 5F C,
03 C, 2E C, ' RDRDP , 8B C, 76 C, 00 C, 45 C, 45 C, FB C,
AD C, 8B C, D8 C, FF C, 27 C,
DECIMAL

```

34

```

9-26-85) ( GAME OPTIONS - .RS testing words
7-1-85)
exit
V= RBASE
V= R0'
V= RP'
: .RS ( addr -- \ dump return stack image names at addr)
DUP RBASE !
DUP B2 + R0' !
R0 @ SWAP @ - R0' @ SWAP - RP' !
BEGIN
RP' @ R0' @ = NOT
WHILE
RP' @ @ 2 RP' +!
2- @ 2+ NFA CR ID.
REPEAT ;

```

35

```

8-28-85) ( GAME OPTIONS - !DISK
9-26-85)
HEAD: !DISK ( mem-seg mem-offset base-blk #bytes tf -- \)
T: ( move data to/from memory to disk, tf=1-->disk )
>R DUP 0 1024 U/MOD ( mseg moff bblk #b rem blks -- tf)
SWAP IF 1+ THEN ( mseg moff bblk #b blks' -- tf)
?DUP IF 0 DO
4 PICK I 64 * + 4 PICK
@DS 5 PICK I + BLOCK 2OVER 2OVER
J NOT IF 2SWAP THEN
9 PICK 1024 UMIN LCMOVE
J IF UPDATE THEN
5 PICK 1024 UMIN CHECKSUM
DROP 2DROP 1024 -
LOOP
I IF SAVE-BUFFERS THEN
THEN R> DROP 2DROP 2DROP T;

```


36

```

0 ( GAME OPTIONS - <DISK>DISK
1
2 HEAD: <DISK
3 T: ( mem-seg mem-offset base-blk #bytes -- \ move data)
4   ( from disk to memory)
5   0 !DISK T;
6
7 HEAD: >DISK
8 T: ( mem-seg mem-offset base-blk #bytes -- \ move data)
9   ( from memory to disk)
10  1 !DISK T;
11
12
13
14
15

```

9-26-85) (GAME OPTIONS - DISK>KRN KRN>DISK

9-26-85)

```

HEAD: DISK>KRN ( -- \ move kernal thru OVA from save disk)
T: @DS 1800 KRNBLK OVA @ 1800 - <DISK T;

HEAD: KRN>DISK ( -- \ move kernal thru OVA to save disk)
T: @DS 1800 KRNBLK OVA @ 1800 - >DISK T;

```

37

```

0 ( GAME OPTIONS - ARR>DISK DISK>ARR
1
2 HEAD: ARR>DISK ( -- \ move array image to save disk)
3 T: LFSEG @ 0 ARRBLK
4   SYSK @ 64 * 1- LFSEG @ - 16* ( seg off blk #bytes --)
5   >DISK T;
6
7 HEAD: DISK>ARR ( -- \ load array image from save disk)
8 T: LFSEG @ 0 ARRBLK
9   SYSK @ 64 * 1- LFSEG @ - 16* ( seg off blk #bytes --)
10  <DISK T;
11
12
13
14
15

```

9-26-85) (GAME OPTIONS - RTN>DISK DISK>RTN PRM>DISK

9-26-85)

```

HEAD: RTN>DISK ( -- \ save return stack to disk)
T: RTNBLK BLOCK RETSAV UPDATE SAVE-BUFFERS T;

HEAD: DISK>RTN
T: ( -- \ get return stack from disk and move it to)
  ( return stack temp array)
  RTNBLK BLOCK RTNTEMP 82 CMOVE T;

HEAD: PRM>DISK ( -- \ save parm stack to disk)
T: PRMBLK BLOCK PRMSAV UPDATE SAVE-BUFFERS T;

HEAD: DISK>PRM ( -- \ get parm stack from disk)
T: PRMBLK BLOCK PRMLD T;

```

38

```

0 ( GAME OPTIONS - DSP>DISK DISK>DSP
1
2 HEAD: DSP>DISK ( -- \ move display image from temporary )
3 T: ( location in cache to save disk)
4   [BCACHE] @ 0 DSPBLK 16 1024 * >DISK T;
5
6
7 HEAD: DISK>DSP ( -- \ move display image from save disk to )
8 T: ( temporary location in cache)
9   [BCACHE] @ 0 DSPBLK 16 1024 * <DISK T;
10
11
12
13
14
15

```

9-26-85) (GAME OPTIONS - DATASAVE

9-26-85)

```

HEAD: DATASAVE ( -- \ save game image data to disk)
T: .SAVING ( give user message)
  CKSUM OFF ( initialize checksum)
  TME>DISK ( timestamp to disk)
  KRN>DISK ( save kernal thru ova)
  DSP>DISK ( save display image)
  ARR>DISK ( save data arrays to disk)
  PRM>DISK ( save parameter stack to disk)
  RTN>DISK ( save return stack to disk)
  CKSUM @ 359 BLOCK 1014 + ! ( save checksum)
  0 359 BLOCK 1008 + ! ( clear signature)
  UPDATE SAVE-BUFFERS T;

```

41

42

45

0 (GAME OPTIONS - SETSAVE CLRSAVE
 1
 2 HEAD: SETSAVE (-- \ set game-saved flag)
 3 T: 349 BLOCK 1016 + ON FLUSH T;
 4
 5 HEAD: CLRSAVE (-- \ clear game-saved flag)
 6 T: 314 BLOCK 1016 + OFF T;
 7
 8
 9
 10
 11
 12
 13
 14
 15

9-26-85) (GAME OPTIONS - SAVEGAME

9-26-85)

HEAD: SAVEGAME (-- \ save current game state)
 T: >DISPLAY DARK
 BORDER (draw border & logo)
 .SAVEHEAD (draw heading)
 SETSAVE (set game saved flag)
 INSERTBLANK (load blank disk for storage)
 DATASAVE (save data to disk)
 DONE T; (exit to system)

43

46

0 (GAME OPTIONS - DATALOAD
 1
 2 HEAD: DATALOAD (-- \ load game image data to memory)
 3 T: .LOADING (give user message)
 4 CKSUM OFF (initialize checksum)
 5 EMPTY-BUFFERS
 6 DISK>KRN (load kernal thru ova)
 7 DISK>DSP (load display image)
 8 DISK>ARR (load data arrays)
 9 DISK>PRM (load parameter stack)
 10 DISK>RTN (load return stack to RTNTEMP)
 11 CKSUM @ 359 BLOCK 1014 + @ = (cmp check sum)
 12 IF .LOADED
 13 ELSE .CKSUMERROR
 14 THEN T;
 15

9-26-85) (GAME OPTIONS - .CONFIGHEAD MONITORTYPE

9-26-85)

HEAD: .CONFIGHEAD (-- \ display configure game header message)
 T: >3FONT 34 171 POS.
 ." CONFIGURE" 10 XBLT +! ." SYSTEM" >0FONT T;
 HEAD: MONITORTYPE (-- \ display monitor type message and get)
 T: (user choice)
 >2FONT ERASEWIND ERASELINE
 37 17 POS. ." SELECT MONITOR TYPE"
 40 120 POS. ." 1. BLACK,WHITE"
 40 100 POS. ." 2. RGB"
 40 80 POS. ." 3. COLOR TV OR"
 38 70 POS. ." COMPOSITE"
 BEGIN 'KEY ASCII 1 ASCII 4 WITHIN UNTIL
 LKEY @ ASCII 0 - MONITOR ! >LDRES T;

44

47

0 (GAME OPTIONS - LOADGAME
 1
 2 : LOADGAME (-- \ load game state from save disk assumes game)
 3 (disk is loaded)
 4 >DISPLAY DARK BORDER (draw border & logo)
 5 .LOADHEAD (draw heading)
 6 EMPTY-BUFFERS OFFCACHE
 7 359 BLOCK 1012 + @ TIMESTAMP ! (restore time id#)
 8 SAVECHK (check if saved)
 9 INSERTLOAD (load save disk)
 10 DATALOAD (load data from disk)
 11 .INSERTGAME CLRSAVE (insert the game disk)
 12 TIMESTAMP ON TME>DISK (mark disk in play)
 13 .KEYTORESUME SET-CURRENT ("press key to resume")
 14 RESUME ON
 15 RTNTEMP RETJMP ; (continue where saved)

9-26-85) (GAME OPTIONS - MEMORYSIZE

9-26-85)

HEAD: MEMORYSIZE
 T: (-- \ display memory size options, get user)
 (choice and set memory up)
 >2FONT ERASEWIND ERASELINE
 39 17 POS. ." SELECT MEMORY SIZE"
 27 120 POS. ." 1. 128 K 6. 448 K"
 27 108 POS. ." 2. 192 K 7. 512 K"
 27 96 POS. ." 3. 256 K 8. 576 K"
 27 84 POS. ." 4. 320 K 9. 640 K"
 27 72 POS. ." 5. 384 K "
 BEGIN 'KEY ASCII 1 ASCII 4 WITHIN UNTIL
 LKEY @ ASCII 1 - 64 * 127 + NSYSK ! T;

48

51

```

0 ( GAME OPTIONS - CONFIGAME                                7-26-85) ( SAVEGAME - GAMEOPCASES RESUMEGAME          9-26-85)
1
2 : CONFIGAME ( -- \ configure game system params)           HEAD: RESUMEGAME ( -- \ resume game in progress)
3 >DISPLAY DARK BORDER      ( draw border & logo)           T: RESUME ON T;
4 .CONFIGHEAD                ( draw heading)
5 MONITORTYPE                ( select monitor type)         CASE GAMEOPCASES ( n -- \ execute game option chosen)
6 MEMORYSIZE                 ( select memory size)           ASCII 1 IS SAVEGAME
7 NSYSK @ SYSK @ = NOT TIMESTAMP @ NOT OR                   ASCII 2 IS RESUMEGAME
8 ( configure if 1st time or mem size change)                ASCII 3 IS CONFIGAME
9 IF TIMESTAMP @ IF INSERTBLANK ARR>DISK THEN                OTHERS UNRAVEL
10     NSYSK @ SYSK ! CONFIGURE-SYSTEM
11     TIMESTAMP @ IF DISK>ARR .INSERTGAME THEN
12 THEN
13 TIMESTAMP @ NOT IF TIMESTAMP ON TME>DISK THEN ;
14
15

```

```

0  PUSH XCHG VECT ( -- \ 49 PUSH CURRENT VECTS EXCHANGE VECTORS WITH )
1  (BACKUPS)
2
3
4
5
6
7
8
9
10
11
12
13
14
15

```

50

53

```
0 ( SAVEGAME - GET-OPTION#                               9-26-85) ( GAME OPTIONS - GAMEOPTIONS                                9-26-85)
```

```
1
```

```
2 HEAD: GET-OPTION# ( -- c \ get user menu choice ascii 1-3)    HEAD: GAMEOPTIONS ( -- \ do game options procedure)
```

```
3 T; BEGIN                                                    T: RESUME OFF                      ( clear resume flag)
```

```
4     'KEY                                                    BEGIN
```

```
5     ASCII 1 ASCII 4 WITHIN                                  .GAMEOPS                          ( display game option menu)
```

```
6     UNTIL LKEY @ T;                                         GET-OPTION#                       ( get user choice)
```

```
7                                                         GAMEOPCASES                        ( execute choice)
```

```
8                                                         RESUME @                           ( check for resume)
```

```
9                                                         UNTIL T;
```

```
10
```

```
11
```

```
12
```

```
13
```

```
14
```

```
15
```

54

57

```

0 ( GAME OPTIONS - <GAMEOPTIONS          9-26-85) ( GAME-OV ----- OVERLAY SUFFIX ----- 9-26-85)
1                                     TIMESTAMP OFF      ( initialize game Play flag)
2 HEAD: <GAMEOPTIONS                     DISPOSE
3 T: ( -- \ do clean up for game options procedure) CLOSE-OVERLAY
4   RESTOREDISPLAY      ( move picture from cache to display) 120 OVERLAY GAME-OV
5   ONCACHE             ( turn cache memory on)              GAME-OV
6   SET-CURRENT         ( make current instance memory resident) FORTH DEFINITIONS
7   ESCENABLE T;        ( turn on escape path) : LOADGAME ( -- )      GAME-OV GAME LOADGAME ;
8                                     : CONFIGAME ( -- )      GAME-OV GAME CONFIGAME ;
9   REST BP VEST : GAMEOPS ( -- )      GAME-OV GAME GAMEOPS ;
10
11                                     : GAMEOPM ' GAMEOPS MODULE ;
12   OV-CANCEL
13   ' GAMEOPM ESC-PFA ! ( patch esc key link)
14   ESC-EN ON          ( turn on path to game save)
15   359 BLOCK 1012 + OFF ( initialize game play flag)

```

55

```

0 ( GAME OPTIONS - GAMEOPS          7-5-85)
1
2 : GAMEOPS ( -- \ top level game option word)
3   >GAMEOPTIONS
4   GAMEOPTIONS
5   <GAMEOPTIONS ;
6
7
8
9
10
11
12
13
14
15

```

56

```

0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

```