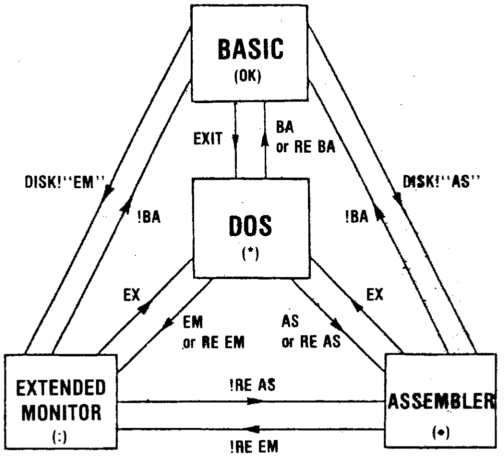
ASCII CODES

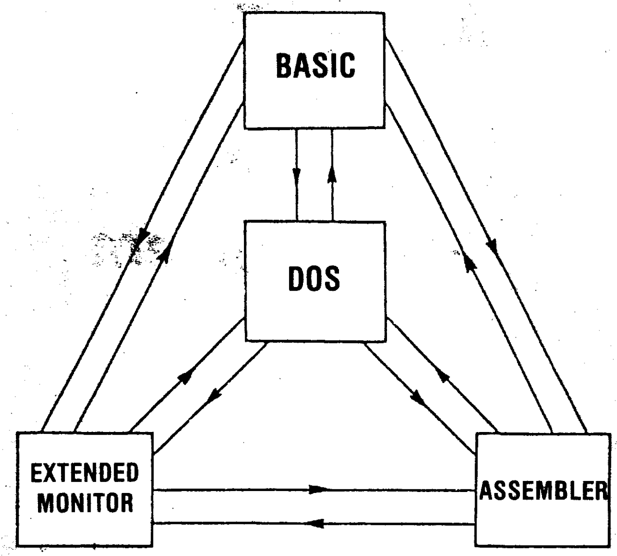
CODE	CHAR	CODE	CHAR	CODE	CHAR
00	NUL	2B	+	56	V
01	SOH	2C	,	57	W
02	STX	2D	-	58	X
03	ETX	2E	.	59	Y
04	EOT	2F	/	5A	Z
05	ENQ	30	0	5B	[
06	ACK	31	1	5C	\
07	BEL	32	2	5D	]
08	BS	33	3	5E	∧
09	HT	34	4	5F	—
0A	LF	35	5	60	
0B	VT	36	6	61	a
0C	FF	37	7	62	b
0D	CR	38	8	63	c
0E	SO	39	9	64	d
0F	SI	3A	:	65	e
10	DLE	3B	;	66	f
11	DC1	3C	<	67	g
12	DC2	3D	=	68	h
13	DC3	3E	>	69	i
14	DC4	3F	?	6A	j
15	NAK	40	@	6B	k
16	SYN	41	A	6C	l
17	ETB	42	B	6D	m
18	CAN	43	C	6E	n
19	EM	44	D	6F	o
1A	SUB	45	E	70	p
1B	ESC	46	F	71	q
1C	FS	47	G	72	r
1D	GS	48	H	73	s
1E	RS	49	I	74	t
1F	US	4A	J	75	u
20	SP	4B	K	76	v
21	!	4C	L	77	w
22	”	4D	M	78	x
23	#	4E	N	79	y
24	$	4F	O	7A	z
25	%	50	P	7B	{
26	&	51	Q	7C	}
27	’	52	R	7D	:
28	(	53	S	7E	÷
29	)	54	T	7F	DEL
2A	*	55	U		



Transfer of Control Commands

OS-65D

DISK OPERATING SYSTEM



DOS and BASIC

QUICK REFERENCE

TO START YOUR COMPUTER

Check to make sure no diskettes are in the disk drives!! Lock the SHIFT LOCK or ALL CAPS key.

1. Turn on the computer, disk drives and terminals -switches are generally located on the back of the device cabinet.
2. Place an OS-65D disk in drive A (the drive whose red light is on or the top drive in dual drive cabinets). Close the disk drive door.
3. Depress the BREAK key on C1P and C4P systems (and hold for a few seconds). Depress the white reset button on C8P and serial systems.
4. When the "H/D/M?" ("D/C/W/M?" on C1P systems) message appears, respond by typing "D". In a few seconds a menu should appear on the screen.
5. To enter the BASIC immediate mode, respond UNLOCK to this menu in OS-65D V3.2; select option 9 in OS-65D V3.3.

65D BASIC

The entries are organized alphabetically according to Keywords used. Each entry consists of the general syntax, examples where appropriate, and a brief description.

The following notation is used:

[n]	see page n of the OS-65D Tutorial and Reference Manual
{n}	see page n of the OSI BASIC Reference Manual
(*)	cannot be used in the immediate (direct) mode; must be used with a program statement number.
(**)	can only be used in the immediate (direct) mode; must not be used within a program.
(2)	not available under OS-65D V3.3.
(3)	available only under OS-65D V3.3.
ae	a numeric constant or arithmetic expression (see {3})
re	a logical constant or relational expression (see {4})
se	a string constant or expression (see {4})
dos	a 65D Disk Operating System (DOS) command.
e	a constant or expression.
V	a variable
c	a constant
nv	a numeric variable
iv	an integer variable
sv	a string variable
niv	a numeric variable or integer variable
rae	a relational expression or arithmtic expression
FILE	a disk file name
loc	a memory location address
sn	a program statement number
dev	an OS-65D device number. [54]

ABS
ABS(ae)
A function. Returns the absolute value of its argument. {19}

AND
re AND re
  IF X < 15 AND X > = Ø THEN 1ØØ
A bitwise Boolean AND operator . re AND re will be TRUE only when both of the operands have the value TRUE. {4}

ASC
ASC(se)
  ASC(X$)   ASC("BIG")
A function. Returns the ASCII value in decimal of the first character in the argument {20}

ATN
ATN(ae)   (-1< ae< 1)
  ATN(Ø.431)
A function. Returns the arctangent of the argument {20} (2) [188]

CHR$
CHR$(ae)   (Ø ≤ae≤ 255)
  CHR$(66)
A function. Returns the character whose decimal ASCII value is the greatest integer less than or equal to the argument. {21}.

CLEAR
CLEAR
Clears the program variable table and restores the data pointer (*) {17}

CLOSE
DISK CLOSE, dev (dev =6 or dev =7)
Closes a disk file that has been previously opened. {28}, [15]

CONT
CONT
Restarts a program whose execution has been interrupted by a STOP or END statement or a CTRL-C. {15} (**)

COS
COS(ae)
A function. Returns the cosine of the argument. {20}

DATA
DATA c, c, c, ...
  DATA 1.7, "BIG", 173, -812
Establishes a list of constants to be input by the program via the READ statement {6}

DEF FN
DEF FNnv(nv) = ae
  DEF FNA(X) = X*7+3
Defines a single variable function for future use within the program segment {23} (*)

DIM
DIM v(ae, ae,...),...
  DIM A(20), B$(6,7)
Declares the variables specified to be subscripted. {18}

DISK!
DISK! "dos"
  DISK! "IO 5,6"
  DISK! "LOAD FILE"
Permits 65D DOS commands to be used within a BASIC program. [202]

DISK CLOSE see CLOSE

DISK FIND See FIND

DISK GET See GET

DISK OPEN See OPEN

DISP PUT See PUT

EDIT
EDIT sn
EDIT 1ØØ
Returns line sn for editing. The short form is !sn. (**)[71] (3)

END
END
Terminates program execution {13}

EXIT
EXIT
Transfers control to the DOS mode {28} [53]

EXP
EXP(ae)   ae< 88.Ø29619
  EXP(41.662)
A function. Returns e = 2.71828...raised to the power equal to the value of the argument. {19}

FIND
DISK FIND, se
  DISK FIND, "BIG"
Beginning at current file pointer location, the data file is searched for the specified string, the pointer is set to the end of the field in which it is found. An unsuccessful search results in a #D error. [96] (3)

FN
See DEF FN

FOR
FOR nv = ae TO ae
FOR nv = ae TO ae STEP ae
  FOR X = 15 TO 45 STEP 5
Opens program loop. End of the loop is indicated by the statement NEXT or NEXT niv. STEP is used to define an increment other than 1 for niv for each iteration of the loop. In the example, the loop is executed 7 times {12}

FRE
FRE(X)   X is a dummy variable
A function. Returns the number of bytes of memory in the workspace that are unused. Save the program before using FRE. {17}

GET
DISK GET, niv
  DISK GET, 15
Brings the record numbered niv from the disk to buffer #6 and sets the I/O pointers to the beginning of the record {28} [17]

GOSUB
GOSUB sn
  GOSUB 1000
Program control is transferred to statement number sn. When the statement RETURN is encountered, control goes back to the statement following sn {23}

GOTO
GOTO sn
  GOTO 1000
Program control is transferred to statement number sn. {11}

IF
IF rae GOTO sn
IF rae THEN sn
If the value of rae is TRUE (arithmetic expressions are considered to be TRUE if they have a value other than 0) program control is transferred to statement sn.
IF rae THEN S   (S is a program statement)
If the value of rae is TRUE, statement S is executed {11}

INPUT
INPUT V, V, ...
INPUT X, Y, A$
Prompts for keyboard input to the specified variables {6} {*}

INPUT#
INPUT#dev, V, V, ...
  INPUT #6, A, B, Q$
Input is from device number dev to the specified variables. {9} [13] (*)

INT
INT (ae)
  INT (16.8)
A function. Returns the greatest integer less than or equal to the argument {19}

LEFT$
LEFT$(se, ae)   ae > 0
  LEFT$("ABCDE", 3)
A function. Truncates ae to an integer and returns that leftmost number of characters from string se. In the example, "ABC" is returned. {21}

LEN
LEN(se)
  LEN(A$)
A function. Returns the length of the string se {21}

LET
LET V = e
  LET A$ = "BIG"
Assignment statement. Keyword LET is optional. {6}

LIST
LIST
LIST sn-sn
    LIST 100-200
    LIST - 1000
    LIST 200
Lists the program in the workspace between the two specified statement numbers. If the first (second) statement number is omitted, the default is the beginning (end) of the program. {15}

LIST#
LIST#dev
  LIST#4
Same as LIST, but the listing is sent to device number dev. {9, 15} [54]

LOG
LOG(ae)   ae > 0
  LOG14.8
A function. Returns the natural logarithm (log to the base e) of the argument. {19}

MID$
MID$(se, ae, ae) first ae > 0, second ae ≥ 0
  MID$("ABCDEFG", 2, 3)
A function. In the example, A string of length 3 starting at position 2 is returned; i.e. "BCD". If the second ae is omitted, the string returned goes to the end of se. {21}

NEW
NEW
Clears the workspace to prepare for creation of a new program {15}

NEXT
see FOR

NOT
NOT re
  NOT (A > 5)
A bitwise Boolean NOT operator. Reverses the truth value of the operand re. {3}

NULL
NULL iv   0 ≤ iv ≤ 8
Inserts iv zeros at the beginning of each line as it is stored on tape. {27} {2}

ON
ON ae GOTO sn, sn,...
ON ae GOSUB sn, sn,...
  ON X + 7 GOTO 100, 200
Depending upon the value of ae (truncated to an integer) program control passes to the ae-th statement in the list of statement numbers {12, 24}

OPEN
DISK OPEN, dev, "FILE"   (dev = 6 or 7)
Opens the disk file FILE for sequential (dev = 6 or 7) or random access (dev = 6 only) {28} [15]

OR
re OR re
  IF A > 5 OR A < 2 THEN 100
A bitwise Boolean OR operator. re OR re is FALSE only when both of the operands are FALSE. {3}

PEEK
PEEK(loc)
A function. Returns the value stored in memory location loc {25}

POKE
POKE loc, ae   ae is an integer.
  POKE 11686, 17
The value ae is stored in memory location loc {25}

POS
POS(X)   X is a dummy variable.
A function. In or following a PRINT statement, returns the current position (between 0 and 132) of the cursor {9}

PRINT
PRINT e, e,...
  PRINT A, B$; C$
Outputs the values stored in the list of expressions. The keyword PRINT can be replaced by a question mark. {7}

PRINT#
PRINT#dev, e, e,...
Same as PRINT, but output is directed to device number dev instead of the screen. {7} [13] [54]

PRINT!
PRINT!(HOC), e, e,...   (HOC = Hazeltine Operation code-see [223])
PRINT!(28) X$, A, B, C
Depending on the value of HOC, certain screen characteristics and cursor positons are selected before beginning output of expression values; emulates certain Hazeltine terminal capabilities. [223] (3)

PRINT CHR$
see CHR$

PRINT&

PRINT&(X, Y), e, e,...
PRINT&(10, 20) A, B$
Moves the screen cursor to screen position (X, Y) (0, 0 = upper left corner) before beginning output of expression values. Identical to:
PRINT!(17,X,Y), e, e,... [79] (3)

PRINT USING

PRINT USING se ae, ae, ...
PRINT USING "####.##" 6.87304
Used to format numeric output; se must be a string expression made up of a decimal point and/or #'s. In the example the output format specified results in printing 6.87 (with three leading blanks) [73] (3)

PUT

DISK PUT
Follows a previous DISK GET; places the current record back to the disk. {28} [17]

READ

READ V, V, V,...
READ A, B$, C
Inputs constants that are specified by DATA statements in the same program into the specified variables {6} (*)

REM

REM any remark
REM THIS IS A TEST PROGRAM
Used for program documentation. Everything appearing after REM is ignored on execution of that line {16}

RESTORE

RESTORE
Resets the pointer in a program's DATA list to the first item. {7}

RETURN

See GOSUB

RIGHT$

RIGHT$(se, ae) ae > 0
RIGHT$("ABCDEF",2)
A function. Truncates ae to an integer and returns that number of rightmost characters. In the example, "EF" is returned. {21}

RND

RND(ae)
RND(-16)
A function. Returns a number between 0 and 1. Can be used repeatedly to generate a sequence of pseudo-random values. If ae > 0, the argument is a dummy argument. If ae = 0, RND returns the previous value again. If ae < 0, ae functions as a "seed" and RND starts a new sequence. The sequence repeats after a certain period determined by the seed. {19}

RUN

RUN
Starts execution of the program in the workspace at the first statement.
RUN sn
Starts execution of the program in the workspace at statement number sn.
RUN "FILE"
Leads the program from disk file FILE and starts execution.
RUN "TT" (TT = a disk track number)
Loads the program from the disk file beginning at track TT and starts execution. {15}

SGN

SGN(ae)
A function. Returns +1 if ae > 0, 0 if ae = 0, -1 if ae < 0 {19}

SIN

SIN(ae)
A function. Returns the value of the sine of the argument ae. {20}

SPC

SPC(ae)
PRINT "A"; SPC(5); "B"
A function. Used to print ae spaces in a PRINT sequence {9}

SQR

SQR(ae) ae ≤ 0
A function. Returns the square root of the argument ae. {20}

STEP

See FOR

STOP

STOP
Halts execution of a program and prints a BREAK message indicating the statement number of the STOP statement {13}

STR$

STR$(ae)
STR$(6.71)
A function. Returns the value of the argument ae as a string. {21}

TAB

TAB(ae) ae is an integer
TAB(10)
A function. Used in a PRINT statement to move the print position for the next character to position ae + 1 on the print line. {8}

TAN

TAN(ae)
A function. Returns the tangent of the argument. {20}

THEN

See IF

TO

See FOR

TRAP

TRAP sn
If an error is encountered in a program after this statement, then control transfers to statement sn.
TRAP 0 disables error trapping. [71] (3)

USR

USR(ae)
Y = USR(X)
Transfers control to a machine language routine at a location determined previously by appropriate POKES. ae may be an input parameter (and USR(ae) an output parameter) or ae may be a dummy parameter. {34}

VAL

VAL(se)
VAL("6.31")
A function. It is the opposite of STR$; returns the numeric value of the string expression se if se represents a number. Otherwise, 0 is returned.

WAIT

WAIT loc, J 0≤J≤255
Halts program execution, Reads the contents of location loc and AND's the result (bitwise) until a nonzero result is obtained, then resumes program execution.
WAIT loc, J, K 0≤J, K≤255
Halts program execution, reads the contents of location loc, exclusive OR's that value (bitwise) with K, and then AND's the result with J until a nonzero result is obtained; then resumes execution {25} {2}

SPECIAL V3.3 COMMANDS

Screen Display Commands:

(ESC) 1 Clears screen; homes cursor to upper left; produces "wide character" display (32x32 on C4P and C8P machines; 24x24 on C1P)

(ESC) 2 Clear screen; homes cursor; produces "narrow character" display
(32x64 on C4P and C8P machines; 12x48 on C1P)

(ESC) 3 Homes cursor to upper left

(ESC) 4 Clears to end of screen (memory of workspace is not altered)

(ESC) 5 Moves cursor up one line

(ESC) 6 Moves cursor down one line

(ESC) 7 Inserts line (lower lines scroll down)

(ESC) 8 Clears line (memory of workspace is not altered)

(ESC) 9 Turns color off

(ESC) 10 Turns color on

PRINT Statement Commands

(These commands must be used in PRINT statements)

Display Size

!(20) Selects "wide character" display (32 x 32 on C4P and C8P, 12 x 14 on C1P), clears screen and homes cursor to upper left screen corner.

!(21) Selects "narrow character" display (32 x 64 on C4P and C8P, 12 x 48 on C1P), clears screen, and homes cursor to upper left screen corner.

!(22, w, h) Selects print window w characters wide and h characters high. Upper left window corner is at current cursor position; screen is not cleared.

Cursor Control

Single Step

CHR$(8) Back one space.
CHR$(16) Forward one space.
!(12) Up one space.
!(11) Down one space.
CHR$(10) Down one space.

Multistep

CHR$(13) Back to front of line.
CHR$(14) Forward to next eight space tab set (seven space for left-most field).

Anywhere

!(17, x, y) Relocate to x, y (0 0 is upper left corner).
&(x, y) Relocate to x, y (00 is upper left corner).

Home

!(18) Relocate to 0, 0 (upper left corner).

Insert

!(26) Inserts line at cursor position; lower lines scroll down.

Clear

Line

!(15) Clears from cursor to end of line.
!(19) Clears entire line (lower lines move up).

Screen

!(24) Clears from cursor to end (lower right) of window

!(28) Clears entire screen **and** homes cursor in window.

Color

Color Select

!(1) Selects color 0 as cell background.
!(25) Selects normal black/white display mode (i.e., black background, white character).
!(31, n) Selects color n as cell background.

Color Change

!(2, n, m) Changes all displayed cells of background color m to background color n.

!(29, n) Clears all displayed cells of background color n (i.e., cell background is changed to black and character is replaced with a blank).

Cursor Sensing

!(5) Sends information for current cursor position x, y, to string variable in following INPUT statement. Information is in the form of two characters for which (x + 65) is the ASCII code. Line feed follows the INPUT statement used with !(5).

!(33) Sends character at cursor position to string variable in following INPUT statement. Line feed follows the INPUT statement used with !(33).

Printer Control

!(67,FL) Initialize Epson Printer Drivers; set form length.

!(80) Send Video Screen to Epson Printer

**** Note to Users of Serial Systems ****

OS-65D V3.3 is only partially compatible with serial systems. If you are using a Hazeltine 1420 terminal, be sure switch 6 is set to the ESC position. Certain features that refer to color, screen size, or windowing are not operable on serial systems. Specifically,

1) The commands that use the ESC key are not operable.
2) The destructive backspace key is <DEL> instead of <SHIFT/O> or <RUB OUT> and the line delete is <@> instead of <SHIFT/P>.
3) The PRINT command !(26) inserts a line but not at the cursor position. The line always starts at the left margin.
4) The following PRINT commands should not be used:

!(1)	!(21)	!(29,n)	!(20)
!(2,n,m)	!(22,w,1)	!(31,n)	!(67,FL)
!(5)	!(25)	!(33)	!(80)
!(20)	!(28)		

V3.3 EDITOR COMMANDS

(CTRL)H Moves cursor one space to the left (non-destructively)

(CTRL)P Moves cursor one space to the right (non-destructively)

(CTRL)F Moves cursor to the front of the line

(CTRL)R Moves cursor to the rear of the line

(CTRL)I Moves the cursor (non-destructively) forward to the next tab position
(i.e., positions 1, 8, 15, 22, 29, 36, 43, 50 57, 64, 71)

(CTRL)T Retypes the line currently being edited (in its present edited form)

(SHIFT)P Clears screen of line currently being edited leaving the line in workspace as it was before calling it to be edited

(RUBOUT) Deletes the character flashing with the cursor. Line closes up from the right.

EDITnn or !nn Calls line number nn for editing

EDIT or ! Calls next line in program for editing

EDIT! or !! Recalls last edited line for re-editing

INDIRECT FILES

To merge two BASIC programs using indirect files:

1) determine the starting page number N of the indirect file,

2) load one program into the workspace,

3) move this program to the indirect file,

4) load the second program into the workspace,

5) move the first program back from the indirect file to the workspace.

If each of the programs has a line with the same number the line in the first program will be the one that appears in the merged program.

STARTING PAGE NUMBER OF INDIRECT FILE

The starting page number N of an indirect file can usually be set at 128 in OS-65D. If the program is quite large this value may not work. The indirect file must fit into memory above the program in the workspace. A value for N is given by:

N = highest page in memory—pages unused in memory

the highest page in memory can be obtained by:

?PEEK(133)

and the number of pages unused in memory can be obtained by

?INT(FRE(X)/256), or

if FRE(X) is negative, by:

?INT((65536 + FRE(X))/256)

The starting page of the workspace is approximately

page 50 (317E) for OS-65D V3.2 on an 8 inch disk.

page 51 (327E) for OS-65D V3.2 on a 5 inch disk.

page 59 (3A7E) for V3.3 systems

(see p. 49 of 65D Reference Manual)

The number of pages used by the program is:

highest page — starting page — pages left.

If the number of pages used exceeds the number of pages left there is not enough memory available to put this program in an indirect file.

FROM WORKSPACE TO INDIRECT FILE

To move a program from the workspace to an indirect file:

1) enable the indirect file function with the following POKES, where N is the starting page number.

POKE 9554,N

2) LIST the program between square brackets as follows: With the program in the workspace. type

LIST[<RETURN>

(wait for listing to end)

< SHIFT/M > < @ > < RETURN>

If the keyboard is a polled keyboard use these commands instead:

LIST <SHIFT/K> <RETURN>

(wait for listing to end)

< SHIFT/M > < @ > < RETURN >

The first bracket "[" , <SHIFT/K> will not appear on the video screen. The second bracket appears twice as "]]"

If the end of the listing appears garbled the indirect file was not placed high enough in memory and the end of the program in the workspace has been overwritten.

FROM INDIRECT FILE TO WORKSPACE

To move a program from an indirect file to the workspace:

1) enter the appropriate POKEs, where N is the starting page number of the indirect file

POKE 9368,N

2) enter the command:

< CTRL/X > <RETURN>

A listing of the program in the indirect file will appear ending with the bracket closure "]]". On some systems there will be a harmless error message before or after the listing. To see the contents of the workspace enter the command LIST.

MOVING PROGRAMS BETWEEN INCOMPATIBLE DISKS

To transfer a program between incompatible disks:

1) determine the starting page number N of the indirect file,

2) boot up BASIC and load the program into the workspace,

3) move the program to the indirect file using the POKEs for the system on this disk,

4) boot up BASIC on the other disk; clear the workspace with NEW,

5) move the program from the indirect file to the workspace using the POKEs for the system on this new disk,

6) PUT the program on the new disk.

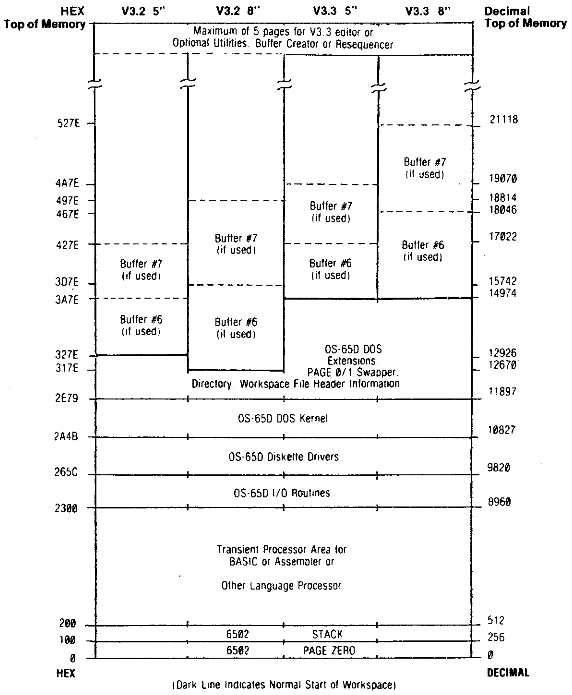
(for additional details. see chapter 12 of the BASIC Reference Manual)

UTILITY PROGRAMS

A brief description of the utility program supplied with the OS-65D system (operating system restrictions are in parenthesis).

ASAMPL	- Sample Assembly language program
ATNENB	- Enables or disables arc tangent and print extensions (V3.3 only)
BEXEC*	- Program which is run upon boot-up; displays menu.
BUFFER	- Check the size of program buffers; add and delte buffers. (V3.3 only - Disk 2)
CHANGE	- Permits adjustment of the following: - Terminal width for BASIC. - The highest page of memory available which is what BASIC and ASM use when loaded. - The adjustment of the workspace limits for BASIC. The result is an empty workspace to the user specifications.
COLORS	- Color adjustment program.
COMPAR	- Utility for comparing diskettes. (V3.3 only)
COPIER	- Utility for copying diskettes. (V3.3 only)
CREATE	- Enter a file name into the directory and zero out the created file on disk.
DATRAN	- copy data files. (V3.3 only - Disk 2)
DELETE	- Remove a file name from directory.
DIR	- Print unsorted disk directory.
DISASM	- Generate an assembly language listing for machine code program. (V3.3 only - Disk 2)
GSOSRT	- Sort data files, including MDMS master files. (V3.3 only - Disk 2)
MODEM	- Sets up a machine code modem routine for use with a standard RS-232 modem. (V3.3 only).
RANLST	- General random access file list utility.
RENAME	- Rename a file name in directory.
REPACK	- Remove REM statements and blank spaces from BASIC program. (V3.3 only - Disk 2)
RSEQ	- Change the numbering of statements in a BASIC program. (V3.3 only - Disk 2)
SECDIR	- Print a sector map directory of disk.
SEQLST	- General sequential file list utility.
TRACE	- Enable or disable statement number trace feature.
ZERO	- Initialize contents of a data file to zeros.

SYSTEM MEMORY MAPS



(Dark Line Indicates Normal Start of Workspace)

POKE AND PEEK LIST

As systems develop, different locations are committed to hold parameters. Many of these parameters have been mentioned in the text material. These parameters are collected here, along with some other useful parameters which may be needed by an advanced programmer. Users of the video systems and systems that include certain options and accessories (e.g., Home Security, Remote Control, High Resolution Graphics, etc.) may need to POKE or PEEK other parameter locations.These locations are fully documented in the appropriate User's Manuals. CAUTION: Care must be taken when POKEing any of these locations to avoid system errors.

LOCATION		CONTENTS	
DECIMAL	HEX	(DEC)	COMMENTS
23	17	132	Terminal width (number of printer characters per line). The default value is 132. Note, this is not to be confused with the video display width (64 characters).
24	18	112	Determines the number of (14 character) output fields in a terminal output line when outputting BASIC variables separated by commas. As long as the contents of this location exceeds the current terminal output position, the terminal output line will continue with a tab to the start of the next output field.
120- 121	78— 79	127 5Ø	Lo-Hi byte address of the beginning of BASIC work space (note 127 = $7F, 5Ø = $32). Normal contents of Location 121 is 58 on V.3.3 and 49 on Serial Systems.
741	2E5	1Ø	Control location for "LIST." Enable with a 76, disable with a 10.
750	2EE	1Ø	Control location for "NEW." Enable with a 78, disable with a 1Ø.
1797	7Ø5	32	Controls line number listing of BASIC programs, enable with a 32, disable with a 44.
2Ø73	819	173	"CONTROL C" termination of BASIC programs. Enable with 173, disable with 96.
22ØØ	898		Track Ø (Load address.)
2888	B48	27	A 27 present here allows any null input (carriage return only) to force into immediate jumping out of the program. Disable this with a Ø. Location 8722 must also be set to Ø.
2893	B4D	55	Alternate "break on null input" enable/disable location. (see 2894)
2894	B4E	Ø8	A null input will produce a "REDO FROM START" message when 2893 and 2894 are POKEd with 28 and 11 respectively.
2972	B9C	58	Normally a comma is a string input termination. This may be disabled with a 13 (see 2976).
2976	BAØ	44	A colon is also a string input terminator, this is disabled with a 13 (see 2972, 9976)
87Ø8	22Ø4	41	Output flag for peripheral devices.
8722	2212	27	Null input if = ØØ, normal input if = 27.
89Ø2	22C6	ØØ	Determines which registers (less 1) RTMON scans (HC systems only).
8917	22D5	—	USR (X) Disk Operation Code: Ø-write to Drive A 3-read from Drive A 6-write to Drive B 9-read from Drive B
8954	22FA	—	Location of JSR to a USR function. Present to JSR $22D4, i.e., set up for USR (X) Disk Operation.
896Ø	23ØØ	—	Has page number of highest RAM location found on OS-65D's cold start boot in. This is the default high memory address for the assembler and BASIC.
8993	2321	—	I/O Distributor INPUT flag
8994	2322	—	1/Ø Distributor OUTPUT flag
8995	2323	—	Index to current ACIA on 550 board. If numbered from 1 to 15 the value POKEd here is a 2 times the ACIA number.
8996	2324	—	Location of a random number seed. This location is constantly incremented during keyboard polling.
(Note: Locations 8998 through 9005, 9132-9133, and 9155-9156 are used for Disk Buffer #6 (I/O Flag Bit 5 device) usage parameters.)			
8998- 8999	2326- 2327	126 *	LO-HI byte address for the start of Buffer #6 (*contents vary: 58 on all V3.3; 50 on 5" V3.2; 49 on 8" V3.2)
9ØØØ- 9ØØ1	2328- 2329	126 *	LO-HI byte address for the end of Buffer #6 (*contents vary: 66 for 5" V3.3; 70 for 8" V3.3; 58 for 5" V3.2; 61 for 8" V3.2)
9ØØ2	232A	—	First track of Buffer #6 File (BCD)
9ØØ3	232B	—	Last track of Buffer #6 File (BCD)
9ØØ4	232C	—	Current rack in Buffer #6 (BCD)
9ØØ5	232D	—	Buffer #6 Dirty Flag (if contents is non-zero, then data has been written to the buffer, but has not yet been transferred to the disk)
(Note: Locations 9006 through 9013, 9213-9214, 9238-9239 are used for Disk Buffer #7 (I/O Flag Bit 6 device) usage parameters)			
9ØØ6- 9ØØ7	232E- 232F	126 *	LO-HI Byte address for the start of Buffer #7 (*contents vary: 58 on 5" 3.2; 61 on 8" V3.2; 66 on 5" V3.3; 70 on 8" V3.3)
9ØØ8- 9ØØ9	233Ø 2331	126 *	LO-HI Byte address for the end of Buffer #7 (*contents vary: 66 on 5" V3.2; 73 on 8" V3.2; 74 on 5" V3.3; 82 on 8" V3.3)
9Ø1Ø	2332	—	First track of Buffer #7 File (BCD)
9Ø11	2333	—	Last track of Buffer #7 File (BCD)
9Ø12	2334	—	Current rrack in Buffer #7 (BDC)
9Ø13	2335	—	Buffer #7 Dirty Flag (Ø = Clean; see comment for location 9ØØ5)
9Ø98- 9Ø99	238A- 238B	— —	Pointer to Memory Storage Input (Lo and Hi Byte).
91Ø5- 91Ø6	2391- 2392	— —	Pointer to Memory Storage Output (Lo and Hi Byte).
9132- 9133	23AC- 23AD	126 *	LO-HI Byte address of Buffer #6 current input. (* 50 on 5" V3.2; 49 on all other systems)
9155- 9156	23C3- 23C4	126 *	LO-HI Byte address of Buffer #6 current output. (*50 on 5" V3.2; 49 on all other systems)
9213- 9214	23FD- 23FE	126 *	LO-HI Byte address of Buffer #7 current input. (*62 on 5" V3.2; 61 on all other systems)
9238- 9239	2416- 2417	126 *	LO-HI Byte address of Buffer #7 current output. (*62 on 5" V3.2; 61 on all other systems)
9368	2498	—	Indirect File Input Address (Hi Byte) (Lo = ØØ)
9554	2552	—	Pointer to Indirect File (Hi Byte only) for output (Lo = ØØ)
9682- 9683	25D2- 25D3	— —	Next Position for Cursor on video screen (HI and LO Bytes) V3.2 Video Systems only)
977Ø	262A	64	Display control parameters. Single Space = 64; Double Space = 128; (V3.2 Video Systems only)
9796	2644	—	Entry point to Keyboard Swap Routine
9822	265D	—	Sector for USR(X) on Disk.
9823	265F	—	Page Count for USR(X). Read or Write.
9824	266Ø	—	Pointer to memory for USR(X). (Lo and Hi Bytes) USR(X) will reside in location pointed to.
9826	2662	—	Contains track number for USR(X) on disk (Decimal)
9976	26F8	—	Disable ":" Terminator. See Location 2976 comments.
1Ø95Ø	2AC6	*	Console terminal number. (*1 on Serial Systems; 2 on Video Systems)
11511	2CF7	—	Page 0/1 Swap Address
12076	2F2C	—	Sets record length for data file use
12042	2F0A	—	Sets Number of records per track for data file use.
13Ø26	32E2	171	Selects cursor character (V3.3 only)
13743	35AF	32	Selects Flashing cursor; 44 selects non-flashing cursor. (V3.3 only)

ERROR MESSAGE CODES

1 - Can't Read Sector (Parity Error).

2 - Can't Write Sector (Reread Error).

3 - Track Zero is Write Protected Against that Operation.

4 - Diskette is Write Protected.

5 - Seek Error (Track Header Doesn't Match Track).

6 - Drive Not Ready.

7 - Syntax Error in Command Line.

8 - Bad Track Number.

9 - Can't Find Track Header Within One Rev. of Diskette.

A - Can't Find Sector Before One Requested.

B - Bad Sector Length Value.

C - Can't Find that Name in Directory.

D - Read/Write Attempted Past End of Named File.

BASIC ERROR MESSAGE CODES

BS Bad subscript: Matrix outside DIM statement range, etc.

CN Continue Errors: Attempt to inappropriately continue from BREAK or STOP.

DD Double Dimension: Variable dimensioned twice. Remember, subscripted variables default to dimension 10.

FC Function Call Error: Parameter passed to function out of range.

ID Illegal Direct: INPUT and DEF statements cannot be used in direct mode.

LS Long String: String longer than 255 characters.

NF NEXT without FOR.

OD Out of Data: More reads than data.

OM Out of Memory: Program too big or too many GOSUBs, FOR-NEXT loops or variables.

OV Overflow: Result of calculation too large.

RG RETURN without GOSUB.

SN Syntax Error: Typo, etc.

ST String Temporaries: String expression too complex.

TM Type Mismatch: String variable mismatched to numeric variable.

UF Undefined Function.

US Undefined Statement: Attempt to jump to nonexistent line number.

/ 0 Division by Zero.

OS Out of String Space: Same as OM.

OHIO SCIENTIFIC

A M/A-COM COMPANY

OHIO SCIENTIFIC, INC.
1333 S. CHILLICOTHE ROAD
AURORA, OHIO 44202

DOS COMMANDS

ASM	Load the assembler and extended monitor. Transfer control to the assembler.
BASIC	Load and transfer control to BASIC
CALL NNNN=TT,S	Load contents of Track "TT", sector "S" to memory location "NNNN".
D9	Disable error 9. This is required to read some earlier version files (V1.5, V2.0). (on 8" systems only)
DIR TT	Print sector map directory of track "TT". For each sector, the number of pages is given.
EM	Load the assembler and extended monitor. Transfer control to the extended monitor.
EXAM NNNN=TT	Examine track. Load entire track contents, including formatting information, into location "NNNN".
GO NNNN	Transfer Control (GO) to location "NNNN".
HOME	Reset track count to zero and HOME the current drive's head to track zero.
INIT	INITIALIZE the entire disk. I.e. erase the entire diskette (except track 0 and write new formatting information on each track.
INIT TT	Same as "INIT", but only operates on Track "TT".
IO NN,MM	Changes the Input I/O distributor flag to "NN", and the Output flag to "MM".
IO ,MM	Changes only the Output flag.
IO NN	Changes only the Input flag.
LOAD FILNAM	Loads named source file, "FILNAM", into memory.
LOAD TT	Loads source file into memory given starting track number "TT".
MEM NNNN,MMMM	Sets the memory I/O device Input pointer to "NNNN", and the Output pointer to "MMMM".
PUT FILNAM	Saves source file in memory on the named disk file "FILNAM".
PUT TT	Saves source file in memory on track "TT" and following tracks.
RET ASM	Restart the assembler.
RET BAS	Restart Basic.
RET EM	Restart the extended monitor.
RET MON	Restart the Prom monitor (via RSTVECTOR).
SAVE TT,S=NNNN/P	Save memory from location "NNNN" on track "TT" sector "S" for "P" pages.
SELECT X	Select disk drive "X" where "X" can be; A, B, C, or D. Select enables the requested drive and homes the head to track 0.
XQT FILNAM	Load the file, "FILNAM" as if it was an object file, and transfer control to location $3A7E (317E on 8"V3.2; 327E on 5" V3.2)
XQT TT	Load the file beginning on track "TT" as if it was an object file and transfer control to location $3A7E (317E on 8" V3.2; 327E on 5" V3.2)

NOTES:

—Only the first 2 characters are used in recognizing a DOS command. The rest up to the blank are ignored.

—Commands can be used in the baisc mode in the form DISK! "DOS" where DOS represents one of the commands above.

—All memory locations should be in hex.