Peeks, Pokes and Pointers

Apple® Zero-Page

DECIMAL — HEX

32 **Text Window Left-Edge** (0-39 / normal is 0) $20
Example: **POKE 32, X** freezes the left X columns of text.
Warning: Don't let PEEK(32)+PEEK(33) exceed the screen width.

33 **Text Window Width** (1-40 or 1-80 / normal is 40 or 80) $21
Note: **POKE 33,33** scrunches listings to remove extra spaces.

34 **Text Window Top-Edge** (0-23 / normal is 0) $22

35 **Text Window Bottom** (1-24 / normal is 24) $23

36 **Horizontal Cursor-Position** (0-39) $24
Examples: If **PEEK(36)=X**, then the cursor is in column X+1.
POKE 36,X puts the cursor in column X+1 (useful with 80-columns, for positioning the cursor beyond the 40-column limit of HTAB).
Note: **POKE 1403,X** works similar1y—and more predictably.

37 **Vertical Cursor-Position** (0-23) $25
Examples: If PEEK(37)=V, then the cursor is on text line Y+1.

43 **Boot Slot*16** (after boot) $2B

44 **Lo-Res Line End-Point** $2C

48 **Lo-Res COLOR*17** $30

50 **Text Output Format** $32
POKE 50, **63**=INVERSE. POKE 50, **255**=NORMAL,
POKE 50, **127**=FLASH (tor ASCII 64-95).

51 **Prompt-Character** $33
Note: **POKE 51,0: GOTO line#** will prevent a false "Not Direct Command" message caused by an immediate GOTO line# command.

78-79 **Random-Number Field** $4E.4F

103-104 **Start of Applesoft Program** $67.68
To Load a program at a non-standard location LOC—
POKE LOC-1. 0: POKE 103, LOC-INT(LOC/256)*256:
POKE 104, INT(L0C/256) *Then* LOAD PROGRAM
Note: **FP** (DOS 3.3 only) sets start-of-program to normal 2049 ($801).

105-106 **LOMEM** $69.6A
Note: LOMEM is the Start of Variable-Space, equivalent to End-of-Program (approx.) unless changed with the **LOMEM:** command.

107-108 **Start of Array-Space** $6B.6C

109-110 **End ot Array-Space** $6D.6E

111-112 **Start of String-Storage** $6F.70

115-116 **HIMEM** $73.74
Note: HIMEM-1 is the highest address avallable for use by an Applesoft program. May be changed with the **HIMEM:** command.

117-118 **Line-Number Being Executed** $75.76

119-120 **Line-No. Where Program Stopped** $77.78

121-122 **Address of Line Executing** $79.7A

123-124 **Current DATA Line-Number** $7B.7C

125-126 **Next DATA Address** $7D.7E

127-128 **INPUT or DATA Address** $7F.80

129-130 **Last-Used Variable Name** $81.82

131-132 **Last-Used-Variable Address** $83.84

175-176 **End of Applesoft Program** $AF.B0

214 **RUN Flag** $D6
Example: **POKE 214, 255** makes *any* command RUN a program.

216 **ONERR Flag** $D8
Example: **POKE 216, 0** cancels the ONERR function.

218-219 **Line-Number of ONERR Error** ..... $DA.DB

220-221 **ONERR Error Address** ..... $DC.DD

222 **ONERR Error Code** ..... $DE

DOS 3.3 and PRODOS	APPLESOFT
1: Language Not Available	0: ?Next Without For
2 or 3[1]: Range Error	16: ?Syntax Error (FP)
3: No Device Connected[2]	22: ?Retum Without Gosub
4: Write-Protected	42: ?Out of Data
5: End of Data	53: ?Illegal Quantity
6: File[1] or Path[2] Not Found	69: ?Overflow
7: Volume Mismatch[1]	77: ?Out of Memory
8: I/O Error	90: ?Undef'd Statement
9: Disk Full	107: ?Bad Subscript
10: File Locked	120: ?Redim'd Array
11: Syntax Error[1] or Invalid Option[2]	133: ?Dlvision by Zero
12: No Buffers Available	163: ?Type Mismatch
13: File Type Mismatch	176: ?String Too Long
14: Program Too Large	191: ?Formula Too Complex
15: Not Direct Command	224: ?Undef'd Function
17: Directory Full[2]	254: ?Re-Enter
18: File Not Open[2]	255: (control-C Interrupt)
19: Duplicate File Name[2]	
20: File Busy[2]	[1]DOS 3.3 only
21: File(s) Still Open[2]	[2]ProDOS only

224-225 **X ot Last HPLOT** (0-279) $E0.E1

226 **Y ot Last HPLOT** (0-191) $E2

228 **HCOLOR Code** $E4
0=**0**, 42=**1**, 85=**2**, 127=**3**, 128=**4**, 170=**5**, 213=**6**, 255=**7**

230 **Hi-Res Plotting Page** $E6
POKE 230,**32** selects Page 1. POKE 230,**96** selects Page 3.
POKE 230,**64** selects Page 2.

231 **SCALE** $E7
Note: **SCALE=0** is equivalent to a SCALE of 256.

232-233 **Shape Table Start Address** $E8.E9

234 **Hl-Res Colllslon-Check** ..... $EA
Example: XDRAW a shape. If **PEEK(234)=0** then the shape started at a *non-black* hi-res point.

241 **SPEED** $F1
Note: **PEEK(241)** is 256 *minus* the current SPEED.

243 **FLASH Mask** $F3

249 **ROT** $F9

Display Switches

DECIMAL (with negative equivalent) — HEX

49232 (-16304) **Graphics** $C050

49233 (-16303) **Text** $C051

49234 (-16302) **Full-Graphics** $C052

49235 (-16301) **Split-Screen** $C053

49236 (-16300) **Page One** $C054

49237 (-16299) **Page Two** $C055

49238 (-16298) **Lo-Res** $C056

49239 (-16297) **Hi-Res** $C057

Note: Activate display switches by Poking each location.
Example: **POKE 49232,0** switches to Graphics display.

Keyboard, etc.

DECIMAL (with negative equivalent) — HEX

49152 (-16384) **Read Keyboard** $C000

49168 (-16368) **Clear Keyboard** $C010

Example: 10 KEY=**PEEK(49152)**: IF KEY<128 THEN 10
20 **POKE 49168, 0**
30 PRINT "KEY: "; CHR$(KEY-128)

49200 (-16336) **Click Speaker** $C030
Example: FOR A=1 TO 99: BUZZ=PEEK(49200): NEXT

49249 (-16287) **Button #0** $C061
Paddle-0 Button or Open (left) Apple key.*

49250 (-16286) **Button #1** $C062
Paddle-1 Button or Closed (right) Apple key*

49251 (-16285) **Button #2** $C063
*Example: If **PEEK(49249+P)** is greater than 127, then Paddle Button #P is being pressed—or it's not connected.

DOS 3.3 Pokes

(assume DOS loaded in main memory)

POKE 40193, PEEK(40193)-N: CALL 42964
Moves DOS buffers down N*256 bytes.

POKE 44452,N+1: POKE 44605,N
Allows N file names before catalog pause.

POKE 44460,88: POKE 44461,252
Clears screen before catalog.

POKE 44505,234: POKE 44506,234
Exposes deleted file names in catalog.

POKE 44596, 234: POKE 44597, 234: POKE 44598, 234 Cancels catalog pause.

POKE 49107,234: POKE 49108,234: POKE 49109, 234 Prevents language card reload.

POKE 49384,0 Stops drive motor.

POKE 49385,0 Starts drive motor.

Notes

Apple's main memory consists of 65,536 *bytes*, numbered zero to 65535. Every byte has a *value* in the range 0-255.

- You may *Peek* (look at) the value in byte number-B with the command— **PRINT PEEK(B)**
- You can usually *Poke* a new value-V into byte-B with the command— **POKE B.V**

Values higher than 255 must be stored in two bytes:

- To look at the value in consecutive bytes B1-B2- **PRINT PEEK(B1)+PEEK(B2)*256**
- To Poke a new value V (D-65535) into bytes B1-B2- **POKE B1, V-INT(V/256)*256** and **POKE B2, INT(V/256)**

Note: Since almost any memory location can be Peeked or Poked, program listings can reveal thousands of Peeks and Pokes not listed on this chart. Pokes are otten used to write machine-language routines that may be activated with the CALL command—the possibilities are *infinite.*

Let A=PEEK(64435) and B=PEEK(64448).
If A=6 and B=0 then Apple IIc.
If A=6 and (B>223 AND B<240) then Apple IIe.
If A<>6 then Apple II or II+.

Page-3 DOS Vectors

DECIMAL — HEX

976-978 **Re-enter-DOS Vector** $3D0.3D2

1010-1012 **Reset Vector** $3F2.3F4
Example: POKE 1012, 0 makes Reset boot.
(POKE 1012,56 to restore normal Reset function,)

1013-1015 **Ampersand Vector** $3F5.3F7
Examples: POKE 1014, 165: POKE 1015, 214 makes "&" LIST.
POKE 1014, 110: POKE 1015, 165 makes "&" CATALOG.
POKE 1014, 18: POKE 1015, 217 makes "&" RUN.

1016-1018 **Control-Y Vector** $3F8.3FA

DOS 3.3 Locations

DECIMAL — HEX

(All values assume DOS is loaded in main memory.)

42350 **Catalog-Routine** $A56E
Example: CALL 42350 catalogs a disk.

40514 **Greeting Program Run-Flag** $9542
POKE 40514,52 and INIT a disk. When booted, DOS will attempt to *BRUN* the greeting program. POKE 40514,20 for *EXEC*.

43140-43271 **Commands** $A884.A907

43378-43582 **Error Messages** $A972.AA3E

43616-43617 **Last Bload Length** $AA60.AA61

43634-43635 **Last Bload Start** $AA72.AA73

43624 **Drive-Number** $AA68
Example: **POKE 43624, D** changes disk input/output to Drive D.

43626 **Slot-Number** $AA6A
Example: **POKE 43626, S** changes disk input/output to Slot S.

43698 **Control-D Command Character** $AAB2

44033 **Catalog Track Number** $AC01

45991-45998 **File-Type Codes** $B3A7.B3AE

45999-46010 **Disk Volume Heading** $B3AF.B3BA

46017 **Disk Volume Number** $B3C1

ProDOS™ Locations

DECIMAL — HEX

48944 **Slot/Drive Value** $BF30
If **PEEK(48944)** is greater than 127 then Drive 2, otherwise Drive 1.

47313-47422 **Commands** $B8D1.B93E

48840-48841 **Last Bload Length** $BEC8.BEC9

48825-48826 **Last Bload Start** $BEB9.BEBA

Useful Calls

DECIMAL (add 65536 for positive equivalent) — HEX

CALL-25153 **Reconnect DOS 3.3** $9DBF

CALL-3100 **Reveal hi-res page 1** $F3E4

CALL-3086 **Clear hi-res screen to black** $F3F2

CALL-3082 **Clear hi-res to last color Hplotted** . $F3F6
Example: HGR2: HCOLOR=5: HPLOT 0,0: **CALL-3082**

CALL-2613 **Hi-res coordinates to Zero-Page** ... $F5CB
Example: The X and Y starting coordinates of the *next* shape table DRAW or XDRAW may be determined with a **CALL-2613**. Then X=PEEK(224)+PEEK(225)*256 and Y=PEEK(226).

CALL-1438 **Pseudo-Reset** $FA62

CALL-1370 **Boot** $FAA6

CALL-1321 **Display all registers** $FAD7

CALL-1184 **Clear screen and print "Apple..."** .. $FB60

CALL-1036 **Move cursor right** $FBF4

CALL-1008 **Move cursor left** $FC10

CALL-998 **Move cursor up** $FC1A

CALL-958 **Clear text from cursor to bottom** ... $FC42

CALL-922 **Move cursor down** $FC66

CALL-868 **Clear text-line from cursor to right** $FC9C

CALL-756 **Wait for any keypress** $FD0C

CALL-678 **Wait for a Return keypress** $FD5A

CALL-657 **Better Input; commas/colons o.k.** . $FD6F
10 PRINT "NAME (LAST, FIRST) :";: **CALL -657**
20 A$= "": FOR X=512 TO 767: IF PEEK(X)<>141 THEN A$=A$+CHR$(PEEK(X)-128): NEXT X

CALL-468 **Memory move** $FE2C
A Basic memory move: OS & OE are the Old-location Start & End, and NS is the *New* Start. GOSUB 5000 to execute the move—
5000 N=OS: LOC=60: GOSUB 5020:
N=OE: LOC=62: GOSUB 5020:
N=NS: LOC=66: GOSUB 5020
5010 POKE 766, 160: POKE 769, 0: POKE 770, 76:
POKE 771, 44: POKE 772, 254: CALL 768: RETURN
5020 POKE LOC, N-INT(N/256)*256:
POKE LOC+1, INT(N/256): RETURN

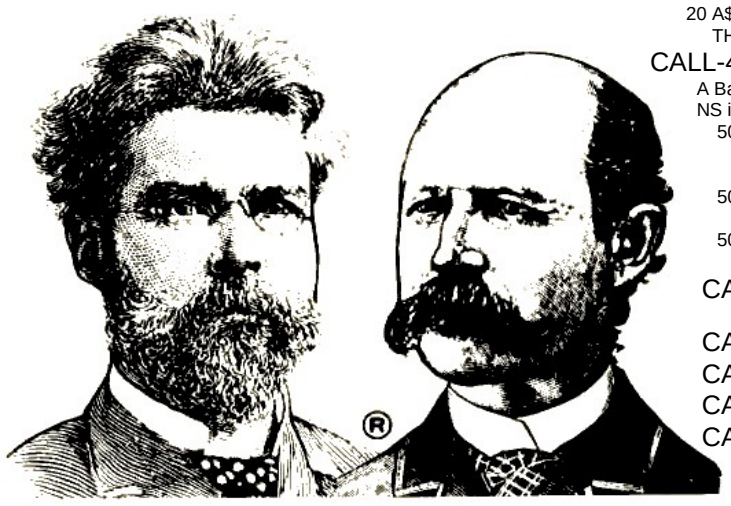
CALL-415 **Disassembler** $FE61
Note: Poke start address at locations 58-59 before Call.

CALL-211 **Ring bell and print "ERR"** $FF2D

CALL-198 **Ring bell** $FF3A

CALL-151 **Enter monitor** $FF69

CALL-144 **Scan input buffer** $FF70
This example uses CALL -144 to execute a machine language routine from Basic (will not work in a subroutine):
100 A$="300: A9 C1 20 ED FD 18 69 01 C9 DB D0 F6 60 300G D823G
110 FOR X=1 TO LEN(A$): POKE 511+X, ASC(M|D$(A$.X,1))+128: NEXT
120 POKE 72, 0: **CALL -144**



To get on a really good mailing list, write:
BEAGLE BROS INC.
3990 Old Town Avenue, Suite 102C
San Diego, California 92110

Beagle Bros Micro Software Inc.

Beagle Bros makes useful and entertaining Utilities, Games and Publications for all versions of Apple II® Computers.

12/11