MYARC ADVANCED BASIC MANUAL NOTES AND CORRECTIONS

The following text comes from Myarc and is related to the Advanced BASIC interpreter. This is an addendum dated September 20, 1989 that Lou released at the Chicago fair. There are quite a few bits of useful information and new or previously unknown commands available. Enjoy!

ADVANCED BASIC INTERPRETER

The Advanced BASIC (ABASIC) diskette included with your 9640 contains the files BASIC1-BASIC7, and a program named “MOUSEDEMO”. To load ABASIC simply insert the ABASIC diskette in drivel type BASIC1 and press ENTER from the MDOS command line. Once ABASIC is loaded it will search drive 1 for a program named “LOAD” to load and begin execution. The initial qraphics mode is (3,1) (see your MYASC Advanced BASIC Manual for the descriptions of the graphics modes).

The following are the additions and changes to the Advanced BASIC Manual.

BEEP — Works only in DISPLAY BEEP and ACCEPT BEEF, not by itself.

CALL — The list of CALL statements in the manual is incomplete. When passing variables in a CALL to a user defined SUB, multi-dimensioned array, variables must include a comma within parentheses for each dimension above 1. For example, to pass three-dimension variable named V to a subroutine called MYSUB, the correct call is: CALL MYSUB(V(,,)). The additional built-in subprograms include:

BCOLOR	HIDEMOUSE	MLOC	PSET	STCR
BTIME	INF	MOUSEDRAG	RESETFLT	TCOLOR
DATE	LDCR	MREL	SEEMOUSE	TIME
ECOLOR	LPR	MYART	SCHAR	VERSION
FILES	MEMSET	OUTP	SPRITE2	
GPOINT	MKEY	PALETTE	SPRITESET	

CDBL — Double precision binary is not implemented

CHAR — No change, but refer to SCHAR for related capability.

CHDIR — A command used to set default directory. See FILES and FWD.

CIRCLE — Works as in XBASIC II: CALL CIRCLE(line—tvpe, pixelrow, pixelcol, radius). See DRAW for line types.

CLOSE — CLOSE ALL added, closing all open files.

COINC — In addition to formats shown, ALL sprites can be checked far overlap and return the point where the intersection occurred: CALL COINC(ALL, numberic variable, pixelrowvar, pixelcolvar) Do not use when MOUSE interrupts are on (MOUSE ON).

COLOR — In the 256 color mode (2,2), the colors are 1-256. In the 4 color mode (3,2), the colors are 1-4. CALL COLOR (#0, foreground color)

	sets the MOUSE color. Also reference TCOLORU PALETTE in addition to references listed.
CON, CONTINUE	CONTINUE is not used. Use only CON.

DATE, DATE$ — The example of CALL DATE in the middle of page 44 erroneously shows CALL DATE$(“01/01/87”). Omit the “$.”

DEFvartype — DEFDBL and DEFSNG are not implemented.

DIM — OK, accepts larger limits than docs say. Limited by the memory in the 9640.

ECOLOR — New command. CALL ECOLOR(color) is used to “color in” the edge between the text and the border in graphics modes (2,2), (2,3), (3,2) and (3,3).

FILES — CALL FILES with no added parentheses tries to read the default directory. Also. CALL FILES(X) will read the directory of floppy drive X. Lastly, if you use a path name in the CALL FILES command, it must end with a “.” like CALL FILESY(“DSKl.”). DO NOT use with a window less than 28 characters wide.

FREESPACE — The format is changed to FREESPACE(memory-type). The possible values for memory type are 0 through 4 as follows:

0 - ALL memory
1 - program space
2 - data space
3 - assembly space
4 - stack space

If the (memory-type) is omitted then ALL memory will be returned.

GPOINT — New command. CALL GPOINT(pixelrow,pixelcolumn,numeric-variable) returns the color of the referenced point in the numeric variable. Use in the bit mapped graphics modes.

GRAPHICS — The modes come up in different colors than documented. Mode (1,1) is the usual black on cyan. Other modes show white on blue or dark blue. When going from a mode with 16 colors, to another 16-color mode, the former colors will be maintained. Mode (3,2) only presents 4 colors that represent colors 1-4 of the standard 16-color mode. A new GRAFHICS mode, (4) will show an 80x24 screen.

INIT — Format is CALL INIT with no numeric-expression. There is always 48k (49,152 bytes) available.

INF — CALL INF(port #,variable[,var2[…J]). See OUTF (listed as OUT in manual).

KEY — In redefining function keys (p98), KEY ON turns on the key labels on line 26 (in Graphics modes (3,1). (3,2) and (3.3)). The command KEY LIST will scroll the key definitions, the default directory and the default printer device to the screen in all graphics modes. If SCROLL LOCK is set, pressing the function key returns the string currently assigned to the function key in command mode and when a program is calling for input. KEY OFF removes the label display. To change the key definition, KEY(number)=(string expression) will replace the given label with the new string. In a program, a key may be armed by the ON KEY format shown or by the format KEY(numeric expression)ON. This differs from the manual by the addition of the equal sign. To disarm the key, the new format is

KEY(numeric expression)OFF. To disable all key checking and zero out the ON KEY line number table, use KEY STOP.

LLIST — Followed with ",W" a width of 132 is assumed. When followed by Wx. the width x is used.

LPT — To change the default device, the format is LPT DEVICENAME. The equal sign is not used and quotation marks are not needed.

MARGINS — Format is CALL MARGINS() with an S (manual says CALL MARGIN).

MEMSET — When a multi-dimensional array variable is set, a comma must be included within parentheses for each dimension above 1. For example, to set all elements of variable LATTICE(x,y,z) to 1, the format is CALL MEMSET(LATTICE(,,),1).

MOUSE — See the next section for mouse commands. There are several substantial changes to these commands in ABASIC.

MYART — New command. CALL MYART(filename) loads and displays a MYART picture. Graphics mode must match before it is called.

OUP — Listed in manual as OUT. Use only ports 1 or 2 as the PIO or PIO/2 ports respectively.

PALETTE — CALL PALETTE(#color,redvalue,bluevalue.greenvalue,#color2) for one or more colors in the 16 color sets and the 4 color set. It does not work in the 256-color mode (2,2). The range for each color is from 1 to 8.

PEEKV — Meaningless for ABASIC

POKEV — Meaningless for ABASIC.

PPT — Misprinted as PPR. PPT is correct to list the print device.

PSET — CALL PSET(x,y) sets the starting point for DRAWTO.

PWD — Print working directory. See CHDIR and FILES.

RESETPLT — New command. CALL RESETPLT resets the palette to the default values.

SAVE — In order to save in I/V254 format, INTERNAL option has been added. This would only be necessary for a large file of less then 24K to be loaded into TI Extended BASIC.

SCHAR — New command. CALL SCHAR(char#,strvariable) works just like CALL CHAR, but it sets sprite characters only, whereas CHAR sets both characters and sprites. The exception is in graphics (1,1) where both the character and sprite patterns get redefined.

SCREEN — Only uses one parameter CALL SCREEN(backdropcolor). The backdrop color is changed to the new color. In text modes, the text background color also changes to the same color.

SPEED	Not used.
SUB	See CALL above for rules on passing multidimensional arrays.
SWAP	Use SWAP A,B not CALL SWAP.
TCOLOR	New command. CALL TCOLOR(foregroundcolor,backgroundcolor) sets the colors of text. In bit map modes, the color set for a given portion of text remains even when subsequent text is changed. In text modes, when colors are changed, all text is changed at the same time. Color numbers range from 1 to the number of colors available to the mode (4, 16, or 256). See PALETTE.
TIME	As with DATE, the examples in the middle of the page of CALL TIME () should omit the “3”.
VAL	The example is wrong. It should be, “Sets LL equal to 3E+15, or 3,000,000,000,000,000.”

WEND, WHILE — The first sample program on page 226 uses a function not found in ABASIC. The INKEY function is incorrect. The first two lines could read:

```
100 WHILE S=0 then 110
110 CALL KEY (6 ,K,S)
```

I/O DEFAULTS — COM, MDM, PCM, and PMD are not used (see page 227).

MOUSE COMMANDS

ON/OFF/STOP — These commands work as written

ON MOUSE — Syntax is changed to ON MOUSE (btnnum) GOSUB (linenum)

CALL MKEY — CALL MKEY(button1status,button2status,button3status,pixelrow,pixelcol). The variables you use for button#status return 1, 0, or -1, as shown in the manual. The variables you use for pixelrow and pixelcol return the mouse's position.

CALL MKEYLAST — NOT USED

CALL MLOC — CALL MLOC(pxlrow,pxlcol) - as written

CALL MREL — CALL MREL(pxlrow,pxlcol) - as written

CALL MOUSE — NOT USED.

CALL MOUSEDRAG — CALL MOUSEDRAG(ON,linecolor) draws a solid line as you move the mouse. The linecolor is 1-16, 1-256, or 1-4, depending or the mode used. The left button controls the drag. MODE (3,2) will usually require a redefined PALLETTE f or effective use. CALL MOUSEDRAG(OFF) works as written

CALL HIDEMOUSE — as written.

CALL SEEMOUSE — CALL SEEMOUSE(pixelrow,pixelcol,speed) where speed ranges from 1 to 8. Additionally, the mouse is always sprite pattern 252. It is also always sprite #0. The mouse shape can be defined by using CALL SCHAR(252,patternstring). The mouse default color is color 16. It can be changed using CALL COLOR(#0,color). You can alternatively change the mouse color by redefining color 16 with the CALL PALLETTE command.

The program listed below shows the use of several of the mouse commands:

```
100 CALL GRAPHICS(2,2)
110 CALL TCOLOR(2,15)
120 CLS
130 MOUSE ON ! Activates interrupts
140 CALL CHAR(252,"36361C3C7CF81818") ! Sets the mouse character shape
150 CALL DCOLOR(5,1) ! Sets the color of the rectangle to be drawn in 230
160 CALL SEEMOUSE(100,100,3) ! Makes the mouse visible at location 100,100
170 DISPLAY AT(2,5):"MOVE ME AROUND"
```

```
180 FOR N=1 TO 1000 :: NEXT N ! Delay loop to let you move the mouse around
190 DISPLAY AT(4,5):"NOW, PRESS THE LEFT BUTTON"
200 CALL MKEY(A,B,C,D,E) :: IF A=0 THEN 2(30 ! Repeats until you press button 1
210 CALL MOUSEDRAG(ON,7) ! Turns on the mouse drawing command
220 CALL MKEY(A,B,C,D,E) ! Sets up the loop while button 1 is pressed per 260
230 CALL RECTANGLE (1,166,16,166,92,190,16)
240 DISPLAY AT(22,4)SIZE(8): "LOCATION"
250 DISPLAY AT(23,4)SIZE(10)INVERT :D;E ! Displays pixel values of position
260 IF A=1 THEN 220 ! Loops as long as button 1 is pressed, allowing drawing
270 PRINT "DONE"
280 END
```

ADVANCED BASIC LOADING OPTIONS

When invoking ABASIC from the MDOS command line you may also want to invoke several available options. One option is the amount of memory allocated to data space. The following depicts the memory allocation for ABASIC:

ABASIC INTERPRETER = 56K
PROGRAM MEMORY = 64K
ASSEMBLY SUBROUTINES = 48K
DATA SPACE (VARIABLES AND STRINGS) = 64K or greater
DATA BUFFERS = 8K

Memory allocation is fixed except for data space. Data space can be as small as 64K (the default amount), or as large as the available memory in your 9640. To request data space greater than 64K, simply type a space followed by the amount of memory desired (in K bytes) after typing BASIC1 to invoke ABASIC. For example, to request 128K of DATA space, type BASIC1 128 and press ENTER. Another optional parameter in the command line is the selection of the initial default directory and the initial program to be loaded and executed. In order to set a different default directory in ABASIC, type a space and the desired directory pathname ending with a period. ABASIC will set this as the default directory and initially try to load and execute a file named "LOAD" on this directory. If you would like to initially execute another program on the default directory, simply enter the file name. Lastly, if you just want to load the ABASIC interpreter and inhibit the auto load of the initial program, enter an asterisk "*". The following examples should help you understand these capabilities better:

BASIC1 *	sets the default directory to that of MDOS (Usually DSK1) and goes to the command mode of BASIC.
BASIC1	sets the default directory to that of MDOS (Usually DSK1) and attempts to load and run the program LOAD.
BASIC1 128 DSK2.PROG1	attempts to allocate 128K to DATA space, sets the default directory to DSK2 and attempts to load and run the program PROG1.
BASIC1 128 DSK2.*	attempts to allocate 128K to DATA space, set the default directory to DSK2 and goes to the command mode of ABASIC.

NOTE ON ADDITIONAL CORRECTIONS

I would appreciate any and all comments on this documentation. Just as ABASIC itself has benefited by going through a period of testing, the docs can similarly benefit from the comments of many users. Send your comments to me at any of the following:

Walt Howe
43 S. Chelmsford Rd.
Westford, MA 01886
Tel. (voice) (508) 692-2707 (evenings)

Compuserve: 70277,353(3
Delphi: WALTHOWE
Genie: WALT.HOWE