Apple II Computer Family Information

AppleSoft BASIC Info:
AppleSoft Internal Entry Points

Crossley - Apple Orchard Mar/Apr 1980

Document # 43

Ex Libris David T. Craig
Applesoft Internal Entry Points

By John Crossley (from the Apple Orchard)

CONTENTS
INTRODUCTION ........................................ 51
LABELS ............................................. 52
ABBREVIATIONS .................................... 51
TXTPTR INPUT ROUTINES ......................... 52
TXTPTR TO INTEGER ROUTINES ................. 52
FLOATING POINT MATH PACKAGE ............... 52
INTRODUCTION .................................... 52
REGISTERS ........................................ 52
OPERATORS ........................................ 52
CONSTANTS ........................................ 53
FUNCTIONS ........................................ 53
MOVE ROUTINES .................................. 53
UTILITIES ......................................... 53
CONVERSIONS ..................................... 53
INTEGER TO FAC .................................. 53
FAC TO INTEGER .................................. 53
TXTPTR TO FAC .................................. 53
STRING UTILITIES ................................ 54
DEVICE INPUT ROUTINES ....................... 54
DEVICE OUTPUT ROUTINES ..................... 54
INTERNAL LOCATOR ROUTINES ............... 55
INITIALIZATION ROUTINES ..................... 55
STORAGE MANAGEMENT ROUTINES ........... 55
MISCELLANEOUS BASIC COMMANDS .......... 55
HIRES GRAPHICS ROUTINES .................... 55
CASESET ROUTINES ............................... 56
ERROR PROCESSOR ROUTINES ................. 56
SYNTAX CHECKING ROUTINES ................. 56
INDEX ............................................ 57

INTRODUCTION

This is a guide for the 6502 machine language programmer who wants to take advantage of the various subroutines in Applesoft. The addresses included assume that the user has an Apple II Plus, an Applesoft firmware card, or a Language Card. This list is believed to be correct, but be warned that it was a spare time project. If you find errors, contact your user group. This data is meant for the experienced programmer, NOT THE BEGINNER. Read your Applesoft Reference manual for more information.

Take special note of CHRGET. This subroutine is the heart of Applesoft. When Applesoft wants the next character or an instruction it points TXTPTR at the program or the input buffer and ISRs to CHRGET. When Applesoft READS DATA, TXTPTR is temporarily set to the last used DATA statement.

LABELS HEX ADDR
A1 3C.3D Apple monitor pointer for cassette routines
A2 3E.3F Apple monitor pointer for cassette routines
ARYTAB 6B.6C Start of array storage
BUF 200.2FF Line input buffer
CHARAC OD Used by STRT2
CURLIN 75.76 The current line number (=FF if in direct mode.
DATLIN 7B.7C Line number of current DATA statement
DATPTR 7D.7E The address the next DATA comes from
DSCTMP 9D.9E Temp string descriptor
ENDCHR OE Used by SRTL2
ERRFLG D8 $80 if ONERR active
ERRLIN DA.DB Line number where error occurred
ERRNUM DE Which error occurred
ERROPOS DC.DD TXTPTR save for HNDLERR
ERRSTK DF Stack pointer value before error
FBUFR 100.110 FOUT buffer
FIRST F0 Used by PLOTFS
FORPNT 85.86 General pointer, see COPY
FRESPC 71.72 Temp pointer for string storage routines
FRETOP 6F.70 Bottom of string storage
H2 2C Used by PLOTFS
HIGHDS 94.96 Used by BLTU
HIGHL 96.97 Used by BLTU
HPAG E6 HIRES page to plot on, ($20 for HGR, $40 for HGR2)
INDEX 5E.5F Temp pointer for moving strings
INVLIG 32 Mask for inverse output
LASTPT 53 Last used temp string pointer
LINNUM 50.51 General purpose 16 bit number location
LOWTR 9B.9C General purpose register, GETARYPT
MMSIZ 73.74 HIMEM
OLDLIN 77.78 Last line executed
ORMASK F3 Mask for flashing output
PRGEND AF.B0 The end of the program text
REMSTK F8 Stack pointer saved before each statement
ROT F9
SCALE E7
SPDBYT F1 Speed = delay number
STREND 6D.6E The top of array storage
STRING1 AB.AC Pointer to a string. See MOVINS
STRING2 AD.AE Pointer to a string. See STRT2
SUBFLG 14 $00 subscriptions allowed, $80=non subscriptions
TEMPPT 02 Last used temporary string descriptor
TXTPTR B8.B9 Next byte to be read
TXTTAB 67.68 Start of program text
V2 2D Used by PLOTFS
VALTyp 11 Flags last FAC operation 0=number, FF= string
VARPNT 83.84 Used by PRTGET
VARTAB 69.6A Start of variable storage
APPLESOFT INTERNAL ENTRY POINTS

ABBREVIATIONS
A the 6502 accumulator
X the 6502 X register
Y the 6502 Y register
Z the zero flag of the 6502 status register
C the carry flag of the 6502 status register
(X,Y) is the number or string whose address is in Y and A with the msb in Y and the lsb in A.
FAC the floating point accumulator
ARG the ARGument register
msb most significant bit or byte
lsb least significant bit or byte
eol end of line token ($00)

TXTPTR INPUT ROUTINES
CHRGET (0B1h(17)) (Increment TXTPTR)
CHRGET (0B7h(183)) (No increment)
These routines load A from TXTPTR and set certain 6502 status flags. X and Y are not changed.
On exit:
A=the character
Z is set if A is ‘:’ or eol ($3A or $00)
C is clear if A is an ASCII number (‘0’ to ‘9’).

TXTPTR TO INTEGER
LINGET DAOC (55820)
Read a line number (integer 0 to 63999) from TXTPTR into LINNUM. LIN GET assumes that the 6502 registers and A have been set up by the CHRGET that fetched the first digit. Normally exits through CHRGET which fetches the character after the number. If the number is greater than 63999 then LINGET exits via SYN TAX ERROR. LINNUM is zero if there is no number at TXTPTR.

GETBYT E6F5 (59125)
JSR to CHRGET to gobble a character and fall into GETBYT.
GETBYT E6F8 (59128)
Evaluates the formula at TXTPTR, leaves the result in FAC, and falls into CONINT. On the entry TXTPTR points to the first character of the formula.

PLOTFNS F1EC (61932)
Get 2 LO-RES plotting coordinates (0:47,0:47) from TXTPTR separated by a comma. On entry TXTPTR points to the first character of the formula for the first number. PLOT FNS puts the first number in FIRST and the second number in H2 and V2.

HFNS F6B9 (63161)
Get HI-RES plotting coordinates (0:279,0:191) from TXTPTR. On entry TXTPTR points to the first character of the formula for the first number. Leaves the 6502 registers set up for HPOSIT.
On exit:
A=vertical coordinate
X=lsb of horizontal coordinate
Y=msb of horizontal coordinate.

FLOATING POINT MATH PACKAGE
INTRODUCTION
This is the number format used throughout Applesoft:
The exponent is a single byte signed number (EXP) in excess 80 form (the signed value has 80 added to it). The mantissa is 4 bytes (HO, MOH, MO, LO). The binary point is assumed to be to the right of the most significant bit. Since in binary floating point notation the msb is always 1, the number’s sign is kept there when the number is stored in packed form in memory. While in the math package the sign is kept in a separate byte (SGN) where only bit 7 is significant. If the exponent is zero, then the number is zero although the mantissa isn’t necessarily zero.

Examples:
EXP HO MOH MO LO SGN
Packed format
-10 84 A0 00 00 00 10 84 20 00 00 00 FAC format
-10 84 A0 00 00 00 FF 10 84 A0 00 00 00 00
Arithmetic routine calling conventions:
For single argument functions:
The argument is in FAC.
The result is left in FAC.
For two argument functions:
The first argument is in ARG (see CONUPK).
The second argument is in FAC.
The result is left in FAC.

FLOATING POINT REGISTERS
NOTE: many of the following locations are used for other things when not being used by the floating point math package.

ARG TEMPI TEMP2 TEMPI2 RND
EXP 9D A5 93 98 8A C9
HO 9E A6 94 99 8B CA
MOH 9F A7 95 9A 8C CB
MO A0 A8 96 9B 8D CC
LO A1 A9 97 9C 8E CD
SGN A2 AA (packed format)

FLOATING POINT OPERATORS
FMULT E97F (59775)
Move the number in memory pointed to by Y,A into ARG and fall into ...
DIVMUL E982 (59778)
Multiply FAC and ARG. On entry A and Z reflect FACEXP.
FDIV EA66 (60006)
Move the number in memory pointed to by Y,A into ARG and fall into ...
DIVFD EA69 (60009)
Divide ARG by FAC. On entry A and Z reflect FACEXP.
FADD E7BE (59326)
Move the number in memory pointed to by Y,A into ARG and fall into ...
FADDD E7C1 (59329)
Add FAC and ARG. On entry A and Z reflect FACEXP.
FSUB E7A7 (59303)
Move the number in memory pointed to by Y,A into ARG and fall into ...
FSUSB E7AA (59306)
Subtract FAC from ARG. On entry A and Z reflect the value of FACEXP.
FPWRT EE97 (61079)
Exponentiation (ARG to the FAC power). On entry A and Z should reflect the value of FACEXP.
NOTE: Most FAC move routines set up A and Z to reflect FACEXP but a LDA $9D will insure the proper values.

FLOATING POINT CONSTANTS
NOTE: The following addresses point to numbers in packed form suitable for use by CONUPK and MOVMP.

RND 00C9 (201)
1/4 F070 (61552)
1/2 EE64 (61028)
-1/2 EE97 (59703)
1 E913 (59667)
10 EA50 (59984)
FLOATING POINT FUNCTIONS

SGN EB90 (60304)
Calls SIGN and floats the result in the FAC.
On exit:
  FAC=1 If FAC was greater than 0
  FAC=0 If FAC was equal to 0
  FAC=1 If FAC was less than 0
ABS EBAF (60335)
Absolute value of FAC
INT EC23 (60451)
Greatest integer value of FAC. Uses QINT and floats the result.
SQR EE8D (61069)
Take the square root of FAC
LOG E941 (59713)
Log base e of FAC
EXP EF09 (61193)
Raise e to the FAC power
RND EFAE (61358)
Form a 'random' number in FAC
COS EFEA (61418)
COS(FAC)
SIN EFF1 (61425)
SIN(FAC)
TAN F03A (61498)
TAN(FAC)
ATN F09E (61598)
ARCTAN(FAC)

FLOATING POINT NUMBER MOVE ROUTINES

MOVFM EAP9 (60153)
Move memory pointed to by Y.A. into FAC. On exit A and Z reflect FACEXP.
MOV2F EB1E (60190)
Pack FAC and move it into temporary register 2. Uses MOVMF. On exit A and Z reflect FACEXP.
MOV1F EB21 (60193)
Pack FAC and move it into temporary register 1. Uses MOVMF. On exit A and Z reflect FACEXP.

MOVML EB23 (60195)
Pack FAC and move it into zero page area pointed to by X. Uses MOVMF. On exit A and Z reflect FACEXP.
MOVMF EB2B (60203)
Pack FAC and move it into memory pointed to by Y.X. On exit A and Z reflect FACEXP.
MOVFA EB53 (60243)
Move ARG into FAC. On exit A=FACEXP and Z is set.
MOVAF EB63 (60259)
Move FAC into ARG. On exit A=FACEXP and Z is set.
CONUPK E9E3 (59875)
Load ARG from memory pointed to by Y.A. On exit A and Z reflect FACEXP.

SUMMARY OF MOVES

FAC =>(Y,A) EB2B
FAC =>(0.X) EB23
FAC => TEMP 1 EB21
FAC => TEMP 2 EB1E
FAC => ARG EB63
(Y,A) => FAC EAF9
(Y,A) => ARG E9E3
ARG => FAC EB63

FLOATING POINT UTILITIES

SIGN EB82 (60290)
Set A according to the value of FAC.
On exit:
  A=1 if FAC is positive.
  A=0 if FAC=0
  A=FF if FAC is negative
FOUT ED34 (60724)
Creates a string in FBUFFR equivalent to the value of FAC. On exit Y.A points to the string. The string ends in a zero. FAC is scrambled. Use STROUT to then print the number.
FCOMP EB2B (60338)
Compare FAC and a packed number in memory pointed to by Y.A.
On exit:
  A=1 if (Y.A)<FAC
  A=0 if (Y.A) =FAC
  A=FF if (Y.A)>FAC
NEGOP E60 (61136)
FAC=-FAC
FADDH E7A0 (59296)
Add 1/2 to FAC
DIV10 EA55 (59989)
Divide FAC by 10. Returns positive numbers only.

MUL10 EA39 (59961)
Multiply FAC by 10. Works for both positive and negative numbers.

CONVERSIONS

INTEGER TO FAC

SNGLT E301 (58113)
Float the unsigned integer in Y.
GIVAYF E2F2 (58098)
Float the signed integer in A.Y.
FLOAT EB93 (60307)
Float the signed integer in A.

FAC TO INTEGER

CONINT E6FB (59131)
Convert FAC into a single byte number in X and FACLO. Normally exits through CHRGET. If FAC is greater than 255 or less than 0 then CONINT exits via ILLEGAL QUANTITY ER ROR.
AYINT E10C (57612)
If FAC is less than +32767 and greater than -32767 then perform QINT.
QINT EBF2 (60402)
Quick greatest integer function. Leaves INT(FAC) in FAC, MO, LO signed. QINT assumes FAC 2 to the 23rd (8388608 decimal)
GETADR E752 (59218)
Convert the number in FAC (+56353 to +55535) into a 2 byte integer (0-65535) in LINNUM.
GETNUM E746 (59206)
Read a 2 byte number into LINNUM from TXTPTR, check for a comma, and get a single byte number in X. On entry TXTPTR points to the first character of the formula for the first number. Uses FRMNUM, GETADR, CHKCOM, GETBYT.
COMBYTE E74C (59212)
Check for a comma and get a byte in X. uses CHKCOM, GETBYT. On entry TXTPTR points to the comma.

TXTPTR TO FAC

FRMEVL DD7B (56699)
Evaluate the formula at TXTPTR using CHRGET and leave the result in FAC. On entry TXTPTR points to the first character of the formula. This is the main subroutine for the commands that use formulas and works for both strings and numbers. If the formula is a string literal, FRMEVL gobbles the opening quote and executes STRLIT and STTXT.
APPLESOFT INTERNAL ENTRY POINTS

FRMNUM DD07 (56679)
Evaluate the formula at TXTPTR, put it in FAC, and make sure it's a number. On entry TXTPTR points to the first character of the formula. TYPE MISMATCH ERROR results if the formula is a string.
FIN EC4A (60490)
Input a floating point number into FAC from CHRGES. FIN assumes that the 6502 registers and A have been set up by the CHRGES that fetched the first digit.

STRING UTILITIES
In Applesoft strings have three parts: the descriptor, a pointer to the descriptor, and the ASCII string. A string descriptor contains the length of the string and the address of its first character. See page 137 of the Applesoft Reference Manual. Through most of the routines the descriptor is left in memory and a pointer is kept in FAC. The pointer is the address of the descriptor. The actual string could be anywhere in memory. In a program, 10 A$="HI" will leave a descriptor pointing to the program text.
CAT E97 (58775)
Concatenate two strings. FACMO,LO point to the first string's descriptor and TXTPTR points to the '*' sign.
STRINI ED5 (58325)
Get space for creation of a string and create a descriptor for it in DSCTMP. On entry A=length of the string.
STRSPA E3DD (58333)
JSR to GETSPA and store the pointer and length in DSCTMP.
COPY DAB7 (55991)
Free the string temporary pointed to by Y.A and move it to the memory pointed to by FORPNT.
MOVINS ED54 (58836)
Move a string whose descriptor is pointed to by STRNG1 to memory pointed to by FRESPA.
MOVSTR E5E2 (58850)
Move the string pointed to by Y.X with a length of A to memory pointed to by FRESPA.
STRTXT DE81 (56961)
Sets Y.A equal to TXTPTR plus C and falls into STRLIT.
STRLIT E3E7 (58343)
Store a quote in ENDCHR and CHARAC so that STRLT2 will stop on it.
STRLT2 E3ED (58349)
Take a string literal whose first character is pointed to by Y,A and build a descriptor for it. The descriptor is built in DSCTMP, but PUT NEW transfers it into a temporary and leaves a pointer to it in FACMO,LO. Characters other than zero that terminate the string should be saved in CHARAC and ENDCHR. Leading quotes should be skipped before STRLT2. On exit the character after the string literal is pointed to by STRNG2. Falls into PUTNEW.
PUTNEW E42A (58410)
Some string function is returning with a result in DSCTMP. Move DSCTMP to a temporary descriptor, put a pointer to the descriptor in FACMO,LO, and flag the result as a string.
GETSPA E452 (58450)
Get space for character string. May force garbage collection. Moves FRESPC and FRETOP down enough to store the string. On entry A=length of characters. Returns with A unaffected and pointer to the space in Y.X, FRESPC, and FRETOP. If there's no space then OUT OF MEMORY error.
FRESTR E5FD (58877)
Make sure that the last FAC result was a string and fall into
FREFAC E600 (58880)
Load the string descriptor pointer in FACMO, LO into Y, A and fall into FRESTR.
FRETEM E604 (58884)
Free up a temporary string. On entry the pointer to the descriptor is in Y.A. A check is done to see if the descriptor is a temporary one allocated by PUTNEW. If so, the temporary is freed up by updating TEMPTT. If a temp is freed up a further check is made to see if the string is the lowest in memory. If so, that area of memory is freed up also by updating FRETOP. On exit the address of the string is in INDEX and Y.X and the string length is in A.
FRETEM E635 (58933)
Free the temporary descriptor without freeing up the string. On entry Y.A point to the descriptor to be freed. On exit Z is set if anything was freed.

DEVICE INPUT ROUTINES
INLIN D52C (54572) (No prompt)
INLIN+2 D52E (54574) (Use character in X for prompt)
Input a line of text from the current input device into the input buffer, BUF, and fall into GDBUSF.
GDBUSF D539 (54985)
Puts a zero at the end of the input buffer, BUF, and masks off the msb on all bytes.
On entry:
X=end of the input line
On exit:
A=0
X=FF
Y=1
INCHR D553 (54611)
Get one character from the current input device in A and mask off the msb. INCHR uses the main Apple input routines and supports normal handcshaking.

DEVICE OUTPUT ROUTINES
STROUT DB3A (56122)
Print string pointed to by Y.A. The string must end with a zero or a quote.
STRPRT DB3D (56125)
Print a string whose descriptor is pointed to by FACMO, FACLO.
OUTDO DB5C (56156)
Print the character in A. INVERSE, FLASH, and NORMAL in effect.
CRDO DAFB (56069)
Print a carriage return.
OUTSPC DB57 (56151)
Print a space.
OUTQST DB5A (56154)
Print a question mark.
INPRT ED19 (60997)
Print "IN" and the current line number from CURLIN. Uses LINPRT.
LINFRT ED24 (60708)
Print the 2 byte unsigned number in X.A.
PRNTFAC E62E (60718)
Prints the current value of FAC. FAC is destroyed. Uses FOUT and STROUT.
INTERNAL LOCATOR ROUTINES
PTRGET DFE3 (57315)
Read a variable name from CHRGET and find it in memory. On entry TXTPTR points to the first character of the variable name. On exit the address to the value of the variable is in VARPNT and Y.A. If PTRGET can't find a simple variable it creates one. If it can't find an array it creates one dimensioned to 0 to 10 and sets all elements equal to zero.
GETARYPT F7D9 (63449)
Read a variable name from CHRGET and find it in memory. On entry TXTPTR points to the first character of the variable name. This routine leaves LOWTR pointing to the name of the variable array. If the array can't be found the result is an OUT OF DATA ERROR.
FDLIN D61A (54810)
Searches the program for the line whose number is in LINNUM.
On exit:
1. If C set LOWTR to the left field of the desired line.
2. If C clear then line not found.
LOWTR to the next higher line.
DATA D995 (55701)
Move TXTPTR to the end of the statement. Looks for ; or end (0).
DATAN D9A3 (55715)
Calculate the offset in Y from TXTPTR to the next ; or end (0).
REMN D9A6 (55718)
Calculate the offset in Y from TXTPTR to the next end (0).
ADDON D998 (55704)
Add Y to TXTPTR.
INITIALIZATION ROUTINES
SCRTCH D64B (54859)
The 'NEW' command. Clears the program, variables, and stack.
CLEAR D66C (54892)
The 'CLEAR' command. Clears the variables and stack.
STKIN D683 (54915)
Clears the stack.
RESTOR D849 (55369)
Sets the DATA pointer, DATPTR, to the beginning of the program.
STXTPT D697 (54935)
Set TXTPTR to the beginning of the program.
STORAGE MANAGEMENT ROUTINES
BLTU D993 (54163)
Block transfer makes room by moving everything forward.
On entry:
Y.A and HIGHL=destination of high address +1
LOWTR=lowest address to be moved
HIGHL=highest address to be moved + 1
On exit:
LOWTR is unchanged
HIGHL=LOWTR — $100
HIGHL=lowest address transferred — $100
REASON D3E3 (54243)
Makes sure there's enough room in memory, checks to be sure that the address Y.A is less than FRETOP. May cause garbage collection. Causes OMERR if there's no room.
GARBAG E484 (58500)
Move all currently used strings up in memory as far as possible. This maximizes the free memory area for more strings or numeric variables.
MISCELLANEOUS BASIC COMMANDS
Note that many commands are not documented because they jump into the new statement fetcher and cannot be used as a subroutine.
CONT D698 (55448)
MOVEST OLDTXT and OLDLIN into TXTPTR and CURLIN.
NEWSTT D7D2 (55250)
Execute a new statement. On entry TXTPTR points to the ; preceding the statement or the zero at the end of the previous line. Use NEWSST to restart the program with CONT.
THIS ROUTINE DOES NOT RETURN.
RUN D566 (54630)
Run the program in memory. THIS ROUTINE DOES NOT RETURN.
GOTO D93E (55614)
Uses LINGET and FDLIN to update TXTPTR. GOTO assumes that the 6502 registers and A have been set up by the CHRGET that fetched the first digit.
LET DA46 (53878)
Uses CHRGET to get address of the variable, =, evaluate the formula, and store it. On entry TXTPTR points to the first character of the variable name.
HIRES GRAPHICS ROUTINES
NOTE: Regardless of which screen is being displayed, HPAG (location SE6) determines which screen is drawn on. ($20 for HGR, $40 for HGR2)
HGR2 F3D8 (62424)
Initialize and clear page 2 HIRES.
HGR F3E2 (62434)
Initialize and clear page 1 HIRES.
HCLR F3F2 (62450)
Clear the HIRES screen to black.
BKGNF F3F6 (62454)
Clear the HIRES screen to last plotted color.
HPOSN F411 (62481)
Positions the HIRES cursor without plotting. HPAG determines which page the cursor is pointed at.
On entry:
Horizontal = Y.X
Vertical = A
HPLTS F457 (62551)
Call HPOSN then try to plot a dot at the cursor's position. No dot may be plotted if plotting non-white at a complementary color X coordinate.
On entry:
Horizontal = Y.X
Vertical = Y
HLIN F53A (62778)
Draws a line from the last plotted point or line destination to the coordinate in the 6502 registers.
On entry:
Horizontal = X.A
Vertical = Y
HFIND F5CB (62923)
Convert the HIRES cursor's position to XY coordinates. Used after SHAPE to find where you've been left.
On exit:
$E0=horizontal lsb
$E1=horizontal msb
$E2=vertical
DRAW F601 (62977)
Draw the shape pointed to by Y.X using the current HCOLOR. On entry
A=rotation factor.

All About Applesoft

Source: David T Craig
APPLESOFT INTERNAL ENTRY POINTS

XDRAW F65D (63069)
Draw the shape pointed to by Y,X by inverting the existing color of the dots the shape draws over. On entry, A=rotation factor.

SETHCOL F66C (63212)
Set the HIRES color to X. X must be less than 8.

SHLOAD F775 (63349)
Loads a shape table into memory from tape above MEMSIZ (HIMEM) and sets up the pointer at $E8.

CASSette ROUTINES
SAVE D800 (55472)
Save the program in memory to tape.
LOAD D8C9 (55497)
Load a program from tape.

VARTIO D880 (55536)
Set up A1 and A2 to save 3 bytes ($50-$52) for the length.

PROGIO D901 (55553)
Set up A1 and A2 to save the program text.

ERROR PROCESSOR ROUTINES
ERROR D412 (54290)
Checks ERRFLG and jumps to HNDL ERR if ONERR is active. Otherwise it prints ['c' or 'r'] '?' [error message $X] 'ERROR'. If this is during program execution then it also prints 'IN' and the CURLIN.

HANDLER P2E9 (62185)
Saves CURLIN in ERRLIN, TXTPTR in ERRPOS, X in ERRNUM, and REMSTK in ERRSTK. REMSTK is equal to the 6502 stack pointer and is set up at the start of each statement. X contains the error code. This may be used to interrupt the execution of a BASIC program. See the Applesoft Reference Manual page 136 for the value of X for a given error.

RESUME F317 (62231)
Restores CURLIN from ERRLIN and TXTPTR from ERRPOS and transfers ERRSTK into the 6502 stack pointer.

SYNTAX CHECKING ROUTINES
ISCNTC D858 (55384)
Checks the Apple keyboard for a control — C ($83). Executes the BREAK routine if there is a control — C.

CHKNUM DD6A (56682)
Make sure FAC is numeric. See CHKVAL.

CHKSTR DD6C (56684)
Make sure FAC is a string. See CHKVAL.

CHKVAL DD6D (56685)
Checks the result of the most recent FAC operation to see if it is a string or numeric variable. A TYPE MISMATCH ERROR results if FAC and C don’t agree.

On entry:
C set checks for strings
C clear checks for numerics

ERRDIR E306 (58118)
Causes ILLEGAL DIRECT ERROR if the program isn’t running. X is modified.

ISLETC E07D (57469)
Checks A for an ASCII letter (‘A’ to ‘Z’). On exit C set if A is a letter.

PARCHK DEB2 (57010)
Checks for ‘.’, evaluates a formula, and checks for ‘)’. Uses CHKOPN and FRMEVL then falls into CHKCLS.

CHKCLS DEB8 (57016)
Checks at TXTPTR for ‘)’. Uses SYNCHR.

CHKOPN DEBB (57019)
Checks at TXTPTR for ‘,’. Uses SYNCHR.

CHKCOM DEBE (57022)
Checks at TXTPTR for ‘,’. Uses SYNCHR.

SYNCHR DECO (57024)
Checks at TXTPTR for the character in A. TXTPTR is not modified. Normally exits through CHRGET. Exits with SYNTAX ERROR if they don’t match.

A utility for A.P.P.L.E. members

AP File: TEXT FILE UTILITIES

by Wes Huntress

Includes:
- Text File Editor — Reader — Lister
- Read Diskette Sector or RAM Memory
- Copies Both Binary and Text Files
- Runs in l. Basic or FP * Uses Single Drive

$18.50
A.P.P.L.E. ORDERS
304 Main Ave. S., Suite 300
Renton, WA 98055

CALL A.P.P.L.E. In Depth

“DTCA2DOC-043-06.PICT” 260 KB 2001-04-03 dpi: 300h x 300v pix: 2244h x 3018v

Source: David T Craig

Page 0007 of 0008
---A---
A1  3C3D          51
A2  3E3F          51
ABS  EBAF         53
ADDR  D998        55
ARYTAB  68.6C     51
ATN  F09E         53
AYINT  E10C        53

---B---
BKGD  F3F2        55
BLTU  D393        55
BUF  290.2FF      55

---C---
CAT  E597         54
CHARAC  OD         51
CHKCLS  D6B8       56
CHKCOM  D6E8       56
CHKNUM  D66A       56
CHKOPN  D6BB       56
CHKSTR  D6C6       56
CHKVAL  D6D6       56
CHRGET  00B1       52
CHRGET  00B7       52
CLEARC  D6EC       55
COMBYTE  E74C      53
CONINT  06FB       53
CONINT  D0BA       54
CONUPK  E9E3       53
COPY  DAB7         54

---D---
CRDO  E6FA        55
CURLIN  75.76      51

---E---
ENDCHR  0E0        51
ERDRIR  E506       56
ERFILG  D58         51
ERRLIN  DAD.B       51
ERNNUM  DE0         51
ERROR  D412        56
ERRPOS  DC.DD      51
ERRSTK  DE0         51
EXP  ER09         53

---F---
FADD  E7BE        52
FADDH  E7A0        52
FADDT  E7C1        52
FBUFRR  100-1FP     51
FCOMP  EBB2        53
FDIV  EA66         52
FDIVT  E6A9        52
FIN  EC4A         53
FIRST  FO          53
FLOAT  EB53        52
FMULT  E37F        52
FMULTT  E982       55
FDLIN  D61A        55
FORPNT  85.86      53
FORR    ED04        53
FPWR    EE97        52

---M---
MEMSIZ  73.74      51
MESSZ  EB21        53
MOVIF  EB1E        53
MOV2F  E63         53
MOVAF  EB31        53
MOVMA  EB33        53
MOVPM  EAF9        53
MOGSN  E3D4        53
MOVMF  EB23        53
MOVML  EB23        53
MOVSTR  E5E2       54
MULTI  EA39        53

---N---
NEGEP  EED0        53
NEWSTT  D7D2       53
OLDLIN  77.78      51
ORMASK  F3          51
OUTDO  DB5C        54
OUTQST  DB5A        54
OUTSPC  DB57        54
PARCHK  DB2B        53
PLOTFS  F1EC        52
PRGEND  AF.BO      51
PROGIO  D901        56
PRNFAC  ED2E        54
PRTGRT  DF2E        55
PUTNEW  E42A        54

---Q---
QINT  EB2B        53
REASON  D363        55
REMNS  D9A6        55
REMSKT  F8          51
RESOR  D649        55
RESUME  F317        56
RNTN  EFAE         53

---R---
SAVE  D8BO        56
SCALE  E7          51
SCRCH  D64B        56
SETHOL  F6EC        56
SGN  EB80         53
SHLDF  F775        56
SIGN  EB82         53
SINF  EFFF         53
SNGFST  E301        55
SPDBYR  E1         51

---T---
TAN  F03A         53
TEMPP  52          51
TXTPT  B8.B9       51
TTXTAB  67.68      51
V2  2D           51
VALTP  11          51
VARTN  83.84      51
VARTAB  99.6A      51
VARTIO  D8FO       56
XDRAW  F6SD        56

All About Applesoft

Source: David T Craig