

## Developer Technical Support

## Apple II Miscellaneous#10:80-Column GetChar Routine

Revised by: Dave Lyons Written by: Cameron Birse September 1989 December 1986

This Technical Note presents an 80-column GetChar routine. Changes since November 1988: Added discussion of single-character input on the unenhanced Apple IIe.

The following is an example of how to display a string on the 80-column screen, reposition the cursor at the beginning of the string, and use the right arrow to get characters which are already there or accept new characters in their place. The routine is a simple BASIC program which displays the string and repositions the cursor before getting incoming characters. If the character input is a right arrow, the program calls the assembly-language routine to get the character from screen memory at the current cursor location.

```
10 PRINT CHR$ (4); "bload getchar.0": REM first install assembly routine
20 B$ = "hello"
30 PRINT CHR$ (4);"pr#3"
40 PRINT B$;:B$ = ""
50 A = PEEK (1403): REM get horiz location
60 A = A - 5: REM move cursor to beginning of string
70 POKE 1403,A
80 GET A$: REM get a character
90 IF A$ = CHR$ (21) THEN GOSUB 130: REM
                                            if char is forward arrow,
   handle with assembly routine (GETCHAR)
100 IF A$ = CHR$ (27) THEN 170: REM if esc key then we're done
110 PRINT A$;:B$ = B$ + A$
120 GOTO 80
130 CALL 768: REM
                   GETCHAR
140 A = PEEK (6)
150 A$ = CHR$ (A)
    RETURN
160
    PRINT : PRINT : PRINT B$: REM and we're done
170
```

An assembled listing of the assembly language GetChar routine follows. It works on the Apple IIe and later.

SOURCE FIL	LE #01 =>	•GETCH	IAR			
NEXT C	BJECT FI	LE NA	AME IS GE	TCHAR.	. 0	
0300:	0300	1		ORG	\$300	
0300:	C01F	2 RE	D80VID	EQU	\$C01F	;80 COLUMN STATE
0300:	C054	3 ТХ	KTPAGE1	EQU	\$C054	;TURN OFF PAGE 2 (READ)
0300:	C055	4 TX	<b>KTPAGE2</b>	EQU	\$C055	TURN ON PAGE 2 (READ)
0300:	C000	5 CI	R80COL	EOU	\$C000	TURN OFF 80 STORE (WRITE)
0300:	C001	6 SF	TODOT	EOU	sc001	TURN ON 80 STORE (WRITE)
0300:	0028	7 BA	AST.	EOU	\$28	BASE ADDRESS OF SCREEN LOCATION
0300.	0029	8 BA	ASH	EOU	\$29	/21102 11001200 01 001201 20011101
0300.	057B	9 01	IRCH	EOU	\$57B	•80 COLUMNS HORTZ POSTTION
0300.	05FB	10 01	IRCV	ean	\$5fb	:80 col vertical pos
0300.	0006	11 0	har	ogu	6	inlace to hand character back to
bagig	0000	11 0	JIAL	equ	0	, prace to hand character back to
Dasic		10 +				
0300:		12 ^				
0300:		13 **	~~~~~	××××××	• • • • • • • •	******
0300:		14 *	GETCHA	.R – Tr	iis routine get	s an ascil character from the *
0300:		15 *	80 COI	umn dı	lsplay memory o	of the Apple IIe. It assumes *
0300:		16 *	that m	ain me	emory is switch	ned in and that the base addrs *
0300:		17 *	of the	e line	has already be	een calculated and resides *
0300:		18 *	in BAS	L and	BASH. It is me	eant to be called from BASIC *
0300:		19 *	as fol	lows:		*
0300:		20 *			CALL 768	} *
0300:		21 *			A = PEEH	く (6) *
0300:		22 *			A\$ = CHE	R\$ (A) *
0300:		23 *	As you	can s	see, the charac	cter is returned in location *
0300:		24 *	\$6 in	zero p	bage. This rout	ine is offered as an example. *
0300:		25 *	No qua	rantie	es are made rec	arding its fitness for any *
			5			
0300:		26 *	purpos	e.	BV Came	eron Birse 6/10/86 *
0300:		26 * 27 **	purpos *******	e. *****	By Came	eron Birse 6/10/86 *
0300: 0300: 0300:		26 * 27 ** 28 *	purpos	e. *****	By Came	eron Birse 6/10/86 *
0300: 0300: 0300: 0300:	0300	26 * 27 ** 28 * 29	purpos	eau	By Came ************************************	eron Birse 6/10/86 *
0300: 0300: 0300: 0300: cursor loc	0300	26 * 27 ** 28 * 29	purpos ******** getchr	e. ****** equ	By Came	get the char at the current
0300: 0300: 0300: 0300: cursor loc.	0300	26 * 27 ** 28 * 29	purpos	equ	By Came ************************************	ron Birse 6/10/86 * ;get the char at the current .mask for horiz test
0300: 0300: 0300: 0300: cursor loc. 0300:A9 01	0300	26 * 27 ** 28 * 29 30	purpos	e. ****** equ lda	By Came ************************************	; get the char at the current
0300: 0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0	0300 05	26 * 27 ** 28 * 29 30 31	purpos ********* getchr	e. equ lda bit	By Came ************************************	<pre>;get the char at the current ;mask for horiz test ;are we in main or aux mem? ;if bit 0. of OUDQU is set then</pre>
0300: 0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17	0300 05 031E	26 * 27 ** 28 * 29 30 31 32	purpos ********* getchr	e. equ lda bit bne	By Came ************************************	<pre>ron Birse 6/10/86 * * ron Birse 6/10/86 * ;get the char at the current ;mask for horiz test ;are we in main or aux mem? ;if bit 0 of OURCH is set, then</pre>
0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem	0300 05 031E	26 * 27 ** 28 * 29 30 31 32	purpos ********* getchr	e. equ lda bit bne	By Came ************************************	<pre>ron Birse 6/10/86 * ;get the char at the current ;mask for horiz test ;are we in main or aux mem? ;if bit 0 of OURCH is set, then</pre>
0300: 0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem 0307:	0300 031E 0307	26 * 27 ** 28 * 29 30 31 32 33 au	purpos ********* getchr	e. ****** equ lda bit bne equ	By Came ************************************	<pre>ron Birse 6/10/86 * * reaction at the current ; mask for horiz test ; are we in main or aux mem? ; if bit 0 of OURCH is set, then </pre>
0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem 0307: 0307:AD 7B 0	0300 031E 0307	26 * 27 ** 28 * 29 30 31 32 33 au 34	getchr	e. ****** equ lda bit bne equ lda	By Came ************************************	<pre>ron Birse 6/10/86 * ;get the char at the current ;mask for horiz test ;are we in main or aux mem? ;if bit 0 of OURCH is set, then ;get horiz pos.</pre>
0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem 0307: 0307:AD 7B 0 030A:18	0300 031E 0307	26 * 27 ** 28 * 29 30 31 32 33 au 34 35	getchr	e. ****** equ bit bne equ lda clc	By Came ************************************	<pre>;get the char at the current ;mask for horiz test ;are we in main or aux mem? ;if bit 0 of OURCH is set, then ;get horiz pos. ;clear the carry for divide</pre>
0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem 0307: 0307:AD 7B 0 030A:18 030B:6A	0300 031E 0307 05	26 * 27 ** 28 * 29 30 31 32 33 au 34 35 36	getchr	e. ******* lda bit bne equ lda clc ror	By Came ************************************	<pre>;get the char at the current ;mask for horiz test ;are we in main or aux mem? ;if bit 0 of OURCH is set, then ;get horiz pos. ;clear the carry for divide ;divide by two</pre>
0300: 0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem 0307: 0307:AD 7B 0 030A:18 030B:6A 030C:A8	0300 031E 0307 05	26 * 27 ** 28 * 29 30 31 32 33 au 34 35 36 37	purpos ***** getchr	e. ******* equ lda bit bne equ lda clc ror tay	By Came ************************************	<pre>;get the char at the current ;mask for horiz test ;are we in main or aux mem? ;if bit 0 of OURCH is set, then ;get horiz pos. ;clear the carry for divide ;divide by two ;put the result in y</pre>
0300: 0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem 0307: 0307:AD 7B 0 030A:18 030B:6A 030C:A8 030D:8D 01 C	0300 031E 0307 05	26 * 27 ** 28 * 29 30 31 32 33 au 34 35 36 37 38	purpos ********* getchr 1x	e. ******* equ bit bne equ lda clc ror tay sta	By Came ************************************	<pre>ron Birse 6/10/86 * ;get the char at the current ;mask for horiz test ;are we in main or aux mem? ;if bit 0 of OURCH is set, then ;get horiz pos. ;clear the carry for divide ;divide by two ;put the result in y ;turn on 80 store</pre>
0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem 0307: 0307:AD 7B 0 030A:18 030B:6A 030C:A8 030D:8D 01 C 0310:AD 55 C	0300 05 031E 0307 05	26 * 27 ** 28 * 29 30 31 32 33 au 34 35 36 37 38 39	purpos ********* getchr	e. ******* lda bit bne equ lda clc ror tay sta lda	By Came ************************************	<pre>ron Birse 6/10/86 * ;get the char at the current ;mask for horiz test ;are we in main or aux mem? ;if bit 0 of OURCH is set, then ;get horiz pos. ;clear the carry for divide ;divide by two ;put the result in y ;turn on 80 store ;flip to aux text page</pre>
0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem 0307: 0307:AD 7B 0 030A:18 030B:6A 030C:A8 030D:8D 01 C 0310:AD 55 C 0313:B1 28	0300 031E 0307 05	26 * 27 ** 28 * 29 30 31 32 33 au 34 35 36 37 38 39 40	getchr	e. ******* equ lda bit bne equ lda clc ror tay sta lda lda	By Came ************************************	<pre>ron Birse 6/10/86 * ;get the char at the current ;mask for horiz test ;are we in main or aux mem? ;if bit 0 of OURCH is set, then ;get horiz pos. ;clear the carry for divide ;divide by two ;put the result in y ;turn on 80 store ;flip to aux text page ;get the character</pre>
0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem 0307: 0307:AD 7B 0 030A:18 030B:6A 030D:8D 01 C 0310:AD 55 C 0313:B1 28 0315:85 06	0300 031E 0307 05	26 * 27 ** 28 * 29 30 31 32 33 au 34 35 36 37 38 39 40 41	getchr	e. ******* equ lda bit bne equ lda clc ror tay sta lda lda sta	By Came ************************************	<pre>ron Birse 6/10/86 * ;get the char at the current ;mask for horiz test ;are we in main or aux mem? ;if bit 0 of OURCH is set, then ;get horiz pos. ;clear the carry for divide ;divide by two ;put the result in y ;turn on 80 store ;flip to aux text page ;get the character</pre>
0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem 0307: 0307:AD 7B 0 030A:18 030B:6A 030D:AD 75 0 0310:AD 55 0 0313:B1 28 0315:85 06 0317:AE 54 0	0300 05 031E 0307 05	26 * 27 ** 28 * 29 30 31 32 33 au 34 35 36 37 38 39 40 41 42	purpos ********* getchr 1x	e. ******* equ lda bit bne equ lda clc ror tay sta lda lda sta lda	By Came ************************************	<pre>ron Birse 6/10/86 * * ;get the char at the current ;mask for horiz test ;are we in main or aux mem? ;if bit 0 of OURCH is set, then ;get horiz pos. ;clear the carry for divide ;divide by two ;put the result in y ;turn on 80 store ;flip to aux text page ;get the character ;turn off aux text page</pre>
0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem 0307: 0307:AD 7B 0 030A:18 030B:6A 030C:A8 030D:8D 01 C 0310:AD 55 C 0317:AE 54 C 031A:8D 00 C	0300 031E 0307 05	26 * 27 ** 28 * 29 30 31 32 33 au 34 35 36 37 38 39 40 41 42 43	getchr	e. ******* equ lda bit bne equ lda clc ror tay sta lda lda sta ldx sta	By Came ************************************	<pre>ron Birse 6/10/86 * * ;get the char at the current ;mask for horiz test ;are we in main or aux mem? ;if bit 0 of OURCH is set, then ;get horiz pos. ;clear the carry for divide ;divide by two ;put the result in y ;turn on 80 store ;flip to aux text page ;get the character ;turn off aux text page ;turn off 80 store</pre>
0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem 0307: 0307:AD 7B 0 030A:18 030B:6A 030C:A8 030D:8D 01 C 0310:AD 55 C 0313:B1 28 0315:85 06 0317:AE 54 CC 031A:8D 00 CC 031D:60	0300 031E 0307 05	26 * 27 ** 28 * 29 30 31 32 33 au 34 35 36 37 38 39 40 41 42 43 44	purpos	e. ******* equ lda bit bne equ lda clc ror tay sta lda lda sta lda sta ldx sta rts	By Came ************************************	<pre>ron Birse 6/10/86 * ;get the char at the current ;mask for horiz test ;are we in main or aux mem? ;if bit 0 of OURCH is set, then ;get horiz pos. ;clear the carry for divide ;divide by two ;put the result in y ;turn on 80 store ;flip to aux text page ;get the character ;turn off aux text page ;turn off 80 store</pre>
0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem 0307: 0307:AD 7B 0 030A:18 030B:6A 030C:A8 030D:8D 01 C 0310:AD 55 C 0313:B1 28 0315:85 06 0317:AE 54 C 031A:8D 00 C 031D:60 031E:	0300 031E 0307 05	26 * 27 ** 28 * 29 30 31 32 33 au 34 35 36 37 38 39 40 41 42 43 44 44 45 ma	getchr	e. ******* equ lda bit bne equ lda clc ror tay sta lda lda sta lda sta lda sta rts equ	By Came ************************************	<pre>ron Birse 6/10/86 * ;get the char at the current ;mask for horiz test ;are we in main or aux mem? ;if bit 0 of OURCH is set, then ;get horiz pos. ;clear the carry for divide ;divide by two ;put the result in y ;turn on 80 store ;flip to aux text page ;get the character ;turn off aux text page ;turn off 80 store</pre>
0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem 0307: 0307:AD 7B 0 030A:18 030B:6A 030C:A8 030D:8D 01 C 0310:AD 55 C 0313:B1 28 0315:85 06 0317:AE 54 C 031A:8D 00 C 031D:60 031E: 031E:AD 7B 0	0300 031E 0307 05 20 20 20 20 20 20 20 20 20 20 20 20 20	26 * 27 ** 28 * 29 30 31 32 33 au 34 35 36 37 38 39 40 41 42 43 44 45 ma	getchr	e. ******* equ lda bit bne equ lda clc ror tay sta lda lda sta lda sta rts equ lda	By Came ************************************	<pre>ron Birse 6/10/86 * * *********************************</pre>
0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem 0307: 0307:AD 7B 0 030A:18 030B:6A 030C:A8 030D:8D 01 C 0310:AD 55 C 0313:B1 28 0315:85 06 0317:AE 54 C 031A:8D 00 C 031D:60 031E: 031E:AD 7B 0 0321:18	0300 031E 0307 05 20 20 20 20 20 20 20 20 20 20 20 20 20	26 * 27 ** 28 * 29 30 31 32 33 au 34 35 36 37 38 39 40 41 42 43 44 45 ma 46 47	getchr	e. ******* equ lda bit bne equ lda clc ror tay sta lda lda sta lda sta lda sta lda clc ror tay sta lda clc clc ror tay sta lda clc clc clc clc clc clc clc clc clc cl	By Came ************************************	<pre>ron Birse 6/10/86 * ;get the char at the current ;mask for horiz test ;are we in main or aux mem? ;if bit 0 of OURCH is set, then ;get horiz pos. ;clear the carry for divide ;divide by two ;put the result in y ;turn on 80 store ;flip to aux text page ;get the character ;turn off aux text page ;turn off 80 store ;get horiz pos. ;clear the carry for divide</pre>
0300: 0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem 0307: 0307:AD 7B 0 030A:18 030B:6A 030C:A8 030D:8D 01 C 0310:AD 55 C 0313:B1 28 0315:85 06 0317:AE 54 C 031A:8D 00 C 031D:60 031E: 031E:AD 7B 0 0321:18 0322:6A	0300 031E 0307 05	26 * 27 ** 28 * 29 30 31 32 33 au 34 35 36 37 38 39 40 41 42 43 44 45 ma 46 47 48	getchr	e. ******* equ lda bit bne equ lda clc ror tay sta lda lda sta lda sta lda clc ror tay sta lda clc ror tay sta lda clc ror tay sta lda clc ror tay sta lda ror to ror tay sta lda ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ror to ro to ro ro to ro ro ror to ro ro ro ro ro ro ro ro ro ro ro ro ro	By Came ************************************	<pre>ron Birse 6/10/86 * ;get the char at the current ;mask for horiz test ;are we in main or aux mem? ;if bit 0 of OURCH is set, then ;get horiz pos. ;clear the carry for divide ;divide by two ;put the result in y ;turn on 80 store ;flip to aux text page ;get the character ;turn off aux text page ;turn off 80 store ;get horiz pos. ;clear the carry for divide .divide by two</pre>
0300: 0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem 0307: 0307:AD 7B 0 030A:18 030B:6A 030C:A8 030D:8D 01 C 0310:AD 55 C 0313:B1 28 0315:85 06 0317:AE 54 C 031A:8D 00 C 031D:60 031E: 031E:AD 7B 0 0321:18 0322:6A 0323:A8	0300 031E 0307 05	26 * 27 ** 28 * 29 30 31 32 33 au 34 35 36 37 38 39 40 41 42 43 44 45 ma 46 47 48 49	getchr	e. ******* equ lda bit bne equ lda clc ror tay sta lda lda sta lda sta lda sta lda clc ror tay sta lda clc ror tay sta lda clc ror tay sta lda to sta lda to sta lda to sta lda to sta lda to sta lda to sta lda to sta lda to sta lda to sta lda to sta lda to sta lda to sta lda to sta lda to sta lda to sta lta lta sta lta lta sta lta lta sta lta lta sta lta lta lta lta lta lta lta lta lta l	By Came ************************************	<pre>ron Birse 6/10/86 * ;get the char at the current ;mask for horiz test ;are we in main or aux mem? ;if bit 0 of OURCH is set, then ;get horiz pos. ;clear the carry for divide ;divide by two ;put the result in y ;turn on 80 store ;flip to aux text page ;get the character ;turn off aux text page ;turn off 80 store ;get horiz pos. ;clear the carry for divide ;divide by two ;put the result in y ;turn off 80 store ;get horiz pos. ;clear the carry for divide ;divide by two ;put the result in y ;</pre>
0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem 0307: 0307:AD 7B 0 030A:18 030B:6A 030C:A8 030D:8D 01 C 0310:AD 55 C 0313:B1 28 0315:85 06 0317:AE 54 C 031A:8D 00 C 031B: 0 C 031E: 031E:AD 7B 0 0321:18 0322:6A 0323:A8 0324:P1 28	0300 031E 0307 05	26 * 27 ** 28 * 29 30 31 32 33 au 33 au 35 36 37 38 39 40 41 42 43 44 45 ma 46 47 48 49 50	getchr	e. ******* equ lda bit bne equ lda clc ror tay sta lda lda sta lda sta lda clc ror tay sta lda clc ror tay sta lda lda sta lda lda sta lda lda clc ror tay sta lda lda lda clc ror tay sta lda lda lda lda sta lda lda lda lda lda lda lda lda lda ld	By Came ************************************	<pre>ron Birse 6/10/86 * ;get the char at the current ;mask for horiz test ;are we in main or aux mem? ;if bit 0 of OURCH is set, then ;get horiz pos. ;clear the carry for divide ;divide by two ;put the result in y ;turn on 80 store ;flip to aux text page ;get the character ;turn off aux text page ;turn off 80 store ;get horiz pos. ;clear the carry for divide ;divide by two ;put the result in y ;ent the result in y ;ent the carry for divide ;divide by two ;put the result in y ;clear the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carry for divide ;divide by two ;put the result in y ;ent the carr</pre>
0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem 0307: 0307:AD 7B 0 030A:18 030D:8D 01 C 0310:AD 55 C 0313:B1 28 0315:85 06 0317:AE 54 C 031A:8D 00 C 031D:60 031E: 031E:AD 7B 0 0321:18 0322:6A 0323:A8 0324:B1 28 0324:B1 28	0300 031E 0307 05	26 * 27 ** 28 * 29 30 31 32 33 au 34 35 36 37 38 39 40 41 42 43 44 45 ma 46 47 48 49 50	getchr	e. ******* equ lda bit bne equ lda clc ror tay sta lda lda sta lda sta lda sta lda clc ror tay sta lda clc ror tay sta lda clc ror tay sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta lda sta sta lda sta sta lda sta lda sta sta lda sta sta sta sta sta sta sta sta sta st	By Came ************************************	<pre>ron Birse 6/10/86 * * *********************************</pre>
0300: 0300: 0300: cursor loc. 0300:A9 01 0302:2C 7B 0 0305:D0 17 main mem 0307: 0307:AD 7B 0 030A:18 030D:8D 01 C 0310:AD 55 C 0313:B1 28 0315:85 06 0317:AE 54 C 031A:8D 00 C 031B: 031E: 031E:AD 7B 0 0321:18 0322:6A 0323:A8 0324:B1 28 0326:85 06 0326:85 06 0320:62	0300 031E 0307 05	26 * 27 ** 28 * 29 30 31 32 33 au 34 35 36 37 38 39 40 41 42 43 40 41 42 43 44 45 ma 46 47 48 49 50 51 52	getchr	e. ******* equ lda bit bne equ lda clc ror tay sta lda lda sta lda sta lda sta lda sta lda sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta ida sta sta ida sta ida sta ida sta ida sta ida sta ida sta sta ida sta sta ida sta sta sta sta sta sta sta sta sta st	By Came ************************************	<pre>ron Birse 6/10/86 * * *********************************</pre>

## **Reading a Single Character**

While the 80-column firmware is active (whether in 40- or 80-column mode), the RDKEY routine on the unenhanced Apple IIe unexpectedly allows the user to press ESC and move the cursor around the screen the same way RDCHAR does.

AppleSoft's GET statement uses RDKEY, so it behaves the same way. The ESC keypress is never returned, so users have problems if you use GET and expect them, for example, to press ESC to return to the previous menu. At this point, the cursor turns into an inverse plus sign (+) and your program is still waiting for a keypress. The user presses ESC a few more times, watching the cursor alternate between an inverse plus sign and an inverse blank, and then turns off the computer in search of a more exciting activity, like throwing darts at your disk.

If your program can run on the unenhanced IIe, either leave the 80-column firmware turned off (PRINT CHR\$(21) to make sure it's off), or read keypresses by polling the keyboard register directly:

 1000 IF PEEK(-16384)<128 THEN 1000</td>
 : REM Wait for a keypress

 1010 A\$ = CHR\$(PEEK(-16384)-128)
 : REM Read the key

 1020 POKE -16368,0
 : REM Clear the keyboard strobe

or

 0300: LDA \$C000
 ; check for a keypress

 0303: BPL \$0300
 ; keep waiting

 0306: AND #\$7F
 ; turn off bit 7

 0308: STA \$C010
 ; clear the keyboard strobe

Note that these code fragments don't display a cursor while waiting for a key.

## **Further Reference**

- Apple IIGS Firmware Reference
- Apple IIe Technical Reference Manual
- Apple IIc Technical Reference Manual, Second Edition