Chapter 1

Overview

- Future 65C816 Support

Chapter 1: Overview

The Commander X16 is a modern home computer in the philosophy of Commodore computers like the VIC-20 and the C64.

Features:

- 8-bit 65C02S CPU at 8 MHz (*)
- 512 KB banked RAM (upgradeable to 2 MB on the X16 Developer Edition)
- 512 KB ROM
- Expansion Cards (Gen 1) & Cartridges (Gen 1 and Gen 2)
  - Up to 3.5MB of RAM/ROM
  - 5 32-byte Memory-Mapped IO slots
- VERA video controller
  - Up to 640x480 resolution
  - 256 colors from a palette of 4096
  - 128 sprites
  - VGA, NTSC and RGB output
  - Powered by a Lattice ICE40UP5K FPGA
- Three sound sources
  - Yamaha YM2151: 8 channels, 4-operator FM synthesis
  - VERA PSG: 16 channels, 4 waveforms
  - VERA PCM: Up to 48 kHz, 16 bit, stereo

- Connectivity:
  - PS/2 keyboard and mouse
  - 4 NES/SNES controllers
  - SD card
  - Commodore Serial Bus ("IEC")
  - Many Free GPIOs ("user port")

As a modern sibling of the line of Commodore home computers, the Commander X16 is reasonably compatible with computers of that line.

- Pure BASIC programs are fully backwards compatible with the VIC-20 and the C64.
- POKEs for video and audio are not compatible with any Commodore computer. (There are no VIC or SID chips, for example.)
- Pure machine language programs ($FF81+ KERNAL API) are compatible with Commodore computers.

Future 65C816 Support

A future upgrade path for the X16 may involve the 65C816. It is almost fully compatible with the 65C02 except for 4 instructions (`BBRx`, `BBSx`, `RMBx`, and `SMBx`). It is advisable not to use these instructions when writing programs for the X16.

<!– For PDF formatting –>

Chapter 2

Getting Started

- Finding and starting programs
- Using The Keyboard
- Special Keys
- WHAT IS

Chapter 2: Getting Started

This is a brief guide to your first few minutes on the Commander X16. For a complete New User experience, please refer to the Commander X16 User Guide.

Finding and starting programs

When starting your Commander X16, you'll notice that it's not like other computers. There is no GUI, and command line commands like `DIR` or `LS` don't get you anywhere. Here are some quick tips to getting started:

The Commander X16 uses a full screen interface known as "Editor". This was unique when it was introduced on the PET in 1977, when most computers still treated the screen as if it was a teletype display. The full screen Editor lets you use the arrow keys to move around the screen and edit or re-enter input from previous interactions.

The first thing you will want to do is view a list of files on your SD card. Type the below command and press the Return key (or Enter key) to see a list of files:

DOS

```
DOS "$"
```

You can also type `@$` and press Return. All of the commands you type must be followed by the Return or Enter key, to actually execute the command.

Let's try some variations on this command:

`DOS "$=D"` lists just the subdirectories in the current directory.

To get the DOS command a little faster, try pressing F8, then

typing `$=D`. You can even leave off the last quote; DOS doesn't care.

So now that you have a list of directories, try moving to one:

```
DOS "CD:BASIC"
```

Press F7 or type `DOS"$` again to list the files in this directory. A file with .PRG at the end is a "Program" file and can be loaded with the LOAD command. The shortcut for LOAD" is the F3 key. Try it now:

LOAD

```
LOAD "MAD.PRG"
```

You can see that the program list loaded by typing `LIST` and pressing Enter.

RUN

Now type `RUN` and Enter. This will start the loaded program.

STOP

STOP isn't a command: it's a key. Press it to stop the running program.

If you are using the official Commander X16 keyboard, the key is labeled `RUN STOP` and is up near the upper right corner of the keyboard.

If you are using a PC keyboard, it's probably labeled `Pause` or `Pause Brk`.

Holding Control and pressing C (also written as `Control+C` or `^C`) will also stop the running program.

There are a few ways to get back to the text screen, but the quickest is to hold Control, Alt, and press the Del key. (Or just press the RESET button.)

Using The Keyboard

The Commander X16's keyboard is a little different than a standard PC: there are three distinct modes of operation, and the keyboard can create graphic symbols known as PETSCII characters. There are also some special keys used for controlling the computer.

PETSCII Characters

When the system first boots up, the X16 will be in PETSCII Upper Case/Graphic mode. Pressing a letter key without the shift key will generate an upper case letter. Unlike a PC or Mac, this mode does not have any lower case text, so everything you type is UPPER CASE.

Now, notice the extra symbols on your keycaps? There are two sets of extra symbols: the ones on the lower-right can be accessed by holding SHIFT and a letter. Go ahead: try pressing Shift and S. You should see a small heart symbol on your screen. We know you'll love the Commander X16 as much as we do. Press Alt and a letter, and you'll get the symbol on the lower-left corner of the screen. Try pressing Alt and the ' key next to the number 1. You should get a large + symbol. One of the Plusses of PETSCII is using these line drawing symbols to draw shapes on the screen.

You can also change colors by pressing Control and a number. Go ahead: Press Control+1 and type a few letters. Notice they come out in black. Now try Alt+1. Notice the cursor changes to orange, and notice the next thing you type comes out orange.

You can also use Control+9 to turn on Reverse Print and Control+0 to turn it off.

There are some unexpected changes to the PC keyboard layout, as follows:

- The Grave key (‘) prints a left arrow (←) symbol.
- Shift+Grave prints the Pi symbol (π). This is actually the constant "pi". Try it by typing `PRINT` and RETURN.
- Shift+6 prints an up arrow (↑)
- The key prints the British Pound (£).
- The pipe (|) is replaced with a triangle corner symbol.
- { and } are replaced with two box drawing symbols.
- Underline (Shift+-) is replaced with a | symbol.

Note that programming languages that need {, }, and _ will alter the character set to show those symbols on the appropriate keys when needed. Or you can use ISO mode when editing C code in EDIT.

Lower Case Mode

WORKING IN PETSCII MIGHT MAKE PEOPLE THINK YOU'RE YELLING ALL THE TIME. Fortunately, there's an upper/lower case mode, too: Hold the Alt key and tap Shift to activate lower case. Notice that the upper case letters shift to lower case, and the shifted graphic symbols (such as the heart) shift to upper case letters. The tradeoff of upper/lower case mode is that half of the graphic symbols are unavailable, but you you get lower case letters.

Now try typing a command. `print "Hello World"` and press RETURN. Notice that you need to type `print` in lower case. If

you did it right, you should see "Hello World" appear on the next line.

Now tap Alt+Shift again. The text will change to |ELLO oORLD. Again, this is the tradeoff: you can have the Shifted graphic symbols or lower case, but not both.

ISO Mode

Finally, the computer has ISO mode. The ISO mode character set operates more like a PC, with upper case text, lower case text, and an assortment of accented and other letters. In addition, ISO mode has the ¦, ¬, {, and } symbols, which are not available in PETSCII modes. ISO mode is useful when you need PC compatibility or want the letters with accents. Elsewhere in this guide, we have a full manual on using the Right Alt key to compose accented symbols, like é or ō. Getting back to PETSCII mode from ISO mode is a little more complicated. Press Control+Alt+RESTORE (or Control+Alt+PrintScreen) to warm start BASIC and switch back to PETSCII mode.

EDIT text modes

The built-in EDIT utility includes a character set mode switch: Press Control+E to cycle through Upper/Graphic, Upper/Lower, and ISO mode.

Special Keys

RUN STOP

This key actually has two separate functions: "RUN" and "STOP". Holding Shift+RUN will load the first program on your SD card and automatically run it. If you are using the SD card that

came with your Commander X16, this will print some information on getting started with your computer.

If you are running a BASIC program, pressing STOP will stop the program.

RESTORE

As mentioned above, RESTORE can be used with Control+Alt to perform a warm start of BASIC. Less drastic than a cold boot, this stops a running program and returns you to the `READY.` prompt. If you had a BASIC program loaded, you can still re-start it with RUN or view it with LIST.

Control+Alt_Delete

Yes, the Commander X16 has the famous "3 fingered salute." This performs a cold boot of the computer, including a full power cycle. You will be returned to the boot screen, and if you have an AUTOEXEC.X16, it will execute on startup.

40/80 DISPLAY

This switches the computer between 80x60 text mode and 40x30 text mode. 40x30 is more useful on CRT screens, so you may want to boot up into 40x30 mode. You can set these modes with BASIC by typing

`SCREEN 1` or `SCREEN 3`.

Protip: you can force your computer to start in 40-column mode by modifying your AUTOBOOT.X16 file:

```
LOAD "AUTOBOOT.X16"
0 SCREEN 3
SAVE "@:AUTOBOOT.X16"
BOOT
```

Don't worry, if you don't like this change, you can change it back:

```
LOAD "AUTOBOOT.X16"
SCREEN 1
SAVE "@:AUTOBOOT.X16"
BOOT
```

F-KEYS

The F-keys, also known as the "Function Keys" are pre-loaded with special shortcuts:

F1 `LIST` Displays your currently loaded BASIC program.

F2 `SAVE"@:` is a quick shortcut for saving a program. The @: allows you to overwrite an existing file with the same name.

F3 `LOAD "` helps you load a program. Protip: if you use @$ to get a directory listing, you can then use the arrow keys to move up to a line with a filename. Press F3 and press RETURN to load a file.

F4 and RETURN swaps between 40 and 80 column screen modes.

F5 `RUN` runs the currently loaded program

F6 `MONITOR` Runs the machine monitor. The monitor allows you to directly edit memory, view assembly language dumps, and even write short assembly language programs at your keyboard.

F7 `DOS"$` Lists the current directory

F8 `DOS"` allows you to enter a disk command, such as CD:. More info can be found in chapter 13.

WHAT IS PC RA RO AC XR YR SP NV#BDIZC?

There are times when the computer will drop to the MONITOR prompt. That looks like this:

```
C*
    PC   RA  RO  AC  XR  YR  SP  NV#BDIZC
.;E3BB  01  04  00  65  2B  F6  ........
.
```

This is the MONITOR screen. You can get there in BASIC by typing `MON`.

Type `X` and Enter to exit back to BASIC. If you just get bounced to MONITOR again, then you'll need to Control+Alt+Restore or Control+Alt+Delete to restore to a working state.

MONITOR is covered in Chapter 7.

<!– For PDF formatting –>

Chapter 3

Editor

- Modes
- ISO Mode
- Background Color
- Scrolling
- New Control Characters
- Keyboard Layouts
- Default Layout

Chapter 3: Editor

The X16 has a built-in screen editor that is backwards-compatible with the C64, but has many new features.

Modes

The editor's default mode is 80x60 text mode. The following text mode resolutions are supported:

Mode	Description
$00	80x60 text
$01	80x30 text
$02	40x60 text
$03	40x30 text
$04	40x15 text
$05	20x30 text
$06	20x15 text
$07	22x23 text
$08	64x50 text
$09	64x25 text
$0A	32x50 text
$0B	32x25 text
$80	320x240@256c 40x30 text

Mode $80 contains two layers: a text layer on top of a graphics screen. In this mode, text color 0 is translucent instead of black.

To switch modes, use the BASIC statement `SCREEN` or the KERNAL API `screen_mode`. In the BASIC editor, the F4 key toggles between modes 0 (80x60) and 3 (40x30).

<!– For PDF formatting –>

ISO Mode

In addition to PETSCII, the X16 also supports the ISO-8859-15 character encoding. In ISO-8859-15 mode ("ISO mode"):

- The character set is switched from Commodore-style (with PETSCII drawing characters) to a new ASCII/ISO-8859-15 compatible set, which covers most Western European writing systems.
- The encoding (`CHR$()` in BASIC and `BSOUT` in machine language) now complies with ASCII and ISO-8859-15.
- The keyboard driver will return ASCII/ISO-8859-15 codes.

This is the encoding:

	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE
0x															
1x															
2x		!	″	#	$	%	&	′	(	)	*	+	,	-	.
3x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>
4x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N
5x	P	Q	R	S	T	U	V	W	X	Y	Z	[		]	^
6x	‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n
7x	p	q	r	s	t	u	v	w	x	y	z	{	|	}	
8x															
9x															
Ax		¡	¢	£	€	¥	Š	§	š	©	ª	«	¬	⌧	®
Bx	°	±	²	³	Ž	µ	¶	·	ž	¹	º	»	Œ	œ	Ÿ
Cx	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î
Dx	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ
Ex	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î
Fx	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ

- The non-printable areas $AD-$1F and $80-$9F in the

character set are filled with inverted variants of the codes $40-$5F and $60-$7F, respectively.

- The code $AD is a non-printable soft hyphen in ISO-8859-15. The ROM character set contains the Commander X16 logo at this location.

ISO mode can be enabled and disabled using two new control codes:

- `CHR$($0F)`: enable ISO mode
- `CHR$($8F)`: enable PETSCII mode (default)

You can also enable ISO mode in direct mode by pressing Ctrl+`O`.

Important: In ISO mode, BASIC keywords need to be written in upper case, that is, they have to be entered with the Shift key down, and abbreviating keywords is no longer possible.

Background Color

In regular BASIC text mode, the video controller supports 16 foreground colors and 16 background colors for each character on the screen.

The new "swap fg/bg color" code is useful to change the background color of the cursor, like this:

```
PRINT CHR$(1);     : REM SWAP FG/BG
PRINT CHR$($1C);   : REM SET FG COLOR TO RED
PRINT CHR$(1);     : REM SWAP FG/BG
```

The new BASIC instruction `COLOR` makes this easier, but the trick above can also be used from machine code programs.

To set the background color of the complete screen, it just has to be cleared after setting the color:

```
PRINT CHR$(147);
```

Scrolling

The C64 editor could only scroll the screen up (when overflowing the last line or printing or entering DOWN on the last line). The X16 editor scrolls both ways: When the cursor is on the first line and UP is printed or entered, the screen contents scroll down by a line.

New Control Characters

This is the set of all supported PETSCII control characters. Entries in bold indicate new codes compared to the C64:

If there are two meanings listed, the first indicates input (a keypress) and the second indicates output.

Code			Co
$00	NULL	**VERBATIM MODE**	$8
$01	**SWAP COLORS**	COLOR: ORANGE	$8
$02	**PAGE DOWN**	**PAGE UP**	$8
$03	STOP	RUN	$8
$04	**END**	**HELP**	$8
$05	COLOR: WHITE	F1	$8
$06	**MENU**	F3	$8
$07	**BELL**	F5	$8
$08	DISALLOW CHARSET SW (SHIFT+ALT)	F7	$8
$09	**TAB** / ALLOW CHARSET SW	F2	$8
$0A	**LF**	F4	$8
$0B	**LAYOUT SWAP**	F6	$8
$0C	-	F8	$8
$0D	RETURN	SHIFTED RETURN	$8
$0E	CHARSET: LOWER/UPPER	CHARSET: UPPER/PETSCII	$8

$0F	**CHARSET: ISO ON**	**CHARSET: ISO OFF**
$10	**F9**	COLOR: BLACK
$11	CURSOR: DOWN	CURSOR: UP
$12	REVERSE ON	REVERSE OFF
$13	HOME	CLEAR
$14	DEL (PS/2 BACKSPACE)	INSERT
$15	**F10**	COLOR: BROWN
$16	**F11**	COLOR: LIGHT RED
$17	**F12**	COLOR: DARK GRAY
$18	**SHIFT+TAB**	COLOR: MIDDLE GRAY
$19	**FWD DEL (PS/2 DEL)**	COLOR: LIGHT GREEN
$1A	-	COLOR: LIGHT BLUE
$1B	ESC	COLOR: LIGHT GRAY
$1C	COLOR: RED	COLOR: PURPLE
$1D	CURSOR: RIGHT	CURSOR: LEFT
$1E	COLOR: GREEN	COLOR: YELLOW
$1F	COLOR: BLUE	COLOR: CYAN

Notes:

- $01: SWAP COLORS swaps the foreground and background colors in text mode
- $07/$09/$0A/$18/$1B: have been added for ASCII compatibility. [$0A/$18/$1B do not have any effect on output. Outputs of $08/$09 have their traditional C64 effect]
- $80: VERBATIM MODE prints the next character (only!) as a glyph without interpretation. This is similar to quote mode, but also includes codes CR ($0D) and DEL ($14).
- F9-F12: these codes match the C65 additions
- $84: This code is generated when pressing SHIFT+END.

- Additionally, the codes $04/$06/$0B/$0C are interpreted when printing in graphics mode using `GRAPH_put_char`.

<!– For PDF formatting –>

Keyboard Layouts

The editor supports multiple keyboard layouts.

Default Layout

On boot, the US layout (`ABC/X16`) is active:

- In PETSCII mode, it matches the US layout where possible, and can reach all PETSCII symbols.
- In ISO mode, it matches the Macintosh US keyboard and can reach all ISO-8859-15 characters. Some characters are reachable through key combinations:

Key	Result
Alt+1	¡
Alt+3	£
Alt+4	¢
Alt+5	§
Alt+7	¶
Alt+9	ª
Alt+0	º
Alt+q	œ
Alt+r	®
Alt+t	Þ
Alt+y	¥
Alt+o	ø
Alt+\	«
Alt+s	ß
Alt+d	ð

Alt+g	©
Alt+l	¬
Alt+'	œ
Alt+m	µ
Alt+/	÷
Shift+Alt+2	€
Shift+Alt+8	º
Shift+Alt+9	·
Shift+Alt+-	X16 logo
Shift+Alt+=	±
Shift+Alt+q	Œ
Shift+Alt+t	þ
Shift+Alt+\	»
Shift+Alt+a	¹
Shift+Alt+d	Ð
Shift+Alt+k	X16 logo
Shift+Alt+'	Æ
Shift+Alt+c	³
Shift+Alt+b	²
Shift+Alt+/	¿

(The X16 logo is code point, SHY, soft-hyphen.)

The following combinations are dead keys:

- Alt+<tt>‘</tt>
- Alt+6
- Alt+e
- Alt+u
- Alt+p
- Alt+a
- Alt+k

- Alt+`;`
- Alt+`x`
- Alt+`c`
- Alt+`v`
- Alt+`n`
- Alt+`,`
- Alt+`.`
- Shift+Alt+`S`

They generate additional characters when combined with a second keypress:

First Key	Second Key	Result
Alt+<tt>‘</tt>	`a`	à
Alt+<tt>‘</tt>	`e`	è
Alt+<tt>‘</tt>	`i`	ì
Alt+<tt>‘</tt>	`o`	ò
Alt+<tt>‘</tt>	`u`	ù
Alt+<tt>‘</tt>	`A`	À
Alt+<tt>‘</tt>	`E`	È
Alt+<tt>‘</tt>	`I`	Ì
Alt+<tt>‘</tt>	`O`	Ò
Alt+<tt>‘</tt>	`U`	Ù
Alt+<tt>‘</tt>	`␣`	‘
Alt+6	`e`	ê
Alt+6	`u`	û
Alt+6	`i`	î
Alt+6	`o`	ô
Alt+6	`a`	â
Alt+6	`E`	Ê

Alt+6	U	Û
Alt+6	I	Î
Alt+6	O	Ô
Alt+6	A	Â
Alt+e	e	é
Alt+e	y	ý
Alt+e	u	ú
Alt+e	i	í
Alt+e	o	ó
Alt+e	a	á
Alt+e	E	É
Alt+e	Y	Ý
Alt+e	U	Ú
Alt+e	I	Í
Alt+e	O	Ó
Alt+e	A	Á
Alt+u	e	ë
Alt+u	y	ÿ
Alt+u	u	ü
Alt+u	i	ï
Alt+u	o	ö
Alt+u	a	ä
Alt+u	E	Ë
Alt+u	Y	Ÿ
Alt+u	U	Ü
Alt+u	I	Ï
Alt+u	O	Ö
Alt+u	A	Ä
Alt+p	␣	’
Alt+a	␣	¯
Alt+k	a	å
Alt+k	A	Å
Alt+x	␣	·
Alt+c	c	ç

Alt+c	C	Ç
Alt+v	s	š
Alt+v	z	ž
Alt+v	S	Š
Alt+v	Z	Ž
Alt+n	o	õ
Alt+n	a	ã
Alt+n	n	ñ
Alt+n	O	Õ
Alt+n	A	Ã
Alt+n	N	Ñ
Shift+Alt+s	␣	
Shift+Alt+;	=	×

"␣" denotes the space bar.

ROM Keyboard Layouts

The following keyboard layouts are available from ROM. You can select one directly with the BASIC `KEYMAP` command, e.g. `KEYMAP"ABC/X16"`, or via the X16 Control Panel with the BASIC `MENU` command.

Identifier	Description	Code
ABC/X16	ABC - Extended (X16)	-
EN-US/INT	United States - International	00020409
EN-GB	United Kingdom	00000809
SV-SE	Swedish	0000041D
DE-DE	German	00000407
DA-DK	Danish	00000406
IT-IT	Italian	00000410
PL-PL	Polish (Programmers)	00000415
NB-NO	Norwegian	00000414
HU-HU	Hungarian	0000040E

ES-ES	Spanish	0000040A
FI-FI	Finnish	0000040B
PT-BR	Portuguese (Brazil ABNT)	00000416
CS-CZ	Czech	00000405
JA-JP	Japanese	Custom IME (since r49)
FR-FR	French	0000040C
DE-CH	Swiss German	00000807
EN-US/DVO	United States - Dvorak	00010409
ET-EE	Estonian	00000425
FR-BE	Belgian French	0000080C
EN-CA	Canadian French	00001009
IS-IS	Icelandic	0000040F
PT-PT	Portuguese	00000816
HR-HR	Croatian	0000041A
SK-SK	Slovak	0000041B
SL-SI	Slovenian	00000424
LV-LV	Latvian	00000426
LT-LT	Lithuanian IBM	00000427

All remaining keyboards are based on the respective Windows layouts. EN-US/INT differs from EN-US only in Alt/AltGr combinations and some dead keys.

The BASIC command KEYMAP allows activating a specific keyboard layout. It can be added to the auto-boot file, e.g.:

```
10 KEYMAP"NB-NO"
SAVE"AUTOBOOT.X16
```

PETSCII code 11 ($0B) or Ctrl+K is used to toggle between the current keyboard layout and the previously loaded one. This function can be activated by pressing Ctrl+K in the BASIC editor or by sending the code to the screen with the BASIC PRINT command or the CHROUT KERNAL API call. The layout

swap function works only with built-in ROM-based layouts and will not properly swap back to a custom layout in RAM.

Loadable Keyboard Layouts

The tables for the active keyboard layout reside in banked RAM, at $A000 on bank 0:

Addresses	Description
$A000-$A07F	Table 0
$A080-$A0FF	Table 1
$A100-$A17F	Table 2
$A180-$A1FF	Table 3
$A200-$A27F	Table 4
$A280-$A07F	Table 5
$A300-$A37F	Table 6
$A380-$A3FF	Table 7
$A400-$A47F	Table 8
$A480-$A4FF	Table 9
$A500-$A57F	Table 10
$A580-$A58F	big-endian bitfield: keynum codes for which Caps m
$A590-$A66F	dead key table
$A670-$A67E	ASCIIZ identifier (e.g. "ABC/X16")

The first byte of each of the 11 tables is the table ID which contains the encoding and the combination of modifiers that this table is for.

Bit	Description
7	0: PETSCII, 1: ISO
6-3	always 0
2	Ctrl
1	Alt
0	Shift

- AltGr is represented by Ctrl+Alt.
- ID $C6 represents Alt or AltGr (ISO only)
- ID $C7 represents Shift+Alt or Shift+AltGr (ISO only)
- Empty tables have an ID of $FF.

The identifier is followed by 127 output codes for the keynum inputs 1-127.

- Dead keys (i.e. keys that don't generate anything by themselves but modify the next key) have a code of 0 and are further described in the dead key table (ISO only)
- Keys that produce nothing have an entry of 0. (They can be distinguished from dead keys as they don't have an entry in the dead key table.)

The dead key table has one section for every dead key with the following layout:

Byte	Description
0	dead key ID (PETSCII/ISO and Shift/Alt/Ctrl)
1	dead key scancode
2	full length of this table in bytes
3	first additional key ISO code
4	first effective key ISO code
5	second additional key ISO code
6	second effective key ISO code
...	...
n-1	terminator 0xFF

Custom layouts can be loaded from disk like this:

```
BLOAD"KEYMAP",8,0,$A000
```

Here is an example that activates a layout derived from "ABC/X16", with unshifted Y and Z swapped in PETSCII mode:

```
100 KEYMAP"ABC/X16"
:REM START WITH DEFAULT LAYOUT
110 BANK 0
:REM ACTIVATE RAM BANK 0
120 FORI=0TO11:B=$A000+128*I:IFPEEK(B)<>0THENNEXT :REM SEARCH
130 POKEB+$2E,ASC("Y")
:REM SET KEYNUM $2E ('Z') to 'Y'
140 POKEB+$16,ASC("Z")
:REM SET KEYNUM $16 ('Y') to 'Z'
170 REM
180 REM *** DOING THE SAME FOR SHIFTED CHARACTERS
190 REM *** IS LEFT AS AN EXERCISE TO THE READER
```

Custom BASIN PETSCII code override handler

Note: This behavior only exists in R44 or later

Some use cases of the BASIN (CHRIN) API call may benefit from being able to modify its behavior, such as intercepting or redirecting certain PETSCII codes. The Machine Language Monitor uses this mechanism to implement custom behavior for F-keys and for loading additional disassembly or memory display when scrolling the screen.

To set up a custom handler, one must configure it before each call to BASIN.

The key handler vector is in RAM bank 0 at addresses $ac03-$ac05. The first two bytes are the call address, and the next byte is the RAM or ROM bank. If your callback routine is in low ram, specifying the bank in $ac05 is not necessary.

The editor will call your callback for every keystroke received

and pass the PETSCII code in the A register with carry set. If your handler does not want to override, simply return with carry set.

If you do wish to override, return with carry clear. The editor will then unblink the cursor and call your callback a second time with carry clear for the same PETSCII code. This is your opportunity to override. Before returning, you are free to update the screen or perform other KERNAL API calls (with the exception of BASIN). At the end of your routine, set `A` to the PETSCII code you wish the editor to process. If you wish to suppress the input keystroke, set `A` to `0`.

```
ram_bank = $00
edkeyvec = $ac03
edkeybk  = $ac05

BASIN    = $ffcf
BSOUT    = $ffd2

.segment "ONCE"
.segment "STARTUP"
    jmp start
.segment "CODE"

keyhandler:
    bcs @check
    cmp #$54 ; 'T'
    bne :+
    lda #$57 ; 'W'
    rts
:   lda #$54 ; 'T'
    rts
@check:
    cmp #$54 ; 'T'
    beq @will_override
```

```
    cmp #$57 ; 'W'
    beq @will_override
    sec
    rts
@will_override:
    clc
    rts

enable_basin_callback:
    lda ram_bank
    pha
    stz ram_bank ; RAM bank 0 contains the handler vector
    php
    sei
    lda #<keyhandler
    sta edkeyvec
    lda #>keyhandler
    sta edkeyvec+1
    ; setting the bank is optional and unnecessary
    ; if the handler is in low RAM.
    ; lda #0
    ; sta edkeybk
    plp
    pla
    sta ram_bank
    rts

start:
    jsr enable_basin_callback
    ; T and W are swapped
@1: jsr BASIN
    cmp #13
    bne @1
    jsr BSOUT
    ; normal BASIN
```

```
@2:  jsr  BASIN
     cmp  #13
     bne  @2

     rts
```

Custom Keyboard Keynum Code Handler

Note: This behavior is for R43 or later, differing from previous releases.

If you need more control over the translation of keynum codes into PETSCII/ISO codes, or if you need to intercept any key down or up event, you can hook the custom scancode handler vector at $032E/$032F.

On all key down and key up events, the keyboard driver calls this vector with

- .A: keycode, where bit 7 (most-significant) is clear on key down, and set on key up.

The keynum codes are enumerated here, and their names, similar to that of PS/2 codes, are based on their function in the US layout.

The handler needs to return a key event the same way in .A

- To remove a keypress so that it is not added to the keyboard queue, return .A = 0.
- To manually add a key to the keyboard queue, use the `kbdbuf_put` KERNAL API.

You can even write a completely custom keyboard translation layer:

- Place the code at $A000-$A58F in RAM bank 0. This is safe, since the tables won't be used in this case, and

the active RAM bank will be set to 0 before entry to the handler.

- Fill the locale at $A590.
- For every keynum that should produce a PETSCII/ISO code, use `kbdbuf_put` to store it in the keyboard buffer.
- Always set .A = 0 before return from the custom handler.

```
;EXAMPLE: A custom handler that prints "A" on Alt key down

setup:
    sei
    lda #<keyhandler
    sta $032e
    lda #>keyhandler
    sta $032f
    cli
    rts

keyhandler:
    pha

    and #$ff      ;ensure A sets flags
    bmi exit      ;A & 0x80 is key up

    cmp #$3c      ;Left Alt keynum
    bne exit

    lda #'a'
    jsr $ffd2

exit:
    pla
    rts
```

Function Key Shortcuts

The following Function key macros are pre-defined for your convenience. These shortcuts only work in the screen editor. When a program is running, the F-keys generate the corresponding PETSCII character code.

Key	Function	Comment
F1	`LIST:`	Lists the current program
F2	`SAVE"@:`	Press F2 and then type a filename to save your pro
F3	`LOAD "`	Load a file directly, or cursor up over a file listing ar
F4	40/80	Toggles between 40 and 80 column screen modes
F5	`RUN:`	Run the current program.
F6	`MONITOR`	Opens the Supermon machine language monitor.
F7	`DOS"$<cr>`	Displays a directory listing.
F8	`DOS"`	Issue DOS commands.
F9	-	Not defined. Formerly cycled through keyboard lay
F10	-	Not defined
F11	-	Not defined
F12	debug	debug features in emulators

<!– For PDF formatting –>