ZEUS 64 ASSEMBLER 
Instructions

Contents

INTRODUCTION

Welcome to ZEUS, the assembler based machine code operating system for the Commodore 64.
Zeus provides the optimum method for writing 65023 machine code on your computer and yet, as you will soon discover, it is even easier to use than BASIC.
The heart of ZEUS is a HIGH SPEED multi pass assembler. It is supported by a full feature TOKENISING editor which allows quick and easy entry and editing of a source file. To complement the editor assembler there is a full feature monitor and disassembler included to help you quickly debug your machine code programs created using the editor assembler.
ZEUS provides all the features necessary to produce machine code programs quickly and efficiently.
ZEUS was written by Graham Stafford aided by (to name but a few): Neil Mottershead, Simon Brattel, David Llewellyn, Nig'm, Johnny and Rainbow Software.

The following terms are used throughout the rest of this manual and their definitions are provided for those unfamiliar with assembler terminology.
SOURCE FILE: The text, including all assembly language mnemonics, labels and comments is collectively known as the source file.
OBJECT CODE: The Object code is the machine code produced by ZEUS from the source file.
SYMBOL: A symbol is a string of alphanumeric characters used to represent a numerical value (either data or addresds).
LABEL: A label is a special type of symbol the value of which corresponds to the address of the instruction which immediately follows it.
DIRECTIVE: An assembler directive is not actually a 6502 instruction but represents an order given by the programmer to the assembler which results in the storing of values either in symbols or into memory.
For beginners to machine code and those unfamiliar with the Commodore 64 it is recommended that they purchase the "Programmers Reference Guide" which is an essential aid to serious machine code programming on the Commodore 64.

SECTION 1 - LOADING AND RUNNING ZEUS

1. Start with the Commodore 64 switched off.
2. Remove all peripherals except tape, disk and screen.
3. Switch on, check tape is fully rewound.
4. Hold down SHIFT and press RUN/STOP.

5. Press PLAY on the tape player.
6. Wait until 'FOUND ZEUS' appears.
7. Press the SPACE bar.
8. ZEUS will auto-run on loading, displaying the start up message and flashing cursor.

(Note: The supplied crack of ZEUS loads and runs as per any normal program - T.M.R)

SECTION 2 - WRITING MACHINE CODE WITH ZEUS

This section is a description of the entry of a short machine code subroutine which, along with the program itself, has been written to illustrate some of the features of ZEUS.

The program loads the byte 1 into each address (memory location) of the screen. The screen is 1000 bytes in length and starts at address 1024. The outline of the program along with the BASIC equivalent follows.

```
Screen  EQU 1024            10 Screen=1024
End     EQU Screen+1000     20 End=Screen+1000
Char    EQU 1               30 Char=1
        LDA #Screen
        LDY ^Screen
        STA $FB
        STY $FC
        LDA #End
        LDY ^End
        STA $FD
        STY $FE
Loop    LDY #0
        LDA #Char
        STA (FB),Y          40 POKE Screen,Char
        INC $FB
        BNE Over            50 Screen=Screen+1
        INC $FC
Over    SEC
        LDA $FD
        SBC $FB
        LDA $FE
        SBC $FC
        BCS Loop            60 IF Screen-End <=0 THEN 50
        RTS                 70 RETURN
```

After start up you are put into the machine code monitor, to enter the assembler type Z followed by 'RETURN'. The following message should appear:

ZEUS
New or Old?

Firstly, type in 'N', this is similar to the BASIC 'NEW' command, except the assembler MUST be told whether you want new or old text, type 'O' if a source file is already present.

Entering lines of text is very similar to writing a BASIC program. Each line must be prefixed by a number which may take any value from 0 to 65534 inclusive. It is important to remember that, providing the line the cursor is on contains a valid line number, pressing RETURN will insert that line at the correct position in the source file, so replacing any existing line which had the same number.

The command to tell ZEUS to provide line numbers automatically is of the form:

```
I x y
```

where

x = line number from which to start
y = increment between successive line numbers

x and y are called 'parameters' because they specify the particular values which the command will use when it executes. For example, in a long BASIC program you may use:

LIST 1000

Here, the parameter 1000 causes the command to list line 1000.
To illustrate the use of parameters type:

I 100 100

and press RETURN several times. To exit from auto-line number mode simply clear the current line number using CLEARLINE : SHIFT RETURN.
Clear the screen, type 'I' and press RETURN several times. As you have not specified particular vaules for the parameters x and y ZEUS has used the 'default' values which are x=10 and y=10. To take an example from BASIC, the default value of the LIST command is the first line number.

To enter the first few lines of text:

1. Type CLEARLINE (SHIFT RETURN)
2. Type 'I' followed by RETURN (once only)
3. Type the following lines of text, pressing RETURN after each to insert the line in the source file and proceed to the next line number:

```
Screen  EQU 1024
End     EQU 1024+1000
Char    EQU 1
        LDA #Screen
        LDY ^Screen
        .
        .
        .
        BCS Loop
        RTS
```

4. Exit from auto-line number mode (SHIFT RETURN).

You may now examine the source file you have created. Type 'V' (short for View). The View command takes up to three parameters of the form:

V x y z

where

x = line number to list from
y = line number to list up to
z = number of lines to be listed before pausing

This command is exactly the same as the BASIC LIST command except for the third parameter. Every z lines the listing will stop. If RUN STOP is pressed the listing will be aborted, any other key will continue the listing.
Now press RETURN, the source file will list from the beginning in groups of 10 lines. You will have noticed that the source file has been indented to make labels more visible, the number of spaces by which the source is indented is set up using the 'Z' command.
Abort the listing by pressing RUN STOP, now press 'Z 16' followed by RETURN and relist the source file (using the 'V' command). You will have noticed that the source file has been indented by 16 spaces this time.
You may edit the source file by using the screen editor, just like BASIC, simply move the cursor and perform the change.
One additional line is required before you can assemble and test the routine. You must set an entry point, ie. the point from which the execute command ('X') will run the code.
Enter:

5 ENT

ZEUS also features a renumber command in the form:

R x y z

See ZEUS assembler commands (Section 4) for definition of the parameters.

To renumber the source file, enter 'R'. Now list the first 5 lines by typing:

```
V ,,5
```

and pressing RETURN once only. Note that the use of a comma to replace a parameter causes ZEUS to use the default value of that parameter. You can now see that the source has been renumbered in accordance with the default values of the 'R' command.
Check carefully that the source file is identical to the listing given earlier.
Now you are ready to assemble the source file to produce the object code, commonly called the machine code. Type 'A' (short for assemble) and press RETURN.
If all is well the following message will be displayed:
Assembly complete, no errors.
If you have made a mistake, an error report will have been issued followed by the offending line. If the error is not apparent, consult the appendix on error report codes.
You may find it worthwhile to deliberately introduce an error by, for example, removing the space between 'A' and '#' in line 50. Simply use the screen editor to effect this change.
Assemble (using 'A').
You should see:
Syntax error
50 LDA#Screen

To correct the error use the screen editor again.
It is advisable to save the source file on tape or disk prior to execution. Loading and Saving is achieved using the 'L' and 'S' commands respectively.
To save on tape type:

```
S "SOURCE" 1
```

To save on disk type:

```
S "SOURCE" 8
```

You are now ready to test the routine. Enter 'X' (short for EXECUTE). If all is well the screen will fill with the character 'A'. If this does not happen, list the source and check it carefully against the listing above. If the computer 'crashes' simply press RESTORE, this will place you in the monitor, type 'Z' followed by RETURN, when in ZEUS type 'O' to recover your source file. If this does not recover from the crash then reload ZEUS, enter the assembler (using 'Z'), tell the assembler you want new text (using 'N') and reload the source using:

```
L "SOURCE" 1 or
L "SOURCE" 8
```

In this section you will have gained an appreciation of the way ZEUS can be used to produce a simple subroutine. Yet there are many additional features at your disposal to aid the programming of more complex routines and programs. The next section provides a comprehensive description of these and also consolidates the features already mentioned.

SECTION 3 - ENTERING AND EDITING TEXT

3.1 General Format

ZEUS uses the ASCII character set. To enter assembly language instructions you must first type in a line number and follow it with the required line of text. The line number must lie in the range 0-65534 inclusive.
The text must consist of one or more statements separated by colons. Each statement comprises:

a) An optional label
b) An instruction
c) An optional comment

a) Optional label
The following rules govern the use of labels:
i) A label may contain upper and lower case letters and digits.
ii) A label must start with a letter.
iii) A label can be up to 31 characters in length.

iv) A label must not be identical to a reserved word (eg using ‘LDA’ as a label is not permitted). However, a label may contain reserved words (eg ‘ALDA’ would be valid).
v) A label must not contain operators.
vi) A label must be separated from an instruction by at least one space.
Note that every reference to a particular label must be identical, character for character, to that label.

b) Instruction
An instruction may be any of the standard 6502 instructions or it may be an assembler directive (see below). Two extra facilities are provided. The first provided by ZEUS is for referring to Accumulator instructions. The ‘A’ character after the instruction is optional. Thus:

ASL A	may be written as	ASL
ROL A	may be written as	ROL

and so on.
The second is for immediate class instructions, if the ‘#’ is replaced by a ‘^’ the high byte of the expression is loaded instead of the low byte. Thus:

LDA #Label is the same as LDA #Label MOD 256
LDA ^Label is the same as LDA #Label DIV 256

c) Optional comment
A comment may be appended to thre end of any instruction. It must be separated from the instruction by a semi-colon.

3.2 Constants
Constants may be expressed in decimal, hexadecimal, character or binary.

Decimal:	65	1	99	4096
Hexadecimal	$41	$1	$63	$1000
Binary:	%01000001	%1	%01100011	%1000000000000
Character (CBM):	'a	'G	')	'£
Character (ASCII)	&A	&g	&)	&£

For example, to load the CBM literal 'C' into the accumulator, use:

LDA #'C

Finally, there is also a system constant '.' which is set to the current assembly position.

eg. BNE . is equivalent to Loop BNE Loop

3.3 Operators
In order to further facilitate the writing of symbolic programs ZEUS allows the use of the following operators:

+	addition
-	subtraction
*	multiplication
DIV	integer divide
MOD	remainder from integer divide
AND	logical AND
OR	logical OR
XOR	logical Exclusive OR

3.4 Expressions
Wherever a constant is required in an instruction an expression may be used in its place. Expressions are built from labels and/or constants separated by operators.
eg. LDA Address+Offset
LDA #Label AND $A

Note that the expressions will be corrupted by the assembler to determine the actual value which will be inserted into the object code. Expressions are computed at assembly time, not at program execution time.

3.5 Assembler Directives

The following 'pseudo-operations' either provide parameters for the operation of ZEUS or instruct the assembler to store values either as symbols or directly into memory:

ORG nnnn,mmmm
Short for ORIGIN. This directive instructs ZEUS to assemble the block of machine code (as translated from the source listing after the ORG statement) from address nnnn and place the object code at address mmmm, if the second parameter is omitted the code is placed at address nnnn.

Multiple ORG's within the same source file are allowed. Each ORG statement will redirect the address from which subsequent code is assembled.

DSP nnnn
Short for DISPLACEMENT. A DSP instruction alters the place from which subsequent code is placed, even though the code so produced is assembled to run at the address specified by the current ORG parameter.

It is sometimes convenient to be able to generate code at a different location from the one at which it is ultimately intended to run. The monitor may be used to relocate the code to the ORG address.

For example, given the following two statements at the start of a source file:

```
ORG 30000
DSP 10000
```

the source following would be assembled from 40000 but would normally run at 30000.

ENT
Set an entry point. The 'X' command executes the assembled code from the last ENT directive in the source file.

EQU
Short for EQUATES or EQUALS. A label may have a value assigned to it using a statement in the form:

```
label EQU value
```

DFB nn,nn...
Inserts bytes nn at the current assembly address, if a 16 bit value is given an error is flagged.

DFL nnnn,...
Inserts the low byte of the word at the current assembly address.

DFH nnnn,...
Inserts the high byte of the word at the current assembly address.

DFW nnnn,...
Inserts words (addresses) nnnn at the current assembly position.

DFB /string/
The ASCII character codes for the text enclosed in the delimiters will be inserted at the current assembly address.

DFC /string/
The CBM character codes for the text enclosed in the delimiters will be inserted at the current assembly address.

DFS nnnn,mm
Inserts nn byte mm's at the current assembly position. If the second parameter is omitted a random byte is inserted.

As with all assembly language instructions, directives may be prefixed by a label.

The above addresses nnnn etc may be replaced by any expression.

3.6 Screen Editor
ZEUS provides all the usual features of the Commodore 64 Screen Editor but also includes the two following extra commands:

CLEARLINE
The current line may be cleared and the cursor placed at the start of the line using SHIFT and RETURN together. NB This command does not remove the current line from the text but simply from the screen/

TAB
The left arrow key moves the cursor to the next TAB position on the current line. The TAB positions may be set up form within the monitor.

SECTION 4 - ZEUS ASSEMBLER COMMANDS

A command consists of a command letter followed by any number of numerical or string parameters.

Numerical parameters consist of either decimal or hexadecimal constants. Entry of numerical parameters overwrites the default parameters for that command.

If it is desired to alter a later parameter without disturbing earlier ones, this can be achieved by entering a comma for every parameter to be skipped.

String parameters are represented by enclosing the string within delimiters. Delimiters can be any character. Hereafter, backslash / is used as the delimiter.

Commands

A x
Assemble the source. The x parameter defines the assembly option, bit 0 of the assembly option defines whether the object code should be dumped or not, 1 = no dump. Bit 1 of the assembly option defines whether the symbol table is to be preserved or not, this allows source files to be linked together, 1 = preserve the symbol table.
DEFAULT: x = 0

EG
A0 Clear symbol table. Dump code.
A1 Clear symbol table. Don't dump code.
A2 Preserve symbol table. Dump code.
A3 Preserve symbol table. Don't dump code.

C /string1//string2/ x y
This command searches the source file from line number x to line number y for occurrences of string1, the line containing the string is then printed, pressing 'Y' will replace string1 with string2 and then continue searching, pressing 'Q' will abort the command, any other key will continue searching the source file for occurrences of string1 leaving the file unchanged.
DEFAULT: string1, string2 no default x = start of source y = end of source

D x y
Delete all lines between and including x and y. If no parameters are given an error is flagged.

F /string/ x y z
This command searches the source file for all occurrences of the given string. The search commences at line x and finishes at line y. Any lines containing the string are displayed and a pause initiated after z lines have been displayed.
DEFAULT: x=start of source y=end of source z=10

I x y
After entry of this command ZEUS will automatically generate line numbers, starting with line number x and printed in increments of y. Pressing CLEARLINE will halt auto line numbering.
DEFAULT: x=10 y=10

L /string/ x
Load source from device x into memory at the current start of source point.
DEFAULT string="" x=1
L+ /string/ x
Load source file from device x into memory at the current end of source point, then renumber. This command allows source files to be appended.

N x
Create a new, empty source file at address x. NB care must be taken when using ths command as placing a source file at certain locations may cause a crash.
DEFAULT: x=$0801

O x
This causes the 'old' source file at address x to become the current source; it is usually used to retrieve a source file when reentering the assembler.
DEFAULT: x=$0801

The two above commands allow the use of multiple sources, but must be used with extreme caution.

Q
QUIT back to the monitor. To return to ZEUS the Z command must be used in the monitor.

R x y z
Renumber the source file. The parameters are as follows:
x: the first new line number
y: the spacing between successive line numbers
z: the line number to start renumbering from.
DEFAULT: x=10 y=10 z=start of source

S /string/ x
Save the current source file to device x with the filename string.
DEFAULT x=1

T x
Print the symbol table. x gives the number of symbols/labels and their associated hexadecimal values listed before pausing.
DEFAULT: x=10

V x y z
View the source from line x to line y. Parameter z gives the number of lines to be listed before pausing.
DEFAULT: x=start of source y=end of source z=10

V Label x
View the source file from the line that starts with Label and pause after x lines.
DEFAULT: x=10

W string
Will send string to the command channel of the CBM disk drive; this allows manipulation of the disk directly from the assembler.

X
Execute the machine code produced by the last assembly. Execution begins at the ENT statement nearest the end of the source file. If no ENT statement existed, the DSP option was used or no assembly occurred then an Error is flagged.

Y
Print out the disk directory.

Z x
List option. The parameter x defines how many spaces the listing will be indented by. If x>0 then all unnecessary spaces are rejected on input. This facility allows source files to be compact yet still readable.

SECTION 5 - THE MONITOR

The co-resident Monitor allows you to directly inspect and manipulate the CPU registers and memory, however caution should be exercised when using the monitor as indiscriminate use of the commands can have unpredictable results.

MONITOR COMMANDS

All commands in the monitor accept hexadecimal unless otherwise stated.

A
Set TAB stops. Any non space character on the same line as the 'A' will define a TAB stop (provided the line is entered).

C x y z
Non intelligent copy. Copies the block of memory between x and y to memory starting at z. This command is unsuitable for moving code up in memory, however it can be used to repeat a set of bytes in a block of memory.

Eg: Press RESTORE then type
C 400 7E2 405 Press RETURN this has repeated the first five bytes of the screen over and over to fill the block of screen memory.

D x
Disassemble from address x for 20 lines, press RETURN when the screen is complete to continue disassembly. The cursor may be moved up to change the hex listed and when RETURN is pressed the screen is redisassembled.

E x
This command displays both the hex and decimal representation of the value x which may be of either form, hex values must be preceded with a '$' character.

F x y z
Fill from memory address x to memory address y with byte z.

G x
Load the 6510 registers with the values given by the R command and start execution at address x, if no address is given execution starts at the address given by the R instruction.

H x y z t …
Search from memory address x to y searching for all occurrences of the bytes z t … if found the address and adjacent bytes are printed out.

H x y 'string
As above but searches for the occurrence of string in CBM format.

H x y &string
As above but searches for the occurrence of string in ASCII format.

I x y z
Intelligent copy. Copies block address x to y to block starting at address z.

K x y z
Change colours, x = foreground y = background z = border

L /string/ x y
Load file string from storage device x and place at address y, if no parameter is specified the fill will load into the same block of memory from which it was saved.

M x
Enter memory modify mode starting at address x. This command will print address x followed by its contents, either the address or the contents may be changed by simply altering the displayed value. Once entered the next address and its contents will be output ready for alteration. To exit modify CLEARLINE must be typed.

O x y
Calculates and prints the offset value from address x to address y, where x is the address of the branch instruction and y is the address to which the branch will occur. An error is flagged if the branch is out of range.

P x y
If x is a non zero value the printer specified by the y parameter will be turned on, a zero value will turn it off, if on all output is echoed to the printer selected. If y is omitted or is 0 the CBM printer will be selected, if y is 1 a Centronics parallel printer must be connected to the user port via a standard cable.

Q
Quick trace. This command starts tracing from the address given by the R command. As each instruction is executed the current position of the program counter is displayed. The user has to press the space bar between each instruction. If 'W' is pressed walk mode is entered (see 'W'). If RUN STOP is pressed control is passed back to the monitor.

R
Display registers. New values can be entered by simply changing the value printed.

S /string/ x y z
Save memory from address y to z to device x with the filename string.

T x y z
Tabulate memory from address x to address y pausing after z groups of 8 bytes have been displayed. Memory contents may be altered by changing the values as described in the 'M' command.

W
Walk through a program. Trace as for the Q command except the register contents are printed together with the disassembly of the next instruction to be executed. Press space to continue, 'Q' to start a quick trace and RUN STOP to pass control back to the monitor.

X
Exit to BASIC, to reenter the monitor type 'SYS49152'. Note the assembler cannot be accessed directly from BASIC, the monitor MUST be entered first.

Z
Jump to the assembler.

APPENDIX 1 - COMMAND LIST

i) Assembler

A x	Assemble with option x
C /string1//string2/ x y	Change occurrences of string1 to string2 from x to y
D x y	Delete lines x to y inclusive.
F /string/ x y z	Find a string between line x and y: print z occurrences at a time.
I x y	Insert starting at line x in steps of y.
L /string/ x	Load source file string from device x.
L+ /string/ x	Append source file string from device x.
N x	Create a new source file at address x.
O x	Recover an old source file at address x.
Q	QUIT. Return to monitor.
R x y z	Renumber the source file starting from line z to commence at line x in steps of y.
S /string/ x	Save source file with name string to device x.
T x	Print out symbol table x lines at a time.
V x y z	View from line x to line y in steps of z lines.
W string	Send string to CBM disk.
X	Execute object code at last ENT statement.
Y	Put out disk directory.
Z x	Set list to indent listing by x spaces.

ii) Monitor

A	Set up tab stops.
C x y z	Non intelligent copy, copy from address x to y and place at address z.
D x	Disassemble from address x.
E x	Print x in decimal and hex.
F x y z	Fill from x to y with byte z.
G x	GO from address x.
H x y z t ...	Hunt from x to y for bytes z, t...
H x y 'string	Hunt from x to y for string in CBM format.
H x y &string	Hunt from x to y for string in ASCII format.
I x y z	Intelligent copy block x to y to address z.
K x y z	Set up foreground, background, border colours.
L /string/ x y	Load file string from device x and place at y.
M x	Modify Memory
O x y	Calculate offset branch.
P x y	Control printer output.
Q	Quick trace.
R	Display and modify CPU registers.
S /string/ x y z	Save y to z to device x with file name string.

T x y z	Tabulate memory from x to y in steps of z lines.
W	Walk trace.
X	Exit to BASIC.
Z	Enter Assembler.

APPENDIX 2 - RESERVED WORDS

ORA	AND	EOR	ADC	STA	LDA	CMP	SBC
ASL	ROL	LSR	ROR	STX	LDX	DEC	INC
BIT	STY	LDY	CPY	CPX	JMP	JSR	BRK
PHP	CLC	PLP	SEC	RTI	PHA	CLI	RTS
PLA	SEI	DEY	TXA	TYA	TXS	TAY	TAX
CLV	TSX	INY	DEX	CLD	INX	NOP	SED
BPL	BMI	BVC	BVS	BCC	BCS	BNE	BEQ
ORG	DFS	DFB	DFW	DFM	DFH	DFL	DFC
DSP	ENT	EQU					

APPENDIX 3 - OPERATORS

,X)	),Y	,X	,Y	MOD	OR	XOR	DIV
AND							

APPENDIX 4 - ERRORS

Undefined Label
Syntax Error in Expression
Expression too complicated
Division by Zero
Syntax Error
Zero Page location expected
Illegal indirect jump
Branch out of range
, Expected
Redefined label
Out of memory
Byte expected
Bad Program

APPENDIX 5 - ZEUS MEMORY MAP

A) Editor Assembler	$A000-$BFFF (The Editor Assembler is paged in by the Z command in the monitor)
B) Monitor	$C000-$CFFF
C) Symbol Table	Starts at $FFF0 and works down in memory, the symbol table is paged in and out by the Editor Assembler automatically.