# A65 Assembler ; Copyright (C) 1984, 1986, 1989 by Charles W. Marslett#



A65 (A very simple 6502 assembler)

The A65 assembler is modeled after the Atari MACRO Assembler (AMAC) that was distributed by APX along with the MEDIT editor and the DDT debugger. The most significant thing about it is that the source files cannot contain line numbers (so editors that use line numbers must be able to write an unnumbered output file). A second characteristic is that it is a disk-to- disk assembler (the source must be in a disk (or cassette) file and the object cannot be inserted directly into memory.

The assembler is run from the DOS menu using the 'L' command, and prompts you for the source file name, the object file name and the listing file name: a RETURN in response to the object and listing prompts will cause the assembler to generate files on the same drive with extensions '.OBJ' and '.LST' using the same file name as the source file. If no '.' appears in the source file name, '.ASM' is its assumed extension. A listing can be printed directly by specifying 'P:' as the list file or dumped to the screen by specifying 'E:'. A '-' disables the listing if you do not want one.

A start at implementing support for the 65C02 opcodes is in the code, but what is there has not been tested very thouroughly and most of the opcodes and addressing modes are not in place yet. Otherwise, the machine instructions are as any other standard 6502 assembler would expect. Expressions can include +, -, * and / operators as well as HIGH[] and LOW[] functions to extract the high and low bytes of 16-bit numbers. Brackets, [ and ], may be used to group terms in an expression if necessary. Symbols may have up to 11 characters, including upper and lower case letters, underscores and numbers. Symbols must start with an underscore or letter, and case is significant except for the predefined assembler opcodes and the register names A, X and Y.

The machine opcodes recognized are:

ADC, AND, ASL, BCC, BCS, BEQ, BIT, BMI, BNE, BPL, BRA (65C02 only), BRK, BVC, BVS, CLC, CLD, CLI, CLV, CMP, CPX, CPY, DEC, DEX, DEY, EOR, INC, INX, INY, JMP, JSR, LDA, LDX, LDY, LSR, NOP, ORA, PHA, PHP, PLA, PLP, ROL, ROR, RTI, RTS, SBC, SEC, SED, SEI, STA, STX, STY, TAX, TAY, TSX, TXA, TXS, TYA

The assembler directives are:

DB defines a byte, in decimal (12), hex ($0C), octal (@14) or binary (%00001100) and may also be a character string (enclosed in quotes) DC defines a byte, as above, but with the high bit set (if defining a string, only on the last byte of the string) DW defines a word, low byte first, as if an indirect pointer DS allocates a number of bytes without storing any data in them

ORG sets the location counter to a fixed address

- = same as ORG

EQU defines a symbol to have a specific value = same as EQU

INCLUDE causes the file specified to be inserted at this point in the assembly (can be nested if the DOS supports sufficient numbers of files open at once -- the object, list and source files are kept open at all times, and each level of nested includes requires one more open file. MYDOS and Atari DOS 2 allow 3 normally, which does not allow a listing and include files. If set to 5, the assembler could generate a listing and handle 1 level of nested includes) TITLE specifies the first title line

SUBTTL allows entering a second title line PAGE causes the assembler to go to the top of th next listing page

END terminates the program and specifies INIT and RUN addresses

LIST a stub for future expansion MACRO another stub (does nothing, not even generate an error) MEND another stub

Octal numbers, EQU, *=, and INIT and RUN addresses may have bugs in them -- good luck. Most of the rest has been debugged reasonably well.

The END statement can have the following forms:

END no RUN or INIT vectors generated at all END ADDR ADDR is the RUN entry point END ADDR, ADDR is the INIT entry point (no RUN vector) END INIT,RUN both vectors specified

(This is what I call minimum documentation, I will add to it as questions are asked)

Charles Marslett 8/21/85

Release 1.1 Changes since the first release:

The assembler now behaves more reasonably when a forward reference is encountered (it used to just generate junque quietly!) -- forward references are forced to 16-bit values if possible, otherwise A65 assumes you know what you are doing, and assumes it will be an 8-bit value when it is defined.

The assembler allows upper and lower case symbols, and it is case sensitive. It still ignores case, however, when handling the directives and machine opcodes as well as the register names ("A", "X" and "Y"). Underscores are also allow in symbols and are treated as letters (makes C people happy!). Symbols may also be up to 11 characters long (rather than 8 as before).

The initial load address is the HIMEM value from DOS (the lowest possible load address) rather than a random value left over in memory.

Errors are marked in the listing file (as before) but they are also echoed to the screen, and the total number of errors (in HEX) are reported at the end of the assembly. If any errors occur, the automatic return to the DOS prompt is delayed until the [RETURN] key is pressed to give you an opportunity to note any (unexpected) errors reported.

Finally, some errors that were previously accepted (and generated bad code) are now caught and flagged as errors). These included multiply defined symbols and some invalid addressing modes.

Charles Marslett 6/16/89

Release 1.2 Changes since the second release:

Source code lines can be delimited by an ATASCII EOL character ($9B) or a Unix style EOL (a single $0A byte) or a CP/M (or MSDOS) EOL (either $0D followed by $0A or $0A followed by $0D). ASM5.ASM was changed.

Charles Marslett 7/4/89

## ATR-Images#

- A65_with_DOS_2.5.atr ; A65 Assembler Version 1.2 with source code and DOS 2.5 ; Thank you so much Mr. Marslett for giving us the source code. :-)))

## ZIP-Archive#

- A65_12.ZIP ; ZIP-file of A65 with source code, runtime and manual ; mirror from the original source Wordmark

#