# Apple II, IIe and IIc: Mini-Assembler

This article last reviewed: 21 September 1984

This note covers use of the Apple II, IIe and IIc mini-assembler only, not the II+. This is not a course in assembly language programming. For a reference on programming the 6502 microprocessor, refer to the Synertek Programming manual or any of the tutorials available. This note assumes the user has a working knowledge of 6502 programming and mnemonics.

The Apple II mini-assembler is a programming aid aimed at reducing the amount of time required to convert a handwritten program to object code. The mini-assembler is basically a look-up table for opcodes. With it, you can type mnemonics with their absolute addresses, and the assembler will convert it to the correct object code and store it in memory.

Typing "F666G" will puts you in mini-assembler mode. While in this mode, any line typed in will be interpreted as an assembly language instruction, assembled, and stored in binary form unless the first character on the command line is a "$".

If the first character of a command line is a "$", the remainder of the line will be interpreted as a normal monitor command, executed, and control returned to the mini-assembler. To get out of the mini-assembler, press RESET.

If the first character on the line is blank, the assembled instruction will be stored starting at the address immediately following the previously assembled instruction. If the first character is not a blank nor a "$", the line is assumed to contain an assembly language instruction preceded by the instruction address (a hex number followed by a ":"). In either case, the instruction will be retyped over the line just entered in dis-assembler format to provide a visual check of what has been assembled.

The counter that keeps track of where the next instruction will be stored is the pseudo PC (Program Counter) and it can be changed by many monitor commands (eg. 'L', 'T', . . .). Therefore, it is advisable to use the explicit instruction address mode after every monitor command and, of course, when the mini-assembler is first entered.

Errors (unrecognized mnemonic, illegal format, etc.) are signalled by a "beep" and a caret ("^") will be printed beneath the last character read from the input line by the mini-assembler.

The mnemonics and formats accepted by the mini-assembler are the same as those listed by the 6502 Programmers Manual, with the following exceptions and differences:

1. All imbedded blanks are ignored, except inside addresses.

2. All addresses entered are assumed to be in hex (rather than decimal or symbolic). A preceding "$" (indicating hex rather than decimal or symbolic) is therefore optional, except that it should not precede the instruction address).

3. Instructions that operate on the accumulator have a blank operand field instead of "A".

4. When entering a branch instruction, the argument of the branch mnemonic should be the address of the target of the branch. If the destination address is not known at the time the instruction is entered, simply enter an address that is in the neighborhood, and later re-enter the branch instruction with the correct target address. NOTE: If a branch target is specified that is out of range, the mini-assembler will flag the address as being in error.

5. The operand field of an instruction can only be followed by a comment field, which starts with a semicolon (";"). Obviously, the mini-assembler ignores the field and in fact will type over it when the line is typed over in disassembler format.

6. Any page zero references will generate page zero instruction formats if such a mode exists. There is no way to force a page zero address to be two bytes, even if the address has leading zeroes.

In general, to specify an addressing type, simply enter it as it would be listed in the disassembly. For information on the disassembler, see page 49 of the Apple II Reference Manual.
<None>

Keywords: <None>

==========

This information is from the Apple Technical Information Library.

19960215 11:05:19.00