a reference manual for

M A C / 6 5

a Macro Assembler and Editor program for
use with 6502-based computers built by
Atari, Incorporated

Rev 1.0

PREFACE

MAC/65 is a logical upgrade from the OSS product EASMD (Edit/ASseMble/Debug) which was itself an outgrowth of the Atari Assembler/Editor cartridge. Users of either of these latter two products will find that MAC/65 has a very familiar "feel". Those who have never experienced previous OSS products in this line should nevertheless find MAC/65 to be an easy-to-use, powerful, and adaptable programming environment. While speed was not necessarily the primary goal in the production of this product, we nevertheless feel that the user will be hard pressed to find a faster assembler system in any home computer market. MAC/65 is an excellent match for the size and features of the machines it is intended for.

MAC/65 was conceived by and completely executed by Stephen D. Lawrow. The current version of MAC/65 is only the latest in a series of increasingly more complex and faster assemblers written by Mr. Lawrow following the lead and style of EASMD. As a measure of our confidence in this assembler, it is entrusted with assembling itself, probably a more difficult task than that to which most users will put it.

TRADEMARKS

The following trademarked names are used in various places within this manual, and credit is hereby given:

DOS XL, BASIC XL, MAC/65, and C/65 are trademarks of Optimized Systems Software, Inc.

Atari, Atari 400, Atari 800, Atari Home Computers, and Atari 850 Interface Module are trademarks of Atari, Inc., Sunnyvale, CA.

TABLE OF CONTENTS

INTRODUCTION

This manual assumes the user is familiar with assembly language. It is not intended to teach assembly language. This manual is a reference for commands, statements, functions, and syntax conventions of MAC65 It is also assumed that the user is familiar with the screen editor of the Atari computer. Consult Atari's Reference Manuals if you are not familiar with the screen editor.

If you need a tutorial level manual, we would recommend that you ask your local dealer or bookstore for suggestions.

Although we are hesitant to suggest ANY of the books currently available (because they do not address Atari Computers properly), two books that have worked well for many of our customers are "Machine Language for Beginners" by Richard Mansfield from COMPUTE! books and "Programming the 6502" by Rodney Zaks.

This manual is divided into two major sections. The first two chapters cover the Editor commands and syntax, source line entry, and executing source program assembly. The next three chapters then cover instruction format, assembler directives, functions and expressions, Macros, and conditional assembly.

Note that DDT--the Dunion Debugging Tool--is described in a separate manual section, which follows this MAC/65 manual.

MAC65 is a fast and powerful machine language development tool. Programs larger than memory can be assembled. MAC65 also contains directives specifically designed for screen format development. With MAC65's line entry syntax feature, less time is spent re-assembling programs due to assembly syntax errors, allowing more time for actual program development.

START UP

Simply turn off the power to your computer and insert your MAC/65 cartridge (in the left cartridge slot if using an Atari 800 Computer).

If you are using a disk drive, insert an appropriate DOS boot disk (e.g., DOS XL or Atari DOS) into drive 1 and be sure the drive's power is on.

Turn on your computer. If you have a drive with a proper diskette inserted, DOS will boot. Depending upon the version and kind of DOS you have, you may find that you need to give a command to DOS in order to enter the MAC/65 cartridge. If so, enter the command.

You should be presented with MAC/65's name and copyright lines and an "EDIT" prompt. If not consult your hardware and/or DOS manuals and try again.

You are now ready to begin using MAC/65.

WARM START

The user can exit to DOS XL by entering the MAC/65 command DOS (followed by [RETURN], of course). To return to MAC/65, the user can use the DOS XL command CAR [RETURN] (or menu command 'T').

Unless you have used certain extrinsic commands, DOS XL will return to MAC/65 via a "warm start" (i.e., without clearing out any source lines in memory). Consult your DOS XL manual for details.

Generally, when using Atari DOS, MAC/65 works much like any other cartridge. The MAC/65 "DOS" command will exit to Atari DOS, and the Atari DOS "B" command will return to MAC/65. If you use a MEM.SAV file, your MAC/65 program should stay intact. See your Atari DOS manual for details.

SYNTAX

The following conventions are used in the syntax descriptions in this manual:

1. Capital letters designate commands, instructions, functions, etc., which must be entered exactly as shown (e.g., ENTER, .INCLUDE, .NOT). (But see NOTE below.)

2. Lower case letters specify items which may be used. The various types are as follows:

lno	- Line number between 0-65535, inclusive.
hxnum	- A hex number. It can be address or data. Hex numbers are treated as unsigned integers.
dcnum	- A positive number. Decimal numbers are rounded to the nearest two byte unsigned integer; 3.5 is rounded to 4 and 100.1 to 100.
exp	- An assembler expression.
string	- A string of ASCII characters enclosed by double quotes (eg. "THIS IS A STRING").
strvar	- A string representation. Can be a string, as above, or a string variable within a Macro call (eg. &$1).
filespec OR file	- A string of ASCII characters that refers to a particular device. See device reference manual for more specific explanation.

3. Items in square brackets denote an optional part of syntax (eg. [,lno]). When an optional item is followed by (...) the item(s) may be repeated as many times as needed.
   Example: .WORD exp [,exp ...]

4. Items in parentheses indicate that any one of the items may be used , eg. (,0) (,A).

NOTE: MAC65 in EDIT mode is NOT case sensitive. Inverse video characters are uninverted. Lower case letters are converted to upper case. EXCEPTIONS: characters between double quotes, following a single quote, or in the comment field of a MAC65 source line will remain unchanged.Text entered in TEXT mode, though, will not be changed.

---this page intentionally left blank--

CHAPTER 1: THE EDITOR

The Editor allows the user to enter and edit MAC/65 source code or ordinary ASCII text files.

To the Editor, there is a real distinction between the two types of files; so much so that there are actually two modes accessible to the user, EDIT mode and TEXTMODE. However, for either mode, source code/text must begin with a line number between 0 and 65535 inclusive, followed by one space.

```
Examples: 10 LABEL LDA #$32
          3020 This is valid in TEXT MODE
```

The first example would be valid in either EDIT or TEXTMODE, while the second example would only be valid in TEXTMODE.

The user chooses which mode he/she wishes to use for editing by selecting NEW (which chooses the MAC/65 EDIT mode) or TEXT (which allows general text entry). There is more discussion of the impact of these two modes below; but, first, there are several points in common to the two modes.

1.1 GENERAL EDITOR USAGE

The source file is manipulated by Editor commands. Since the Editor recognizes a command by the absence of a line number, a line beginning with a line number is assumed to be a valid source/text line. As such, it is merged with, added to, or inserted into the source/text lines already in memory in accordance with its line number. An entered line which has the same line number as one already in memory will replace the line in memory.

Also, as a special case of the above, a source line can be deleted from memory by entering its line number only. (And also see DEL command for deleting a group of lines.)

Any line that does not start with a line number is assumed to be command line. The Editor will examine the line to determine what function is to be performed. If the line is a valid command, the Editor will execute the command. The Editor will prompt the user each time a command has been executed or terminated by printing:

```
EDIT    for syntax (MAC/65 source) mode
TEXTMODE for text mode
```

The cursor will appear on the following line. Since some commands may take a while to execute, the prompt signals the user that more input is allowed. The user can terminate a command before completion by hitting the break key (escape key on Apple II).

And one last point: If the line is neither a source line or a valid command. The Editor will print:

```
WHAT?
```

1.2 TEXT MODE

The Editor supports a text mode. The text mode is entered with the command TEXT. This mode will NOT syntax check lines entered, allowing the user to enter and edit non-assembly language files. All Editor commands funtion in text mode.

Remember, though, that all text lines must begin with a line number; and, even in TEXTMODE, the space following the line number is necessary.

1.3 EDIT MODE

MAC/65 is nearly unique among assembler/editor systems in that it allows the assembly language user to enter source code and have it IMMEDIATELY checked for syntax validity. Of course, since assembly language syntax is fairly flexible (especially when macros are allowable, as they are with MAC/65), syntax checking will by no means catch all errors in user source code. For example, the existence of and validity of labels and/or zero page locations is not and can not be checked until assembly time. However, we still feel that this syntax checking will be a boon to the beginner and experienced programmer alike.

Again, remember that source lines must begin with a line number which must, in turn, be followed by one space. Then, the second space after the line number is the label column. The label must start in this column. The third space after the line number is the instruction column. Instructions may either start in at least the third column after the line number or at least one space after the label. The operand may begin anywhere after the instruction, and comments may begin anywhere after the operand or instruction. Refer to Assembler Section for specific instruction syntax.

As noted, the Editor syntax checks each source line at entry. If the syntax of a line is in error, the Editor will list the line with a cursor turned on (i.e., by using an inverse or blinking character) at the point of error.

The source lines are tokenized and stored in memory, starting at an address in low memory and building towards high memory. The resultant tokenized file is 60% to 80% smaller than its ASCII counterpart, thus allowing larger programs to be entered and edited in memory.

SPECIAL NOTE: If, upon entry, a source line contains a syntax error and is so flagged by the Editor, the line is entered into Editor memory anyway. This feature allows raw ASCII text files (possibly from other assemblers and possibly containing one or several syntax errors as far as MAC/65 is concerned) to be ENTERed into the Editor without losing any lines. The user can note the lines with errors and then edit them later.

---this page intentionally left blank---

CHAPTER 2: EDITOR COMMANDS

This chapter lists all the valid Editor-level commands, in alphabetical order, along with a short description of the purpose and function of each.

Again, remember that when the "TEXTMODE" or "EDIT" prompt is present any input line not preceded by a line number is presumed to be an Editor command.

If in the process of executing a command any error is encountered, the Editor will abort execution and return to the user, displaying the error number and descriptive message of the error before re-prompting the user. Refer to Appendix for possible causes of errors.

Section 2.1

edit command: ASM

purpose : ASseMble MAC/65 source files

usage: ASM [#file1],[#file2],[#file3],[#file4]

ASM will assemble the specified source file and will produce a listing and object code output; the listing may include a full cross reference of all non-local labels. File1 is the source device, file2 is the list device, file3 is the object device, and file4 is a temporary file used to help generate the cross reference listing.

Any or all of the four filespec's may be omitted, in which case MAC/65 assumes the following default filespec(s) are to be used:

file1 - user source memory.
file2 - screen editor.
file3 - memory (CAUTION: see below)
file4 - none, therefore no cross reference

A filespec (#file1, #file3, etc.) can be omitted by substituting a comma in which case the respective default will be used.

For the listing file ONLY, you may use the special form "#-", to indicate that you do NOT want a listing file at all.

Some Examples:

Example: ASM #D2:SOURCE,#D:LIST,#D2:OBJECT

In this example, the source will come from D2:SOURCE, the assembler will list to D:LIST, and the object code will be written to D2:OBJECT.

Example: ASM #D:SOURCE , , #D:OBJECT

In this example, the source will be read from D:SOURCE and the object will be written to D:OBJECT. The assembly listing will be written to the screen.

Example: ASM , #P: , , #D:TEMP

In this example, the source will be read from memory, the object will be written to memory (but ONLY if the ".OPT OBJ" directive is in the source), and the assembly listing will be written to the printer along with the complete label cross reference. The file TEMP on disk drive 1 will be created and used as a temporary file for the cross reference.

Example: ASM #D:SOURCE , #P:

In this example, the source will be read from D:SOURCE and the assembly listing will be written to the printer. If the ".OPT OBJ" directive has been selected in the source, the object code will be placed in memory.

Example: ASM ,#-

This produces what is probably the fastest possible MAC/65 assembly. Source code is read from memory and no listing is produced (because of the "#-"). If your program does not contain a ".OPT OBJ" line, this becomes what is essentially simply an error checking assembly. (Though even if you ARE producing object code, the assembly speed is extremely fast.)

SPECIAL NOTES

Note: If assembling from a "filespec", the source MUST have been a SAVEd file.

Note: Refer to the .OPT directive for specific information on assembler listing and object output.

Note: The object code file will have the format of compound files created by the DOS XL SAVE command. See the DOS XL manual for a discussion of LOAD and SAVE file formats.

Note: You may use #C: as a device for the listing or object files. You may NOT use #C: for the source or cross reference files (thus implying that you may not get a cross reference unless you have a disk drive). HOWEVER, we do not recommend using the cassette as the object file device, since you may get an excessively long leader tone (which will be difficult to re-BLOAD later). Instead, we suggest using BSAVE (after assembling directly to memory) whenever practicable.