Introduction

Congratulations on choosing **Let's C**, the Mark Williams C compiler for the IBM PC and compatibles. **Let's C** has the state-of-the-art power and flexibility that the professional programmer needs, but is easy enough for the beginner to learn quickly.

Let's C is part of the Mark Williams Company family of C compilers, which supports many different operating systems and processors. The operating systems supported include:

COHERENT	MS-DOS	TOS
CP/M-68K	RMX	VAX/VMS
ISIS-II		

The processors supported include:

PDP-11	68000	80186
Z8001	68020	80286
Z8002	8086	

What is Let's C?

Let's C is a professional C programming system designed for the IBM PC and compatibles. It consists of the following:

- The Mark Williams C compiler, plus an assembler, a preprocessor, and other tools.
- A set of commands selected from the COHERENT operating system, including the MicroEMACS screen editor and the **make** programming discipline.
- A full set of C libraries.
- A set of sample programs, including full source code for the MicroEMACS editor.
- The Mark Williams shell MWS. MWS will help you build commands for **Let's C**, and will accelerate the operation of your software. By using MWS's display interface, you build commands for **Let's C** and its utilities. MWS also includes an accelerator that speeds up **Let's C**. If you prefer to type commands directly into MS-DOS, MWS will let you and it will still accelerate your software for you.

Hardware requirements

Let's C runs on an IBM PC, XT, or AT, or any compatible computer that has at least 320 kilobytes of RAM and either two double-sided floppy disk drives or at least one floppy disk drive and a hard disk.

Changes in release 4.0

Let's C version 4.0 has been greatly expanded and improved over earlier releases. Its new features include the following:

- **Let's C** now compiles into MS-DOS format, rather than the proprietary Mark Williams format used in earlier versions.
- It supports LARGE model as well as SMALL model.
- **Let's C** supports the i8087 mathematics co-processor, to speed up mathematics routines.

- **Let's C** supports **csd**, the revolutionary Mark Williams C source debugger. **csd** can now debug programs in MS-DOS object format, and that are compiled into either SMALL or LARGE model.
- **Let's C** libraries will sense the presence of the i8087 co-processor. If one is present, then it is used to execute mathematics routines; if one is not present, mathematics routines are emulated in software. Your programs will now make maximum use of your computer, whether an i8087 is present or not.
- The **make** programming discipline is included. This helps you to construct large programs that use many modules, with a minimum of difficulty.
- **Let's C**'s assembler, **as**, has been improved to support both LARGE and SMALL model, as well as the i8087 co-processor. It generates MS-DOS object format rather than the Mark Williams proprietary format, as before. **as** now supports a macro processing feature, which allows you to write model-independent versions of your assembly-language programs.
- The utility **fixobj** lets you edit object modules and libraries so they can be linked with object modules generated by **Let's C**. You can now use libraries from any other C compiler with **Let's C**.
- The Mark Williams shell MWS now makes it easy to build commands for **Let's C** and its utilities. MWS also supports RAM compiling, to speed up compilation on your system.
- The MicroEMACS screen editor is now integrated with **Let's C**. If you wish, you can have MicroEMACS display your source code automatically whenever an error occurs during compilation; you can then fix your error and recompile by using only a few keystrokes.
- Floating-point numbers now use IEEE format. Floating-point routines have been rewritten to execute more quickly than before.
- Division of **long**s has been rewritten, and is much faster than in previous versions.
- **Let's C** will now compile programs using i80286 instructions. Although such programs cannot be run on a computer that uses the i8086 microprocessor, they will run more quickly on the IBM PC-AT and compatibles.
- Finally, the manual for **Let's C** has been entirely rewritten, and now uses the Mark Williams Lexicon format. This format has set the standard for language documentation on the Atari ST, and **Let's C** is the first C compiler to bring Lexicon format to the IBM PC.

How to use this manual

This manual is in nine sections. Section 1, which you are now reading, introduces **Let's C**.

Section 2 shows you how to install **Let's C** on your computer. It also introduces the **Let's C** shell, introduces the MicroEMACS screen editor, and shows you how to compile simple C programs.

Section 3 is entitled **C for Beginners**. If you are new to the C programming language, this section will introduce you to C. It is not a full tutorial on C, but it will show you the basics of C programming, so you will be better able to follow the rest of this manual and use the example programs in it.

Section 4 introduces compiling with **Let's C**. It describes the options to the compiler controller **cc**, and shows you how to compile using different memory models and different formats. Technical issues that involve the i8086 microprocessor and MS-DOS are also discussed.

Section 5 is a tutorial on the MicroEMACS screen editor. It introduces most of the MicroEMACS commands and includes exercises to help sharpen your skills at editing programs.

Section 6 is a tutorial on **make**, the Mark Williams programming discipline. **make** is one of the most useful tools available for constructing and maintaining large, intricate programs. This section describes **make**, from building relatively simple programs to using **make** to control work other than compiling C programs.

Section 7 presents some questions and answers about **Let's C**. New users of **Let's C** often ask the same questions about how to use it; so if you have a question, look here first. You could well find the answer you need.

Section 8 lists all of the error messages that the **Let's C** compiler, assembler, and utilities can produce. Many entries have hints to help you correct or avoid the error that the message describes.

Finally, section 9 is the Lexicon. This is by far the largest part of the manual. The Lexicon contains several hundred individual entries; each describes a command, a function, defines a C technical term, or gives you other useful information. All of the Lexicon's entries are in alphabetical order, and are designed to be easily used. For example, if you want information on how to use the STDIO routines, simply turn to the entry in the Lexicon on **STDIO**; there, you will find a list of all the STDIO routines, a description of each, and instructions on how to use them. Or, if you want information on how **Let's C** encodes floating point numbers, simply turn to the entry on **float**. There, you will find a full description of floating point numbers. Many Lexicon entries have full C programs as examples; all have cross-references to related entries.

The opening sections of this manual will refer constantly to the Lexicon. If you are unfamiliar with a technical term used in this manual, look it up in the Lexicon. Chances are, you will find a full explanation. If you are not sure how to use the Lexicon, look up the entry for **Lexicon** within the Lexicon. This will help you get started.

Finally, the back of the manual lists the Lexicon's entries sorted by category, and gives an index.

User registration and reaction report

Before you continue, fill out the User Registration Card that came with your copy of **Let's C**. When you return this card, you become eligible for direct telephone support from the Mark Williams Company technical staff, and you will automatically receive information about all new releases and updates.

If you have comments or reactions to the **Let's C** software or documentation, please fill out and mail the User Reaction Report included at the end of the manual. We especially wish to know if you found errors in this manual. Mark Williams Company needs your comments to continue to improve **Let's C**.

Technical support

Mark Williams Company provides free technical support to all registered users of **Let's C**. If you are experiencing difficulties with **Let's C**, outside the area of programming errors, feel free to contact the Mark Williams Technical Support Staff. You can telephone during business hours (Central time), or write. This support is available *only* if you have returned your User Registration Card for **Let's C**.

If you telephone Mark Williams Company, please have at hand your manual for **Let's C**. Please collect as much information as you can concerning your difficulty before you call. If you write, be sure to include the product serial number (from the sticker on the back of this manual) and your return address.

Bibliography

The following books may be helpful in developing your skills with C. This list also contains all books that are referenced in this manual. It is by no means exhaustive; however, it should prove helpful to both beginners and experienced programmers.

American National Standards Institute: *Draft Programming Language C (October 1986 Draft)*. Washington, D.C.: X3 Secretariat, Computer and Business Equipment Manufacturers Association, 1986.

AT&T Bell Laboratories: *The C Programmer's Handbook*. Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1985.

Chirlin, P.M.: *Introduction to C*. Beaverton, Or.: Matrix Publishers, Inc., 1984.

Derman, B. (ed.): *Applied C*. New York: Van Nostrand Reinhold Co., Inc., 1986.

Feuer, A.R.: *The C Puzzle Book*. Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1982.

Gehani, G.: *Advanced C: Food for the Educated Palate*. Rockville, Md.: Computer Science Press, 1985.

Hancock, L.: Krieger, M.: *The C Primer*. New York: McGraw-Hill Book Publishers, Inc., 1982.

Harbison, S.; Steele, G.: *C: A Reference Manual*. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1984.

Hogan, T.: *The C Programmer's Handbook*. Bowie, Md.: Brady Publishing, 1984.

Kelley, A.; Pohl, I.: *C by Dissection: The Essentials of C Programming*. Menlo Park, Ca.: The Benjamin/Cummings Publishing Company, Inc., 1987.

Kernighan, B.W.; Ritchie, D.M.: *The C Programming Language*. Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1978.

Kernighan, B.W.; Plauger, P.J.: *The Elements of Programming Style*, ed. 2. New York: McGraw-Hill Book Co., 1978.

Kochan, S.G.: *Programming in C*. Hasbrouck Heights, N.J.: Hayden Book Co., Inc., 1983.

Knuth, D.E.: *The Art of Computer Programming*, vol. 1: *Basic Algorithms*. Reading, Ma.: Addison-Wesley Publishing Co., 1969.

Knuth, D.E.: *The Art of Computer Programming*, vol. 2: *Seminumerical Algorithms*. Reading, Ma.: Addison-Wesley Publishing Co., 1969.

Knuth, D.E.: *The Art of Computer Programming*, vol. 3: *Sorting and Searching*. Reading, Ma.: Addison-Wesley Publishing Co., 1969.

Plum, T.: *Learning to Program in C*. Cardiff, N.J.: Plum Hall, Inc., 1983.

Plum, T.: *C Programming Guidelines*. Cardiff, N.J.: Plum Hall, Inc., 1984.

Plum, T.; Brodie, J.: *Efficient C*. Cardiff, NJ: Plum Hall, Inc., 1985.

Purdum, J.: *C Programming Guide*. Indianapolis: Que Corp., 1983.

Purdum, J.; Leslie, T.C.; Stegemoller, A.L.: *C Programmer's Library*. Indianapolis: Que Corp., 1984.

Traister, R.J.: *Programming in C for the Microprocessor User*. Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1984.

Traister, R.J.: *Going from BASIC to C*. Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1984.

Vile, R.C., Jr.: *Programming in C with Let's C*. Glenview, IL: Scott, Foresman and Company, 1988.

Waite, M.; Prata, S.; Martin, D.: *C Primer Plus*. Indianapolis: Howard W. Sams, Inc., 1984.

Weber Systems, Inc.: *C Language User's Handbook*. New York: Ballantine Books, 1984.

Zahn, C.T.: *C Notes*. New York: Yourdan Press, 1979.

i8086/MS-DOS information

Duncan, R.: *Advanced MS-DOS: The Microsoft guide for assembly language and C programmers.* Redmond, WA: Microsoft Press, 1986. *Recommended.*

IBM Corporation: *Technical Reference, Personal Computer XT*. Boca Raton, FL: International Business Machines Corporation, 1983.

Intel Corporation: *8086 Relocatable Object Module Formats*. Document No. 121748-001. Santa Clara, CA: Intel Corporation.

Intel Corporation: *8086/8087/8088 Assembly Language Reference Manual for 8086-Based Development Systems*. Santa Clara, CA: Intel Corporation, 1983.

Intel Corporation: *iAPX 286 Programmer's Reference Manual*. Santa Clara, CA: Intel Corporation, 1985.

Microsoft Corporation: *MS-DOS Technical Reference Encyclopedia.* Bellevue, WA: Microsoft Press, 1986.

Norton, P.: *The Peter Norton Programmer's Guide to the IBM PC.* Bellevue, WA: Microsoft Press, 1985.

Young, M.: *Performance Programming Under MS-DOS.* Alameda, CA: SYBEX, Inc., 1987. *Recommended.*