



Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!



Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832

Email & Skype: info@chipsmall.com Web: www.chipsmall.com

Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China





**MPLAB[®] C17
C COMPILER
USER'S GUIDE**

Note the following details of the code protection feature on PICmicro® MCUs.

- The PICmicro family meets the specifications contained in the Microchip Data Sheet.
- Microchip believes that its family of PICmicro microcontrollers is one of the most secure products of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the PICmicro microcontroller in a manner outside the operating specifications contained in the data sheet. The person doing so may be engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable”.
- Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our product.

If you have any further questions about this matter, please contact the local sales office nearest to you.

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip's products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights.

Trademarks

The Microchip name and logo, the Microchip logo, KEELOQ, MPLAB, PIC, PICmicro, PICSTART and PRO MATE are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.


FilterLab, microID, MXDEV, MXLAB, PICMASTER, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

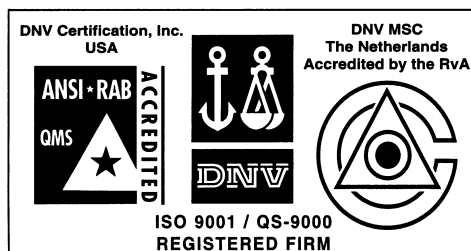
dsPIC, dsPICDEM.net, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, microPort, Migratable Memory, MPASM, MPLIB, MPLINK, MPSIM, PICC, PICDEM, PICDEM.net, rPIC, Select Mode and Total Endurance are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

Serialized Quick Turn Programming (SQTP) is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2002, Microchip Technology Incorporated. Printed in the U.S.A., All Rights Reserved.

 Printed on recycled paper.



Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999 and Mountain View, California in March 2002. The Company's quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, non-volatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001 certified.

Table of Contents

Preface	1
SECTION 1 – MPLAB C17 BASICS	
<hr/>	
Chapter 1. Compiler Overview and Installation	
1.1 Introduction	9
1.2 Highlights	9
1.3 MPLAB C17 Description	9
1.4 Basic Functionality	10
1.5 Input/Output Files	12
1.6 Reserved Resources	12
1.7 Host Computer System Requirements	12
1.8 Compiler Versions	13
1.9 Install/Uninstall the Compiler	13
Chapter 2. Differences Between MPLAB C17 and ANSI C	
2.1 Introduction	15
2.2 Highlights	15
2.3 MPLAB C17 vs. ANSI C	15
2.4 Components of a Basic MPLAB C17 Program	16
2.5 Keyword Differences	16
2.6 Statement Differences	17
Chapter 3. Using MPLAB C17 with MPLAB IDE	
3.1 Introduction	23
3.2 Highlights	23
3.3 MPLAB Projects Overview	23
3.4 Using MPLAB C17 with MPLAB IDE	25
3.5 Code Development	36
3.6 Additional Options and Library Information	36
Chapter 4. Using MPLAB C17 on the Command Line	
4.1 Introduction	37
4.2 Highlights	37
4.3 Command Line Overview	37
4.4 Using MPLAB C17 on the Command Line	39
4.5 Code Development	43
4.6 Library Information	43

MPLAB® C17 C Compiler User's Guide

SECTION 2 – MPLAB C17 ADVANCED USAGE

Chapter 5. Runtime Environment

5.1 Introduction	47
5.2 Highlights	47
5.3 Code and Data Sections	47
5.4 Startup and Initialization	49
5.5 Memory Models	51
5.6 Locating Code	51
5.7 Locating Data	51
5.8 Software Stack	52
5.9 Software Stack Call Conventions	53
5.10 Function Call Conventions	53
5.11 Interrupt Support Macros	54

Chapter 6. Data Types

6.1 Introduction	59
6.2 Highlights	59
6.3 Data Representation	59
6.4 Integer	59
6.5 Floating Point	60

Chapter 7. Device Support Files

7.1 Introduction	61
7.2 Highlights	61
7.3 Processor Header File	61
7.4 Register Definitions File	62
7.5 Using SFRs	64

Chapter 8. Interrupts

8.1 Introduction	65
8.2 Highlights	65
8.3 Writing an Interrupt Service Routine	66
8.4 Writing the Interrupt Vector	67
8.5 Interrupt Service Routine Context Saving	68
8.6 Latency	68
8.7 Nesting Interrupts	68
8.8 Enabling/Disabling Interrupts	69

Chapter 9. Mixing Assembly Language and C Modules

9.1 Introduction	71
9.2 Highlights	71
9.3 Internal Assembler	71
9.4 Calling Conventions	72

Table of Contents

9.5 Mixing Assembly Language and C Variables and Functions	72
9.6 Using Inline Assembly Language	73
Chapter 10. Writing Efficient Code	
10.1 Introduction	75
10.2 Highlights	75
10.3 Static Locals And Parameters	75
10.4 Optimization Tips	75
SECTION 3 – REFERENCES	
Chapter 11. Enabling/Disabling Interrupts	
11.1 Introduction	79
11.2 Highlights	79
11.3 Enabling Interrupts	79
11.4 Disabling Interrupts	102
Chapter 12. Implementation-Defined Behavior	
12.1 Introduction	113
12.2 Highlights	113
12.3 Identifiers	113
12.4 Characters	113
12.5 Integers	114
12.6 Floating Point	114
12.7 Arrays and Pointers	115
12.8 Registers	115
12.9 Structures and Unions	115
12.10 Bit-Fields	115
12.11 Enumerations	115
12.12 Switch Statement	116
12.13 Preprocessing Directives	116
Chapter 13. MPLAB C17 Diagnostics	
13.1 Introduction	117
13.2 Highlights	117
13.3 Errors	117
13.4 Warnings	121

MPLAB® C17 C Compiler User's Guide

SECTION 4 – APPENDICES

Appendix A. Reference Documents

A.1 Introduction	125
A.2 Highlights	125
A.3 C Standards Information	125
A.4 General C Information	125

Appendix B. Example Programs

B.1 Introduction	127
B.2 Highlights	127
B.3 Overview of Example Files	127
B.4 Example Details	127

Appendix C. ASCII Character Set 129

Glossary 131

Index 147

Worldwide Sales and Service 154

Preface

INTRODUCTION

The purpose of this user's guide is to help you get up and running with Microchip's MPLAB C17 C Compiler.

This manual is written with the intent that you are at least familiar with using the C programming language. If not, check Appendix A for a list of reference books that cover C programming.

HIGHLIGHTS

Items discussed in this chapter are:

- About this Guide
- Warranty Registration
- Recommended Reading
- Troubleshooting
- Microchip On-Line Support
- Customer Change Notification Service
- Customer Support

ABOUT THIS GUIDE

Document Layout

This document describes how to use MPLAB C17 to write C code for PICmicro® microcontroller applications. For a detailed discussion about basic MPLAB IDE functions, refer to the *MPLAB IDE User's Guide (DS51025)*.

This user's guide layout is as follows:

Section 1 – MPLAB C17 Basics

- **Chapter 1: Compiler Overview and Installation** – provides an overview of the MPLAB C17 C compiler, including compiler operation, input/output files and resource requirements. Also gives instructions on how to install or uninstall the compiler onto your system.
- **Chapter 2: Differences Between MPLAB C17 and ANSI C** – describes how MPLAB C17 differs from standard ANSI C. Compares MPLAB C17 and ANSI C and then highlights keyword, statement and standard function differences.
- **Chapter 3: Using MPLAB C17 with MPLAB IDE** – describes how to use MPLAB C17 with the MPLAB IDE v5.xx or below. Code development and other hardware tools are available when using MPLAB IDE.
- **Chapter 4: Using MPLAB C17 on the Command Line** – describes how to use the MPLAB C17 compiler from the command line interface. More compiler options are available on the command line.

Section 2 – MPLAB C17 Advanced Usage

- **Chapter 5: Runtime Environment** – describes the MPLAB C17 runtime environment. Includes information on code and data sections, startup and initialization, memory models and the software stack.
- **Chapter 6: Data Types** – describes MPLAB C17 data types.
- **Chapter 7: Device Support Files** – discusses the device support files used by MPLAB C17, namely processor header files and register definitions files.
- **Chapter 8: Interrupts** – describes how to use interrupts. Detailed interrupt usage may be found in the reference section.
- **Chapter 9: Mixing Assembly Language and C Modules** – provides guidelines to using C with assembly language modules.
- **Chapter 10: Writing Efficient Code** – provides guidelines to writing efficient MPLAB C17 code.

Section 3 – References

- **Chapter 11: Enabling/Disabling Interrupts** – detailed information on how to enable and disable interrupts.
- **Chapter 12: Implementation-Defined Behavior** – details MPLAB C17 specific parameters described as implementation-defined in the ANSI standard.
- **Chapter 13: MPLAB C17 Diagnostics** – lists errors and warnings generated by MPLAB C17.

Section 4 – Appendices

- **Appendix A: Reference Documents** – gives references that may be helpful in programming with MPLAB C17.
- **Appendix B: Example Programs** – discusses the MPLAB C17 examples included in the `examples` directory.
- **Appendix C: ASCII Character Set** – contains the ASCII character set.
- **Glossary** – A glossary of terms used in this guide.
- **Index** – Cross-reference listing of terms, features and sections of this document.
- **Worldwide Sales and Service** – gives the address, telephone and fax numbers for Microchip Technology Inc. sales and service locations throughout the world.

Conventions Used in this Guide

This manual uses the following documentation conventions:

Table: Documentation Conventions

Description	Represents	Examples
Code (Courier font):		
Plain characters	Sample code Filenames and paths	#define START c:\autoexec.bat
Angle brackets: < >	Variables	<label>, <exp>
Square brackets []	Optional arguments	MPASMWIN [main.asm]
Curly brackets and pipe character: { }	Choice of mutually exclusive arguments; An OR selection	errorlevel {0 1}
Lower case characters in quotes	Type of data	"filename"
Ellipses...	Used to imply (but not show) additional text that is not relevant to the example	list ["list_option...", "list_option"]
0xnnn	A hexadecimal number where n is a hexadecimal digit	0xFFFF, 0x007A
Italic characters	A variable argument; it can be either a type of data (in lower case characters) or a specific example (in uppercase characters).	char isascii (char, ch);
Interface (Arial font):		
Underlined, italic text with right arrow	A menu selection from the menu bar	<u>File</u> > <u>Save</u>
Bold characters	A window or dialog button to click	OK, Cancel
Characters in angle brackets < >	A key on the keyboard	<Tab>, <Ctrl-C>
Documents (Arial font):		
Italic characters	Referenced books	<i>MPLAB IDE User's Guide</i>

Documentation Updates

All documentation becomes dated, and this user's guide is no exception. Since MPLAB IDE, MPLAB C17 and other Microchip tools are constantly evolving to meet customer needs, some actual dialogs and/or tool descriptions may differ from those in this document. Please refer to our web site to obtain the latest documentation available.

Documentation Numbering Conventions

Documents are numbered with a "DS" number. The number is located on the bottom of each page, in front of the page number. The numbering convention for the DS Number is: DSXXXXXA,

where:

- XXXXX = The document number.
- A = The revision level of the document.

MPLAB® C17 C Compiler User's Guide

WARRANTY REGISTRATION

Please complete the enclosed Warranty Registration Card and mail it promptly. Sending in your Warranty Registration Card entitles you to receive new product updates. Interim software releases are available at the Microchip web site.

RECOMMENDED READING

This user's guide describes how to use MPLAB C17 C Compiler. For more information on included libraries and precompiled object files for the compilers, the operation of MPLAB IDE and the use of other tools, the following are recommended reading.

README.C17

For the latest information on using MPLAB C17 C Compiler, read the README.C17 file (ASCII text) included with the software. This README file contains update information that may not be included in this document.

README.XXX

For the latest information on other Microchip tools (MPLAB IDE, MPLINK™ linker, etc.), read the associated README files (ASCII text file) included with the MPLAB IDE software.

MPLAB C17 C Compiler Libraries (DS51296)

Reference guide for MPLAB C17 libraries and precompiled object files. Lists all library functions with a detailed description of their use.

MPLAB IDE User's Guide (DS51025)

Comprehensive guide that describes installation and features of Microchip's MPLAB Integrated Development Environment (IDE), as well as the editor and simulator functions in the MPLAB IDE environment.

MPASM™ User's Guide with MPLINK™ and MPLIB™ (DS33014)

This user's guide describes how to use the Microchip PICmicro MCU MPASM assembler, the MPLINK object linker and the MPLIB object librarian.

Technical Library CD-ROM (DS00161)

This CD-ROM contains comprehensive application notes, data sheets and technical briefs for all Microchip products. To obtain this CD-ROM, contact the nearest Microchip Sales and Service location (see back page).

Microchip Web Site

The Microchip web site (www.microchip.com) contains a wealth of documentation. Individual data sheets, application notes, tutorials and user's guides are all available for easy download. All documentation is in Adobe™ Acrobat (pdf) format.

Microsoft® Windows® Manuals

This manual assumes that users are familiar with the Microsoft Windows operating system. Many excellent references exist for this software program, and should be consulted for general operation of Windows.

TROUBLESHOOTING

See the README files for information on common problems not addressed in this user's guide.

MICROCHIP ON-LINE SUPPORT

Microchip provides on-line support on the Microchip web site at:

www.microchip.com

A file transfer site is also available by using an FTP service connecting to:

ftp://ftp.microchip.com

The web site and file transfer site provide a variety of services. Users may download files for the latest development tools, data sheets, application notes, user's guides, articles and sample programs. A variety of Microchip specific business information is also available, including listings of Microchip sales offices and distributors. Other information available on the web site includes:

- Latest Microchip press releases
- Technical support section with FAQs
- Design tips
- Device errata
- Job postings
- Microchip consultant program member listing
- Links to other useful web sites related to Microchip products
- Conferences for products, development systems, technical information and more
- Listing of seminars and events

CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip started the customer notification service to help customers stay current on Microchip products with the least amount of effort. Once you subscribe, you will receive email notification whenever we change, update, revise or have errata related to your specified product family or development tool of interest.

Go to the Microchip web site (www.microchip.com) and click on Customer Change Notification. Follow the instructions to register.

The Development Systems product group categories are:

- Compilers
- Emulators
- In-Circuit Debuggers
- MPLAB IDE
- Programmers

Here is a description of these categories:

Compilers - The latest information on Microchip C compilers and other language tools. These include the MPLAB C17, MPLAB C18 and MPLAB C30 C Compilers; MPASM and MPLAB ASM30 assemblers; MPLINK and MPLAB LINK30 linkers; and MPLIB and MPLAB LIB30 librarians.

Emulators - The latest information on Microchip in-circuit emulators. This includes the MPLAB ICE 2000.

In-Circuit Debuggers - The latest information on Microchip in-circuit debuggers. These include the MPLAB ICD and MPLAB ICD 2.

MPLAB - The latest information on Microchip MPLAB IDE, the Windows Integrated Development Environment for development systems tools. This list is focused on the MPLAB IDE, MPLAB SIM simulator, MPLAB IDE Project Manager and general editing and debugging features.

Programmers - The latest information on Microchip device programmers. These include the PRO MATE II device programmer and PICSTART Plus development programmer.

CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributors
- Local Sales Office
- Field Application Engineers (FAEs)
- Corporate Applications Engineers (CAEs)
- Systems Information and Upgrade Hot Line

Customers should call their distributor or field application engineer (FAE) for support. Local sales offices are also available to help customers. See the last page of this document for a listing of sales offices and locations.

Corporate applications engineers (CAEs) may be contacted at (480) 792-7627.

Systems Information and Upgrade Line

The Systems Information and Upgrade Information Line provides system users with a listing of the latest versions of all of Microchip's development systems software products. Plus, this line provides information on how customers can receive the most current upgrade kits. The Information Line Numbers are:

1-800-755-2345 for U.S. and most of Canada.

1-480-792-7302 for the rest of the world.



MPLAB[®] C17 C COMPILER USER'S GUIDE

Section 1 – MPLAB C17 Basics

Chapter 1. Compiler Overview and Installation	9
Chapter 2. Differences Between MPLAB C17 and ANSI C.....	15
Chapter 3. Using MPLAB C17 with MPLAB IDE.....	23
Chapter 4. Using MPLAB C17 on the Command Line	37

Part
1

Basics

Part
2

Advanced Usage

Part
3

References

Part
4

Appendices

Chapter 1. Compiler Overview and Installation

1.1 INTRODUCTION

This chapter provides an overview of the MPLAB C17 C compiler and how to install the compiler on your computer.

1.2 HIGHLIGHTS

This chapter covers the following topics:

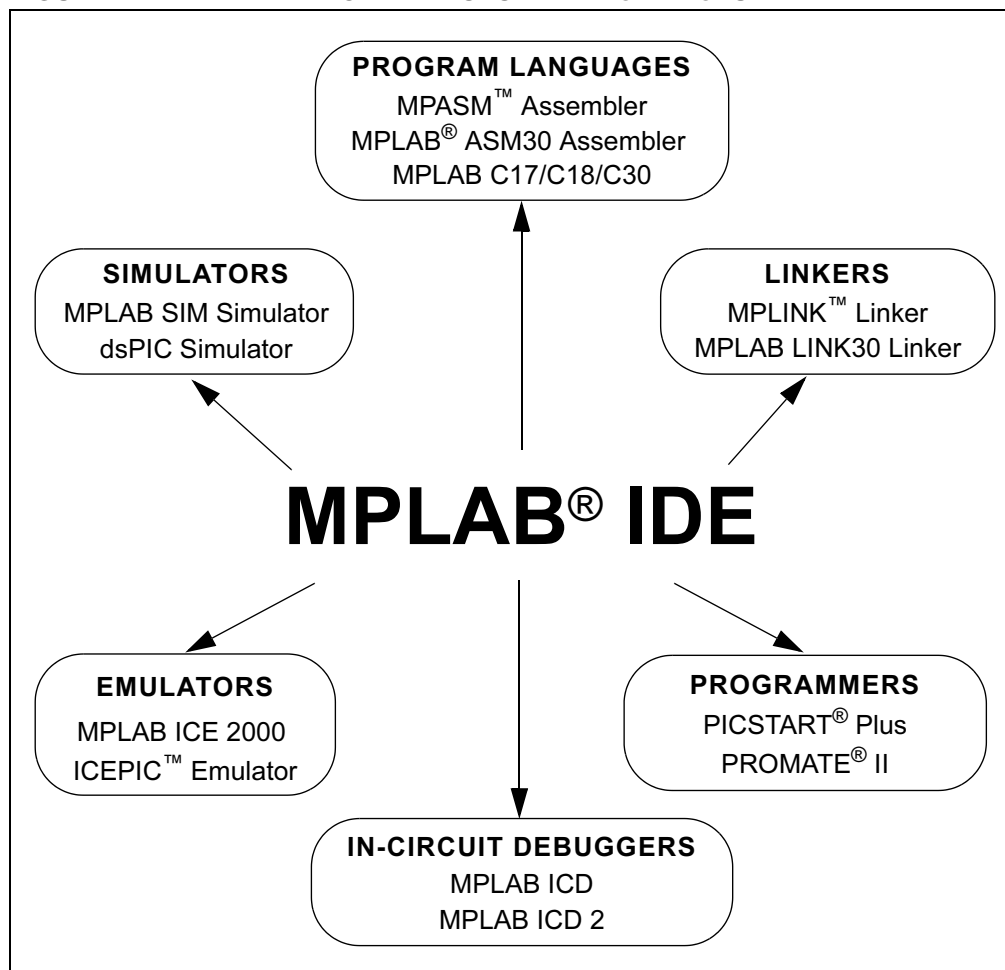
- MPLAB C17 Description
- Basic Functionality
- Input/Output Files
- Reserved Resources
- Host Computer System Requirements
- Compiler Versions
- Install/Uninstall the Compiler

1.3 MPLAB C17 DESCRIPTION

The MPLAB C17 compiler is a full-featured ANSI C compiler for Microchip's PIC17 PICmicro microcontrollers (MCU). The compiler is fully compatible with Microchip's MPLAB Integrated Development Environment (IDE) (Figure 1-1), allowing source-level debugging with both the MPLAB ICE in-circuit emulator and the MPLAB SIM simulator. MPLAB IDE provides a convenient, project-oriented development environment that reduces development time.

MPLAB C17 has implemented extensions to the C language to provide specific support for Microchip's PICmicro MCU peripherals. The C libraries include: A/D converter, Character Classification, External LCD, I²C™, Input Capture, Interrupt Support Macros, I/O Port, Memory/String Manipulation, Number/Text Conversion, Pulse Width Modulation, RESET, Relay, Software I²C, Software SPI™, Software USART, SPI, Timers and USART.

FIGURE 1-1: DEVELOPMENT SYSTEM ARCHITECTURE

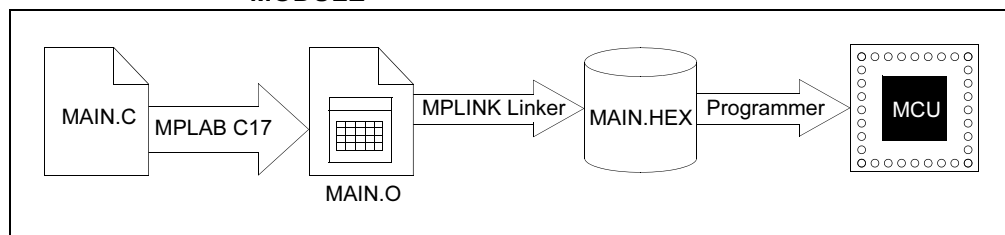


1.4 BASIC FUNCTIONALITY

MPLAB C17 generates object code from C source code. This object code is then input into Microchip's MPLINK linker to form the final executable code. A single C source file may be compiled into a single executable as shown in Figure 1-2, or it can be linked with other separately assembled or compiled modules as shown in Figure 1-3. Related modules can also be grouped and stored together in a library using Microchip's MPLIB Librarian (Figure 1-4). Required libraries can be specified at link time, and only the modules that are needed will be included in the final executable.

For more information on MPLINK linker and MPLIB librarian operation, please refer to the *MPASM™ User's Guide with MPLINK™ and MPLIB™* (DS33014).

FIGURE 1-2: GENERATING EXECUTABLE CODE FROM ONE OBJECT MODULE



Compiler Overview and Installation

FIGURE 1-3: GENERATING EXECUTABLE CODE FROM OBJECT MODULES

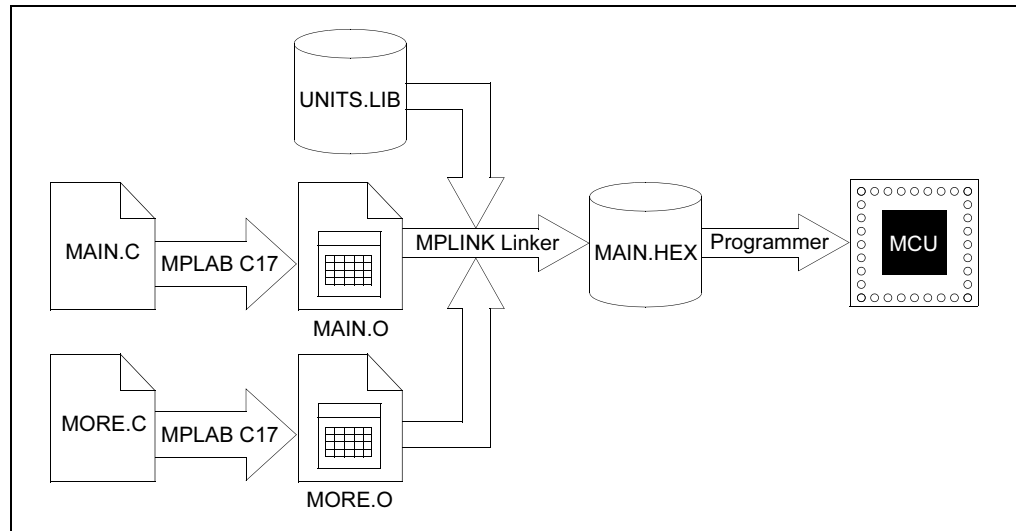
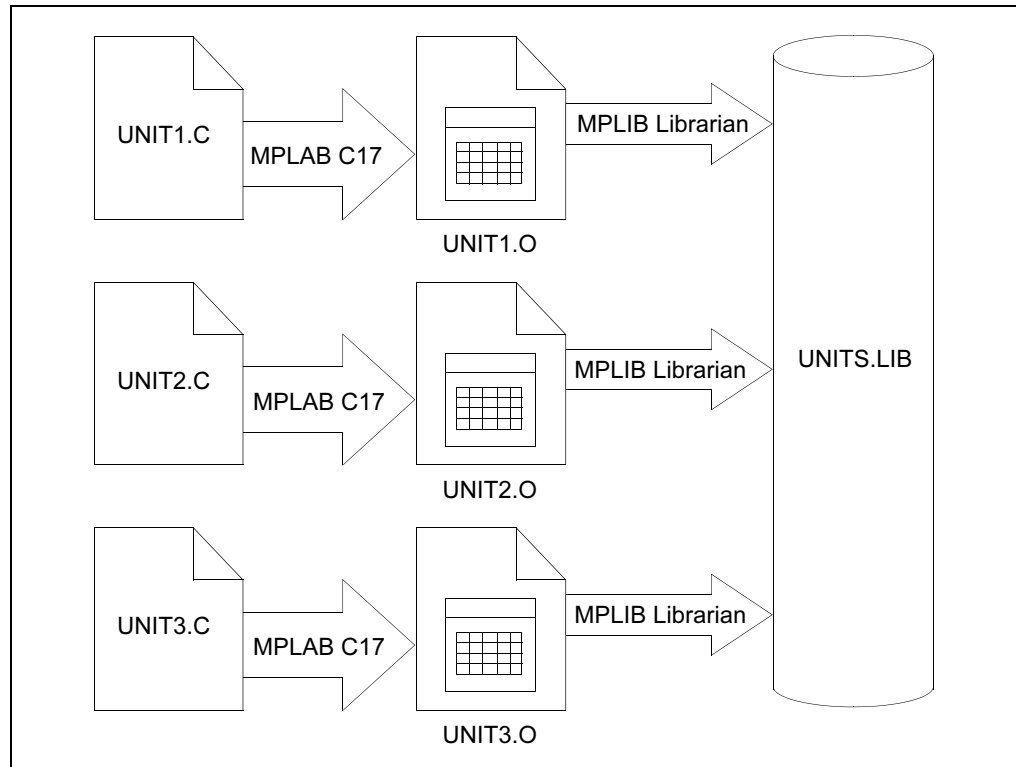


FIGURE 1-4: CREATING A REUSABLE OBJECT LIBRARY



Part
1

Basics

Part
2

Advanced Usage

Part
3

References

Part
4

Appendices

1.5 INPUT/OUTPUT FILES

These are the default file extensions used by MPLAB C17.

TABLE 1-1: MPLAB C17 DEFAULT EXTENSIONS

Extension	Purpose
.c	Default source file extension input to MPLAB C17: <source_name>.c
.err	Output extension from MPLAB C17 for error files: <source_name>.err
.o	Output extension from MPLAB C17 for object files: <source_name>.o

1.5.1 Source Code Format (.c)

The source code file can be created using any ASCII text file editor. It should conform to C language programming guidelines. For information on how to program using C, please refer to Appendix A.

1.5.2 Error File Format (.err)

By default MPLAB C17 generates an error file. This file can be useful when debugging your code. The MPLAB IDE automatically opens this file in the case of an error. The format of the messages in the error file is:

```
<type>[<number>] <file> <line> <description>
```

For example:

```
Error[113] C:\prog.c 7 : Symbol not previously defined  
  (start)
```

See the appendices for descriptions of error messages generated.

1.5.3 Object File Format (.o)

Object files are the relocatable code produced from source files.

1.6 RESERVED RESOURCES

The following are PICmicro MCU resource impacts from the compiler:

- **FSR0**: Can be used, but compiler may use also. Don't expect value to stay the same.
- **FSR1**: Reserved for compiler use.
- **PRODH, PRODL**: Can be used, but compiler may use also. Don't expect value to stay the same.
- **TBLPTRH, TBLPTRL, TBLAT**: Can be used, but compiler may use also. Don't expect value to stay the same.

1.7 HOST COMPUTER SYSTEM REQUIREMENTS

MPLAB C17 requires:

- PC-compatible 386 or better class system
- 16 MB memory (32 MB recommended)
- 5 MB hard disk space (10 MB recommended)
- MS-DOS® PC-DOS version 5.0 or greater, or Microsoft Windows® operating system (version 3.x or greater).

1.8 COMPILER VERSIONS

There are two versions of MPLAB C17:

- a DOS-extended or Windows 3.x version, `mcc17d.exe`
- a Windows 32-bit version (Windows 95 or greater), `mcc17.exe`

You can use both versions with MPLAB IDE; however, the Windows 32-bit version is recommended.

1.9 INSTALL/UNINSTALL THE COMPILER

If you are going to use MPLAB C17 with the MPLAB IDE, install the MPLAB IDE first. To install MPLAB C17, enter your Windows operating system, run the file `SETUP.EXE` on the CD-ROM, and follow the prompts.

The install program will use or create the directory you chose in the setup program. Then, it will install the MPLAB C17 components into seven subdirectories:

- `bin` – executable versions
- `doc` – help files
- `examples` – source code examples, with example-specific header, linker and batch files
- `h` – general header files
- `lib` – library and pre-compiled object files
- `lkr` – linker script files
- `src` – source code for files found in `lib` directory

In addition, the MPLAB C17 install will create an environment variable, `MCC_INCLUDE`, in your `AUTOEXEC.BAT` file. The `MCC_INCLUDE` environment variable specifies the directories to search for included files. For more information, refer to the `#include` directive. The install program will also add the compiler `bin` directory to your `PATH` so you can run the compiler from any other directory.

To uninstall the compiler:

1. From the Start menu, select *Settings > Control* Panel to display the Control Panel directory.
2. Double-click the Add/Remove Programs icon. Scroll down the list and locate the program you want to remove. Click **Remove**.

MPLAB® C17 C Compiler User's Guide

NOTES:

Chapter 2. Differences Between MPLAB C17 and ANSI C

2.1 INTRODUCTION

This chapter discusses the differences between MPLAB C17 and ANSI C.

2.2 HIGHLIGHTS

Items discussed in this chapter are:

- MPLAB C17 vs. ANSI C
- Components of a Basic MPLAB C17 Program
- Keyword Differences
- Statement Differences
- Oddities of Standard Functions

2.3 MPLAB C17 vs. ANSI C

Most C programmers have gained their experience programming C on computers where there was an operating system to handle such things as memory management, input/output, interdevice communications, etc. Microcontrollers, by their very nature, do not have the memory overhead for an operating system. Therefore, the compiler expects the user to implement memory allocation, I/O operation through a peripheral, and other specialized tasks. Libraries and precompiled object files are available with MPLAB C17 to aid the programmer in this endeavor.

An MPLAB C17 program is a collection of declarations, statements, comments and preprocessor directives that typically do the following:

- Declare data structures
- Allocate data space
- Evaluate expressions
- Perform program control operations
- Control PICmicro MCU peripherals

Additionally, after source code is compiled, it must be programmed into a device. In the device environment, RAM is in an undefined state on power-up. The program must take care of initializing any variables that are set in the code. This is accomplished by storing the variable values in program memory and then moving them to RAM before the `main()` function executes. There are other `main()` pre-execution items that may be necessary, such as setting up a software stack. These specialized items may be written in C or assembly code. In either case, the programmer must decide what is needed.

MPLAB® C17 C Compiler User's Guide

2.4 COMPONENTS OF A BASIC MPLAB C17 PROGRAM

The following is the shell for a basic MPLAB C17 source file, highlighting the elements of the program:

```
#include <p17CXX.h> ← preprocessor directive
void function1(void); ← prototyped function
void main(void) ← main routine
{
  /* User source code here */
}
void function1(void) ← function definition
{
  /* User function code here */
}
```

The first line is a preprocessor directive that includes the processor definition file (7.3 “Processor Header File”). This file defines processor-specific information such as special function registers.

Note: `p17CXX.h` includes proper processor-specific header file based on the processor selected on the command line.

The next line is a declaration of a function (2.6.9 “Functions”) that will be used in the main routine (`function1`). Placing the function declaration here is called prototyping. The function itself may then be defined after the main routine. Another option is to place the entire function definition in the prototype location.

Finally, the main routine is defined, with the appropriate source code between the braces. Note that the main routine is itself a function.

2.5 KEYWORD DIFFERENCES

The ANSI C standard defines 32 keywords for use in the C language. Typically, C compilers add keywords that take advantage of the processor's architecture. The following table shows the ANSI C and the MPLAB C17 keywords, where MPLAB C17 keywords are shown in bold.

TABLE 2-1: ANSI C AND MPLAB C17 KEYWORDS

<code>_asm</code>	<code>double</code>	<code>long</code>	<code>struct</code>
<code>_endasm</code>	<code>else</code>	<code>near</code>	<code>switch</code>
<code>auto</code>	<code>enum</code>	<code>ram</code>	<code>typedef</code>
<code>break</code>	<code>extern</code>	<code>register**</code>	<code>union</code>
<code>case</code>	<code>far</code>	<code>return</code>	<code>unsigned</code>
<code>char</code>	<code>float</code>	<code>rom</code>	<code>void</code>
<code>const</code>	<code>for</code>	<code>short</code>	<code>volatile</code>
<code>continue</code>	<code>goto</code>	<code>signed</code>	<code>while</code>
<code>default</code>	<code>if</code>	<code>sizeof</code>	
<code>do</code>	<code>int</code>	<code>static</code>	

** has no effect in MPLAB C17

2.6 STATEMENT DIFFERENCES

There are differences between how MPLAB C17 uses some statements and how these statements would be implemented under ANSI C. These specialized MPLAB C17 statements are:

- `#include filename`
- `#pragma` Statements
- Constants
- Variables
- Storage Classes
- Functions
- Operators
- `switch` Statement
- Initializing Arrays
- Pointers
- Structures
- Bit-fields

2.6.1 `#include filename`

Include the file *filename* into the MPLAB C17 program. Usually at least the processor header file is included, so that register and pin names may be used in code as opposed to addresses.

When “*filename*” is used, MPLAB C17 looks for the file in the current directory and then in the directories specified by the current include search path, which refers to the environment variable `MCC_INCLUDE` and command-line option ‘-i’.

When `<filename>` is used, MPLAB C17 looks for the file in the directories specified by the current include search path.

2.6.2 `#pragma interrupt fname`

2.6.2.1 DESCRIPTION

Declare a function to be an interrupt function. This pragma must come before the function definition, but may come after a prototype. The compiler will generate a separate temporary storage section dedicated to the function.

See Chapter 9 for more details on interrupt handling.

2.6.2.2 SYNTAX

```
interrupt-directive:  
    #pragma interrupt function-name [section-name]  
                    new-line
```

2.6.3 `#pragma list / #pragma nolist`

2.6.3.1 DESCRIPTION

The `#pragma list` directive turns on list file generation for all code following the directive. The `#pragma nolist` directive turns off list file generation for all code following the directive.

2.6.3.2 SYNTAX

```
list-directive:  
    #pragma list new-line  
    #pragma nolist new-line
```


2.6.4 #pragma sectiontype

2.6.4.1 DESCRIPTION

The section declaration family of pragmas changes the section into which MPLAB C17 will allocate data of the associated type. Optionally, the section may be allocated at an absolute address.

A section declaration with no name resets the allocation of data of the associated type to the default section for the current module.

A data section qualified as `shared` will be located in a `SHAREBANK` by the linker. Similarly, a data section qualified as `access` will be located in an `ACCESSBANK` by the linker.

Specifying a section name which has been previously declared causes MPLAB C17 to resume allocating data of the associated type into the specified section. The section qualifiers must match the previous declaration.

For `udata` and `idata` sections in MPLAB C17, the data section type, SFR or GPR, and a bank number may be optionally specified instead of an absolute address. This is functionally equivalent to specifying a `varlocate` pragma with the same information for each symbol declared in the section. Like `varlocate`, this qualifier provides information to the compiler only and is not enforced by the linker; therefore, care should be exercised in its use.

Note: Logical sections are used to specify which of the defined memory regions should be used for a portion of source code. For more on sections, refer to the MPLINK linker section of the *MPASM™ User's Guide with MPLINK™ and MPLIB™* (DS33014).

2.6.4.2 SYNTAX

section-directive:

```
#pragma udata [data-qualifier-list] [section-name
              [location]] new-line
#pragma idata [data-qualifier-list] [section-name
            [location]] new-line
#pragma romdata [overlay] [section-name] new-line
#pragma code [overlay] [section-name] new-line
```

data-qualifier:

```
shared
overlay
```

location:

```
= address
gpr bank-number
sfr bank-number
```

2.6.4.3 EXAMPLE

Declare a section for `udata` allocation at address `0x120`. The linker will enforce that the section will be located at address `0x120`.

```
#pragma udata myNewDataSection = 0x120
```

Resume allocation of `romdata` into the default section.

```
#pragma romdata
```

Declare a new code section at address `0x8000`.

```
#pragma code myExternalCodeSection=0x8000
```

2.6.4.4 SEE ALSO

```
#pragma varlocate
```

Differences Between MPLAB C17 and ANSI C

2.6.5 #pragma varlocate n name #pragma varlocate {gpr | sfr} name

2.6.5.1 DESCRIPTION

The `varlocate` pragma tells the compiler where a variable will be located at link time, enabling the compiler to perform more efficient bank switching. The bank may be specified (`n`) or the GPR or SFR address range may be specified.

`varlocate` specifications are not enforced by the compiler or linker. The sections which contain the variables should be assigned explicitly in the linker script, or via absolute sections in the module(s) where they are defined, into the correct bank.

2.6.5.2 SYNTAX

variable-locate-directive:

```
#pragma varlocate bank variable-name new-line
#pragma varlocate [bank-reg] variable-name new-line
#pragma varlocate section-name variable-name new-line
```

2.6.6 Constants

The MPLAB C17 compiler supports the usual four different kinds of constants:

- Integers
- Floating-point numbers
- Characters
- Strings

See Chapter 6 for more on data types.

2.6.7 Variables

The MPLAB C17 compiler supports the standard integer and floating-point numeric types. A plain `char` is signed by default. See Chapter 6 for more on data types.

The ranges of the standard integer types are documented in Table 2-2.

TABLE 2-2: STANDARD INTEGER TYPES

Name	Value	Meaning
CHAR_BIT	8	Width of <code>char</code> type, in bits
SCHAR_MIN	-128	Minimum value of signed <code>char</code>
SCHAR_MAX	127	Maximum value of signed <code>char</code>
UCHAR_MAX	255	Maximum value of unsigned <code>char</code>
SHRT_MIN	-32,768	Minimum value of short <code>int</code>
SHRT_MAX	32,767	Maximum value of short <code>int</code>
USHRT_MAX	65,535	Maximum value of unsigned short
INT_MIN	-32,768	Minimum value of <code>int</code>
INT_MAX	32,767	Maximum value of <code>int</code>
UINT_MAX	65,535	Maximum value of unsigned <code>int</code>
LONG_MIN	-2,147,483,648	Minimum value of long <code>int</code>
LONG_MAX	2,147,483,647	Maximum value of long <code>int</code>
ULONG_MAX	4,294,967,295	Maximum value of unsigned long
CHAR_MIN	If type <code>char</code> is signed by default, then SCHAR_MIN else 0.	Minimum value of <code>char</code>
CHAR_MAX	If type <code>char</code> is signed by default then SCHAR_MAX, else UCHAR_MAX.	Maximum value of <code>char</code>