Freescale Semiconductor
Application Note

Document Number: AN4094

Exporting CodeWarrior IDE Build Tools Settings into Command-Line Tools Options

by: Stanislav Slíva

Contents

1. Introduction

The purpose of this application note is to help you convert a project created in CodeWarrior IDE into a project that could build using CodeWarrior command-line build tools. Unfortunately there is no automatic conversion tool that would export IDE *project.mcp* file into a make file. This document helps you to translate settings made using IDE to equivalent command-line options.
It also includes a section that maps each compiler/linker/assembler settings in an IDE panel to the corresponding command-line options.

2. Command-Line Tools (CLT)

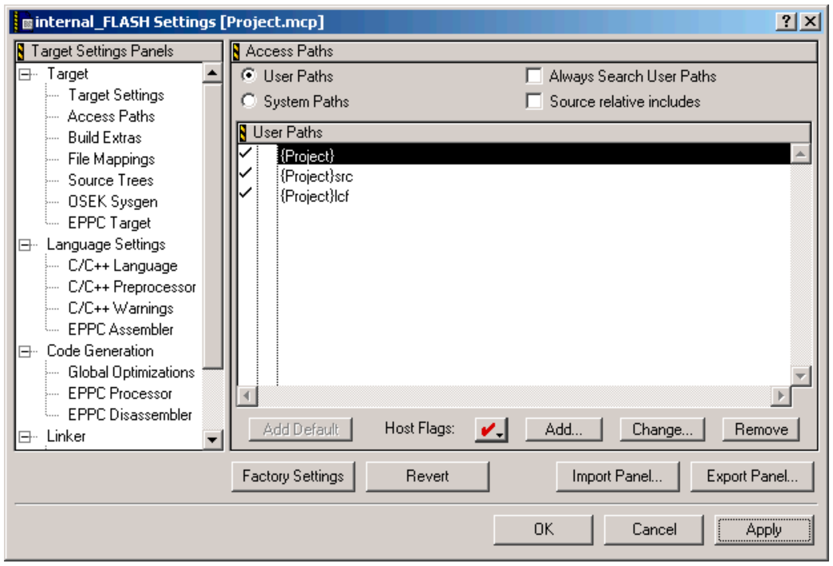
CodeWarrior build tools may be invoked from the command-line. These command-line tools operate almost identically to their counterparts in an Integrated Development Environment (IDE). CodeWarrior command-line compiler (*mwcceppc.exe*) and assembler (*mwasmeppc.exe*) translate source code files into object code files. CodeWarrior command-line linker (*mwldeppc.exe*) then combine one or more object code files to produce an executable image file, ready to load and execute on the target platform. Each command-line tool has options that you configure when you invoke the tool.
These tools are usually located here:

<CW installation directory>\PowerPC_EABI_Tool\Command_Line_Tools\*.exe

3. Target Settings

3.1. Access Paths Settings Panel

Figure 1. Access Paths Settings Panel


internal_FLASH Settings [Project.mcp]

Access Paths

(● User Paths) ○ System Paths

☐ Always Search User Paths

☐ Source relative includes

User Paths:

✓ {Project}

✓ {Project}src

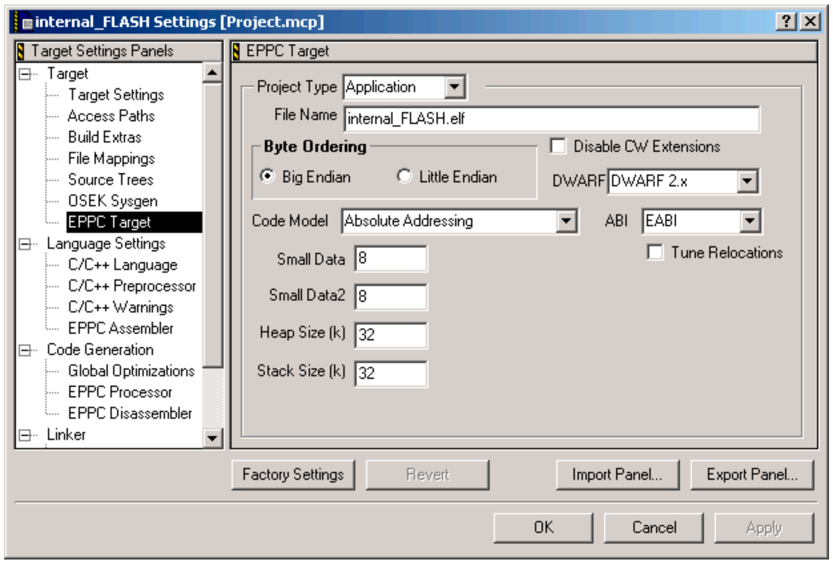
✓ {Project}lcf

Table 1. Command Line Options for Access Paths Settings Panel

Tool	IDE Options	Command-Line Options	Notes
Asm. Comp.	User Paths	-cwd **proj** | source | explicit | include	Controls where a search begins for #include "..." files.
Asm. Comp.	System Paths	-[no]**stdinc**	Uses standard system include paths for #include <...> files
Asm. Comp.	Always Search User Paths	-nosyspath	Treats #include <...> statements the same as #include "..." statements.
Comp.	Source relative includes	-[no]**convertpaths**	Searches for dependent files in the same location as the source file. If the dependent file is not found in this location, specified User and System paths are searched. If this option is enabled, the **Always Search User Paths** should also be enabled.

3.2. EPPC Target Settings Panel

Figure 2. EPPC Target Settings Panel



internal_FLASH Settings [Project.mcp]

Target Settings Panels

- Target
  - Target Settings
  - Access Paths
  - Build Extras
  - File Mappings
  - Source Trees
  - OSEK Sysgen
  - EPPC Target
- Language Settings
  - C/C++ Language
  - C/C++ Preprocessor
  - C/C++ Warnings
  - EPPC Assembler
- Code Generation
  - Global Optimizations
  - EPPC Processor
  - EPPC Disassembler
- Linker

EPPC Target

Project Type: Application
File Name: internal_FLASH.elf
Byte Ordering: [✓ Big Endian] Little Endian
Disable CW Extensions: ☐
DWARF: DWARF 2.x
Code Model: Absolute Addressing
ABI: EABI
Tune Relocations: ☐
Small Data: 8
Small Data2: 8
Heap Size (k): 32
Stack Size (k): 32

Table 2. Command-Line Options for EPPC Target Settings

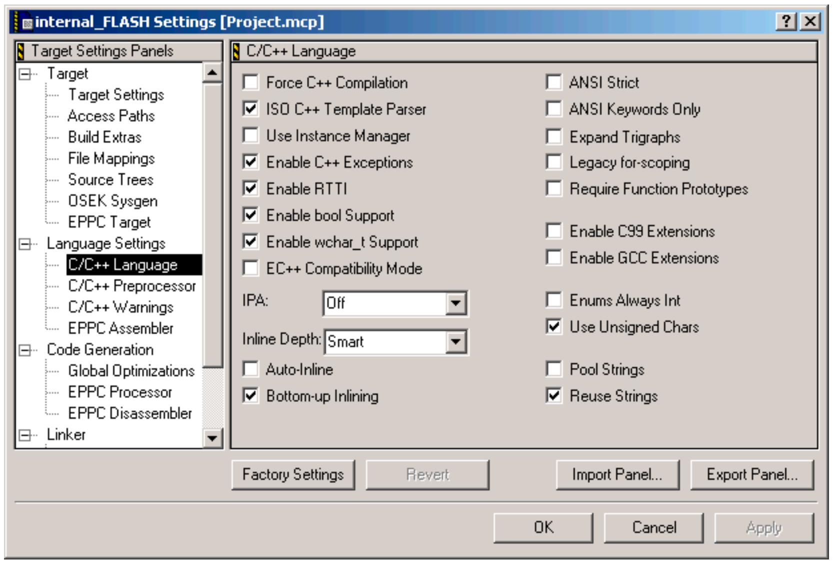
Tool	IDE Options	Command-Line Options	Notes
Linker	Project Type	-**application** | -library | (-partial | -r)	Application, Library, Partial Link
Linker	File Name	-o file_name	File name of the project will be generated when you build the (.mot, .elf, .map, .bin) project.
Linker Comp.	Byte Ordering	-**big** | -little	**Big Endian**: The bytes are organized in decreasing order with the last significant byte as first (B3, B2, B1, and B0). **Little Endian**: The bytes are organized in the increasing order with the least significant byte as first (B0, B1, B2, and B3).
Linker Comp.	Code Model	-model **absolute** | sda_pic_pid	**Absolute Addressing**: Generates non-relocatable binary files. **SDA Based PIC/PID Addressing**: Generates relocatable binary files that use Position-Independent-Code (PIC) / Position-Independent-Data (PID) addressing.
Linker Comp.	Small Data	-sdata[threshold] short	Default value is **8.** The linker stores small data items in the Small Data address space. The compiler can generate faster code to access this data.
Linker Comp.	Small Data2	-sdata2[threshold] short	Default value is **8** The linker stores read-only small data items in the Small Data2 address space. The compiler can generate faster code to access this data.
Linker	Heap Size (k)	-heapsize long	Default value is **1024.** The value that you enter is in kilobytes. Heaps are associated only with applications.
Linker	Stack Size (k)	-stacksize long	Default value is **64.** You can allocate stack and heap space based on the amount of memory that you have on your target hardware.

Tool	IDE Options	Command-Line Options	Notes
			If you allocate memory more than the available RAM, for the heap and/or stack then, your program will not run correctly.
Comp.	Disable CW Extensions	-disable_extensions on| **off**	If you are exporting code libraries from CodeWarrior software to other compilers/linkers, check the **Disable CW Extensions** checkbox to disable CodeWarrior features that may be incompatible.
Comp. Linker	DWARF	(-g[dwarf] | -sym dwarf-1,full) | (-gdwarf-2 | -sym dwarf-2,full)	Use the **DWARF** list box to select the version of the Debug With Arbitrary Record Format (DWARF) debugging information format. The linker ignores debugging information that is not in the format that you select from the DWARF list box. (Dwarf 1, Dwarf 2)
Linker	Tune Relocations	-tune_relocations	**Tune Relocations** option has the following effects: For EABI, the 14-bit branch relocations are changed to 24-bit branch relocations only if they cannot reach the calling site from the original relocation. For SDA PIC/PID, the absolute addressed references of data from code are changed to use a small data register instead of r0; absolute code is changed to code references to use the PC relative relocations.
Linker Comp.	ABI	-abi **eabi**| …	Use the **ABI** list box to select the Application Binary Interface (ABI) used for function calls and structure layout.

4. Language Settings

4.1. C/C++ Language Settings Panel

Figure 3. C/C++ Language Settings Panel



C/C++ Language

- [ ] Force C++ Compilation
- [✓] ISO C++ Template Parser
- [ ] Use Instance Manager
- [✓] Enable C++ Exceptions
- [✓] Enable RTTI
- [✓] Enable bool Support
- [✓] Enable wchar_t Support
- [ ] EC++ Compatibility Mode
- IPA: Off
- Inline Depth: Smart
- [ ] Auto-Inline
- [✓] Bottom-up Inlining
- [ ] ANSI Strict
- [ ] ANSI Keywords Only
- [ ] Expand Trigraphs
- [ ] Legacy for-scoping
- [ ] Require Function Prototypes
- [ ] Enable C99 Extensions
- [ ] Enable GCC Extensions
- [ ] Enums Always Int
- [✓] Use Unsigned Chars
- [ ] Pool Strings
- [✓] Reuse Strings

Table 3. Command-Line Options for Language Settings

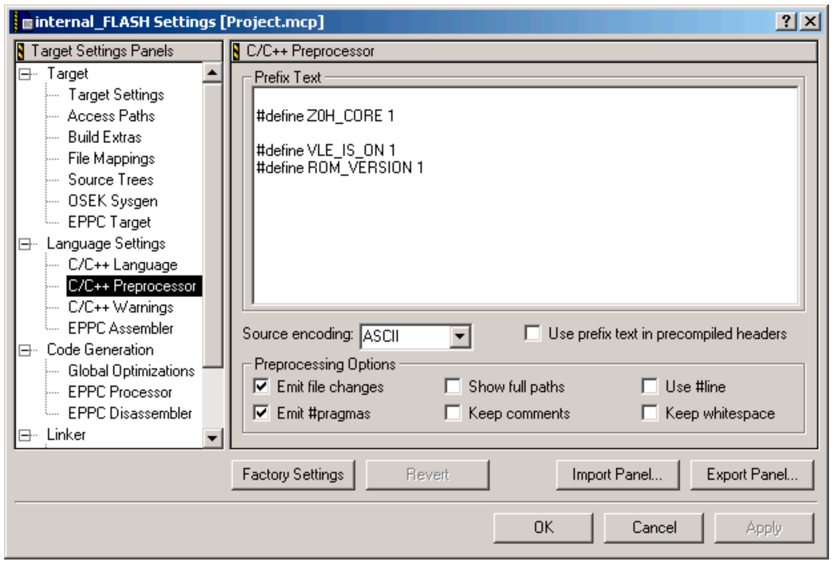
Tool	IDE Options	Command-Line Options	Notes
Comp.	Force C++ Compilation	-lang c++ | -dialect c++	When **on**, translates all C source files as C++ source code. When **off**, the IDE uses the file name's extension to determine whether to use the C or C++ compiler. This setting corresponds to the *#pragma cplusplus*
Comp.	ISO C++ Template Parser	-iso_templates on | **off**	When **on**, follows the ISO/IEC 14882-1998 standard for C++ to translate templates, enforcing more careful use of the typename and template keywords. This option corresponds to the *#pragma parse_func_templ*
Comp.	Use Instance Manager	-inst[mgr | ance_manager] on | **off**	You can control where the instance database is stored using the *#pragma instmgr_file*
Comp.	Enable C++ Exceptions	-Cpp_exceptions **on** | off	When **on**, allows you to use the try, throw, and catch statements specified in the ISO/IEC 14882-1998 C++ standard. Otherwise, set it **off** to generate smaller and faster code. This setting corresponds to the *#pragma exceptions*
Comp.	Enable RTTI	-RTTI **on** | off	When **on**, allows the use of the C++ Runtime Type Information (RTTI) capabilities, including the dynamic_cast and type_id operators. This setting corresponds to the *#pragma RTTI*
Comp.	Enable bool Support	-bool **on** | off	When **on**, the C++ compiler recognizes the bool type and its true and false values specified in the ISO/IEC 14882-1998 C++ standard. This setting corresponds to the *#pragma bool*
Comp.	Enable wchar_t Support	-wchar_t **on** | off	When **on**, the C++ compiler recognizes the wchar t data type specified in the ISO/IEC 14882-1998 C++ standard. Turn **off** this option when compiling source code that defines its own wchar_t type. This setting corresponds to the *#pragma wchar_type*
Comp	EC++ Compatibility Mode	-lang ec++ | -dialect ec++	When **on**, expects C++ source code files to contain Embedded C++ source code. This setting corresponds to the *#pragma ecplusplus*
Comp.	ANSI Strict	-strict on | **off**	Only recognizes source code that conforms to the ISO/IEC 9899-1990 standard for C. You cannot enable individual extensions that are controlled by the ANSI Strict setting. This setting corresponds to the *#pragma ANSI_strict*
Comp.	ANSI Keywords Only	-stdkeywords on | **off**	Controls whether the compiler recognizes non-standard keywords. Turn the ANSI Strict option **on** if you want to write source code that strictly adheres to the ISO standard. This setting corresponds to the *#pragma only_std_keywords*

Tool	IDE Options	Command-Line Options	Notes
Comp.	Expand Trigraphs	-trigraphs on | **off**	Many common character constants look like trigraph sequences, and this extension lets you use them without including escape characters. This setting corresponds to the *#pragma trigraphs*
Comp.	Legacy for-scoping	-for_scoping on | **off**	Generates an error message when the compiler encounters a variable scope usage that the ISO/IEC 14882-1998 C++ standard disallows. This setting corresponds to the *#pragma ARM_scoping*
Comp.	Require Function Prototypes	-r[equireprotos]	Turn **on** this option, the compiler generates an error message if you define a previously referenced function that does not have a prototype. This setting corresponds to the *#pragma require_prototypes*
Comp.	Enable C99 Extensions	-dialect | -lang c99	Recognizes ISO/IEC 9899-1999 ("C99") language features that are supported by the CodeWarrior compiler. This setting corresponds to the *#pragma c99*
Comp.	Enable GCC Extensions	-gcc[ext | _extensions] on | **off**	Lets you use language features of the GCC (Gnu Compiler Collection) C compiler that are supported by CodeWarrior. This setting corresponds to the *#pragma gcc_extensions*
Comp.	Enums Always Int	-enum min | **int**	Uses signed integers to represent enumerated constants. This option corresponds to the *#pragma enumsalwaysint*
Comp. Linker	Use Unsigned Chars	-char **signed** | unsigned	Treats char declarations as unsigned char declarations. This option corresponds to the *#pragma unsigned_char*
Comp.	Pool Strings	-str[ings] [no]pool	If you enable this setting, the compiler collects all string constants into a single data section in the object code it generates. This option corresponds to the #pragma pool_strings
Comp.	Reuse Strings	-str[ings] [no]**reuse**	When on, the compiler stores only one copy of identical string literals. When off, the compiler stores each string literal separately. This option corresponds to the *#pragma dont_reuse_strings*
Comp.	IPA	-ipa **off** | file | program	Select interprocedural analysis level. This option corresponds to the *#pragma ipa*
Comp.	Inline depth: Do not Inline	-inline none | off	Inlines no functions, not even C or C++ functions declared inline. This option corresponds to the *#pragma dont_inline*
Comp.	Inline depth: Smart	-inline **on** | smart	Turns **on** inlining for functions declared with the inline qualifier.
Comp.	Inline depth: Inline level	-inline level=1 | 2 | … | 8	Inlines to the depth specified by the numerical selection. This option corresponds to the

Tool	IDE Options	Command-Line Options	Notes
			#pragma inline_depth
Comp.	Auto-Inline	-inline auto	Lets the compiler choose which functions to inline. This option corresponds to the *#pragma auto_inline*
Comp.	Bottom-up Inlining	-inline [no]bottomup	Turn this option on, the compiler performs inline analysis from the last function to the first function in a chain of function calls. This setting corresponds to the #pragma inline_bottom_up

4.2. C/C++ Preprocessor Settings Panel

Figure 4. C/C++ Preprocessor Panel Settings



```
#define Z0H_CORE 1

#define VLE_IS_ON 1
#define ROM_VERSION 1
```

Source encoding: ASCII
Use prefix text in precompiled headers: ☐

Preprocessing Options
- ☑ Emit file changes
- ☐ Show full paths
- ☐ Use #line
- ☑ Emit #pragmas
- ☐ Keep comments
- ☐ Keep whitespace

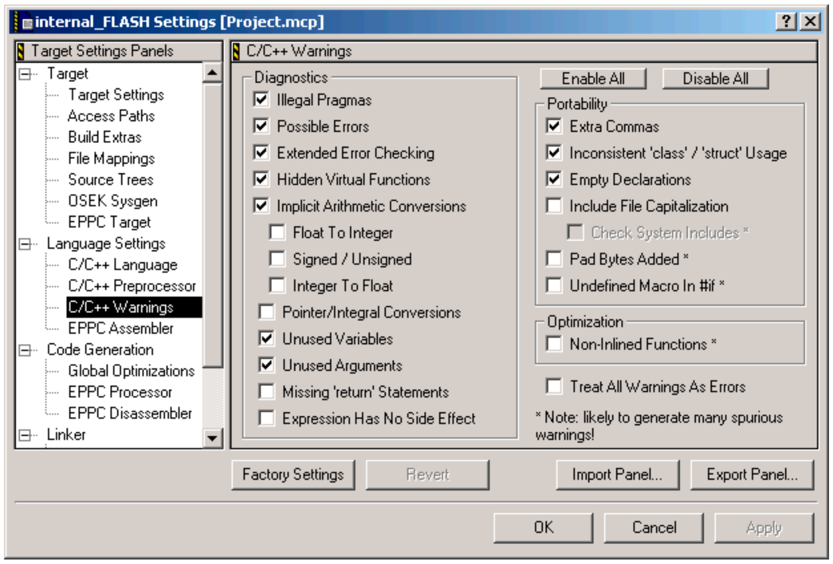
Table 4. Command-Line Options for C/C++ Preprocessor Panel

Tool	IDE Options	Command-Line Options	Notes
Comp.	Right click on a file and select preprocess	-preprocess	Pre-processes the source files.
Comp.	Source encoding	-enc[oding] **ascii** | (autodetect | multibyte | mb) | system | (UTF[8|-8]) | (SJIS | Shift-JIS | ShiftJIS) | (EUC[JP | -JP]) | (ISO[2022JP | -2022-JP])	The compiler automatically detects UTF-8 (Unicode Transformation Format) header or UCS-2/UCS-4 (Uniform Communications Standard) encodings regardless of setting.
Comp.	Use prefix text in precompiled header	-prefix file	Prefix the specified text file or precompiled header onto all source files.
Comp.	Emit file changes	-ppopt [no]break	Controls whether notification of file changes (or #line changes) appear in the output.
Comp.	Emit #pragmas	-ppopt [no]pragma	Controls whether pragmas directives encountered in the source text appear in the preprocessor output.

Tool	IDE Options	Command-Line Options	Notes
Comp.	Show full paths	-ppopt [no]full[path]	Controls whether file changes show the full path or the base filename of the file.
Comp.	Keep comments	-ppopt [no]comment	Controls whether comments are emitted in the output.
Comp.	Use #line	-ppopt [no]line	Controls whether file changes appear in comments (as before) or in #line directives.
Comp.	Keep white space	-ppopt [no]space	Controls whether whitespace is stripped out or copied into the output. This doesn't apply when macros are expanded.

4.3. C/C++ Warnings Settings Panel

Figure 5. C/C++ Warnings Settings Panel



C/C++ Warnings

Diagnostics
- [✓] Illegal Pragmas
- [✓] Possible Errors
- [✓] Extended Error Checking
- [✓] Hidden Virtual Functions
- [✓] Implicit Arithmetic Conversions
  - [ ] Float To Integer
  - [ ] Signed / Unsigned
  - [ ] Integer To Float
- [ ] Pointer/Integral Conversions
- [✓] Unused Variables
- [✓] Unused Arguments
- [ ] Missing 'return' Statements
- [ ] Expression Has No Side Effect

Portability
- [✓] Extra Commas
- [✓] Inconsistent 'class' / 'struct' Usage
- [✓] Empty Declarations
- [ ] Include File Capitalization
  - [ ] Check System Includes *
- [ ] Pad Bytes Added *
- [ ] Undefined Macro In #if *

Optimization
- [ ] Non-Inlined Functions *

- [ ] Treat All Warnings As Errors

* Note: likely to generate many spurious warnings!

Table 5. Command-Line Options for C/C++ Warnings Panel

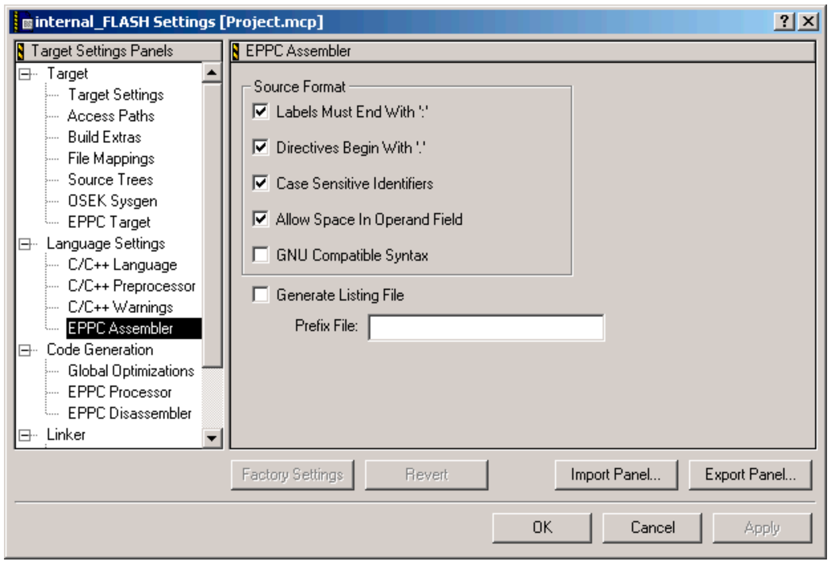
Tool	IDE Options	Command-Line Options	Notes
Comp.	Illegal Pragmas	-w[arn[ings]] [no]pragmas | [no]illpragmas	Issues a warning message if the compiler encounters an unrecognized pragma. This option corresponds to the *#pragma warn_illpragma*
Comp.	Possible Errors	-w[arn[ings]] [no]possible | [no]unwanted	Issues warning messages for common, usually-unintended logical errors. This option corresponds to the *#pragma warn_possunwant*
Comp.	Extended Error Checking	-w[arn[ings]] [no]pedantic | [no]extended	Issues warning messages for common programming errors. This option corresponds to the *#pragma extended_errorcheck*
Comp.	Hidden Virtual Functions	-w[arn[ings]] [no]hidevirtual | [no]hidden[virtual]	Issues a warning message if you declare a non-virtual member function that prevents a virtual function,that was defined in a superclass, from being called. This setting corresponds to the *#pragma warn_hidevirtual*

Tool	IDE Options	Command-Line Options	Notes
Comp.	Implicit Arithmetic Conversions	-w[arn[ings]] [no]implicit[conv]	Issues a warning message when the compiler applies implicit conversions that may not give results you intend. This option corresponds to the *#pragma warn_implicitconv*
Comp.	Float to Integer	-w[arn[ings]] [no]impl_float2int	Issues a warning message for implicit conversions from floating point values to integer values. This option corresponds to the *#pragma warn_impl_f2i_conv*
Comp.	Signed / Unsigned	-w[arn[ings]] [no]impl_signedunsigned	Issues a warning message for implicit conversions from a signed or unsigned integer value to an unsigned or signed value, respectively. This option corresponds to the *#pragma warn_impl_s2u_conv*
Comp.	Integer to Float	-w[arn[ings]] [no]impl_int2float	Issues a warning message for implicit conversions from integer to floating-point values. This option corresponds to the *#pragma warn_impl_i2f_conv*
Comp.	Pointer / Integral Conversions	-w[arn[ings]] [no]ptrintconv	Issues a warning message for implicit conversions from pointer values to integer values and vice versa. This option corresponds to the *#pragma warn_any_ptr_int_conv* *#pragma warn_ptr_int_conv*
Comp.	Unused Variables	-w[arn[ings]] [no]unusedvar	Issues a warning message for local variables that are not referred to in a function. This option corresponds to the *#pragma warn_unusedvar*
Comp.	Unused Arguments	-w[arn[ings]] [no]unusedarg	Issues a warning message for function arguments that are not referred to in a function. This option corresponds to the *#pragma warn_unusedarg*
Comp.	Missing 'return' Statements	-w[arn[ings]] [no]missingreturn	Issues a warning message if a function that is defined to return a value has no return statement. This setting corresponds to the *#pragma warn_missingreturn*
Comp.	Expression Has No Side Effect	-w[arn[ings]] [no]unusedexpr	Issues a warning message if a statement does not change the program's state. This option corresponds to the *#pragma warn_no_side_effect*
Comp.	Extra Commas	-w[arn[ings]] [no]extracomma | [no]comma	Issues a warning message if a list in an enumeration terminates with a comma. This option corresponds to the #pragma warn_extracomma
Comp.	Inconsistent 'class' / 'struct' Usage	-w[arn[ings]] [no]structclass	Issues a warning message if the class and struct keywords are used interchangeably in the definition and declaration of the same identifier in C++ source code. This option corresponds to the *#pragma warn_structclass*
Comp.	Empty Declarations	-w[arn[ings]] [no]empty[decl]	Issues a warning message if a declaration has no variable name. This setting corresponds to the *#pragma warn_emptydecl*

Tool	IDE Options	Command-Line Options	Notes
Comp.	Include File Capitalization	-w[arn[ings]] [no]filecaps	Issues a warning message if the name of the file specified in a #include "file" directive uses different letter case from a file on disk. This option corresponds to the *#pragma warn_filenamecaps*
Comp.	Check System Includes	-w[arn[ings]] [no]sysfilecaps	Issues a warning message if the name of the file specified in a #include <file> directive uses different letter case from a file on disk. This option corresponds to the *#pragma warn_filenamecaps_system*
Comp.	Pad Bytes Added *	-w[arn[ings]] [no]padding	Issues a warning message when the compiler adjusts the alignment of components in a data structure. This option corresponds to the *#pragma warn_padding*
Comp.	Undefined Macro In #if *	-w[arn[ings]] [no]undef[macro]	Issues a warning message if an undefined macro appears in #if and #elif directives. This option corresponds to the *#pragma warn_undefmacro*
Comp.	None-Inlined Functions *	-w[arn[ings]] [no]notinlined	Issues a warning message if a call to a function defined with the inline, __inline__, or __inline keywords could not be replaced with the function body. This option corresponds to the *#pragma warn_notinlined*
Comp.	Treat All Warnings As Errors	-w[arn[ings]] [no]err[or] | [no]iserr[or]	Issues warning messages as error messages. This option corresponds to the *#pragma warning_errors*

4.4. C/C++ Assembler Settings Panel

Figure 6. C/C++ Assembler Settings Panel


internal_FLASH Settings [Project.mcp]

Target Settings Panels
- Target
  - Target Settings
  - Access Paths
  - Build Extras
  - File Mappings
  - Source Trees
  - OSEK Sysgen
  - EPPC Target
- Language Settings
  - C/C++ Language
  - C/C++ Preprocessor
  - C/C++ Warnings
  - EPPC Assembler
- Code Generation
  - Global Optimizations
  - EPPC Processor
  - EPPC Disassembler
- Linker

EPPC Assembler

Source Format
- [✓] Labels Must End With ':'
- [✓] Directives Begin With '.'
- [✓] Case Sensitive Identifiers
- [✓] Allow Space In Operand Field
- [ ] GNU Compatible Syntax

- [ ] Generate Listing File

Prefix File:

Table 6. Command-Line Options for C/C++ Assembler Settings Panel

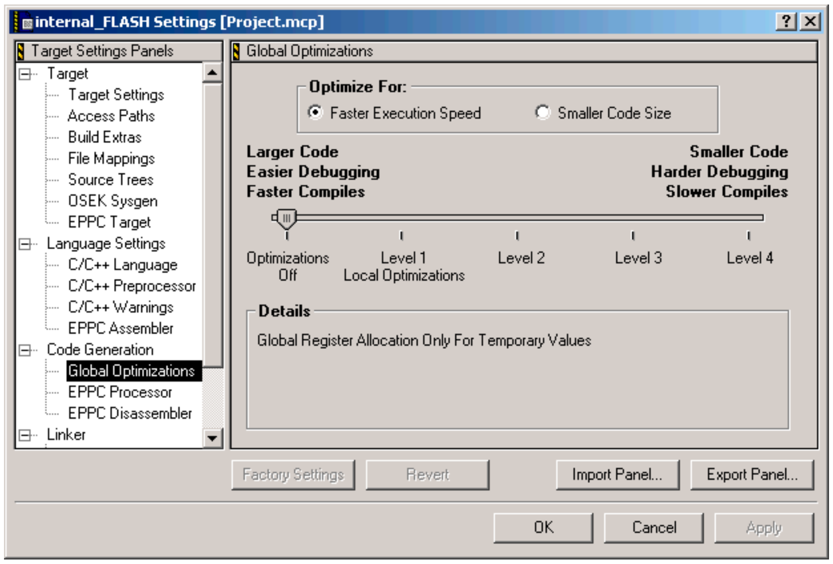
Tool	IDE Options	Command-Line Options	Notes
Asm	Labels Must End With ':'	-[no]**colons**	If enabled, assembler labels must end with a ":".

Tool	IDE Options	Command-Line Options	Notes
Asm	Directives Begin With '.'	-[no]**period**	If enabled, assembler labels must end with ":".
Asm	Case Sensitive Identifiers	-[no]**case**	If enabled, assembler identifiers are case sensitive.
Asm	Allow Space In Operand Field	-[no]**space**	If enabled, assembler allows space in operand field.
Asm	GNU Compatible Syntax	-[no]gccdep[ends]	If set, write dependency file with name and location based on output file.
Asm	Generate Listing File	-list	A listing file contains file source along with line numbers, relocation information, and macro expansions.
Asm	Prefix File	-prefix file	The **Prefix File** text box specifies a prefix file that is automatically included in all assembly files in the project.

5. Code Generation Settings

5.1. Global Optimization Settings Panel

Figure 7. Global Optimization Settings Panel


internal_FLASH Settings [Project.mcp]

Global Optimizations

Optimize For: [✓ Faster Execution Speed] Smaller Code Size

Larger Code / Easier Debugging / Faster Compiles — Smaller Code / Harder Debugging / Slower Compiles

Slider: [✓ Optimizations Off] Level 1 Local Optimizations, Level 2, Level 3, Level 4

Details: Global Register Allocation Only For Temporary Values

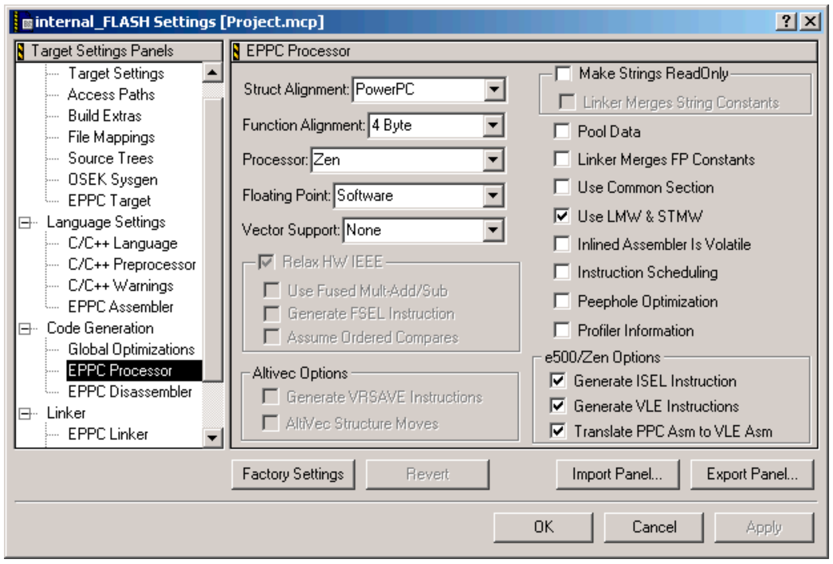
Table 7. Command-Line Options for Global Optimization Settings

Tool	IDE Options	Command-Line Options	Notes
Comp.	Faster Execution Speed	-opt speed or -Op	Optimize object code for speed (but code size is usually large). This option corresponds to the *#pragma optimize_for_size on*
Comp.	Smaller Code Size	-opt space or -Os	This option is selected when you want to optimize size (but speed slowly). This option corresponds to the *#pragma optimize_for_size off*
Comp.	Optimization off	-opt **off** or -O0	Global Register Allocation Only For Temporary Values.

Tool	IDE Options	Command-Line Options	Notes
			This option corresponds to the *#pragma optimization_level 0*
Comp.	Level 1	-opt l=1 or -O1	O0 + Dead Code Elimination + Branch and Arithmetic Optimizations + Peephole Optimization. This option corresponds to the *#pragma optimization_level 1*
Comp.	Level 2	-opt l=2 or -O2	O1 + Common Subexpression Elimination + Copy and Expression Propagation + Stack Frame Compression + Stack Alignment + Fast Floating-Point to Integer Conversions. This option corresponds to the *#pragma optimization_level 2*
Comp.	Level 3	-opt l=3 or -O3	O2 + Dead Store Elimination + Live Range Splitting + Loop-Invariant Code Motion + Loop Transformations + Loop Unrolling + Strength Reduction + Loop Vectorization + Lifetime-Based Register Allocation + Instruction Scheduling. This option corresponds to the *#pragma optimization_level 3*
Comp.	Level 4	-opt l=4 or -O4	O3 + more comprehensive optimizations from levels 1 and 2. This option corresponds to the *#pragma optimization_level 4*

5.2. EPPC Processor Settings Panel

Figure 8. EPPC Processor Settings Panel



internal_FLASH Settings [Project.mcp]

EPPC Processor

Struct Alignment: PowerPC
Function Alignment: 4 Byte
Processor: Zen
Floating Point: Software
Vector Support: None

[✓] Relax HW IEEE
[ ] Use Fused Mult-Add/Sub
[ ] Generate FSEL Instruction
[ ] Assume Ordered Compares

Altivec Options
[ ] Generate VRSAVE Instructions
[ ] AltiVec Structure Moves

[ ] Make Strings ReadOnly
[ ] Linker Merges String Constants
[ ] Pool Data
[ ] Linker Merges FP Constants
[ ] Use Common Section
[✓] Use LMW & STMW
[ ] Inlined Assembler Is Volatile
[ ] Instruction Scheduling
[ ] Peephole Optimization
[ ] Profiler Information

e500/Zen Options
[✓] Generate ISEL Instruction
[✓] Generate VLE Instructions
[✓] Translate PPC Asm to VLE Asm

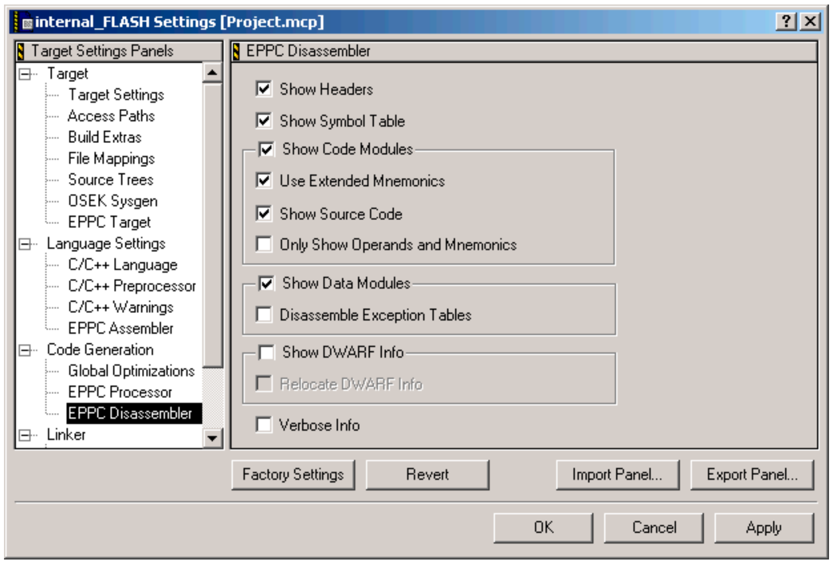
Table 8. Command-Line Options for EPPC Processor Settings

Tool	IDE Options	Command-Line Options	Notes
Comp.	Struct Alignment	-align **power[pc]** | 68K …	The value for this option should always be **PowerPC** to conform to the PowerPC EABI and interoperate with third-party object code. Other settings may lead to reduced performance or alignment violation exceptions.
Comp.	Function Alignment	-func_align **4**..128	If the core is capable of fetching multiple instructions at a time, you may achieve slightly better performance by aligning functions to the width of the fetch. This option corresponds to the *#pragma function_align*
Asm. Comp. Linker	Processor	-proc Zen | generic…	Use the Processor list box to specify the target processor. For MPC55xx/MPC56xx the Processor value should be **Zen**.
Comp. Linker	Floating Point	-fp none none|off soft[ware] hard[ware] efpu|spfp dpfp spfp_only	This selection specifies how the compiler handles floating-point operations in your code. The Runtime and MSL libraries must match with the option used.
Comp.	Vector Support	-vector on|off… -spe_vector -spe_addl_vector -spe2_vector	If you want to allow vector instructions for your processor, select a vector type that your processor supports. Currently, only the Altivec and SPE vector units are supported.
Comp.	Make Strings ReadOnly	-rostr | -readonlystrings	Make string constants read-only. This option corresponds to the #pragma readonly_strings
Comp	Linker Merges String Constants	-str nopool	This option instructs the compiler not to pool the identical string constants, so that linker will do the work to deadstrip the unreferenced string.
Comp.	Pool Data	-pool[data] on | off	Instruct the compiler to organize some of the data in the large data sections of .data , .bss, and .rodata so that the program can access it more quickly.
Comp.	Linker Merges FP Constants	-flag merge_float_consts -str nopool	This allows the linker to merge the floating-point constants automatically. This option corresponds to the #pragma fp_constants_merge
Comp.	Use Common Section	-common on | off	If on the compiler places global uninitialized data in the common section.
Comp.	Use LMW & STMW	-use_lmw_stmw on | off	LMW / STMW (Load / Store Multiple Word) is a single PowerPC instruction that loads/stores a group of registers; The compiler sometimes uses these instructions in a function's prologue and epilogue to save and restore volatile registers. This option corresponds to the #pragma use_lmw_stmw
Comp.	Inlined Assembler Is Volatile	-[no]volatileasm	Check to have the compiler treat all asm blocks (including inline asm blocks) as if the volatile keyword was present. This prevents the asm block from being optimized. This option corresponds to the

Tool	IDE Options	Command-Line Options	Notes
			#pragma volatileasm
Comp.	Instruction Scheduling	-schedule on | off	If **on** (checked), scheduling of instructions is optimized for the specific processor you are targeting (determined by which processor is selected in the Processor list box) Enabling the Instruction Scheduling can make source-level debugging more difficult. This option corresponds to the #pragma schedule
Comp.	Peephole Optimization	-opt [no]peep[hole]	Enables peephole optimization.This option corresponds to the #pragma peephole
Comp.	Profiler Information	-profile on | off	Generates calls to a function entry and exit for use with a profiler. This option corresponds to the #pragma profile
Comp.	Relax HW IEEE	-[no]relax_ieee	The **Relax HW IEEE** checkbox is available only if you select Hardware from the **Floating Point** list box.
Comp.	Use Fused Multi-Add/Sub	-fp_contract | -maf on | off	Turn this option **on** to generate PowerPC Fused Multi-Add/Sub instructions, which result in smaller and faster floating-point code.
Comp.	Generate FSEL Instruction	-use_fsel on | off	Turn this option **on** to generate the faster executing FSEL instruction. The FSEL option allows the compiler to optimize the pattern x = (condition ? y : z), where x and y are floating-point values.
Comp.	Assume Ordered Compares	-[no-]ordered-fp-compares	Enables the compiler to ignore issues with unordered numbers, such as NAN, while comparing floating point values.
Comp.	Generate VRSAVE Instructions	-vector [no]vrsave	The VRSAVE register indicates to the operating system, which vector registers to save and reload when a context switch happens.
Comp.	Altivec Structure Moves	-[no]altivec_move_block	Controls the use of Altivec instructions to optimize block moves.
Comp.	Generate ISEL Instruction	-use_isel on | off	Do not turn on this option if the Power Architecture processor of your target platform does not implement the Freescale ISEL APU.
Asm. Comp. Linker	Generate VLE Instructions	-vle	Prompts the assembler, compiler and linker to generate and lay out Variable Length Encoded (VLE). This option sets the -processor Zen option
Comp.	Translate PPC Asm to VLE Asm	-ppc_asm_to_vle	Sets the -processor Zen and enables VLE code generation in C/C++ files.

5.3. EPPC Disassembler Settings Panel

Figure 9. EPPC Disassembler Settings Panel



internal_FLASH Settings [Project.mcp]

Target Settings Panels: Target (Target Settings, Access Paths, Build Extras, File Mappings, Source Trees, OSEK Sysgen, EPPC Target); Language Settings (C/C++ Language, C/C++ Preprocessor, C/C++ Warnings, EPPC Assembler); Code Generation (Global Optimizations, EPPC Processor, EPPC Disassembler); Linker

EPPC Disassembler

- [✓] Show Headers
- [✓] Show Symbol Table
- [✓] Show Code Modules
  - [✓] Use Extended Mnemonics
  - [✓] Show Source Code
  - [ ] Only Show Operands and Mnemonics
- [✓] Show Data Modules
  - [ ] Disassemble Exception Tables
- [ ] Show DWARF Info
  - [ ] Relocate DWARF Info
- [ ] Verbose Info

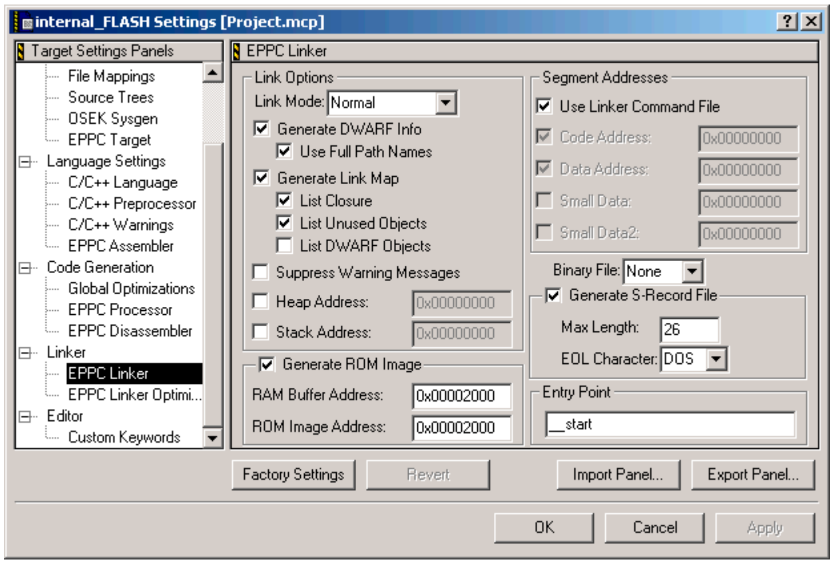
Table 9. Command-Line Options for EPPC Disassembler Settings

Tool	IDE Options	Command-Line Options	Notes
Linker	Show Headers	-show [no]**headers**	Disassembled file lists any ELF header information in the disassembled output.
Linker	Show Symbol Table	-show [no]**tables**	Disassembler lists the symbol table for the disassembled module.
Linker	Show Code Modules	-show [no]**code** | -show [no]**text**	Disassembler provides ELF code sections in the disassembled output for a module.
Linker	Use Extended Mnemonics	-show [no]**extended**	Disassembler lists the extended mnemonics for each instruction for the disassembled module
Linker	Show Source Code	-show [no]source	Interleave the code disassembly with c/c++ source
Linker	Only Show Operands and Mnemonics	-show only|none	Disassembler lists the offset for any functions in the disassembled module.
Linker	Show Data Modules	-show [no]**data**	Disassembler provides ELF data sections (such as .rodata and .bss) in the disassembled output for a module.
Linker	Disassemble Exception Tables	-show [no]xtables	Disassembler shows exception tables.
Linker	Show DWARF Info	-show [no]debug | -show [no]dwarf	Disassembler includes DWARF symbol information in the disassembled output.
Linker	Relocate DWARF Info	-[no]relocate	This option lets you relocate object and function addresses in the DWARF information (rela.text and .rela.debug)
Linker	Verbose Info	-v[erbose]	Disassembler shows additional information about certain types of information in the ELF file.

6. Linker Settings

6.1. EPPC Linker Settings Panel

Figure 10. EPPC Linker Settings Panel



internal_FLASH Settings [Project.mcp]

Target Settings Panels
- File Mappings
- Source Trees
- OSEK Sysgen
- EPPC Target
- Language Settings
  - C/C++ Language
  - C/C++ Preprocessor
  - C/C++ Warnings
  - EPPC Assembler
- Code Generation
  - Global Optimizations
  - EPPC Processor
  - EPPC Disassembler
- Linker
  - EPPC Linker
  - EPPC Linker Optimi...
- Editor
  - Custom Keywords

EPPC Linker

Link Options
- Link Mode: Normal
- [✓] Generate DWARF Info
  - [✓] Use Full Path Names
- [✓] Generate Link Map
  - [✓] List Closure
  - [✓] List Unused Objects
  - [ ] List DWARF Objects
- [ ] Suppress Warning Messages
- [ ] Heap Address: 0x00000000
- [ ] Stack Address: 0x00000000

[✓] Generate ROM Image
- RAM Buffer Address: 0x00002000
- ROM Image Address: 0x00002000

Segment Addresses
- [✓] Use Linker Command File
- [✓] Code Address: 0x00000000
- [✓] Data Address: 0x00000000
- [ ] Small Data: 0x00000000
- [ ] Small Data2: 0x00000000

Binary File: None

[✓] Generate S-Record File
- Max Length: 26
- EOL Character: DOS

Entry Point
- __start

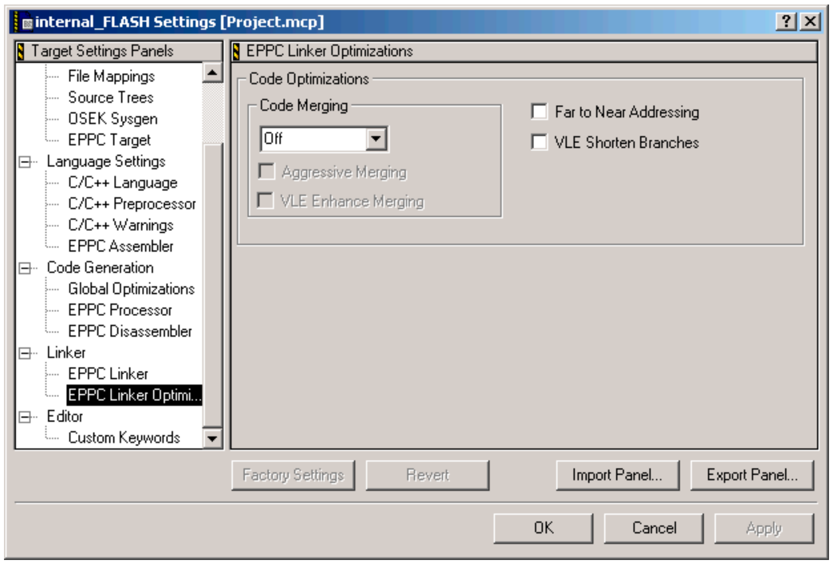
Table 10. Command-Line Options for EPPC Linker Settings

Tool	IDE Options	Command-Line Options	Notes
Linker	Link Mode	-linkmode lessram | **normal** | moreram	Linking requires enough RAM space to hold all of the input files and the numerous structures that the linker uses for housekeeping. The housekeeping allocations occur before the linker writes the output file to the disk.
Linker	Generate DWARF Info	-sym **off** | on	Instructs the linker to generate debugging information. The debugger information is included within the linked ELF file.
Linker	Generate DWARF Info: Use Full Path Names	-sym full[path]	To avoid problems while having the debugger locate your source code, clear the **Use Full Path Names** checkbox when building and debugging on different machines or platforms.
Linker	Generate Link Map	-map [filename]	Generates link map file.
Linker	Generate Link Map: List Closure	-[**no**]listclosure	If enabled all the functions invoked by the starting point of the program are listed in the link map.
Linker	Generate Link Map: List Unused Objects	-[**no**]unused | -[no]mapunused	If enabled the linker includes unused objects in the link map. This setting is useful in cases where you may discover that an object you expect to be used is not in use.
Linker	Generate Link Map: List DWARF Objects	-[**no**]listdwarf	If enabled list dwarf objects in map file.
Linker	Suppress Warning Messages	-w[arn[ings]] **off** | on	If on then linker do not to display warnings in the CodeWarrior message window.
Linker	Heap Address	-heapaddr addr_value	Specifies the location in memory where the program heap resides. The heap is used if your program calls malloc or new. If you do not call malloc or new, consider setting Heap Size (k) to 0 to maximize the memory available for code, data, and the stack.

Tool	IDE Options	Command-Line Options	Notes
Linker	Stack Address	-stackaddr addr_value	Specifies the location in memory where the program stack resides. Since the stack grows downward, it is common to place the stack as high as possible.
Linker	Use Linker Command File	-lcf filename	Select/enable to have the segment addresses specified in a linker command file.
Linker	Code Address	-codeaddr addr_value	Specifies the location in memory where the executable code resides.
Linker	Data Address	-dataaddr addr_value	Specifies the location in memory where the global data of the program resides. Data must reside in RAM. If not checked/disabled, the linker calculates the data address to begin immediately following the read-only code and data (.text, .rodata, extab and extabindex).
Linker	Small Data	-sdataaddr addr_value	Specifies the location in memory where the small data section resides. All types of data must reside in RAM. If not checked/disabled, the linker calculates the small data address to begin immediately following the .data section.
Linker	Small Data2	-sdata2addr addr_value	Specifies the location in memory where the small data2 section resides. If not checked/disabled, the linker calculates the small data2 address to begin immediately following the .sbss section.
Linker	Generate ROM Image: RAM Buffer Address	-rambuffer addr_value	The CodeWarrior flash programmer does not use a separate RAM buffer. As a result, if you use the CodeWarrior Flash Programmer (or any other flash programmer that does not use a RAM buffer), the RAM Buffer Address must be equal to the ROM Image Address.
Linker	Generate ROM Image: ROM Image Address	-romaddr addr_value	Set address for ROM image.
Linker	Generate S-Record File	-srec [filename]	Generate an S-Record file based on the application object image. This file has the same name as the executable file, but with a .mot extension. The linker generates S3 type S-Records.
Linker	Sort S-Record	-sortsrec	Sort S-record in ascending address order
Linker	Max Length	-sreclength max_length	The default value is **26.** Specifies length of S-record line.
Linker	EOL Character	-sreceol mac | **dos** | unix	Sets end-of-line separator for S-record file.
Linker	Entry Point	-m[ain] symbol	This is the starting point of the program The default function is bootstrap or glue code that sets up the PowerPC EABI environment before your code executes. This function is in the *__start.c* file. The default value is **__start**.

6.2. EPPC Linker Options Settings Panel

Figure 11. EPPC Linker Options Settings Panel



internal_FLASH Settings [Project.mcp]

Target Settings Panels: File Mappings, Source Trees, OSEK Sysgen, EPPC Target, Language Settings, C/C++ Language, C/C++ Preprocessor, C/C++ Warnings, EPPC Assembler, Code Generation, Global Optimizations, EPPC Processor, EPPC Disassembler, Linker, EPPC Linker, EPPC Linker Optimi..., Editor, Custom Keywords

EPPC Linker Optimizations

Code Optimizations

Code Merging: Off

- [ ] Aggressive Merging
- [ ] VLE Enhance Merging
- [ ] Far to Near Addressing
- [ ] VLE Shorten Branches

Table 11. Command-Line Options for EPPC Linker Options Settings

Tools	IDE Option	Command-Line Option	Notes
Linker	Code Merging	-code_merging **off** | all | safe	Select code merging optimization.
Linker	Aggressive Merging	-code_mering aggressive	Don't check if the functions addresses are used.
Linker	VLE Enhance Merging	-[no]vle_enhance_merging	VLE enhance code merging optimization.
Linker	Far to Near Addressing	-[no]nofar_near_addressing	Far to near addressing optimization.
Linker	VLE Shorten Branches	-[no]vle_bl_opt	VLE shorten BL optimization.

7. MPC5674F Makefile Example

Listing 1. Sample Makefile for MPC5674F

```
#=========================================================================
# Makefile: Sample makefile for MPC5674F to work with Freescale CodeWarrior v2.x
#=========================================================================
#
# Note: This Makefile uses GNU make features and was tested using Cygwin
#       tools.  Visit http://www.cygwin.com to obtain these tools.
#
#=========================================================================
#!!!! Update this to the root of your CodeWarrior installation !!!!
CWROOT=c:/CW_MPC55xx

CW=$(CWROOT)/PowerPC_EABI_Tools/Command_Line_Tools
LIBROOT=$(CWROOT)/PowerPC_EABI_Support

MSLDIR = $(LIBROOT)/MSL/MSL_C/PPC_EABI/Lib
RTMDIR = $(LIBROOT)/Runtime/Lib
```

```
# MSL library selected
MSLLIB   = MSL_C.PPCEABI.bare.V.UC.a

# Runtime library selected (should match with MSL lib)
RTMLIB   = Runtime.PPCEABI.V.UC.a

CC = $(CW)/mwcceppc
AS = $(CW)/mwasmeppc
LD = $(CW)/mwldeppc

# Defautl to Flash target...
ifndef TARGET
TARGET=Flash
endif

# Note that we will build the RAM and FLASH versions with debug information
INCLUDES = -I- -ir $(CWROOT)/PowerPC_EABI_Support

# common linker and compiler options
CLOPTS    = -proc Zen -char unsigned -fp SPFP -gdwarf-2 -sdata 71 -sdata2 127

# Compiler specific options
COPTS     = $(CLOPTS) -c -DVLE_IS_ON=1 -vle -ppc_asm_to_vle  -O4 -opt speed -pragma "schedule
750" -spe_vector -spe_addl_vector -RTTI off -wchar_t off -cpp_exceptions off -align powerpc -
use_lmw_stmw on -use_isel on -cwd include $(INCLUDES) -msgstyle GCC -ppopt BREAK, pragma -w
unwanted

# assembler specific options
AOPTS     = -c -gdwarf-2 -proc Zen $(INCLUDES)

# linker specific options
LOPTS     = $(CLOPTS) -lr $(MSLDIR) -lr $(RTMDIR) -srec -map -code_merging all,aggressive -
far_near_addressing -vle_enhance_merging -vle_bl_opt

EXECUTABLE-RAM   = test_RAM
EXECUTABLE-FLASH     = test_Flash

# The output will be placed in...
O=bin

# Common objects...
OBJS = $(O)/main.o $(O)/MPC5674F_HWInit.o $(O)/IntcInterrupts.o $(O)/Exceptions.o
$(O)/__ppc_eabi_init.o

# Options for the Flash target...
ifeq ($(TARGET),Flash)

LOPTS += -romaddr 0x00020000 -rambuffer 0x00020000
COPTS += -DROM_VERSION=1
TARG = $(O)/$(EXECUTABLE-FLASH).elf
LCF  = LCF/MPC5674F.lcf
OBJS += $(O)/MPC55xx_init.o
EXTRA = @echo "Use make clean; make TARGET=RAM to produce the RAM version"
```

```
# Options for the RAM target...
else

TARG = $(O)/$(EXECUTABLE-RAM).elf
LCF  = LCF/MPC5674F_DEBUG.lcf
OBJS += $(O)/MPC55xx_init_debug.o
EXTRA = @echo "Use make clean; make TARGET=Flash to produce the Flash version"

endif

.SUFFIXES: .c .s

$(O)/%.o : Sources/%.c
	$(CC) $(COPTS) -o $@ $<

$(O)/%.o : Sources/%.s
	$(AS) $(AOPTS) -o $@ $<

default: bindir $(TARG)

$(O)/$(EXECUTABLE-RAM).elf: $(OBJS)
	$(LD) -lcf $(LCF) $(LOPTS) -l$(RTMLIB) -l$(MSLLIB) $^ -o $@
	@echo ""
	$(EXTRA)

$(O)/$(EXECUTABLE-FLASH).elf: $(OBJS)
	$(LD) -lcf $(LCF) $(LOPTS) -l$(RTMLIB) -l$(MSLLIB) $^ -o $@
	@echo ""
	$(EXTRA)

# Create the directory if it doesn't exist
.PHONY: bindir
bindir:
	@test -d $(O) || mkdir -p $(O)

clean:
	@rm -rf $(O)
```

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064, Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-521-6274 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: AN4094

13 April 2010