

# CodeWarrior Development Studio for Microcontrollers V10.x FAQ Guide

Document Number: CWMCUFAQ  
Rev 10.6, 02/2014



## Contents

Section number	Title	Page
<b>Chapter 1</b>		
<b>Introduction</b>		
1.1	Contents of this Manual.....	19
1.2	What's New for Microcontrollers 10.5.....	20
1.3	Accompanying Documentation.....	21
<b>Chapter 2</b>		
<b>IDE and Installation</b>		
2.1	Updating and Installing Software.....	23
2.1.1	I have installed CodeWarrior for Microcontrollers on Linux Ubuntu 8.04, 64-bit. However, Multilink drivers are not working. What do I do?.....	24
2.1.2	I want to install updates for the CodeWarrior software. However, when I check for updates, no updates are found. What do I do?.....	24
2.1.3	I am facing issues in accessing the updater sites. How do I resolve this?.....	26
2.1.4	My attempt to install the CodeWarrior tool fails with the error "Another version of this product is already installed". What do I do?.....	27
2.1.5	How do I uninstall the third party plugins?.....	27
2.1.6	After updating the CodeWarrior, I receive the following error while restarting the application, "Initialize RSE' has encountered a problem". What should I do?.....	29
2.1.7	How do I know what updates/patches are already installed?.....	30
2.1.8	I have installed the CodeWarrior patches on Windows 7. The patches have installed correctly, however the CodeWarrior software generates error while building the libraries. Is there any way to resolve this?.....	30
2.1.9	How can I check if the compiler updater is successfully installed?.....	31
2.2	Editor.....	33
2.2.1	How can I quickly open declaration of any variable, function name, macro, or header file from within Editor?.....	34
2.2.2	How can I open definition of a function or header file from within the editor?.....	35
2.2.3	If I am using a Macintosh computer, what is the alternate to using the F3 key in Editor to open declarations?.....	37
2.2.4	How can I change the color that highlights the current line in a source code file?.....	38
2.2.5	Is it possible to view definition of a macro or a variable in the source code file?.....	39

Section number	Title	Page
2.2.6	Can the CodeWarrior IDE assist me in writing the source code? .....	41
2.2.7	How can I change the tab width/size?.....	43
2.2.8	Is it possible to display line numbers in Editor?.....	44
2.2.9	Is it possible to change the default code format settings?.....	45
2.2.10	How can I quickly switch to a header file from within the source code file?.....	46
2.2.11	How does the CodeWarrior IDE differentiate between enabled and disabled macros?.....	46
2.2.12	How can I configure predefined macros in the CodeWarrior IDE?.....	47
2.2.13	Is it possible to view evaluated expansion of a macro in the CodeWarrior IDE?.....	47
2.2.14	Is it possible to roll back the changes I did to my source code?.....	48
2.2.15	Is there a way to change the default color of comments that span across multiple lines?.....	49
2.2.16	Is there a way to revert to the original contents of a source code file?.....	50
2.2.17	How do I convert Line Delimiters to the Unix format?.....	51
2.2.18	Why do I get the following error message when I try to edit a source code file?.....	51
2.2.19	What does an asterisk on the Editor title bar mean?.....	52
2.2.20	How can I find out where a function is declared in my source code?.....	52
2.2.21	How can I open .tcl extension files in the Editor window?.....	52
2.2.22	Is it possible to condense a large source file by folding/unfolding different portions of code?.....	55
2.2.23	How can I generate S19, Intel Hex, and Binary files for my project?.....	56
2.2.24	How can I use the Burner for the Kinetis derivatives?.....	61
2.2.25	I have created two build configurations one for debug and another for release. How can I specify a burner bbl file for each build configuration?.....	63
2.2.26	Is there a way to determine in which workspace I am working currently?.....	64
2.2.27	How can I share same installation of the CodeWarrior software with multiple users?.....	64
2.2.28	How can I comment/uncomment a block of code in the editor?.....	65
2.2.29	How can I compare files in CodeWarrior IDE?.....	69
2.2.30	Is there a way to compare files that are not in project?.....	71
2.2.31	How do I switch between the source files in the Editor view?.....	72
2.2.32	I want to edit my source files, but all the other views are using the space as well. How do I maximize the editor view to have better working area?.....	72

Section number	Title	Page
2.2.33	How do I remove blank lines in the Editor view?.....	73
2.3	Workbench Window.....	73
2.3.1	What controls the initial layout of a view in a Workbench window?.....	74
2.3.2	Is it possible to restrict the number of resources that appear on my Workbench window?.....	74
2.3.3	How do I minimize a Workbench window?.....	76
2.3.4	How do I customize my Workspace perspectives?.....	76
2.3.5	How do I enable full screen mode in my CodeWarrior IDE?.....	78
2.3.6	How do I hide the toolbar from the IDE for better visibility of other views that I am currently working with?.....	79
2.3.7	How can I view all keyboard shortcuts available in the CodeWarrior IDE?.....	80
2.3.8	How do I edit the keyboard shortcuts in the CodeWarrior IDE?.....	81
2.4	CVS.....	82
2.4.1	Does the CodeWarrior IDE support version control systems?.....	82
2.4.2	How can I add my project to CVS from within the CodeWarrior IDE?.....	85
2.5	Search and Replace Action.....	90
2.5.1	How do I perform a simple search in the CodeWarrior IDE?.....	91
2.5.2	Is it possible to preview the changes of the Replace action?.....	91
2.5.3	Is there a way to change the variable or the function name in a CodeWarrior project?.....	93
2.5.4	How can I search for specific files in a workspace?.....	95
2.5.5	Is there a way to replace a variable easily?.....	97
2.5.6	How can I go back to the previous location if the Editor takes me to another source or header file when, for example, I select the Open Declaration option?.....	99
2.5.7	Is there a shortcut to open the search window for a specified text that is equivalent to the Ctrl + Shift + M shortcut key available in the Classic CodeWarrior IDE?.....	99
2.5.8	Which set of files I need to backup or store in a version control system to completely recover project settings and related debug settings? .....	100
2.5.9	I want to perform a Regular expressions search, but I do not remember the syntax of the regular expressions. What do I do?.....	101
2.5.10	How do I merge the multiple empty lines into a single one in the Editors view?.....	102
2.6	Miscellaneous.....	102
2.6.1	How can I improve the performance of the CodeWarrior IDE I am working with?.....	105

Section number	Title	Page
2.6.2	What is the first thing that I see when I start the CodeWarrior IDE?.....	106
2.6.3	What is a perspective?.....	107
2.6.4	How can I find the version of the CodeWarrior that I am using?.....	107
2.6.5	Is it possible to retrieve a file that I deleted accidentally from my CodeWarrior project?.....	107
2.6.6	I am using my workspace since long time and adding/removing projects frequently, this has slowed down my CodeWarrior software. How can I improve the performance?.....	108
2.6.7	How do I update my local history settings?.....	109
2.6.8	Is there a way to filter the settings in the Preferences window to find a particular setting?.....	111
2.6.9	How can I change a key binding?.....	112
2.6.10	Is it possible to view the change history of a source code file?.....	114
2.6.11	How can I view a graphical representation of source code in the CodeWarrior IDE using Dot and Doxygen?.....	116
2.6.12	How can I determine which header files are associated with my source code file?.....	117
2.6.13	How do I view call hierarchy in my source code?.....	118
2.6.14	Why the project that I just created is not visible in the CodeWarrior Projects view?.....	119
2.6.15	I created a new file in Windows Explorer but the file does not appear in the CodeWarrior Projects view. Why?.....	119
2.6.16	Why does the Rename option appear grayed out?.....	120
2.6.17	Why duplicating a configuration in the debugger perspective does not duplicate the run configuration?.....	120
2.6.18	Can I still use Ctrl+Tab keys to navigate between open windows?.....	120
2.6.19	How can I change the debugger key bindings to the ones that I used to have in the Classic CodeWarrior IDE?.....	120
2.6.20	How can I change the name of the executable that is generated when I build my project?.....	122
2.6.21	How can I see printf output in the console window?.....	123
2.6.22	Why the console view does not display all the warning and error messages by default?.....	123
2.6.23	Is there a file that contains all the launch configuration settings?.....	124
2.6.24	How the CodeWarrior Project Importer handles recursive access paths?.....	124
2.6.25	Can I disassemble my source code file?.....	125
2.6.26	Where exactly the disassemble file gets created?.....	125
2.6.27	Why the .metadata folder in my workspace stores a huge history?.....	125
2.6.28	How to have projects from different workspaces or locations in a workspace?.....	126

Section number	Title	Page
2.6.29	How can I switch to another workspace?.....	126
2.6.30	My workspace takes long to close. How can I resolve this?.....	126
2.6.31	What is the purpose of the Tasks view?.....	128
2.6.32	What is the purpose of the Properties view?.....	129
2.6.33	What is the purpose of the Outline view?.....	129
2.6.34	How do I use the Outline View to get an overview of a source file and modify it directly?.....	129
2.6.35	How can I find out if certain files contain debug information?.....	130
2.6.36	How can I resolve the following error message that I get when I start the CodeWarrior IDE?.....	131
2.6.37	How can I open an existing project in the CodeWarrior IDE?.....	131
2.6.38	What is the Manage Configurations button in the C/C++ perspective toolbar used for? Could it be replaced by the Properties button?.....	132
2.6.39	How can I start the post-build steps in the CodeWarrior IDE?.....	132
2.6.40	Is it necessary to have the project name identical to the name of the directory that contains the .project file? .	132
2.6.41	Where does the CodeWarrior IDE save the debug configuration as a local file by default?.....	132
2.6.42	When I save my launch configurations as a local file and then delete the project, all the local configurations get deleted as well. How can I resolve this issue?.....	133
2.6.43	How can I modify and save the files in GBK encoding using the CodeWarrior IDE?.....	134
2.6.44	How are the PARENT-COUNT-MyVariable definitions defined in the CodeWarrior IDE?.....	135
2.6.45	Is there a way to instruct the CodeWarrior IDE to use relative paths instead of absolute ones to store the project file location in a workspace?.....	135
2.6.46	How do I solve the probable mismatch of the OSJTAG Firmware with P&E Virtual Serial Port Utility?.....	135
2.6.47	I am using CW10.1SE and MQX3.7 with all patches and updates. How can I solve this problem?.....	136
2.6.48	What is a Working Set? How do I build/compile a Working Set?.....	136
2.6.49	How do I search within the Working Set?.....	138
2.6.50	How do I manage the Working Sets?.....	139
2.6.51	How can I can filter the projects listed in the CodeWarrior Projects view based on Working Sets?.....	140
2.6.52	How can I import/export a Working Set in the CodeWarrior eclipse IDE?.....	141
2.6.53	What are Icon decorators?.....	144
2.6.54	How do I enable or disable an icon decorator?.....	144
2.6.55	Can the icon decorators be combined?.....	145

Section number	Title	Page
2.6.56	When I open a large file in CodeWarrior IDE, an Editor Scalability dialog box appears, listing the message, "You are opening a large file. Scalability mode has been turned on for this editor to help improve performance by disabling features requiring to parse the source file language." What does it mean?.....	146
2.6.57	How do I configure the Scalability settings?.....	146
2.6.58	What is Timing System?.....	147
2.6.59	Can I program a different Freedom Board using Freedom board as SWD programmer?.....	152
2.6.60	Can MCU10.4 recover a bricked OpenSDA Freedom Board?.....	152
2.6.61	Do I need to initialize my port pin?.....	153
2.6.62	For my bootloader I need a way to send a file with a terminal program. As my serial connection has only Tx and Rx, but no RTS/CTS lines for flow control, it is useful if the terminal program either implements software flow control (XON/XOFF), or a delay value for sending a file. What do I do?.....	153
2.6.63	How can I add my own code to be inserted as part of the system startup?.....	155
2.6.64	How do I bootstrap an application?.....	157
2.6.65	How do I decode S19 Files?.....	158
2.6.66	How do I fix the 3.3V voltage drop on FRDM-KL25Z Board?.....	161
2.6.67	How do I install open source USBDM?.....	163
2.6.68	How do I load the Debug Application on my FRDM-K20D50M board?.....	166
2.6.69	How do I make sure that application does not overlap with the FLASH area of the bootloader?.....	166
2.6.70	How do I make sure that the bootloader gets linked only into its specified space?.....	169
2.6.71	How do I set up criteria for bootloader to decide if it shall run the bootloader or the application at the startup?.....	170
2.6.72	How do I use CMSIS-DAP with IAR on KL25Z Freedom Board?.....	171
2.6.73	How do I use Freedom board as SWD programmer?.....	172
2.6.74	How do I use printf() with the FRDM-KL25Z Board without using the Processor Expert?.....	178
2.6.75	How does a bootloader work?.....	181
2.6.76	I am debugging FRDM-KL05Z with USBDM, but I received a 'Failed to resume target process., Downloading binary to target..' error message. What do I do?.....	186
2.6.77	My bootloader resides in the first lower flash pages. How do I avoid it to get destroyed and overwritten by the application?.....	190
2.6.78	The OpenSDA connection is not available in the Connection page of the wizard. What do I do?.....	191
2.6.79	What are the disadvantages of using printf() in the embedded system programming?.....	193



Section number	Title	Page
2.6.80	What are the features of freedom board FRDM-K20D50M?.....	193
2.6.81	What Bootloader Shell commands I can use?.....	195
2.6.82	What is a bootloader?.....	196
2.6.83	What is Bootloader Sequence?.....	197
2.6.84	What is the code size required for the bootloader for FRDM-KL25Z board?.....	198
2.6.85	Where can I find the software for my FRDM-K20D50M board?.....	199
2.6.86	Which device should I use to create a new project for my FRDM-K20D50M board using CodeWarrior?.....	199

### Chapter 3 Project

3.1	FAQs - Project.....	201
3.1.1	How do I add a new Source Folder to my project?.....	202
3.1.2	How can I add to my project a folder that exists outside the current workspace?.....	202
3.1.3	How do I add to my project a new file that exists outside the current workspace?.....	204
3.1.4	How can I export a project that I could send to someone else?.....	204
3.1.5	How do I instruct the CodeWarrior IDE to save changes to my source code files automatically before building the project?.....	205
3.1.6	How do I compare two source code files?.....	206
3.1.7	How do I compare two directories?.....	207
3.1.8	How can I specify the number of days to keep local history for source code files?.....	207
3.1.9	How can I see or change the preprocessor macros defined for my project?.....	208
3.1.10	Is there a shortcut to remove object code that is equivalent to the Ctrl + - shortcut key available in the Classic CodeWarrior IDE?.....	209
3.1.11	How can I view beans in a Processor Expert project?.....	210
3.1.12	How to use a configured bean from one project to the other?.....	210
3.1.13	Where can I find CodeWarrior for Microcontrollers V10.x example projects?.....	210
3.1.14	While launching a project, the CodeWarrior software automatically builds the current active target. How can I specify the build configuration I want to debug?.....	211
3.1.15	While building a project, the CodeWarrior fails to write an ELF file. How do I resolve this issue?.....	213
3.1.16	I am unable to create the new project at the desired location. What could be the possible reason and how to resolve this?.....	214

Section number	Title	Page
3.1.17	I want to build a library (.lib) instead of a final elf (.abs) file. How can I do this?.....	214
3.1.18	How do I filter the items listed in the Problems view based on the project selected?.....	215
3.1.19	Is there a way I can view the hierarchical structure for include files in my project?.....	216
3.1.20	How do I add/remove floating point format for S08 Projects?.....	218
3.1.21	How do I restore deleted files in Eclipse with local history?.....	220

## Chapter 4 Project Management

4.1	FAQs - Project Management.....	223
4.1.1	How can I find out where the source files related to a CodeWarrior project are saved?.....	224
4.1.2	How can I add another source or header file to my project?.....	224
4.1.3	What is a workspace?.....	225
4.1.4	What is a project?.....	225
4.1.5	How can I view the contents of various types of source code files that appear in my project folder?.....	225
4.1.6	Is there a way to compare the Property Settings of two different projects?.....	226
4.1.7	How do I resize the various views in a Perspective?.....	226
4.1.8	How can I add a project from some other workspace to my current workspace?.....	226
4.1.9	How do I ensure that the CodeWarrior IDE saves all modifications that I make to the project properties?.....	227
4.1.10	While working with the CodeWarrior IDE, the Remote System Changed dialog box appears. Why does it appear and how can I resolve the issue?.....	227
4.1.11	Which set of files I need to backup or store in a version control system to completely recover project settings and related debug settings?.....	229
4.1.12	I need to share a project without using a version control system. Is there any way to do this?.....	230
4.1.13	While importing a project, is there any way to find out which file mapping are missing?.....	232

## Chapter 5 Compiler

5.1	FAQs - Compiler.....	235
5.1.1	How do I compile my project?.....	236
5.1.2	Where do I see the warning and error messages?.....	236
5.1.3	When I compile my project, where can I see the commands that are being executed?.....	236
5.1.4	Does CodeWarrior for Microcontrollers v10.x supports the CMSIS 2.0 DSP library?.....	237

Section number	Title	Page
5.1.5	Is there a way to specify a half-precision data type in MCU 10.x?.....	237
5.1.6	#pragma pack() does not work in my Kinetis project. What do I do?.....	238
5.1.7	How do I generate a Cyclic Redundancy Check in Eclipse based CodeWarrior for Microcontrollers v10.x?...	238
5.1.8	I am using CodeWarrior for MCU10, Special Edition which allows up to 128 KByte of code to download for my ARM/Kinetis projects. But, when I tried to debug an Example that I have downloaded from the web, I get this dialog: "Download size limit has been exceeded. Please check your license." What do I do?.....	241
5.1.9	What is PC-lint?.....	242
5.1.10	What is Linticator?.....	242
5.1.11	How do I lint my project without using PC-lint plugin?.....	244
5.1.12	How do I configure the build settings to lint my project without plugin? .....	246
5.1.13	How do I set up a batch file to call PC-lint?.....	249
5.1.14	How do I define the list of files to lint my project without plugin?.....	250
5.1.15	How do I pass the project specific options to lint my project without plugin?.....	251
5.1.16	How do I define the message format to lint my project without plugin?.....	251

## Chapter 6 Linker

6.1	FAQs - Linker.....	253
6.1.1	Can I use a different linker command file than the one being used in my project?.....	253

## Chapter 7 Debugger

7.1	Breakpoints.....	255
7.1.1	How do I set breakpoints?.....	256
7.1.2	How do I view the breakpoints that have been set?.....	257
7.1.3	How do I disable existing breakpoints?.....	257
7.1.4	What does the checkmark next to the breakpoint represent?.....	258
7.1.5	What is the difference between the Toggle Breakpoint and Enable/Disable breakpoint options?.....	258
7.1.6	Which breakpoint option do I select if I want to add a breakpoint?.....	258
7.1.7	Which breakpoint option do I select if I want to skip a breakpoint while still keeping it in my breakpoint list?.....	258
7.1.8	What if I want to delete a breakpoint?.....	258

Section number	Title	Page
7.1.9	How do I place a breakpoint on a given physical address?.....	259
7.1.10	How do I add hardware breakpoints? Can I add them through the GUI?.....	259
7.1.11	How can I view the hardware breakpoints that have been set?.....	260
7.1.12	What is the difference between the hardware breakpoint and the software breakpoint? Why would I use one over the other?.....	260
7.1.13	What is the purpose of the Limit New Breakpoint to Active Debug Context button in the Breakpoint window?.....	260
7.1.14	Is there a way to add breakpoints to C and Assembly files that exist in a library?.....	260
7.1.15	How do I set a breakpoint at a particular function?.....	261
7.1.16	What is the difference between a breakpoint and a line breakpoint?.....	261
7.1.17	The warnings symbols in Editor distort visibility of breakpoints. How can I stop these warning symbols from appearing?.....	261
7.1.18	How do I use breakpoints on data?.....	262
7.1.19	Can I use any number of software breakpoints while debugging in RAM? .....	264
7.1.20	How does software breakpoints works while debugging in RAM?.....	264
7.1.21	While debugging in RAM my debugger halts and reports an "illegal breakpoint" error. What do I do?.....	266
7.1.22	Can I use software breakpoints on code running in flash memory?.....	266
7.1.23	What is a hardware breakpoint?.....	266
7.1.24	How do I select the type of breakpoint for my application in the Editor view?.....	266
7.1.25	How do I monitor that how many hardware breakpoints are used in my application?.....	267
7.1.26	If I launch my debug sessions with too many breakpoints, then the debugger shows this error message: "Breakpoint Warning: Unable to set all breakpoints requested. Some breakpoints may be disabled and will not stop the processor." What do I do?.....	268
7.1.27	How do I step over the breakpoints without using Breakpoints view?.....	268
7.1.28	What are the points to consider while using Stepping Backwards feature in Debugging?.....	270
7.2	Variables.....	270
7.2.1	By default, the debugger shows me the global variables of the currently debugged source file. How can I view my variables from another module?.....	270
7.2.2	How do I debug my application using global variables?.....	271
7.2.3	How do I filter variables in files?.....	272

Section number	Title	Page
7.2.4	How do I enable watch on a variable?.....	272
7.2.5	What is a static local variable?.....	273
7.2.6	When I enable watch on a static variable, it shows the error, "<error(s)_during_the_evaluation>". What do I do?.....	273
7.3	Build Configuration.....	273
7.3.1	How do I create a new build configuration for my project?.....	274
7.3.2	Is it possible to exclude a single or multiple files from the project while building it?.....	276
7.3.3	Is it possible to configure build settings for a particular source code file?.....	278
7.3.4	Is it possible to store output of the build configuration of my project in a separate directory?.....	279
7.3.5	How can I disable the option of having the CodeWarrior IDE build my project before launching it?.....	281
7.3.6	How can I export the launch configurations settings and then import them back into the project?.....	281
7.3.7	How do I ensure that the launch configurations are a part of the project when it is exported?.....	282
7.4	Flash Programmer.....	282
7.4.1	How can I flash or download a binary file to the target?.....	283
7.4.2	How can I create the flash programmer, hardware diagnostics, or import/export fill memory tasks in the CodeWarrior IDE?.....	283
7.4.3	I want to download my application directly into a RAM target. How do I do it?.....	284
7.4.4	I need to flash a large size application (more than 64 KB) , and I also need to perform target tasks, such as secure/unsecure the device and additional verify or erase. How do I do it?.....	285
7.4.5	Is there a way to quickly flash multiple boards with different S-Records?.....	288
7.4.6	Is it possible to debug the target multiple times with the same binary application, without flashing the device?.....	288
7.4.7	I need to enable logging for flash programming. How do I do it?.....	290
7.4.8	How can I program my boards without a project?.....	291
7.5	Miscellaneous.....	296
7.5.1	How can I view only certain registers?.....	297
7.5.2	How can I view addresses in memory?.....	299
7.5.3	How do I open up multiple memory locations in the memory view?.....	300
7.5.4	How can I view variables that I have set in my project?.....	301
7.5.5	How can I change the values of variables?.....	301

Section number	Title	Page
7.5.6	How do I view global variables that have been accessed in my project?.....	302
7.5.7	Is the command-line debugger available in the Eclipse-based CodeWarrior IDE as well?.....	303
7.5.8	Is it possible to specify a different name for the binary file (.abs)?.....	303
7.5.9	Can I debug the code from the generated assembly file step-by-step?.....	305
7.5.10	If I have a debugger session running on my application, how can I halt the application?.....	306
7.5.11	Why do I get the following error message when I attempt to debug my project?.....	306
7.5.12	Can I change the format of the Memory view?.....	306
7.5.13	Why does the DE.exe appear in the Windows Task Manager even after terminating and closing the debugger perspective?.....	309
7.5.14	Is there a way to display the variables always in the Hexadecimal format?.....	309
7.5.15	Can I change the endianness that is displayed in the Memory view?.....	311
7.5.16	In the Memory view, is it possible to go back to the address that the rendering was created for?.....	313
7.5.17	Is there a way to view disassembly at any location?.....	314
7.5.18	Is it possible to display the expressions always in the Hexadecimal format instead of the default decimal format? .....	315
7.5.19	What does the Reset on Connect option do?.....	316
7.5.20	Can I change the Program Counter (pc) value while in the debugger without having to restart the debugger?.	317
7.5.21	How do I specify the program entry point?.....	317
7.5.22	How can I run a debug session repeatedly?.....	318
7.5.23	Can I direct console output to a file instead of the console window?.....	319
7.5.24	How do I resolve the following error message that I get when I use Ctrl-Click to open variable declaration?.	320
7.5.25	How can I configure and launch an external tool or debugger?.....	321
7.5.26	How can I monitor a variable in the debugger while a target is running?.....	323
7.5.27	How can I inspect global variables in the variables view?.....	324
7.5.28	How do I resolve the error encountered on the Kwikstik board?.....	325
7.5.29	At the moment serial data is output in the console in the debugger, which is slow with J-Link. Is there any way to view the output of the debugger in the HyperTerminal?.....	326
7.5.30	Is it possible to flash a target MCU from the command-line?.....	327
7.5.31	Is there an easy way to suppress the target re-programming when there has been no change since the previous session?.....	328

Section number	Title	Page
7.5.32	The microcontrollers are usually trimmed in the factory. How can I trim it myself if I need something better?.....	328
7.5.33	How do I get a full dump of my device memory for offline analysis?.....	330
7.5.34	When I build my application I received the message, "Device is secure. Erase to unsecure?" What do I do?...	334
7.5.35	When I connect to my board, it shows a connection failure. What do I do?.....	335
7.5.36	How do I debug the strings?.....	336
7.5.37	I am debugging the strings using Variables view, but instead of the string, only the address of the first character or the location is shown in the preview panel. What do I do?.....	337
7.5.38	How do I view the declared type of the variables in the Variables view?.....	338

## Chapter 8 Debugger Shell

8.1	FAQs - Debugger Shell.....	341
8.1.1	How can I find the default value of memory width? How can I change the value if I want to?.....	341
8.1.2	Can I change the Program Counter (pc) value?.....	342
8.1.3	Does the config page off option still available in the Eclipse-based CodeWarrior IDE?.....	342
8.1.4	Is there a way to accelerate the execution speed and turn off the printing when I use a lot of mem commands in the debugger shell?.....	342
8.1.5	Is there any printf command in Debugger shell or command line?.....	343
8.1.6	How can I do test automation in the Debugger Shell?.....	344
8.1.7	How can I do CodeWarrior Flash programming from DOS Shell?.....	348
8.1.8	How do I create shell for Kinetis KL25Z freedom board?.....	350

## Chapter 9 Profiling and Analysis Tool

9.1	FAQs - Profiling.....	357
9.1.1	What do we mean by trigger A, trigger B, and trigger C?.....	358
9.1.2	What are start and stop triggers/tracepoints?.....	358
9.1.3	How do I set triggers A, B and C?.....	359
9.1.4	How do I enable trace and profiling?.....	359
9.1.5	How do I view the triggers/tracepoints that I have set?.....	360
9.1.6	How do I enable and disable triggers?.....	360

Section number	Title	Page
9.1.7	What is the difference between the Toggle Trace Trigger A/B and Enable/Disable Tracepoint options?.....	360
9.1.8	Which option do I select if I want to add a trigger?.....	361
9.1.9	Which option do I select if I want to skip a trigger still keeping it in my trigger list?.....	361
9.1.10	What if I want to delete a trigger?.....	361
9.1.11	How do I collect data?.....	361
9.1.12	How do I view trace, critical code, timeline, performance, and call tree data?.....	361
9.1.13	What is Timeline?.....	362
9.1.14	What does Critical Code data display?.....	362
9.1.15	What is Performance data?.....	362
9.1.16	What is Call Tree data?.....	363
9.1.17	What is the use of Selection Mode in TraceTimelineEditor?.....	363
9.1.18	How do I export critical code and performance data in a CSV file?.....	363
9.1.19	How can I save my trace results for later use?.....	364
9.1.20	When do we use the Halt the Target when Trace Buffer gets Full option?.....	364
9.1.21	When do we use the Break on Trigger Hit and Break on FIFO Full options?.....	364
9.1.22	When do I use the "Keep Last Buffer Before Trigger" option?.....	364
9.1.23	What are memory access triggers?.....	365
9.1.24	How to set a trigger from the Disassembly view?.....	365
9.1.25	What is LOOP1 Mode?.....	365
9.1.26	What is Profile-Only mode?.....	365
9.1.27	What is the difference between Automatic and Continuous mode?.....	366
9.1.28	What is the purpose of Resume and Suspend buttons in the Profile Results view?.....	366
9.1.29	How do I add a new address tracepoint while debugging a project?.....	366
9.1.30	Are there any limitations to use the debugger and the profiling tool (with tracepoints) simultaneously?.....	367
9.1.31	How are ColdFire V2 - V4 targets different from ColdFire V1 target?.....	367
9.1.32	Is trace feature supported on OSJTAG?.....	368
9.1.33	Which derivatives of the MPC56xx target support tracing?.....	368
9.1.34	What is Data Visualization?.....	368
9.1.35	Can I set analysispoints for data visualization?.....	368



<b>Section number</b>	<b>Title</b>	<b>Page</b>
9.1.36	What are Hardware and Software tracepoints?.....	368
9.1.37	How do I set Hardware/Software tracepoints?.....	369
9.1.38	What is Embedded Trace Macrocell (ETM)?.....	369
9.1.39	What is Instrumentation Trace Macrocell (ITM)?.....	369
9.1.40	What is Embedded Trace Buffer (ETB)?.....	370
9.1.41	What is J-Trace? How do I collect data using J-Trace?.....	370
9.1.42	How can I enable software tracepoints automatically on Kinetis target?.....	371



# Chapter 1

## Introduction

This FAQ Guide lists most frequently asked or most anticipated questions about Microcontrollers V10.x. In this document, the FAQs are divided into various categories based on the Microcontrollers V10.x feature they are associated with, such as Installation, IDE, Editor, Project Management, Project, Compiler, Assembler, Linker, Debugger, Debugger Shell, USBTAP, Ethernet TAP, and Profiling and Analysis Tools.

In this chapter:

- [Contents of this Manual](#) - Describes the contents of this manual.
- [What's New](#) - Lists new FAQs in this manual.
- [Accompanying Documentation](#) - Describes supplementary CodeWarrior documentation, third-party documentation, and references.

### 1.1 Contents of this Manual

The following table lists and describes each chapter in this manual. Each chapter lists a specific category of FAQs.

**Table 1-1. Manual Contents**

Chapter	Description
<a href="#">IDE and Installation</a>	Lists IDE related FAQs
<a href="#">Project Management</a>	Lists project management related FAQs
<a href="#">Project</a>	Lists project related FAQs
<a href="#">Compiler</a>	Lists compiler related FAQs
<a href="#">Linker</a>	Lists linker related FAQs
<a href="#">Debugger</a>	Lists debugger related FAQs
<a href="#">Debugger Shell</a>	Lists debugger shell related FAQs
<a href="#">Profiling and Analysis Tool</a>	Lists profiler related FAQs

## 1.2 What's New for Microcontrollers 10.5

The manual includes these new FAQs.

- **NEW** What are the features of freedom board FRDM-K20D50M?
- **NEW** What are the disadvantages of using printf() in the embedded system programming?
- **NEW** The OpenSDA connection is not available in the Connection page of the wizard. What do I do?
- **NEW** My bootloader resides in the first lower flash pages. How do I avoid it to get destroyed and overwritten by the application?
- **NEW** I am debugging FRDM-KL05Z with USBDM, but I received a 'Failed to resume target process., Downloading binary to target..' error message. What do I do?
- **NEW** How does a bootloader work?
- **NEW** How do I use printf() with the FRDM-KL25Z Board without using the Processor Expert?
- **NEW** How do I use Freedom board as SWD programmer?
- **NEW** How do I use CMSIS-DAP with IAR on KL25Z Freedom Board?
- **NEW** How do I set up criteria for bootloader to decide if it shall run the bootloader or the application at the startup?
- **NEW** How do I make sure that the bootloader gets linked only into its specified space?
- **NEW** How do I make sure that application does not overlap with the FLASH area of the bootloader?
- **NEW** How do I load the Debug Application on my FRDM-K20D50M board?
- **NEW** How do I install open source USBDM?
- **NEW** How do I fix the 3.3V voltage drop on FRDM-KL25Z Board?
- **NEW** How do I decode S19 Files?
- **NEW** How do I bootload an application?
- **NEW** How can I add my own code to be inserted as part of the system startup?
- **NEW** For my bootloader I need a way to send a file with a terminal program. As my serial connection has only Tx and Rx, but no RTS/CTS lines for flow control, it is useful if the terminal program either implements software flow control (XON/XOFF), or a delay value for sending a file. What do I do?
- **NEW** Do I need to initialize my port pin?
- **NEW** Can MCU10.4 recover a bricked OpenSDA Freedom Board?
- **NEW** Can I program a different Freedom Board using Freedom board as SWD programmer?
- **NEW** What Bootloader Shell commands I can use?
- **NEW** What is a bootloader?
- **NEW** What is Bootloader Sequence?

- **NEW** What is the code size required for the bootloader for FRDM-KL25Z board?
- **NEW** Where can I find the software for my FRDM-K20D50M board?
- **NEW** Which device should I use to create a new project for my FRDM-K20D50M board using CodeWarrior?
- **NEW** How do I add/remove floating point format for S08 Projects?
- **NEW** How do I restore deleted files in Eclipse with local history?
- **NEW** What are the points to consider while using Stepping Backwards feature in Debugging?

## 1.3 Accompanying Documentation

The **Documentation** page describes the documentation included in the CodeWarrior Development Studio for Microcontrollers v10.x.

You can access the **Documentation** by:

- opening the `START_HERE.html` in `<CWInstallDir>\MCU\Help` folder,
- selecting **Help > Documentation** from the IDE's menu bar, or selecting the **Start > Programs > Freescale CodeWarrior > CW for MCU v10.x > Documentation** from the Windows taskbar.

### NOTE

To view the online help for the CodeWarrior tools, first select **Help > Help Contents** from the IDE's menu bar. Next, select required manual from the **Contents** list. For general information about the CodeWarrior IDE and debugger, refer to the *Codewarrior Common Features Guide* in this folder: `<CWInstallDir>\MCU\Help\PDF`



## Chapter 2

# IDE and Installation

In this chapter, you find Frequently Asked Questions (FAQs) related to the CodeWarrior IDE.

The FAQs listed in this chapter are divided into five categories:

- [Updating and Installing Software](#)
- [Editor](#)
- [Workbench Window](#)
- [CVS](#)
- [Search and Replace Action](#)
- [Miscellaneous](#)

### 2.1 Updating and Installing Software

In this topic, installation related FAQs are listed.

- [I have installed CodeWarrior for Microcontrollers on Linux Ubuntu 8.04, 64-bit. However, Multilink drivers are not working. What do I do?](#)
- [I want to install updates for the CodeWarrior software. However, when I check for updates, no updates are found. What do I do?](#)
- [I am facing issues in accessing the updater sites. How do I resolve this?](#)
- [My attempt to install the CodeWarrior tool fails with the error "Another version of this product is already installed". What do I do?](#)
- [How do I uninstall the third party plugins?](#)
- [After updating the CodeWarrior, I receive the following error while restarting the application, "'Initialize RSE' has encountered a problem". What should I do?](#)
- [How do I know what updates/patches are already installed?](#)

- I have installed the CodeWarrior patches on Windows 7. The patches have installed correctly, however the CodeWarrior software generates error while building the libraries. Is there any way to resolve this?
- How can I check if the compiler updater is successfully installed?

### **2.1.1 I have installed CodeWarrior for Microcontrollers on Linux Ubuntu 8.04, 64-bit. However, Multilink drivers are not working. What do I do?**

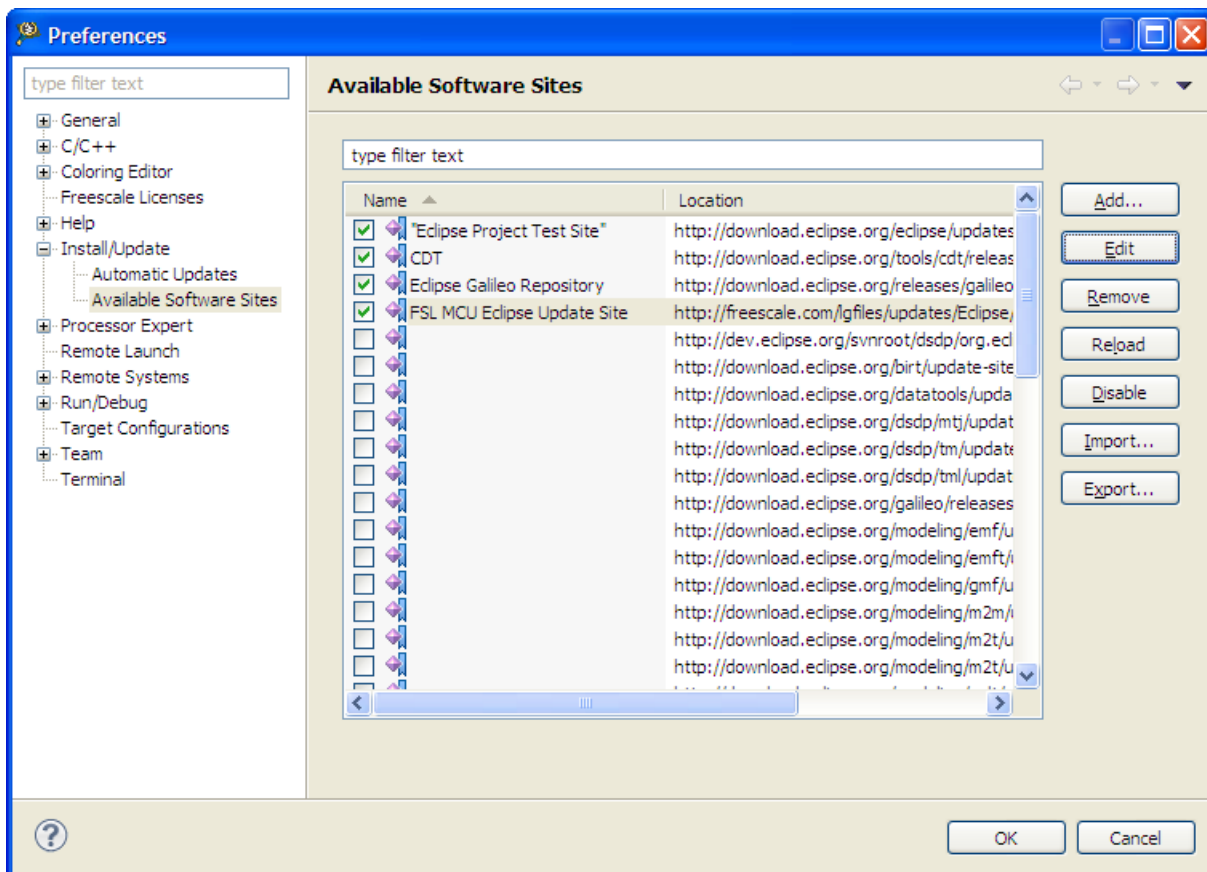
64-bit drivers for P&E Devices are not supported for CodeWarrior for Microcontrollers v10.1. However, this support is available for CodeWarrior for Microcontrollers v10.2.

### **2.1.2 I want to install updates for the CodeWarrior software. However, when I check for updates, no updates are found. What do I do?**

If you are using **Help > Check for Updates** option to check for new service packs or patches, you may get the message "There is nothing to update", even when patches and service packs are available.

This happens because the **Check for Updates** option, checks for updates available for already installed packages or features. The **Check for Updates** option scans through the list of the installed plugins and contacts the configured and enabled update sites (for your reference, see the image listed below).





**Figure 2-1. Available Software Sites**

For example, if you have installed a service pack, the CodeWarrior software will search for an update available for the service pack. However, it will not search for the recent Kinetis compiler, as the Kinetis compiler has not been installed previously as a separate package or feature.

To install new service packs or patches, use the **Help > Install New Software** option. While installing new software, make sure that you clear the **Contact all update sites during install to find required software** checkbox.

Also, note that if the CodeWarrior software needs to access updates through proxy, you need to configure the network connections.

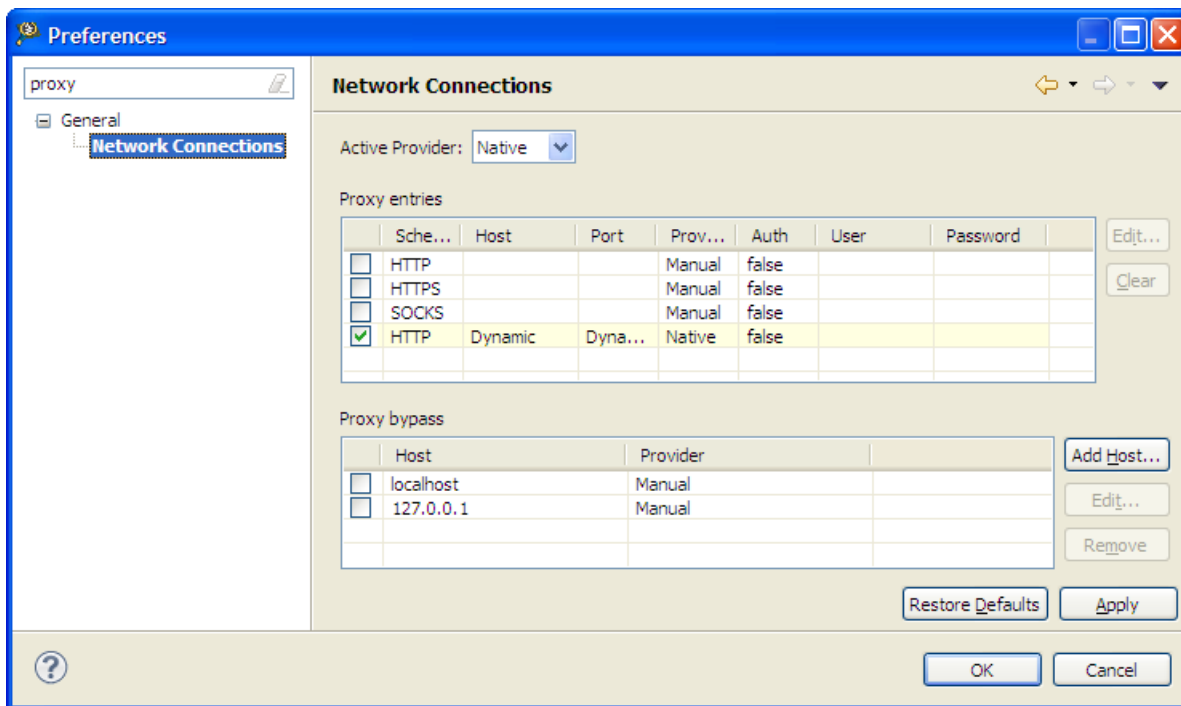


Figure 2-2. Network Connections

### 2.1.3 I am facing issues in accessing the updater sites. How do I resolve this?

The **Install New Software** option requires minor modification to the CodeWarrior V10.x site pointers to enable correct behavior.

If you have problem to access an updater site, add a backslash to the end.

For example, add a backslash at the end of the Freescale update site:

[http://freescale.com/lgfiles/updates/Eclipse/MCU10\\_1/com.freescale.mcu.update.site/](http://freescale.com/lgfiles/updates/Eclipse/MCU10_1/com.freescale.mcu.update.site/)

Similarly, for SVN Connectors site:

<http://www.polarion.org/projects/subversive/download/eclipse/2.0/update-site/>

For other sites, such as:

<http://download.gna.org/eclox/update>

backslash is not required, however it seems to be a safe practice to add `/' at the end of all the updater sites.

## 2.1.4 My attempt to install the CodeWarrior tool fails with the error "Another version of this product is already installed". What do I do?

This happens when while un-installing the CodeWarrior tool, the installer crashes in between. The files are removed from the hard drive and the icons are removed from the Start menu. Also, the CodeWarrior tool does not appear in the Add/Remove Programs panel. But, your system will not let you install another version of the CodeWarrior tool, as there are still some files remaining.

To remove the remaining CodeWarrior files:

1. Open a command prompt and go to the folder containing the `setup.exe` of the CodeWarrior tool you want to uninstall.
2. In this folder, edit the `setup.ini` file and search for the value, `ProductCode`.
3. Enter the following command at the command prompt:

```
msiexec /i[ProductCode]
```

The CodeWarrior setup dialog box appears.

4. Repair or remove the current installation.
5. Remove the remaining files.
6. Re-launch the setup of the CodeWarrior tool you want to install.

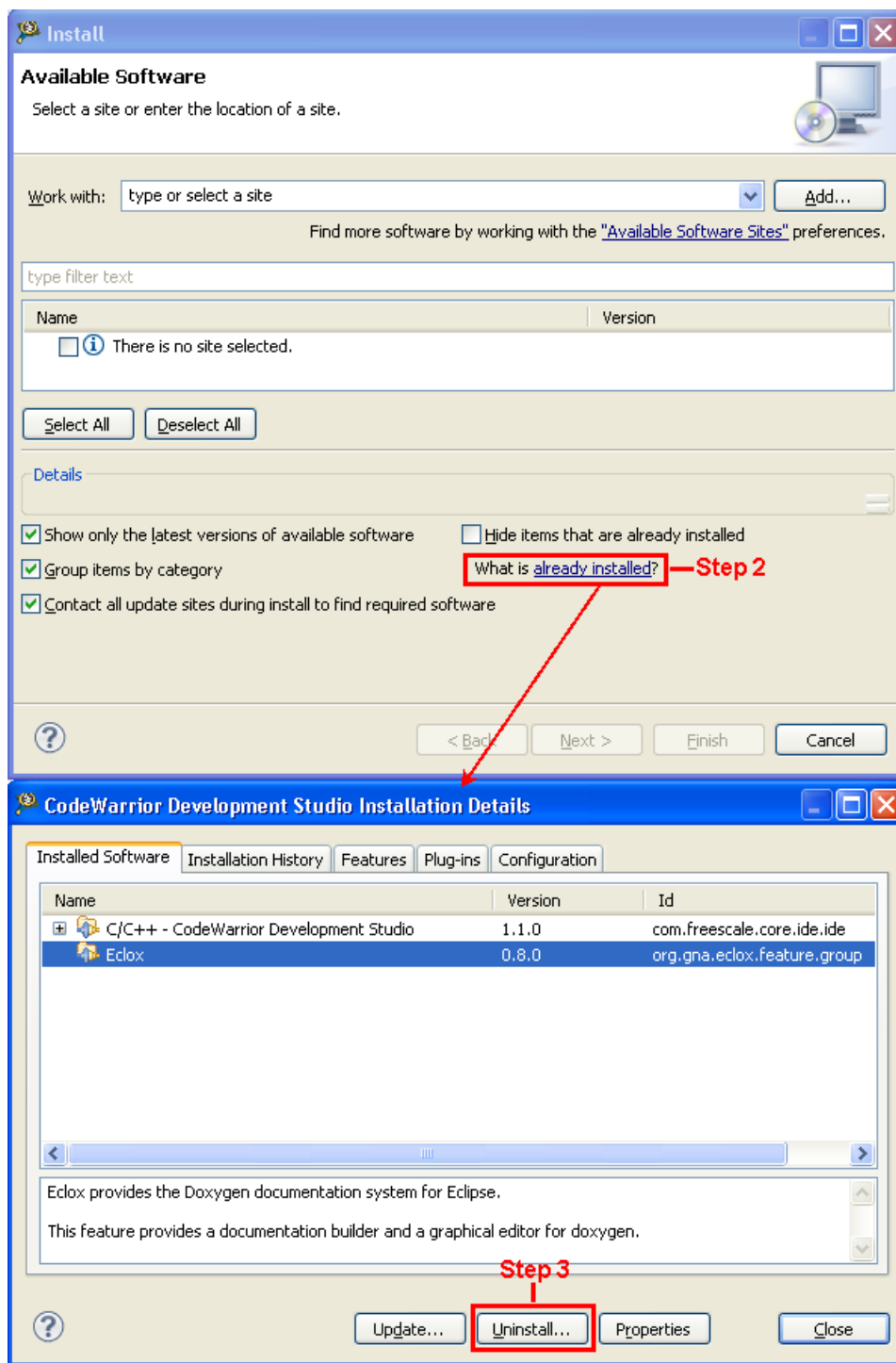
You can now successfully install another version of the CodeWarrior tool.

## 2.1.5 How do I uninstall the third party plugins?

To uninstall a third party plugin, follow the steps listed below:

1. Select **Help > Install New Software** from the IDE menu bar.

The **Install** dialog box appears.



**Figure 2-3. Install Dialog Box**

2. Click the **What is already installed?** option.

The **CodeWarrior Development Studio Installation Details** dialog box appears.

3. Select the desired plugin and click **Uninstall** .

The **Uninstall** dialog box appears.

4. Click **Finish** .

The IDE uninstalls the selected plugin.

### 2.1.6 After updating the CodeWarrior, I receive the following error while restarting the application, "'Initialize RSE' has encountered a problem". What should I do?

You have to check that whether all the features available into the MCU10.1 Update package has been installed or not.

For this, open the **Install** dialog box by selecting **Help > Install New Software** from the IDE menu bar. When you check the **Group items by category** and **Hide items that are already installed** checkboxes, the **All Items are installed** message (refer to the image listed below) appears under the **CW MCU v10.1 Update** category.

If this is not the case, you have to install all the other features that are appearing in the list.

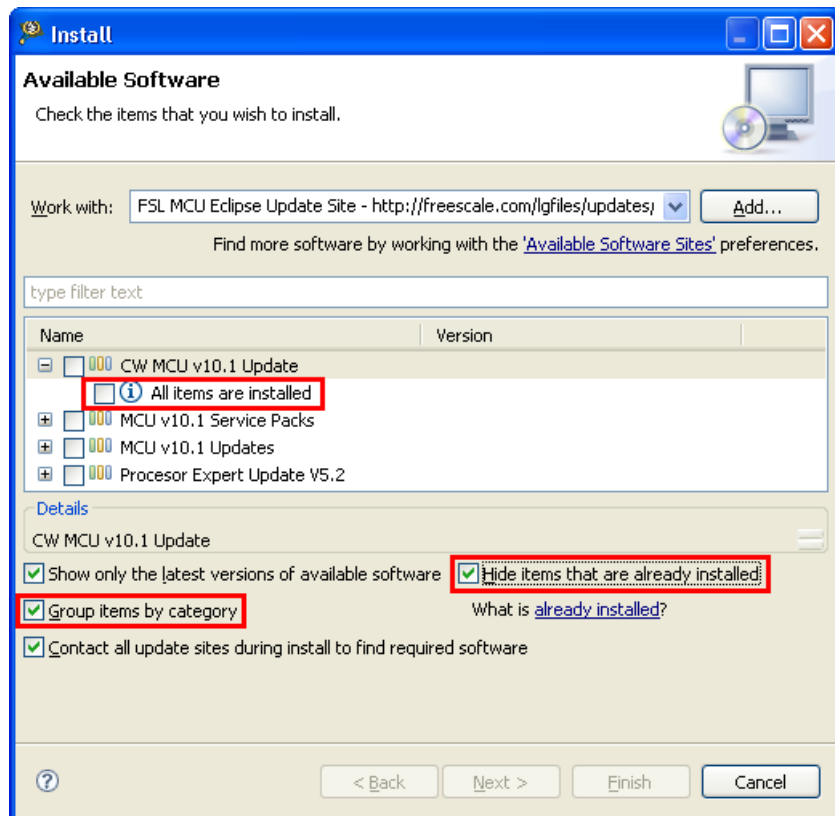


Figure 2-4. MCU Update - Install Dialog Box

## 2.1.7 How do I know what updates/patches are already installed?

To view the list of installed updates/patches, follow the steps listed below:

1. Select **Help > Install New Software** from the IDE menu bar.

The **Install** dialog box appears.

2. Select **All Available Sites** from the Work with drop-down list.

The already installed updates/patches appears in gray icons.

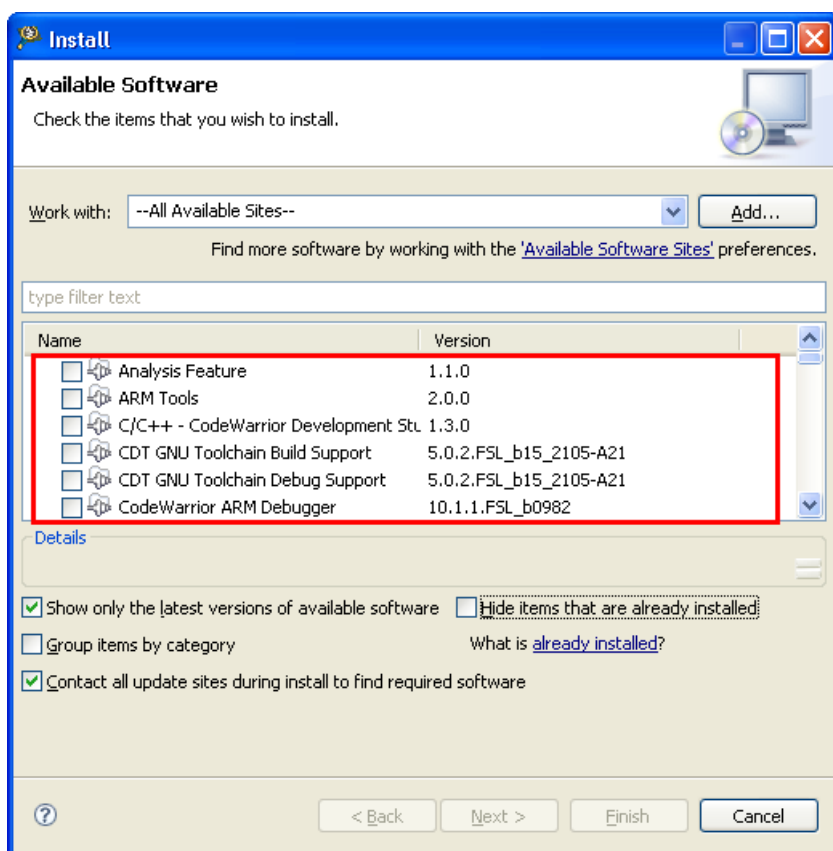


Figure 2-5. Install dialog box - Listing Installed Updates

## 2.1.8 I have installed the CodeWarrior patches on Windows 7. The patches have installed correctly, however the

## CodeWarrior software generates error while building the libraries. Is there any way to resolve this?

If you install patches under non-admin privileges, all looks like it goes well. It shows that the patches are installed. However, when you build the libraries, you still get the error. Also, there is no way to uninstall the patches so you can re-install them with the admin privileges.

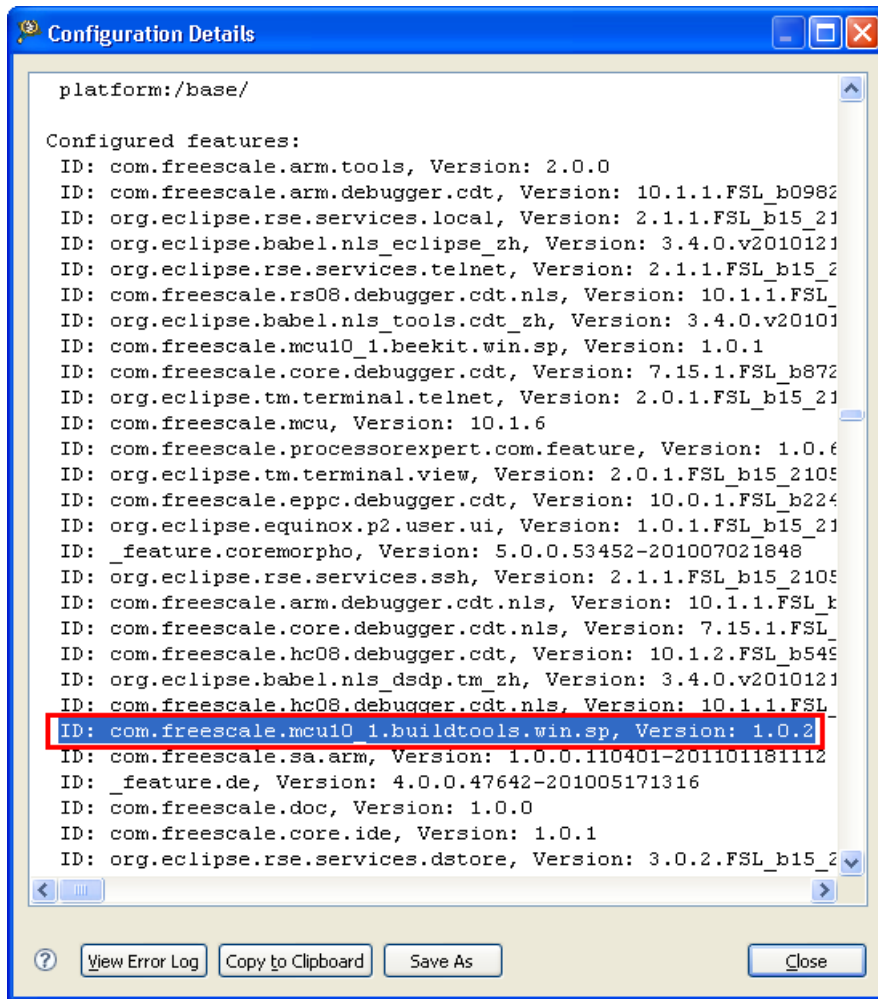
There are two ways to work this around:

- Install the Codewarrior with RUN AS ADMIN and run the updates with admin privileges when you install patches.
- Install the CodeWarrior in the non-default location (means, not in the Windows default `Program Files` folder) but install it in a directory that you can have admin rights (IE-Users/profile etc).

### 2.1.9 How can I check if the compiler updater is successfully installed?

Follow the steps listed below for CodeWarrior for Microcontrollers v10.1:

Select **Help > About CodeWarrior Development Studio > Configuration Details** from the IDE menu bar and search for the **Configured Features**. The `com.freescale.mcu10_1.buildtools.win.sp` feature (refer to the image listed below) in the list shows that the compiler updater is successfully installed.



**Figure 2-6. Configuration Details Dialog Box**

Follow the steps listed below for CodeWarrior for Microcontrollers v10.2:

Select **Help > About CodeWarrior Development Studio > Installation Details** from the IDE menu bar.

All installed softwares are listed in the **Installation Details** dialog box.



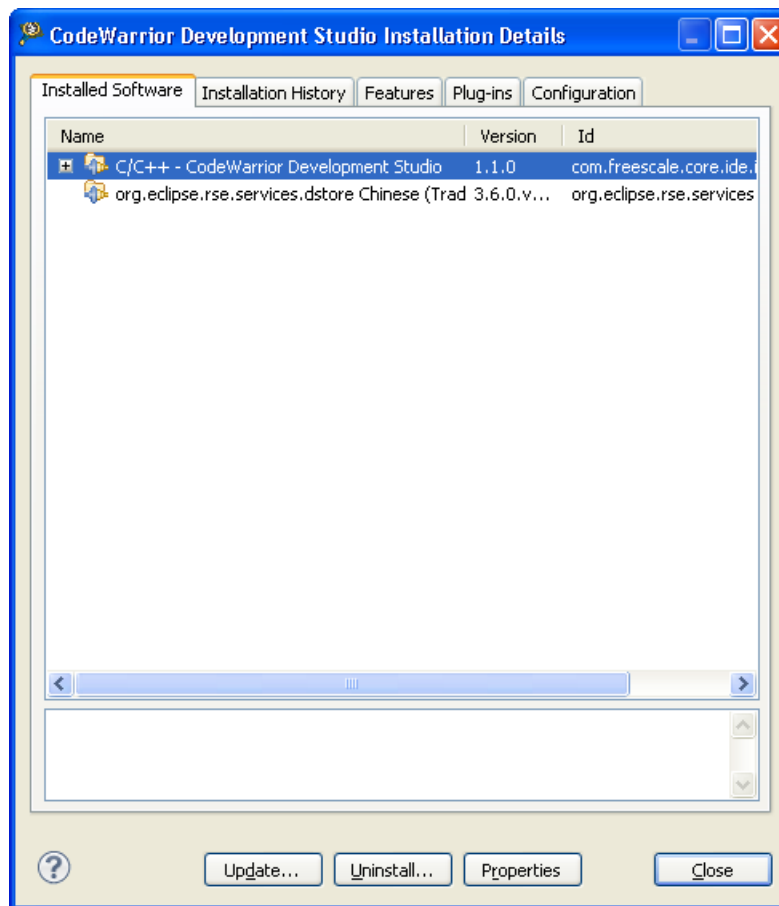


Figure 2-7. Installation Details Dialog Box

## 2.2 Editor

In this topic, Editor related FAQs are listed.

- How can I quickly open declaration of any variable, function name, macro, or header file from within Editor?
- How can I open definition of a function or header file from within the editor?
- If I am using a Macintosh computer, what is the alternate to using the F3 key in Editor to open declarations?
- How can I change the color that highlights the current line in a source code file?
- Is it possible to view definition of a macro or a variable in the source code file?
- Can the CodeWarrior IDE assist me in writing the source code?
- How can I change the tab width/size?
- Is it possible to display line numbers in Editor?
- Is it possible to change the default code format settings?
- How can I quickly switch to a header file from within the source code file?

- How does the CodeWarrior IDE differentiate between enabled and disabled macros?
- How can I configure predefined macros in the CodeWarrior IDE?
- Is it possible to view evaluated expansion of a macro in the CodeWarrior IDE?
- Is it possible to roll back the changes I did to my source code?
- Is there a way to change the default color of comments that span across multiple lines?
- Is there a way to revert to the original contents of a source code file?
- How do I convert Line Delimiters to the Unix format?
- Why do I get the following error message when I try to edit a source code file?
- What does an asterisk on the Editor title bar mean?
- How can I find out where a function is declared in my source code?
- How can I open .tcl extension files in the Editor window?
- Is it possible to condense a large source file by folding/unfolding different portions of code?
- How can I generate S19, Intel Hex, and Binary files for my project?
- How can I use the Burner for the Kinetis derivatives?
- I have created two build configurations one for debug and another for release. How can I specify a burner bbl file for each build configuration?
- Is there a way to determine in which workspace I am working currently?
- How can I share same installation of the CodeWarrior software with multiple users?
- How can I comment/uncomment a block of code in the editor?
- How can I compare files in CodeWarrior IDE?
- Is there a way to compare files that are not in project?
- How do I switch between the source files in the Editor view?
- I want to edit my source files, but all the other views are using the space as well. How do I maximize the editor view to have better working area?
- How do I remove blank lines in the Editor view?

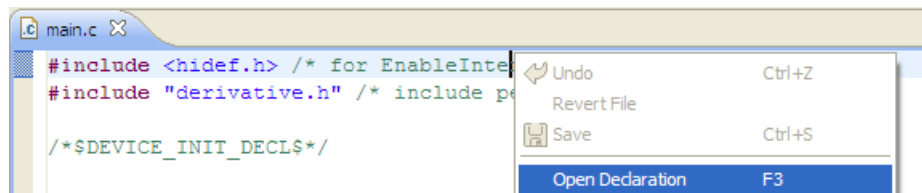
### 2.2.1 How can I quickly open declaration of any variable, function name, macro, or header file from within Editor?

To quickly open declaration of any variable, function name, macro or header file from within Editor, press the **Ctrl** key and click the text. Alternatively, place the mouse cursor on the text and perform either of the following:

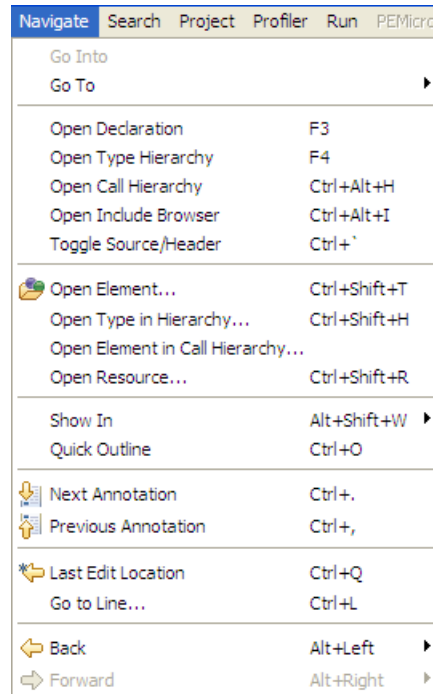
- press F3,

- right-click and select **Open Declaration** (as the figure, *Open Declaration* listed below shows), or
- select **Navigate > Open Declaration** from the IDE menu bar ( as the figure, *Navigate Menu* listed below shows).

This will open the logical file associated with the selected text. For example, for a header file include, the header file will open. For variables, the file containing the definition or declaration of the variable will open.



**Figure 2-8. Open Declaration**



**Figure 2-9. Navigate Menu**

The **Navigate** menu lets you perform various navigation actions in the source file, such as opening header files, viewing macro or variable definitions, navigating to the last edit location or previous source file. You can also perform these actions using the keyboard shortcuts.

## 2.2.2 How can I open definition of a function or header file from within the editor?

The CodeWarrior Eclipse IDE uses an indexer to build index of all sources in the background. To open definition, you need to configure indexer preferences. The indexer can be configured per project or per workspace basis.

1. Select **Window > Preferences** .

The Preferences dialog box appears.

2. Type **indexer** as the filter text to narrow down the list of preferences.
3. Select **C/C++ > Indexer** .

The **Indexer** page appears in the right panel.

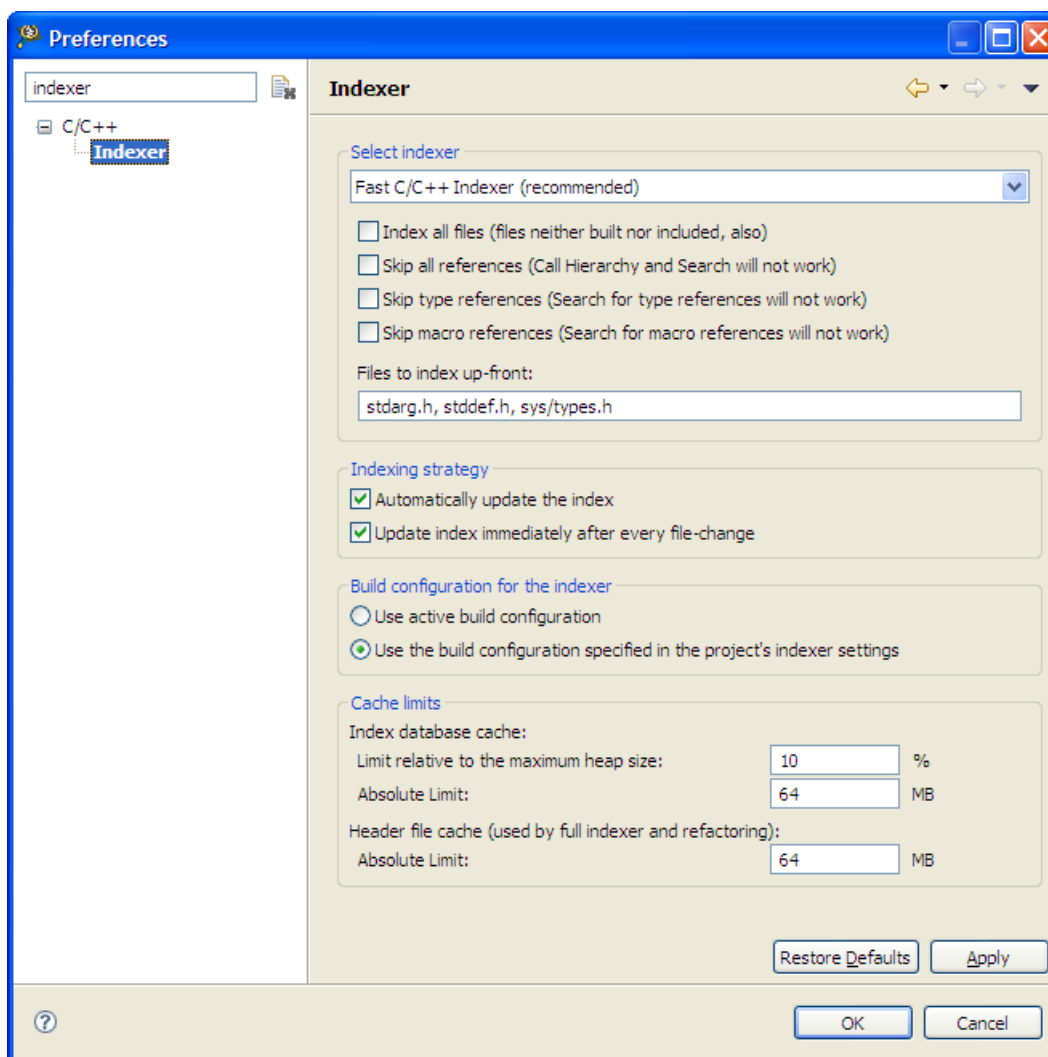


Figure 2-10. Indexer Page

4. From the **Select indexer** drop-down list, select the **Full C/C++ Indexer** option.
5. Click **OK** .
6. Right-click the project with which the source file is associated in the **CodeWarrior Projects** view and select **Index > Rebuild** .
7. Press F3, definition will open.

```
void TestFunc(void) {
}
```

**Figure 2-11. Function Definition**

8. Press F3 again to switch to the function declaration.

```
void TestFunc(void);
```

**Figure 2-12. Function Declaration**

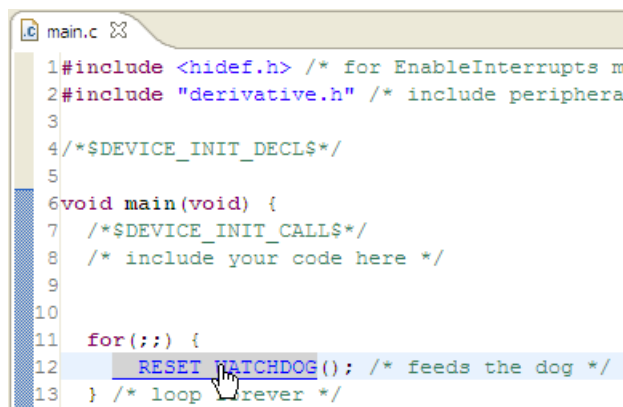
9. Press ALT + Left to switch to the original location.

Repeat step 7-9 to open function declarations and definitions of other functions in the source file.

### 2.2.3 If I am using a Macintosh computer, what is the alternate to using the F3 key in Editor to open declarations?

If you are using a Macintosh computer, just keep the CTRL key pressed and move the mouse cursor over the header file or function whose declaration or definition you want to open. The hyperlink for that header file or function enables.

Click the hyperlink to switch to the required declaration or definition, as the following figure shows:



**Figure 2-13. Hyperlink in Source Files**

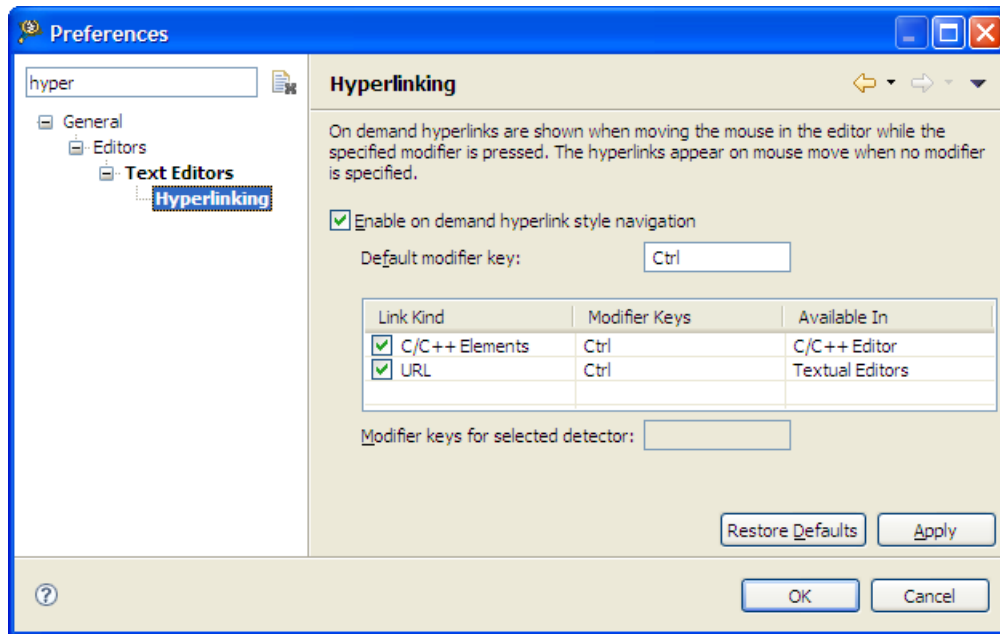
By default, the modifier key for hyperlinks in editor is Ctrl; however, you can change the modifier key as per your preference.

1. Select **Window > Preferences**.

The **Preferences** dialog box appears.

2. Type **Hyperlink** as the filter text to narrow down the list of preferences.
3. Select **General > Editors > Text Editors > Hyperlinking**.

The **Hyperlinking** page appears in the right panel.



**Figure 2-14. Hyperlinking Page**

4. Enter the required modifier key in the **Default modifier key** text box.
5. Click **OK**.

The modifier key changes, as specified.

## 2.2.4 How can I change the color that highlights the current line in a source code file?

To change the color for the current line highlight in a source file:

1. Select **Window > Preferences** .

The **Preferences** dialog box appears.

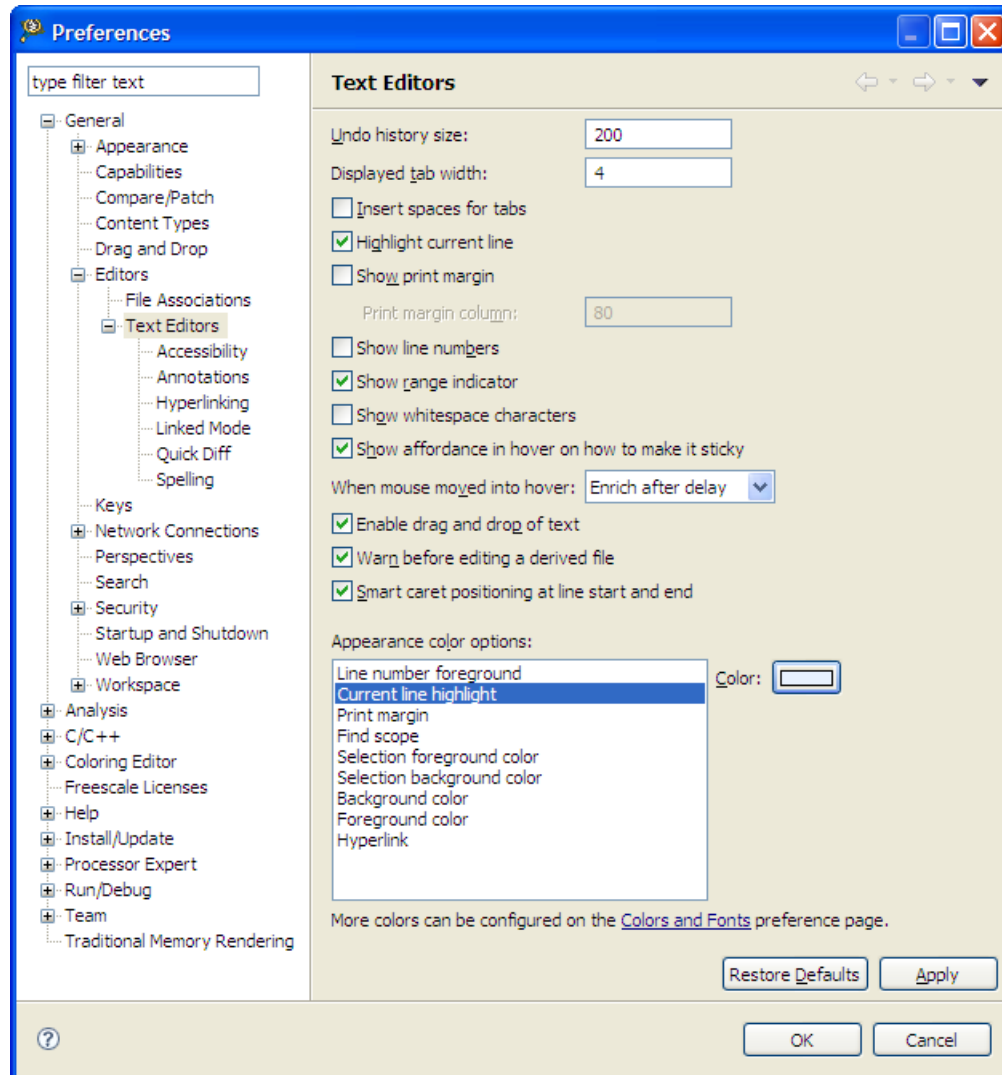
2. Type **text editor** as the filter text to narrow down the list of preferences.

3. Select **General > Editors > Text Editors** .

The **Text Editors** page appears in the right panel.

4. Ensure that **Highlight current line** checkbox is checked.

5. Select **Current line highlight** in the **Appearance color options** list.



**Figure 2-15. Text Editors Page > Current line highlight Selected**

6. Click the **Color** button to open the color panel, and select the required color.

7. Click **OK** to close the color panel.

8. Click **OK** to save the settings.

The current line highlight color changes, as specified.

## 2.2.5 Is it possible to view definition of a macro or a variable in the source code file?

Yes, you can open definition of a macro or variable by hovering the mouse cursor over the macro or variable. A small popup window displaying the definition of the macro ( refer to the image, *Macro Definition* as listed below) or variable ( refer to the image, *Variable Declarations and Occurrences* as listed below) appears.

Also, the IDE highlights all the occurrences of the variable in the source file, so you do not need to search for it manually.

```

for(;;) {
    RESET_WATCHDOG(); /* feeds the dog */
}
/* you never leave main */
{asm sta _SRS.Byte;}
Press 'F2' for focus
    
```

Figure 2-16. Macro Definition

```

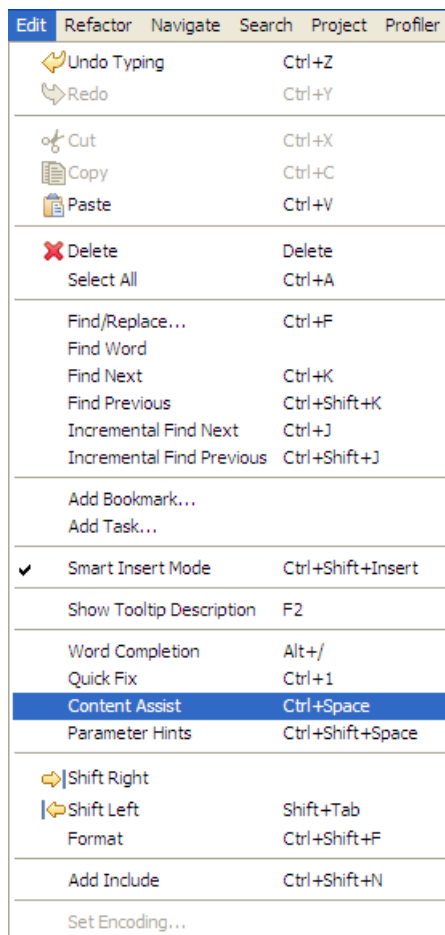
timer.c | Timer.txt | keyboard.h | events.c | main.c | trigger.c
64
65 #if 1
66 void DBNC_Process(DBNC_FSMDData *data) {
67     DBNC_KeySet keys;
68
69     switch(KBD_KeyState) {
70     case static KBD_KeyStateKinds KBD_KeyState = KBD_KEY_IDLE;
71         KBD_LongKeyCount = 0;
72         KBD_KeyState = KBD_KEY_PRESSED; /* advance to next state */
73         TRG_AddTrigger(TRG_KEYPRESS, KBD_KEY_IDLE_TO_PRESSED_NOF_1
74         break;
75
76     case KBD_KEY_PRESSED:
77         keys = data->GetKeysFn();
78         if (keys == KBD_ScanValue) { /* still pressing the same ke
79             KBD_LongKeyCount++;
80             if (KBD_LongKeyCount >= (KBD_LONG_KEY_TIME_MS/(KBD_KEY_F
81                 /* yes, long key press detected */
82                 KBD_LongKeyCount = KBD_LONG_KEY_ITERATIONS; /* avoid c
83                 data->LongKeyEventFn(keys);
84                 KBD_KeyState = KBD_KEY_WAIT_RELEASE; /* advance to nex
85                 TRG_AddTrigger(TRG_KEYPRESS, KBD_KEY_PRESSED_TO_WAIT_F
86             } else {
87                 TRG_AddTrigger(TRG_KEYPRESS, KBD_KEY_PRESSED_TO_PRESSE
88             }
89         } else { /* we got another key set pressed: so it was only
90             data->ShortKeyEventFn(KBD_ScanValue);
91             KBD_KeyState = KBD_KEY_WAIT_RELEASE; /* advance to next
92             TRG_AddTrigger(TRG_KEYPRESS, KBD_KEY_WAIT_RELEASE_TO_WAI
93         }
94     }
    
```

Figure 2-17. Variable Declaration and Occurrences



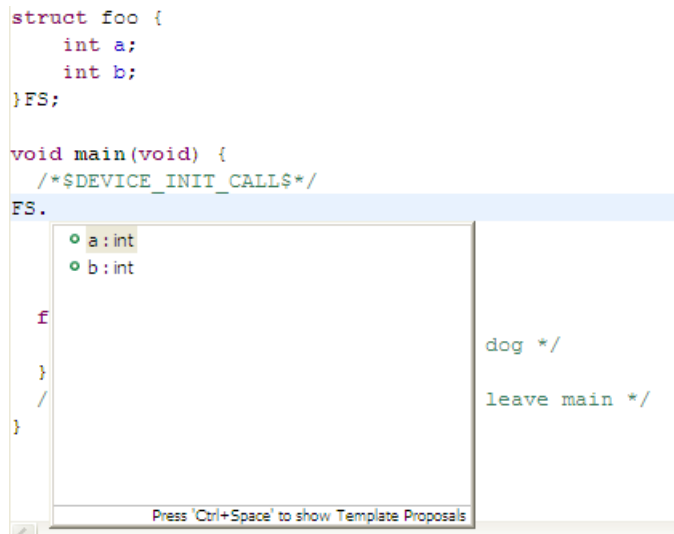
## 2.2.6 Can the CodeWarrior IDE assist me in writing the source code?

Yes, CodeWarrior IDE can parse the source files in the background. Therefore, it provides you the code completion feature. To use the code completion feature, select **Edit > Content Assist** from the IDE menu bar.



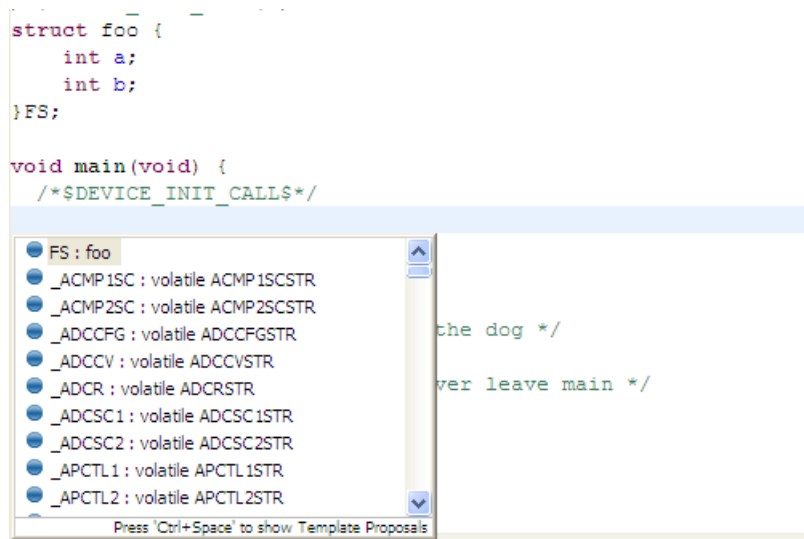
**Figure 2-18. Edit > Content Assist**

The Content Assist feature allows you to view the list of the field member of a structure, class, or union. The list of the field members appears automatically the moment you type ``.'`` following the structure, class, or union name (refer to the image listed below). You can also press **Ctrl+Space** to view the list.



**Figure 2-19. View Field Members of Structure**

If you press **Ctrl+Space** without typing anything or by placing the cursor in a blank line in the source file, the complete list appears, as shown in the figure listed below.



**Figure 2-20. Content Assist - Without Typing Anything**

You can also view the list of bitfields using Content Assist, as shown in the figure listed below.

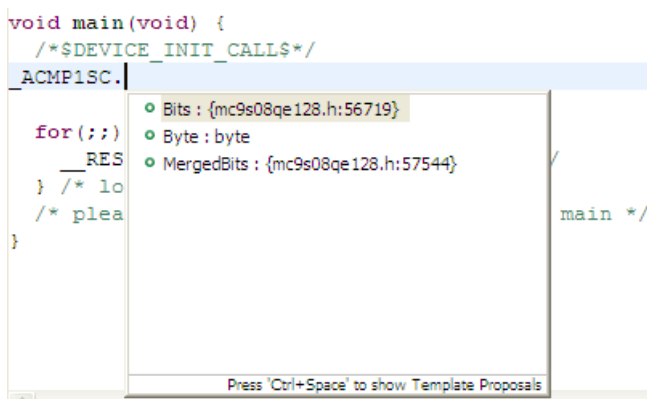


Figure 2-21. Bitfields

## 2.2.7 How can I change the tab width/size?

By default, the tab size in the CodeWarrior IDE is 4. To change the default tab size:

1. From the IDE menu bar, select **Window > Preferences** .
2. Type **tab** as the filter text to narrow down the list of preferences.
3. Select **Text Editors** .

The **Text Editors** preference page appears in the right panel of the **Preferences** dialog box.

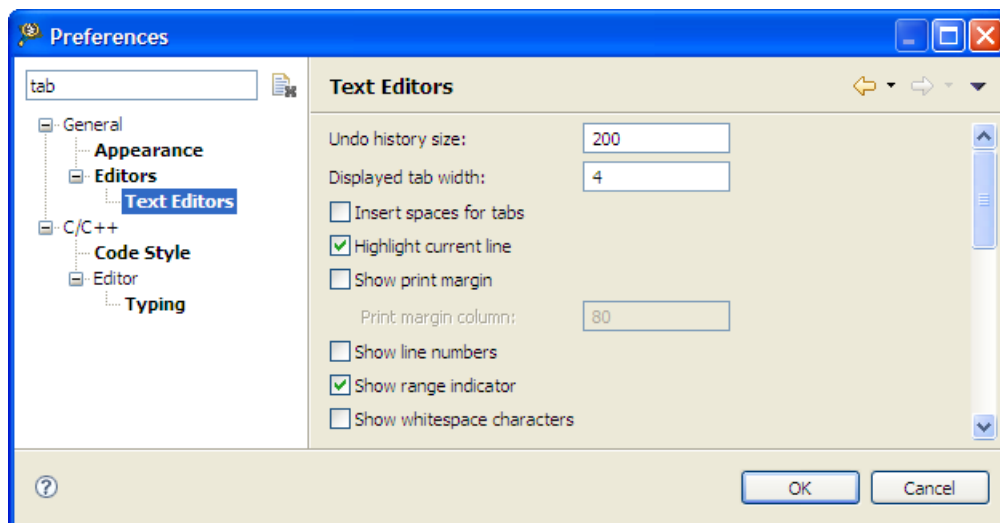


Figure 2-22. Text Editors Preferences Page

4. Enter the desired tab size in the **Displayed tab width** text box.
5. Click **OK** .

The default tab size changes.

## 2.2.8 Is it possible to display line numbers in Editor?

Yes, to display line numbers in the editor area:

1. From the IDE menu bar, select **Window > Preferences** .

The **Preferences** dialog box appears.

2. Type **number** as the filter text to narrow down the list of preferences.
3. Select **Text Editors** . The **Text Editors** preference page appears in the right panel of the **Preferences** dialog box.
4. Check the **Show line numbers** checkbox.

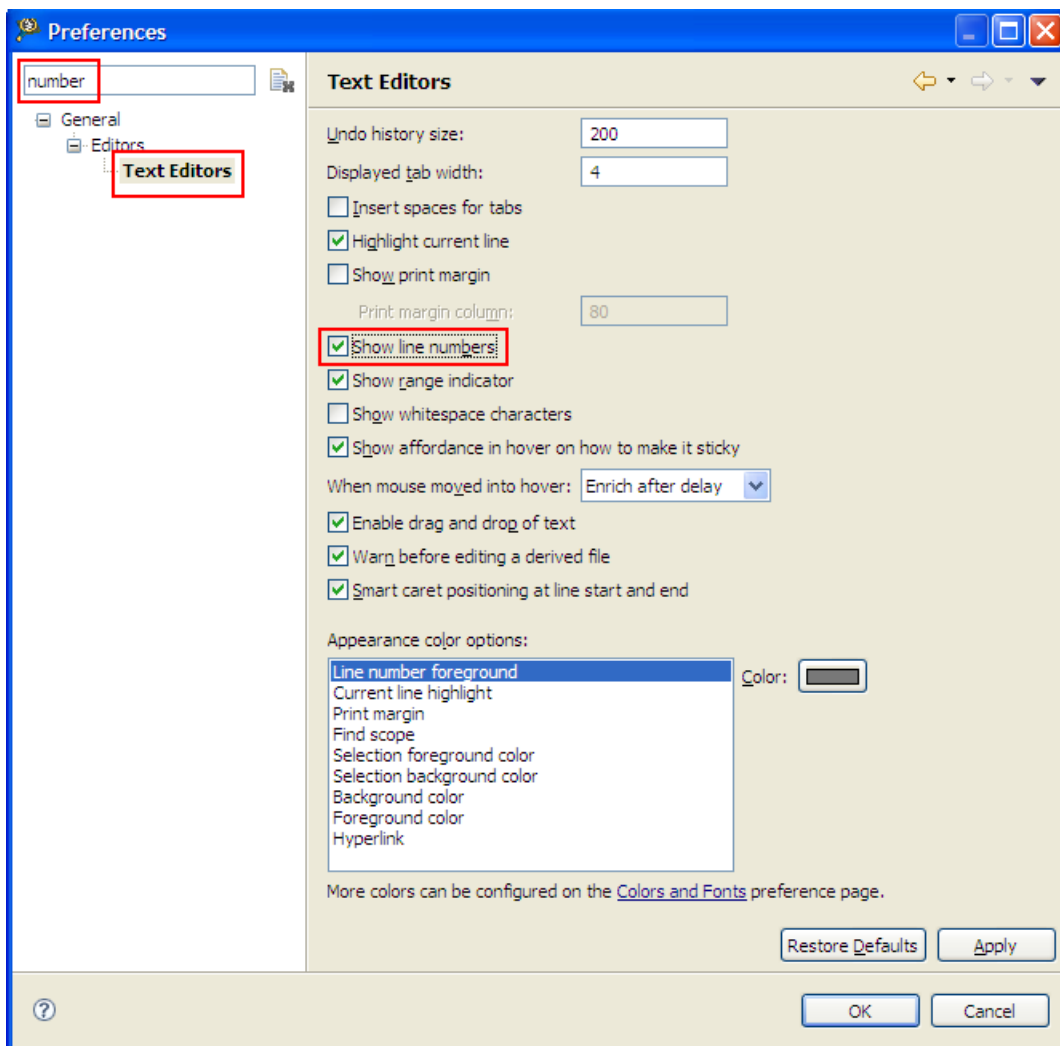


Figure 2-23. Text Editors Preference Page

5. Click **OK** .

The editor will display line numbers.

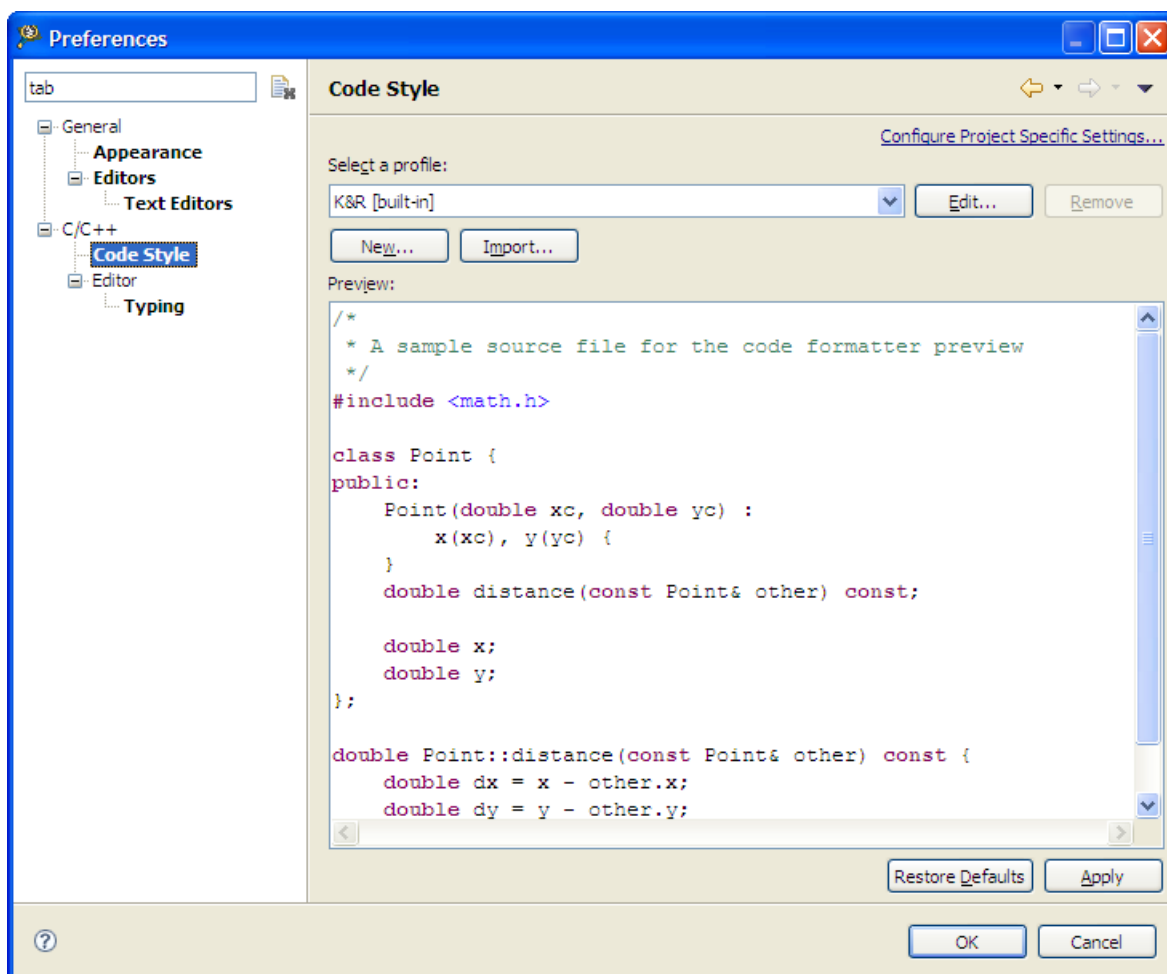
## 2.2.9 Is it possible to change the default code format settings?

Yes, to change the default code format settings:

1. From the IDE menu bar, select **Window > Preferences**.

The **Preferences** dialog box appears.

2. Type **code** as the filter text to narrow down the list of preferences.
3. Select **Code Style** in the left pane to modify the code formatting.



**Figure 2-24. Code Style Preference Page**

4. Select a predefined profile from the **Select a profile** drop-down list. Each of the profile specify different ways of formatting the code.

If the predefined profiles does not suit your requirements, you can create a new profile.

- a. Click the **New** button.

The **New Code Formatter Profile** dialog box appears.

- b. Enter the name of the new profile in the **Profile name** text box.
- c. Select the profile based on which you want to create the new profile from the **Initialize settings with the following profile** drop-down list.
- d. Click **OK**.

The edit dialog box appears.

- e. Specify the code formatting settings as required and click **OK**.
5. Click **OK** to close the **Preferences** dialog box.

The editor will now use the selected profile to format the code.

## 2.2.10 How can I quickly switch to a header file from within the source code file?

If you are working in an implementation (\*.c/\*.cpp) file, and you quickly want to open the corresponding header file, perform either of the following:

- press **CTRL+`** or
- right-click and select **Toggle Source/Header** .

Similarly, you can also toggle back to the source file from the header file.

## 2.2.11 How does the CodeWarrior IDE differentiate between enabled and disabled macros?

CodeWarrior IDE automatically marks the disabled macros in gray to help you determine which macros are active and which not.

```

58
59 #if PL_KBD_PULLUP_RESISTORS /* then check on bit cleared */
60 #define KBD_TEST_MACRO(keys, switch) (!testReg8Bits(keys, switch))
61 #else
62 #define KBD_TEST_MACRO(keys, switch) (testReg8Bits(keys, switch))
63 #endif
64
    
```

Figure 2-25. Disabled Macros Marked in Gray

## 2.2.12 How can I configure predefined macros in the CodeWarrior IDE?

To configure predefined macros in the CodeWarrior IDE:

1. Select the project for which you want to configure the predefined macros in the **CodeWarrior Projects** view.
2. Select **Project > Properties** in the IDE menu bar.

The **Properties** dialog box appears.

3. Select **C/C++ General > Paths and Symbols**.

The **Paths and Symbols** properties page opens in the right panel of the **Properties** dialog box.

4. Select the **Symbols** tab.
5. Configure the symbols listed in the **Symbols** tab page.
6. Click **OK**.

The predefined macros are configured for the selected project.

## 2.2.13 Is it possible to view evaluated expansion of a macro in the CodeWarrior IDE?

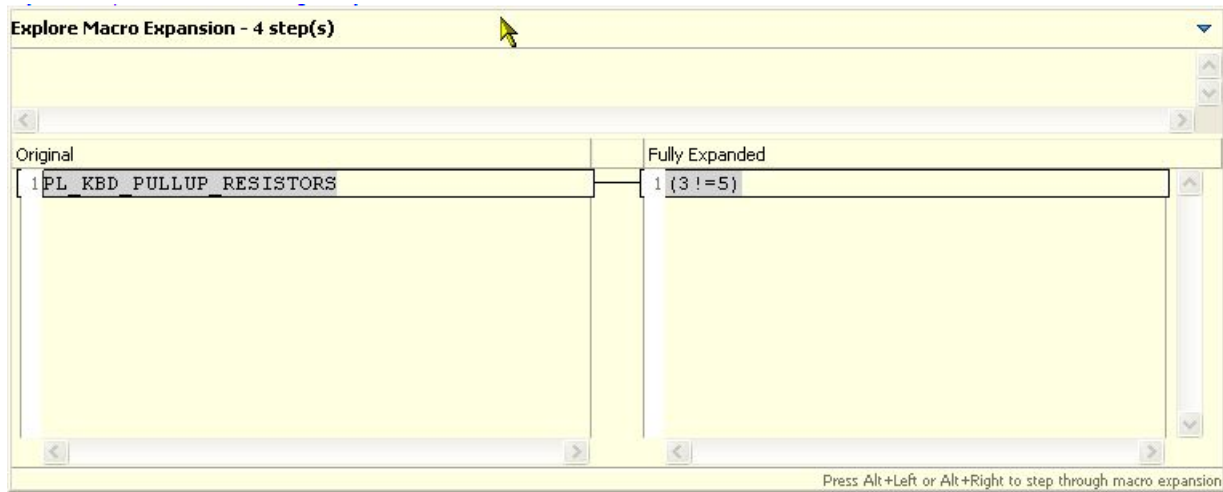
Yes, if you hover mouse cursor over a complex macro, the CodeWarrior IDE shows the evaluated expression (refer to the image listed below).

```

59 #if PL_KBD_PULLUP_RESISTORS /* then check on bit cleared */
60 #define KBD_TEST_MACRO(keys, switch) (!testReg8Bits(keys, switch))
61 #else
62 #define KBD_TEST_MACRO(keys, switch) (testReg8Bits(keys, switch))
63 #endif
    
```

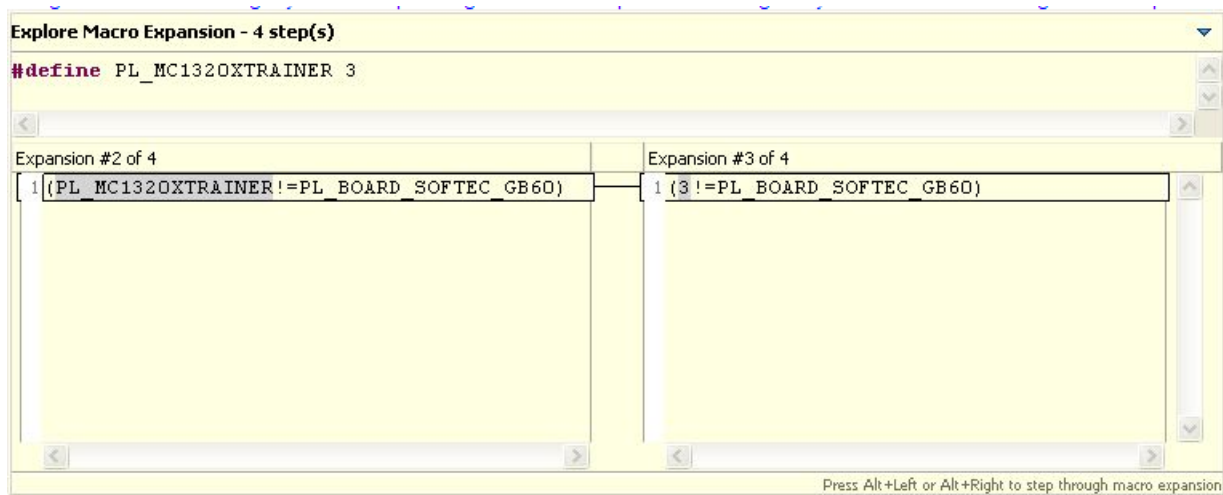
Figure 2-26. Pop-up Window Displaying Macro Expansion

Press F2 to bring in focus the macro expansion (refer to the image listed below).



**Figure 2-27. Focus over Macro Expansion**

Using **Alt+Left** and **Alt+Right** you can step through the macro expansion. This gives you details of the steps in the macro expansion (refer to the image listed below).



**Figure 2-28. Macro Expansion Steps**

## 2.2.14 Is it possible to roll back the changes I did to my source code?

Yes, local history can help roll back the changes you make to your source code.

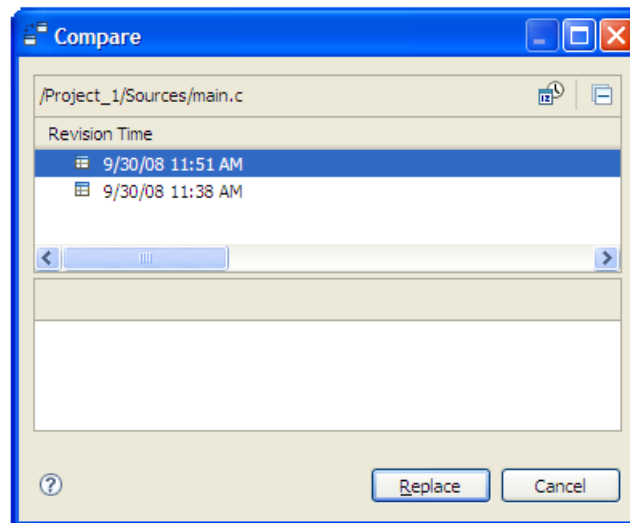
To roll back the changes, perform these steps.

1. Right-click the updated source code in the editor.



2. From the context menu, select the **Replace with > Local History** command.

The **Compare** dialog box appears. The dialog box lists the date and time of the changes you saved.



**Figure 2-29. Compare Dialog Box**

3. From the **Revision Time** options, select the desired date and time.
4. Click the **Replace** button.

The changes made to the file will be roll backed to the selected date and time.

#### **NOTE**

The local history is maintained for files only and not for projects or folders.

### **2.2.15 Is there a way to change the default color of comments that span across multiple lines?**

Yes, in order to change the default color of comments that span across multiple lines in source code follow steps below:

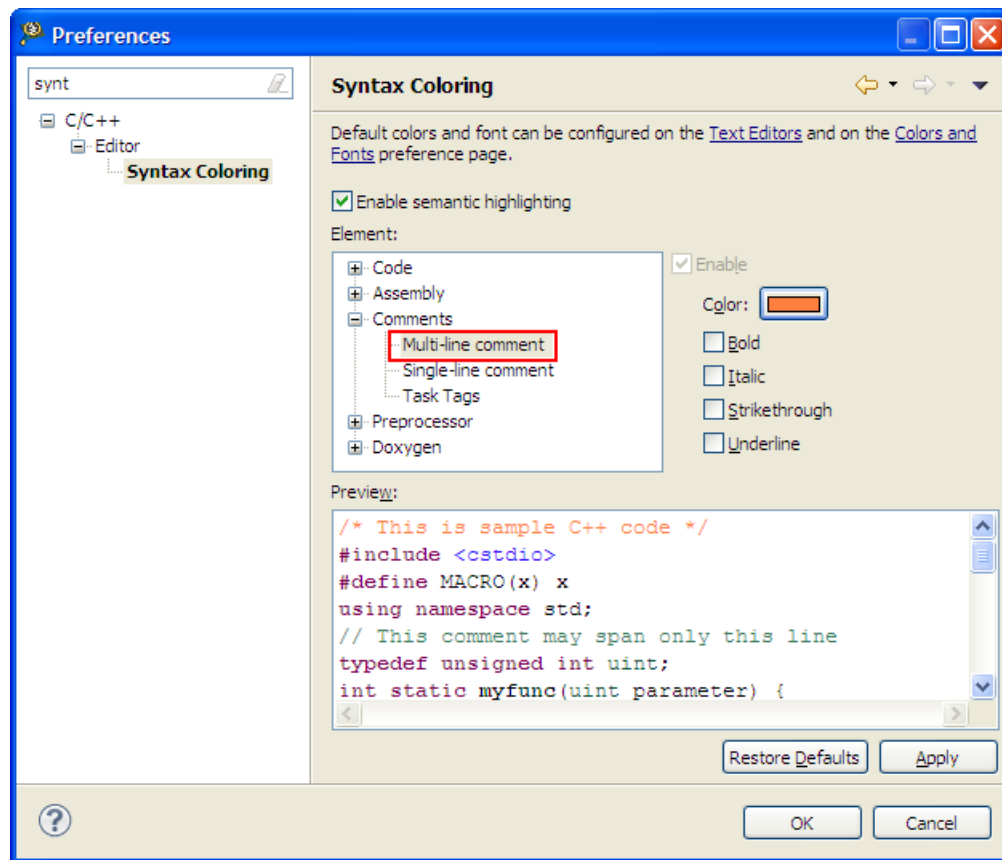
1. From C/C++ perspective toolbar, select **Preferences**.

The **Preferences** window appears.

2. Select **C/C++ > Editor > Syntax Coloring**.

The **Syntax Coloring** page appears in the right panel (refer to the image listed below).

### 3. Select **Comments > Multi-line comment**.



**Figure 2-30. Preferences Dialog Box - Syntax Coloring**

### 4. Select **Color**.

The color palette window appears.

5. From Color Palette, select a color of your choice.
6. Click **OK**.
7. Click **Apply**.
8. Click **OK**.

### **NOTE**

If you want to change the color of a comment that is across a single line, then follow the steps given above, except selecting **Single-line comment** instead of **Multi-line comment**.

The color of the comments in the source code changes, as specified.

## 2.2.16 Is there a way to revert to the original contents of a source code file?

Yes, if undesired changes were made to source file, there is a way to revert to the previous file. In order to do this follow steps below:

1. Right-click the source file and select **Replace With > Local History** from the context menu.

The **Compare** view appears.

2. Double-click **Revision Time** .
3. Double-click **Revision Time** tab.

The **Compare** window appears.

4. Select **Replace** .

The file is replaced.

### NOTE

Another way to replace the file is to select **Replace With > Previous from Local History** from context menu and this command replaces the file with the last file that you saved.

## 2.2.17 How do I convert Line Delimiters to the Unix format?

The Default is Windows. To change to UNIX, select **File > Convert Line Delimiters To > Unix** from the IDE menu bar.

## 2.2.18 Why do I get the following error message when I try to edit a source code file?

```
File '../..../filename.c' is read-only. Do you wish to make  
it writable?
```

You get this error message because the file that you are trying to edit is read-only. If you do not want to make it writable then select **No** . If you want to find out the settings for this file, follow steps below:

1. From the CodeWarrior Projects window, right-click on the source file.

A context menu appears.

2. From context menu, select **Properties** .

The **Properties** window appears.

3. Select **Resource** .

On the right-hand side, the settings for the file appear.

## 2.2.19 What does an asterisk on the Editor title bar mean?

The asterisk on the editor's toolbar indicates there are unsaved changes.

## 2.2.20 How can I find out where a function is declared in my source code?

To find out where a function is declared in source file, follow steps below:

1. Double-click on source file.

Source file appears in the **Editor** view.

2. In Source file, right-click the function name.

A context menu appears.

3. From context menu, select **Open Declaration** .

The source file that contains the function declaration appears.

## 2.2.21 How can I open .tcl extension files in the Editor window?

In order to open .tcl extension files in the Editor window, a File Association needs to be created. To create a File Association, follow steps below:

1. From the CodeWarrior toolbar, select **Window > Preferences**.

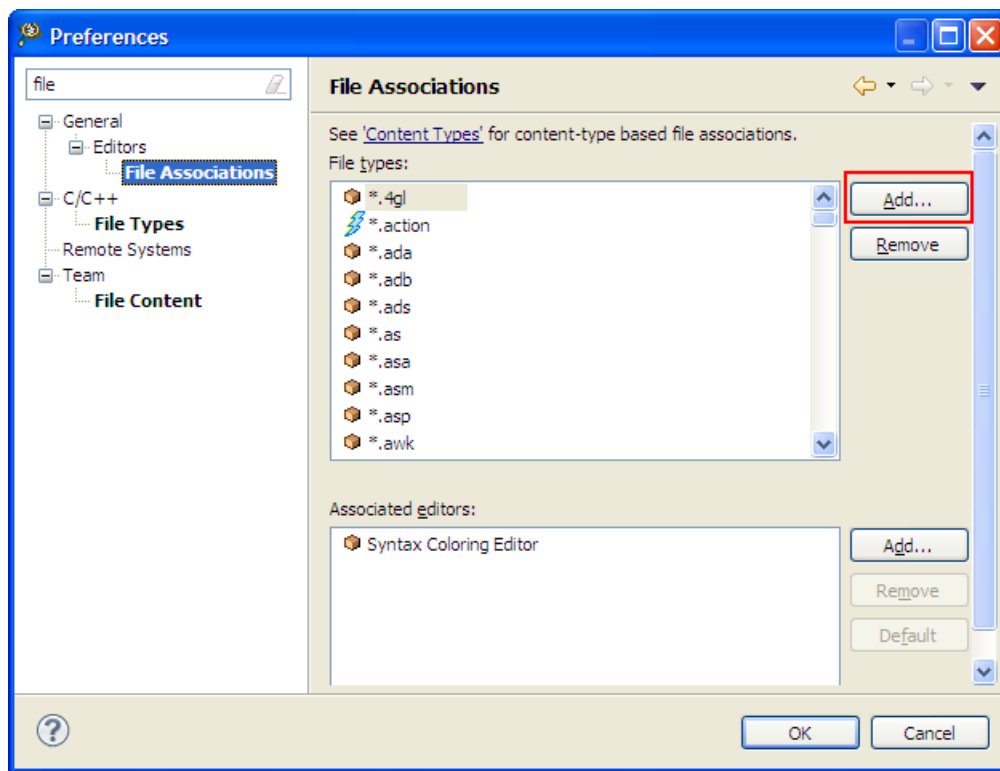
The **Preferences** dialog box appears.

2. Select **General > Editors > File Associations**.

The **File Associations** preferences appear on the right hand side.

3. Select **File Associations**.

The **File Associations** pane appears.



**Figure 2-31. Preferences Dialog Box - File Association**

4. Click **Add**.

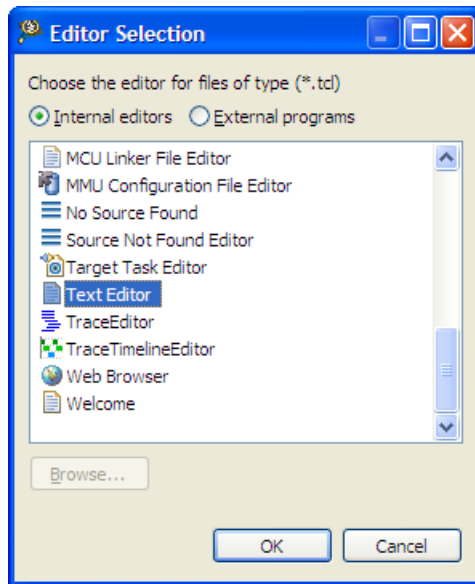
The **New File Type** dialog box appears.

5. In the **File Type** textbox, enter `.tcl`.
6. Click **OK**.

The `.tcl` extension appears under **File Types**.

7. Under **Associated editors** pane, Click **Add**.

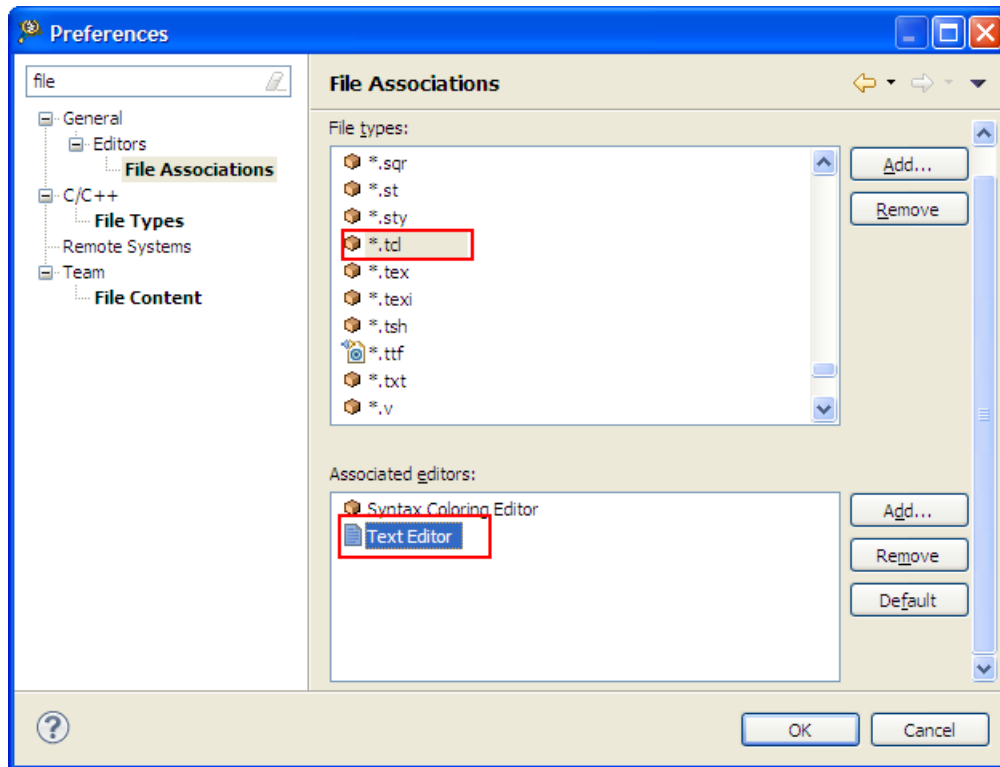
The **Editor Selection** dialog box appears.



**Figure 2-32. Editor Selection Dialog Box**

8. Click **OK**.

The `.tcl` extension file types get associated with a Text Editor (as the following figure shows).



**Figure 2-33. Preferences Dialog Box - File Association**

9. Click **OK**.

You can now open the `tcl` extension files in the Editor window.

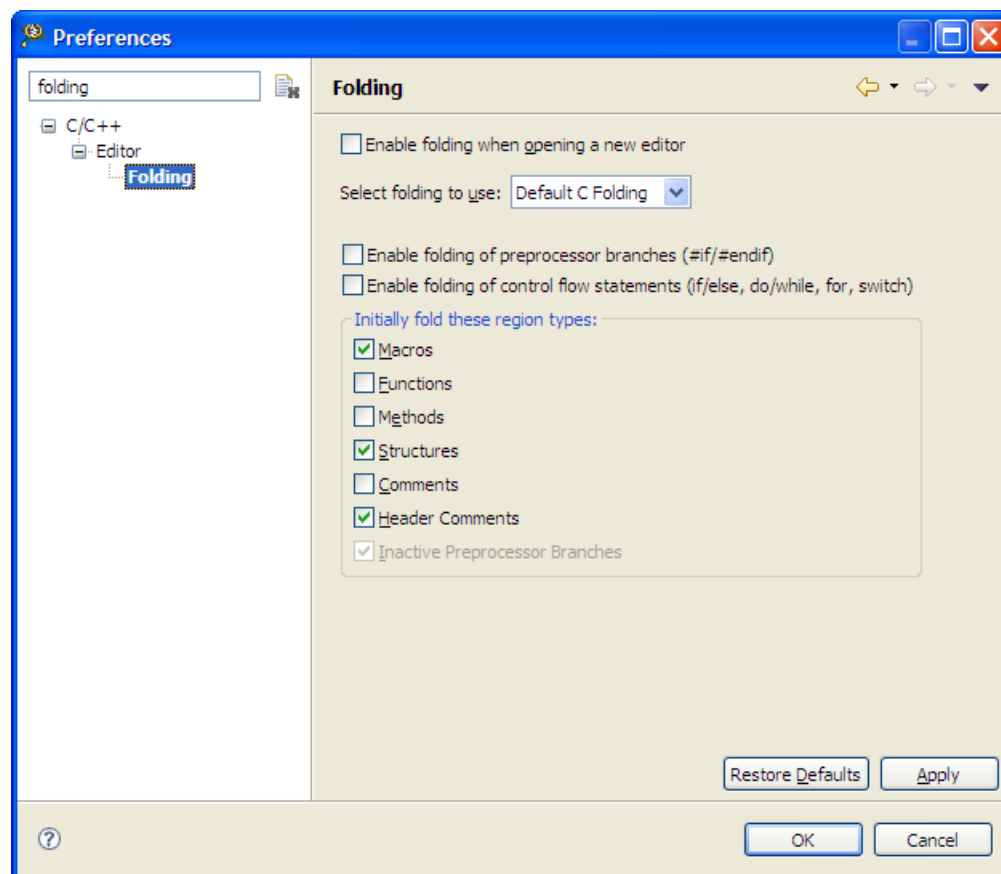
## 2.2.22 Is it possible to condense a large source file by folding/unfolding different portions of code?

Yes, you can enable folding in editor and specify levels of folding and which portion of code needs to be folded. To enable and configure folding/unfolding, perform these steps.

1. From the IDE menu bar, select **Window > Preferences**.

The **Preferences** dialog box appears.

2. Type folding as the filter text to narrow down the list of preferences.
3. Select **C/C++ > Editor > Folding**. The **Folding** preference page appears in the right panel of the **Preferences** dialog box.



**Figure 2-34. Folding Preference Page**

4. Check the **Enable folding when opening a new editor** checkbox.
5. Configure level of folding and select which regions of code should be automatically folded.
6. Click **OK** and reopen your source file.

Now you can fold/unfold portions in your source code, as the figure listed below shows. The fold/unfold functionality is added with new markers (+/-) at the left side in the source file. This improves readability of the source code.

```

main.c X
#include <hidef.h> /* for EnableInterrupts macro */
#include "derivative.h" /* include peripheral declarations */

-void InitSystems() {
    SOPT = 0x10;           // Disable COP,RSTO
    SPMSC1 = 0x00;        // Disable LVD
    MCGTRM = NVMCGTRM;    // Load Trim Values from flash
    MCGSC_FTRIM = NVFTRIM_FTRIM; // Load Fine Trim Value from fl
} // end InitSystems

// Initialize the FTM1CH3 for Output Compare (LED1)
-void FTM1C3_Output_Compare() {
    FTM1SC_CPWMS =0;      // Edge aligned
    FTM1C3SC_CH3IE = 1;   // Enable channel 3 interrupt
    FTM1C3SC_MS3x = 1;    // Output compare
    FTM1C3SC_ELS3x = 1;   // toggle output on compare
    FTM1C3V = 0x8000;     // Interrupt after 32,768 decimal cycles
                        // At 32.768Khz, 32,768 cycles yields in
    FTM1SC_PS = 7;        // 4.194304Mhz/128 = 32.768Khz
    FTM1SC_CLKSx = 1;     // bus clock and start the timer
} //end FTM1C3_Output_Compare

// Initialize the FTM1CH2 for Output Compare (LED2)
-void FTM1C2_Output_Compare() {
    FTM1SC_CPWMS =0;      // Edge aligned
    FTM1C2SC_CH2IE = 1;  // Enable channel 2 interrupt
}
    
```

**Figure 2-35. Editor With Folding Enabled**

To preview what is inside a folded region, you do not need to unfold it again. Hover your mouse cursor over a folded region and a popup appears displaying the details.

```

-void InitSystems() {
    void InitSystems() {
        SOPT = 0x10;           // Disable COP,RS
        // In
        SPMSC1 = 0x00;        // Disable LVD
    -void
        MCGTRM = NVMCGTRM;    // Load Trim Valu
        MCGSC_FTRIM = NVFTRIM_FTRIM; // Load Fine Trim
        // In
    } // end InitSystems
-void
    FTM1SC_CPWMS =0;         // Edge aligned
    }
    
```

**Figure 2-36. Folded Region Preview**

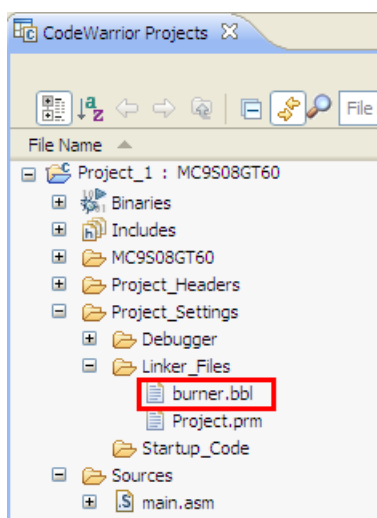


## 2.2.23 How can I generate S19, Intel Hex, and Binary files for my project?

For many embedded applications, you not only want the file to debug the target, that is generate \*.abs or \*.elf file, but you also need an S19 (Motorola S-Record) file, an Intel Hex file, or a Binary file of the application. Very likely you need these files for production programming or as input to other tools.

The wizard generated 8/16 bit projects for MCU10 includes a `burner.bbl` file for exactly this purpose. BBL stands for Batch Burner Language and is a simple batch/script language to generate and process S-Records, Intel Hex, and Binary files.

You can find the `burner.bbl` file within the `Project_settings\Linker_Files` folder in the project.



**Figure 2-37. Burner File**

The file includes a script to generate an S19 file.

### Listing: Burner file contents

```
OPENFILE "%ABS_FILE%.s19"  
format=motorola  
  
busWidth=1  
  
origin=0  
  
len=0x1000000  
  
destination=0  
  
SRECORD=Sx  
  
SENDBYTE 1 "%ABS_FILE%"
```

CLOSE

The script in burner file opens/creates a `.s19` file, configures some settings, such as the output file format, sends all bytes from the application `.abs` file to the output file, and then closes it.

The `*.bb1` file is processed by the make file after linking the application binary. There is a graphical way to configure or create such a script file.

### NOTE

For more information about the burner file, refer to the *Microcontrollers V10.x HC(S)08/RS08 Build Tools Utilities Manual*.

To configure burner script file graphically, perform these steps.

1. Browse to the `CWInstallDir\MCU\prog` folder.
2. Double-click `burner.exe`.

The **Burner Default Configuration** dialog box appears.

3. Click the **Burner Dialog** button on the toolbar (as the image, *Burner Default Configuration Dialog Box* listed below shows).

The **Burner** dialog box appears (as the image, *Burner Dialog Box - Input/Output Tab* listed below shows).

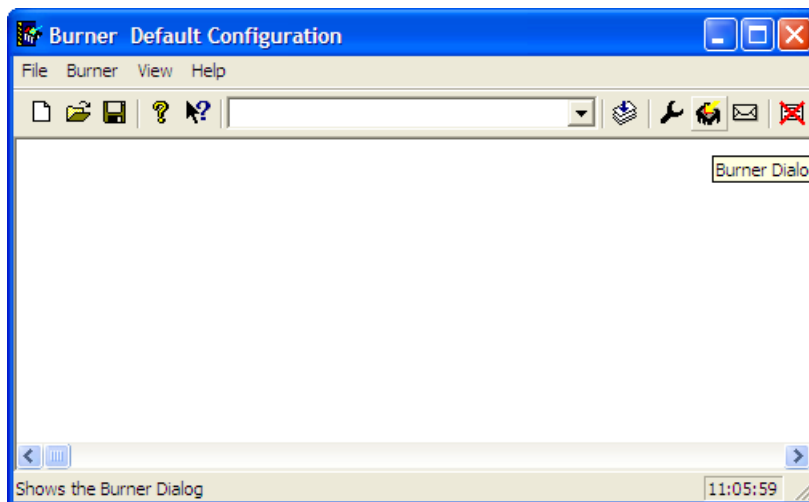
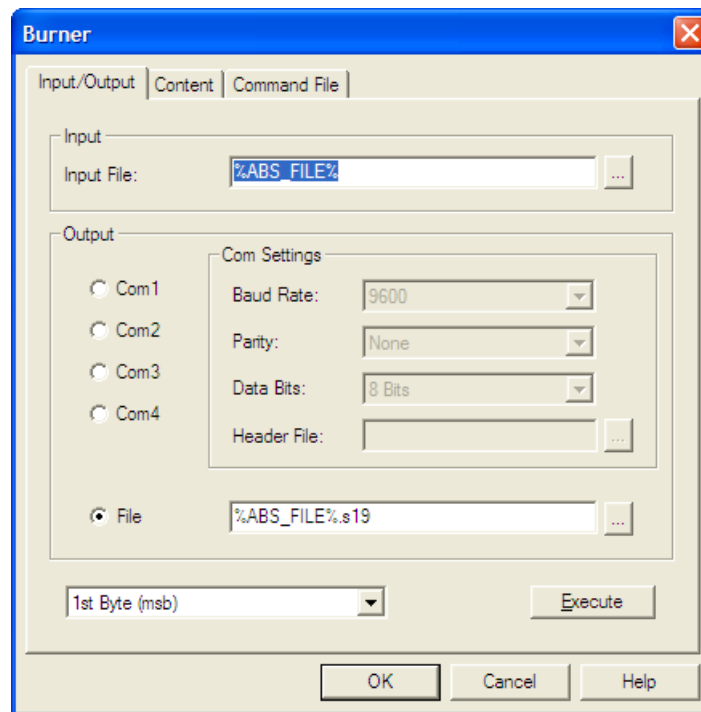
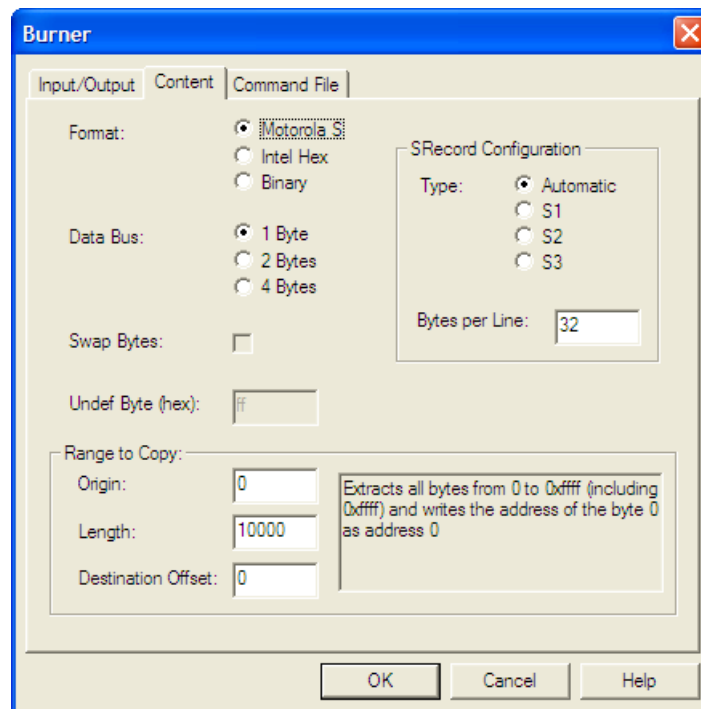


Figure 2-38. Burner Default Configuration Dialog Box



**Figure 2-39. Burner Dialog Box - Input/Output Tab**

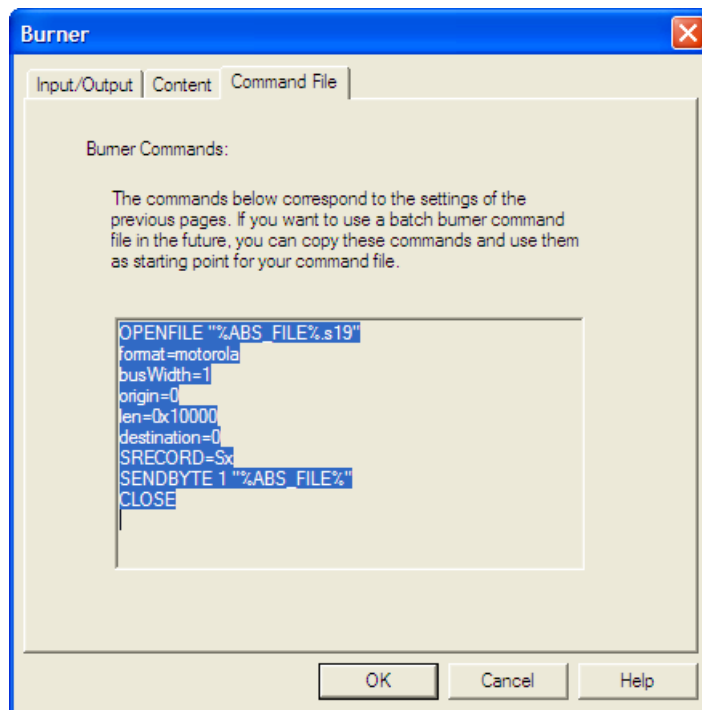
4. Configure the input file in the **Input/Output** tab.
5. Click the **Content** tab and configure the output format in the **Content** tab page.



**Figure 2-40. Burner Dialog Box - Content Tab**

6. Click the **Command File** tab.

The **Command File** tab page displays the script as per the settings done in the **Input/Output** and **Content** tabs.



**Figure 2-41. Burner Dialog Box - Command File Tab**

7. Copy the commands from the **Command File** tab page and paste the commands in your script file.

Now, you can easily create a `burner.bb1` file that can generate three different output files, S19, Intel Hex, and Binary, as shown in the following listing.

### Listing: Script for generating S19, Intel Hex, and Binary files

```
busWidth=1
origin=0

len=0x1000000

destination=0

SRECORD=Sx

undefByte=0xff

format=motorola

OPENFILE "%ABS_FILE%.s19"

SENDBYTE 1 "%ABS_FILE%"

CLOSE

format=binary

OPENFILE "%ABS_FILE%.bin"
```

```
SENDBYTE 1 "%ABS_FILE%"  
  
CLOSE  
  
format=intel  
  
OPENFILE "%ABS_FILE%.hex"  
  
SENDBYTE 1 "%ABS_FILE%"  
  
CLOSE
```

Also, the BBL file allows you to merge multiple files. For example, you can take a Binary file as input and convert it into an S-Record file. Or you add additional information to your files.

For example, if there is an S19 file that defines a firmware signature, and you want to make sure this signature is included into the S19 file for production.

You can implement the burner.bbl file as shown in the following listing.

### **Listing: Adding additional information to file using burner file**

```
OPENFILE "%ABS_FILE%.s19"  
format=motorola  
  
busWidth=1  
  
origin=0  
  
len=0x1000000  
  
destination=0  
  
SRECORD=Sx  
  
SENDBYTE 1 "%ABS_FILE%"  
  
SENDBYTE 1 "MyFirmwareSignature.S19"  
  
CLOSE
```

## **2.2.24 How can I use the Burner for the Kinetis derivatives?**

In the CodeWarrior for Microcontrollers v10.x, the Burner Utility integrated with RS08, S08 and ColdFire projects, but not for e200 yet. But you can use it, using a post-build step. **Post-build** is something similar as the **post-linker** in the classic CodeWarrior. It allows you to do something (whatever you want) at the end of the build process.

We are using it here for Kinetis to create an S19 file with the Burner utility after the file has been linked. To use the **Burner** for Kinetis derivatives:

- First, make sure that the Kinetis linker does not produce the S-Record, as we are going to generate it with the burner.
- Next, create or add your burner file to the project. The location or name does not matter, you just need to reference it properly in the next steps.

You might go with the default `burner.bbl` content as listed below:

```
OPENFILE "%ABS_FILE%.s19"
```

```
format=motorola
```

```
busWidth=1
```

```
origin=0
```

```
len=0x1000000
```

```
destination=0
```

```
SRECORD=Sx
```

```
SENDBYTE 1 "%ABS_FILE%"
```

```
CLOSE
```

Notice the "`%ABS_FILE%`" which is a variable to tell the burner, what file (`.elf/.afx/.abs`) is to use. You need to pass this to the burner.

In the Build Steps, define the Post-build steps, with something like this:

```
"${MCU_TOOLS_HOME}\prog\Burner.exe"
```

```
-F="{PROJECT_LOC}\Sources\burner.bbl"
```

```
-env"ABS_FILE=${BuildArtifactFileName}"
```

The first item specifies the executable, the `-F` option tells the burner which command file it has to use, and the `-env` option passes the environment variable used in the `.bbl`.

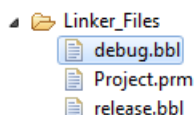
Now if you run a `make/build`, you can see the result in the **Console** view (and of course having the S19).

## 2.2.25 I have created two build configurations one for debug and another for release. How can I specify a burner bbl file for each build configuration?

Suppose, you have created two Build Configurations for a bareboard project, one for debug and another one for release, in CodeWarrior for Microcontrollers. You want to configure the build configurations such that each one can use a different burner bbl file. This way the IDE will generate different binaries depending on the active configuration.

To specify a burner bbl file for each build configuration:

1. Create two `.bbl` files.



**Figure 2-42. Burner Files**

2. Assign each `.bbl` file to the corresponding build configuration.
  - a. Right-click the file and select **Resource Configurations > Exclude from build**.

The **Exclude from build** dialog box appears.

- b. Check the build configurations from which you want to exclude the file.
- c. Click **OK**.

Alternatively, open the file properties dialog box. Check the **Exclude resource from build** checkbox.

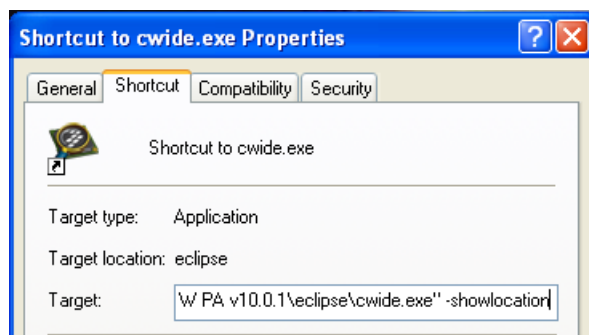
Similarly assign build configuration to the second bbl file.

Now, if you build the Debug configuration, the `debug.bb1` file will be used. And, if you build the Release configuration, the `release.bb1` file will be used.

## 2.2.26 Is there a way to determine in which workspace I am working currently?

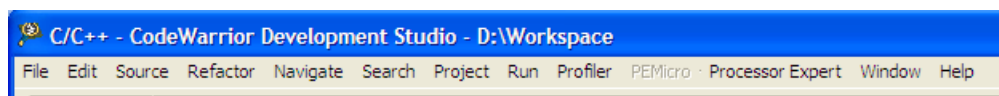
Yes, there is a command line switch to help you determine the location of the workspace your CodeWarrior IDE is using currently. You may need this information if you have multiple instances of CodeWarrior IDE open and if you switch workspaces frequently.

Add the `-showlocation` command line shortcut to the properties of the CodeWarrior IDE shortcut, as the following figure shows.



**Figure 2-43. Determine Workspace Location**

The current workspace location gets displayed on the title bar of your CodeWarrior IDE.



**Figure 2-44. Current Workspace Location**

## 2.2.27 How can I share same installation of the CodeWarrior software with multiple users?

By default, CodeWarrior IDE opens with the previously used workspace. However, you can configure the CodeWarrior IDE to open with the user-specific workspace, that is depending upon the user logged on to the system.

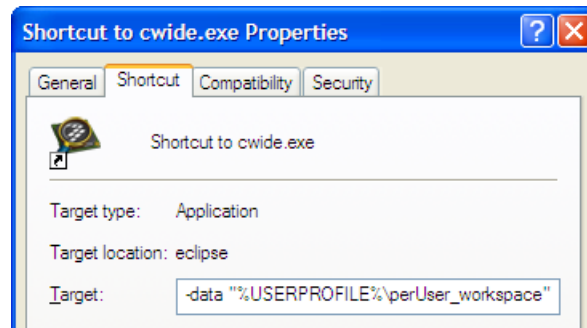
You can specify workspace path by passing the following command switch to the CodeWarrior IDE on startup.



```
-
data workspace path
```

If you want to open workspace folder per user basis, pass the following command line switch to the CodeWarrior IDE.

```
-dat
a "%USERPROFILE%\perUser_workspace
```



**Figure 2-45. Specify User-Specific Workspace**

### 2.2.28 How can I comment/uncomment a block of code in the editor?

In ANSI C, you use `/*` and `*/` for commenting a code line. For example:

```
/* this is a comment */
```

You can also span a comment over multiple code lines.

```
/*
do {
```

```

EVNT1_HandleEvent();

cnt--;

} while (EVNT1_EventsPending() && cnt>0);

*/

```

However, you cannot nest comments with `/* */` comments. CodeWarrior IDE marks it with ``?'` symbol.



In addition, most compilers, if not put into the strict ANSI C mode, also allow the usage of the `//` characters for commenting a code line.

```
// this is a comment too
```

This also allows you to comment out a line with comments.

```
// Function(); /* this is a call to my function*/
```

However, if you want to comment a whole code block, for example, the following code lines, commenting may not work for all the code lines in the block.

```

do {

EVNT1_HandleEvent();

cnt--; /* limit number of iterations */

} while (EVNT1_EventsPending() && cnt>0);

```

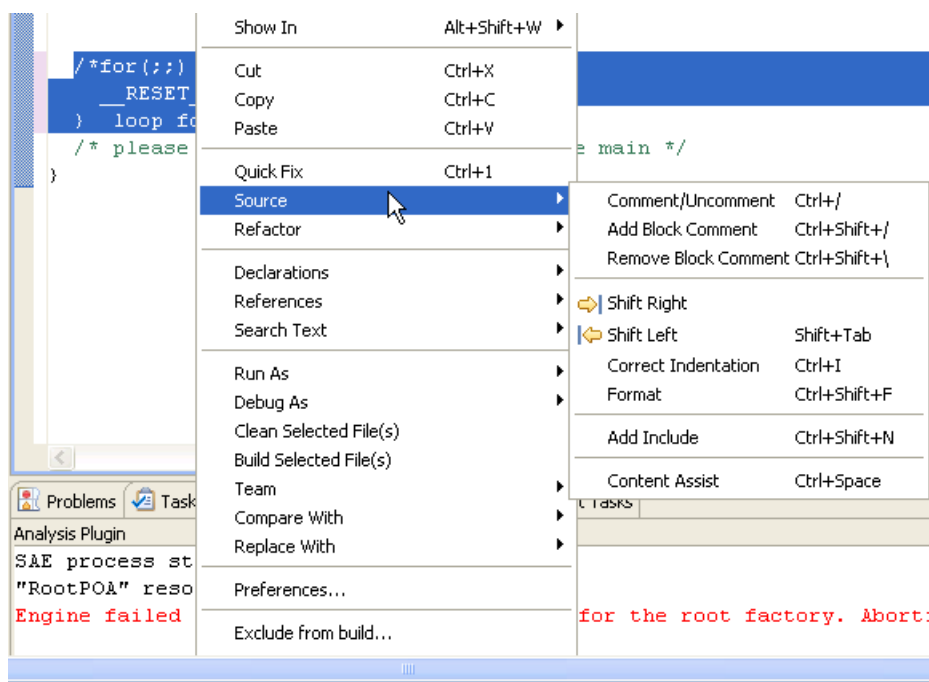
One way to comment a code block is to use #if 0 ... #endif.

```
#if 0
do {
    EVNT1_HandleEvent();
    cnt--; /* limit number of iterations */
} while (EVNT1_EventsPending() && cnt>0);
#endif
```

**Figure 2-46. Commenting Code Block**

The source file parser in CodeWarrior IDE automatically detects the `false' condition and grays out the code for better readability.

Another way is to comment/uncomment is to select the code lines you want to comment/uncomment and right-click. The following figure shows the context menu that appears.



**Figure 2-47. Editor Context Menu**

Select **Source > Comment/Uncomment** from the editor context menu. The selected code lines are commented using the // characters.

```
cnt = 16;

//do {
```

```
//EVNT1_HandleEvent();

//cnt--; /* limit number of iterations */

//} while (EVNT1_EventsPending() && cnt>0);

} else {
```

Similarly, you can also uncomment the code block.

To comment code block using the `/* */` characters and uncomment, use **Add Block Comment** and **Remove Block Comment** in the editor context menu.

```
cnt = 16;

/*

do {

EVNT1_HandleEvent();

cnt--; limit number of iterations

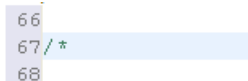
} while (EVNT1_EventsPending() && cnt>0);

*/

} else {
```

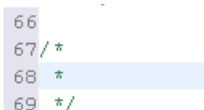
But note that, nested `/* */` comments are not allowed. The CodeWarrior IDE removes the nested comment tokens, as shown in the example above. You need to check the comments in the code block after applying comment using the `/* */` characters.

In addition, CodeWarrior IDE editors include comment auto-templates. For example, if you type `/*` in the editor and press enter, editor automatically creates a comment block for you.



```
66
67 /*
68
```

Figure 2-48. Comment Block



```
66
67 /*
68
69 */
```

You can immediately start typing your comment text inside the comment block.

To configure comment template settings:

1. Select **Window > Preferences** .

The **Preferences** dialog box appears.

2. Type "comment" as the filter text to narrow down the list of the preferences.
3. Select **Code Style > Code Templates**.
4. Make the required settings in the **Code Template** preference panel and click **OK** .

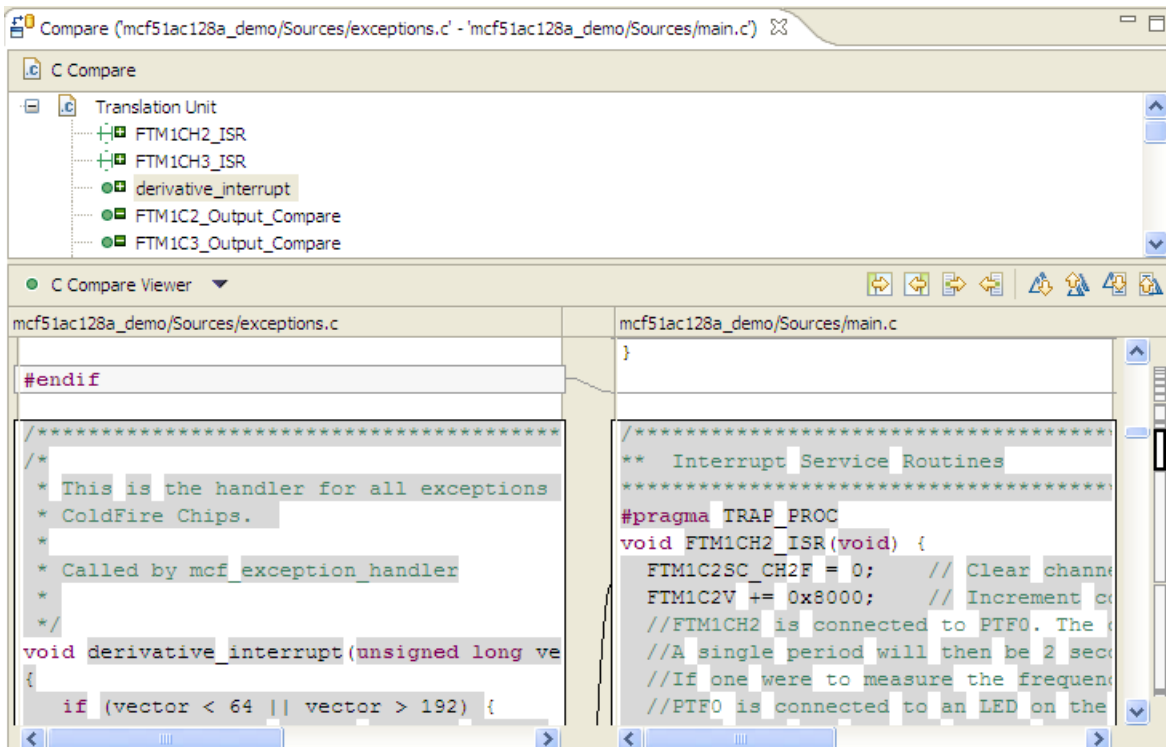
The template settings are configured, as specified.

## 2.2.29 How can I compare files in CodeWarrior IDE?

To compare files:

1. Hold the CTRL key and select the files you want to compare.
2. Right-click and select **Compare With > Each Other**.

The **Compare** view appears.



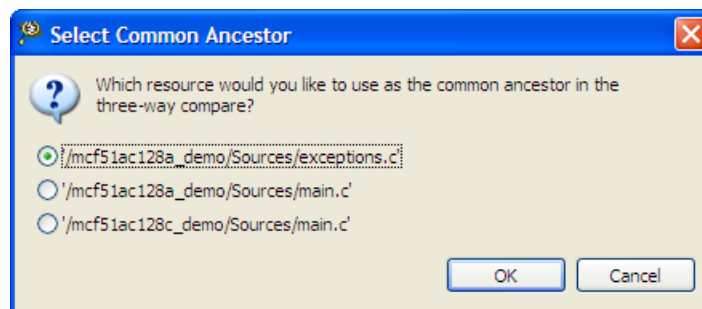
**Figure 2-49. Compare View**

The **C Compare** area displays the list of items added or removed from the files. The items are marked with specific addition and deletion icons. Double-click an item to navigate to it in the files.

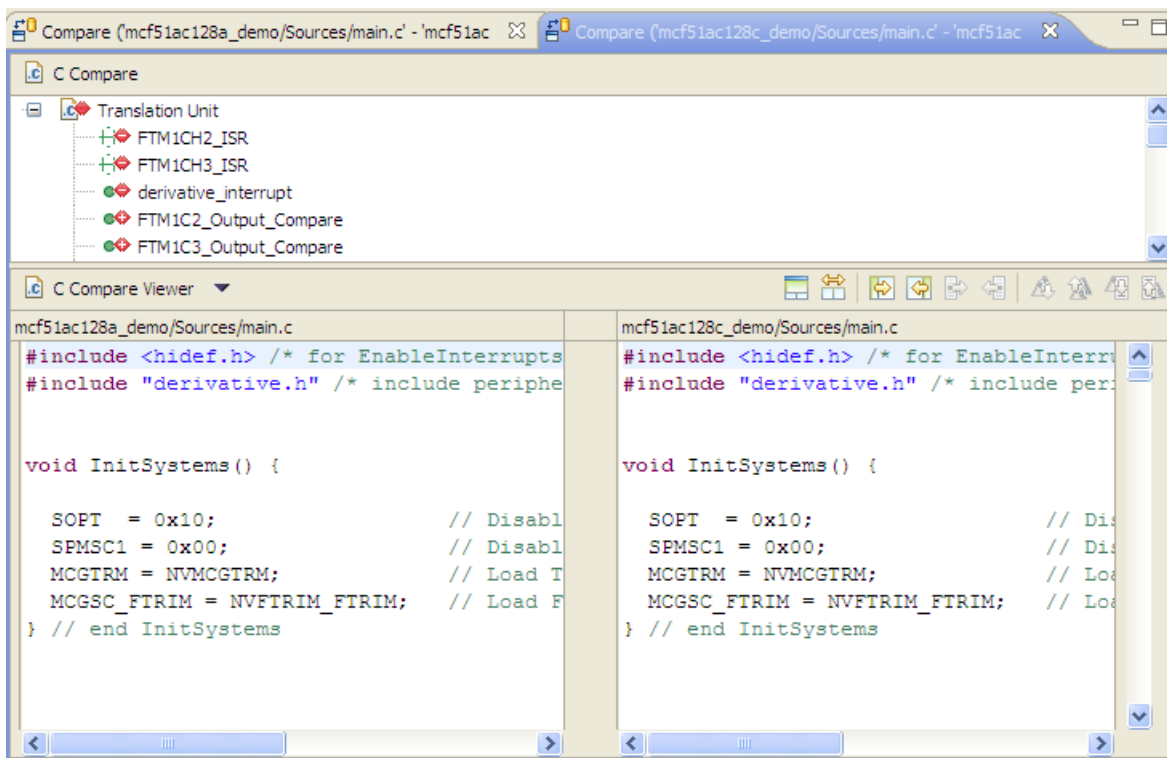
Using the toolbar, you can:

- Copy all changes, or only the not conflicting ones
- Copy the current change to left or right
- Go to the next or previous change or difference

Similarly, you can also compare more than two files in CodeWarrior IDE. To compare more than two files, you need to specify which file should be used as the common ancestor.



**Figure 2-50. Specify Common Ancestor File**



**Figure 2-51. Compare More Than Two Files**

Using the **Show Ancestor Pane** and **Two-Way Compare (Ignore Ancestor)** toolbar buttons, you can display or close the **Ancestor** view.

### 2.2.30 Is there a way to compare files that are not in project?

To compare files that are not in a project, you need to use an external diff tool, for example KDiff, which is an open source tool.

1. Download and install KDiff from the following location:

<http://kdiff3.sourceforge.net/>

2. Download the plugin file from the following location:

<http://externaldiff.sourceforge.net/>

3. Select **Window > Preferences** to configure the tool.
4. Type `compare' as the filter text and select **Compare/External Tool** panel.
5. Browse to the location where KDiff is installed and select **kdiff.exe** .
6. Click **OK** .
7. Select the files you want to compare using KDiff.
8. Right click and select **Compare With > Each Other (external tool)** .

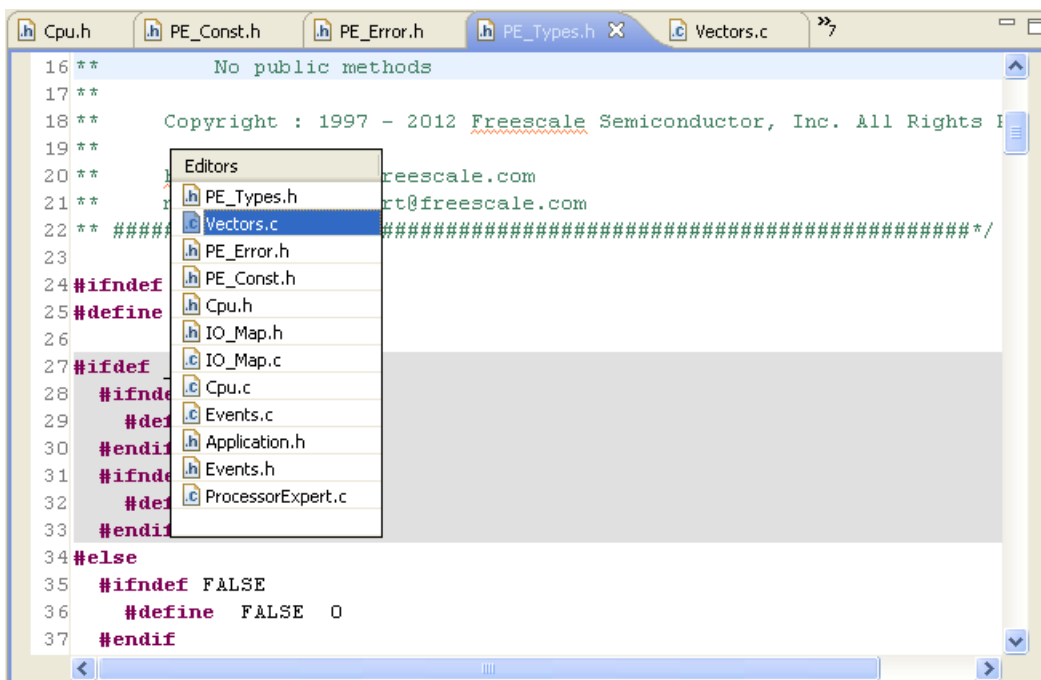
The result of comparison appears in the KDiff editor.

### 2.2.31 How do I switch between the source files in the Editor view?

To switch between the source files in the Editor view while creating your project:

1. Press **Ctrl+Tab** .

A small pop-up window appears listing all open source files in the Editor view.



**Figure 2-52. Switching Between Source Files in Editor View**

2. Press **Ctrl+Tab** repeatedly to iterate through the source files list.

By using this shortcut, you can quickly swap between different source files.

### 2.2.32 I want to edit my source files, but all the other views are using the space as well. How do I maximize the editor view to have better working area?

To maximize the views in the CodeWarrior IDE, just double-click on the view tab to toggle between maximized view and the normal view. Alternatively, right-click on the view tab and select **Maximize** from the context-menu to maximize the view. Similarly, right-click on the maximized view tab select **Minimize** from the context-menu to return to the normal view.



## 2.2.33 How do I remove blank lines in the Editor view?

You can remove the empty or blank lines in the Editor view by following any of these methods:

- By using the keyboard shortcut

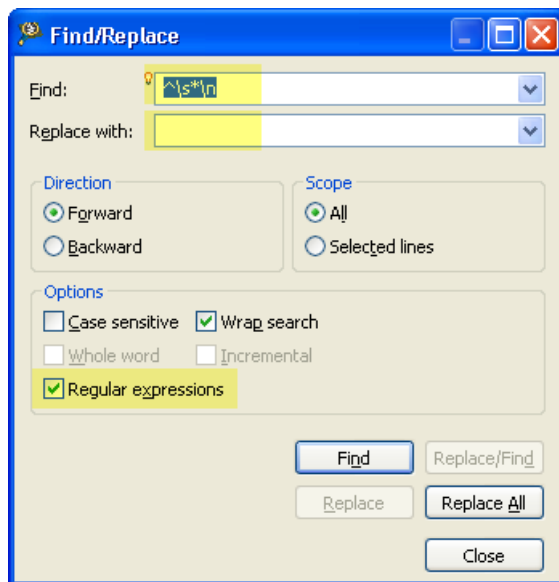
Place the cursor on a line and press **Ctrl+D** (for line Delete). You can also use **Ctrl+D** to delete multiple selected line.

### Tip

To get more commands for *deleting* the blank lines, press **Ctrl+3**, enter *delete* in the textbox, the related commands with their keyboard shortcut appears in the **Commands** list.

- By using the search and replace method
  - a. Select **Edit > Find/Change** from IDE menu bar, alternatively press **Ctrl+F**.

The **Find/Replace** dialog box appears.



**Figure 2-53. Find/Replace Dialog Box**

- b. Enter **^\s\*\n** in the **Find** textbox and leave the **Replace with** textbox empty.
- c. Check the **Regular expressions** checkbox.
- d. Click **Replace All**.

The editor removes the blank lines.

## 2.3 Workbench Window

In this topic, Workbench window related FAQs are listed.

- [What controls the initial layout of a view in a Workbench window?](#)
- [Is it possible to restrict the number of resources that appear on my Workbench window?](#)
- [How do I minimize a Workbench window?](#)
- [How do I customize my Workspace perspectives?](#)
- [How do I enable full screen mode in my CodeWarrior IDE?](#)
- [How do I hide the toolbar from the IDE for better visibility of other views that I am currently working with?](#)
- [How can I view all keyboard shortcuts available in the CodeWarrior IDE?](#)
- [How do I edit the keyboard shortcuts in the CodeWarrior IDE?](#)

### 2.3.1 What controls the initial layout of a view in a Workbench window?

A perspective defines the initial layout of the views in a workbench window. One workbench window contains many perspectives. Each of the perspectives is task oriented.

### 2.3.2 Is it possible to restrict the number of resources that appear on my Workbench window?

Yes, working sets help reduce the number of resources displayed. A working set is a group of elements you want to display.

To create or define a new working set, perform the following steps.

1. Click  in the **CodeWarrior Projects** view or **Project Explorer** view toolbar.

The pull-down menu appears.

2. Select the **Select Working Set** command.

The **Select Working Set** dialog box appears.

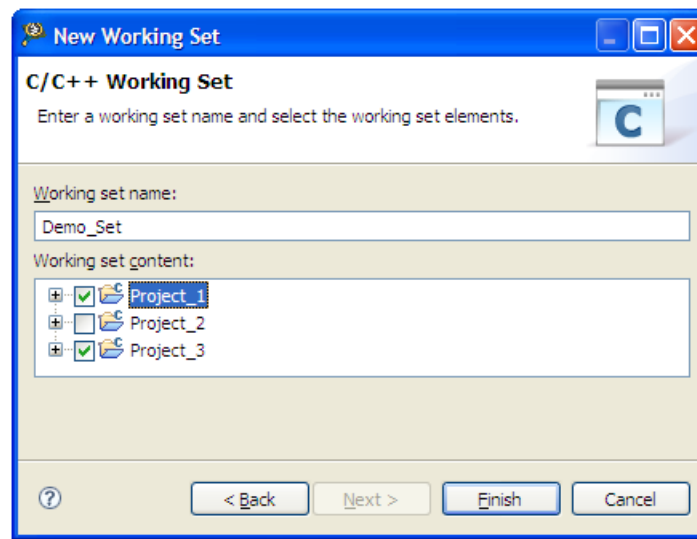
3. Click the **New** button.

The **New Working Set** wizard appears.

4. From the **Working set type** options, select the appropriate working set. For example, select `C/C++`.
5. Click **Next**.

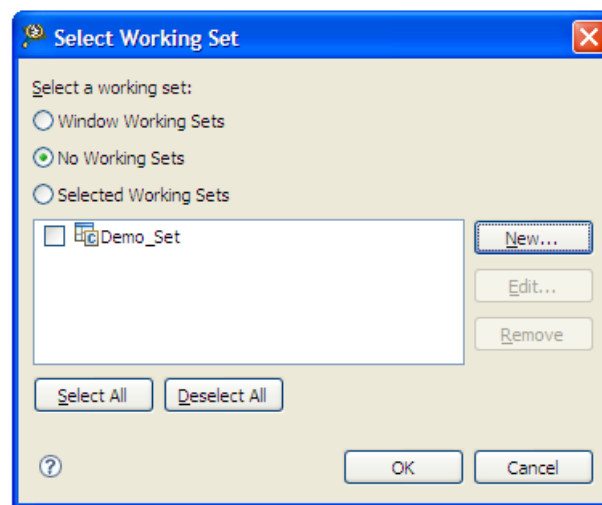
The **<selected> Working Set** page appears. For example, `C/C++ Working Set`.

6. In the **Working set name** field, enter an appropriate name for the working set. For example, `Demo_Set`.
7. In the **Working set content** tree, check the items you want in your view. For example, check `Project_1` and `Project_3` (refer to the image listed below).



**Figure 2-54. New Working Set Wizard - C/C++ Working Set Page**

8. Click **Finish** to close the **New Working Set** wizard.
9. The **Select Working Set** dialog box reappears with the newly created working set.



**Figure 2-55. Select Working Set Dialog Box**

10. Check the checkbox corresponding to the desired working set and click **OK**.

The **CodeWarrior Projects** view or **Project Explorer** view will now display the selected resources only.


#### **NOTE**

To deselect an active working set, select the **Deselect Working Set** command from the view pull-down menu.

#### **NOTE**

To edit an active working set, select the **Edit Active Working Set** command from the view pull-down menu.

### **2.3.3 How do I minimize a Workbench window?**

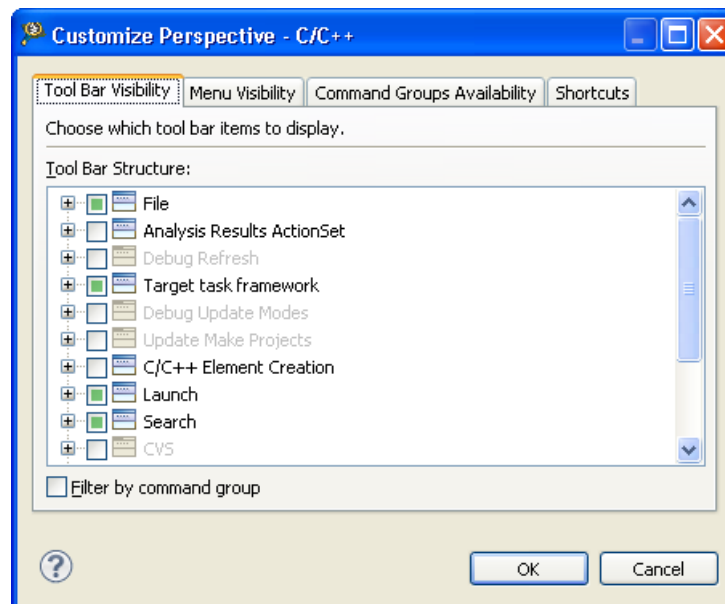
When initiated for the first time, the Workbench window tends to appear in a peculiar state. The window seems to stretch to fit the entire screen but might not be maximized. This generally happens, because the window takes the dimensions of the last application window you opened. To minimize such a window, ensure that the **Maximize** button indicates an unmaximized window. Next, drag the window edge (the cursor will appear as a left-right arrow ) to the desired width.

### **2.3.4 How do I customize my Workspace perspectives?**

To customize your Workspace perspectives:

1. Create a new workspace. This way you have a clean start for customization and no changes from an existing modified workspace.
2. Close the views that you do not want to use for your project manually.
3. For customizing the buttons and menus visibility in the desired perspective:
  - a. Select **Window > Customize Perspective** from the IDE menu bar.

The **Customize Perspective - <perspective name>** dialog box appears.



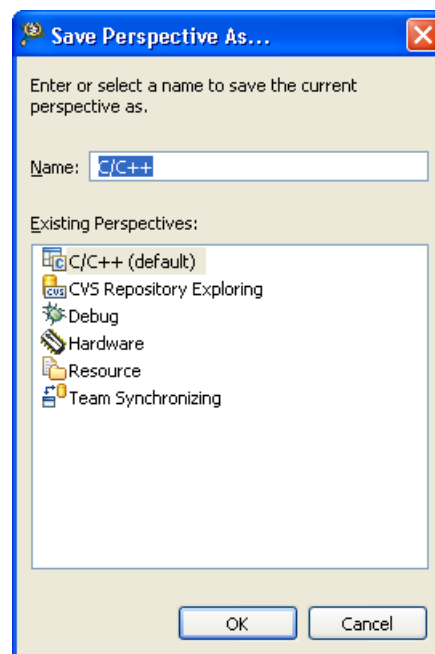
**Figure 2-56. Customize Perspective - <perspective name> Dialog Box**

- b. Click the desired tab to select the desired settings for tool bar, menu, command groups and shortcuts visibility.
- c. Click **OK**.

The **Customize Perspective - <perspective name>** dialog box closes.

4. To save the changes you have made to the IDE's perspective, select **Window > Save Perspective As** from the IDE menu bar.

The **Save Perspective As** dialog box appears.



**Figure 2-57. Save Perspective As Dialog Box**

5. Select **OK**.
6. The **Overwrite Perspective** dialog box appears.
7. Click **Yes** to overwrite the previous default setting with the changes you made.

### 2.3.5 How do I enable full screen mode in my CodeWarrior IDE?

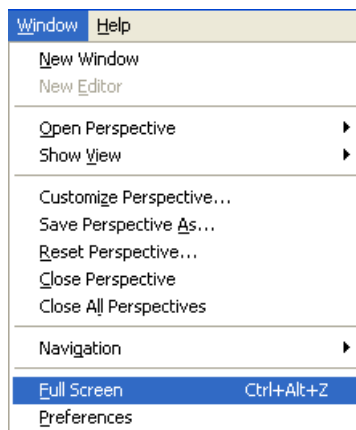
To change your CodeWarrior IDE to full screen mode:

1. The Eclipse full screen plugin is available at <http://code.google.com/p/eclipse-fullscreen/> web address.
2. Download the zip file.
3. Extract the `cn.pande.eclipsesx.fullscreen_<version>.jar` file from the zip file you have downloaded.
4. Copy the extracted jar file in the `<CWInstallDir>\CW MCU v10.x\eclipse\plugins` folder, where `<CWInstallDir>` is the installation directory of your CodeWarrior software.

The full screen mode option is now available in your CodeWarrior Eclipse IDE.

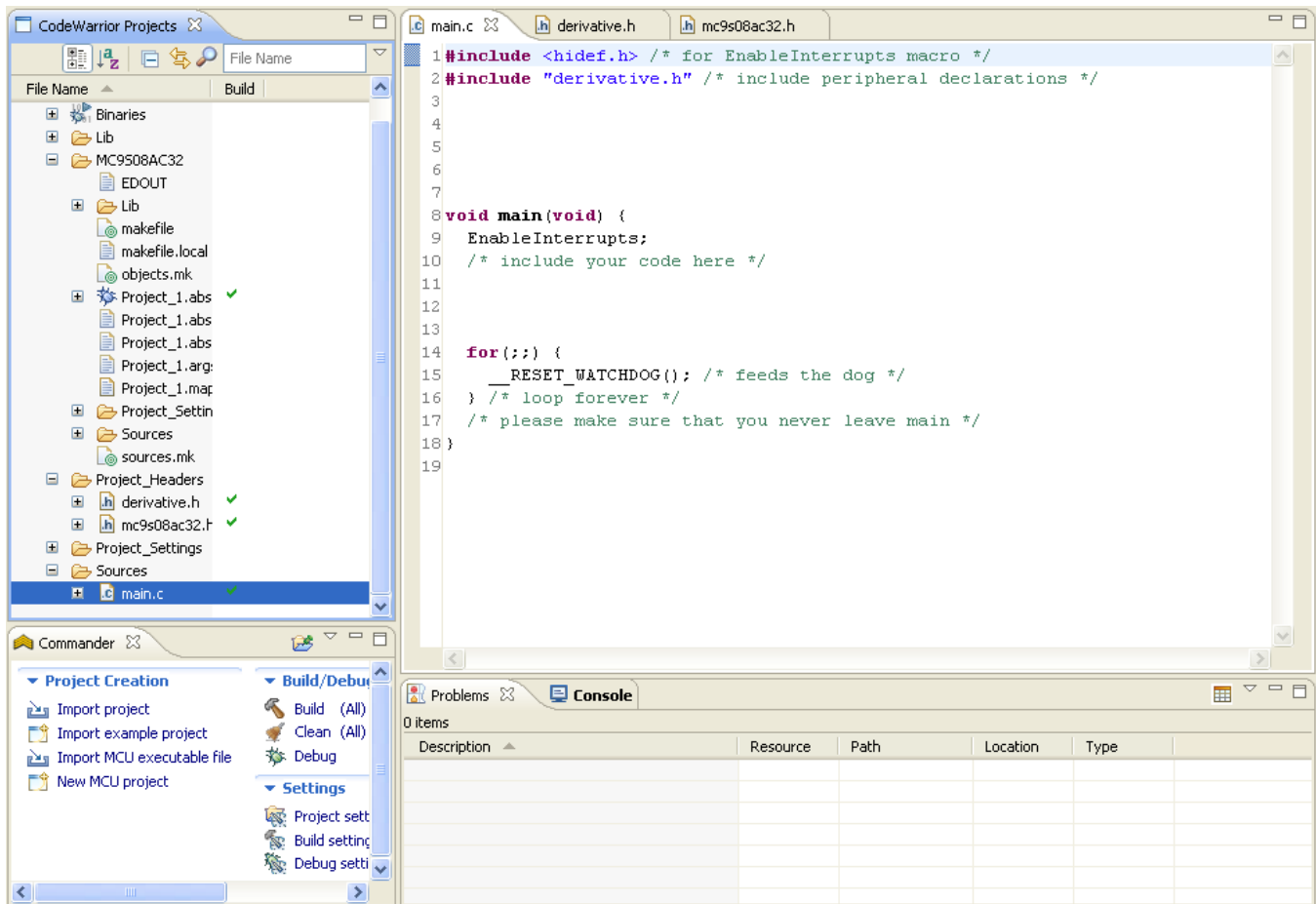
To use the full screen mode feature:

1. Open the CodeWarrior IDE.
2. Select **Window > Full Screen** from the IDE menu bar, alternatively, press **Ctrl+Alt+Z** to switch to the full screen mode.



**Figure 2-58. Full Screen Menu Option**

The CodeWarrior Eclipse IDE switches to full screen mode, hiding the toolbar, menu bar and status bar from the screen.



**Figure 2-59. CodeWarrior Eclipse IDE - Full Screen Mode**

3. Press **Esc** , or **Ctrl+Alt+Z** to exit the full screen mode.

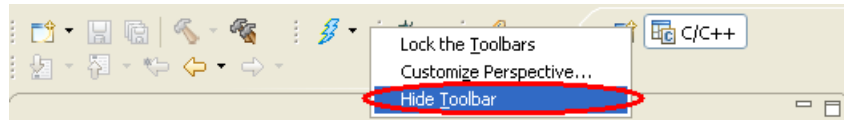
The CodeWarrior Eclipse IDE switches back to the normal screen mode.

### NOTE

The **Ctrl+Alt+Z** is the default shortcut to toggle between *normal* and the *full screen* mode for the CodeWarrior Eclipse IDE.

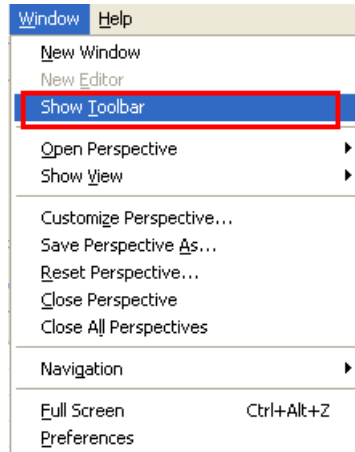
## 2.3.6 How do I hide the toolbar from the IDE for better visibility of other views that I am currently working with?

To hide the toolbar from the IDE, right-click on the toolbar and select **Hide Toolbar** from the context-menu. The following figure shows the IDE toolbar context-menu:



**Figure 2-60. Hide Toolbar - Toolbar Context-menu Option**

To display the toolbar again in the IDE, select **Window > Show Toolbar** from the IDE menu bar. The following figure shows the **Show Toolbar** menu option:



**Figure 2-61. Show Toolbar - Menu Option**

### 2.3.7 How can I view all keyboard shortcuts available in the CodeWarrior IDE?

To view all keyboard shortcuts available in the CodeWarrior IDE:

1. Open the CodeWarrior IDE.
2. Press **Ctrl+Shift+L** .

The drop-down list containing the commands and their keyboard shortcuts appears in the IDE.



Activate Editor	F12
Add Block Comment	Ctrl+Shift+/,
Add Include	Ctrl+Shift+N
Backward History	Alt+Left
Build All	Ctrl+B
Build Project	Alt+B, B
Clean All Projects	Alt+B, L
Close	Ctrl+W
Close All	Ctrl+Shift+W
Collapse	Ctrl+Numpad_Subtract
Collapse All	Ctrl+Shift+Numpad_Divide
Comment/Uncomment	Ctrl+/,
Content Assist	Ctrl+Space
Context Information	Ctrl+Shift+Space
Copy	Ctrl+C
Copy Lines	Ctrl+Alt+Down
Cut	Ctrl+X
Debug	Alt+B, D
Delete	Delete
Delete Line	Ctrl+D
Delete Next Word	Ctrl+Delete
Delete Previous Word	Ctrl+Backspace
Delete to End of Line	Ctrl+Shift+Delete
Duplicate Lines	Ctrl+Alt+Up
Expand	Ctrl+Numpad_Add
Expand All	Ctrl+Numpad_Multiply
Explore Macro Expansion	Ctrl+=

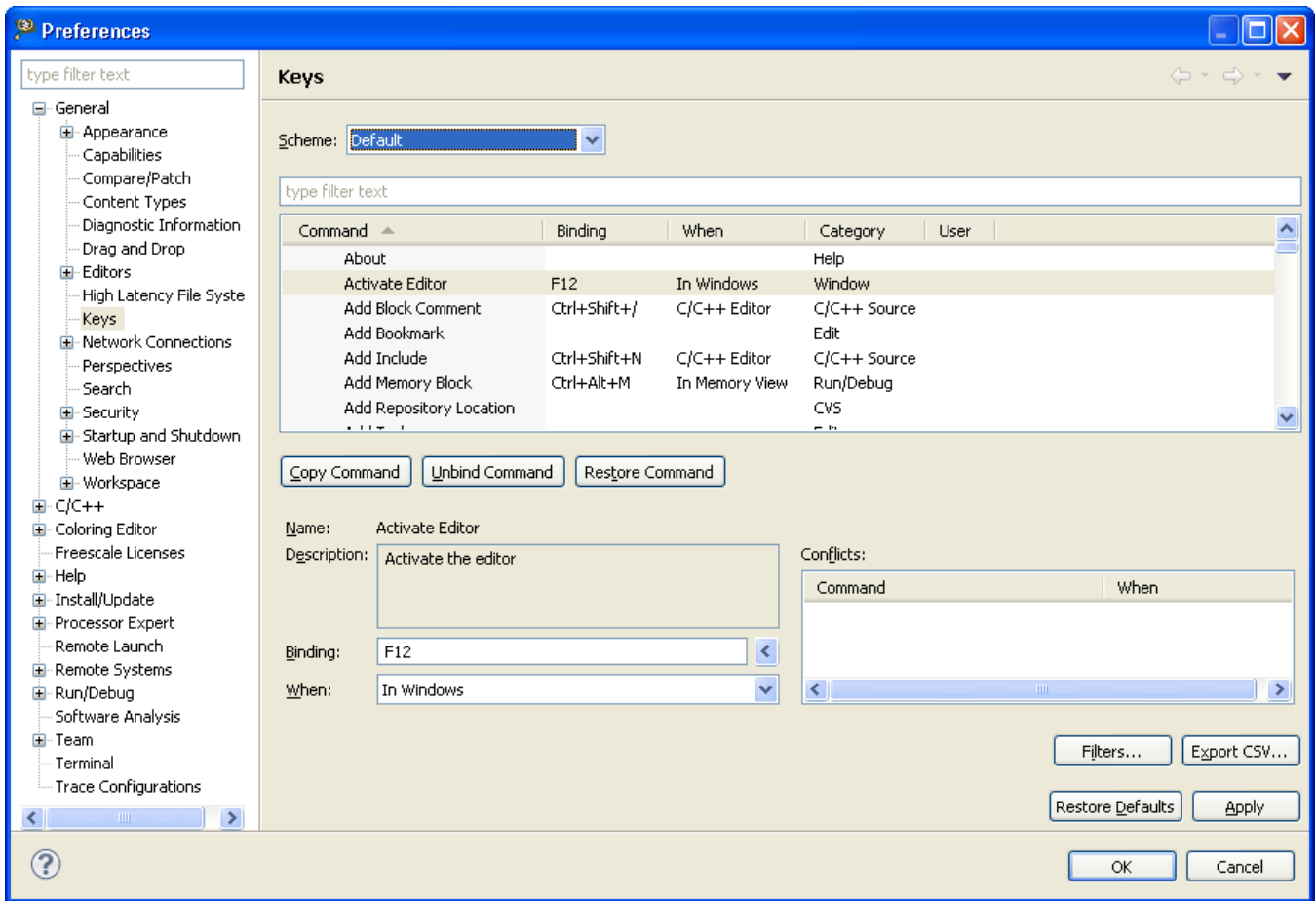
Press "Ctrl+Shift+L" to open the preference page.

**Figure 2-62. Keyboard Shortcuts Drop-down List**

### 2.3.8 How do I edit the keyboard shortcuts in the CodeWarrior IDE?

To edit the keyboard shortcuts available in the CodeWarrior IDE:

1. Press **Ctrl+Shift+L** twice.
2. The **Preferences** dialog box with the **Keys** page appears.



**Figure 2-63. Preferences Dialog Box - Keys Page**

3. Select the desired command.
4. Click the **Copy Command** , **Unbind Command** , and **Restore Command** buttons to copy, unbind or restore the selected command respectively.
5. Click **Apply** .
6. Click **Ok** .

The IDE saves and applies the changes you made.

## 2.4 CVS

In this topic, CVS related FAQs are listed.

- [Does the CodeWarrior IDE support version control systems?](#)
- [How can I add my project to CVS from within the CodeWarrior IDE?](#)

## 2.4.1 Does the CodeWarrior IDE support version control systems?

Yes, CodeWarrior IDE does provide support for the version control systems. One of the version control system supported by the CodeWarrior IDE is CVS.

To use CVS with the CodeWarrior IDE:

1. From the IDE menu bar, select **Window > Show View > Other**.

The **Show View** dialog box appears.

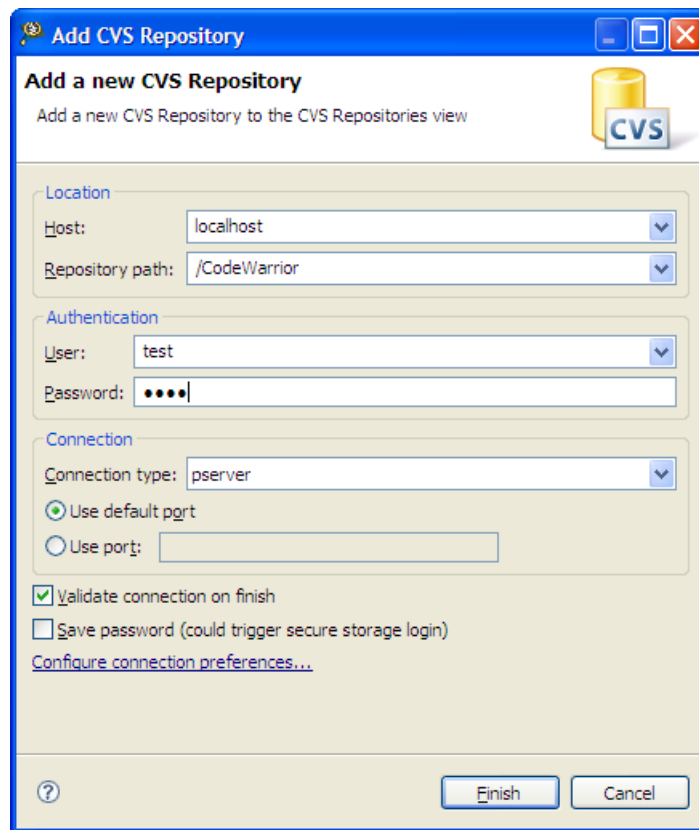
2. Expand the **CVS** tree control and select **CVS Repositories**.
3. Click **OK**.

The **CVS Repositories** view appears.

4. Click the **Add CVS Repository** command on the **CVS Repositories** view toolbar.

The **Add CVS Repository** dialog box appears (refer to the image, *Add CVS Repository Dialog Box* listed below).

5. Enter host name and path of the repository that you want to browse through in the **Host** and **Repository path** text boxes respectively.
6. Enter user name and password in the **User** and **Password** text boxes respectively.

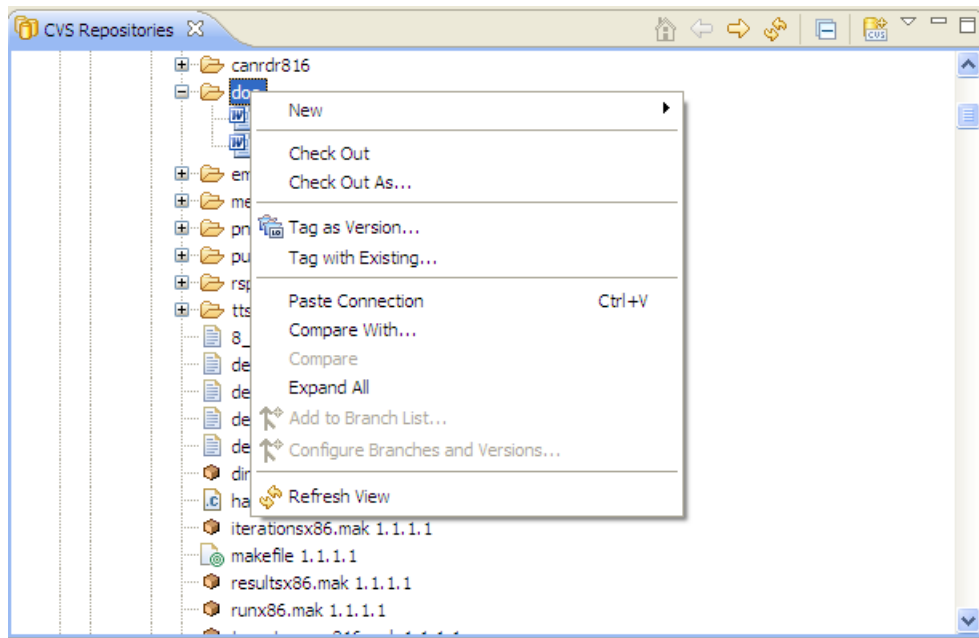


**Figure 2-64. Add CVS Repository Dialog Box**

7. Click **Finish**.

You can browse through the specified repository in the **CVS Repository** view.

8. Right-click the desired folder and select the desired action from the context menu to perform CVS actions, such as **Check Out** (refer to the image listed below).



**Figure 2-65. CVS Repository View**

You can now use CVS with the CodeWarrior IDE.

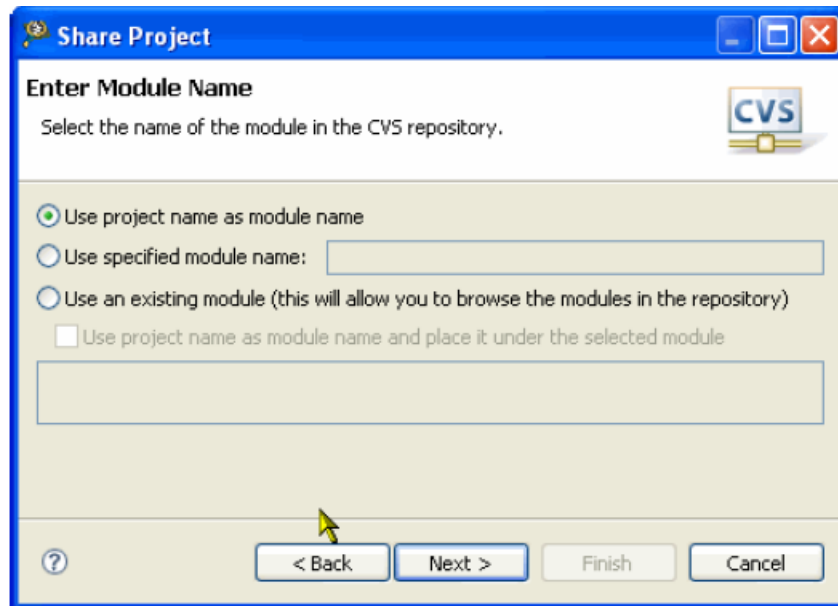
## 2.4.2 How can I add my project to CVS from within the CodeWarrior IDE?

To add a project to the CVS:

1. Right-click the project in the **CodeWarrior Projects** view and select **Team > ShareProject**.

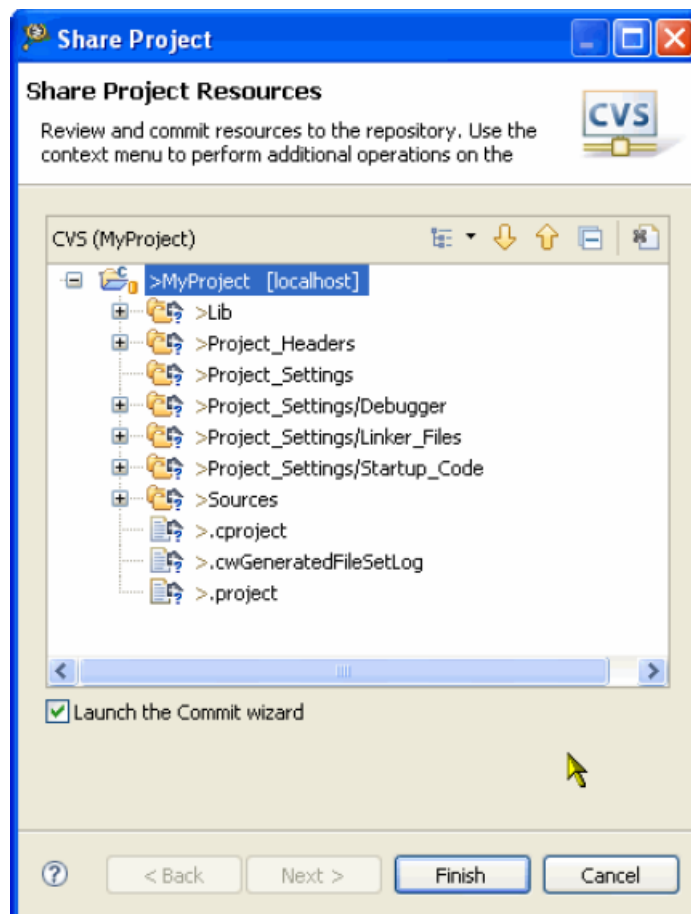
If you have already configured a CVS repository with the CodeWarrior IDE, the **Share Project with CVS Repository** page of the **Share Project** wizard appears. Otherwise, the **Enter Repository Location Information** page appears.

2. Specify the repository to which you want to add the project and click **Next**.  
The **Enter Module Name** page appears.



**Figure 2-66. Enter Module Name Page**

3. Specify name of the module in the CVS repository and click **Next**.  
 The **Share Project Resources** page appears.



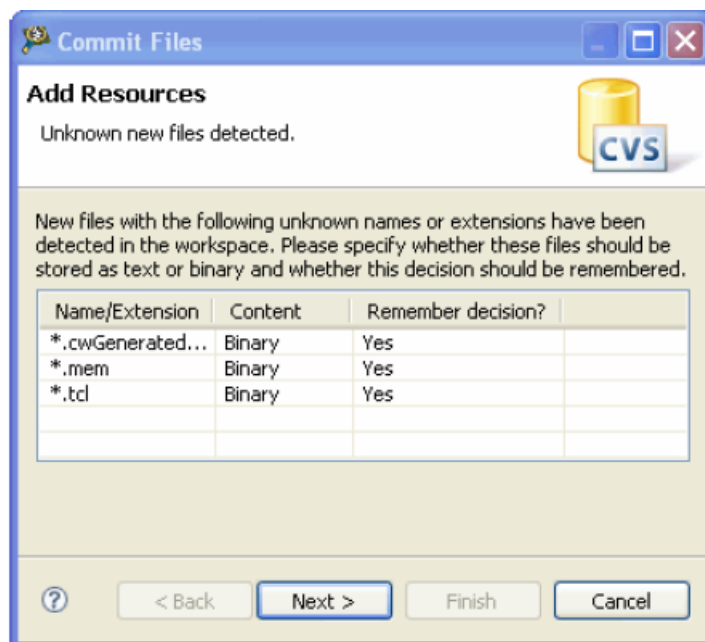
**Figure 2-67. Share Project Resources Page**

4. Select the files that you want to add to the CVS.

If the project includes generated file, then you may not want to add them to the CVS module. You can add such files to the `.cvsignore` list. The `.cvsignore` is a special text file which specifies all the files that should be ignored/filtered out. The `.cvsignore` file can also use wildcards, such as `*.tmp` for filtering.

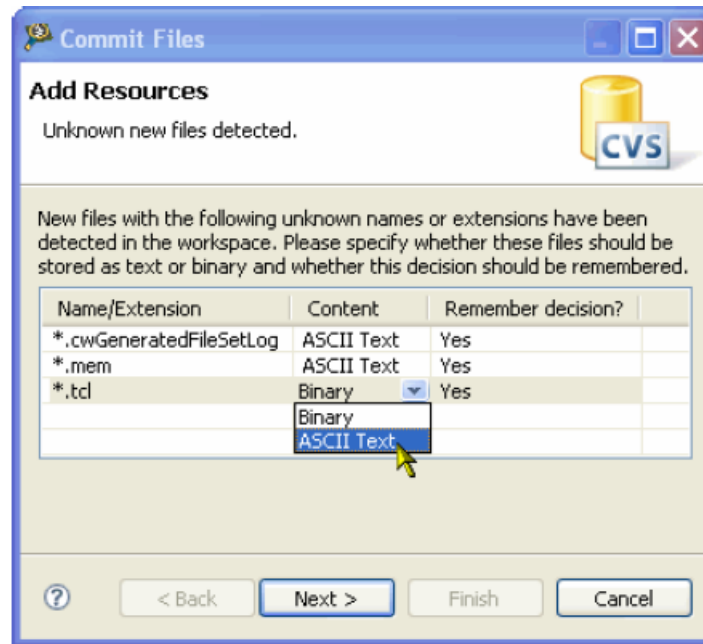
5. Check the **Launch the Commit wizard** checkbox and click **Finish**.

The **Commit Files** wizard starts displaying the list of unknown files.



**Figure 2-68. Commit Files Wizard**

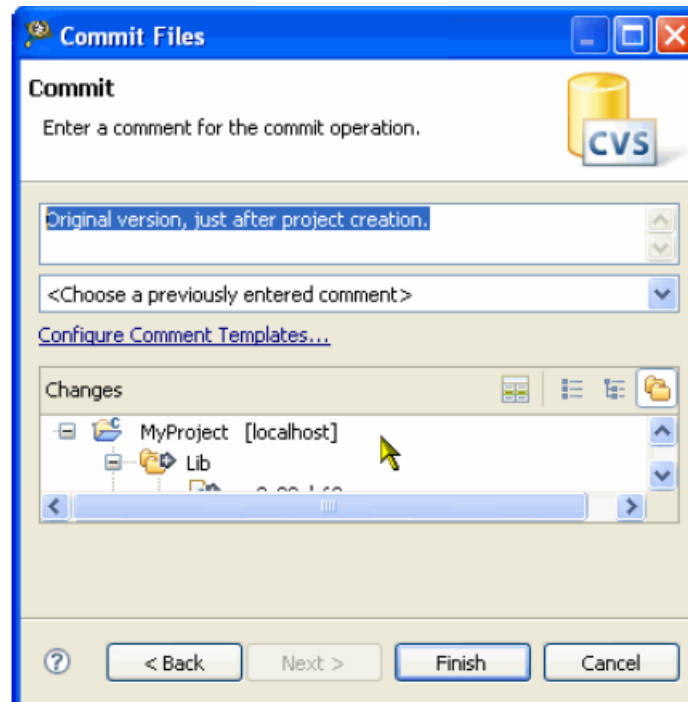
6. Specify whether these files are binary or ASCII Text. In this example, CVS does not identify three file extensions and assumes them as binary files. However, as `*.mem`, `*.tcl`, and `*.cwGeneratedFileSetLog` are text files, the file type is changed to ASCII Text.



**Figure 2-69. Specify How to Store Unknown Files**

7. Click **Next**.

The **Commit** page appears.

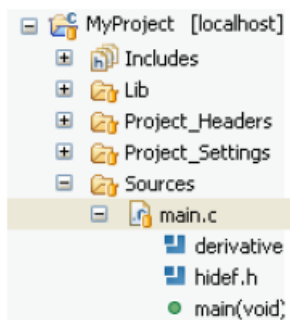


**Figure 2-70. Commit Page**

8. Enter appropriate comment for the commit action and click **Finish**.

The icons of the files and folders in the project change, which specify that the project is now under CVS control.

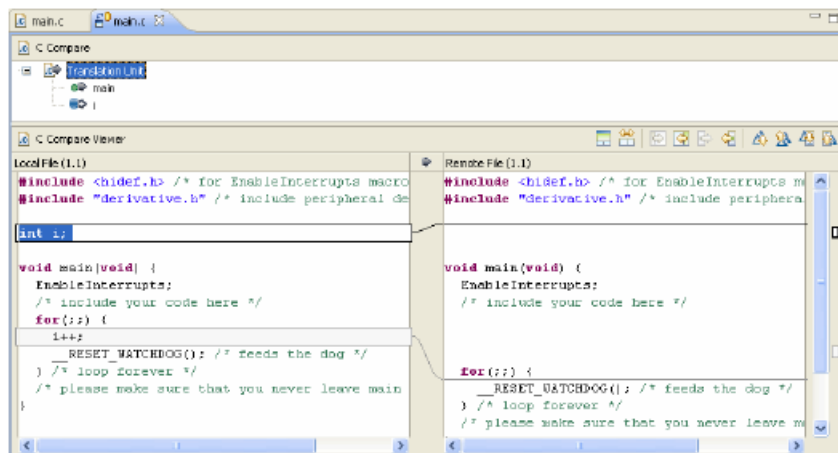




**Figure 2-71. Project Under CVS Control**

Now, if you have done changes to a source file, and you need to view the differences between your local copy and the repository copy, perform these steps.

1. Right-click the modified file and select **Team > Synchronize with Repository**. The **C Compare Viewer** opens displaying comparison between the files (refer to the image listed below).



**Figure 2-72. C Compare Viewer**

2. Review the changes. If you are ready to commit the changes, right-click the file and select **Team > Commit**.
3. Enter the appropriate comment and click **Finish**.

You can also perform the synchronization in the **Synchronize** view in the **Team Synchronizing** perspective (refer to the image listed below).

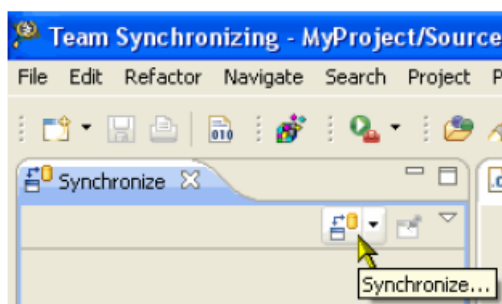


Figure 2-73. Synchronize View

The **Synchorize** view displays what has been changed either locally or remotely. For example, in the **Synchorize** view (in the following figure), the file `main.c` is on revision 1.2 with some outgoing changes, that is some changes are done locally in the file, and there is a new file `test.c` which is not committed yet.

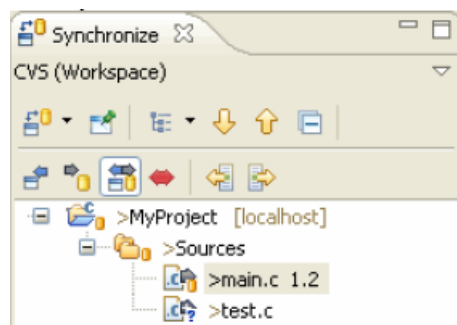


Figure 2-74. Synchronize View Displaying File Revisions

## 2.5 Search and Replace Action

In this topic, Search and Replace related FAQs are listed.

- [How do I perform a simple search in the CodeWarrior IDE?](#)
- [Is it possible to preview the changes of the Replace action?](#)
- [Is there a way to change the variable or the function name in a CodeWarrior project?](#)
- [How can I search for specific files in a workspace?](#)
- [Is there a way to replace a variable easily?](#)
- [How can I go back to the previous location if the Editor takes me to another source or header file when, for example, I select the Open Declaration option?](#)
- [Is there a shortcut to open the search window for a specified text that is equivalent to the Ctrl + Shift + M shortcut key available in the Classic CodeWarrior IDE?](#)
- [Which set of files I need to backup or store in a version control system to completely recover project settings and related debug settings?](#)

- I want to perform a Regular expressions search, but I do not remember the syntax of the regular expressions. What do I do?
- How do I merge the multiple empty lines into a single one in the Editors view?

### 2.5.1 How do I perform a simple search in the CodeWarrior IDE?

You can either use the **Search** menu in the menu bar or select the **Edit > Find/Replace** command. However, there is a difference. The **Search** menu displays the commands **C/C++**, **Search**, **File**, and **Text**.

- **C/C++** - Opens the search dialog on the C/C++ search page
- **Search** - Opens the search dialog for your current editor
- **File** - Opens the search dialog on the File search page
- **Text** - Opens the submenu for full-text search in given scope (workspace, project, file or working set)

Selecting the **Select** menu opens the **Search** dialog box that includes two specialized tabs: **File Search** and **C/C++ Search**.

Whereas, the **Find/Replace** command displays the **Find/Replace** dialog box that lets you search for an expression in the active editor, and replace the expression with a new expression.

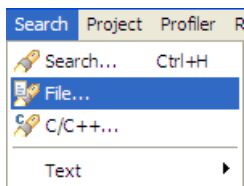
Thus, for a simple find and replace use **Edit > Find/Replace**.

### 2.5.2 Is it possible to preview the changes of the Replace action?

Yes, to preview the changes of the Replace action, follow these steps:

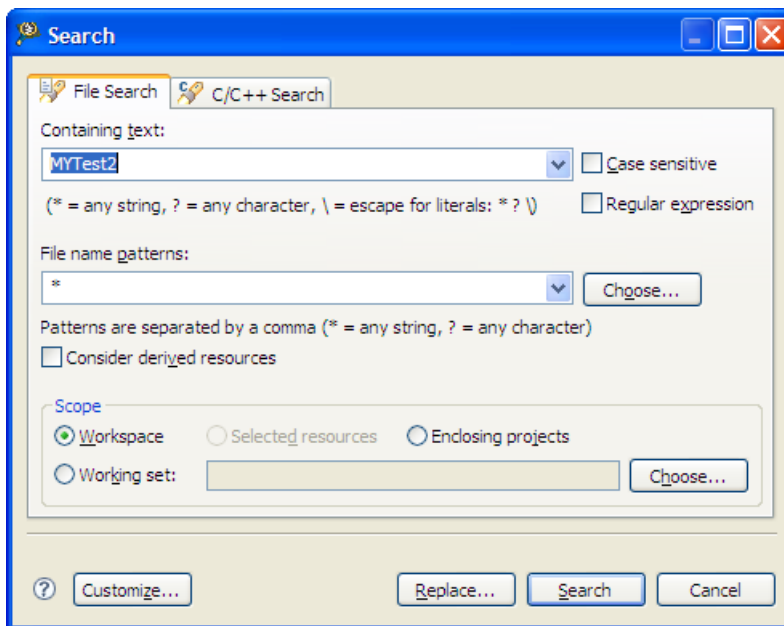
1. Select **Search > File** (refer to the image, *Search Menu* as listed below) from the IDE menu bar.

The **File Search** tab page of the **Search** dialog box appears (refer to the image, *Search Dialog Box* as listed below).



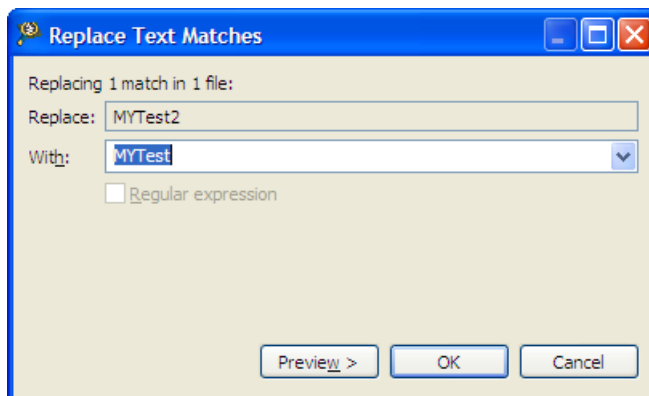
**Figure 2-75. Search Menu**

- Specify the text that you want to replace in the **Containing text** text box.



**Figure 2-76. Search Dialog Box**

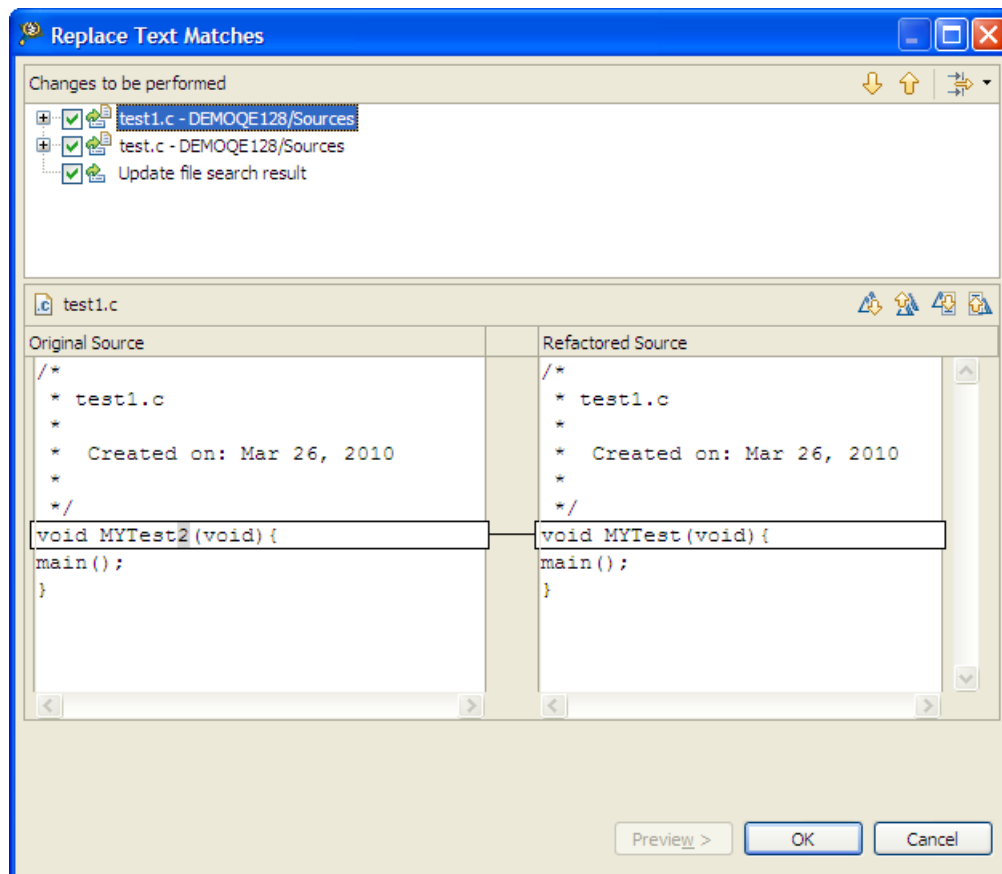
- Click the **Replace** button. The **Replace Text Matches** dialog box appears.
- In the **With** text box, enter the text with which you want to replace the text entered in the **Replace** text box.



**Figure 2-77. Replace Text Matches Dialog Box**

- Click the **Preview** button.

The dialog box lists the files to which the changes will be performed as a result of the replace action. You can preview each of the change and check or clear a file to accept or reject the change as required.



**Figure 2-78. Preview Replace Action**

6. Click **OK**.

The **Replace Text Matches** dialog box is closed.

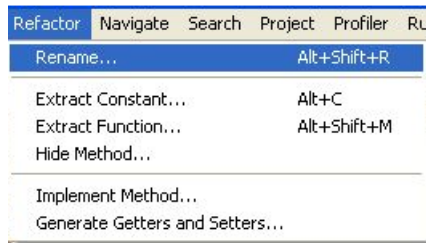
The changes as a result of the replace action are performed to the selected files.

### 2.5.3 Is there a way to change the variable or the function name in a CodeWarrior project?

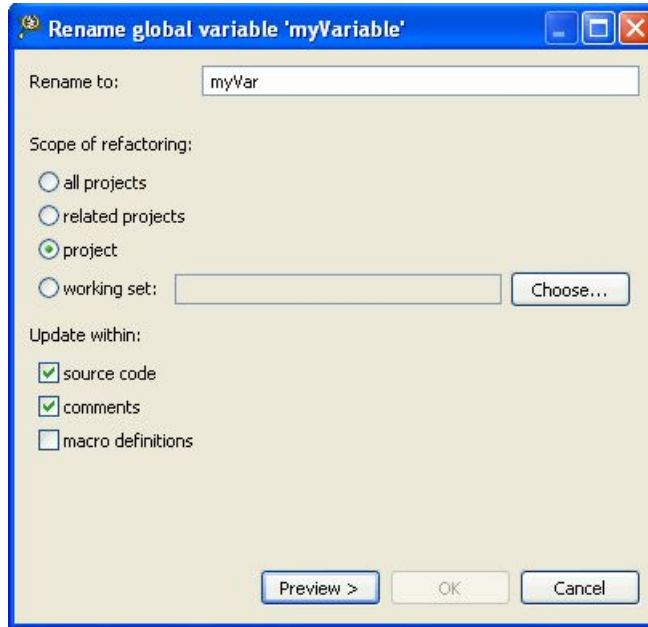
Yes, you can use the Refactoring feature of the CodeWarrior IDE to replace a function or a variable name.

1. Select the variable or function and select **Refactor > Rename** from the IDE menu bar.

The **Rename global variable** `*<variable name>*` dialog box appears.

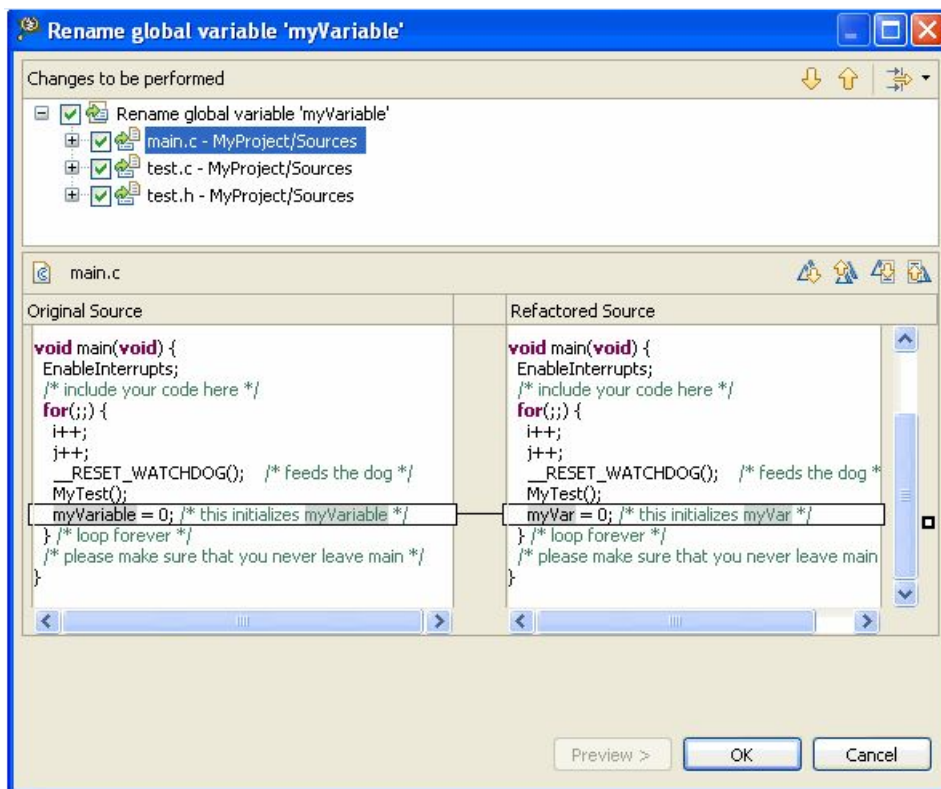


**Figure 2-79. Refactor Menu**



**Figure 2-80. Rename global variable Dialog Box**

2. Specify the new name for the variable.
3. Specify the scope of refactoring, such as all projects, related projects, project, or working set.
4. Specify where in the source file you want to make the changes, such as source code, comments, and/or macro definitions.
5. Click the **Preview** button to preview the changes.



**Figure 2-81. Preview Replace Action**

6. Check or clear a file to accept or reject a change as required.
7. Click **OK**.

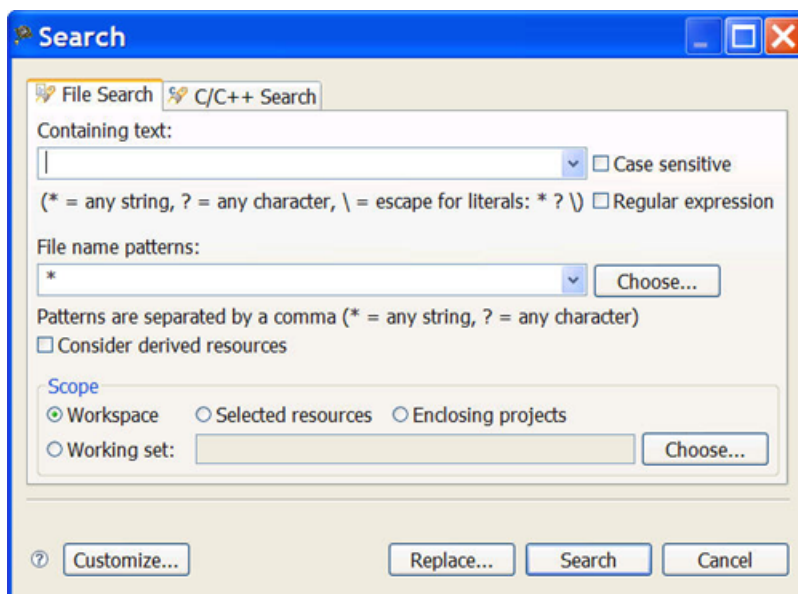
The specified changes are performed.

### 2.5.4 How can I search for specific files in a workspace?

To search for specific files in a workspace, follow these steps:

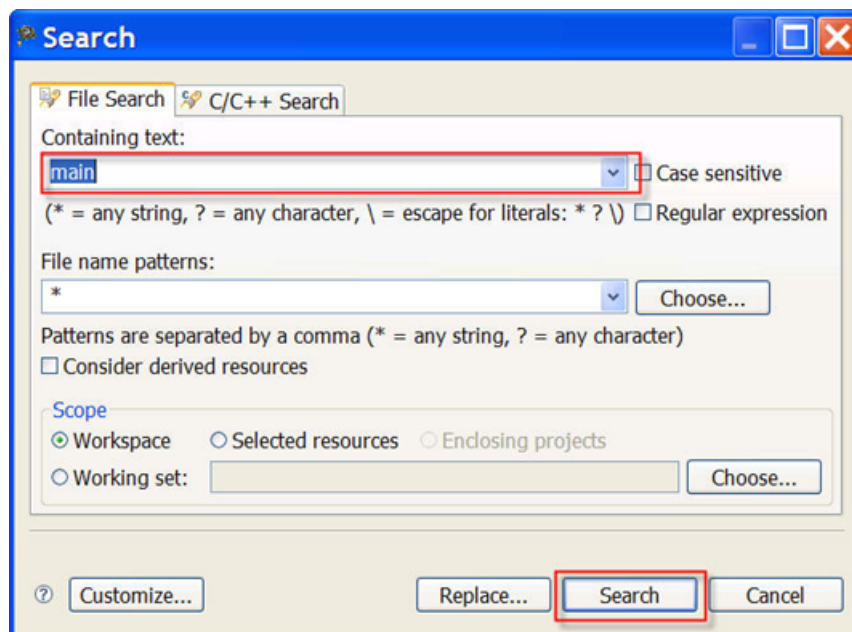
1. From the IDE menu bar, select the **Search > File...** option.

The **Search** dialog box appears.



**Figure 2-82. Search Dialog Box**

2. Specify the search string in the **Containing Text** field. Specify other search options according to your requirements (refer to the image listed below).



**Figure 2-83. Search Dialog Box**

3. Click the **Search** button.

The **Search** view appears (as the figure listed below shows). The **Search** view displays the results of your search. Right-click any item in the **Search** view to open a pop-up menu that allows you to remove items from the list, copy search results to the clipboard, or rerun the search.



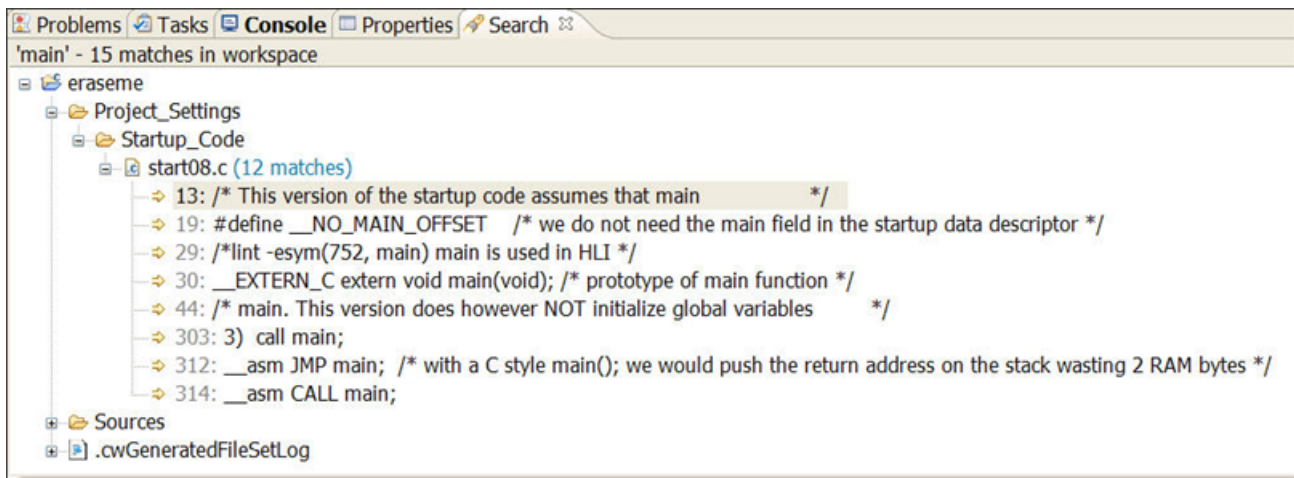


Figure 2-84. Search View

## 2.5.5 Is there a way to replace a variable easily?

Yes, to replace a variable, follow these steps:

1. From the IDE menu bar, select the **Search > File...** option.

The **Search** dialog box appears.

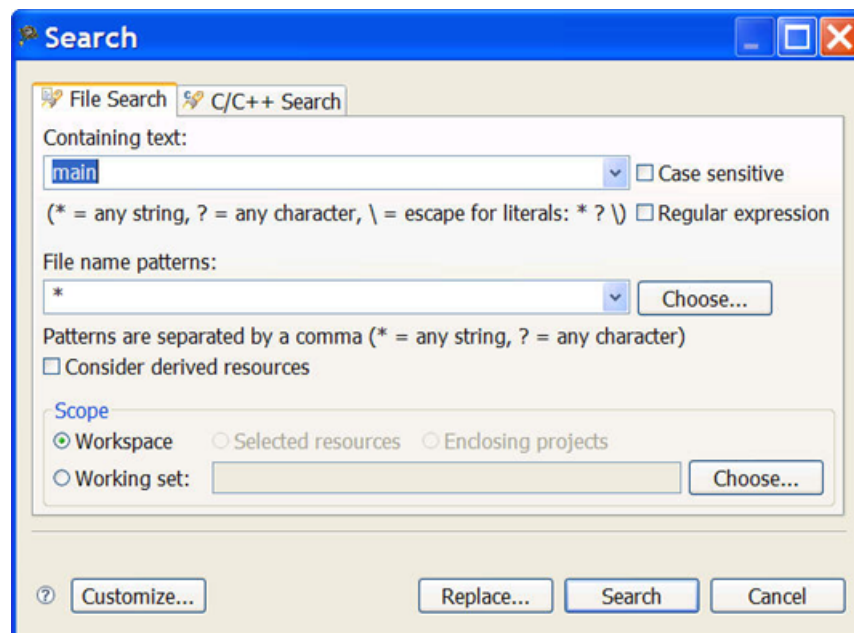
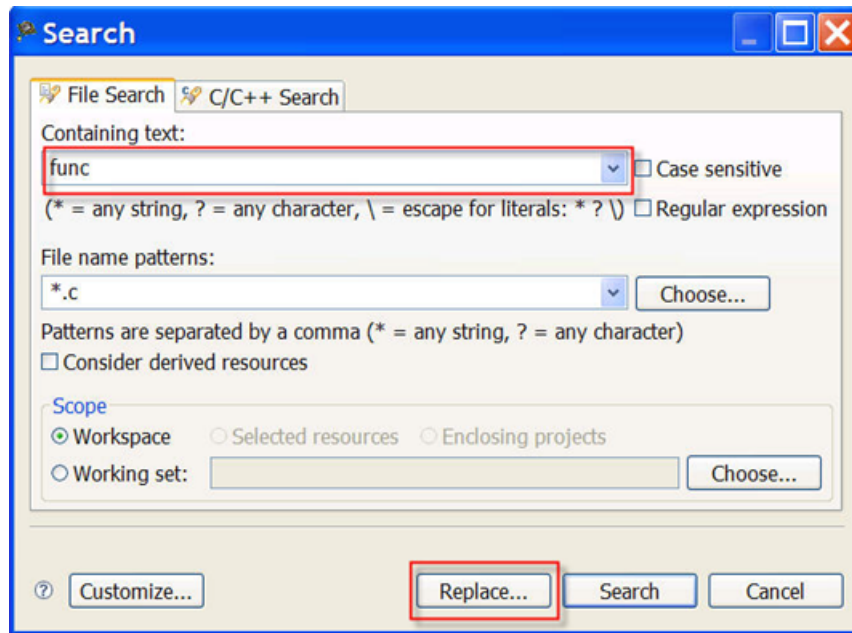


Figure 2-85. Search Dialog Box

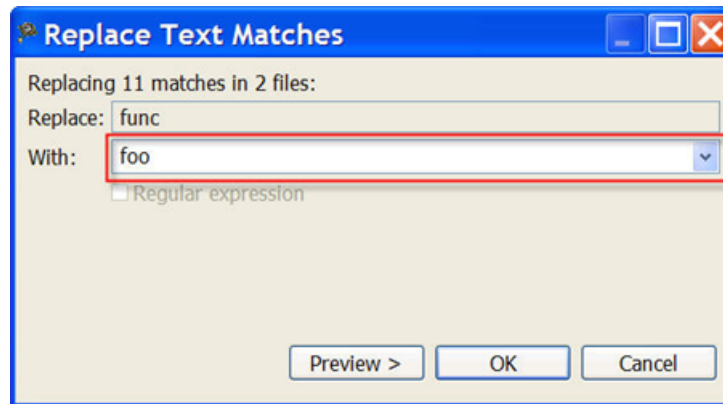
2. Specify the search string in the **Containing Text** field. Specify other search options according to your requirements (refer to the image listed below).



**Figure 2-86. Search Dialog Box**

3. Click **Replace**.

The **Replace Text Matches** dialog box appears.



**Figure 2-87. Replace Text Matches Window**

4. Click **Preview**.

The **Replace Text Matches** window appears (refer to the image listed below). This window gives lets you preview the changes that will be made. This allows you to inspect each proposed change and accept or deny it individually.

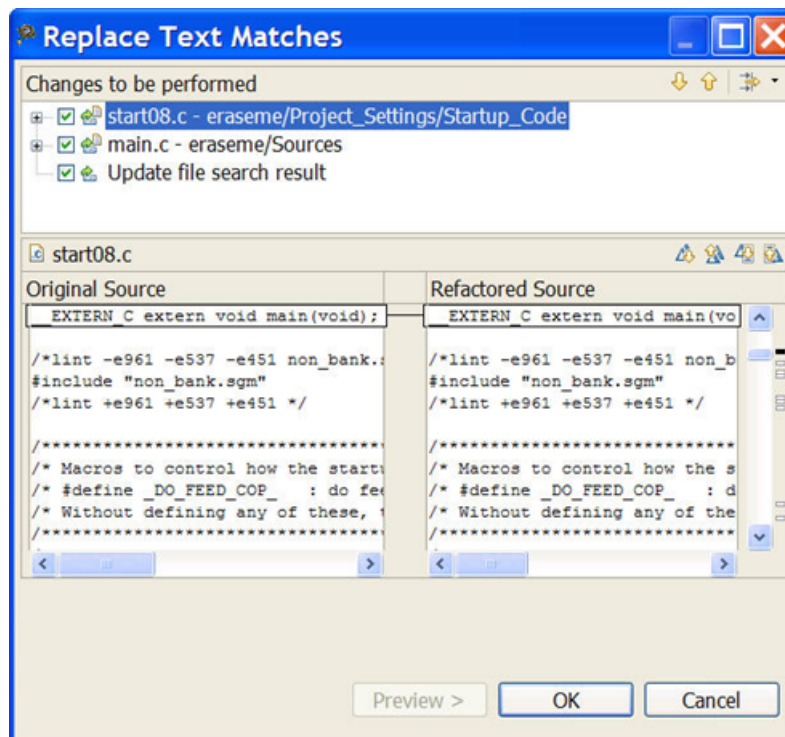


Figure 2-88. Replace Text Matches Window

### 2.5.6 How can I go back to the previous location if the Editor takes me to another source or header file when, for example, I select the Open Declaration option?

Use the **Alt+left** shortcut key. This option is also listed under the **Navigation** menu.

### 2.5.7 Is there a shortcut to open the search window for a specified text that is equivalent to the **Ctrl + Shift + M** shortcut key available in the Classic CodeWarrior IDE?

You can create a custom key binding for the **Ctrl+Shift+M** shortcut key in the **Preferences > General > Keys** dialog box.

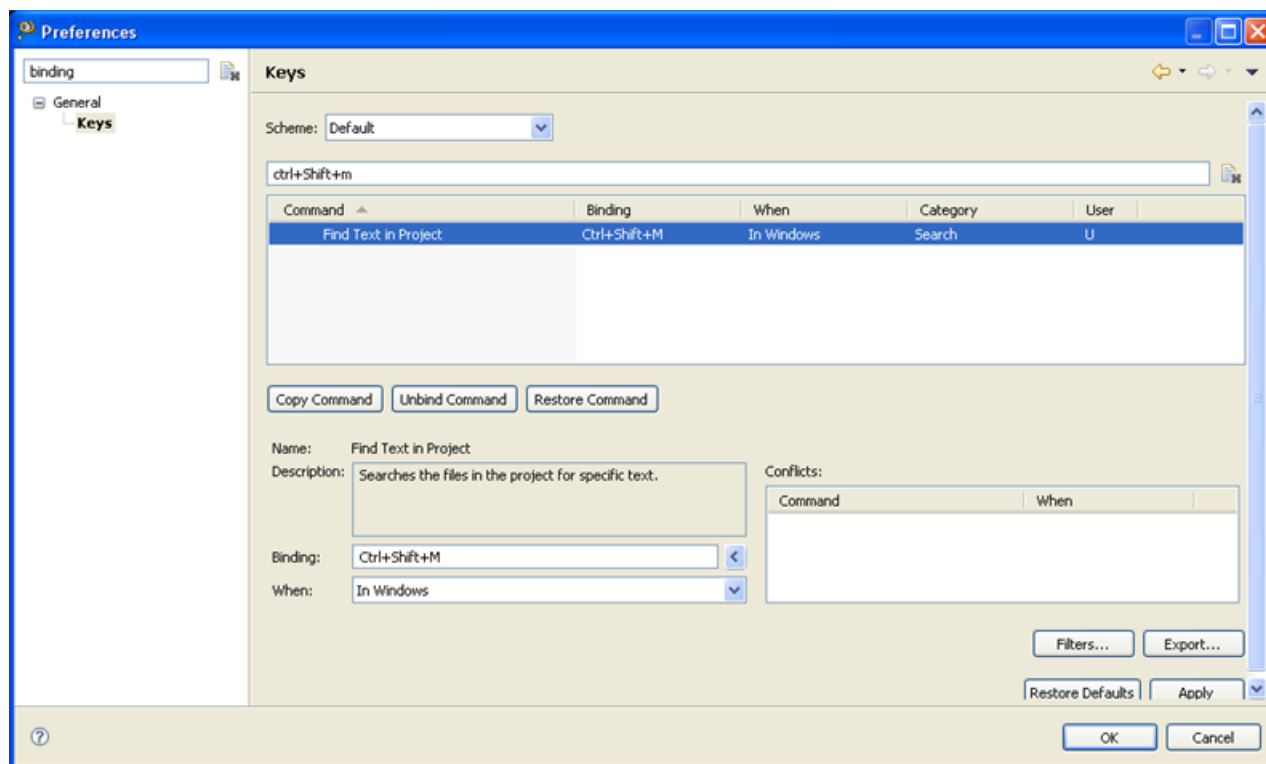


Figure 2-89. Preferences Dialog Box - Keys Page

**NOTE**

For more information on key binding, refer to the topic [How can I change a key binding?](#)

### 2.5.8 Which set of files I need to backup or store in a version control system to completely recover project settings and related debug settings?

The files you need to store in a version control system:

- `.project` and `.cproject` - contains the list of files and the build tool settings.
- `.ttf` - targets the needing target tasks for flash programming (ColdFire V2 and Kinetis IAR JLink)
- `.launch` - required for the debugger launch configuration
- `rseHostSettingsCache.xml`- contains the cached RSE settings. You need this to have the RSE settings imported.
- `*.c/*.h/etc` source files and linker file, if not using Processor Expert
- If you are using Processor Expert - `ProcessorExpert.pe` (contains all the settings), `events.c/h`, and the `ProcessorExpert.c` with user modifications

Any generated folders (like PE Generated\_Code, PE Documentation or the folder where the object/make files are placed (usually named by the CPU name) can be stripped off to reduce file size.

You can also export/store the workspace settings, like editor/VCS settings/etc, by selecting, **File > Export > Preferences** from the IDE menu bar.

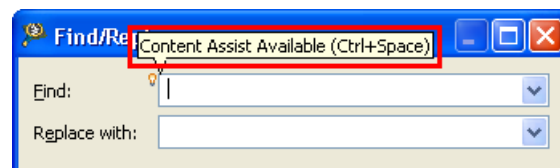
## 2.5.9 I want to perform a Regular expressions search, but I do not remember the syntax of the regular expressions. What do I do?

To get the list of regular expressions:

1. Press **Ctrl+F** .

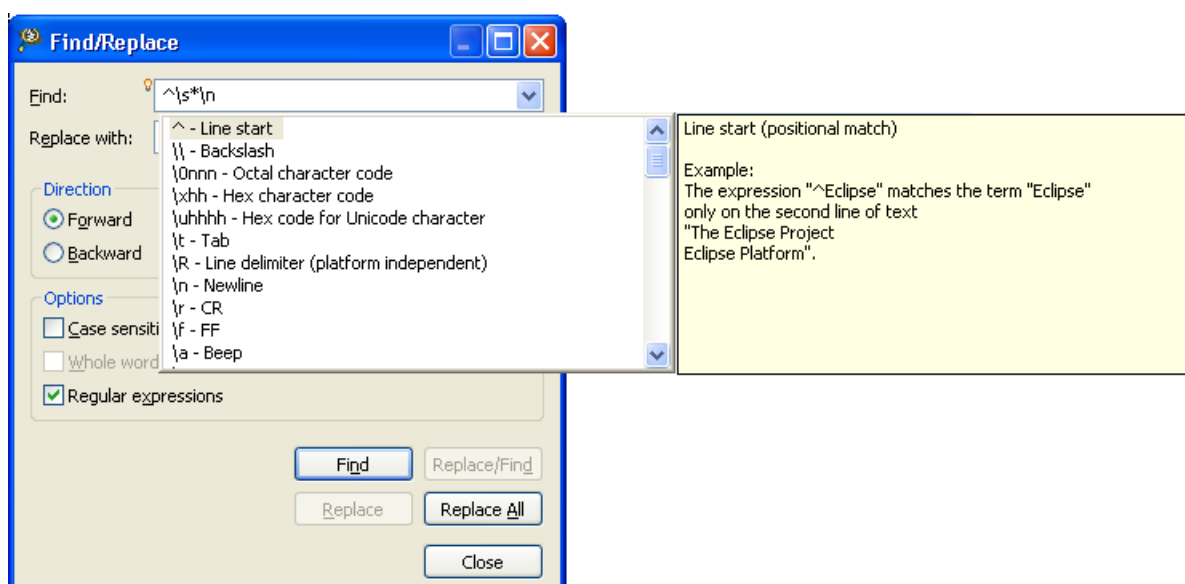
The **Find/Replace** dialog box appears.

2. Click the small icon in the beginning of the **Find** textbox, alternatively press **Ctrl+Space** .



**Figure 2-90. Find/Replace Dialog Box - Content Assist Icon**

The content assist drop-down list appears listing all the regular expressions.



**Figure 2-91. Find/Replace Dialog Box - Content Assist Drop-down List**

3. Double-click the expression to insert it in the textbox.

The selected expression appears in the textbox.

You can now perform the search for the selected expression.

## 2.5.10 How do I merge the multiple empty lines into a single one in the Editors view?

To merge the multiple empty lines into a single one in the Editors view:

1. Press Ctrl+F.

The **Find/Replace** dialog box appears.

2. Enter the expression `^\s*\n` in the **Find** textbox.
3. Enter the expression `\R` in the **Replace** with dialog box.

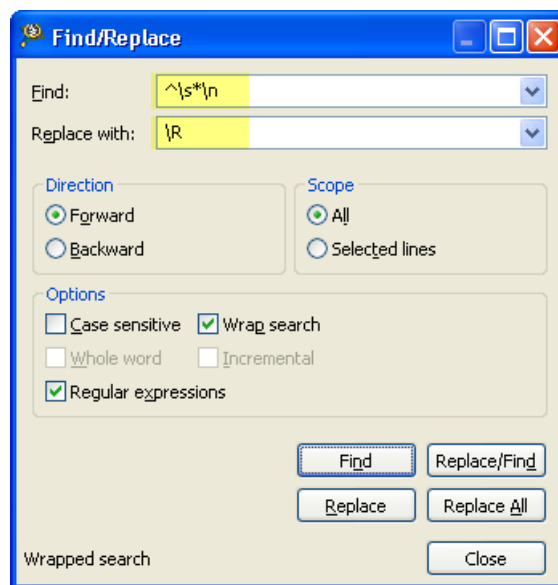


Figure 2-92. Merging Multiple Blank Lines into One


4. Click **Replace All**.

The editor merges the multiple blank lines into one for each appearance.

## 2.6 Miscellaneous

In this topic, miscellaneous FAQs related to the CodeWarrior IDE are listed.

- How can I improve the performance of the CodeWarrior IDE I am working with?
- What is the first thing that I see when I start the CodeWarrior IDE?
- What is a perspective?
- How can I find the version of the CodeWarrior that I am using?
- Is it possible to retrieve a file that I deleted accidentally from my CodeWarrior project?
- I am using my workspace since long time and adding/removing projects frequently, this has slowed down my CodeWarrior software. How can I improve the performance?
- How do I update my local history settings?
- Is there a way to filter the settings in the Preferences window to find a particular setting?
- How can I change a key binding?
- Is it possible to view the change history of a source code file?
- How can I view a graphical representation of source code in the CodeWarrior IDE using Dot and Doxygen?
- How can I determine which header files are associated with my source code file?
- How do I view call hierarchy in my source code?
- Why the project that I just created is not visible in the CodeWarrior Projects view?
- I created a new file in Windows Explorer but the file does not appear in the CodeWarrior Projects view. Why?
- Why does the Rename option appear grayed out?
- Why duplicating a configuration in the debugger perspective does not duplicate the run configuration?
- Can I still use Ctrl+Tab keys to navigate between open windows?
- How can I change the debugger key bindings to the ones that I used to have in the Classic CodeWarrior IDE?
- How can I change the name of the executable that is generated when I build my project?
- How can I see printf output in the console window?
- Why the console view does not display all the warning and error messages by default?
- Is there a file that contains all the launch configuration settings?
- How the CodeWarrior Project Importer handles recursive access paths?
- Can I disassemble my source code file?
- Where exactly the disassemble file gets created?
- Why the .metadata folder in my workspace stores a huge history?
- How to have projects from different workspaces or locations in a workspace?
- How can I switch to another workspace?
- My workspace takes long to close. How can I resolve this?
- What is the purpose of the Tasks view?
- What is the purpose of the Properties view?

- What is the purpose of the Outline view?
- How do I use the Outline View to get an overview of a source file and modify it directly?
- How can I find out if certain files contain debug information?
- How can I resolve the following error message that I get when I start the CodeWarrior IDE?
- How can I open an existing project in the CodeWarrior IDE?
- What is the Manage Configurations button in the C/C++ perspective toolbar used for? Could it be replaced by the Properties button?
- How can I start the post-build steps in the CodeWarrior IDE?
- Is it necessary to have the project name identical to the name of the directory that contains the .project file?
- Where does the CodeWarrior IDE save the debug configuration as a local file by default?
- When I save my launch configurations as a local file and then delete the project, all the local configurations get deleted as well. How can I resolve this issue?
- How can I modify and save the files in GBK encoding using the CodeWarrior IDE?
- How are the PARENT-COUNT-MyVariable definitions defined in the CodeWarrior IDE?
- Is there a way to instruct the CodeWarrior IDE to use relative paths instead of absolute ones to store the project file location in a workspace?
- How do I solve the probable mismatch of the OSJTAG Firmware with P&E Virtual Serial Port Utility?
- I am using CW10.1SE and MQX3.7 with all patches and updates. How can I solve this problem?
- What is a Working Set? How do I build/compile a Working Set?
- How do I search within the Working Set?
- How do I manage the Working Sets?
- How can I filter the projects listed in the CodeWarrior Projects view based on Working Sets?
- How can I import/export a Working Set in the CodeWarrior eclipse IDE?
- What are Icon decorators?
- How do I enable or disable an icon decorator?
- Can the icon decorators be combined?
- When I open a large file in CodeWarrior IDE, an Editor Scalability dialog box appears, listing the message, "You are opening a large file. Scalability mode has been turned on for this editor to help improve performance by disabling features requiring to parse the source file language." What does it mean?
- How do I configure the Scalability settings?
- What is Timing System?
-  What are the features of freedom board FRDM-K20D50M?



- **NEW** What are the disadvantages of using printf() in the embedded system programming?
- **NEW** The OpenSDA connection is not available in the Connection page of the wizard. What do I do?
- **NEW** My bootloader resides in the first lower flash pages. How do I avoid it to get destroyed and overwritten by the application?
- **NEW** I am debugging FRDM-KL05Z with USBDM, but I received a 'Failed to resume target process., Downloading binary to target..' error message. What do I do?
- **NEW** How does a bootloader work?
- **NEW** How do I use printf() with the FRDM-KL25Z Board without using the Processor Expert?
- **NEW** How do I use Freedom board as SWD programmer?
- **NEW** How do I use CMSIS-DAP with IAR on KL25Z Freedom Board?
- **NEW** How do I set up criteria for bootloader to decide if it shall run the bootloader or the application at the startup?
- **NEW** How do I make sure that the bootloader gets linked only into its specified space?
- **NEW** How do I make sure that application does not overlap with the FLASH area of the bootloader?
- **NEW** How do I load the Debug Application on my FRDM-K20D50M board?
- **NEW** How do I install open source USBDM?
- **NEW** How do I fix the 3.3V voltage drop on FRDM-KL25Z Board?
- **NEW** How do I decode S19 Files?
- **NEW** How do I bootload an application?
- **NEW** How can I add my own code to be inserted as part of the system startup?
- **NEW** For my bootloader I need a way to send a file with a terminal program. As my serial connection has only Tx and Rx, but no RTS/CTS lines for flow control, it is useful if the terminal program either implements software flow control (XON/XOFF), or a delay value for sending a file. What do I do?
- **NEW** Do I need to initialize my port pin?
- **NEW** Can MCU10.4 recover a bricked OpenSDA Freedom Board?
- **NEW** Can I program a different Freedom Board using Freedom board as SWD programmer?
- **NEW** What Bootloader Shell commands I can use?
- **NEW** What is a bootloader?
- **NEW** What is Bootloader Sequence?
- **NEW** What is the code size required for the bootloader for FRDM-KL25Z board?
- **NEW** Where can I find the software for my FRDM-K20D50M board?
- **NEW** Which device should I use to create a new project for my FRDM-K20D50M board using CodeWarrior?

## 2.6.1 How can I improve the performance of the CodeWarrior IDE I am working with?

Yes, you can improve the performance of your CodeWarrior IDE by disabling the features that slow down the software.

### 1. Disable Indexer Options

- a. From the CodeWarrior IDE menu bar, select **Window > Preferences**.

The **Preferences** dialog box appears.

- b. Type **index** as the filter text to narrow down the list of preferences.
- c. Select **C/C++ > Indexer**.

The **Indexer** page appears in the right panel of the **Preferences** dialog box.

- d. Clear the **Enable indexer** checkbox.

### 2. Disable Scalability Mode Settings

- a. In the **Preferences** dialog box, type **scalability** as the filter text.
- b. Select **C/C++ > Editor > Scalability**.

The **Scalability** page appears in the right panel of the **Preferences** dialog box.

- c. Clear the **Enable all scalability mode options** checkbox.

### 3. Disable Content Assist Auto Activation

- a. In the **Preferences** dialog box, type **content** as the filter text.
- b. Select **C/C++ > Editor > Content Assist**.

The **Content Assist** page appears in the right panel of the **Preferences** dialog box.

- c. Clear all checkboxes in the **Auto-Activation** section.

### 4. Click **OK**.

The performance of the CodeWarrior IDE improves.

## 2.6.2 What is the first thing that I see when I start the CodeWarrior IDE?

The window that you see when you bring up the IDE is called a workbench. A workbench consists of perspective, views, and editors. The term *workbench* refers to the host development environment.

### 2.6.3 What is a perspective?

A perspective is an arrangement of views and editors. You can quickly switch between different perspectives.

### 2.6.4 How can I find the version of the CodeWarrior that I am using?

To find the version of the CodeWarrior that you are using, follow these steps:

1. From the IDE menu bar, select the **Help > About CodeWarrior Development Studio** option.

The **About CodeWarrior Development Studio** dialog box appears.

The dialog box that appears displays the version number and the build number under **Installed Products** section.

### 2.6.5 Is it possible to retrieve a file that I deleted accidentally from my CodeWarrior project?

Yes, you can retrieve your deleted files. However, instead of the system's Recycle Bin you need to retrieve the file from the local history. The local history maintains a copy of files you modify or delete.

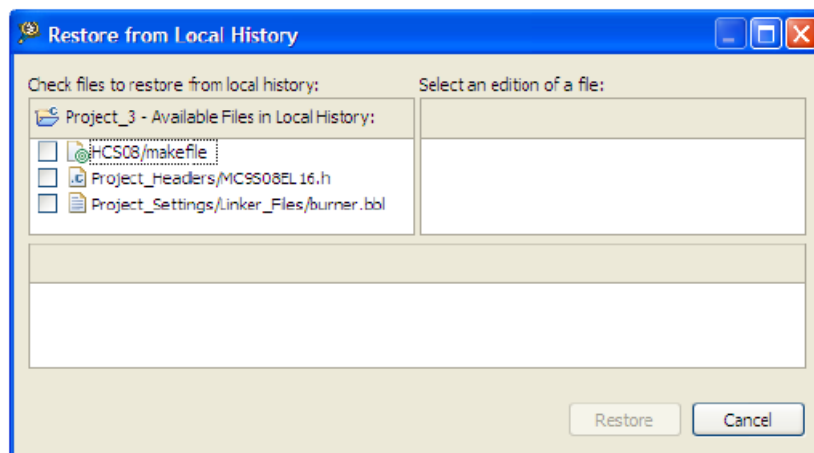
To restore a deleted file, you need to perform the following steps.

1. In the **CodeWarrior Projects** view, right-click the project branch you deleted the file from.
2. From the context menu, select the **Restore from Local History** command.

The **Restore from Local History** dialog box appears.

#### NOTE

The dialog box contains a list of deleted files, along with a checkbox for each file in the list.



**Figure 2-93. Restore from Local History Dialog Box**

3. Check the files you want to restore from the local history.
4. Click the **Restore** button.

The selected files are restored.

### 2.6.6 I am using my workspace since long time and adding/removing projects frequently, this has slowed down my CodeWarrior software. How can I improve the performance?

To improve the speed and performance of the CodeWarrior IDE:

1. Close all the unrequired/unrelated projects to improve the performance.
2. Export the workspace settings. To export the workspace settings:
  - a. Select **File > Export** from the IDE menu bar.

The **Export** dialog box appears.

- b. Expand the **General** tree-control and select **Preferences** .
- c. Click **Next** .

The **Export Preferences** dialog box appears.

- d. Select the file to which want to save the preferences in the **To preference file** combo-box.
- e. Click **Finish** .

The current workspace settings are saved in the specified file.

3. Close the CodeWarrior IDE.

4. Delete the `.metadata` folder from the workspace.
5. Open the CodeWarrior IDE with that workspace.
6. Import the previously exported workspace settings. To import the previous workspace settings:
  - a. Select **File > Import** from the IDE menu bar.

The **Import** dialog box appears.

- b. Expand the **General** tree-control and select **Preferences** .
- c. Click **Next** .

The **Import Preferences** dialog box appears.

- d. Select the file from which want to load the preferences in the **From preference file** combo-box.
- e. Click **Finish** .

The previous workspace settings are loaded from the specified file.

## 2.6.7 How do I update my local history settings?

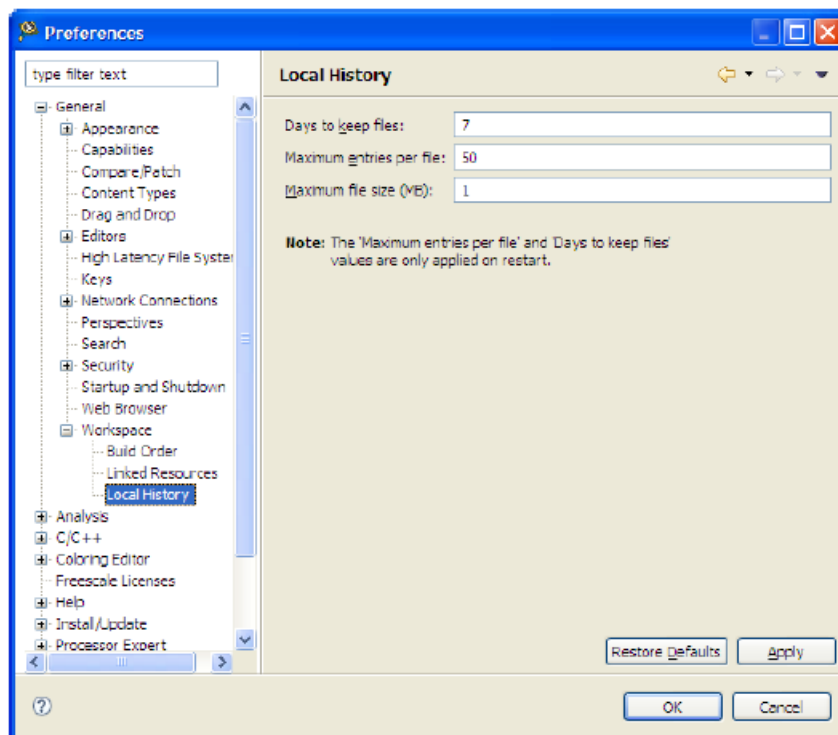
To update the local history settings or preferences, perform the following steps.

1. From the IDE menu bar, select the **Window > Preferences** option.

The **Preferences** dialog box appears.

2. Expand the tree control to select the **General > Workspace > Local History** option.

The **Local History** page appears.



**Figure 2-94. Preferences Dialog Box - Local History Page**

3. In the **Days to keep files** field, specify the number of days you want to maintain changes in the local history. The default value is 7.

#### **NOTE**

History state older than the default value will be lost.

4. In the **Maximum entries per file** field, specify how many history states per file you want to maintain in the local history. The default value is 50.

#### **NOTE**

If you exceed the default value, you will lose older history to make room for new history.

5. In the **Maximum file size (MB)** field, specify the maximum size of individual states in the history store. The default value is 1.

#### **NOTE**

If you exceed the default value, the individual states will not be stored.

6. Click **Apply** to confirm and save the settings.
7. Click **OK**.

The **Preferences** dialog box closes.

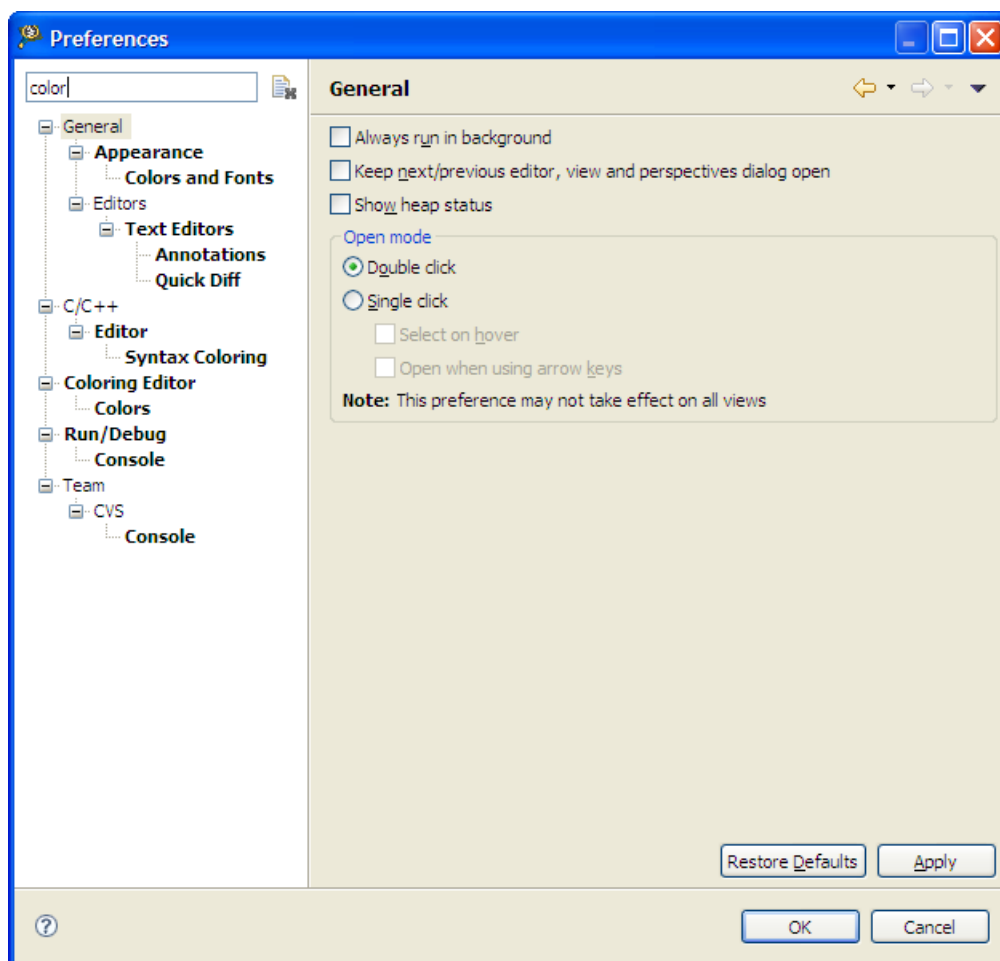
The local history settings is updated.

## NOTE

To revert to the default settings, click the **Restore Defaults** button in the **Preferences** dialog box.

### 2.6.8 Is there a way to filter the settings in the Preferences window to find a particular setting?

Yes, you can filter settings using the filter text. For example, if you are looking for some settings to change the color in a view, type **color** as the filter text in the Preferences dialog box (refer to the image listed below). The preferences with text **color** in it are listed. This reduces the number of preferences to browse through.



**Figure 2-95. Filtered Preferences**

You can clear the filter text using either the keyboard or the 'X' icon.

You can also use wildcards in the filter text.

- \* - Substitutes for zero or more characters
- ? - Substitutes for any one character

For example, if you are looking for a device that has 'J' in its name, however you are not sure if it is a 16k or a 12k device, then you can search for the device using the wildcards in filter text, refer the following figure.



Figure 2-96. Filtering Using Wildcards

## 2.6.9 How can I change a key binding?

To change a keyboard shortcut or a keyboard binding:

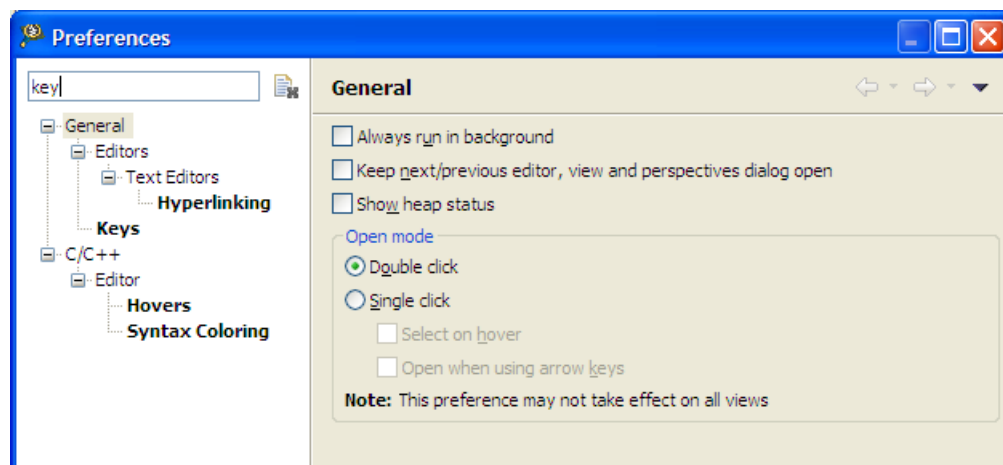
1. From the IDE menu bar, select **Window > Preferences** .

The **Preferences** dialog box appears.

2. Type filter text as **key** to narrow down the list of the preferences.

The modifications need to be done in the **General > Editors > Keys** preference page.

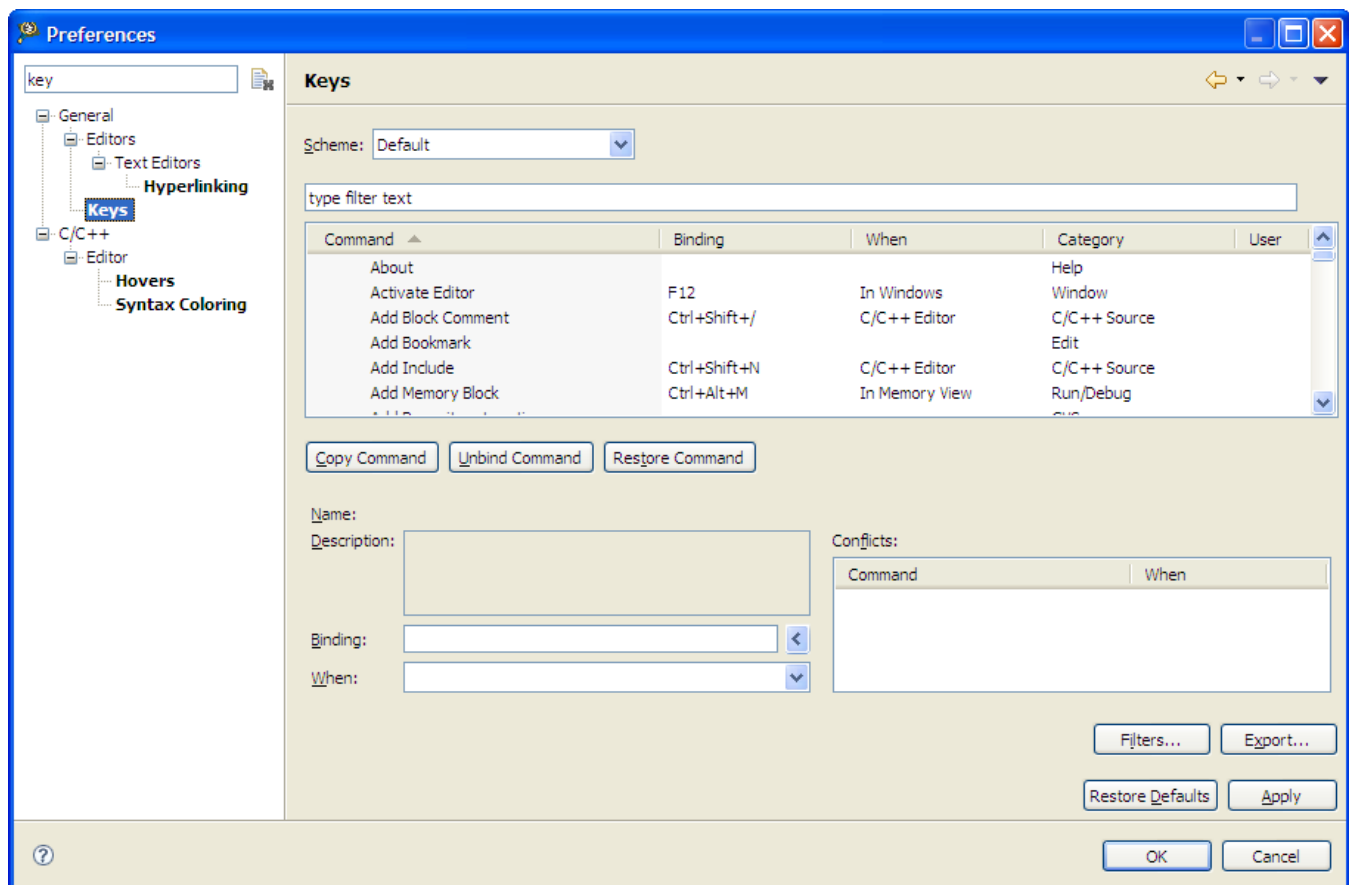




**Figure 2-97. Preferences Dialog Box**

3. Select **General > Editors > Keys** .

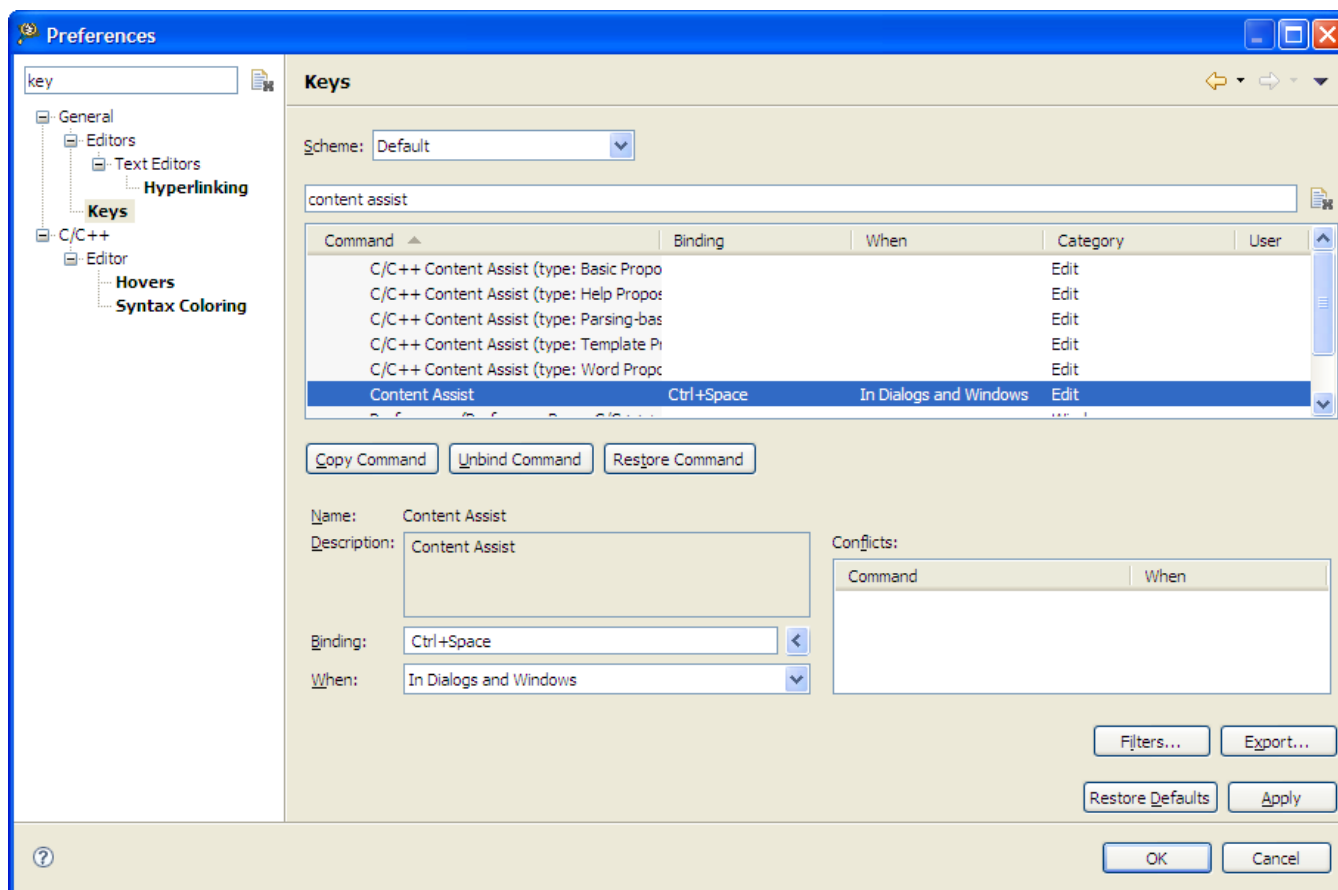
The **Keys** preferences page appears in the right panel of the **Preferences** dialog box.



**Figure 2-98. Keys Preferences Page**

4. From the **Scheme** drop-down list select the require scheme, such as **Microsoft Visual Studio** or **Emacs** .

5. Type **content assist** as the filter text to filter the list of keys.



**Figure 2-99. Filtered List of Commands in Keys Preference Page**

6. Select the entry you want to modify and type the new key binding, such as **Ctrl+Alt +C** in the **Binding** text box. You can also assign multiple key bindings.
7. From the **When** drop-down list, select the context in which the key binding apply.
8. Click **OK** .

The **Preferences** dialog box closes.

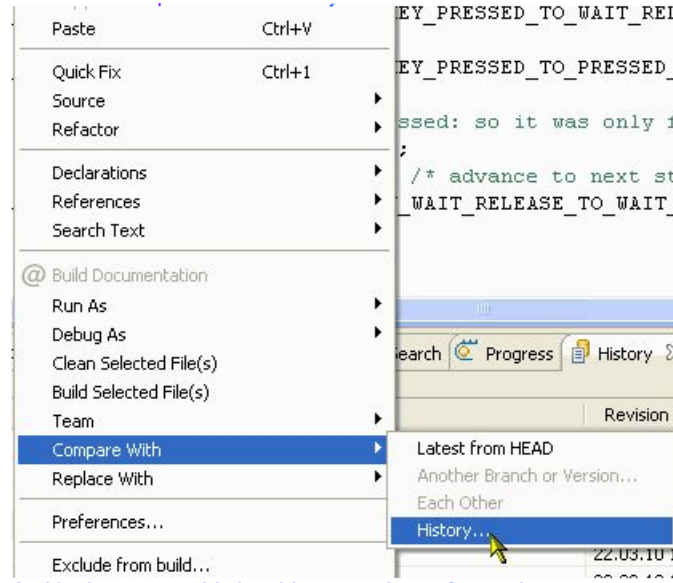
The keyboard shortcut or the keyboard binding are modified, as specified.

### 2.6.10 Is it possible to view the change history of a source code file?

Yes, you can view the history of the changes done in a source file by following the steps given below:

1. Right-click the source file and select **Compare With > History** from the context menu.

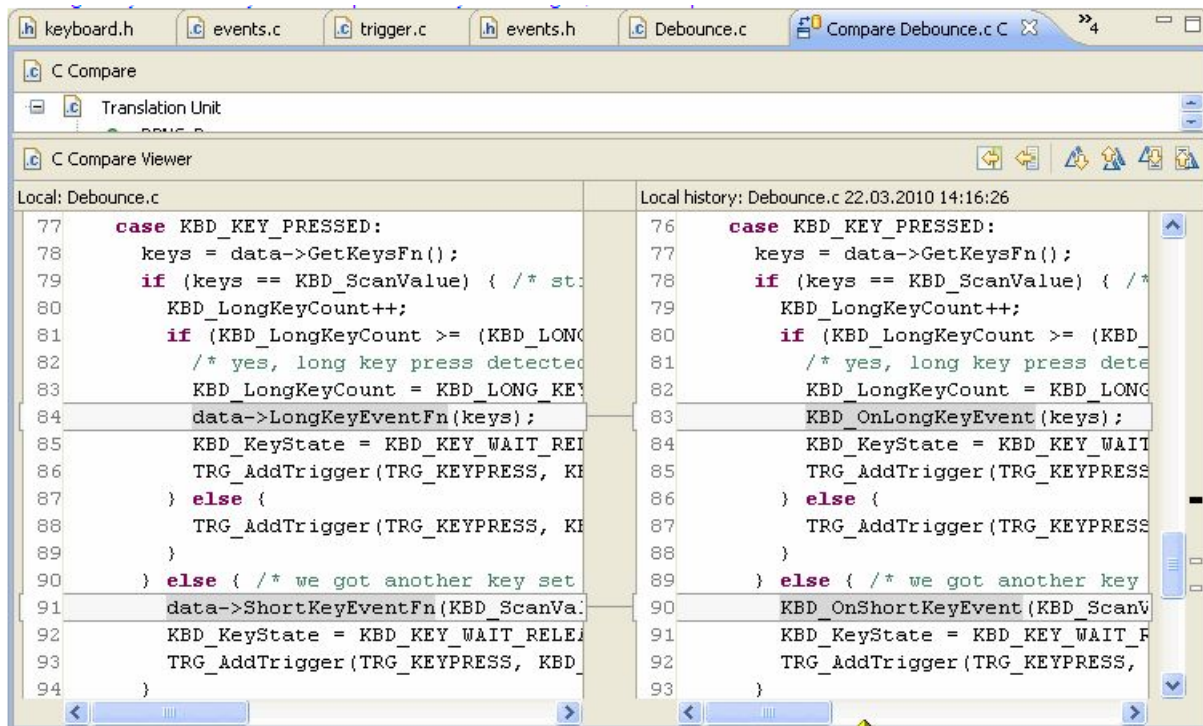
The **History** view opens displaying the date and time when the changes were made in the source file.



**Figure 2-100. Compare With > History**

2. Double-click the required revision time in the **History** view.

The editor area displays the comparison of different file revisions, and also highlights the changes. You can determine the changes done in the source file by browsing through the compared files.



**Figure 2-101. Comparison of File Revisions**

## 2.6.11 How can I view a graphical representation of source code in the CodeWarrior IDE using Dot and Doxygen?

You need to install doxygen, dot/Graphviz, and Eclox eclipse plugins from the following locations respectively:

[www.doxygen.org](http://www.doxygen.org)

[www.graphviz.org](http://www.graphviz.org)

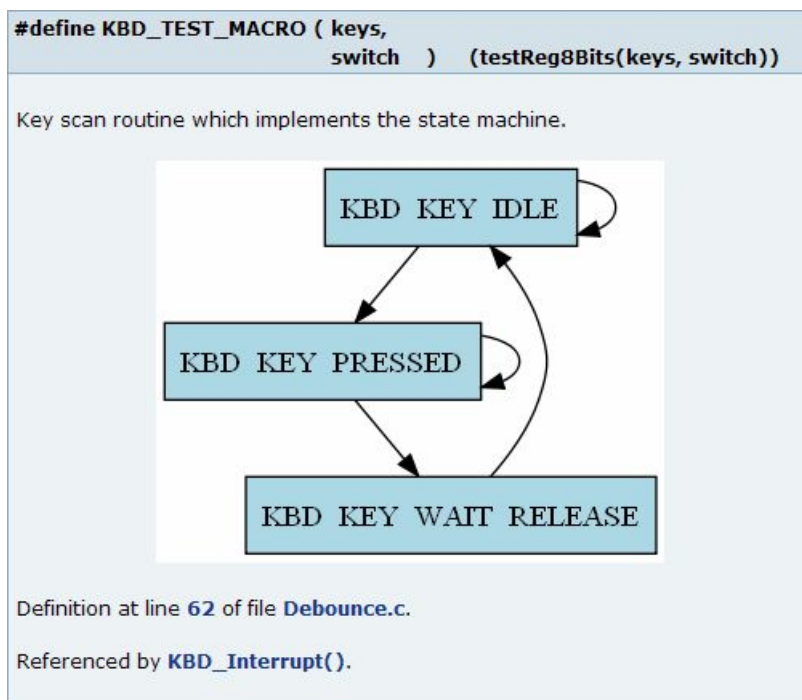
<http://download.gna.org/eclox/update>

While working with Dot+Doxygen, you can view graphical representation of the source code. For example, if you have a state machine, describe the state machine (as shown in the figure, *State Machine Description* listed below) in the source file.

Dot+Doxygen will then display the graphical representation of the state machine, as shown in the figure *Graphical Representation of State Machine* listed below.

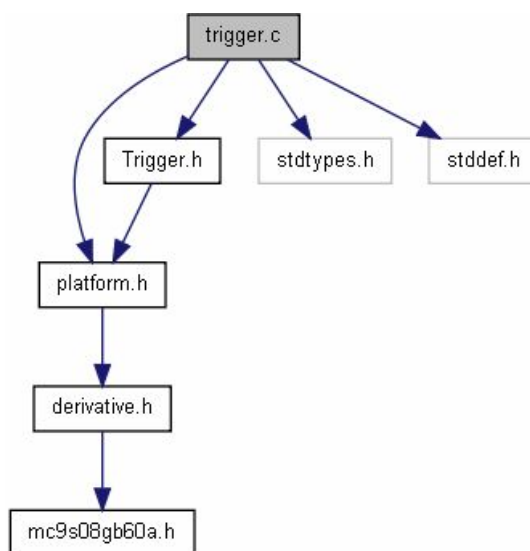
```
42 #define KBD_LONG_KEY_ITERATIONS (KBD_LONG_KEY_TIME_MS/(KBD_KEY_PRESSED_TO_PRESSED_NOF_10MS*10))
43 /*!< Number of iterations we need to go for a long key detection. */
44
45 /*! \brief Key scan routine which implements the state machine.
46 \dot
47 digraph example_api_graph {
48     node [shape=box];
49     KBD_IDLE    [fillcolor=lightblue,style=filled,label="KBD_KEY_IDLE" ];
50     KBD_PRESSED [fillcolor=lightblue,style=filled,label="KBD_KEY_PRESSED" ];
51     KBD_WAIT    [fillcolor=lightblue,style=filled,label="KBD_KEY_WAIT_RELEASE"];
52     KBD_IDLE -> KBD_PRESSED -> KBD_WAIT -> KBD_IDLE ;
53     KBD_PRESSED -> KBD_PRESSED ;
54     KBD_IDLE -> KBD_IDLE ;
55 }
56 \enddot
57 */
58
```

Figure 2-102. State Machine Description



**Figure 2-103. Graphical Representation of State Machine**

Dot+Doxygen also gives you a graphical view of all include file in the source file.



**Figure 2-104. Graphical View of Include Files**

### 2.6.12 How can I determine which header files are associated with my source code file?

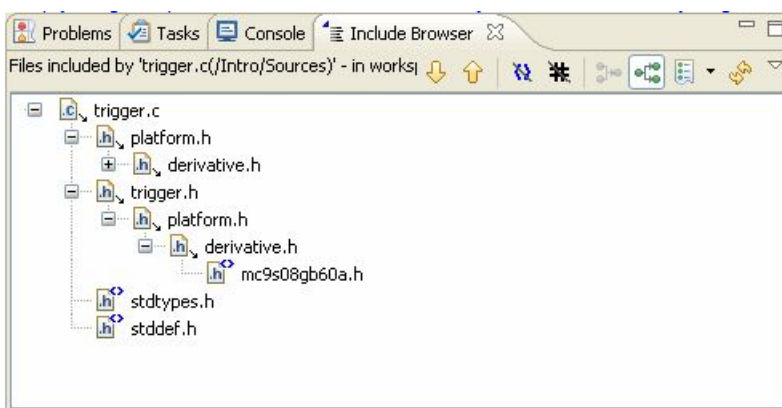
To determine which header files are associated with your source code file, you can view the graphical representation of the include files in the **Include Browser** view.

1. Select **Window > Show View > Other**.
2. Select **C/C++ > Include Browser** in the **Show View** dialog box and click **OK**.

The **Include Browser** view appears.

3. Drag and drop the required source file from the **CodeWarrior Projects** view in the **Include Browser** view.

The **Include Browser** view displays all the include file in the selected source file (refer to the image listed below).



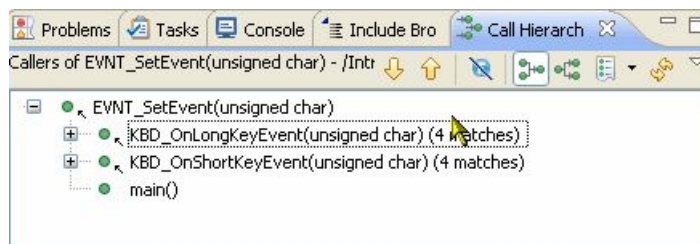
**Figure 2-105. Include Browser View**

### 2.6.13 How do I view call hierarchy in my source code?

To view or browse through the call hierarchy:

1. Right-click on a symbol in the source file and select **Open Call Hierarchy**.

The call graph appears in the **Call Hierarchy** view.



**Figure 2-106. Call Hierarchy View**

2. Click the **Show Callees** icon in the **Call Hierarchy** view toolbar.

The callees appear in the view.

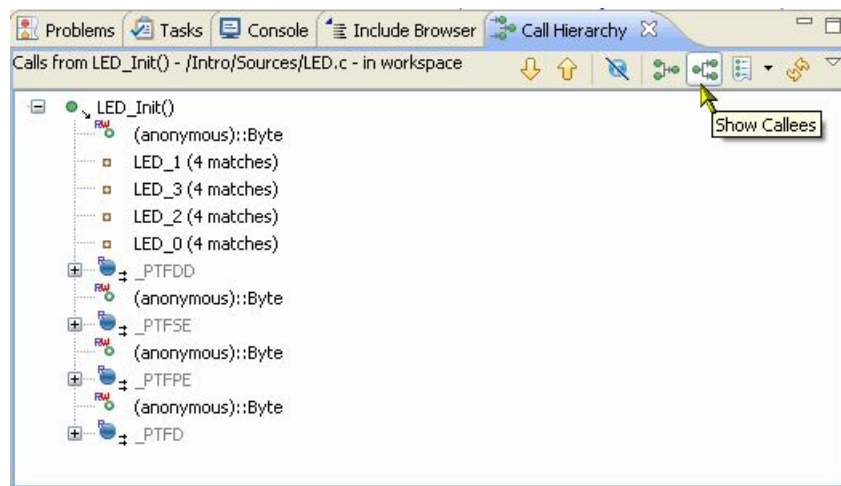


Figure 2-107. Show Callees

### 2.6.14 Why the project that I just created is not visible in the CodeWarrior Projects view?

A newly created project might not be visible in the CodeWarrior Projects view if you choose to use working sets. This is because a new project is not added to the active working set automatically. Thus, to see the new project in your active working set, select **Edit Active Working Set** from the view pull-down menu. From the **Working set content** options, select the newly created project and click the **Finish** button. The new project will now be visible in the active working set.

### 2.6.15 I created a new file in Windows Explorer but the file does not appear in the CodeWarrior Projects view. Why?

The **CodeWarrior Projects** view cannot watch for changes in your computer's file system. When you create a new file away from the Eclipse platform the CodeWarrior application is not aware of the changes. Thus to view the new file, you need to force the **CodeWarrior Projects** view to update its content. To forcefully update contents, select the project whose directory contains the updated file and from the **File** menu bar select the **Refresh** command. Alternatively, you can press the F5 key. The new file will now appear in the **CodeWarrior Projects** view.

## 2.6.16 Why does the Rename option appear grayed out?

The **Rename** command will be enabled only if a file or folder is selected in the **CodeWarrior Projects** view. In case you select a file or folder and click elsewhere on the Workbench, the **Rename** command appears grayed out. So, ensure that you select the desired file or folder (highlighted in blue) before trying to rename it.

## 2.6.17 Why duplicating a configuration in the debugger perspective does not duplicate the run configuration?

In CodeWarrior, the debug and the launch configuration are not associated with a build configuration. Instead, they are associated with an executable and a project. For example, by duplicating the build configuration, the executable remains the same. Therefore, even for the new build configuration, the old launch/debug configurations still apply. This behavior is different than the previous CodeWarrior IDE tools in that respect.

## 2.6.18 Can I still use Ctrl+Tab keys to navigate between open windows?

Yes, you can. When you select Ctrl+Tab, a new floating window appears that lets you navigate between the open CodeWarrior windows.

## 2.6.19 How can I change the debugger key bindings to the ones that I used to have in the Classic CodeWarrior IDE?

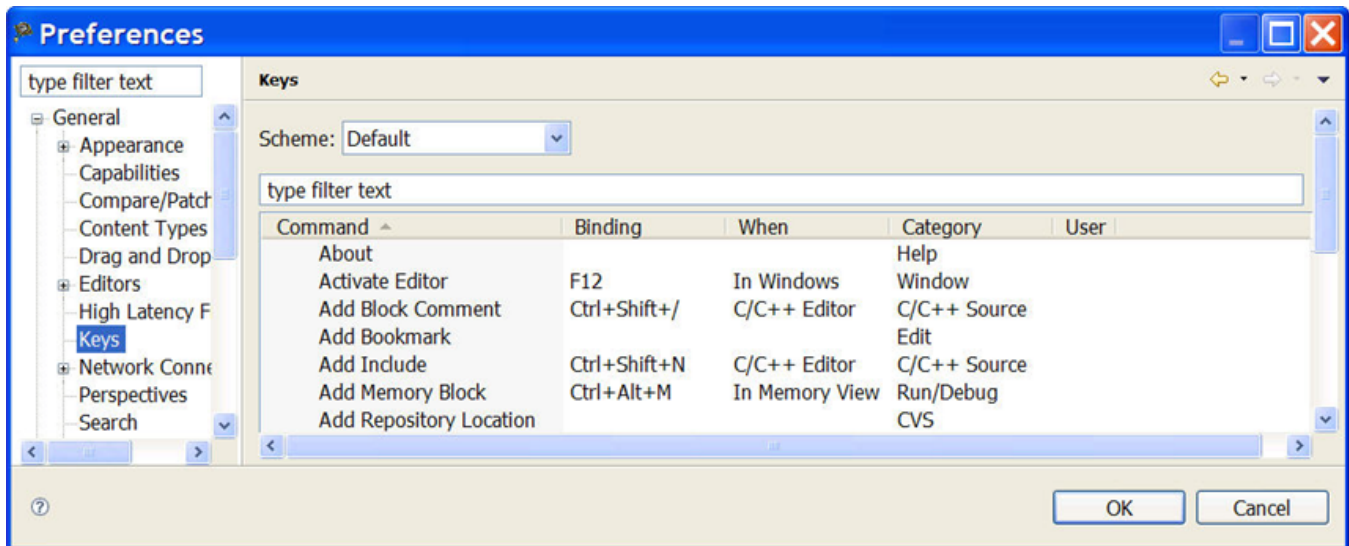
The corresponding default binding for this functionality is Ctrl+F6. You can reassign this feature to Ctrl+Tab by doing the following steps:

1. From the IDE menu bar, select **Window > Preferences**.

The **Preferences** dialog box appears.

2. Select **General > Keys**.



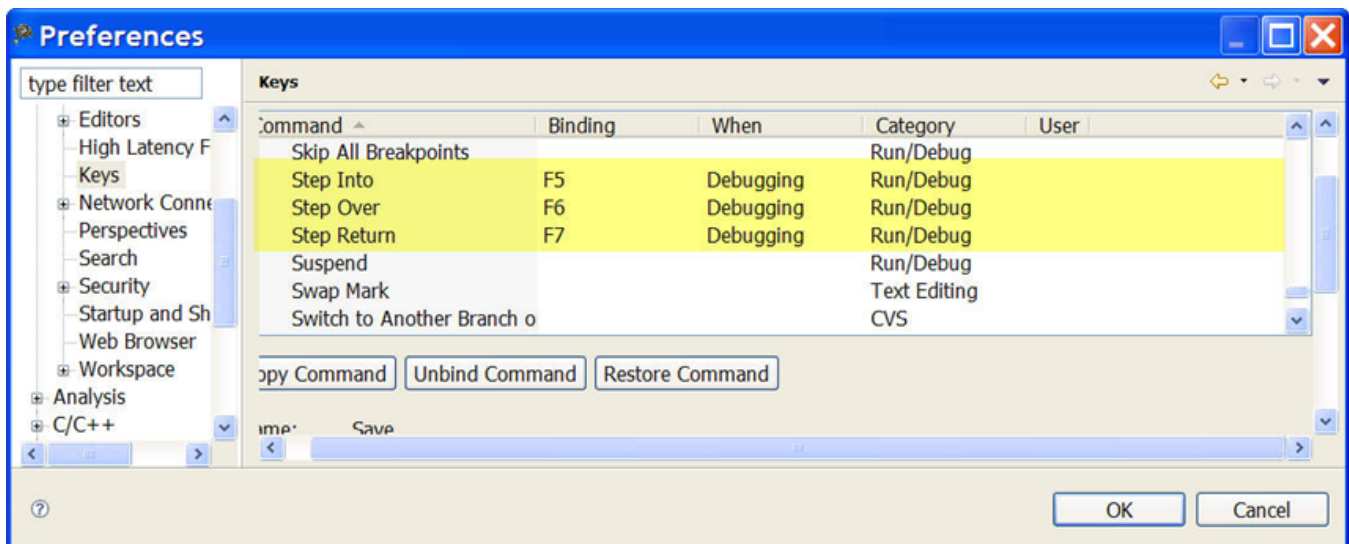


**Figure 2-108. Preferences Dialog Box-Keys Page**

Under the **Command** column is a list of available commands. The **Binding** column contains the keys that can be used to invoke the command. The **When** column indicates when this command will be executed.

3. Click on the **Command** column to get the key sequence in alphabetical order.

For the debugger, the keys to change are **Step Into**, **Step Over** and **Step Return**.



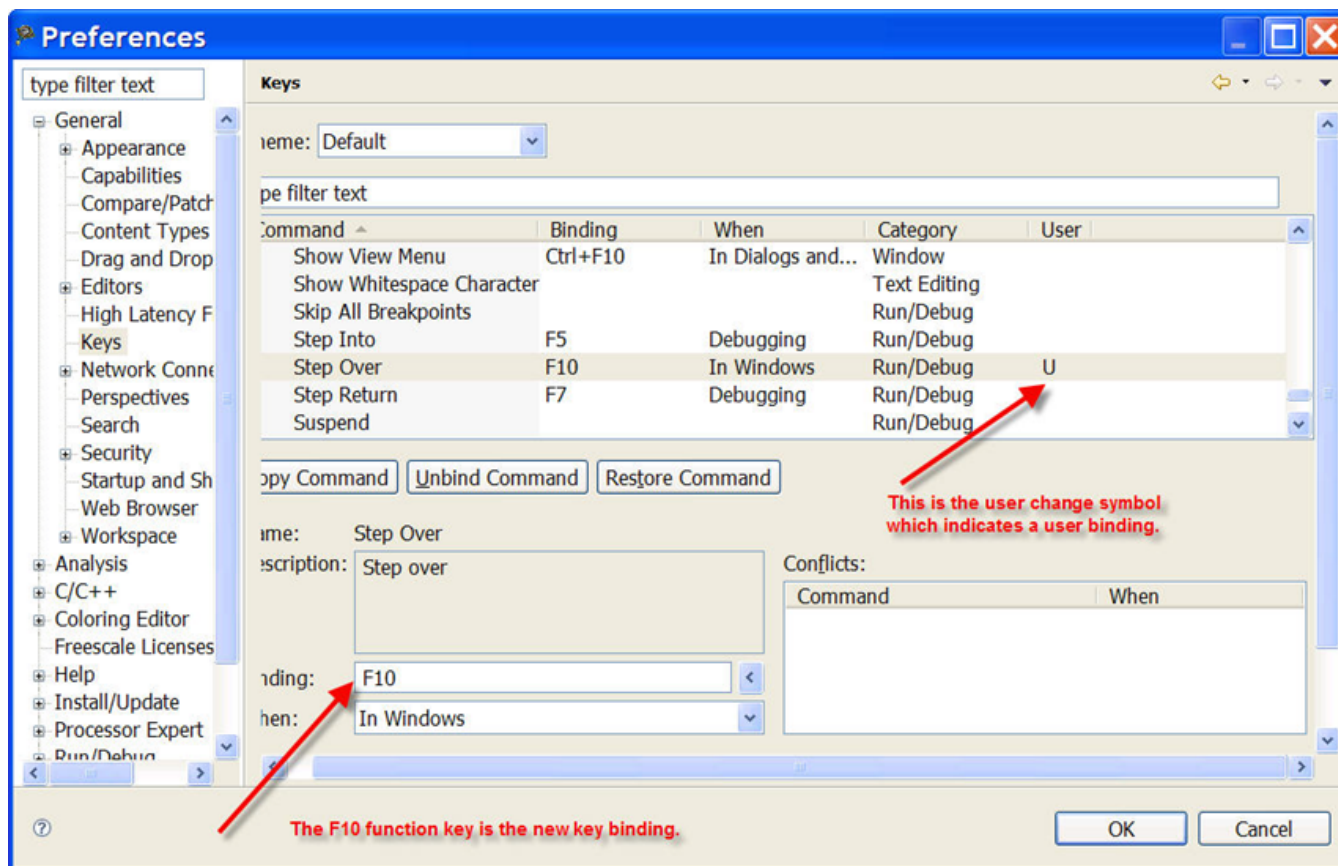
**Figure 2-109. Preferences Dialog Box-Debugger Keys**

By default, these key bindings are set to F5, F6 and F7 respectively. In order to change the key sequence for Step Over from F6 to F10, follow these steps:

- a. Select **Step Over** in the **Command** column.
- b. From the **Binding** textbox, remove F6 by deleting it from the binding text box.

- c. Press F10 on computer keyboard so that F10 will be inserted into Binding textbox.

The user change symbol, which is a blue triangle, will appear in the right-most column to indicate a user binding.



**Figure 2-110. Preferences Dialog Box-New Key Binding**

- d. Click **Apply**.

The Binding for **Step Over** appear as **F10**. When you use debugger, press the **F10**key for a **Step Over**. In the context menu that appears on the Debugger window, you see **Step Over F10**.

- 4. Click **OK**.

The **Preferences** dialog box closes.

The default key bindings changes, as specified.

## 2.6.20 How can I change the name of the executable that is generated when I build my project?

To change the name of the executable that is generated when you build your project, follow the steps given below:

1. Select the project in the **CodeWarrior Projects** view and select **Project > Properties** from the IDE menu bar.

The **Properties** dialog box appears.

2. Select **C/C++ Build> Settings** from the left panel.

The **Settings** page appears in the right panel of the **Properties** dialog box.

3. Select **Build Artifact** tab.

The **Artifact name** textbox contains the name of the executable that will be created when the **Build Project** option is selected. The **Artifact extension** textbox contains the name extension of the binary that will be created. If you want to change the name of the executable, replace the one in the **Artifact name** textbox.

4. Click **Apply** .
5. Click **OK** .

Now when you build the project using the **Build Project** option, the new binary file name appears in the project folder.

## 2.6.21 How can I see printf output in the console window?

Select the board which supports `printf()` output in the console window, for example, K60 board supports the console view for `printf()` output.

In case the **Console** view is not appearing in the IDE view stack, select **Window > Show View > Console** from the IDE menu bar.

The **Console** view appears at the bottom of the IDE and lists the output of the `printf()`.

## 2.6.22 Why the console view does not display all the warning and error messages by default?

By default, only 500 lines are displayed in the console window. In order to change the number of lines displayed in the console window, follow these steps:

1. From the IDE menu bar, select **Window > Preferences** .

The **Preferences** dialog box appears.

2. Select **C/C++ > Build Console** from the left panel.

The **Build Console Settings** page appears.

3. Change the **Limit console output (#lines)** to the number of lines that you would like to appear in the console window. By default, it is set to 500.
4. Click **Apply** .
5. Click **OK** .

The number of lines displayed in the console window changes, as specified.

#### **NOTE**

Since this is a global setting, the console window will display the specified number of lines for every project that is compiled within the workspace.

### **2.6.23 Is there a file that contains all the launch configuration settings?**

Yes there is a file that describes the launch configuration settings. This file is located in the project folder with a **.launch** extension.

### **2.6.24 How the CodeWarrior Project Importer handles recursive access paths?**

The CodeWarrior Project Importer allows the user to automatically have the recursive access paths flattened. For example, if the original project has a recursive access path "`{\Project}\`", the imported access paths could be something like this:

```
"{Project}\"
```

```
"{Project}\src"
```

```
"{Project}\src\subdir"
```

## 2.6.25 Can I disassemble my source code file?

Yes you can disassemble source code files. To create a disassemble file follow steps given below:

1. Right-click the source file

A context menu appears.

2. From the context menu, select **Disassemble** option.

The **Disassemble Job** window appears.

The disassembling file provides a way to show the results of object code produced from a C/C++ source file in the Editor. Once the Disassemble command is executed, it will proceed to compile, disassemble the file and show the resulting disassembled file in a new editor window, titled **sourcefilenameXXXXX.lst**, where **XXXXX** represent random numbers.

## 2.6.26 Where exactly the disassemble file gets created?

This file gets created in the `temp` directory located under:

```
<Profiles>\<username>\<LocalSettings>\<Temp>
```

Under this path you find **sourcefilenameXXXX.lst** file.

## 2.6.27 Why the .metadata folder in my workspace stores a huge history?

This is a standard Eclipse feature. Eclipse keeps a local history of edited files by default. It lets you restore a file content from a previously saved local history by right-clicking a source file and selecting the **Restore from local history...** option. You can configure how the local history is stored. See [How do I update my local history settings?](#)

## 2.6.28 How to have projects from different workspaces or locations in a workspace?

You can have projects which are in different workspaces or locations in your workspace. For this, you can import projects from other workspaces, without copying them into your workspace.

However, keep in mind that you can only have one workspace open in one eclipse instance. If you want to view multiple workspaces side by side, then run another instance of the CodeWarrior software with a different workspace.

## 2.6.29 How can I switch to another workspace?

To switch to another workspace:

1. From the CodeWarrior IDE menu bar, select **File > Switch Workspace > Other** .

The **Workspace Launcher** dialog box appears.

2. Click **Browse** .

The **Select Workspace Directory** dialog box appears.

3. Select the workspace you want to switch to.
4. Click **OK**.
5. Click **OK**.

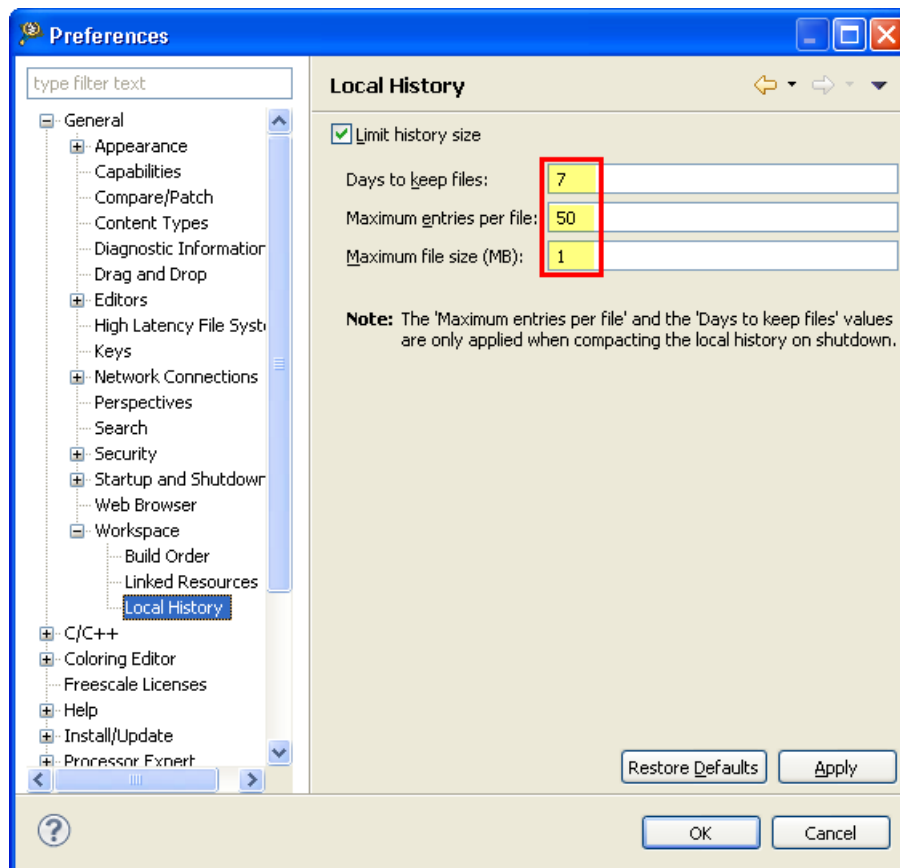
The selected workspace appears in the **CodeWarrior Projects** view.

## 2.6.30 My workspace takes long to close. How can I resolve this?

The reason behind this might be that the CodeWarrior has Eclipse based IDE, and it keeps a history of changes within your workspace. Using **Team > Show Local History** you can compare your changes and go back and forward in history. But sometimes it slows down the IDE, especially if there are a lot of local history files. To resolve this problem, check your workspace for the size and number of files in the following folder:

```
.metadata\.plugins\org.eclipse.core.resources\.history
```

In case you notice there are lots of files, and if you are not interested in the history, or you want to limit the number of files to improve the performance, select **Window > Preferences > General > Workspace > Local History** (refer to the image listed below) and reduce the number (e.g. everything to 1).



**Figure 2-111. Preferences Dialog Box - Local History Page**

This reduces the history information to the minimum every time you exit the workspace.

A more radical approach is to delete (e.g. with the **Windows Explorer**) the `.history` folder, from the following location:

```
.metadata\.plugins\org.eclipse.core.resources\.history
```

Applying the above settings, will improve your workspace performance.

## 2.6.31 What is the purpose of the Tasks view?

The **Tasks** view lets you view tasks that have been created.

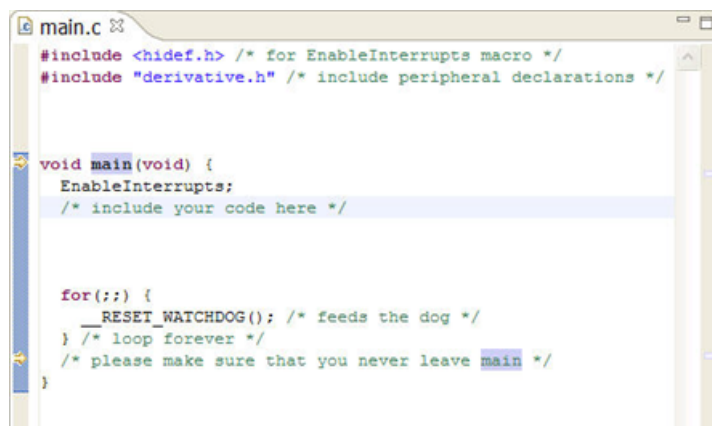
### NOTE

The task does not refer to operating system related process or task, but with a (engineering) work task.

A task could be created to record reminders or follow-up on something later. In order to create a **Task**, select **Add Task** in the **Task** view. Tasks can also be linked to a resource that would allow you to write a note to yourself so that it can be executed or looked at a later time. In order to create a new task that is associated with a resource follow steps given below:

1. Double-click the source file.

Source file appears in the Editor view (refer to the image listed below).



```
main.c
#include <hidef.h> /* for EnableInterrupts macro */
#include "derivative.h" /* include peripheral declarations */

void main(void) {
    EnableInterrupts;
    /* include your code here */

    for(;;) {
        __RESET_WATCHDOG(); /* feeds the dog */
    } /* loop forever */
    /* please make sure that you never leave main */
}
```

**Figure 2-112. Source Code File**

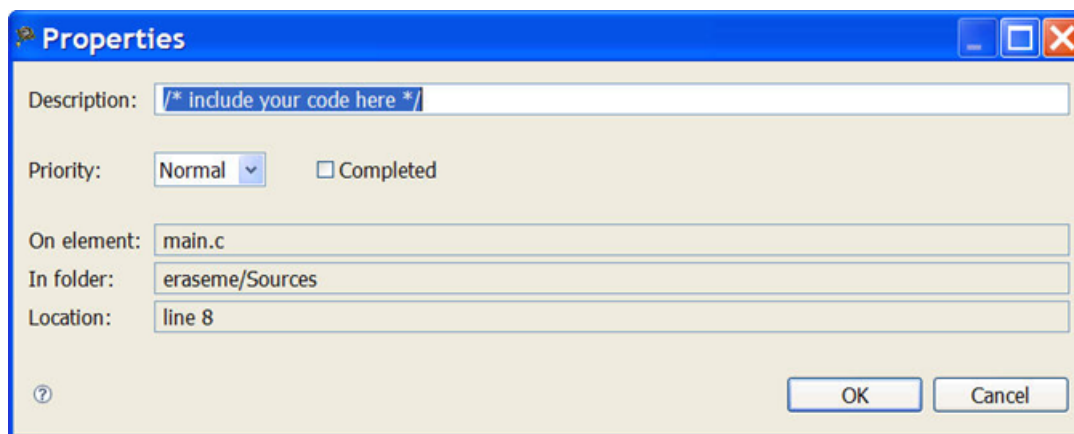
2. Right-click the left-hand side of source line where a task should be added.

A context menu appears.

3. From the context menu, select **Add Task**.

The **Properties** dialog box appears.





**Figure 2-113. Properties Dialog Box**

4. Type a description of what should be done with this task in the **Description** textbox.
5. Click **OK**.

The new task appears in the **Task** view.

Alternatively, you can also add a comment block with a `\todo` tag in the source code file. For example, following comment block in the source code file creates a new task in the **Tasks** view.

```
/*! \todo I need to fix this bug here... */
```

## 2.6.32 What is the purpose of the Properties view?

The purpose of the Properties view is to display property names and the basic properties of a selected resource. The kind of properties that are displayed depends on the specific resource selected as well as the features and plug-ins that are installed on the Workbench.

## 2.6.33 What is the purpose of the Outline view?

The Outline view displays an outline of a structured file that is currently open in the editor area, and lists the structural elements.

## 2.6.34 How do I use the Outline View to get an overview of a source file and modify it directly?

The following listed are some useful tips to get an overview of the currently selected source file and to directly modify it using the **Outline** view:

- Open the outline view (if not already open) by selecting the **Window > ShowView> Outline**.
- To get an overview about the currently selected source file (for this example, the `Radio.c` source file is used):
  - The red 'S' symbol above an icon indicates that the object is static (S) or external (no red `S').
  - Clicking the items in the **Outline** view jumps directly to the object in the source. This makes navigation really easy in the source file.
  - Using the **Include Browser** on the includes gives a nice view about what is included from my main module:
  - Similar options exists as well for variables and functions. You can jump to variable declarations and definitions, you can refactor (globally rename objects) or check who is using what. On variables you even can directly place a watchpoint which makes this view a standard view for me while debugging.
- To directly modify the source file:
  - The view shows you the order of objects in the source file, so it is easy with this view to change the order of objects. To understand it, consider the following example,

```
RADIO_SniffPacket() is implemented after RADIO_DataIndicationPacket(). Now if you want to call RADIO_SniffPacket() from RADIO_DataIndicationPacket(), you need to make sure there are the appropriate prototypes declared, or RADIO_SniffPacket() is implemented earlier in the file than RADIO_DataIndicationPacket().
```

For the later you could move the implementation in the file. But instead doing this manually, you can simply drag & drop (move) items in the **Outline** view.

## 2.6.35 How can I find out if certain files contain debug information?

To find out if certain files contain debug information, see the file listing in the CodeWarrior Project view.

### 2.6.36 How can I resolve the following error message that I get when I start the CodeWarrior IDE?

```
A debug or protocol-plugin license for the product expired  
or was not found
```

To resolve this error message, add the path to the V10.x license file to the LM\_LICENSE\_PATH environment variable. In addition, it is also possible that the launch config file is corrupted.

### 2.6.37 How can I open an existing project in the CodeWarrior IDE?

You can open only those projects that are already in your workspace and have been closed. To bring an external project (one that is created on another machine or in another workspace) into your workspace you can follow the steps given below:

1. From the C/C++ Perspective toolbar, select **File > Import** .

The **Import** dialog box appears.

2. Select **General Folder > Existing Project into Workspace**
3. Click **Next** .

The **Import Project s** dialog box appears.

4. Click **Browse** .

The **Browse For Folder** dialog box appears.

5. Select **Project to import**
6. Click **OK** .

The imported project appears under Projects in the Import Projects window.

7. Click **Finish** .

The selected project is imported into the workspace.

### 2.6.38 What is the Manage Configurations button in the C/C++ perspective toolbar used for? Could it be replaced by the Properties button?

It can not be replaced by the Properties button. The **Manage Configuration** button displays the list of defined configurations for a given project. You can select a button to add, rename, or remove a configuration. You can also set an active configuration.

### 2.6.39 How can I start the post-build steps in the CodeWarrior IDE?

In order to start a batch file after build, you just need to specify the name of the .bat file in the Command for post-build step.

Note that the current directory is the **Build location** specified in the project properties **C/C++ Build** panel. Therefore, you need to adjust the path to retrieve the .bat file relative to this location.

For example, consider that:

- the project is in `D:\Freescale\Work\Eclipse\testbss`
- the build location is `D:\Freescale\Work\Eclipse\testbss\Debug_SC3x50_PACC`
- the batch file is located in `D:\Freescale\Work\Eclipse\testcbe\testbss` and is called `postb.bat`.

Then, the post-build command is `..\postb.bat`.

### 2.6.40 Is it necessary to have the project name identical to the name of the directory that contains the .project file?

Yes, it has to be located in the folder with the same name as the project.

### 2.6.41 Where does the CodeWarrior IDE save the debug configuration as a local file by default?

It goes inside the .metadata subfolders and then it becomes user dependent. Perform a search on the launch and you will still find it. If you create a new launch configuration, it is saved as a local file by default.

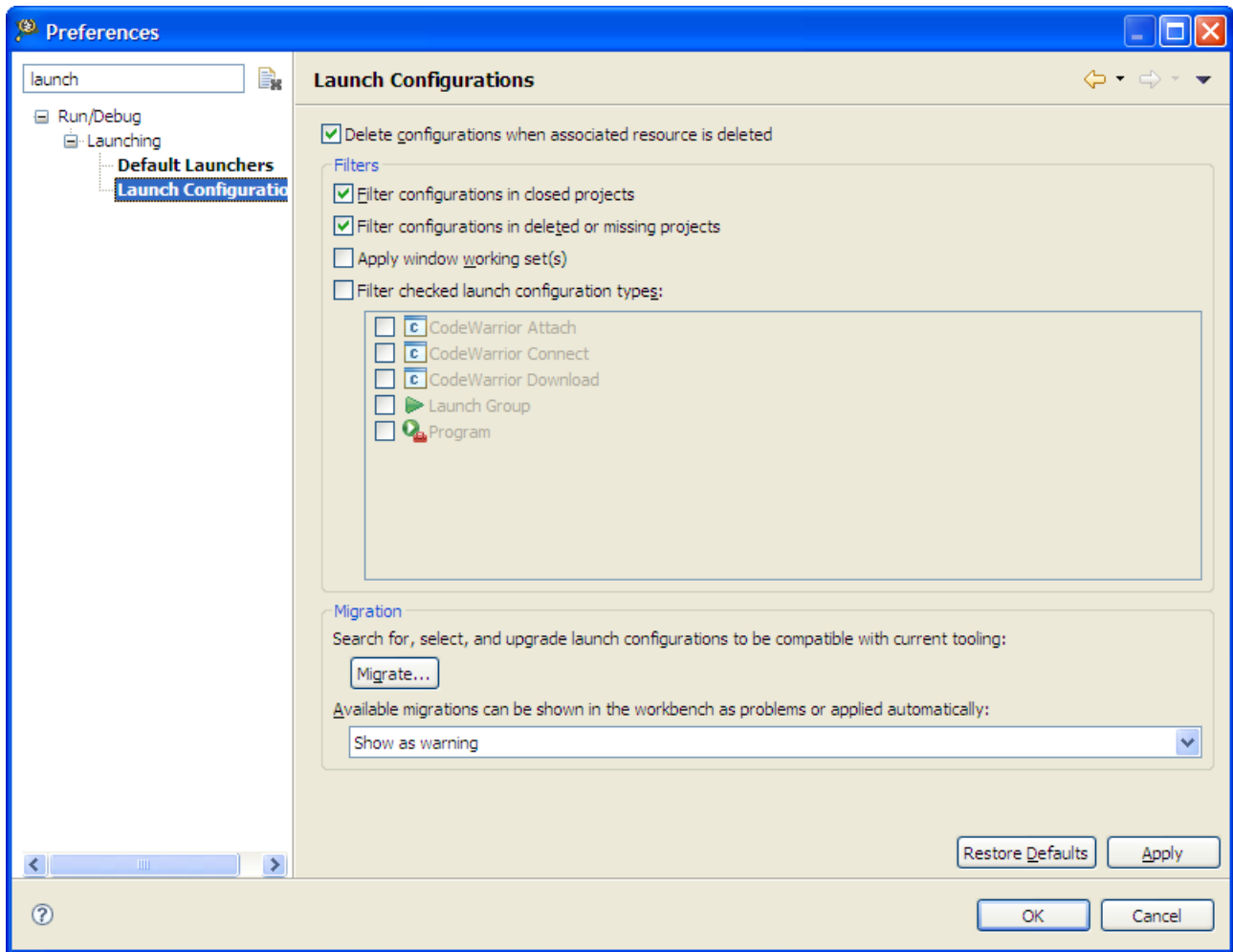
### 2.6.42 When I save my launch configurations as a local file and then delete the project, all the local configurations get deleted as well. How can I resolve this issue?

Follow the steps given below:

1. From the IDE menu bar, select **Window > Preferences** .
2. Select **Run/Debug> Launching > Launch Configurations** .

The **Launch Configuration** settings appear.

3. Make sure that the **Delete configurations when associated resource is deleted** checkbox is cleared.



**Figure 2-114. Preferences Dialog Box - Launch Configuration**

4. Click **OK** .

Now, the configurations will not be deleted when the associated resource is project.

### 2.6.43 How can I modify and save the files in GBK encoding using the CodeWarrior IDE?

In order to be able to save these files, follow the steps given below:

1. From C/C++ perspective toolbar, select **Window > Preferences** .

The **Preferences** window appears.

2. Select **General > Workspace** .

3. Under **T ext file encoding** , type **GB18030** in the **Other** textbox.

4. Click **Apply** .

5. Click **OK** .

Now, you can save the files in GBK encoding.

### **2.6.44 How are the PARENT-COUNT-MyVariable definitions defined in the CodeWarrior IDE?**

The Linked resource variable PARENT can be used to define path variables relative to a parent directory of another one. For example, a path variable FOO can be set to `${PARENT-2-PROJECT_LOC}/foo`, which is equivalent of setting it to `${PROJECT_LOC}../../foo`. The reason of this strange syntax is for the Eclipse compatibility constraints. You can not have the `"..."` characters inside a path variable value (at the resource level).

### **2.6.45 Is there a way to instruct the CodeWarrior IDE to use relative paths instead of absolute ones to store the project file location in a workspace?**

Yes. When you create the project in the default location which is the workspace directory, no `.location` file gets created and you just have to zip the project and the workspace. Before you create the zip file make sure that you remove the following:

- The project build directory since there will be absolute paths in the `.mk` and `.args` files.
- The workspace `.metadata\plugins\org.eclipse.ltk.core.refactoring` directory.
- The workspace `.metadata\plugins\org.eclipse.ltk.core.resources\history` directory.
- The `.PDOM` files from `.metadata\plugins\org.eclipse.cdt.core` directory.

### **2.6.46 How do I solve the probable mismatch of the OSJTAG Firmware with P&E Virtual Serial Port Utility?**

The P&E Toolkit on Freescale website DOES NOT WORK with the latest Firmware of the JTAG Micro. The available version in Freescale website is version 2.01, and the P&E has version 2.05.

You can get the latest firmware updater, P&E Toolkit by accessing the following link (works for OSJTAG as well):

[www.pemicro.com/osbdm](http://www.pemicro.com/osbdm)

The available version in the CDs and Freescale website, installs under **P and E Kinetis Tower Toolkit**, whereas the working version installs as, **P and E OSBDM OSJTAG VIRTUAL SERIAL TOOLKIT**.

### 2.6.47 I am using CW10.1SE and MQX3.7 with all patches and updates. How can I solve this problem?

**While trying to run or debug the MQX example "Hello World" program, I got the error message, "Failed to resume target process, Unable to use plugin \"Cold Fire RTOS debugger\" \ no valid license key was found." Under the MQX tab, in MQX Task Aware Plugin shows that the DLL not installed.**

The solution for this problem is that, if you do not have the professional edition, you need to disable the OS awareness (set to none). To disable the OS awareness, follow the steps listed below:

1. Select **Run > Debug Configurations** from the IDE menu bar.
2. Expand the **CodeWarrior Download** tree control and select the desired launch configuration.
3. Click the **Debugger** tab.
4. Click the **OS Awareness** tab.
5. Select **None** from the **Target OS** drop-down list.
6. Click **Apply** to save the changes.

The OS awareness is disabled.

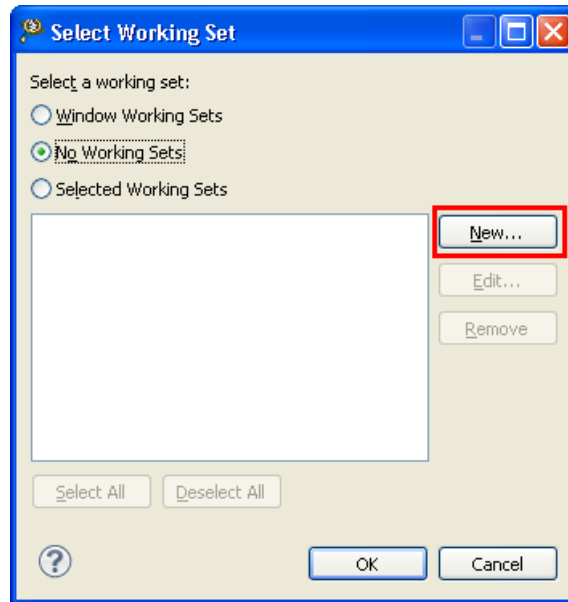
### 2.6.48 What is a Working Set? How do I build/compile a Working Set?

The Working Set is a CodeWarrior eclipse IDE feature, that allows you to group elements for display in views. With that, you can do operations on a set of elements in that working set. Especially when you are using many projects the same time, working sets are a big help to focus on the right set of things at a time.

To build/compile a Working Set:

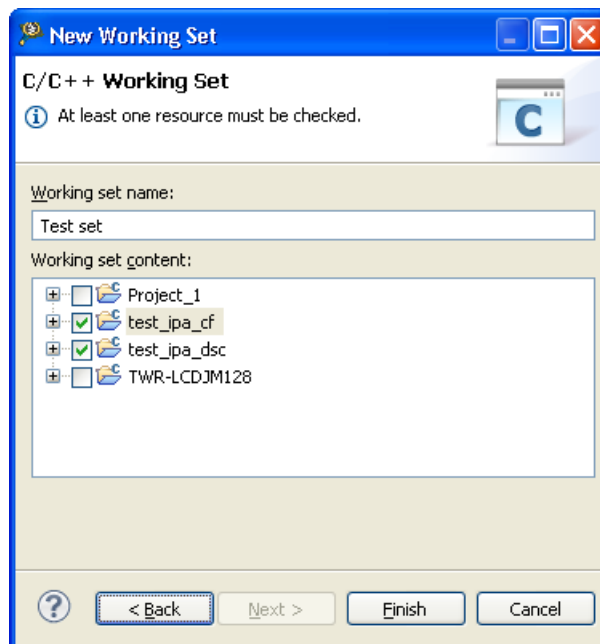


1. Select **Project > Build Working Set > Select Working Set** from the IDE menu bar.  
The **Select Working Set** dialog box appears.



**Figure 2-115. Select Working Set Dialog Box**

2. If you have no Working Set defined, then click **New**.  
The **New Working Set** dialog box appears.
3. Select **C/C++** and click **Next**.  
The **C/C++ Working Set** page appears.

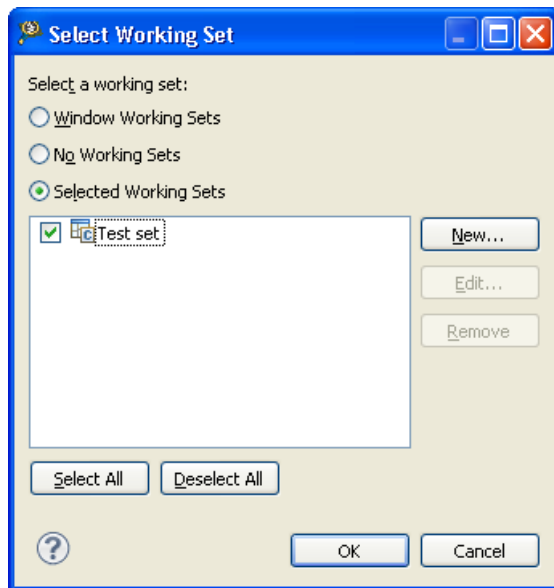


**Figure 2-116. New Working Set - C/C++ Working Set Page**

- Specify a name in the **Working set name** textbox, and select the desired projects that you want to include from the **Working set content** list.
- Click **Finish**.

The **New Working Set** dialog box closes and the newly created working set appears in the **Select Working Set** dialog box.

- Select the newly created Working Set.



**Figure 2-117. Select Working Set Dialog Box with Newly Created Working Set**

- Click **OK**.

The IDE builds/compiles the projects included in the Working Set.

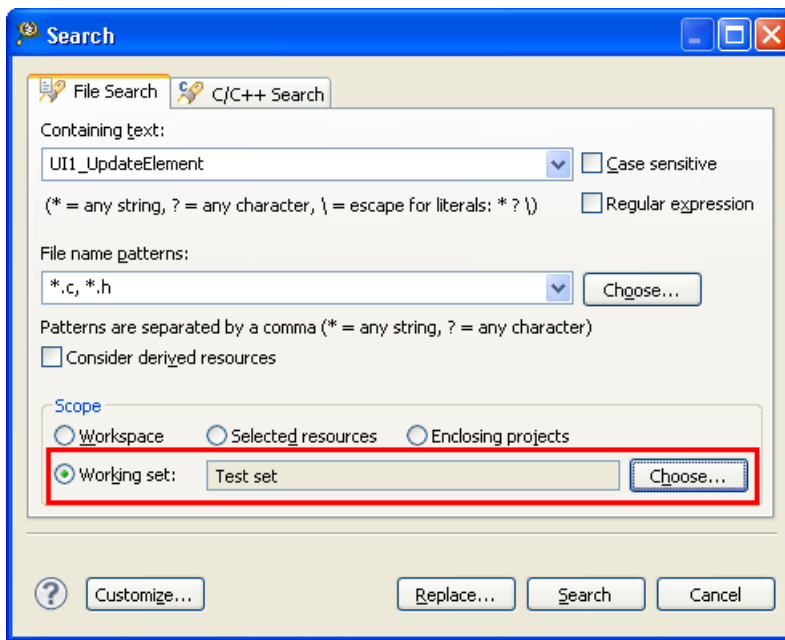
## 2.6.49 How do I search within the Working Set?

To search within a Working Set:

- Select **Search > Search** from the IDE menu bar. Alternatively, press **Ctrl+H**.

The **Search** dialog box appears.

- Specify the desired settings in the **Containing text** and the **File name patterns** textboxes.



**Figure 2-118. Search Dialog Box**

3. Select **Working set** from the **Scope** group and click **Choose** to select the desired working set where you want the IDE to conduct the search.
4. Click **Search** .

The IDE conducts the search and lists the result in the **Search** view.

### 2.6.50 How do I manage the Working Sets?

To manage the Working Sets:

1. Select **Project > Build Configurations > Manage Working Sets** from the IDE menu bar.

The **Manage Working Set Configurations** dialog box appears listing the available working sets.

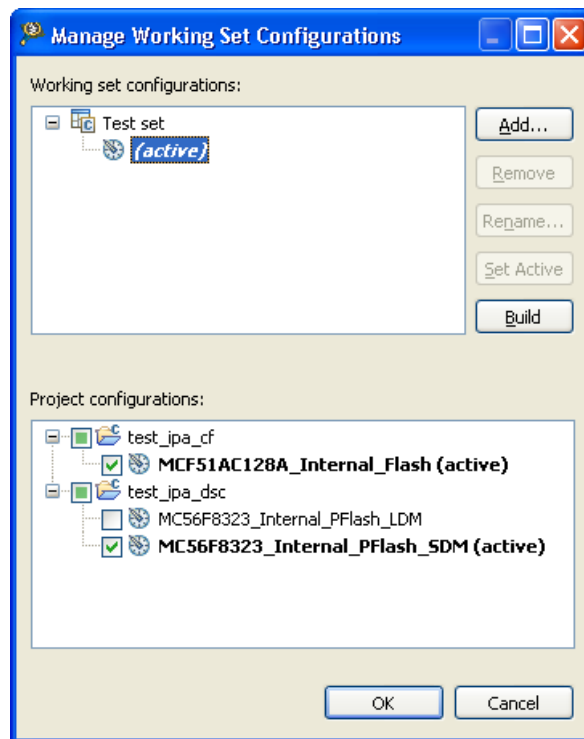


Figure 2-119. Manage Working Set Configurations Dialog Box

Using this dialog box, you can add, remove, rename, set active, and build the desired Working Sets.

### 2.6.51 How can I can filter the projects listed in the CodeWarrior Projects view based on Working Sets?

To filter the projects listed in the **CodeWarrior Projects** view based on working sets, you have to click the *inverted triangle* from the toolbar and select the **Select Working Set** or select one from the most recently used listed working sets.

The following figure shows the inverted triangle menu to filter the projects listed in the **CodeWarrior Projects** view based on working sets:

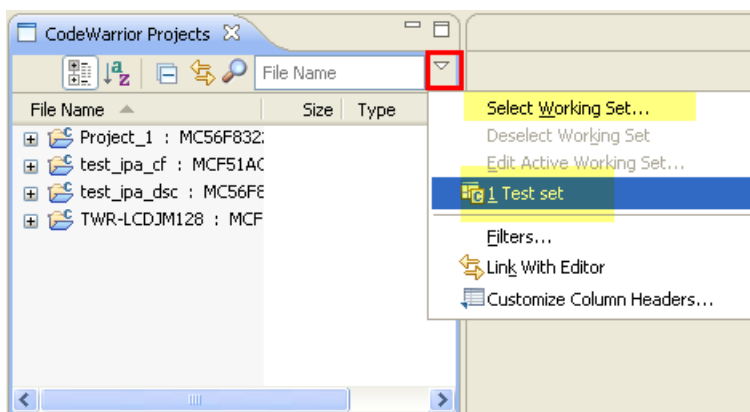


Figure 2-120. CodeWarrior Projects View - Select Working Set

## 2.6.52 How can I import/export a Working Set in the CodeWarrior eclipse IDE?

To import/export a Working Set in the CodeWarrior eclipse IDE, you have to install AnyEdit Tools plug-in provided by Andrey Loskutov. This plug-in allows you to:

- Import and Export of Eclipse Working Sets
- Text Conversion and Sorting actions
- Extended Compare and Replace
- Save Console View
- Open File under Cursor

To install AnyEdit Tools plug-in:

1. Select **Help > Install New Software** from the IDE menu bar.

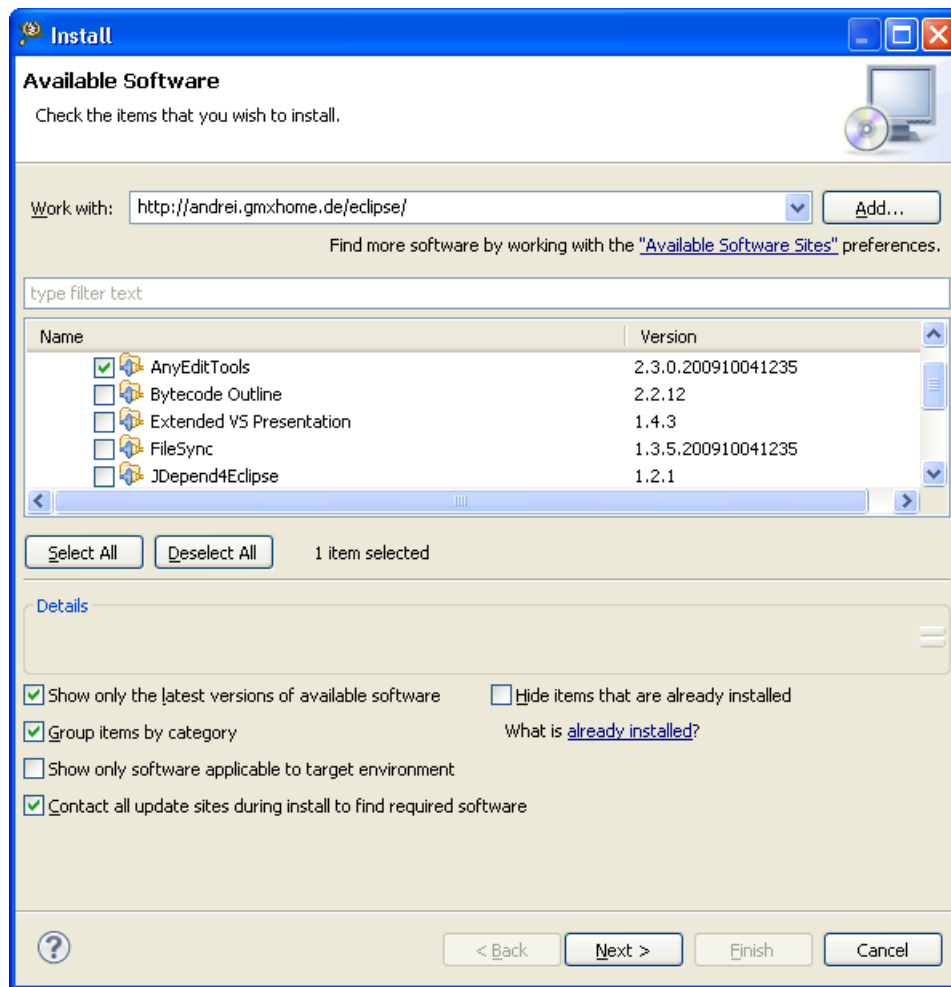
The **Install** dialog box appears.

2. Enter the following web address in the **Work with** textbox and press *Enter*:

`http://andrei.gmxhome.de/eclipse/`

The available plug-in for various eclipse versions appears in the list.

3. Expand the tree-control of the desired eclipse version and select **AnyEditTools**.



**Figure 2-121. Install Dialog Box - AnyEditTools Plug-in**

4. Click **Next**.

The **Install details** page appears.

5. Select **AnyEditTools** from the list.

6. Click **Next**.

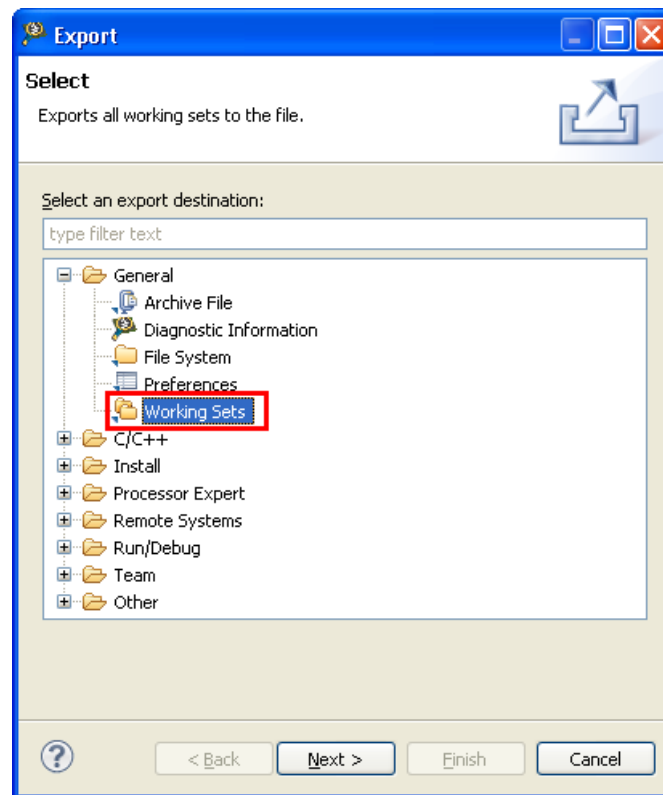
7. Select the option to accept the license agreement.

8. Click **Finish**.

The IDE installs the AnyEdit Tools for your CodeWarrior Software.

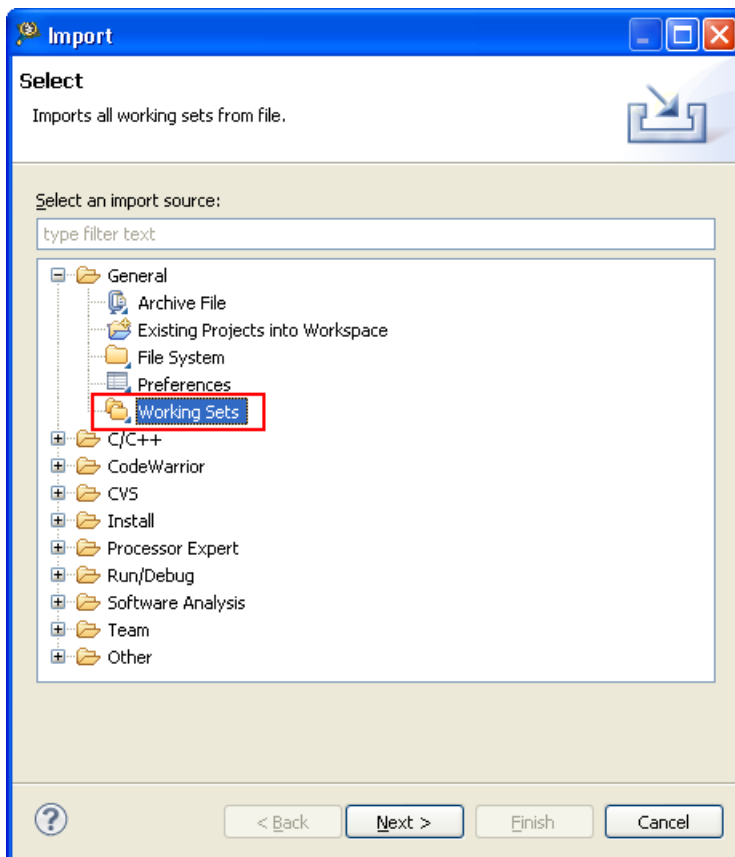
The AnyEdit Tools includes an importer and exporter File action for Working Sets. To export and import the Working Sets:

- Select **File > Export** from the IDE menu bar and follow the onscreen instructions to export a working set. The following figure shows the **Export** dialog box with **Working Sets** option:



**Figure 2-122. Export Dialog Box with Working Sets Option**

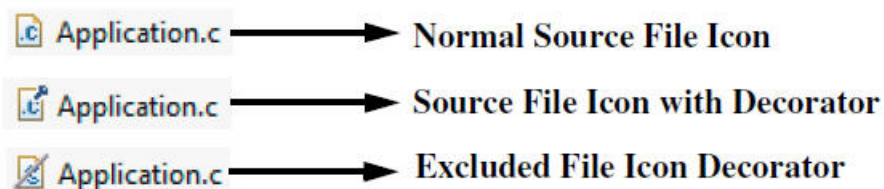
- Select **File > Import** from the IDE menu bar and follow the onscreen instructions to import a working set. The following figure shows the **Import** dialog box with **Working Sets** option:



**Figure 2-123. Import Dialog Box with Working Sets Option**

### 2.6.53 What are Icon decorators?

The icon and label decorators are the symbols used for annotating the icon or label with the useful information. The following figures shows the example of source file icon decorators:



**Figure 2-124. Example - Decorator for Application.c File**

### 2.6.54 How do I enable or disable an icon decorator?

To enable/disable an icon decorator:

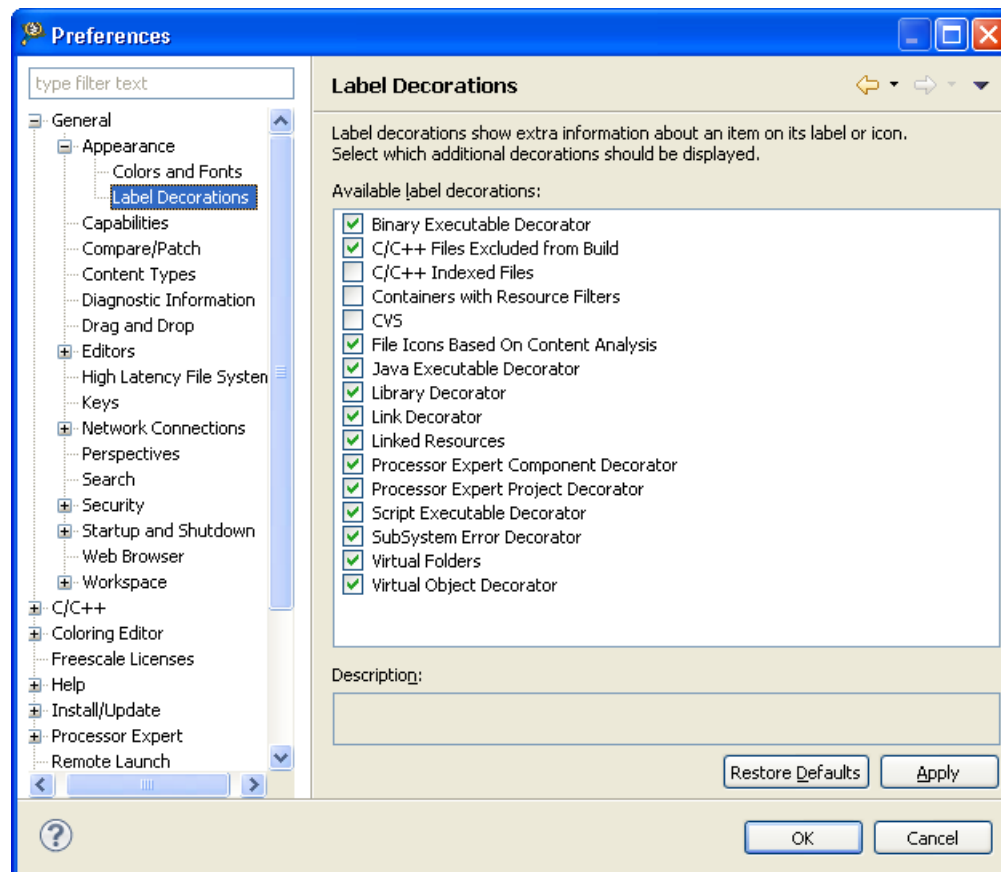


1. Select **Window > Preferences** from the IDE menu bar.

The **Preferences** dialog box appears.

2. Expand the **General** tree control and select **Label Decorations** from the left panel.

The **Label Decorations** page appears in the right panel. The following figure shows the **Preferences** dialog box with **Label Decorations** page:



**Figure 2-125. Preferences Dialog Box - Label Decorations Page**

3. Check or clear the checkboxes for the desired label decorators to enable or disable the decorators respectively.
4. Click **Apply** to save the changes.
5. Click **OK** to close the dialog box.

The IDE applies and saves the changes you selected for the icon decorations.

## 2.6.55 Can the icon decorators be combined?

Yes, the icon decorators can be combined. The following figure shows an example:

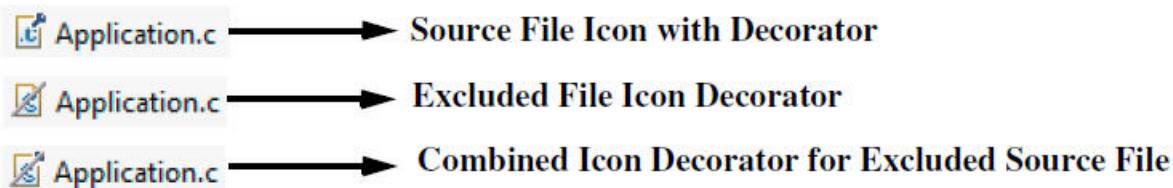


Figure 2-126. Example - Combined Icon Decorators

### 2.6.56 When I open a large file in CodeWarrior IDE, an Editor Scalability dialog box appears, listing the message, "You are opening a large file. Scalability mode has been turned on for this editor to help improve performance by disabling features requiring to parse the source file language." What does it mean?

While processing a file in the editor, there are various other processes that runs in the background, this includes *Eclipse Indexer*, *Outline View* or *Syntax Coloring*.

When you open a large file, the IDE tends to get slower to use more memory to process that file. To avoid this, the IDE turns on the *Scalability Mode*, and close the functions like, *Eclipse Indexer*, *Outline View* or *Syntax Coloring* to increase the performance.

### 2.6.57 How do I configure the Scalability settings?

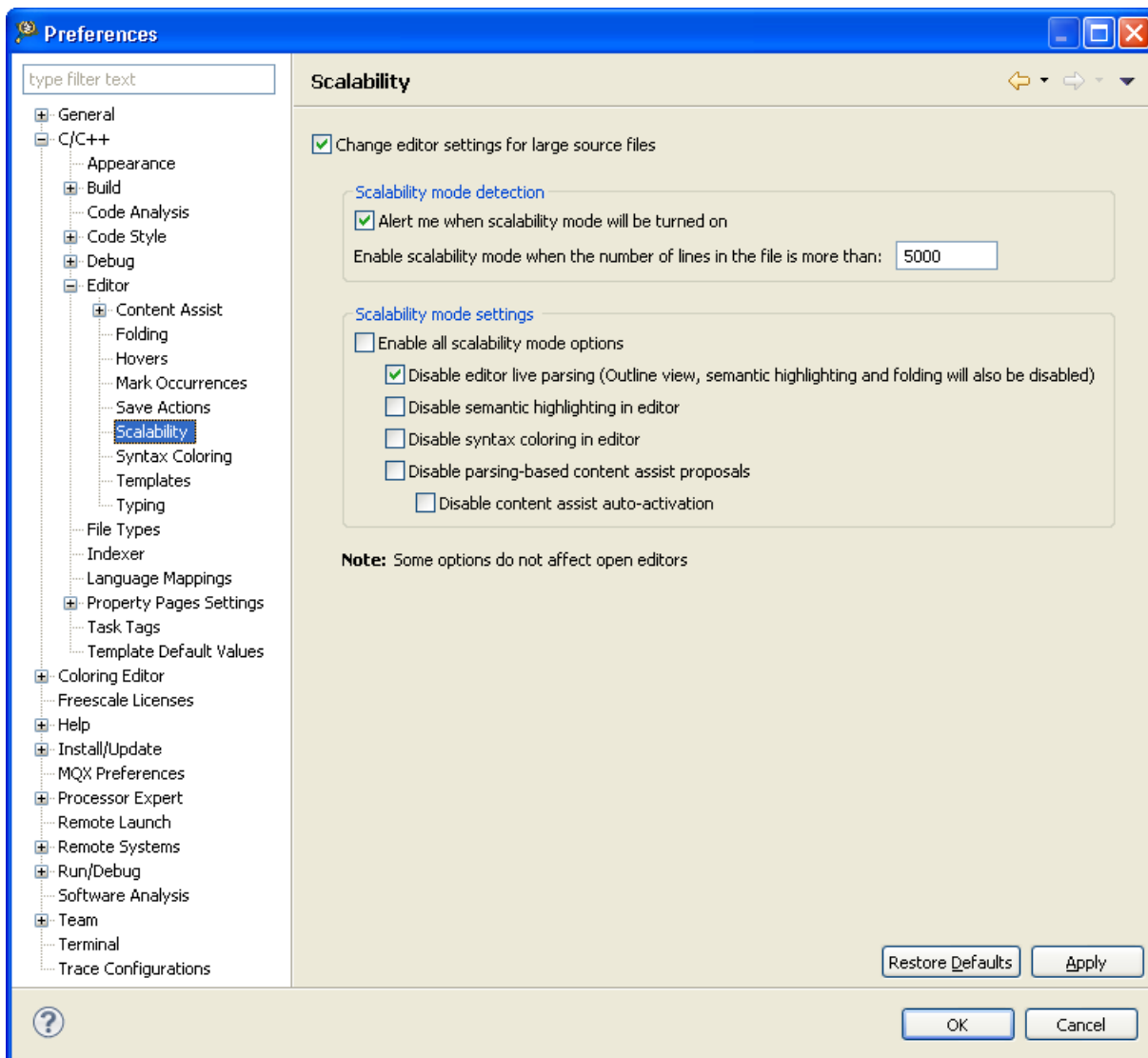
You can configure the Scalability Mode to be enabled only for very large files. To configure the **Scalability** settings:

1. Select **Window > Preferences** from the IDE menu bar.

The **Preferences** dialog box appears.

2. From the left panel, expand the **C/C++ > Editor** tree controls and select **Scalability**.

The **Scalability** page appears in the right panel.



**Figure 2-127. Preferences Dialog Box - Scalability Page**

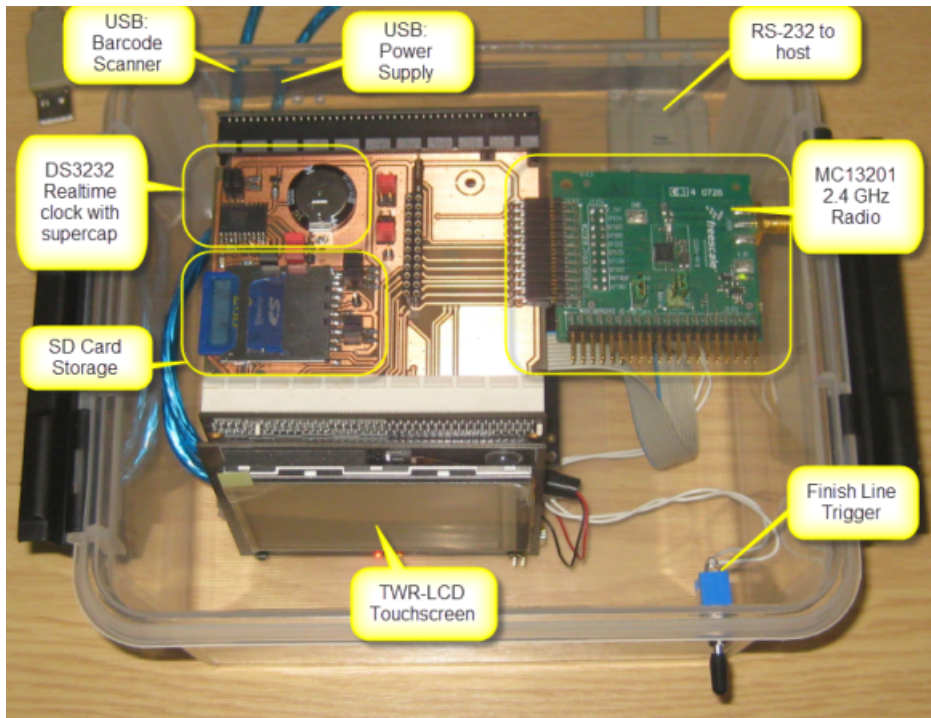
3. Specify the desired settings in the **Scalability mode detection** and the **Scalability mode settings** groups.
4. Click **Apply** .
5. Click **OK** .

The **Preferences** dialog box closes. The IDE saves and applies the changes you specified.

## 2.6.58 What is Timing System?

The Timing System uses the TWR-MCF52259, TWR-SER, TWR-LCD and a custom PCB. The whole system fits into a box. An SD Card with the FatFS file system is used for storage (see as well FatFs with Kinetis). For precise timing a DS3232 realtime clock

with super-cap is integrated. The PCB it provides a wireless link using the MC13201 2.4 GHz transceiver. The power is provided through USB, so either a notebook, a battery pack or a USB wall power supply can be used.



**Figure 2-128. Timing System in a Lunch Box**

The main CPU runs a Processor Expert based USB stack: it reads in data from a USB Laser Bar-code scanner. The Bar-code scanner is used to register the runner number after they have passed the finish line. To trigger the finish line time, several options can be used: push button switches or multiple laser light barriers.

The firmware for the Timing System is implemented with Processor Expert components using the Eclipse based CodeWarrior for MCU v10.x, in two projects: one for the main CPU (as the figure *FreeRTOS running the Timing System* shows) and second for the LCD subsystem (as the figure *Tower LCD Subsystem Components with FreeRTOS* shows). Both subsystems and all remote systems runs with FreeRTOS.

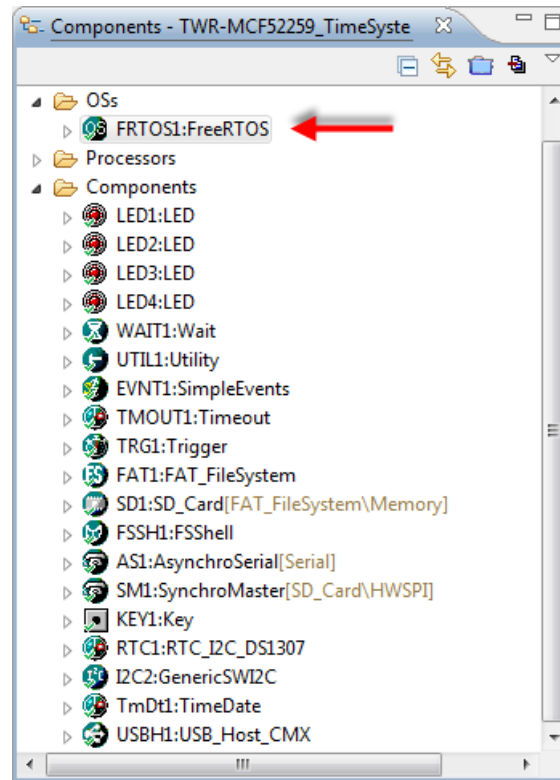
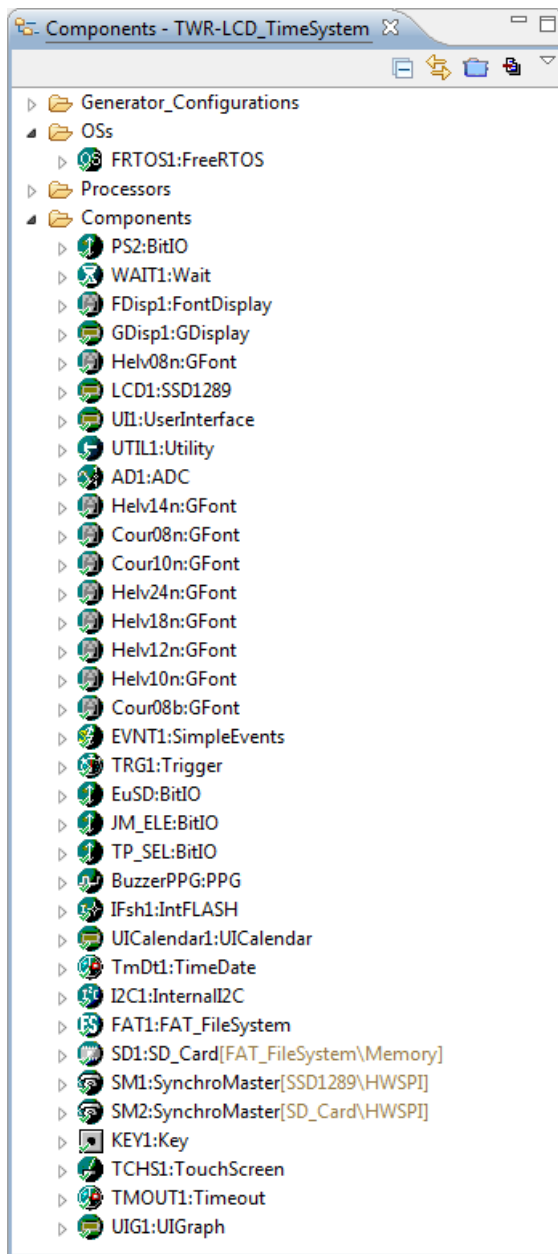


Figure 2-129. FreeRTOS running the Timing System



**Figure 2-130. Tower LCD Subsystem Components with FreeRTOS**

The FSShell is used to access to all commands plus reports that status of the operating system with all the tasks, as the following figure shows:

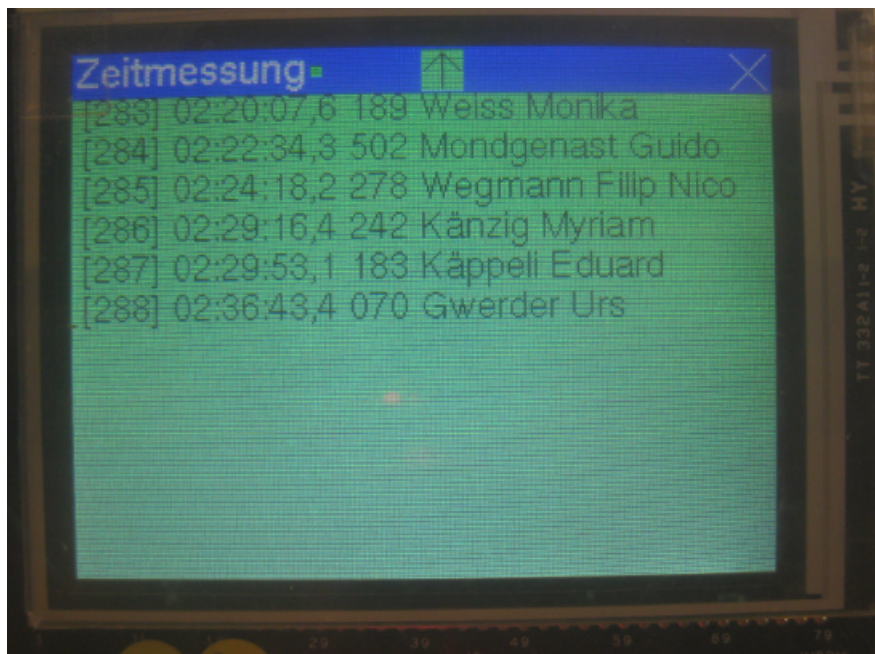
```

Terminate 2.9 (by CompuPhase)
COM1 38400 bps, 8N1, no handshake  Settings  Clear  About  Close

list ranking          ; make a finisher ranking
verbose on/off       ; turn verbose mode on or off (default)
-----
CMD> status
-----
SYSTEM STATUS
-----
TASK LIST:
Name      Status   Prio  Stack TCB#
-----
Shell     R        1     297   3
IDLE     R        0     253   1
Key       B        3     549   4
Scan     B        3     538   5
SD        B        3     297   2
Main     B        2     499   0
-----
RTOS RUN-TIME STATISTICS:
Name  Time (ms)  %Time
-----
Shell      3598        1%
IDLE    206485       97%
Main     1846        <1%
Scan      15          <1%
SD        554         <1%
Key       91          <1%
RTOS ticks : 100 Hz, 10 ms
Free heap  : 6248 bytes
Firmware   : Jul  3 2012 20:05:14
SW RTC    : 14.08.2012 23:06:10,70
HW RTC    : 14.08.2012 23:06:10
SD present : yes
SD protected : no
Nof records : 289
Max records : 1169
-----
Maximum for startlist: 384
Number in startlist  : 306
Radio                : no
Verbose mode         : off
Start time           : 09:45:00
-----
CMD>
    
```

**Figure 2-131. Shell Output with FreeRTOS Tasks and Performance**

With the transparency of the box you can read easily the LCD output. For example, the TWR-LCD display shows the list of finishers with index, running time, running number and the name of the athlete plus overall status, as the following figure shows:



**Figure 2-132. TWR-LCD for Timing System**

The same information (and more) can be displayed on a terminal window if a notebook is attached. The serial link or the wireless link synchronizes with an Microsoft Access data base. That database can be local or remote and can be used to print the output.

### **2.6.59** NEW **Can I program a different Freedom Board using Freedom board as SWD programmer?**

No, you cannot program a different Freedom Board using Freedom board as SWD programmer. This is because the OpenSDA checks the target processor, and only allows to program the same processor as on the Freedom board.

### **2.6.60** NEW **Can MCU10.4 recover a bricked OpenSDA Freedom Board?**

Yes, installing the CodeWarrior MCU10.4 with new drivers loaded properly can unlock your board and you can successfully debug your board with OpenSDA.



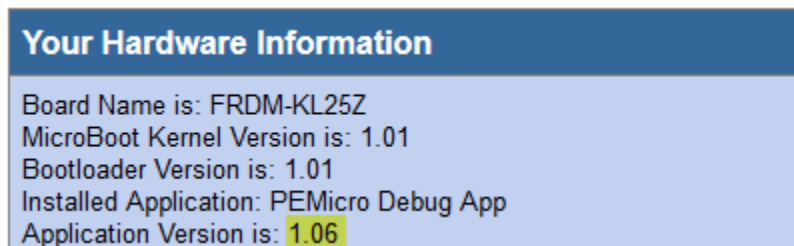


Figure 2-133. Unblocked Board with PEMicro Debug App V1.06

## 2.6.61 NEW Do I need to initialize my port pin?

Yes, you have to initialize your port pin by using `BitIO_Ldd3_Init()` function, which is part of the BitIO component for the push button:

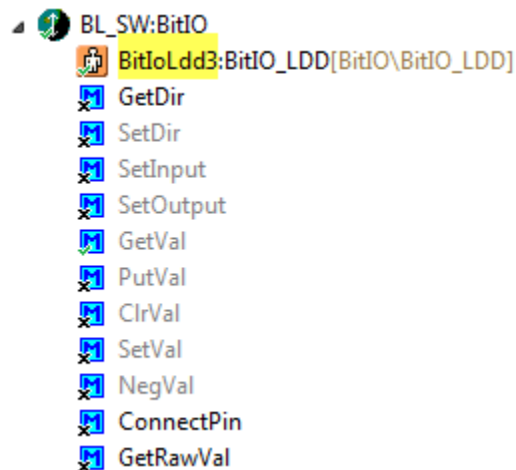


Figure 2-134. BitIO\_Ldd Component

You need to initialize it before `main()` function gets called, very early in the `startup()` code.

### NOTE

When creating a BitIO component for Kinetis, Processor Expert automatically creates a BitIO\_LDD component for it.

## 2.6.62 NEW For my bootloader I need a way to send a file with a terminal program. As my serial connection has only Tx and Rx, but no RTS/CTS lines for flow control, it is useful if the terminal program either implements software flow

## control (XONXOFF), or a delay value for sending a file. What do I do?

To send a file with a terminal program, you may use an open source terminal program, available in the following location:

<https://sites.google.com/site/terminalbpp/>

It supports sending a file with a delay (as the following figure shows, with 1 ms delay), and supports XON and XOFF.

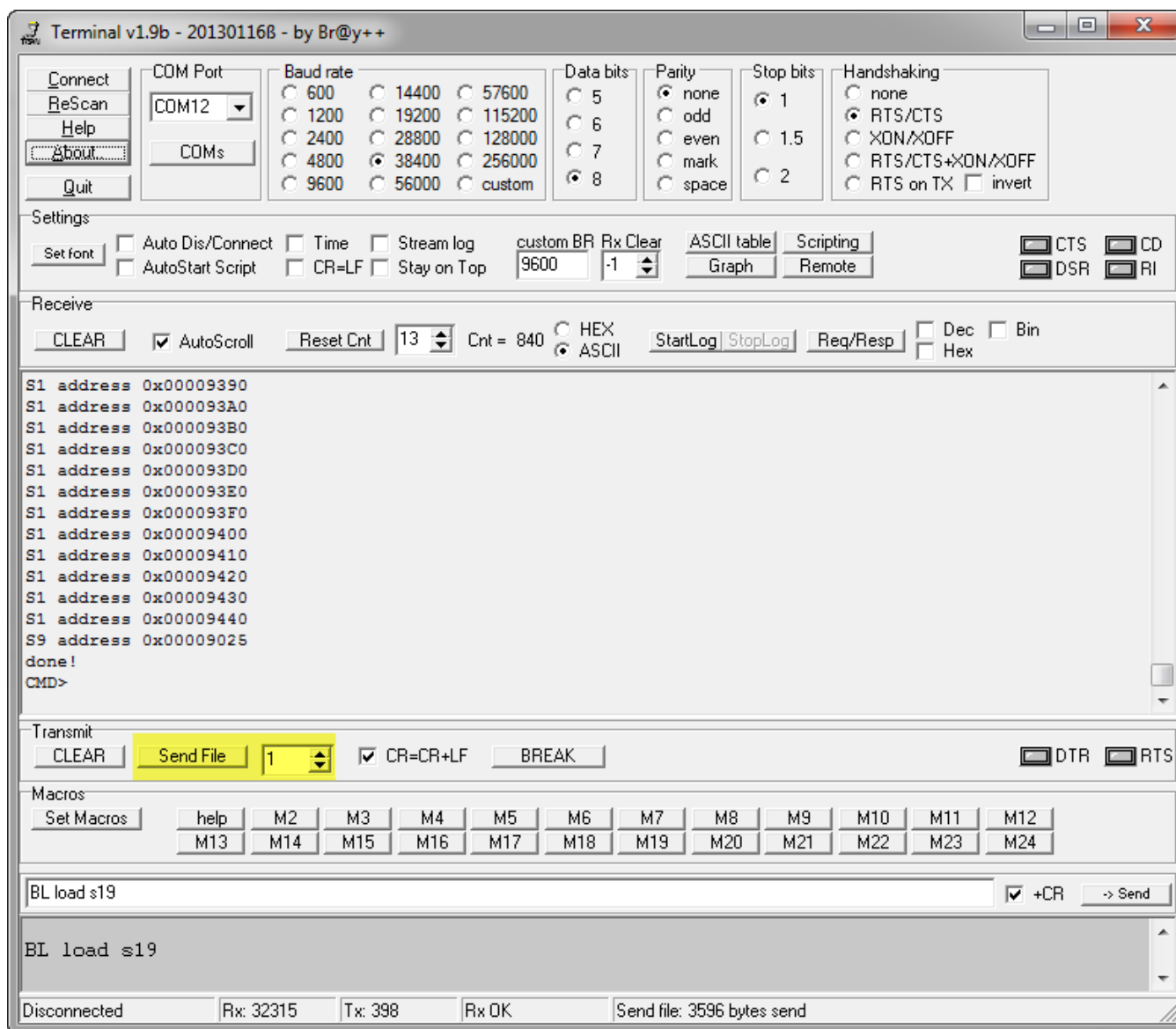


Figure 2-135. Terminal Program Window

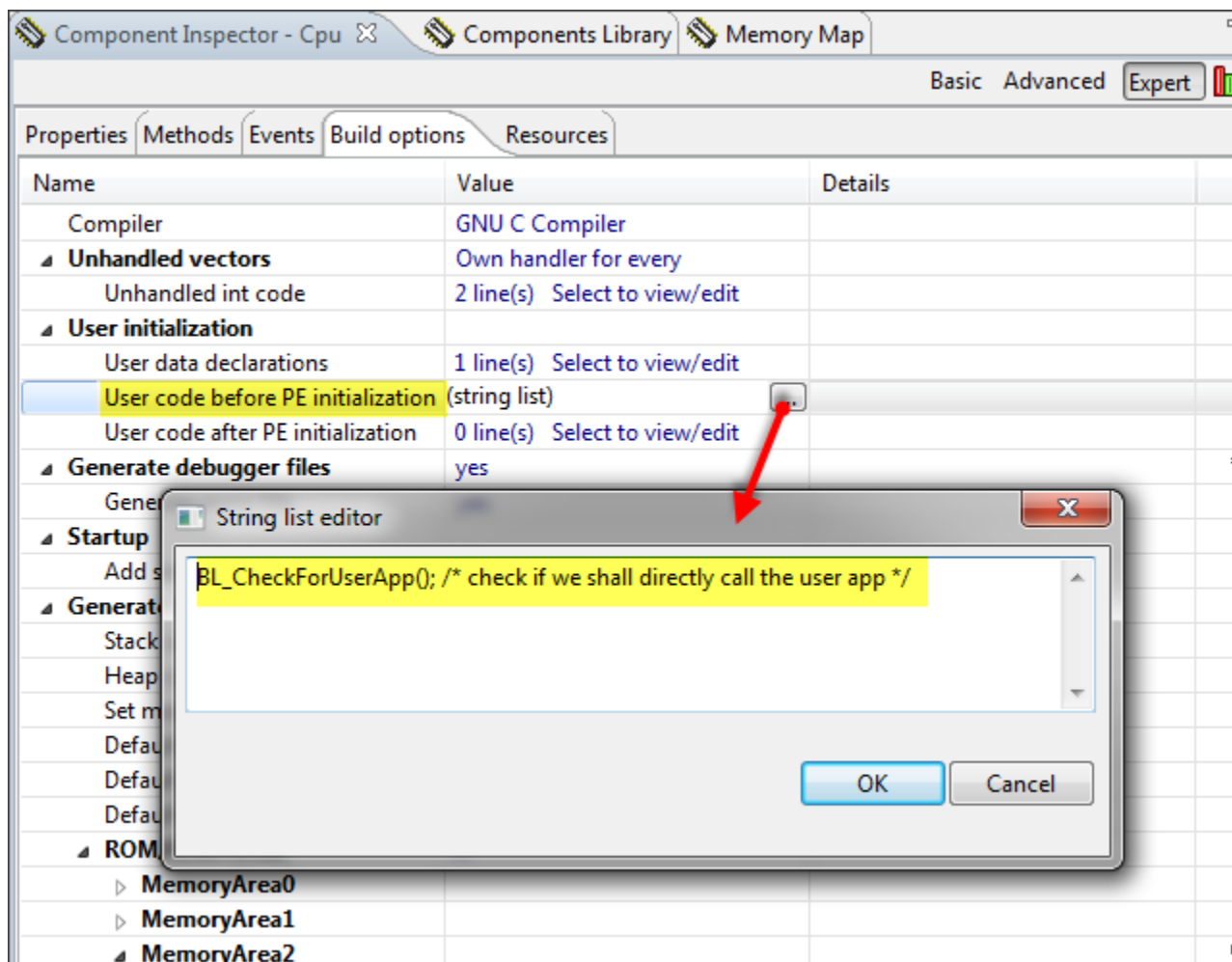
### Tip

Using a zero delay does not work in all the cases. Instead, send a file with a 1 ms delay setting.

## 2.6.63 NEW How can I add my own code to be inserted as part of the system startup?

To add your own code:

1. In the CPU component properties, click the **Build options** tab.
2. Select **User code before PE initialization** from the **User initialization** group.

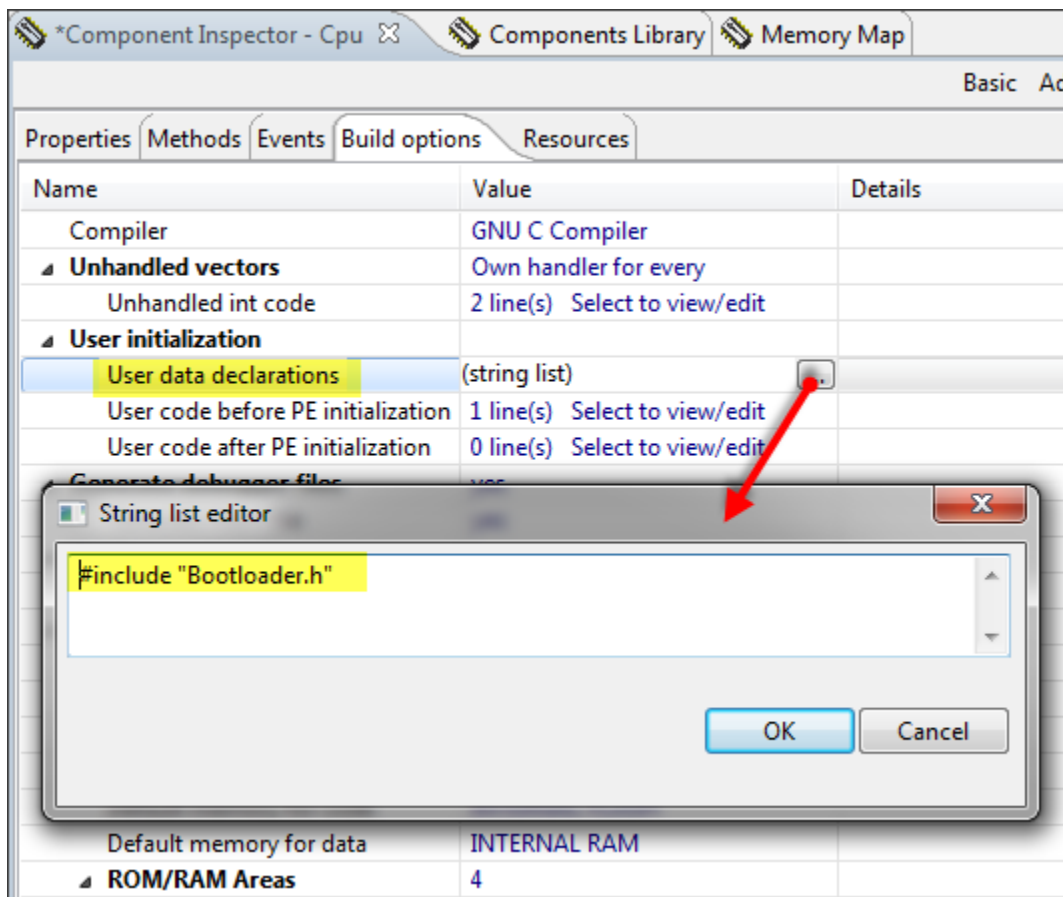


**Figure 2-136. User Code before PE initialization**

3. Click (...) button from the **Value** column.

The **String list editor** dialog box appears.

4. In the **String list editor** dialog box, specify the code you want to be inserted as part of the system startup.
5. Click **OK**.
6. To add the user defined header file, select **User data declarations** from the **User Initialization** group.



**Figure 2-137. User data declarations**

7. Click (...) button from the **Value** column.

The **String list editor** dialog box appears.

8. In the **String list editor** dialog box, specify the header file to be inserted as part of the system startup.
9. Click **OK**.

The code snippets get added to the `__init_hardware()` function which is called from the bootloader startup code:

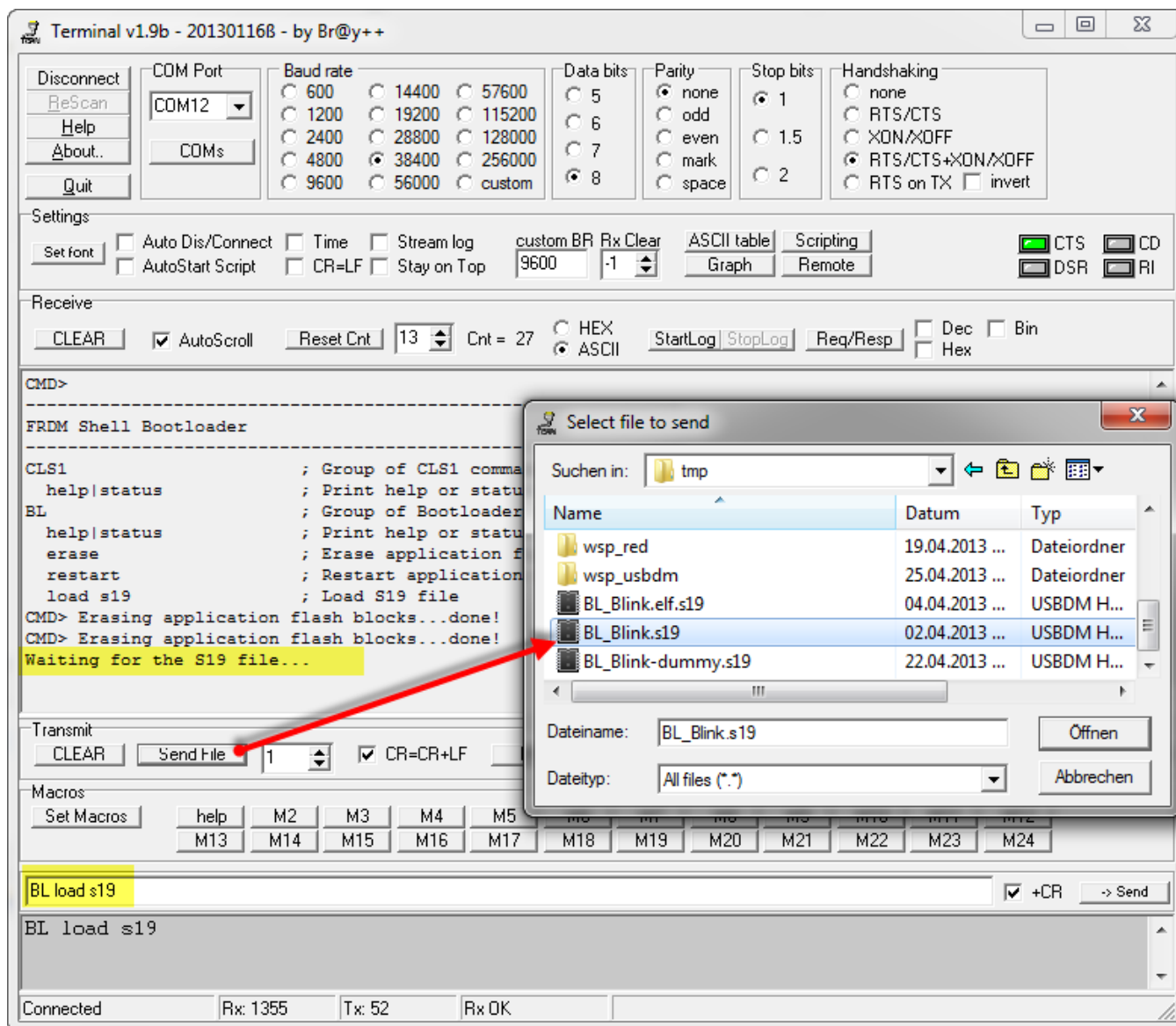
```

169
170
171 void __thumb_startup(void) _EWL_NAKED;
172
173 void __thumb_startup(void)
174 {
175 {
176
177 // Setup registers
178 __init_registers();
179
180
181
182 // setup hardware
183
184
185 init_hardware();
186
187
188
189
190
191 #if defined(__APCS_ROPI) || defined(__APCS_f
192
721 /* User declarations */
722 #include "Bootloader.h"
723
724 /*lint -esym(765,__init_hardware) Disable MISRA rule (8.10) check
725 /*lint -esym(765,Cpu_Interrupt) Disable MISRA rule (8.10) checkin
726 void init_hardware(void)
727 {
728 /*** User code before PE initialization ***/
729 BL_CheckForUserApp(); /* check if we shall directly call the u
730 /*** End of user code before PE initialization ***/
731
732 /*** ### MKL25Z128VLK4 "Cpu" init code ... ***/
733 /*** PE initialization code after reset ***/
734 SCB_VTOR = (uint32_t)&_vect_table); /* Set the interrupt vec
735 /* Disable the WDOG module */
736 /* SIM_COPC: ??=0, ??=0, ??=0, ??=0, ??=0, ??=0, ??=0, ??=0, ??=0, ??=0
737 SIM_COPC = (uint32_t)0x000UL;
738 /* System clock initialization */
739 /* SIM_SCGC5: PORTB=1, PORTA=1 */
740 SIM_SCGC5 |= (uint32_t)0x0600UL; /* Enable clock gate for
741 /* SIM_CLKDIV1: OUTDIV1=3, ??=0, ??=0, ??=0, ??=0, ??=0, ??=0, ??=0, ?
742 SIM_CLKDIV1 = (uint32_t)0x30000000UL; /* Update system prescal
743 /* SIM_SOPT2: PLLFLLSEL=1 */
744 SIM_SOPT2 |= (uint32_t)0x00010000UL; /* Select PLL as a clock
    
```

Figure 2-138. Custom Startup Code with Bootloader Code Inserted

## 2.6.64 NEW How do I bootload an application?

You can bootload a new application file with BL load s19 command. It first erases the application flash blocks, and then waits for the S19. To send the file, you have to use the **Send File** button, as the following figure shows:



**Figure 2-139. Terminal Window - Bootloading an Application**

When you click the **Send File** button, then it writes the address of each S19 line programmed to the shell console, as the following listing shows:

```

CMD> Erasing application flash blocks...done!
Waiting for the S19 file...
S0 address 0x00000000
S1 address 0x00008000
S1 address 0x00008010
...
S1 address 0x00009420
S1 address 0x00009430
S1 address 0x00009440
S9 address 0x00009025
done!
CMD>

```

## 2.6.65 NEW How do I decode S19 Files?

To decode an S19 file, follow these steps:

- Double-click the `decoder.exe` file, available in `CW_InstallDir\MCU\prog` folder.

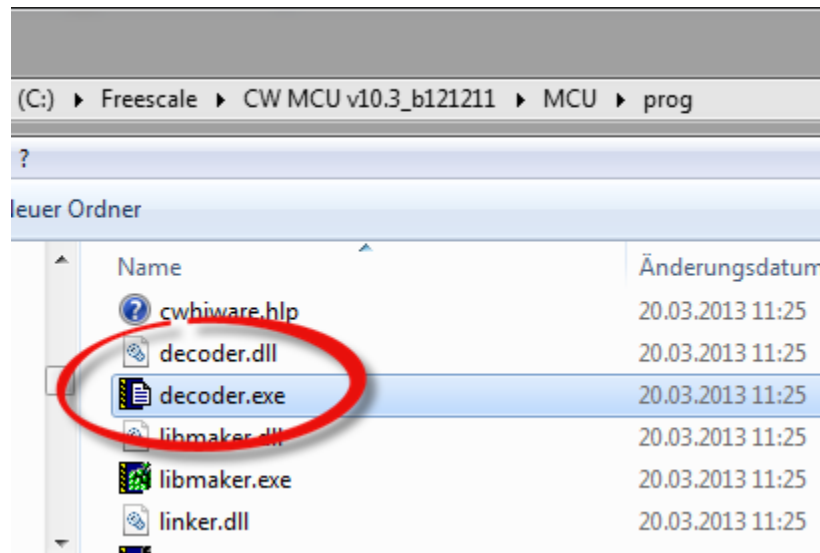


Figure 2-140. Decoder.exe

The **Decoder Default Configuration** window appears.

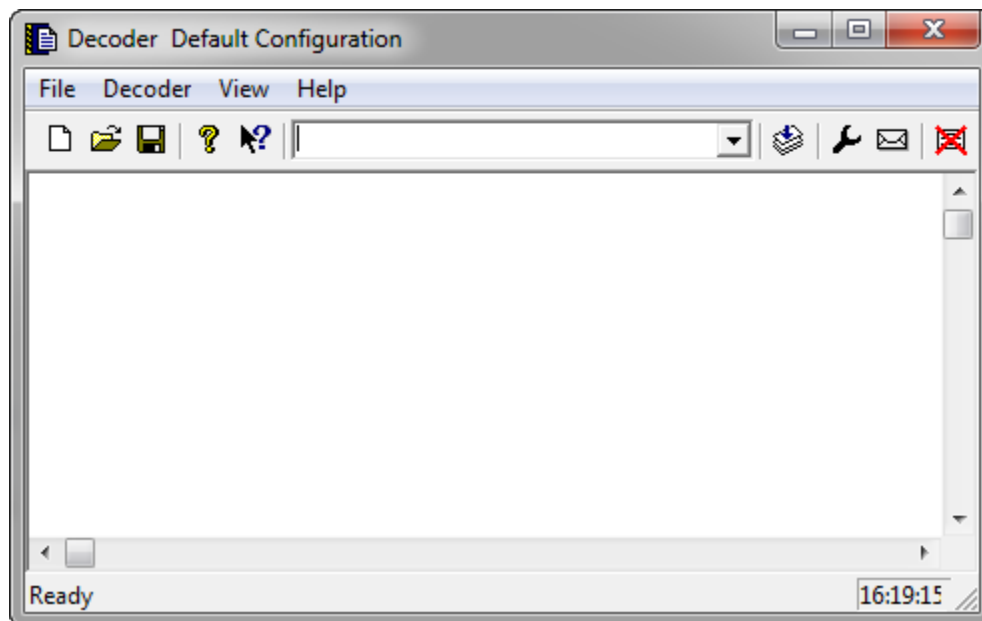
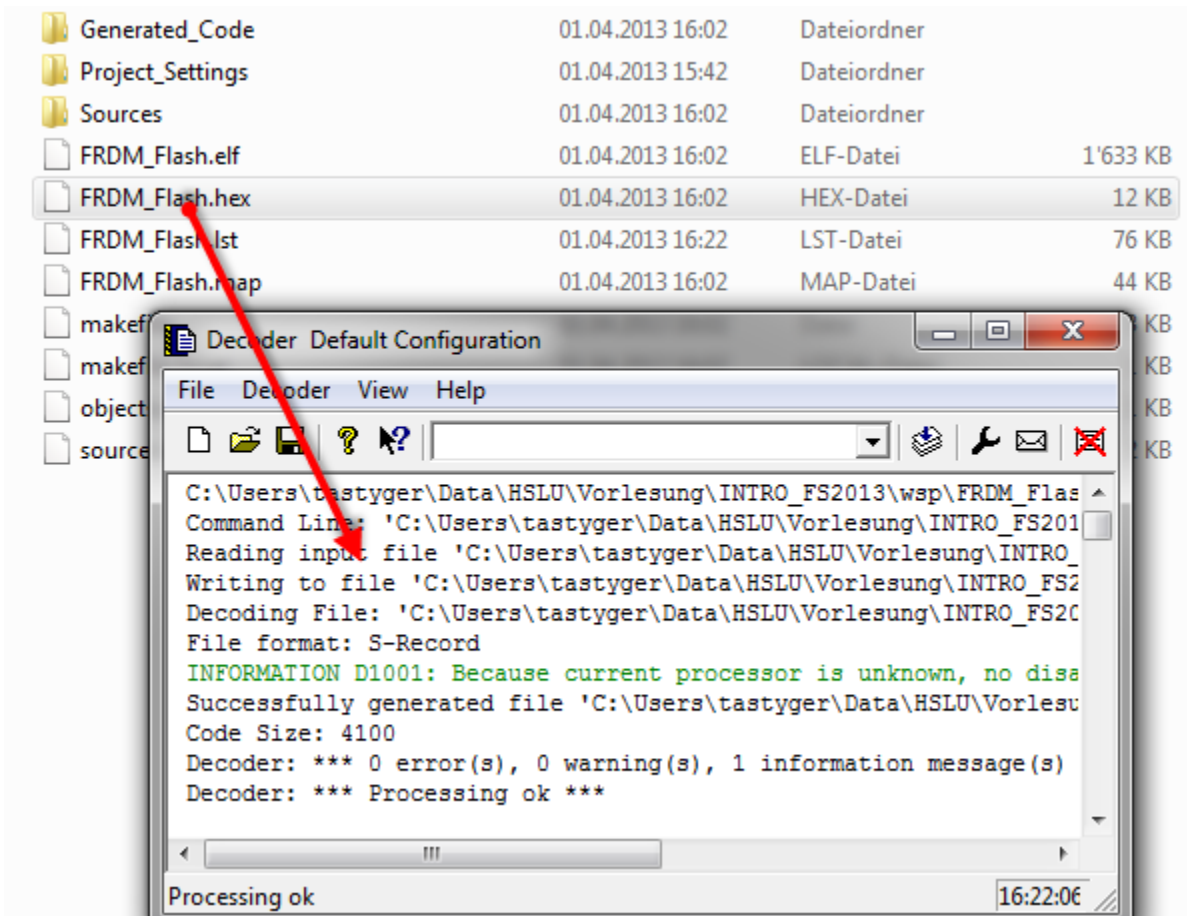


Figure 2-141. Decoder Default Configuration

- To decode an S19 file simply drag and drop it in the Decoder window.



**Figure 2-142. Drag and Drop of S19 file into decoder**

This creates the .lst listing which you can open in a text editor.



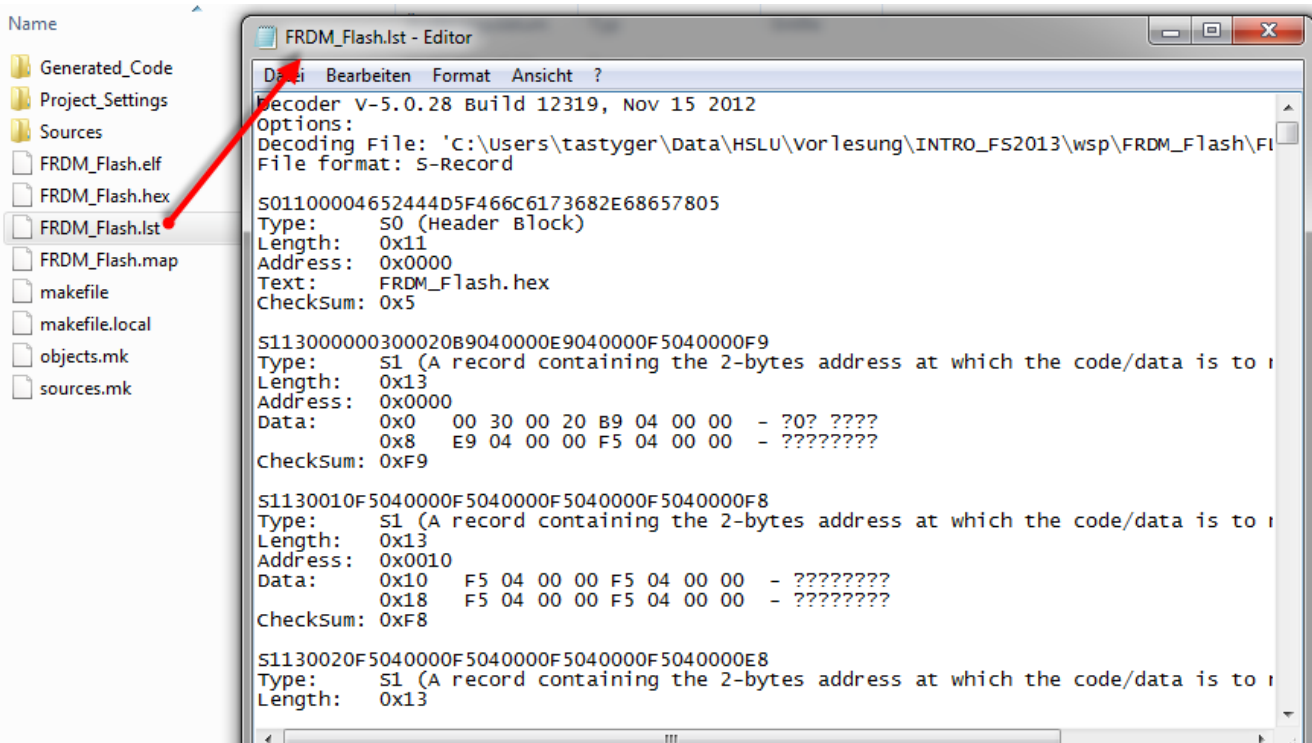


Figure 2-143. Decoded S19 File

Now you can easily inspect each S-Record line with type, length, address, data and checksum.

### 2.6.66 NEW How do I fix the 3.3V voltage drop on FRDM-KL25Z Board?

The problem is because of Schottky Diode 1. It causes a the voltage of about 0.5V and even larger for higher currents. So the simple *fix* is to shortcut pin 1 and pin 3 of that diode D1:

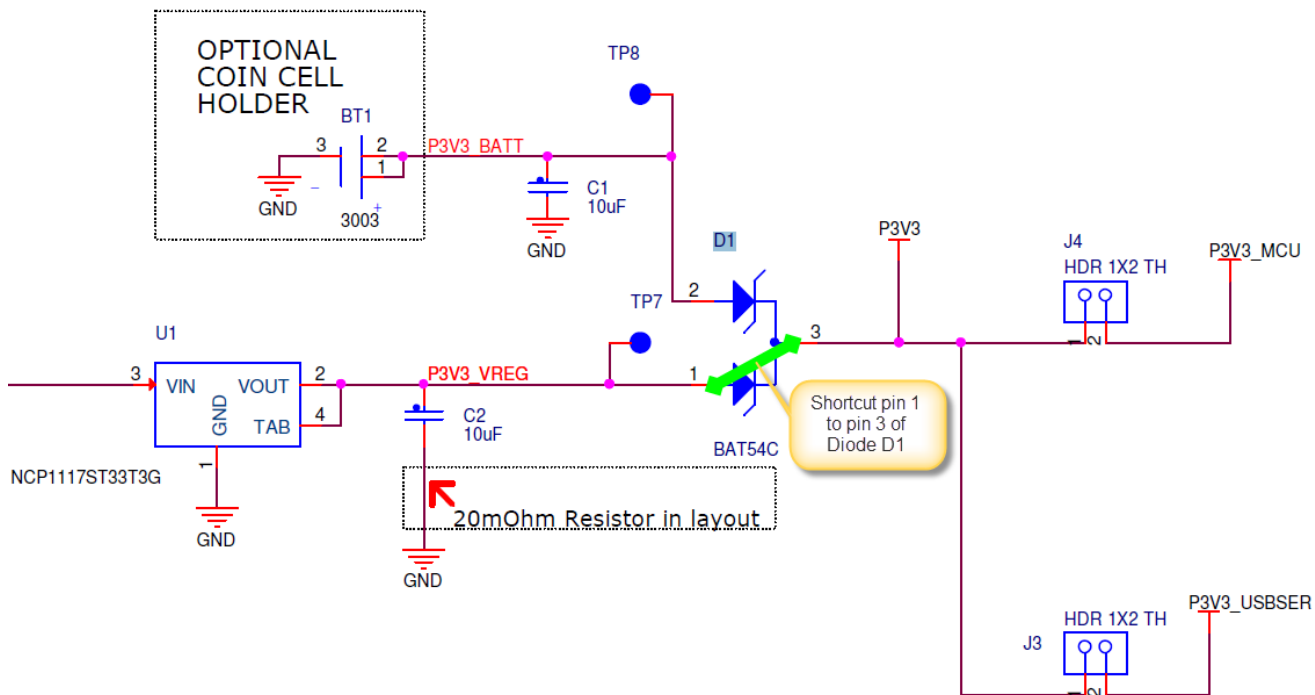
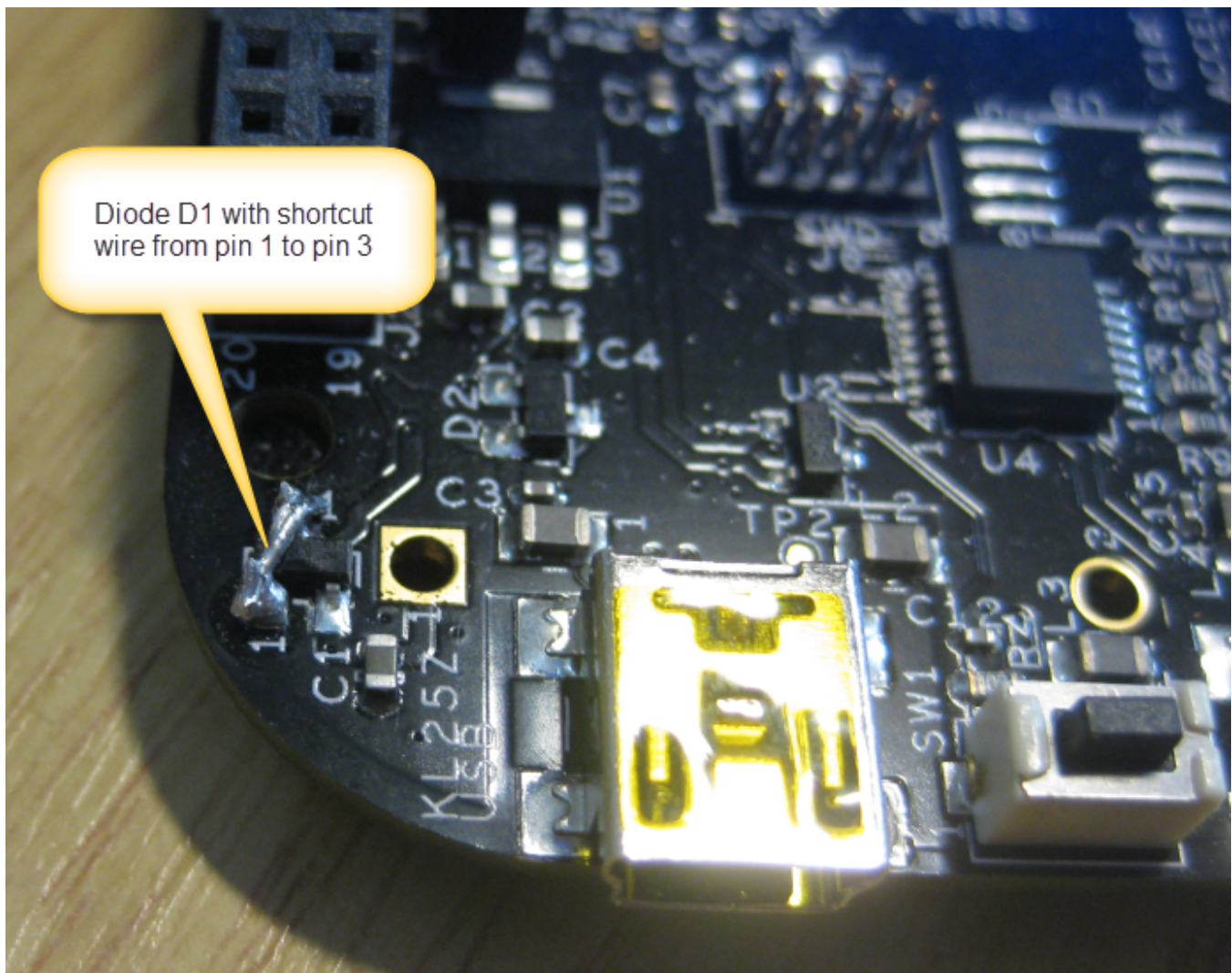


Figure 2-144. Shortcut on D1 of FRDM-KL25Z

That diode D1 is located nearby the KL25Z USB port:



**Figure 2-145. Diode D1 Shortcut Wire Applied**

**NOTE**

Change the hardware at your own risk. Make sure your power supply is not reverted, and that you only draw a few 100 mA.

### 2.6.67 How do I install open source USBDM?

USBDM is a free open source (GPLv2) debugging/programming interface for a range of Freescale microcontrollers.

**NOTE**

USBDM is not OSBDM. OSBDM (or OSJTAG) is an older *free* debugging interface present on many of the Freescale Tower Boards.

It currently supports CodeWarrior and CodeSourcery Eclipse. With it, you can overcome the OpenSDA limitation and using it as general purpose debug/programming device with CodeWarrior for MCU10.3.


**Tip**



















USBDM comes with a standalone flash programmer and scripting engine.

To install the USBDM:

1. Download the installer (USBDM\_4\_10\_4a\_Win.msi) from the location listed below:

<http://sourceforge.net/projects/usbdm/files/Version%204.10.4/Installation/>

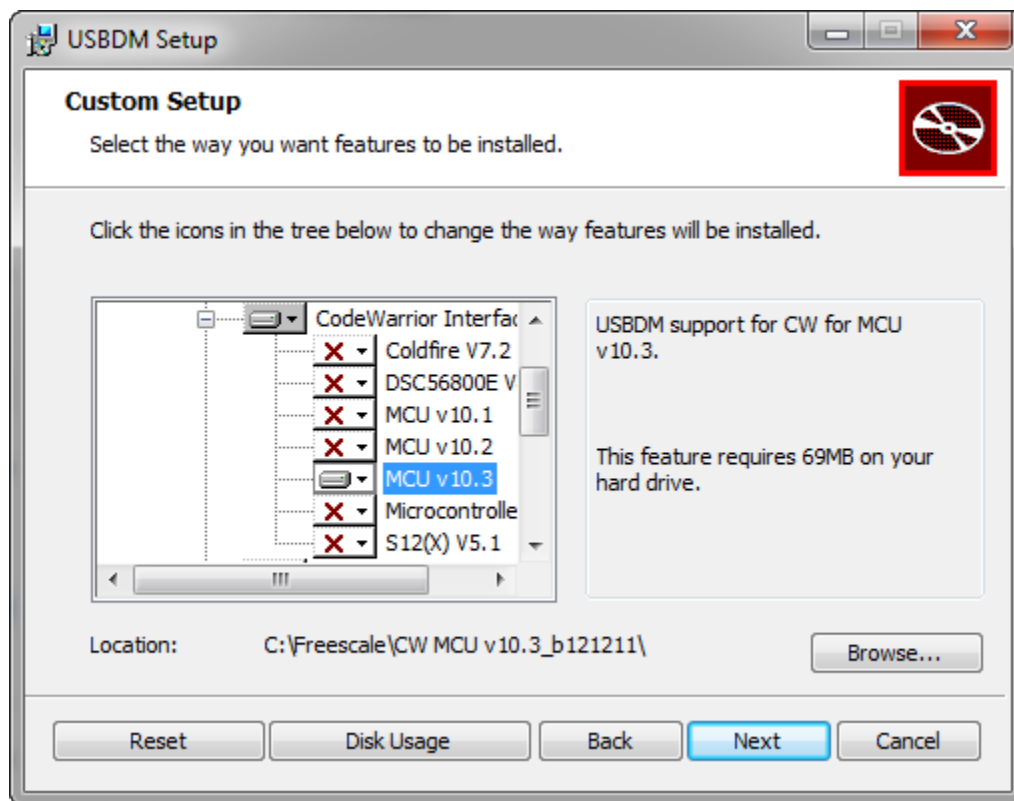
Home / Version 4.10.4 / Installation 

Name ▾	Modified ▾	Size ▾	Downloads ▾		
↑ Parent folder					
USBDM_JNI.zip	2013-03-18	1.5 MB	7		
ReadMe.txt	2013-03-02	3.1 kB	13		
USBDM_OpenSDA.zip	2013-03-02	60.3 kB	14		
TBDML.zip	2013-02-13	10.2 MB	4		
<b>USBDM_4_10_4a_Win.msi</b>	2013-01-23	38.8 MB	12		
USBDM_Linux_V4.10.4.tar.gz	2013-01-18	26.7 MB	5		
USBDM_Drivers_1_0_1_Win_x64.msi	2013-01-11	6.2 MB	16		
USBDM_Drivers_1_0_1_Win_x32.msi	2013-01-11	5.9 MB	14		
USBDM_4_10_4_Win.msi	2013-01-11	40.4 MB	121		
<b>Totals: 9 Items</b>		<b>129.7 MB</b>	<b>206</b>		

**Figure 2-146. USBDM BDM Interface for Freescale Microcontroller - USBDM\_4\_10\_4a\_Win.msi**

2. Save the installer at desired location.
3. Double click the installer (USBDM\_4\_10\_4a\_Win.msi) to run the installation and follow the onscreen instructions.

The setup provides the flexibility only to install what is needed:



**Figure 2-147. USBDM Setup**

### NOTE

In above dialog box, you need to specify the path to your CodeWarrior installation.

4. Download the USBDM\_Drivers file (USBDM\_Drivers\_1\_0\_1\_Win\_x64.msi for Windows7 64bit machine) from the location listed below:

<http://sourceforge.net/projects/usbdm/files/Version%204.10.4/Installation/>

Home / Version 4.10.4 / Installation



Name ↕	Modified ↕	Size ↕	Downloads ↕		
↑ Parent folder					
USBDM_JNI.zip	2013-03-18	1.5 MB	7		
ReadMe.txt	2013-03-02	3.1 kB	13		
USBDM_OpenSDA.zip	2013-03-02	60.3 kB	14		
TBDML.zip	2013-02-13	10.2 MB	4		
USBDM_4_10_4a_Win.msi	2013-01-23	38.8 MB	12		
USBDM_Linux_V4.10.4.tar.gz	2013-01-18	26.7 MB	5		
USBDM_Drivers_1_0_1_Win_x64.msi	2013-01-11	6.2 MB	16		
USBDM_Drivers_1_0_1_Win_x32.msi	2013-01-11	5.9 MB	14		
USBDM_4_10_4_Win.msi	2013-01-11	40.4 MB	121		
Totals: 9 Items		129.7 MB	206		

**Figure 2-148. USBDM\_Drivers for Windows**

5. Save the installer at desired location.
6. Double click the installer (USBDM\_Drivers\_1\_0\_1\_Win\_x64.msi) to run the installation and follow the onscreen instructions.

The installer installs the driver at the location you specified.

## 2.6.68 NEW How do I load the Debug Application on my FRDM-K20D50M board?

The FRDM-K20D50M board has the OpenSDA on-board debug connection, so to load the debug application, follow these steps:

1. Press the reset button on power-up.
2. Load the OpenSDA Debug Application.
3. Re-power the board.

## 2.6.69 **NEW** How do I make sure that application does not overlap with the FLASH area of the bootloader?

To make sure that application does not overlap with the FLASH area of the bootloader, you need to change the addresses for `m_interrupts` and `m_text` in the CPU build options:

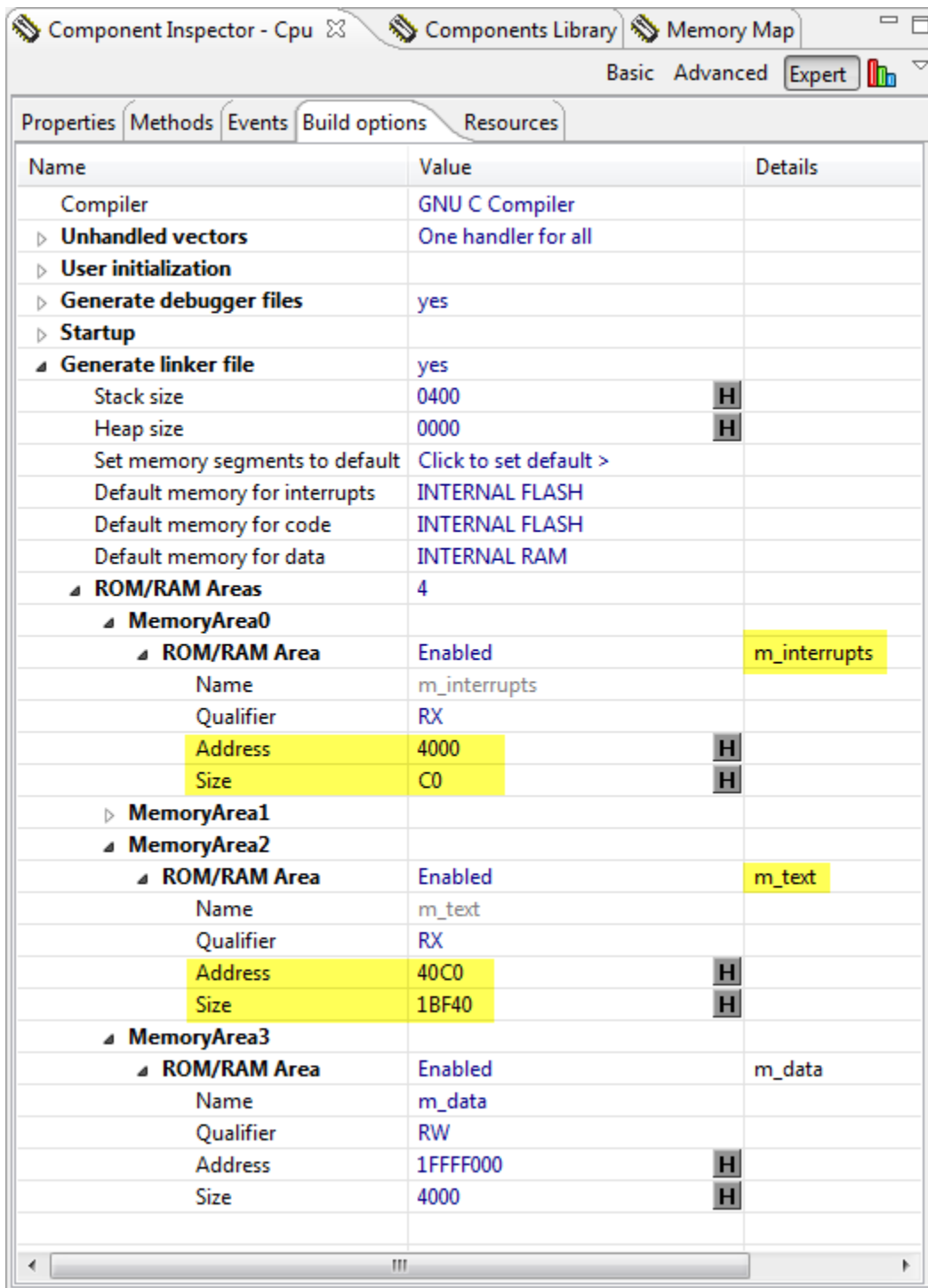


Figure 2-149. Application Memory Area Settings

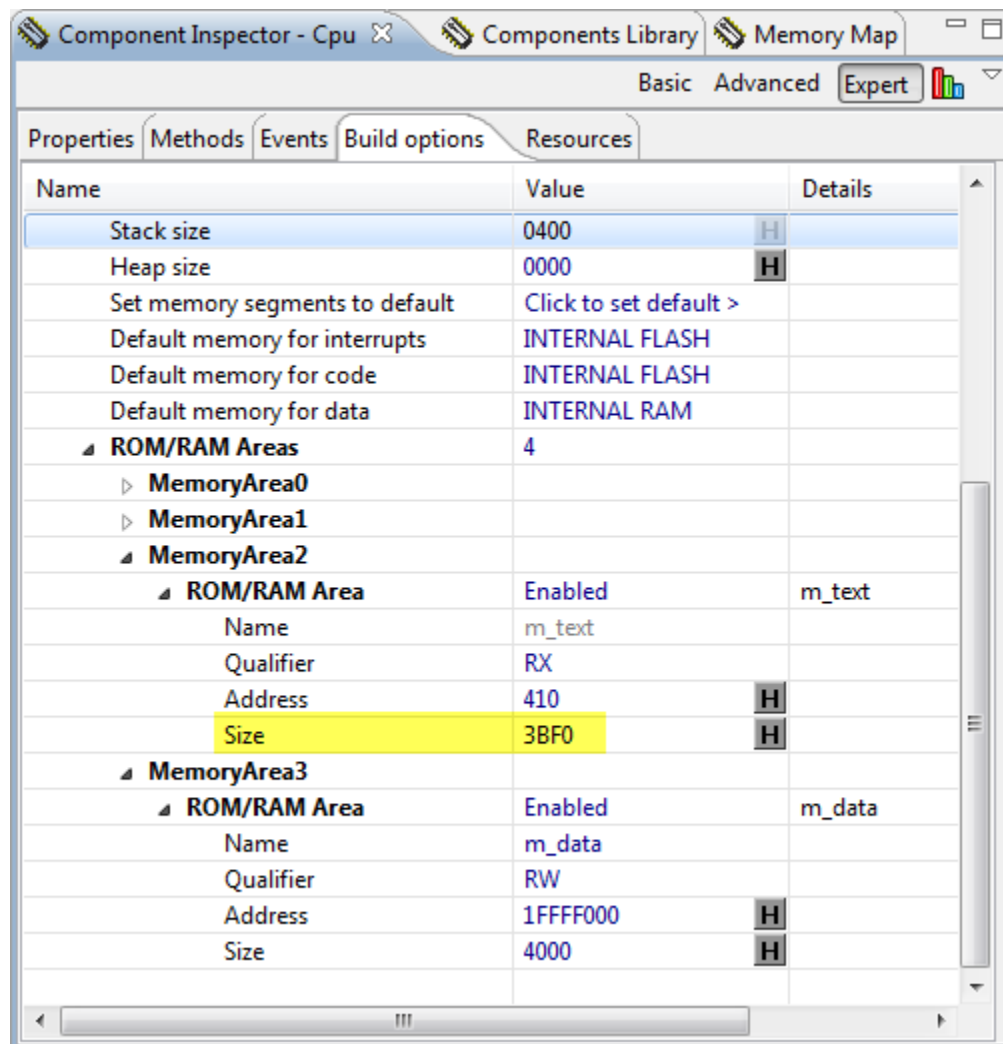
**NOTE**

For the ARM Cortex-M4/0+ you do not need to copy the vector table in the bootloader to a different location, you can debug the application easily without the bootloader.



## 2.6.70 NEW How do I make sure that the bootloader gets linked only into its specified space?

To make sure the bootloader gets linked only into its space, you have to reduce the FLASH memory for it. For example, the settings in the following figure limits the FLASH memory from 0x0000 (vector table) up to 0x3FFF. That means your application memory area starts at 0x4000.



**Figure 2-150. Bootloader FLASH Area**

Change the available flash for the bootloader in the CPU properties, and cut the available FLASH size on the KL25Z128 from 0x1FBF0 (~128 KByte) in the **Build options** tab to 0x3FB0. With this, the bootloader occupies the space from address 0x0000 (vector table) up to 0x3FFF.

## 2.6.71 **NEW** How do I set up criteria for bootloader to decide if it shall run the bootloader or the application at the startup?

For this you need to have a decision criterion, which is typically a jumper or a push button to be pressed at power up to enter bootloader mode. This is performed by the function `BL_CheckForUserApp()`:

```

/*!
 * \brief This method is called during startup! It decides if we enter bootloader mode or if
 * we run the application.
 */
void BL_CheckForUserApp(void) {
    uint32_t startup; /* assuming 32bit function pointers */

    startup = ((uint32_t*)APP_FLASH_VECTOR_START)[1]; /* this is the reset vector (__startup
function) */
    if (startup!=-1 && !BL_CheckBootloaderMode()) { /* we do have a valid application vector?
-1/0xffffffff would mean flash erased */
        ((void(*) (void))startup)(); /* Jump to application startup code */
    }
}

```

It first fills a callback structure of type `S19_ParserStruct`, as the following listing shows:

```

typedef struct S19_ParserStruct {
    uint8_t (*GetCharIterator)(uint8_t*, void*); /* character stream iterator */
    void *voidP; /* void pointer passed to iterator function */
    uint8_t (*S19Flash)(struct S19_ParserStruct*); /* called for each S19 line to be flashed */
    /* the following fields will be used by the iterator */
    S19_FileStatus status; /* current status of the parser */
    uint8_t currType; /* current S19 record, e.g. 1 for S1 */
    uint32_t currAddress; /* current code address of S19 record */
    uint16_t codeSize; /* size of code in bytes in code buffer */
    uint8_t *codeBuf; /* code buffer */
    uint16_t codeBufSize; /* total size of code buffer, in bytes */
} S19_ParserStruct;

```

This function checks if the startup function in the vector table (index 1) is valid or not. If the application flash has been erased, it reads -1 (or 0xffffffff). So if you have an application present and the user does not want to run the bootloader, we jump to the application startup.

Listed below is the code to decide that on pressing the button, bootloader enters the startup code:

```

static bool BL_CheckBootloaderMode(void) {
    /* let's check if the user presses the BTLDD switch. Need to configure the pin first */
    /* PTB8 as input */
    /* clock all port pins */
    SIM_SCGC5 |= SIM_SCGC5_PORTA_MASK |
                SIM_SCGC5_PORTB_MASK |
                SIM_SCGC5_PORTC_MASK |
                SIM_SCGC5_PORTD_MASK |
                SIM_SCGC5_PORTE_MASK;
    /* Configure pin as input */
    (void)BitIoLdd3_Init(NULL); /* initialize the port pin */
}

```

```

if (!BL_SW_GetVal()) { /* button pressed (has pull-up!) */
    WAIT1_Waitms(50); /* wait to debounce */
    if (!BL_SW_GetVal()) { /* still pressed */
        return TRUE; /* go into bootloader mode */
    }
}
/* BTLD switch not pressed, and we have a valid user application vector */
return FALSE; /* do not enter bootloader mode */
}
    
```

## 2.6.72 NEW How do I use CMSIS-DAP with IAR on KL25Z Freedom Board?

CMSIS-DAP stands for *Cortex Microcontroller Software Interface Standard – Debug Access Port* has been published by ARM Inc. With this, there is an open source alternative to proprietary implementation (e.g. P&E OpenSDA or Segger OpenSDA). Beside of the ARM MDK IDE, CMSIS-DAP is supported by Coocox and IAR.

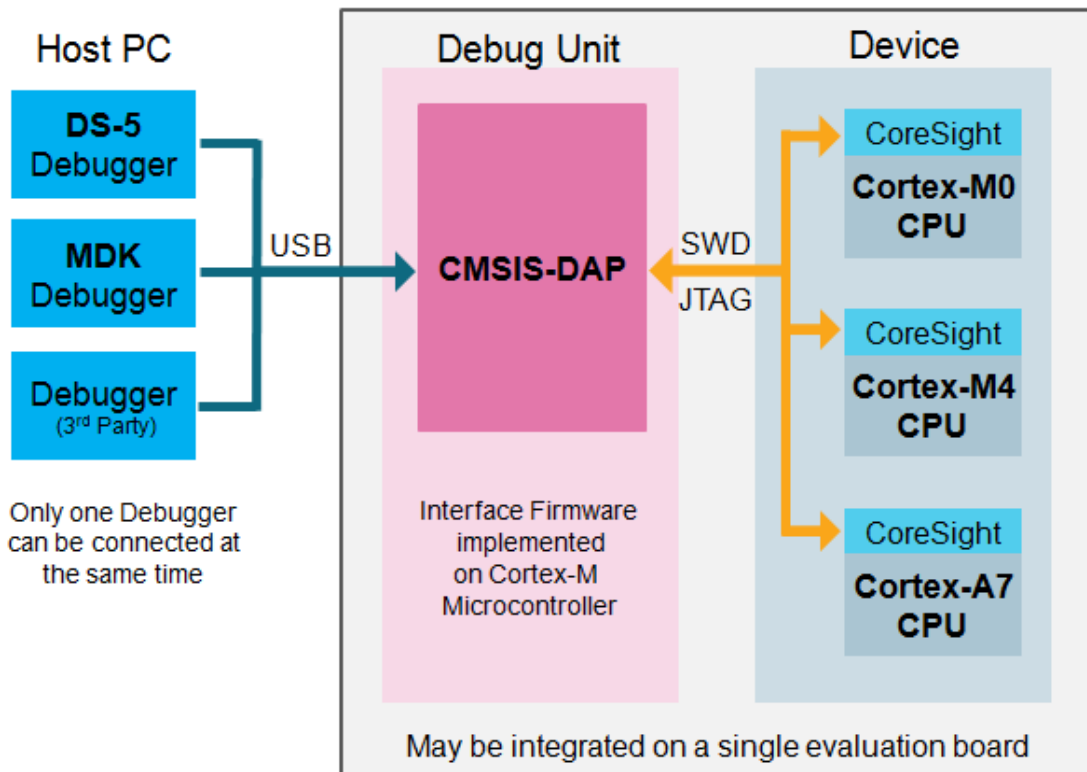


Figure 2-151. CMSIS-DAP block diagram

The CMIS-DAP firmware sources are available at (requires registration):

<https://silver.arm.com/browse/CMSISDAP>

The *Freescale Quick Start Package* also includes a pre-built version of the firmware, available at the following location:

[http://cache.freescale.com/files/32bit/software/board\\_support\\_packages/FRDM-KL25Z\\_QSP.zip](http://cache.freescale.com/files/32bit/software/board_support_packages/FRDM-KL25Z_QSP.zip)

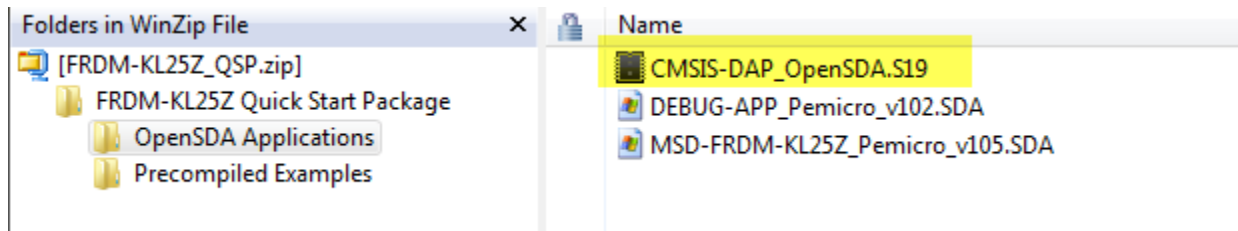


Figure 2-152. CMSIS-DAP\_OpenSDA.s19 in FRDM-KL25Z\_QSP.zip

to use the CMSIS-DAP with IAR on your KL25Z Freedom Board, follow these steps:

1. Download that zip file from the location listed above.
2. Unzip the files.
3. Update the OpenSDA firmware, as follows:
  - a. Power the FRDM-KL25Z with reset button pressed: green LED flashes.
  - b. Copy the CMSIS-DAP\_OpenSDA.s19 to the BOOTLOADER device.
  - c. Repower the board without the reset button pressed.

The device appears as **penSDA CMSIS-DAP**:

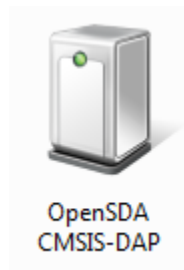


Figure 2-153. OpenSDA CMSIS-DAP

### 2.6.73 **NEW** How do I use Freedom board as SWD programmer?

The Freescale Freedom boards have the on-board OpenSDA. The OpenSDA features an MSD (Mass Storage Device) Bootloader application. With this you can copy an S19 file to the *programmer* and program another board. The Freedom boards come with an unpopulated SWD header which can be used to debug the processor with a Multilink or J-Link debug probe. The idea is to disconnect the processor on the board from OpenSDA, and connect it instead with an SWD cable to a microcontroller on another board.

To do this, you need:

- The OpenSDA Firmware with Bootloader.

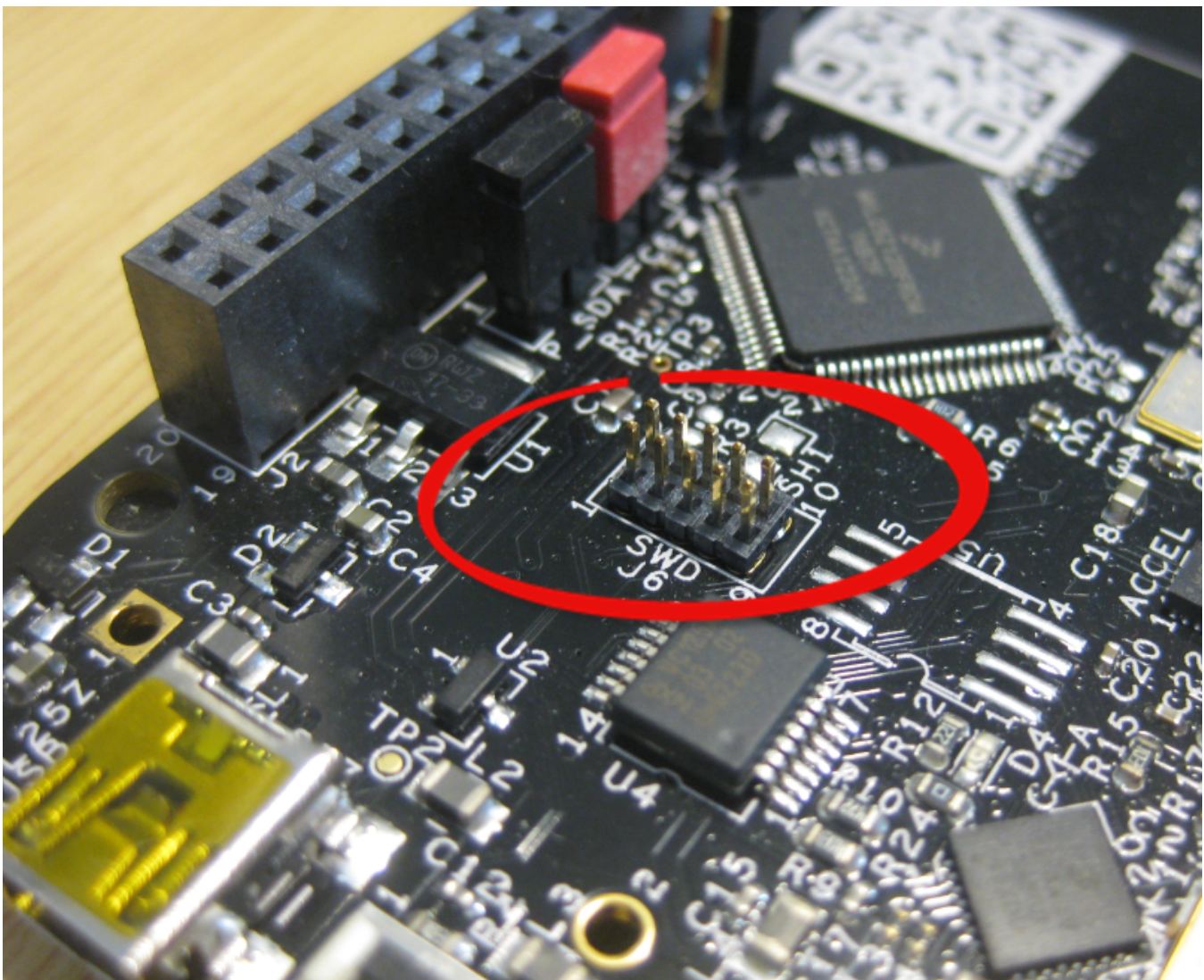
You can obtain the latest firmware from <http://www.pemicro.com/opensda>.

- A 10pin ARM Cortex SWD debug header to populate the one on the Freedom board.
- To cut J11 trace on the Freedom board to isolate the microcontroller from OpenSDA.
- A 10pin ARM Cortex SWD debug cable to connect the two boards.

To use the Freedom Board as SWD programmer, follow these steps:

**1. Populate the J11 header**

- a. By default, the Freedom boards come without the 10pin ARM Cortex debug header populated. So you need to add and solder such a header on J6:

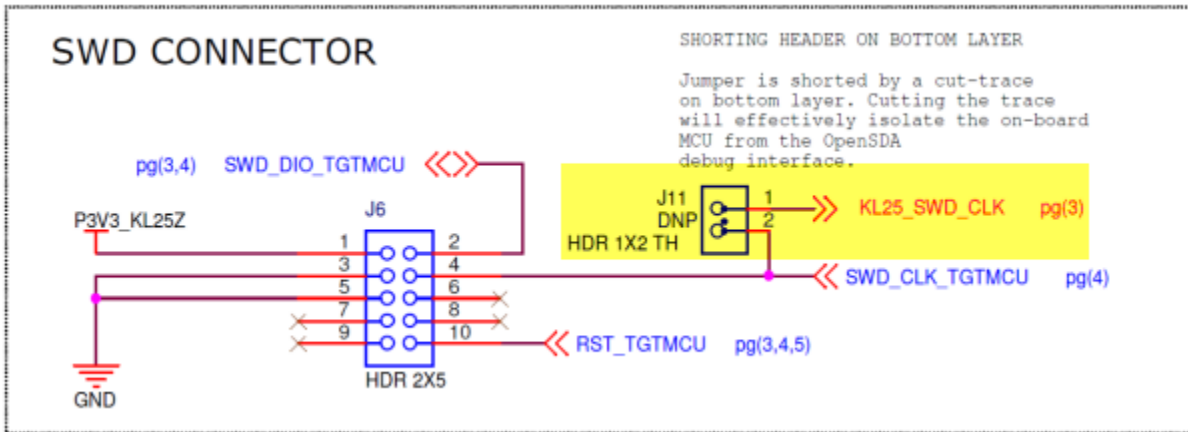


**Figure 2-154. SWD J6 Populated on the FRDM-KL25Z**

### Tip

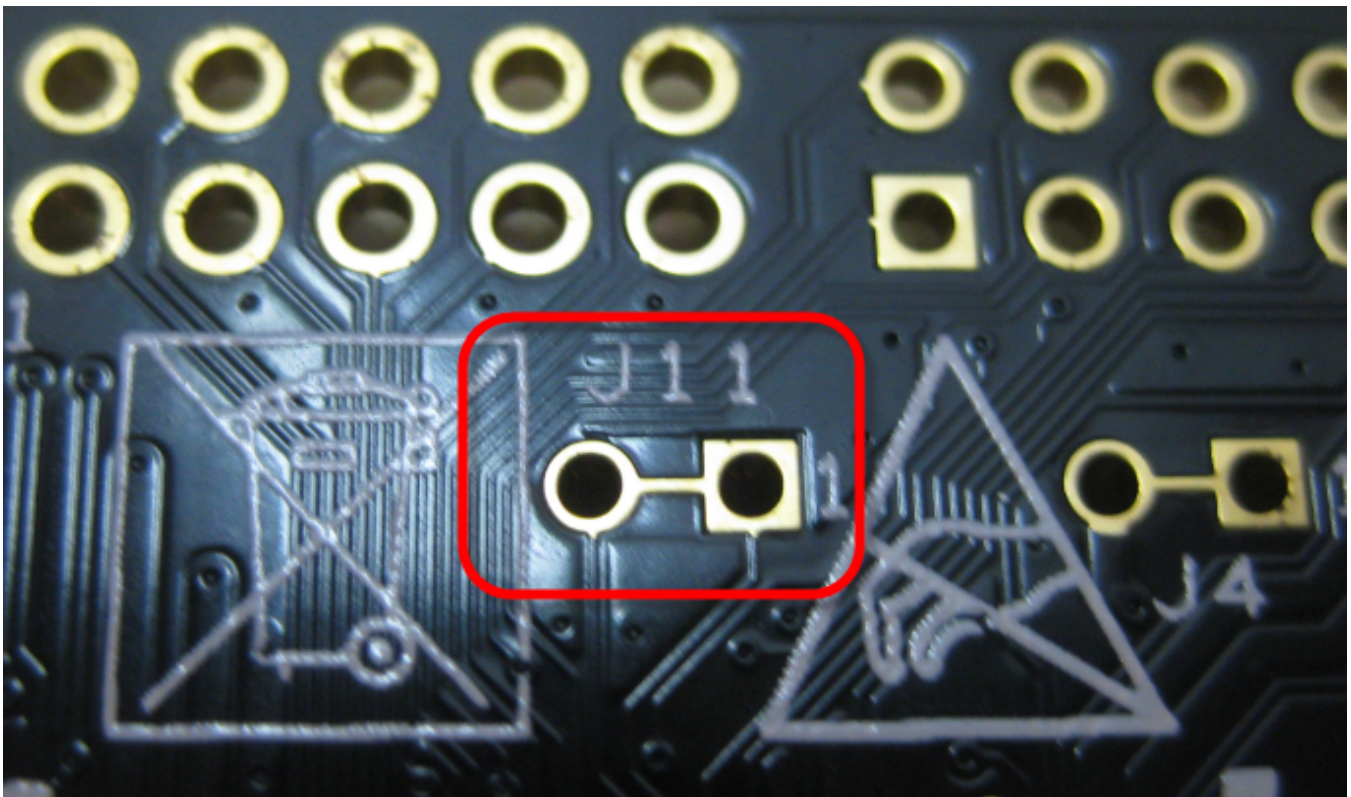
Use a good and fine soldering tip. The pads are very close to each other, so if using too much solder or not carefully solder the connector, there can be shortcuts between the pins. It is recommend to carefully inspect the soldered pins.

- b. Next, cut the J11 trace on the board. J11 needs to be used to remove the SWD clock to the target microcontroller on *programmer* Freedom board. With this you cut off the resident microcontroller from OpenSDA on the same board:



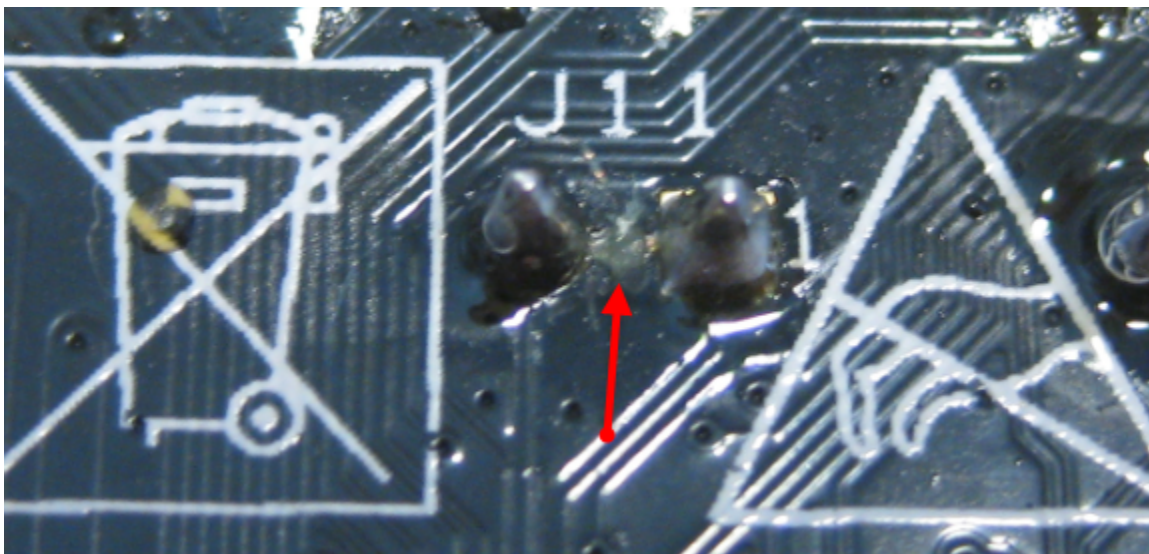
**Figure 2-155. J11 to isolate SWD Clock**

- c. Then you need to cut the connection between J11 on the backside of the board:



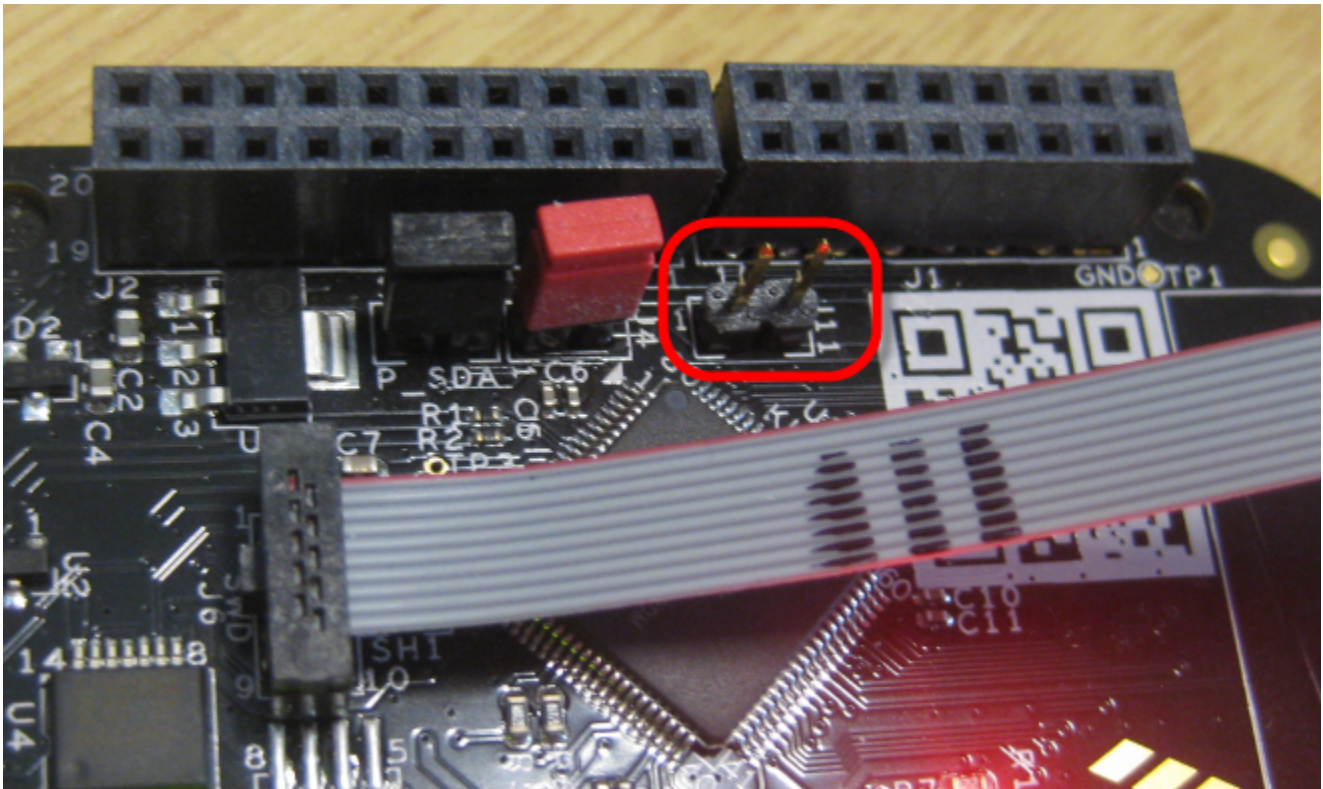
**Figure 2-156. J11 on the board backside**

It is recommend to cut the trace and install a dual-pin header/jumper:



**Figure 2-157. Cut J11 Trace**

d. Then populate the J11 header:



**Figure 2-158. J11 Header - Populated**

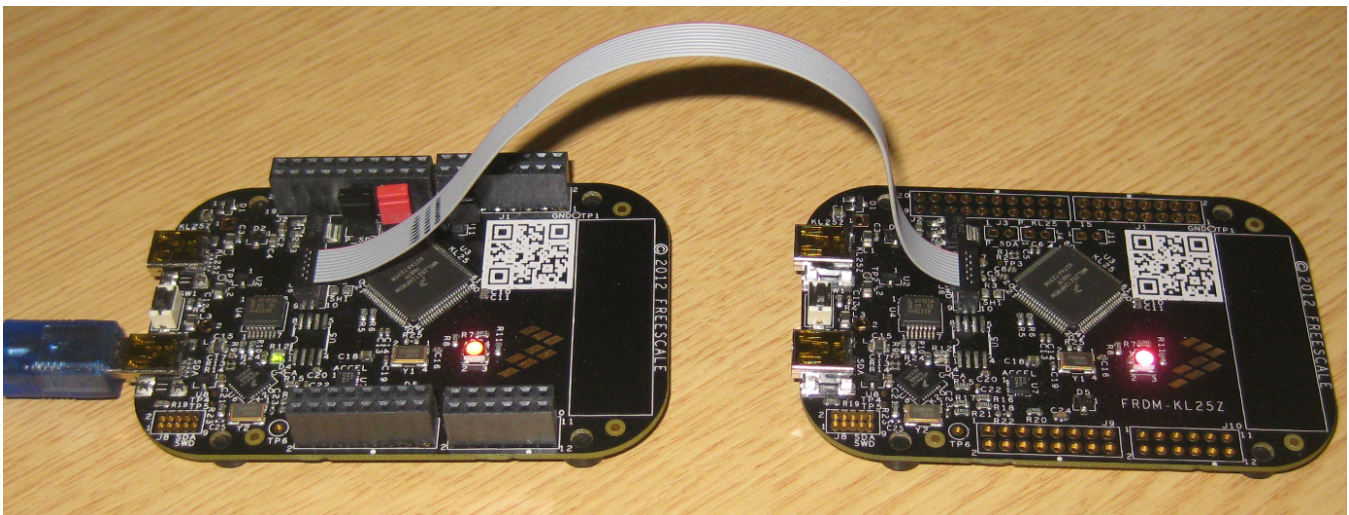
#### **NOTE**

Cutting the J11 trace is only needed for the Freedom *programmer* board. If you want to program another Freedom board with the programmer, the *target* board only needs the 10pin ARM Cortex Header populated. Otherwise, you do not need to modify the target board. If you are using a custom target board, then you only need ARM Cortex header for the connection.

## **2. Connect the Boards**

You have to connect the two boards with the 10pin flat cable, making sure pin 1 connects to pin 1:





**Figure 2-159. Connected Boards**

### Tip

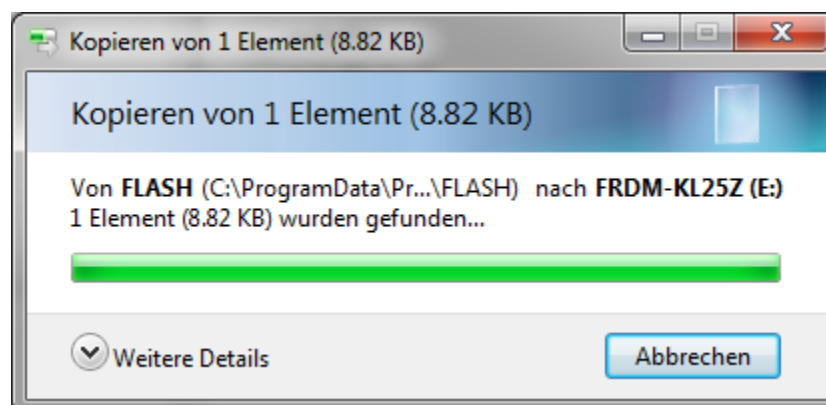
The debug cable provides 3.3V for the target board. In the case of the Freedom board, that's sufficient to power the target board. Otherwise make sure you properly power the target board.

## 3. Program the board with MSD

To program the board with MSD (Mass Storage Device), you need to:

- a. Power the board with reset button pressed
- b. Wait until it enumerates as 'Bootloader' device
- c. Copy the MSD-FRDM-KL25Z\_Pemicro\_v109.SDA to it
- d. Re-power the board (without reset button pressed)

The MSD Bootloader accepts S19 files. You can either drag&drop or copy the files to the bootloader device, and it will program the target board microcontroller (instead the resident microprocessor):



**Figure 2-160. S-Record Generation with gcc for ARM/Kinetis**

## 2.6.74 **NEW** How do I use printf() with the FRDM-KL25Z Board without using the Processor Expert?

To use printf() with the FRDM-KL25Z Board without using the Processor Expert, follow these steps:

1. Create a new bareboard project, and in the Language and Build Tools Options page select **UART** from the I/O Support group.

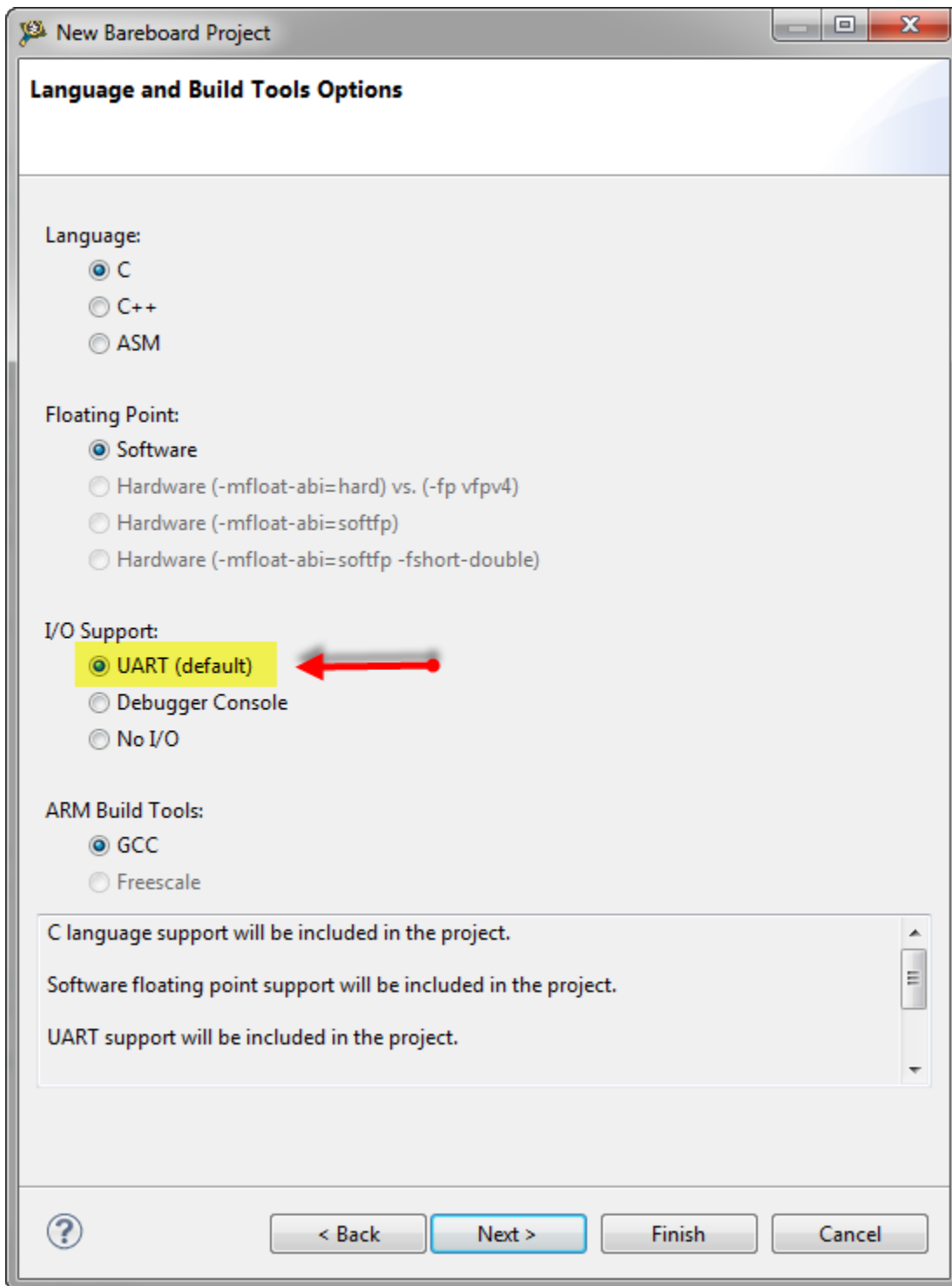


Figure 2-161. Language and Build Tools Options Page - Selecting Uart Option

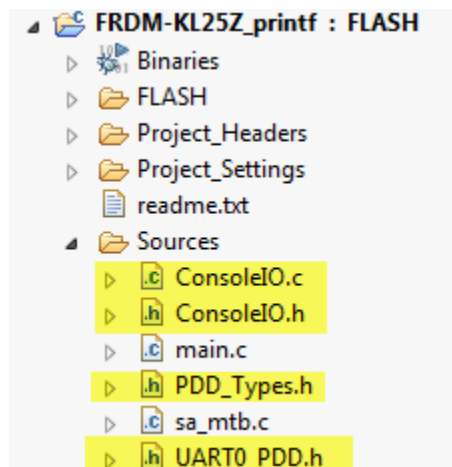
### NOTE

For more information about creating a new project using the **New Bareboard Project** wizard, refer to the *Codewarrior For Microcontrollers Targeting Manual*.

- Copy the `ConsoleIO.c`, `ConsoleIO.h`, `PDD_Types.h` and `UART0_PDD.h` files for the TWR-KL25Z from the following location in the `Sources` folder of your project:

`CW_InstallDir\MCU\ARM_GCC_Support\UART\TWR-KL25Z128`

where, `CW_InstallDir` is the installation directory of your CodeWarrior software.



**Figure 2-162. Copied Low Level UART Support Files**

- The TWR-KL25Z uses UART0 with PTA14/PTA15, while the FRDM-KL25Z uses UART0, but PTA1/PTA2, So you need to change the pin settings. For this disable the settings for the Tower board in `ConsoleIO.c` and `ConsoleIO_Init()`, and add changes to use PTA1/PTA2, as the following code shows:

```
void ConsoleIO_Init()
{
    InitClock();

    /* SIM_SCGC4: UART0=1 */
    SIM_SCGC4 |= SIM_SCGC4_UART0_MASK;
#ifdef TWR /* TWR version: PTA14, PTA15 */
    /* PORTA_PCR15: ISF=0,MUX=3 */
    PORTA_PCR15 = (uint32_t)((PORTA_PCR15 & (uint32_t)~(uint32_t)(
        PORT_PCR_ISF_MASK |
        PORT_PCR_MUX(0x04)
    )) | (uint32_t)(
        PORT_PCR_MUX(0x03)
    ));
    /* PORTA_PCR14: ISF=0,MUX=3 */
    PORTA_PCR14 = (uint32_t)((PORTA_PCR14 & (uint32_t)~(uint32_t)(
        PORT_PCR_ISF_MASK |
        PORT_PCR_MUX(0x04)
    )) | (uint32_t)(
        PORT_PCR_MUX(0x03)
    ));
#else /* FRDM-KL25Z: PTA1/PTA2 */
    /* PORTA_PCR1: ISF=0,MUX=2 */
    PORTA_PCR1 = (uint32_t)((PORTA_PCR1 & (uint32_t)~0x01000500UL) |
        (uint32_t)0x0200UL);
    /* PORTA_PCR2: ISF=0,MUX=2 */
    PORTA_PCR2 = (uint32_t)((PORTA_PCR2 & (uint32_t)~0x01000500UL) |
        (uint32_t)0x0200UL);
#endif
    UART0_PDD_EnableTransmitter(UART0_BASE_PTR, PDD_DISABLE); /* Disable transmitter. */
    UART0_PDD_EnableReceiver(UART0_BASE_PTR, PDD_DISABLE); /* Disable receiver. */
    /* UART0_C1: LOOPS=0,DOZEEN=0,RSRC=0,M=0,WAKE=0,ILT=0,PE=0,PT=0 */
}
```

```

UART0_C1 = 0x00U;                               /* Set the C1 register */
/* UART0_C3: R8T9=0,R9T8=0,TXDIR=0,TXINV=0,ORIE=0,NEIE=0,FEIE=0,PEIE=0 */
UART0_C3 = 0x00U;                               /* Set the C3 register */
/* UART0_S2: LBKDIF=0,RXEDGIF=0,MSBF=0,RXINV=0,RWUID=0,BRK13=0,LBKDE=0,RAF=0 */
UART0_S2 = 0x00U;                               /* Set the S2 register */
UART0_PDD_SetClockSource(UART0_BASE_PTR, UART0_PDD_PLL_FLL_CLOCK);
UART0_PDD_SetBaudRate(UART0_BASE_PTR, 313U); /* Set the baud rate register. */
UART0_PDD_SetOversamplingRatio(UART0_BASE_PTR, 3U);
UART0_PDD_EnableSamplingOnBothEdges(UART0_BASE_PTR, PDD_ENABLE);
UART0_PDD_EnableTransmitter(UART0_BASE_PTR, PDD_ENABLE); /* Enable transmitter */
UART0_PDD_EnableReceiver(UART0_BASE_PTR, PDD_ENABLE); /* Enable receiver */
}

```

The following code shows an example to print a *hello world* from the `main()` in `main.c`, to test the changes you made:

```

/*
 * main implementation: use this 'C' sample to create your own application
 *
 */

#include "derivative.h" /* include peripheral declarations */
#include
#include "ConsoleIO.h"

int main(void)
{
    int counter = 0;

    ConsoleIO_Init();
    for(;;) {
        counter++;
        printf("Hello world!\r\n");
    }
    return 0;
}

```

### Tip

Include the correct header files and call `ConsoleIO_Init()` while you create your project.

Build and run the project, the message prints and appears on the console.

## 2.6.75 NEW How does a bootloader work?

The following process tells you how a bootloader works:

When you enter the command `BL Load S19` at the command prompt, it executes the function `BL_LoadS19()` in `Bootloader.c`. The following listing shows an example:

```

static uint8_t BL_LoadS19(CLS1_ConstStdIOType *io) {
    unsigned char buf[16];
    uint8_t res = ERR_OK;

    /* first, erase flash */
    if (BL_EraseAppFlash(io) != ERR_OK) {
        return ERR_FAILED;
    }
}

```

```

}

/* load S19 file */
CLS1_SendStr((unsigned char*)"Waiting for the S19 file...", io->stdOut);
parserInfo.GetCharIterator = GetChar;
parserInfo.voidP = (void*)io;
parserInfo.S19Flash = BL_onS19Flash;
parserInfo.status = S19_FILE_STATUS_NOT_STARTED;
parserInfo.currType = 0;
parserInfo.currAddress = 0;
parserInfo.codeSize = 0;
parserInfo.codeBuf = codeBuf;
parserInfo.codeBufSize = sizeof(codeBuf);
while (AS1_GetCharsInRxBuf()>0) { /* clear any pending characters in rx buffer */
    AS1_ClearRxBuf();
    WAIT1_Waitms(100);
}
do {
    if (S19_ParseLine(&parserInfo)!=ERR_OK) {
        CLS1_SendStr((unsigned char*)"ERROR!\r\nFailed at address 0x", io->stdErr);
        buf[0] = '\0';
        UTIL1_strcatNum32Hex(buf, sizeof(buf), parserInfo.currAddress);
        CLS1_SendStr(buf, io->stdErr);
        CLS1_SendStr((unsigned char*)"\r\n", io->stdErr);
        res = ERR_FAILED;
        break;
    } else {
        CLS1_SendStr((unsigned char*)"\r\nS", io->stdOut);
        buf[0] = parserInfo.currType;
        buf[1] = '\0';
        CLS1_SendStr(buf, io->stdOut);
        CLS1_SendStr((unsigned char*)" address 0x", io->stdOut);
        buf[0] = '\0';
        UTIL1_strcatNum32Hex(buf, sizeof(buf), parserInfo.currAddress);
        CLS1_SendStr(buf, io->stdOut);
    }
    if (parserInfo.currType=='7' || parserInfo.currType=='8' || parserInfo.currType=='9') {
        /* end of records */
        break;
    }
} while (1);
if (res==ERR_OK) {
    CLS1_SendStr((unsigned char*)"\r\ndone!\r\n", io->stdOut);
} else {
    while (AS1_GetCharsInRxBuf()>0) { /* clear buffer */
        AS1_ClearRxBuf();
        WAIT1_Waitms(100);
    }
    CLS1_SendStr((unsigned char*)"\r\nfailed!\r\n", io->stdOut);
    /* erase flash again to be sure we do not have illegal application image */
    if (BL_EraseAppFlash(io)!=ERR_OK) {
        res = ERR_FAILED;
    }
}
return res;
}

```

It first fills a callback structure of type `S19_ParserStruct`, as the following listing shows:

```

typedef struct S19_ParserStruct {
    uint8_t (*GetCharIterator)(uint8_t*, void*); /* character stream iterator */
    void *voidP; /* void pointer passed to iterator function */
    uint8_t (*S19Flash)(struct S19_ParserStruct*); /* called for each S19 line to be flashed */
    /* the following fields will be used by the iterator */
    S19_FileStatus status; /* current status of the parser */
    uint8_t currType; /* current S19 record, e.g. 1 for S1 */
    uint32_t currAddress; /* current code address of S19 record */
    uint16_t codeSize; /* size of code in bytes in code buffer */
    uint8_t *codeBuf; /* code buffer */
}

```

```

uint16_t codeBufSize; /* total size of code buffer, in bytes */
} S19_ParserStruct;
    
```

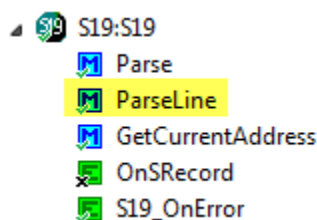
That structure contains a callback to read from the input stream, as the following listing shows:

```

static uint8_t GetChar(uint8_t *data, void *q) {
    CLS1_ConstStdIOType *io;

    io = (CLS1_ConstStdIOType*)q;
    if (!io->keyPressed()) {
#ifdef USE_XON_XOFF
        SendXONOFF(io, XON);
#endif
        while(!io->keyPressed()) {
            /* wait until there is something in the input buffer */
        }
#ifdef USE_XON_XOFF
        SendXONOFF(io, XOFF);
#endif
#ifdef
        io->stdIn(data); /* read character */
        if (*data=='\0') { /* end of input? */
            return ERR_RXEMPTY;
        }
        return ERR_OK;
    }
}
    
```

Parsing of the S19 file is done in S19\_ParesLine() which is implemented in a Processor Expert component, as the following figure shows:



**Figure 2-163. S19 Parser**

This parser calls the callback BL\_onS19Flash() for every S19 line:

```

static uint8_t BL_onS19Flash(S19_ParserStruct *info) {
    uint8_t res = ERR_OK;

    switch (info->currType) {
        case '1':
        case '2':
        case '3':
            if (!BL_ValidAppAddress(info->currAddress)) {
                info->status = S19_FILE_INVALID_ADDRESS;
                res = ERR_FAILED;
            } else {
                /* Write buffered data to Flash */
                if (BL_Flash_Prog(info->currAddress, info->codeBuf, info->codeSize) != ERR_OK) {
                    info->status = S19_FILE_FLASH_FAILED;
                    res = ERR_FAILED;
                }
            }
            break;
        case '7':
        case '8':
        case '9': /* S7, S8 or S9 mark the end of the block/s-record file */
    
```

## miscellaneous

```

        break;
    case '0':
    case '4':
    case '5':
    case '6':
    default:
        break;
} /* switch */
return res;
}

```

The S1, S2 and S3 records contains the code. With `BL_ValidAppAddress()` it checks if the address is within the application FLASH memory range:

```

/*!
 * \brief Determines if the address is a valid address for the application (outside the
 * bootloader)
 * \param addr Address to check
 * \return TRUE if an application memory address, FALSE otherwise
 */
static bool BL_ValidAppAddress(dword addr) {
return ((addr>=MIN_APP_FLASH_ADDRESS) && (addr<=MAX_APP_FLASH_ADDRESS)); /* must be in
application space */
}

```

On successful execution, it flashes the memory block:

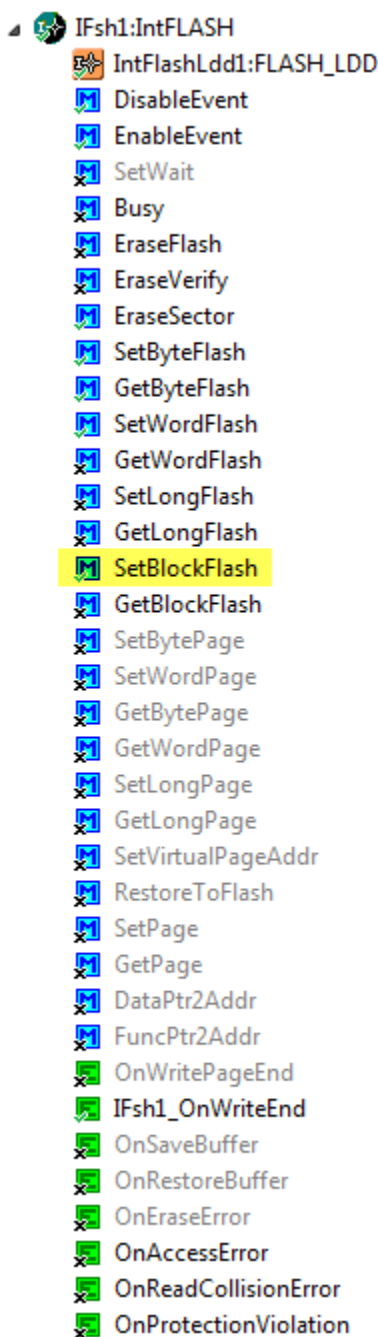
```

/*!
 * \brief Performs flash programming
 * \param flash_addr Destination address for programming.
 * \param data_addr Pointer to data.
 * \param nofDataBytes Number of data bytes.
 * \return ERR_OK if everything was ok, ERR_FAILED otherwise.
 */
static byte BL_Flash_Prog(dword flash_addr, uint8_t *data_addr, uint16_t nofDataBytes) {
    /* only flash into application space. Everything else will be ignored */
    if(BL_ValidAppAddress(flash_addr)) {
        if (!IFsh1_SetBlockFlash((IFsh1_TDataAddress)data_addr, flash_addr, nofDataBytes) !=
ERR_OK) {
            return ERR_FAILED; /* flash programming failed */
        }
    }
    return ERR_OK;
}

```

The Flash Programming is performed by the **IntFLASH** Processor Expert components:





**Figure 2-164. Processor Expert Flash Programming Component**

You can also use this component for erasing:

```

/!*
 * \brief Erases all unprotected pages of flash
 * \return ERR_OK if everything is ok; ERR_FAILED otherwise
 */
static byte BL_EraseApplicationFlash(void) {
    dword addr;

    /* erase application flash pages */
    for(addr=MIN_APP_FLASH_ADDRESS;addr<=MAX_APP_FLASH_ADDRESS;addr+=FLASH_PAGE_SIZE) {
        if(IFsh1_EraseSector(addr) != ERR_OK) { /* Error Erasing Flash */
            return ERR_FAILED;
        }
    }
}
    
```

miscellaneous

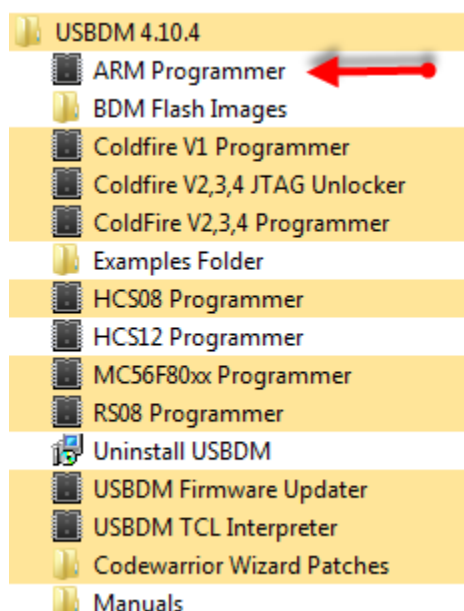
```

    }
  }
  return ERR_OK;
}

```

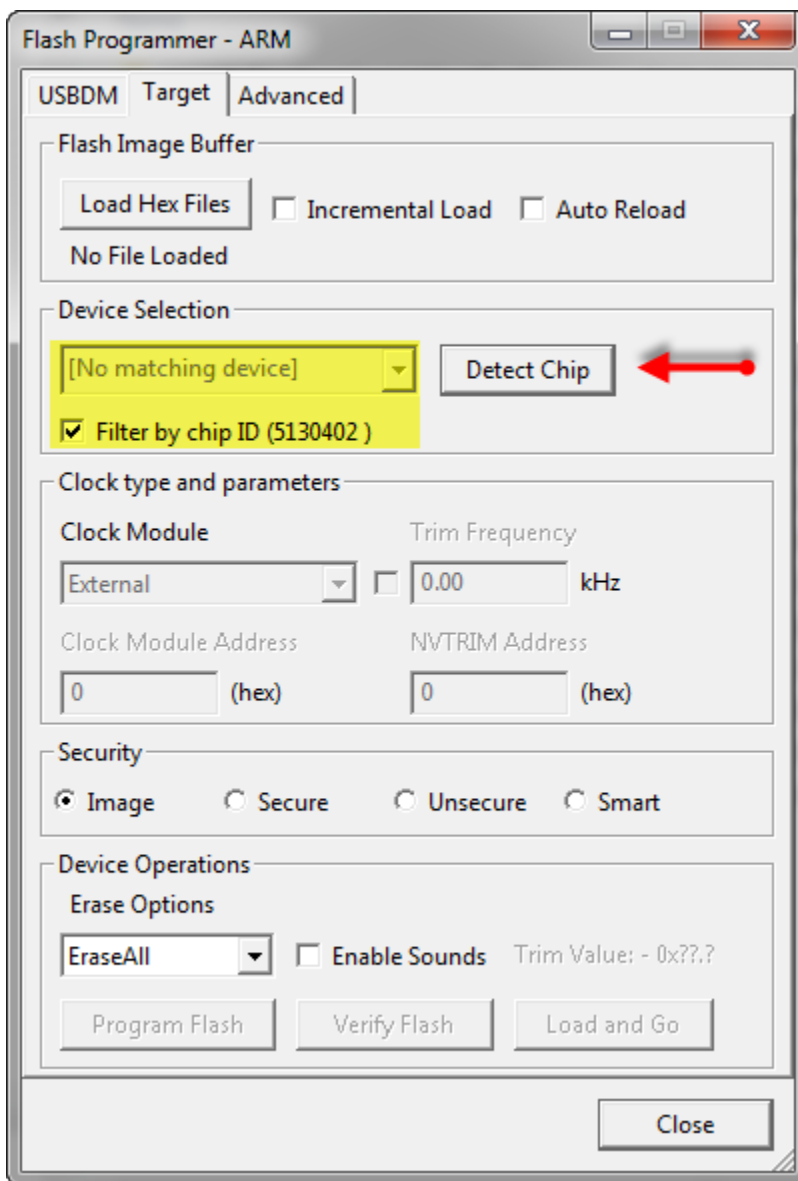
## 2.6.76 NEW I am debugging FRDM-KL05Z with USBDM, but I received a 'Failed to resume target process., Downloading binary to target..' error message. What do I do?

To solve this problem, use the standalone flash programmers with ARM based devices that comes with USBDM.



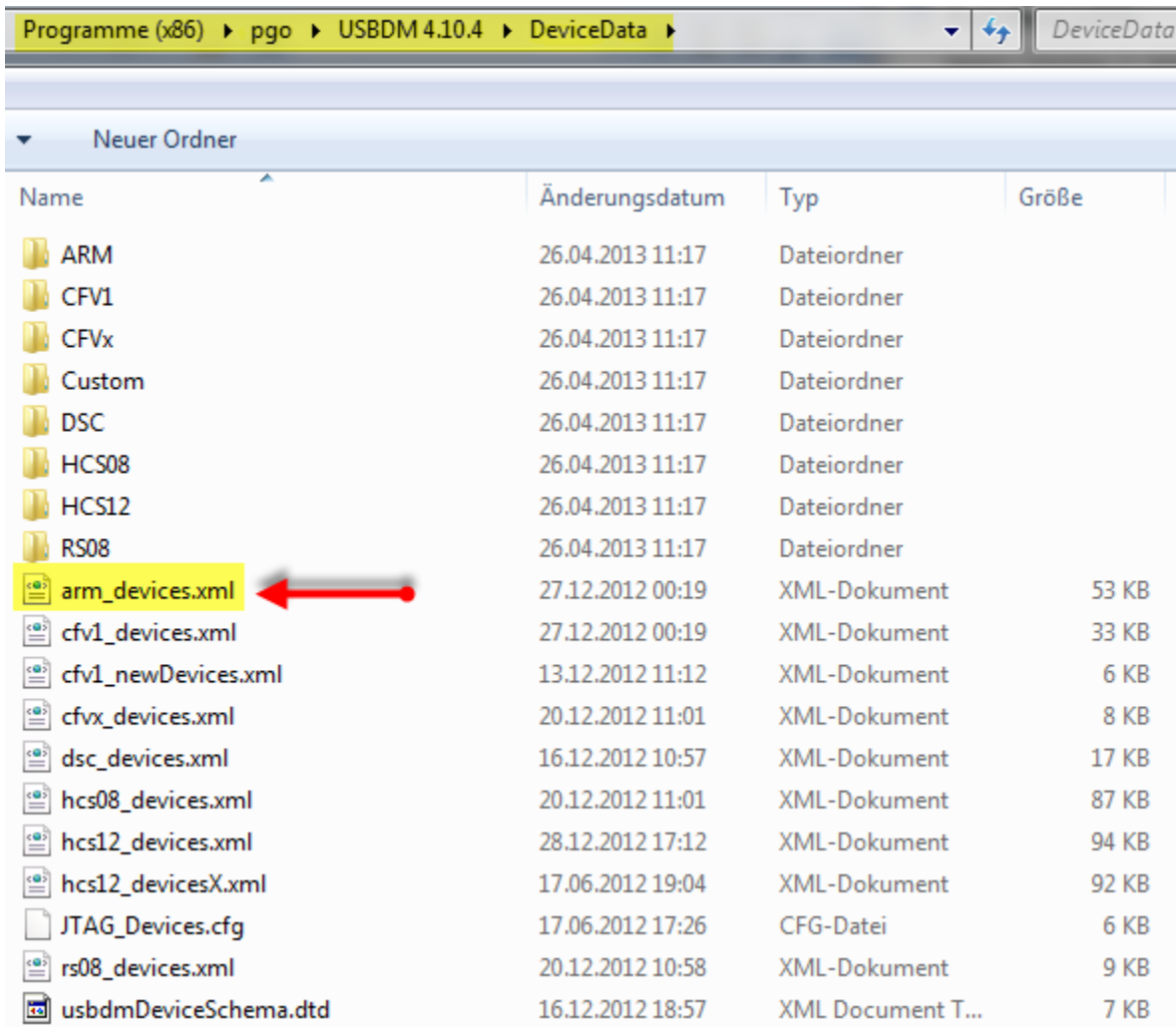
**Figure 2-165. USBDM 4.10.4 Installed Program Shortcuts**

The Flash Programmer has a *Detect* functionality, in this case it is not able to detect the KL05Z32 on the FRDM-KL05Z board:



**Figure 2-166. Flash Programmer for ARM**

Flash Programmer for ARM also reports the chip ID. The ID mapping is described in the `arm_devices.xml` file inside the USBDM installation folder:



**Figure 2-167. arm\_devices.xml Configuration**

The arm\_devices.xml file has the KL05Z32 listed as:

```

1 | MKL05Z32M4" family="ARM" subfamily="KL05">
2 |     <memoryRef ref="kinetisL4K_Ram" />
3 |     <memoryRef ref="kinetis32K_Flash_B0" />
4 |     <memoryRef ref="kinetisIO_KL" />
5 |     <sdid value="0x05130480" />
6 |     <tclScriptRef ref="Kinetis-KLxx-Scripts" />
7 |     <flashProgramRef ref="Kinetis-KLxx-FlashProgram" />
8 | </device>

```

**Figure 2-168. arm\_devices.xml**

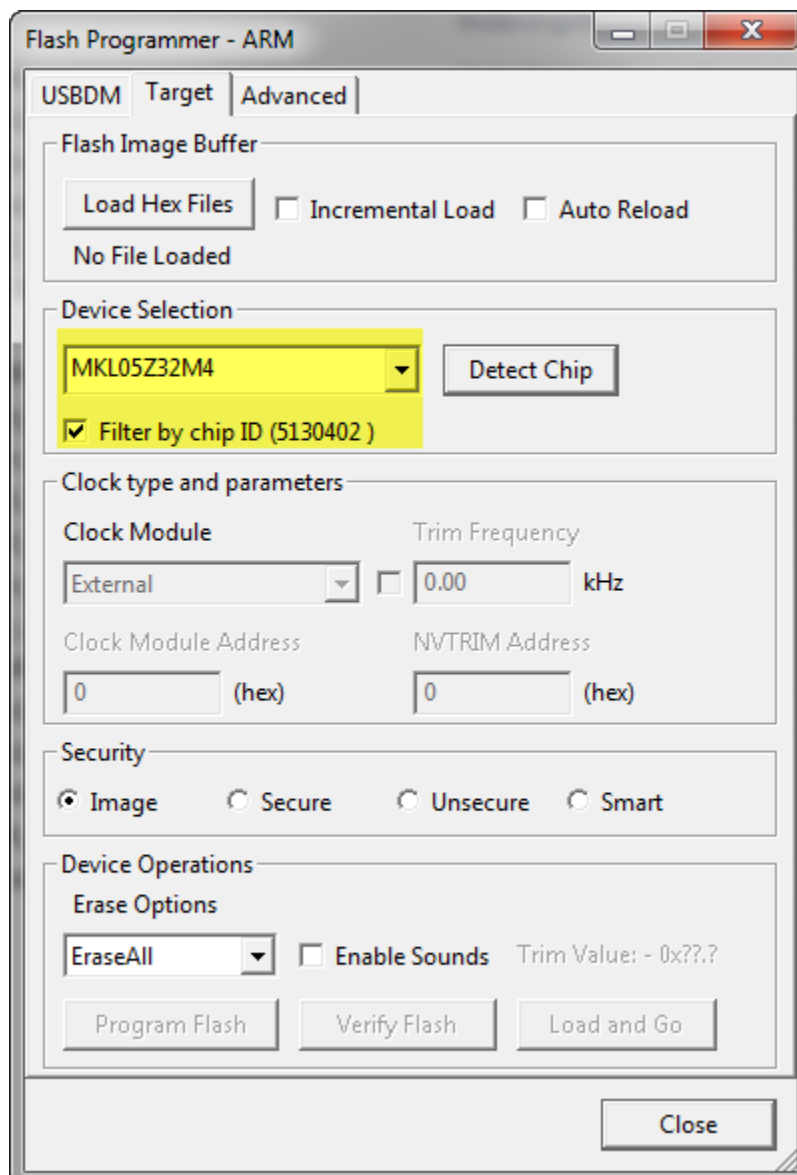
As the code in the above figure shows that the device ID (0x05130480) does not match what the programmer reported (0x5130402). Testing the ID with the working FRDM-KL25Z, it shows that the last nibble is zero, so you have to change the above entry, as the following figure shows:

```

1 MKL05Z32M4" family="ARM" subfamily="KL05">
2   <memoryRef ref="kinetisL4K_Ram" />
3   <memoryRef ref="kinetis32K_Flash_B0" />
4   <memoryRef ref="kinetisIO_KL" />
5   <sdid value="0x05130420" />
6   <tc1ScriptRef ref="Kinetis-KLxx-Scripts" />
7   <flashProgramRef ref="Kinetis-KLxx-FlashProgram" />
8 </device>
    
```

**Figure 2-169. Changed arm\_devices.xml**

With this your device detects properly, as the following figure shows:



**Figure 2-170. MKL05Z32M4 Detected**

Now you can successfully debug the FRDM-KL05Z with USBDM using CodeWarrior for MCU10.x.

### 2.6.77 **NEW** My bootloader resides in the first lower flash pages. How do I avoid it to get destroyed and overwritten by the application?

To avoid the bootloader from getting destroyed and overwritten by the application, you have to protect the bootloader flash blocks. To protect, you have to specify the value as **Protected** in the **Value** column of the **Protected regions** group in the **Component Inspector - Cpu** tab of the CPU component properties, where you can protect 4 KByte regions, as the following figure shows:

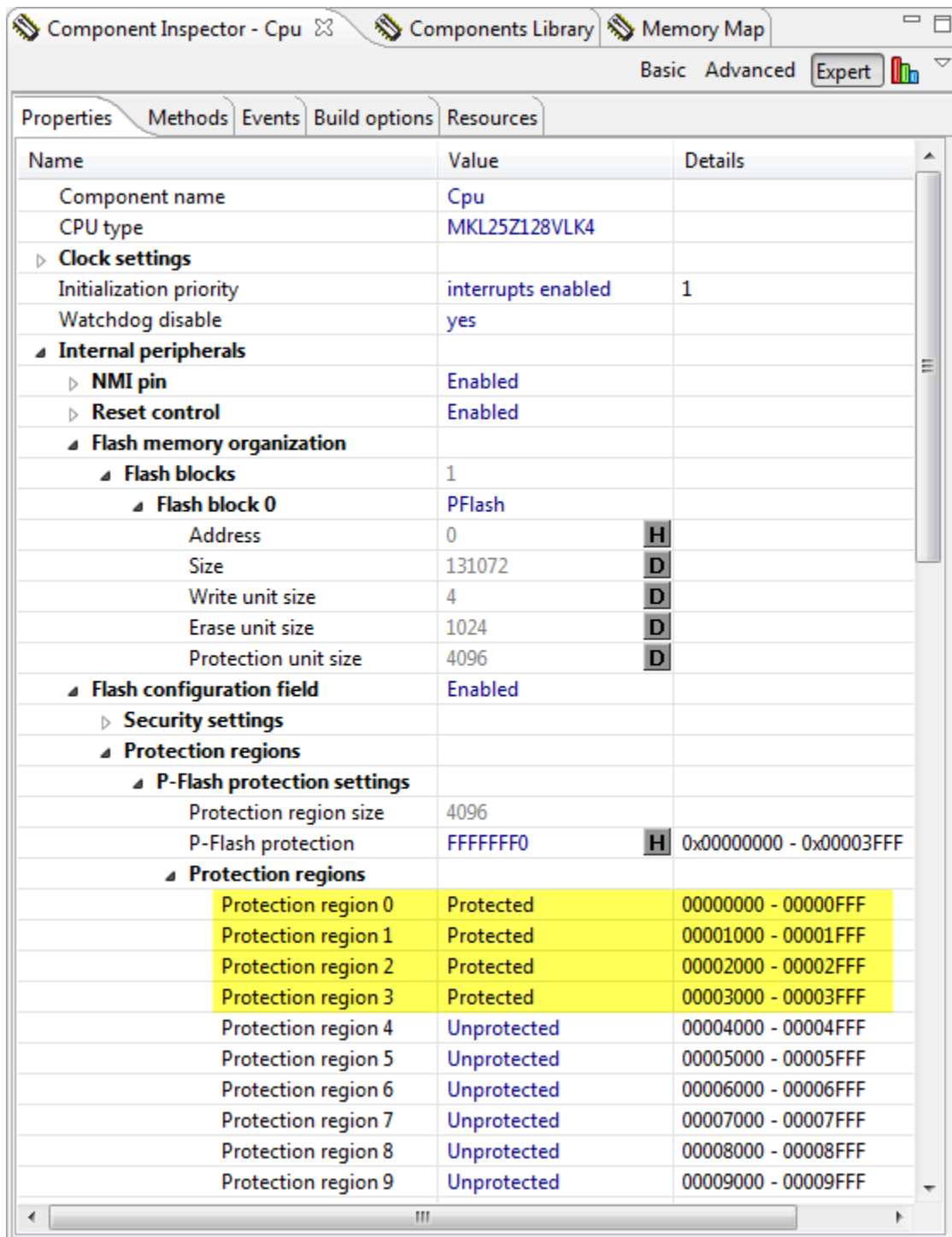


Figure 2-171. Protecting Regions in the CPU Component

## 2.6.78 NEW The OpenSDA connection is not available in the Connection page of the wizard. What do I do?

You can use the normal P&E Multilink connection instead. For that, you only have to change the **Interface** in the debugger settings from P&E Multilink to OpenSDA in the **Connection** tab.

To do this:

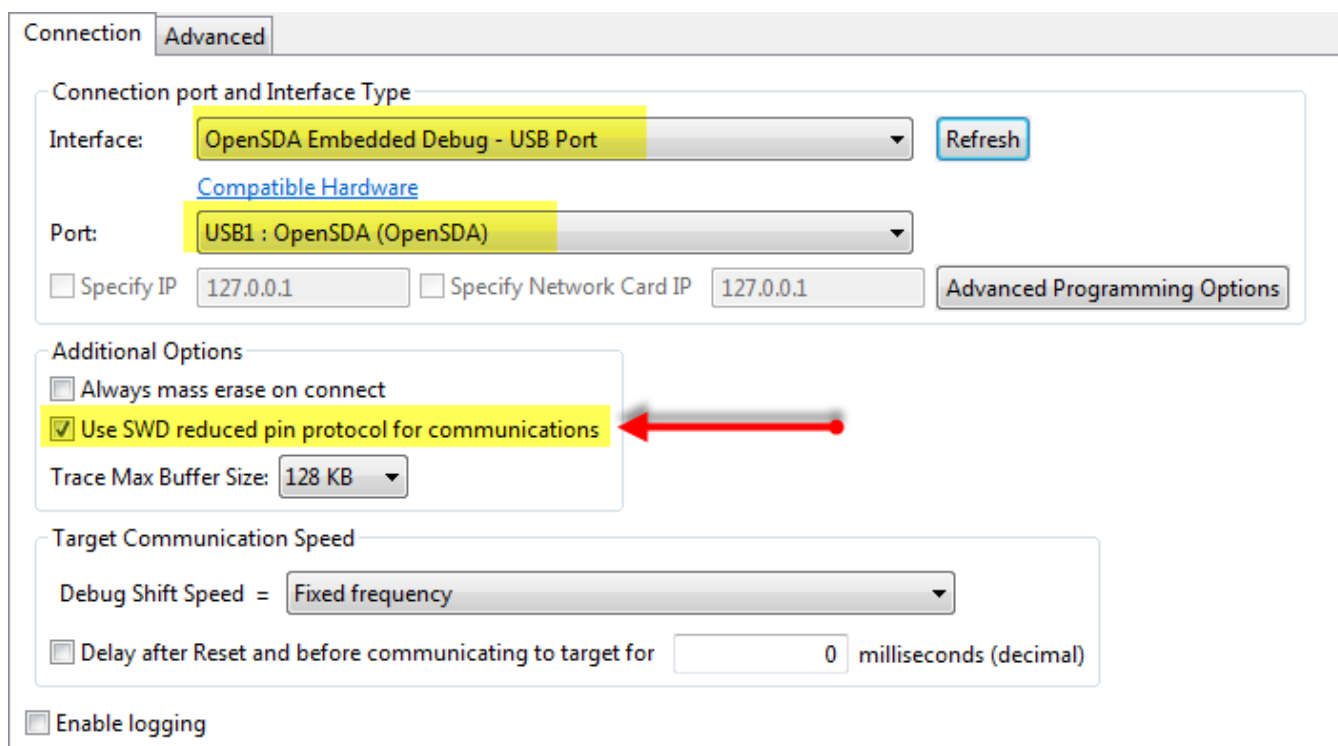
1. Select **Run > Debug Configurations** from the IDE menu bar.
2. Select the desired build configuration from the left panel.

The **Main** tab appears in the right panel.

3. Click the **Edit** button from the **Connection** group.

The **Properties for >connection\_name<** dialog box appears.

4. In the **Connection port and Interface Type** group:
  - a. Select the **OpenSDA Embedded Debug – USB Port** option from the **Interface** drop-down list.
  - b. Select the **USB1 : OpenSDA (OpenSDA)** option from the **Port** drop-down list.
5. Make sure to check the **Use SWD reduced pin protocol for communications** checkbox from the **Additional Options** group.



**Figure 2-172. OpenSDA Debug Connection for FRDM-K20D50M**



**NOTE**

In case the **Use SWD reduced pin protocol for communications** option is not enabled, the debugger throws the error, “ERR: Can not enter background mode”.

**2.6.79** **NEW** **What are the disadvantages of using printf() in the embedded system programming?**

The major disadvantages of using printf() in the embedded system programming are as follows:

- Using printf() adds greatly to the code size of the application. This is because, as defined by the ANSI library standard, printf() supports all the different format string, including formatting octal numbers, floating point, etc even if you do not use them in your programming.
- It uses a lot of stack space. This is because, printf() handles all the formatters plus the variable argument list. Depending on the printf() implementation, it can easily add 512-1024 bytes on the stack, resulting in stack overflow.
- Printf() is a constant source of programming errors. For example:

```
void printString(char *str) {  
    printf(str);  
}
```

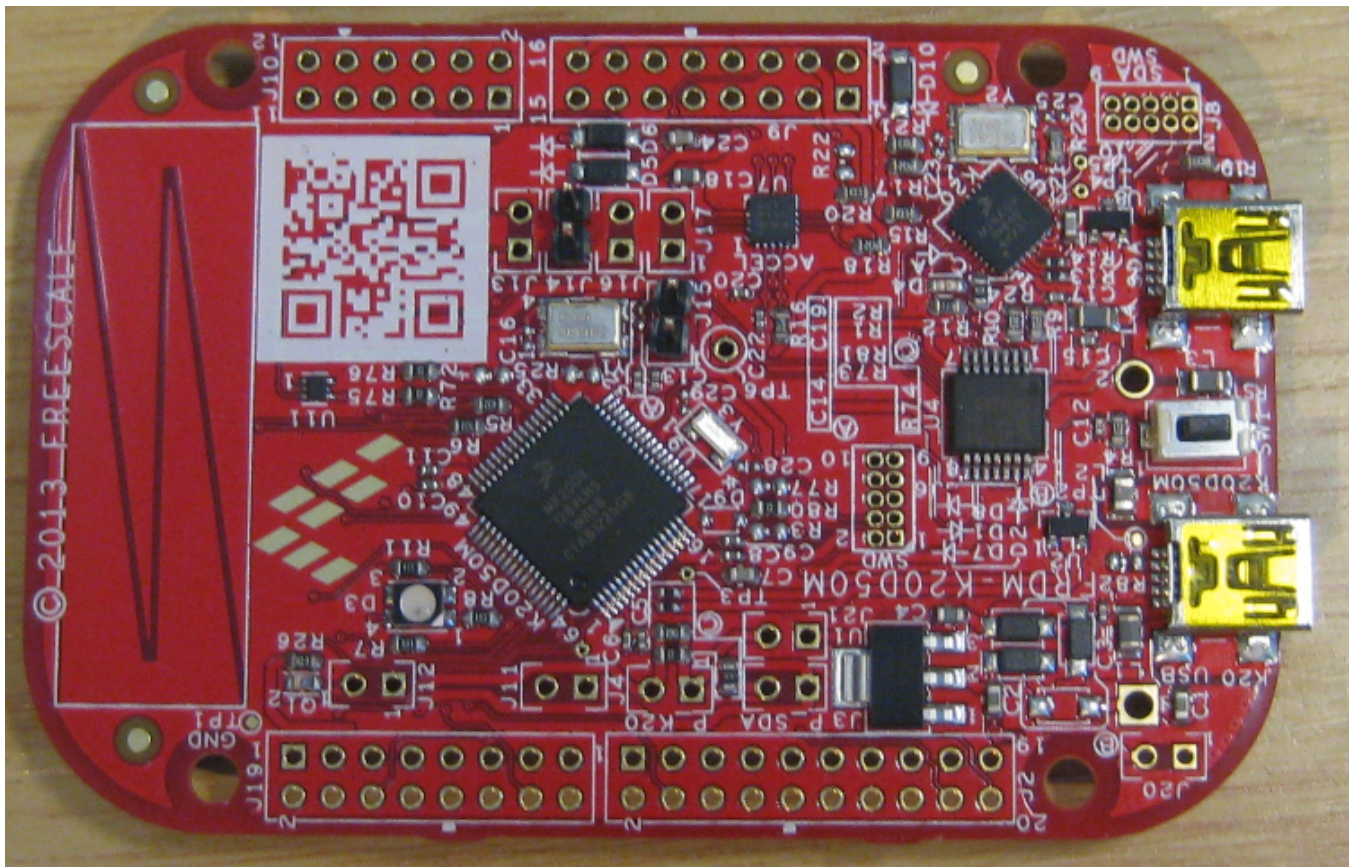
When the the function received a string with % in it, it crashes or opens a program vulnerability. This is because the printf() parses the % and then will pop arguments from the stack which do not exist, causing a stack error.

Hence, it is not recommended to use printf() in embedded system programming.

**2.6.80** **NEW** **What are the features of freedom board FRDM-K20D50M?**

The FRDM-K20D50M is Freescale’s freedom board with ARM Cortex-M4 core. It comes with the 4 dual-row headers, a temperature sensor and an ambient light sensor.

The following figure shows the FRDM-K20D50M board:



**Figure 2-173. Freedom Board FRDM-K20D50M**

The following are the new features of the FRDM-K20D50M board compared to the previous FRDM-KL25Z board:

- There are two crystals: one 32 kHz and one 8 MHz one
- New analog Ambient Light Sensor with J12 jumper to isolate it
- New Temperature Sensor (U11) with jumper J13 to isolate it
- Two jumpers (J16 and J17) to disconnect the I2C lines to the accelerometer
- Jumper J14 to cut the reset line between OpenSDA and the target CPU
- Jumper J13 to isolate the BD1020HFV-TR temperature sensor
- Jumper J21 to enable USB Host support (unprotected)

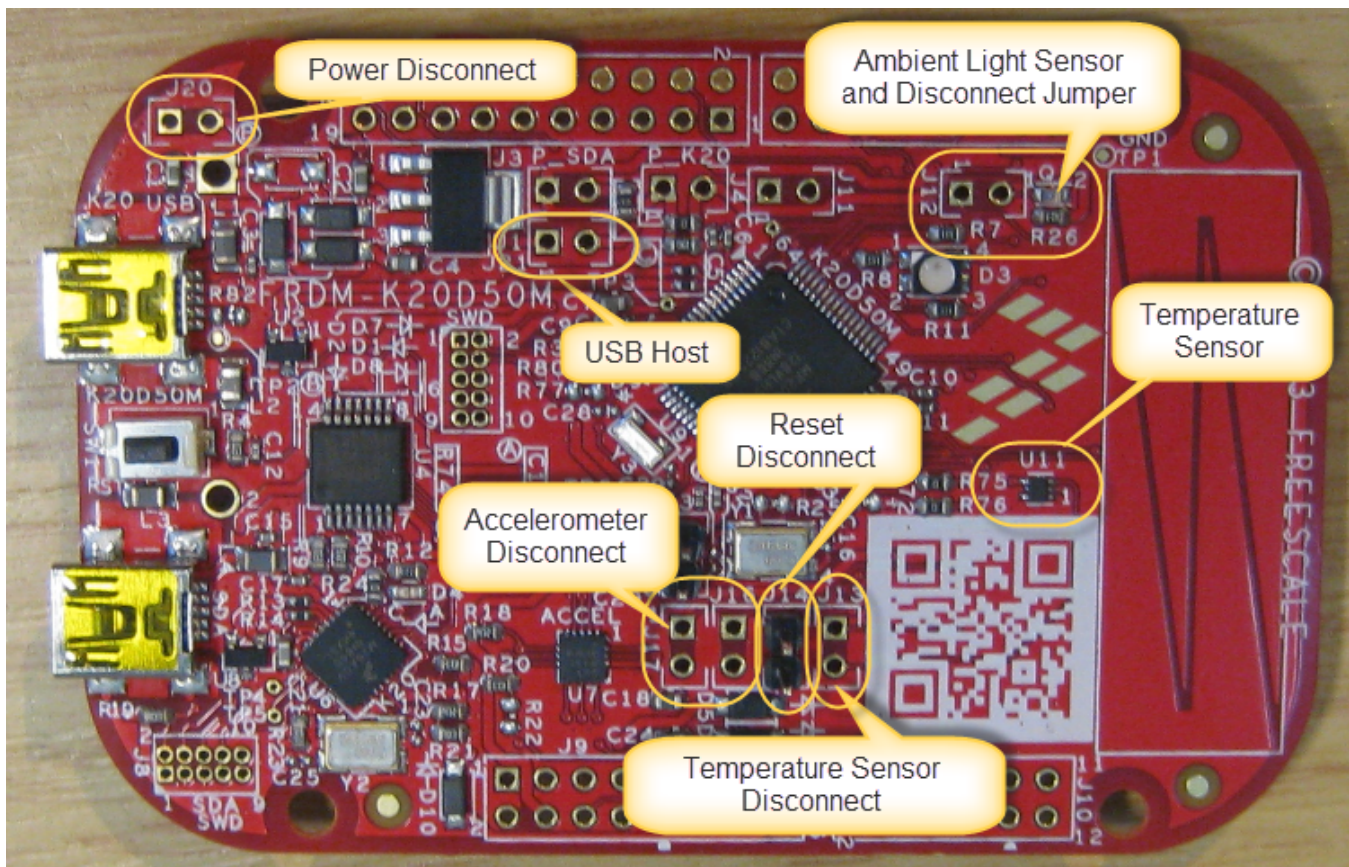


Figure 2-174. Changes on board compared to FRDM-KL25Z

### 2.6.81 NEW What Bootloader Shell commands I can use?

You can use the following bootloader shell commands:

```
CLS1 help | status
```

This command prints help or status information.

```
BL help
```

This command prints help information.

```
BL status
```

This command shows the application flash range, and the content of the application vector table.

```
BL erase
```

This command erases the application flash blocks.

```
BL restart
```

This command restarts the user application (if any) with jump to the reset vector.

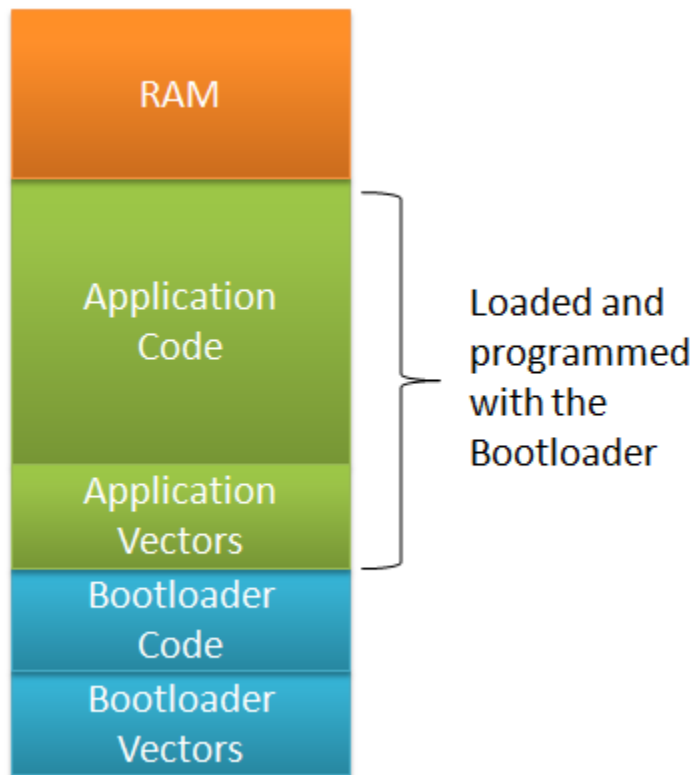
```
BL load s19
```

This command loads the S19 file.

## 2.6.82 NEW What is a bootloader?

A bootloader is a program which is able to load another program (the application program). Typically, the bootloader program is not changed, and is kept in the microcontroller. That way the bootloader loads different program again and again.

The following figure shows a typical diagram of a serial bootloader:



**Figure 2-175. Serial Bootloader**

### NOTE

Depending on the architecture, there can be a mini or micro bootloader that can load the real bootloader. For example, the OpenSDA bootloader on the Freedom boards have this capability.

The **Bootloader Code** and the **Bootloader Vectors** are programmed into a new part (for example, with a debugger) or, a standalone flash programmer (for example, with USBDM). Then the bootloader can be used to load or change the **Application Code** and **Application Vectors**. With this, the bootloader remains the same, while the application can be updated.

### 2.6.83 **NEW** What is Bootloader Sequence?

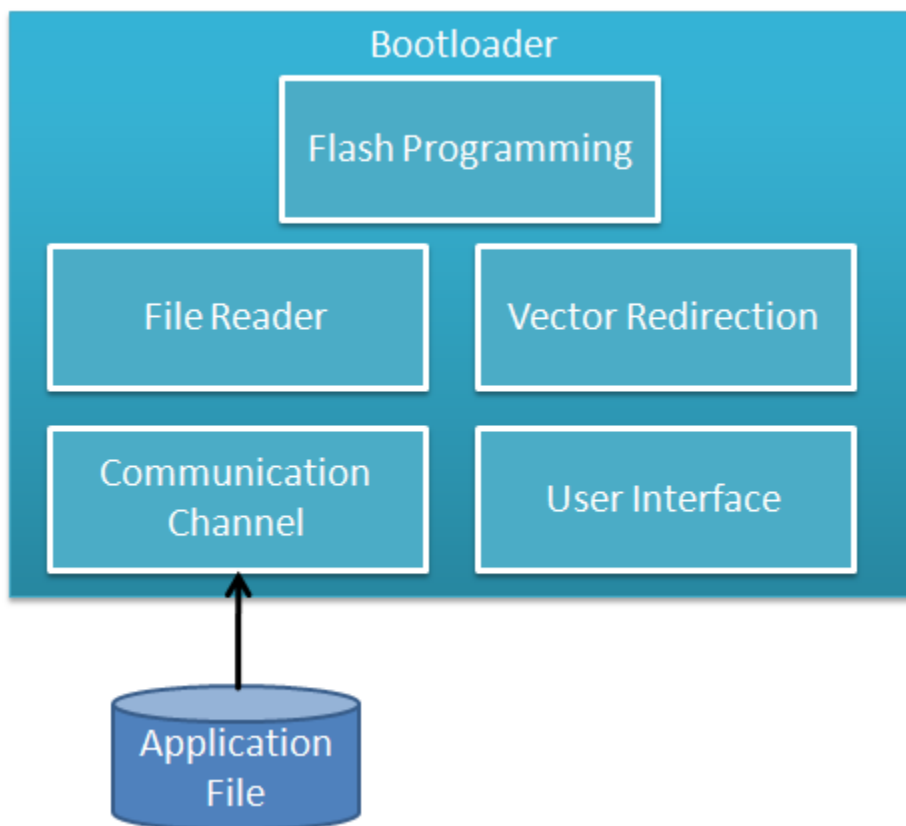
A typical bootloader has the following sequence:

1. At the startup the bootloader either enters in a bootloader mode or in the application running mode. Typically, this is decided with a button or jumper set (or removed). If the bootloader starts in application running mode, it calls the required application and the system starts automatically.
2. Otherwise, the bootloader reprograms the application with a new file. The S19 (S-Record) files are often used for this, as they are easy to parse and every tool chain can produce them.
3. The bootloader needs to use a communication channel to read that file. That can be RS-232, USB or an SD card file system (e.g. FatFS).
4. Using that file, the bootloader programs the flash memory. Special consideration has to be taken into account for the application vector table. When the bootloader runs out of reset, it uses its own (default) vector table, and needs to relocate the vector table if running the application.

#### **Tip**

It is possible to use the reset button on the FRDM-KL25Z board as a user. To keep things simple, you may use a dedicated bootloader push button on PTB8.

The following figure shows the Bootloader System Block Diagram:



**Figure 2-176. Bootloader System Block Diagram**

The components of the Bootloader System Block Diagram are as follows:

- **Communication Channel** - File I/O or any other means to read the Application File.
- **File Reader** - A reader which reads the Application File.
- **Flash Programming** - To program the Application.
- **Vector Redirection** - To switch between the Bootloader and Application Vector Table.
- **User Interface** - Showing status and information to the user, and to switch between the Application and Bootloader mode at system startup.

### 2.6.84 NEW What is the code size required for the bootloader for FRDM-KL25Z board?

For the FRDM-KL25Z board, the bootloader is compiled with gcc compiler. Without optimization (-O0), it needs 13 KByte of FLASH. But when optimized for size, it needs only 8 Kbyte.

## 2.6.85 NEW Where can I find the software for my FRDM-K20D50M board?

To get the software for your FRDM-K20D50M board, download the Quick-Start Package (FRDM-K20D50M\_QSP.zip) from the following location:

[https://www.freescale.com/webapp/sps/site/prod\\_summary.jsp?code=FRDM-K20D50M&fpssp=1&tab=Design\\_Tools\\_Tab](https://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=FRDM-K20D50M&fpssp=1&tab=Design_Tools_Tab)

The Quick-Start Package has an updated OpenSDA debug application and a mass storage bootloader, as the following figure shows:

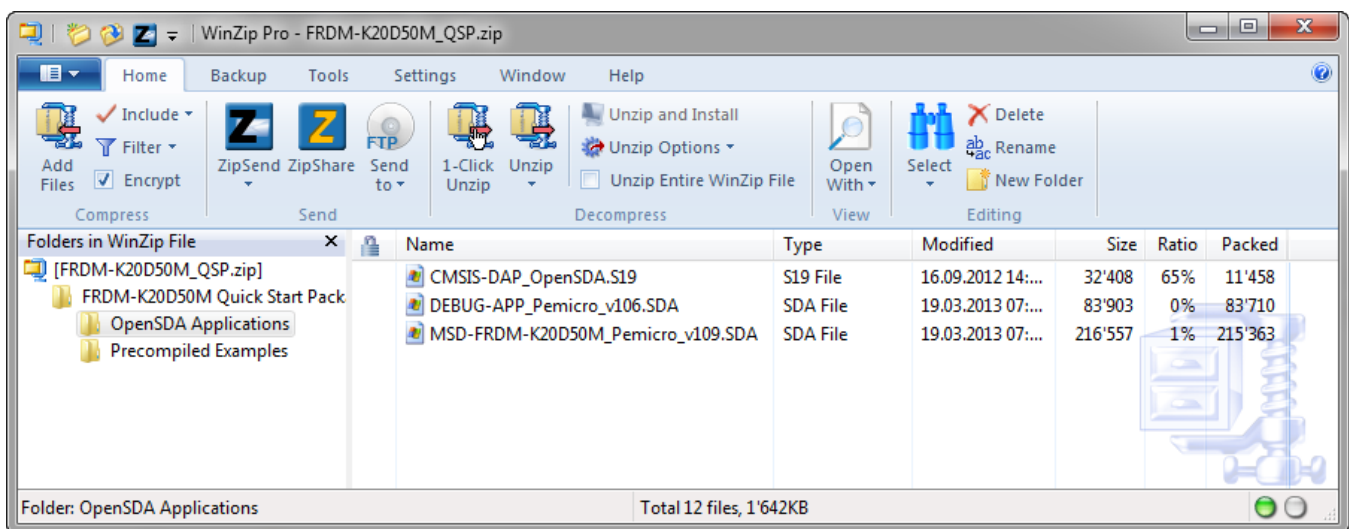
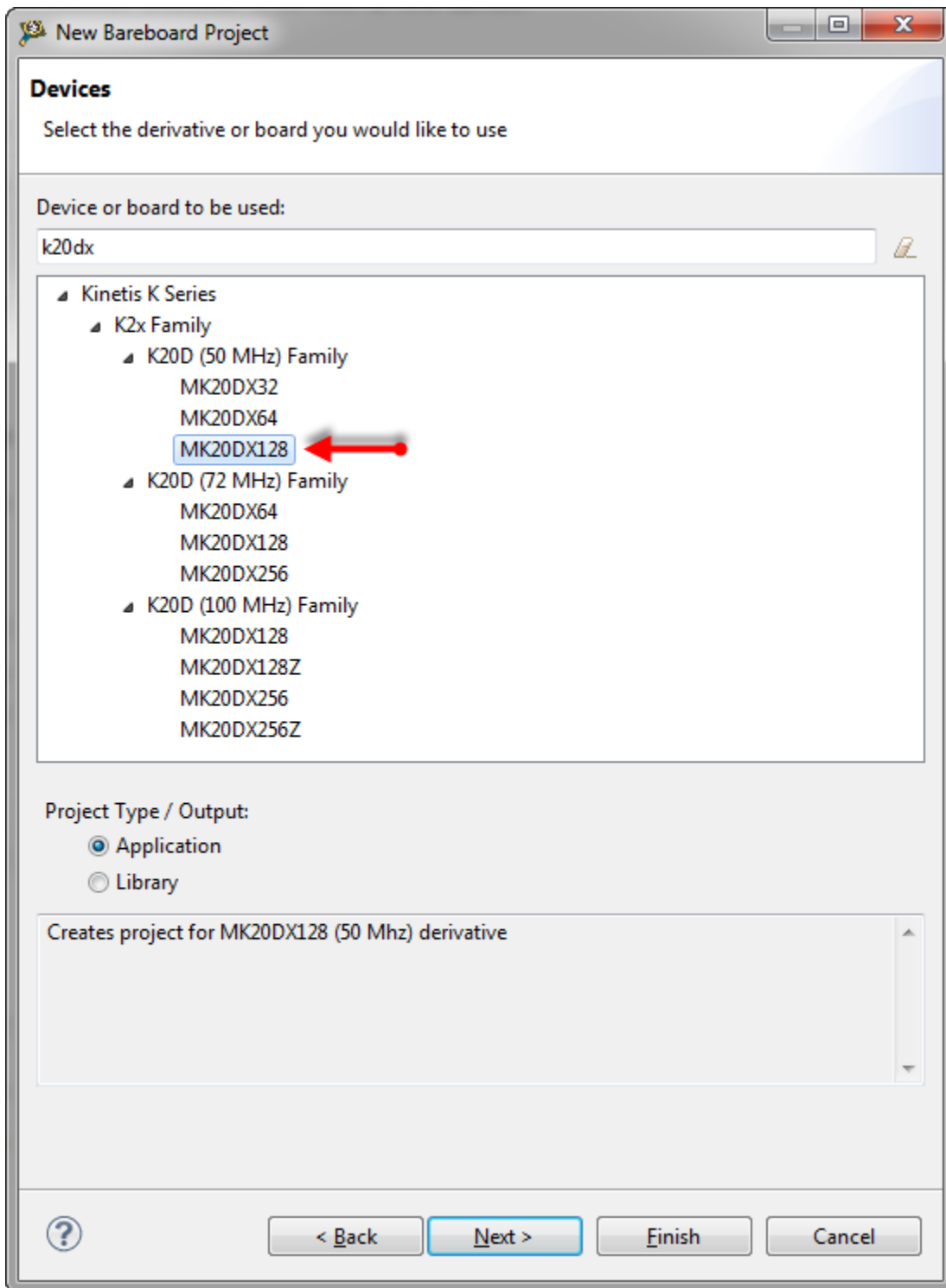


Figure 2-177. FRDM-K20D50M Quick Start

## 2.6.86 NEW Which device should I use to create a new project for my FRDM-K20D50M board using CodeWarrior?

The CodeWarrior for MCU10.3 supports the **MK20DX128VLH5** (mask 1N86B) board. To use this device, select **Kinetis K Series > K2x Family > K20D (50 Mhz) Family > MK20DX128** from the **Devices** page of the **New Bareboard Project** wizard, as the following figure shows:



**Figure 2-178. New Bareboard Project Wizard - Devices Page**



## Chapter 3

# Project

In this chapter, you find Frequently Asked Questions (FAQs) related to projects in the CodeWarrior IDE.

- [FAQs - Project](#)

### 3.1 FAQs - Project

In this topic:

- [How do I add a new Source Folder to my project?](#)
- [How can I add to my project a folder that exists outside the current workspace?](#)
- [How do I add to my project a new file that exists outside the current workspace?](#)
- [How can I export a project that I could send to someone else?](#)
- [How do I instruct the CodeWarrior IDE to save changes to my source code files automatically before building the project?](#)
- [How do I compare two source code files?](#)
- [How do I compare two directories?](#)
- [How can I specify the number of days to keep local history for source code files?](#)
- [How can I see or change the preprocessor macros defined for my project?](#)
- [Is there a shortcut to remove object code that is equivalent to the Ctrl + - shortcut key available in the Classic CodeWarrior IDE?](#)
- [How can I view beans in a Processor Expert project?](#)
- [How to use a configured bean from one project to the other?](#)
- [Where can I find CodeWarrior for Microcontrollers V10.x example projects?](#)
- [While launching a project, the CodeWarrior software automatically builds the current active target. How can I specify the build configuration I want to debug?](#)
- [While building a project, the CodeWarrior fails to write an ELF file. How do I resolve this issue?](#)
- [I am unable to create the new project at the desired location. What could be the possible reason and how to resolve this?](#)
- [I want to build a library \(.lib\) instead of a final elf \(.abs\) file. How can I do this?](#)

- How do I filter the items listed in the Problems view based on the project selected?
- Is there a way I can view the hierarchical structure for include files in my project?
- **NEW** How do I add/remove floating point format for S08 Projects?
- **NEW** How do I restore deleted files in Eclipse with local history?

### 3.1.1 How do I add a new Source Folder to my project?

To add a new source folder to a project follow steps below:

1. In the CodeWarrior Project window, right-click on project name.

A context menu appears.

2. From context menu, select **New > SourceFolder** .

The **New Source Folder** window appears.

3. In the folder name textbox, enter the name for new folder.
4. Click **Finish** .

The new source folder appears in window.

### 3.1.2 How can I add to my project a folder that exists outside the current workspace?

To add files to an existing project, follow steps below:

1. In the CodeWarrior Project window, right-click on project name.

A context menu appears.

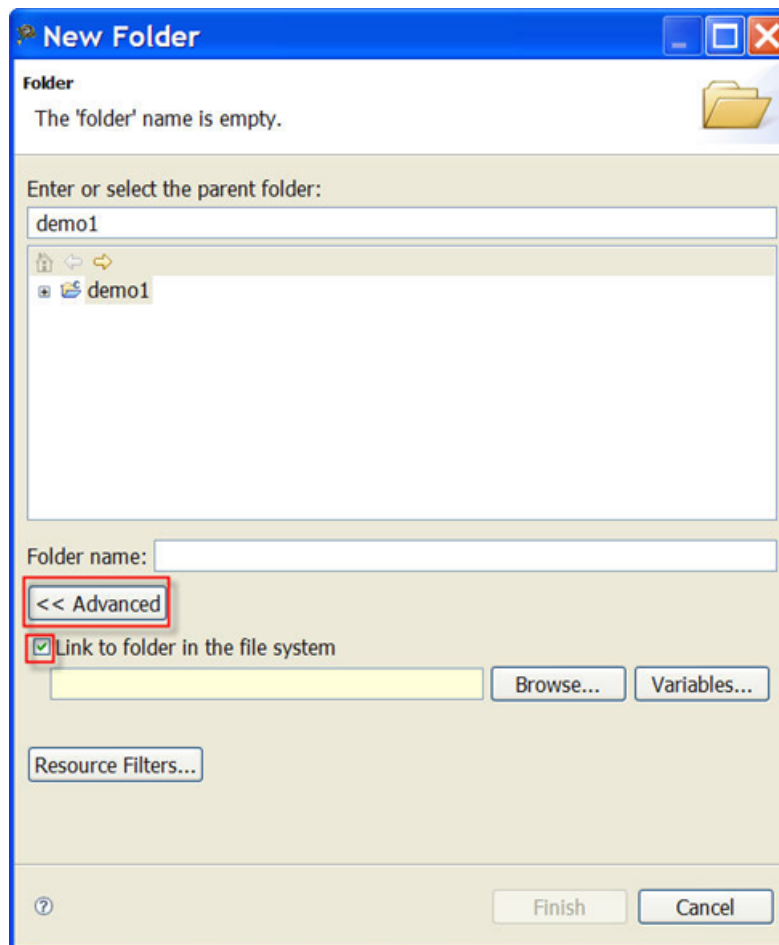
2. From context menu, select **New > Folder** .

The **New Folder** window appears.

3. Click **Advanced** .

The Advanced Settings appear.

4. Select **Link to folder in the file system** checkbox.



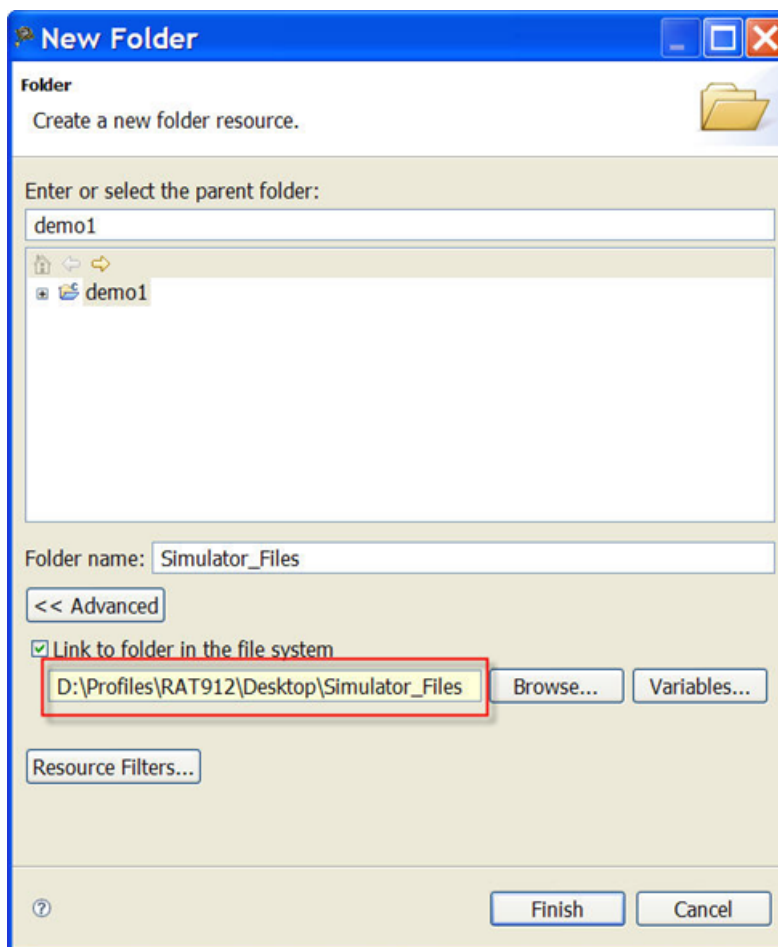
**Figure 3-1. New Folder**

5. Click **Browse** .

The **Browse for Folder** window appears.

6. Select the folder that you want to add to the project.

7. Click **OK** .



**Figure 3-2. New Folder**

8. Click **Finish** .

This will put the new folder into the project under CodeWarrior projects. When the new folder is expanded, it will point to the location where the folder is linked from. If you delete the whole folder from the project, it will be deleted from the project only. If you try to delete a file from this folder, it will be deleted from that location leaving no way to recover it.

### 3.1.3 How do I add to my project a new file that exists outside the current workspace?

To add a specific file to an existing project, drag the file that you want to add from the Windows Explorer into the CodeWarrior Projects.

### 3.1.4 How can I export a project that I could send to someone else?

To export a project follow steps below:

1. In CodeWarrior Project window, right-click on project name.

A context menu appears.

2. From context menu, select **Export** .

The **Export** window appears.

3. Expand **General Folder** .
4. Select **Archive File** .
5. Click **Next** .

The **Archive File** window appears.

#### NOTE

By default, the project that you select will be exported along with their children. Optionally, use the checkboxes in the left and right panes to select the set of resources to export. Also, use radio buttons, such as **Select Types** to filter the types of files that you want to export.

6. In the **To Archive File** textbox, enter the archive file which you want to export the selected resources to.
7. Click **Finish** .

A zip file is created in the **Eclipse** folder within the **Tools Installation** directory.

### 3.1.5 How do I instruct the CodeWarrior IDE to save changes to my source code files automatically before building the project?

To instruct the CodeWarrior IDE to save changes to your source code files automatically before building the project, follow steps below:

1. From the C/C++ perspective toolbar, select **Window > Preferences** .

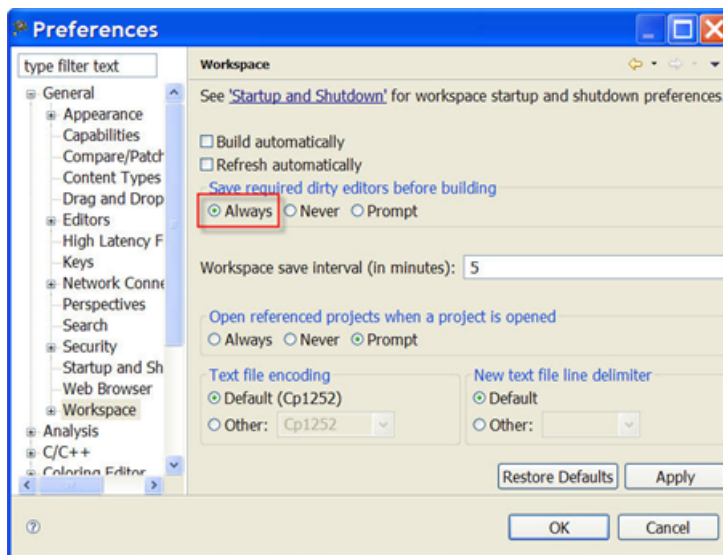
The **Preferences** window appears.

2. Expand **General** .

3. Select **Workspace** .

The **Workspace** settings appear.

4. Under **Save required dirty editors before building** , select the **Always** radio button.



**Figure 3-3. Preferences-Workspace Page**

5. Click **Apply** .
6. Click **OK** .

Now, the CodeWarrior IDE will save changes to the source code files automatically before building the project.

### 3.1.6 How do I compare two source code files?

In order to compare two source files, follow steps below:

1. In the CodeWarrior Projects folder, click on source file.  
The source file is highlighted.
2. Press the **Ctrl** key.
3. In the CodeWarrior Projects folder, click on second source file.  
The source file is highlighted.
4. Right-click on source file.  
A context menu appears.
5. From context menu, select **Compare With > Each Other** .

The **Compare** window appears.

You can compare the source files using the **Compare** window.

### 3.1.7 How do I compare two directories?

In order to compare two directories follow steps below:

1. In the CodeWarrior Projects folder, click on directory.

The directory folder is highlighted.

2. Press the **Ctrl** key.
3. In the CodeWarrior Projects folder, click on second directory folder.

The directory folder is highlighted.

4. Right-click on directory folder.

A context menu appears.

5. From context menu, select **Compare With > Each Other** .

The **Compare** window appears.

You can compare the directories files using the **Compare** window.

### 3.1.8 How can I specify the number of days to keep local history for source code files?

To change the history length for the C source editor, follow steps below:

1. From the C/C++ perspective toolbar, select **Window > Preferences** .

The **Preferences** window appears.

2. Select **General > Workspace > Local History** .
3. The **Local History** settings appear.
4. In the **Days to Keep Files** textbox, change the days to keep files value.
5. Click **Apply** .
6. Click **OK** .

The history length for the C source editor is updated.

### 3.1.9 How can I see or change the preprocessor macros defined for my project?

To see or change the preprocessor macros defined for your project, follow steps below:

1. In the CodeWarrior Project window, right-click on project name.

A context menu appears.

2. From the context menu, select **Properties** .

The **Properties** window appears.

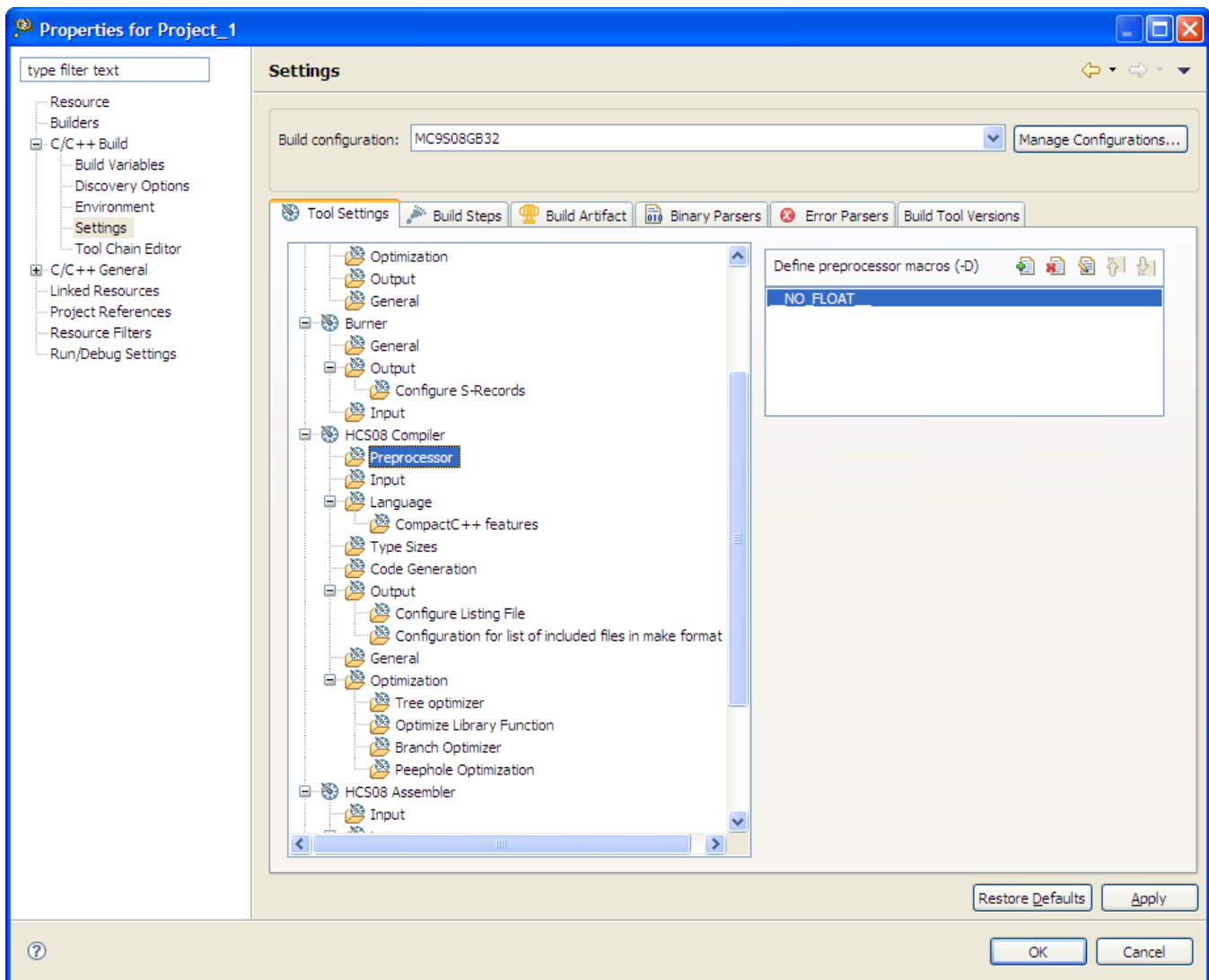
3. Select **C/C++ Build > Settings** .

The **Settings** panel appears on the right-hand side.

4. Under *target compiler*, select **Preprocessor** .

The **Macros** panel appears on the right-hand side displaying `#define` preprocessor macros.





**Figure 3-4. Properties Dialog Box-Settings Page**

5. Make the required changes and click **Apply** .
6. Click **OK** .

The preprocessor macros defined for your project changes as specified.

### 3.1.10 Is there a shortcut to remove object code that is equivalent to the **Ctrl + -** shortcut key available in the Classic CodeWarrior IDE?

No. However, if you want to assign the **Ctrl+-** shortcut key for removing the object code, then you can manually assign it by changing the key binding. For more information on how to assign or change a key binding, see [How can I change a key binding?](#)

When you change the key binding, a dialog box appears asking you which project you want to clean, and if you want to automatically start the build thereafter. If you do not want this dialog box to appear, then you can select the **Build Automatically** checkbox under **Window > Preferences > General > Workspace Settings** .

### 3.1.11 How can I view beans in a Processor Expert project?

To view beans, you need to open the **Project Panel** view.

1. Select **Window > Show View > Other**.

The **Show View** dialog box appears.

2. Type "Project Panel" as the filter text to narrow down the list of the views in the dialog box.
3. Select **Processor Expert > Project Panel** and click **OK**.

The **Project Panel** view opens in the Workbench window.

You can view the beans in the **Project Panel** view.

#### NOTE

For more information about the **Project Panel** view, refer to the *Processor Expert Users Manual*.

### 3.1.12 How to use a configured bean from one project to the other?

To use a configured bean from one project to the another, simply drag & drop the component from one project to another.

### 3.1.13 Where can I find CodeWarrior for Microcontrollers V10.x example projects?

The example projects are available at the following location:

```
<install_dir>\MCU\CodeWarrior_Examples
```

, where *<install\_dir>* refers to the CodeWarrior installation directory.

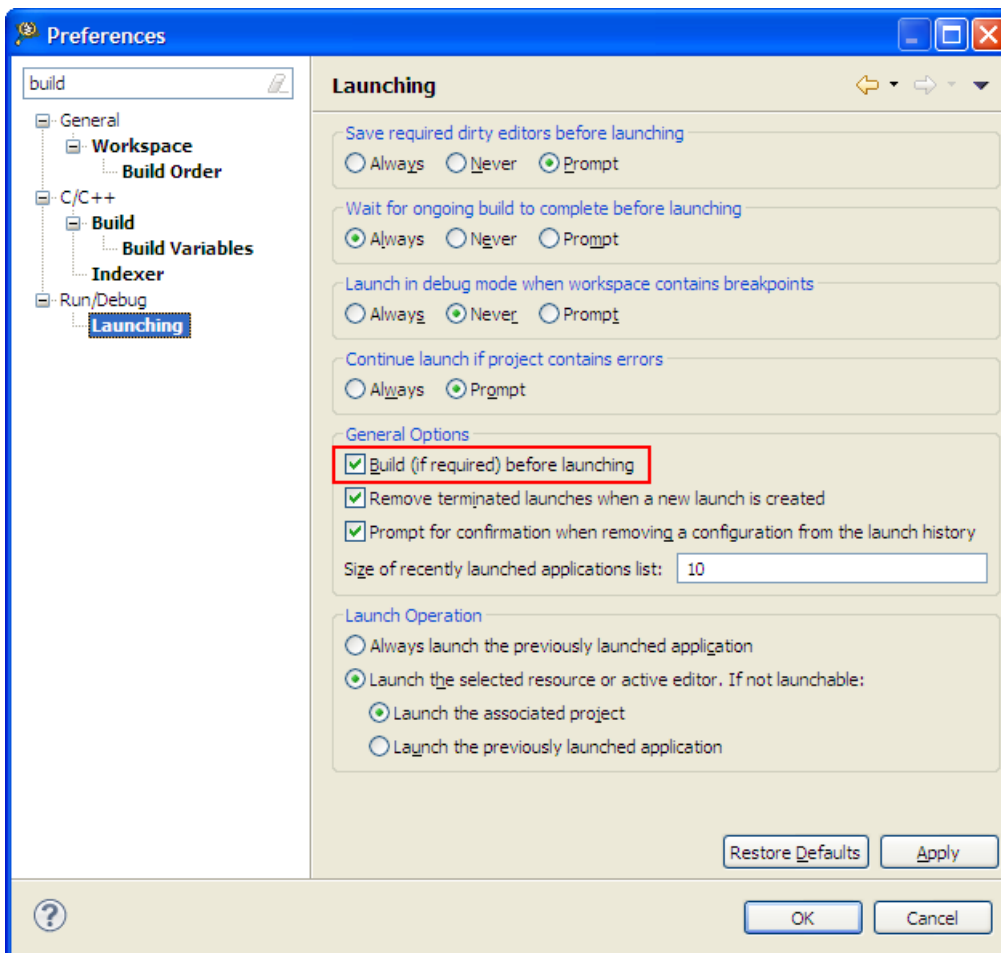
Additional Example Projects are available on the Web. They can be found in the **Self-Paced Training** section on the **Training & Support** tab of the CW MCU Web page.

[http://www.freescale.com/webapp/sps/site\\_prod\\_summary.jsp?code=CW-MCU10&tab=Design\\_Support\\_Tab](http://www.freescale.com/webapp/sps/site_prod_summary.jsp?code=CW-MCU10&tab=Design_Support_Tab)

- Processor Expert Example Project for TWR-K60N512: Includes a CW MCU v10.1 project using Processor Expert to create a bare board (no RTOS) application to use GPIO and Timers to blink LEDs on the TWR-K60N512 board. It also includes the step by step instructions to recreate the project.
- Example Projects for TWR-K40X256-KIT and TWR-K60N512-KIT: Includes CW MCU v10.1 bare board (no RTOS) projects for TWR-K40X256-KIT and TWR-K60N512KIT. The projects for each kit include: adc\_demo, can\_loopback\_node, crc\_demo, dac12bit\_demo, flexbus, flexmem, gpio, hello\_world, hscmp, i2c, lptmr, mcg\_demo, pdb\_adc\_demo, pmc, rtc, sci2can, slcd, tsi, USB\_DCD, USB\_device, USB\_dual\_role, USB\_HOST, USB\_MAX3353.

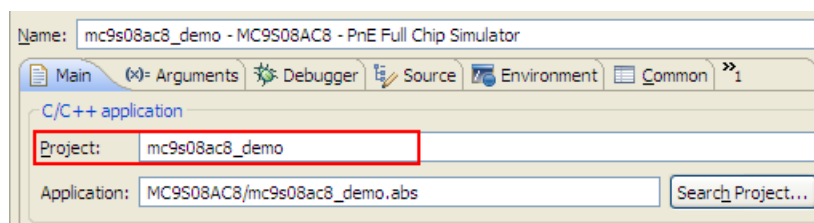
### 3.1.14 While launching a project, the CodeWarrior software automatically builds the current active target. How can I specify the build configuration I want to debug?

The CodeWarrior software has the following default settings.



**Figure 3-5. Build Before Launching**

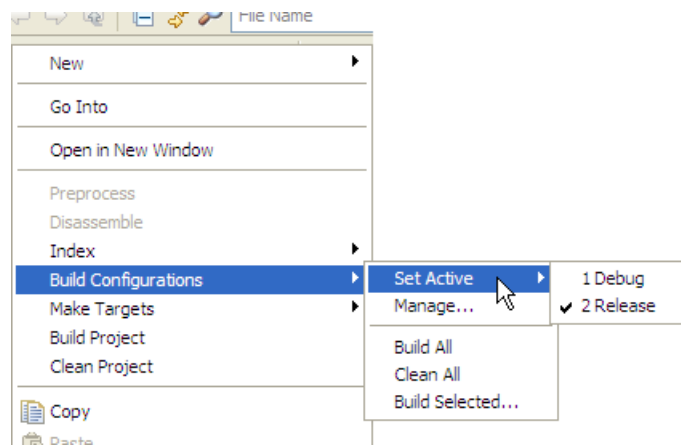
And, the launch configuration refers the project and not the build target of the project.



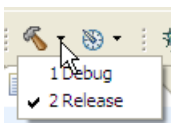
**Figure 3-6. Launch Configuration - Refers Project**

This is why the CodeWarrior software automatically build the current active build target. To ensure that the CodeWarrior software builds the right target in CodeWarrior for Microcontroller V10.1:

- Make sure you have the right build set as active, or

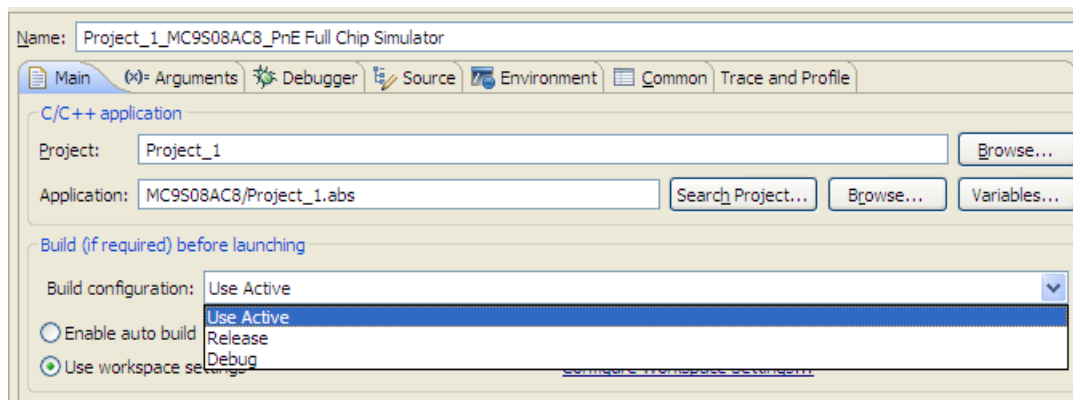


**Figure 3-7. Set Active Build Configuration**



- Or, clear the **Build (if required) before launching** checkbox, and build the project yourself before launching.

However, in Microcontrollers V10.2, you can specify the build target while launching the project.



**Figure 3-8. Specify Build Target - In Microcontrollers V10.2**

### 3.1.15 While building a project, the CodeWarrior fails to write an ELF file. How do I resolve this issue?

The `DE.EXE` might locks the file. The `DE.EXE` (Debugger Engine Executable) locks the file during debugging, but might have failed to release it. To release the lock, simply close (kill) the `DE.EXE` application from the task manager.

### 3.1.16 I am unable to create the new project at the desired location. What could be the possible reason and how to resolve this?

To create a project at the desired location, you have to specify a folder that does not exist yet (even if you have created specially from the **Browse For Folder** dialog box using, **Make New Folder** option). For example,

If the desired location for your project is,

```
C:\...\FSLMQX3.7\demo
```

then in the **Location** text box, append the non-existing yet project folder name, for example, " PE\_TEST\_K60", therefore, the final location should appear as:

```
C:\...\FSLMQX3.7\demo\PE_TEST_K60
```

Another reason could be that you are creating a project with a folder name which already exists in the workspace. Check your workspace folder if the same name already exists and choose a different project name.

### 3.1.17 I want to build a library (.lib) instead of a final elf (.abs) file. How can I do this?

To build a library project, follow the steps listed below:

1. Select the desired project in the **CodeWarrior Projects** view.

The **Properties for <Project\_name>** dialog box appears.

2. Select **Tool Chain Editor** from the left panel.

The **Tool Chain Editor** options page appears in the right panel.

3. Click the **Select Tools** button.

The **Select tools** dialog box appears.

4. Replace **Linker** with **Libmaker** in **Select tools** dialog box.

#### NOTE

For CodeWarrior for Microcontrollers v10.2, replace **S08 Linker** with **S08Libmaker** in the **Select tools** dialog box.

5. Click **OK**.

The **Tools settings** dialog box closes.

6. Select **Settings** from the left panel.

The **Settings** options page appears in the right panel.

7. Click the **Build Artifact** tab.

8. Change the **Artifact extension** to the desired extension (e.g. 'lib').

9. Click **Apply** to save the settings.

10. Click **OK**.

The **Properties for <Project\_name>** dialog box closes.

11. Right-click the project in the **CodeWarrior Projects** view and select **Clean Project** from the context-menu.

This removes the previous build artifact.

12. Remove the `start08.c` file from the project.

13. Change the libmaker command as listed below:

- **Change from:** `${COMMAND} ${FLAGS} ${OUTPUT_FLAG} -cmd( ${INPUTS} = ${OUTPUT})`
- **To:** `${COMMAND} ${FLAGS} ${OUTPUT_FLAG} -cmd( ${INPUTS} = myLib.lib)`

An alternate and easier way to build a library project is, while creating a new project using the **New BareBoard Project** wizard, select **Library** in the **Project Type/Output** option in the **Devices** page.

### 3.1.18 How do I filter the items listed in the Problems view based on the project selected?

To filter the items listed in the **Problems** view based on the project selected:

1. Select the inverted triangle from the the top left corner of the **Problems** view.

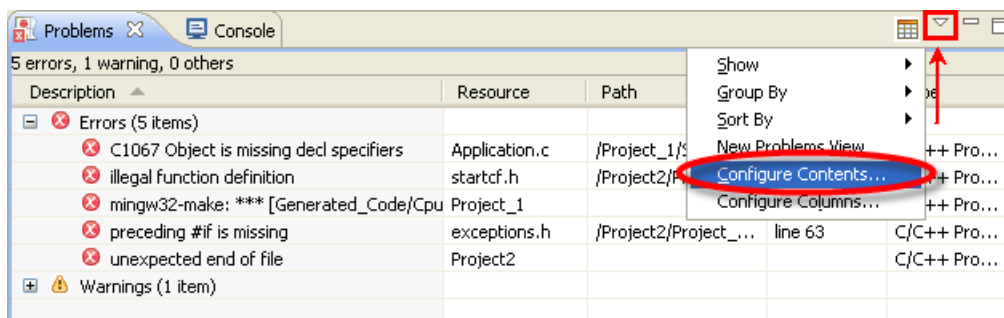
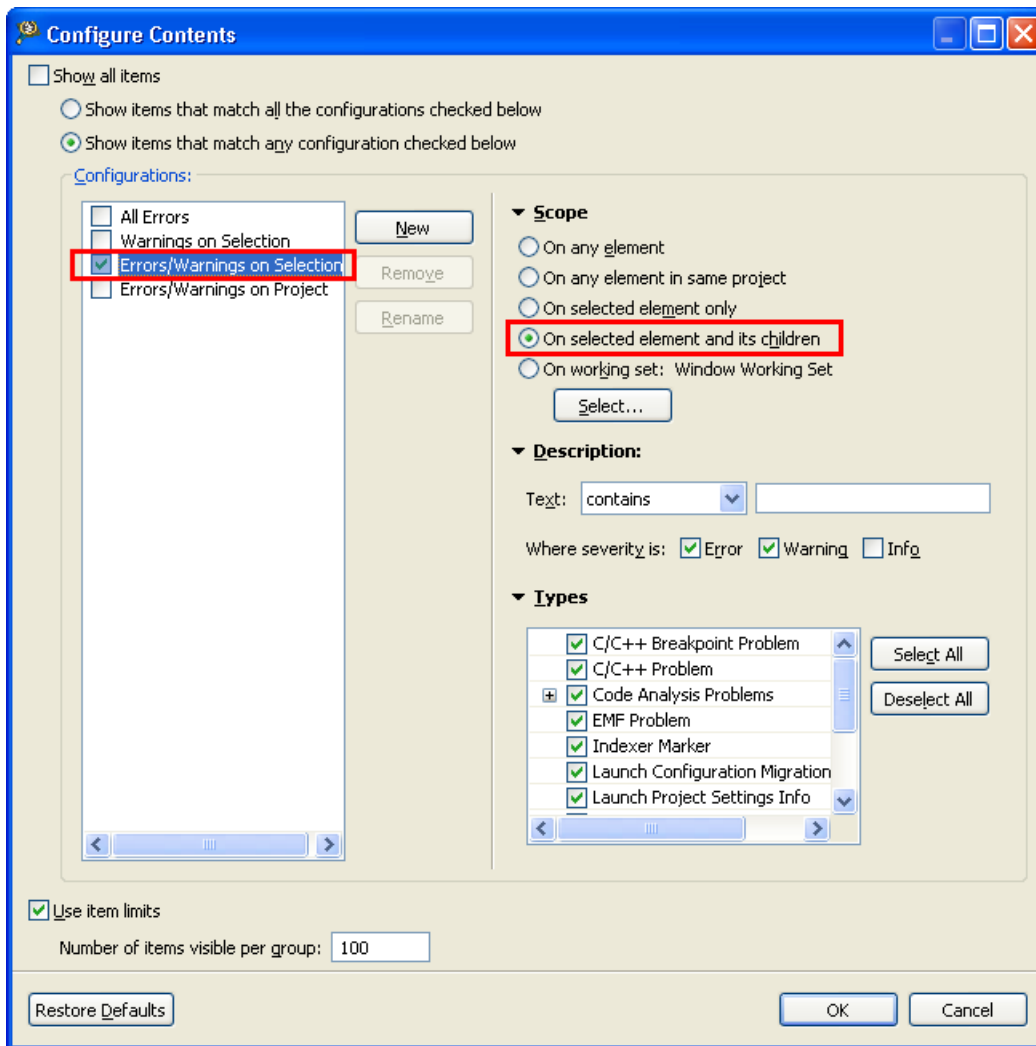


Figure 3-9. Problems View

2. Select **Configure Contents** from the menu.

The **Configure Contents** dialog box appears.



**Figure 3-10. Configure Contents Dialog Box**

3. Check the **Errors/Warnings on Selection** checkbox from the **Configurations** list in the left panel.
4. Make sure that the **On selected element and its children** option is selected from the **Scope** list in the right panel.
5. Click **OK** .

The **Problems** view lists the problems related to the selected project only.

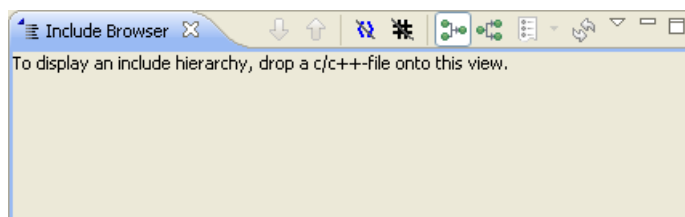


### 3.1.19 Is there a way I can view the hierarchical structure for include files in my project?

Yes, you can view include files hierarchy structure for your project by using the **Include Browser** view. To use the **Include Browser** view:

1. Select **Window > Show View > Include Browser** from the IDE menu bar.

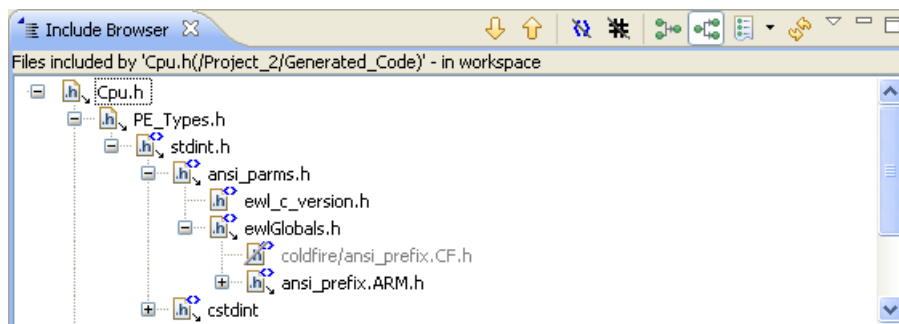
The **Include Browser** view appears in the view stack at the bottom of the CodeWarrior IDE window.



**Figure 3-11. Include Browser View**

2. Just drag & drop the desired source or header file from the **CodeWarrior Projects** view to the **Include Browser** view.

The selected file appears in the **Include Browser** view listing all the included files in their chain of command order.



**Figure 3-12. Include Browser View - With Files Hierarchy**

By using this view you can:

- see what is included directly and indirectly by your files, what is included as the system include (marked with  $\langle \rangle$ ), and also the inactive includes (appears grayed out).
- double click on a file to open the exact position of the include file.
- view includers by using the **Show Includers** button from the **Include Browser** view toolbar.

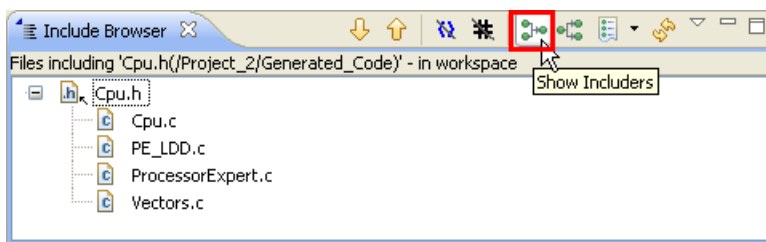


Figure 3-13. Include Browser View - Show Includers Toolbar Button

### 3.1.20 NEW How do I add/remove floating point format for S08 Projects?

You can add/remove the floating point support in your existing S08 project, by using any of the following methods:

- By adding/removing `__NO_FLOAT__` preprocessor macro set, in Build Properties dialog box, as the following figure shows:

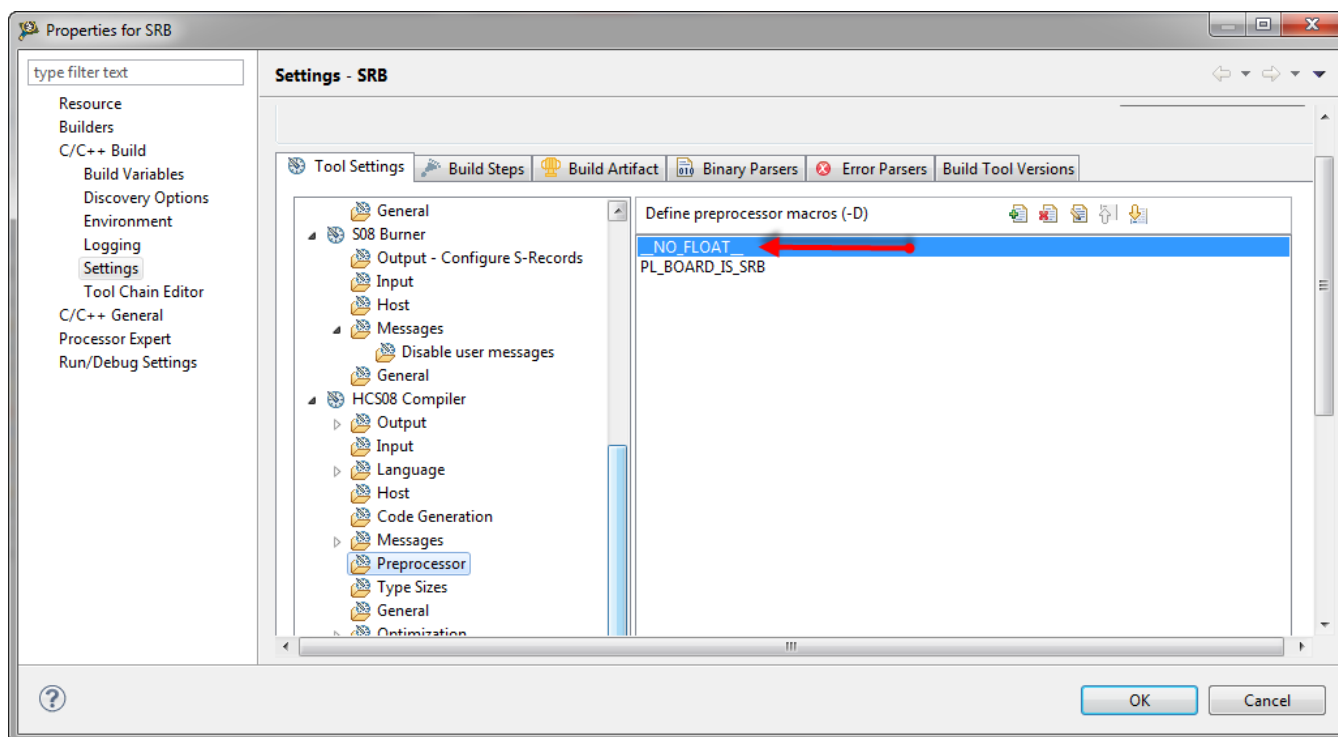
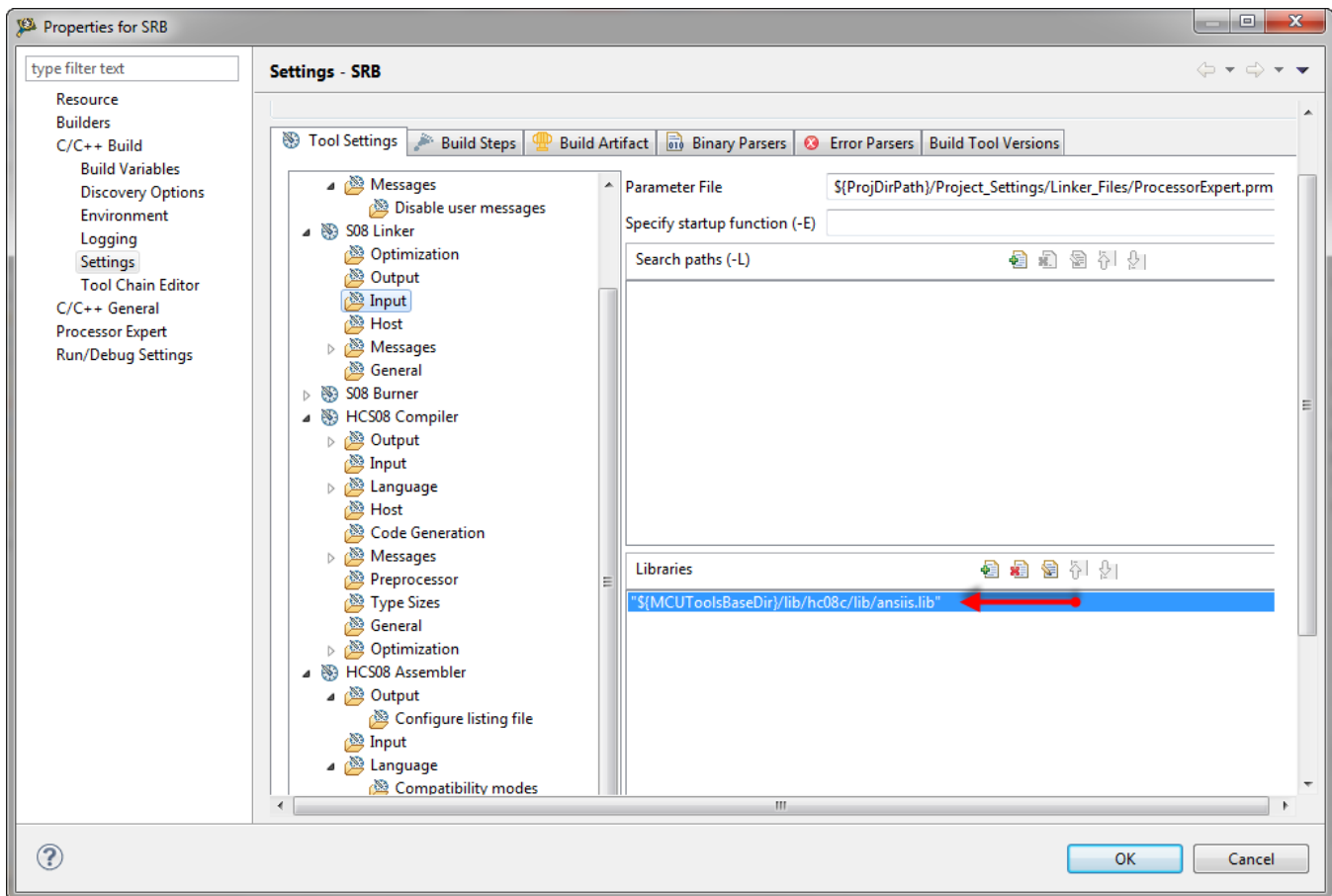


Figure 3-14. S08 Build Properties — `__NO_FLOAT__` Macro

- By changing the **Libraries** settings for floating point support in the **S08 Linker > Input** options of the project, as the following figure shows:



**Figure 3-15. S08 Build Properties Panel — Linker Libraries Setting**

- By selecting **Use IEEE32 for double(double is IEEE64)** option in the **S08 Compiler & Code Generation** settings. The compiler/library supports a model where doubles have only 32bits. So if you decide to use such a format, you need to make sure that the correct library is linked. Additionally, you need to check that the compiler option for the floating point format is set correctly.

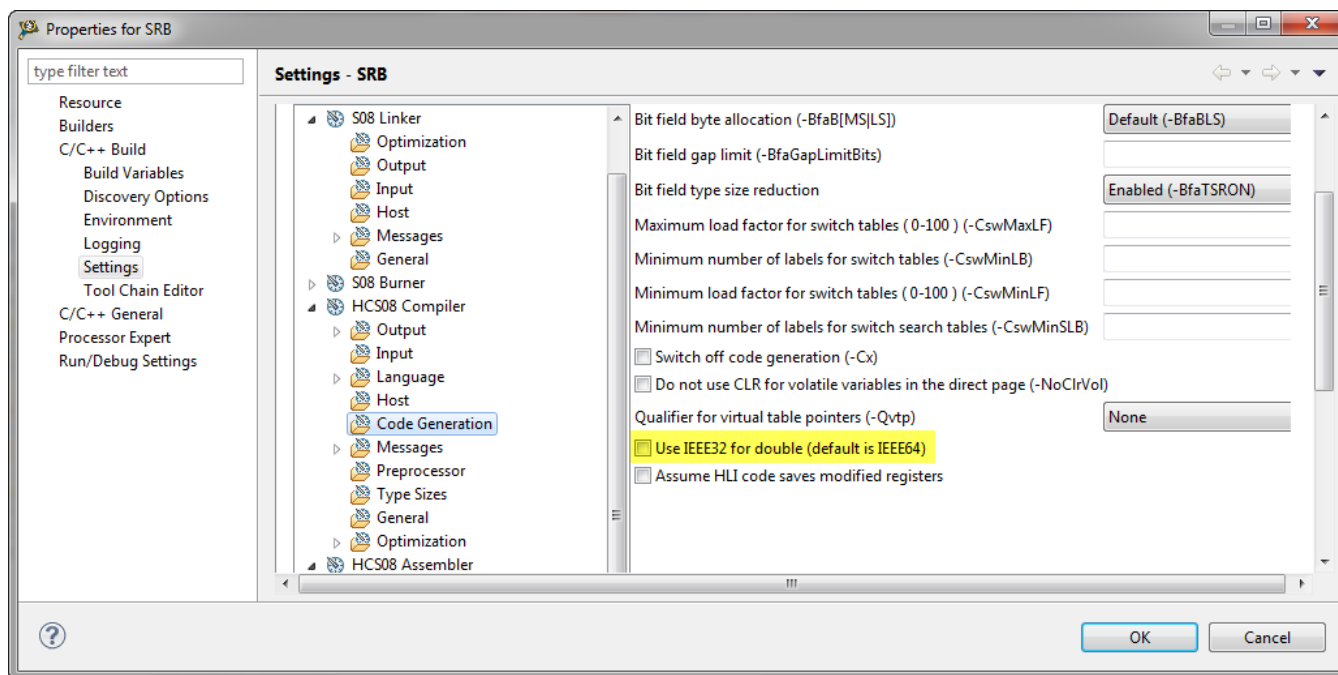
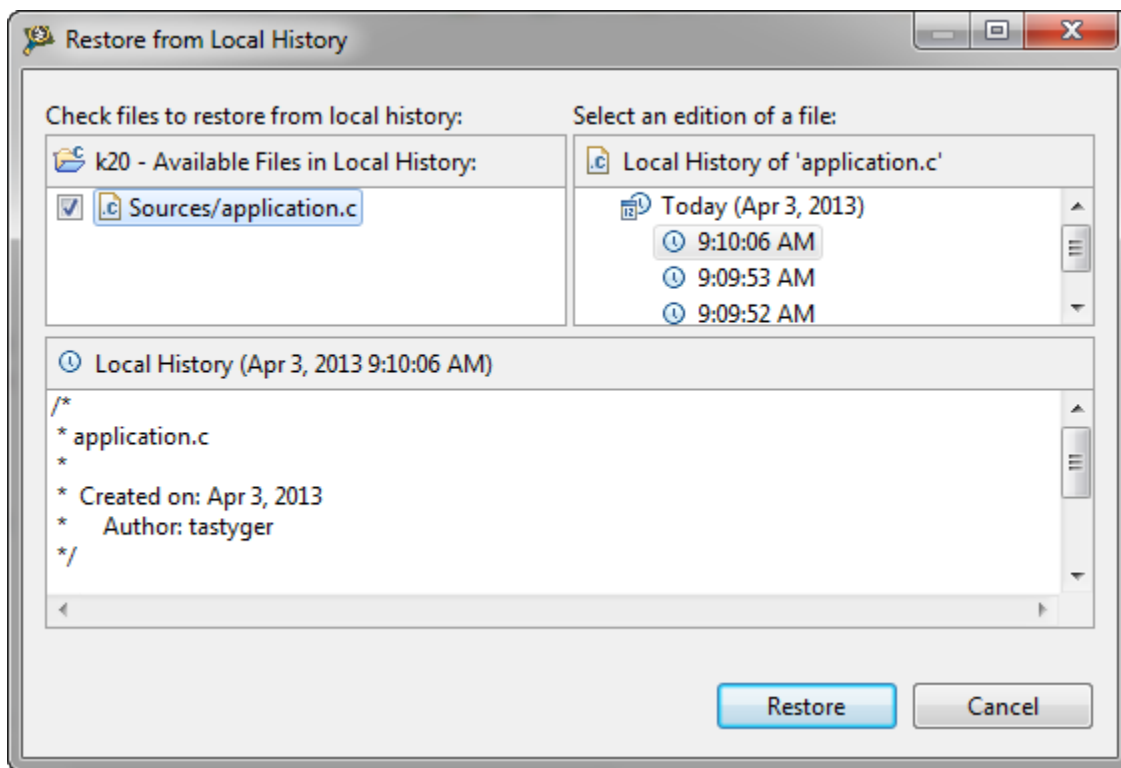


Figure 3-16. Floating Point Compiler Option

### 3.1.21 NEW How do I restore deleted files in Eclipse with local history?

To restore a deleted file, right-click on your project folder and select **Restore from Local History** from the context-menu.

This opens a dialog box where you can select what exactly I want to restore:



**Figure 3-17. Restore from Local History Dialog Box**

#### **NOTE**

The Local History is not only useful to restore files, it can also restore any changes or edits in a resource.

#### **NOTE**

The local history is stored based on the project, and the *Restore* operation needs to be performed on the project folder. So if you delete your project by using the **Delete project contents on the disk (cannot be undone)** option. In such case your project is permanently deleted, and you cannot recover it using the **Restore from Local History** option.



## Chapter 4

# Project Management

In this chapter, you find Frequently Asked Questions (FAQs) related to project management in CodeWarrior IDE.

- [FAQs - Project Management](#)

### 4.1 FAQs - Project Management

In this topic:

- [How can I find out where the source files related to a CodeWarrior project are saved?](#)
- [How can I add another source or header file to my project?](#)
- [What is a workspace?](#)
- [What is a project?](#)
- [How can I view the contents of various types of source code files that appear in my project folder?](#)
- [Is there a way to compare the Property Settings of two different projects?](#)
- [How do I resize the various views in a Perspective?](#)
- [How can I add a project from some other workspace to my current workspace?](#)
- [How do I ensure that the CodeWarrior IDE saves all modifications that I make to the project properties?](#)
- [While working with the CodeWarrior IDE, the Remote System Changed dialog box appears. Why does it appear and how can I resolve the issue?](#)
- [Which set of files I need to backup or store in a version control system to completely recover project settings and related debug settings?](#)
- [I need to share a project without using a version control system. Is there any way to do this?](#)
- [While importing a project, is there any way to find out which file mapping are missing?](#)

### 4.1.1 How can I find out where the source files related to a CodeWarrior project are saved?

To know where your source file is located, follow these steps:

1. In the C/C++ project window, right-click on source file.  
A context menu appears.
2. In the context menu, select **Show in Windows Explorer** .  
The Window Explorer window appears.

Alternatively, follow these steps:

1. In the C/C++ project window, right-click on source file.  
A context menu appears.
2. In the context menu, select **Properties** .  
The **Properties** window appears.
3. Select **Resource** .

The **Location** field displays the full path of the folder.

#### NOTE

CodeWarrior always warns the user before performing a delete operation and describes whether it is about to delete just the link, or to delete the file or folder permanently.

### 4.1.2 How can I add another source or header file to my project?

To add a source file or header to project follow steps below:

1. In the CodeWarrior Project window, right-click on the folder where you want the new source file or header to be added.  
A context menu appears.
2. From the context menu, select **New > Source File** .



The **New Source File** window appears.

3. In the **Source File** textbox, enter the name of the source file.
4. Click **Finish** .

The source file gets added to your project.

#### **NOTE**

Another way to accomplish is to drag and drop the file from Windows Explorer. A copy of the file is added to the project's workspace. If you make changes to the source file, the original source file is not modified. Only source file in the project is modified.

### **4.1.3 What is a workspace?**

A workspace is a directory for your work. This is where your project is located. The name and location of the workspace is specified when you start the CodeWarrior IDE.

### **4.1.4 What is a project?**

A project is a container for organizing files and folders related to a specific area.

### **4.1.5 How can I view the contents of various types of source code files that appear in my project folder?**

To view the contents of source files, follow steps below:

1. From the CodeWarrior Projects window, right-click on the source file.

A context menu appears.

2. From the context menu, select **Open With > Text Editor** .

The text editor window appears showing the contents of the file.

### Tip

Double-click on the source file to view its contents in the **C/C++ Editor**.

#### 4.1.6 Is there a way to compare the Property Settings of two different projects?

Currently the way to do this is to do a text compare of the .project files. To do this, follow steps below:

1. In the CodeWarrior Projects folder, click on .cproject.

The .cproject is highlighted.

2. Press the **Ctrl** key.

3. In the CodeWarrior Projects folder, click on second .cproject.

The second .cproject is highlighted.

4. Right-click on .cproject.

A context menu appears.

5. From context menu, select **Compare With > Each Other**.

The **Compare** window appears.

You can compare property settings of the projects using the **Compare** window.

#### 4.1.7 How do I resize the various views in a Perspective?

To resize the various view in a Perspective, click on the dividers (gray bands) between the different views and drag them with the mouse.

#### 4.1.8 How can I add a project from some other workspace to my current workspace?

In order to add a project from another workspace follow steps below:

1. Open the workspace in Windows Explorer.
2. Select the project folder and drag it over to the CodeWarrior IDE.

The IDE effectively handles the files and folders dropped to the workbench. A link to the existing project is created in the **CodeWarrior Project** view.

#### 4.1.9 How do I ensure that the CodeWarrior IDE saves all modifications that I make to the project properties?

The settings specified in the Properties dialog box are saved when you click **Apply** and then **OK** to close the dialog box. The only reason this does not work could be:

- You only have read access to the `.project` file in the project directory.

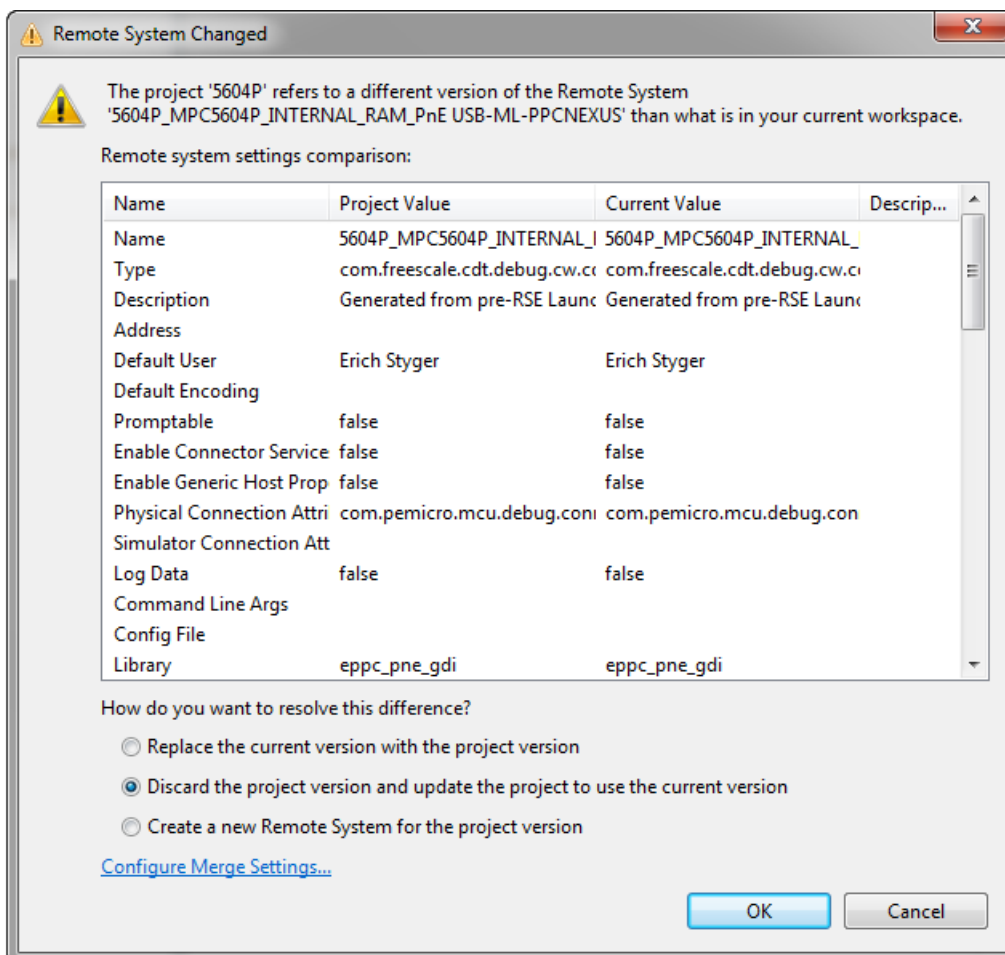
-or-

- You closed the **Properties** dialog box by clicking **Cancel** instead of **OK**.

#### 4.1.10 While working with the CodeWarrior IDE, the Remote System Changed dialog box appears. Why does it appear and how can I resolve the issue?

The projects that reference s Explorer (RSE) Systems contain an internal cache of the referenced RSE Systems. When such projects are imported in the workspace, the RSE settings in the cache may differ from the actual RSE settings in the workspace. Or the RSE settings may be missing entirely from the project.

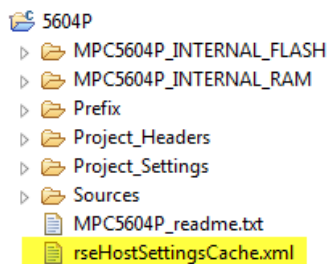
In such situations, the **Changed** dialog box appears.



**Figure 4-1. Changed Dialog Box**

The RSE settings are stored in the `.metadata` workspace folder as a part of the project settings and in the `rseHostSettingsCache.xml` file (refer to the image listed below), as the cached version, in the project.

As with any cache, the cache might be out of sync or needs to be flushed, which causes a mismatch between the RSE settings in the cache and the actual RSE setting in the project.

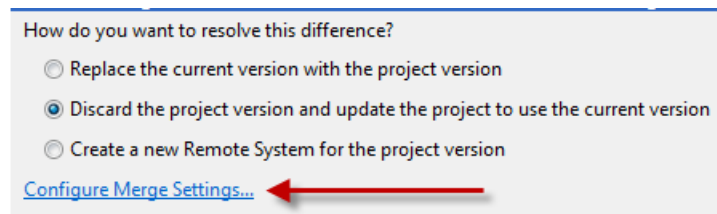


**Figure 4-2. RSE Settings**

To resolve the issue, you need to configure the merge settings.

To configure IDE's response when an RSE system mismatch is detected, perform the following steps:

1. Click **Configure Merge Settings** in the **s Changed** dialog box.



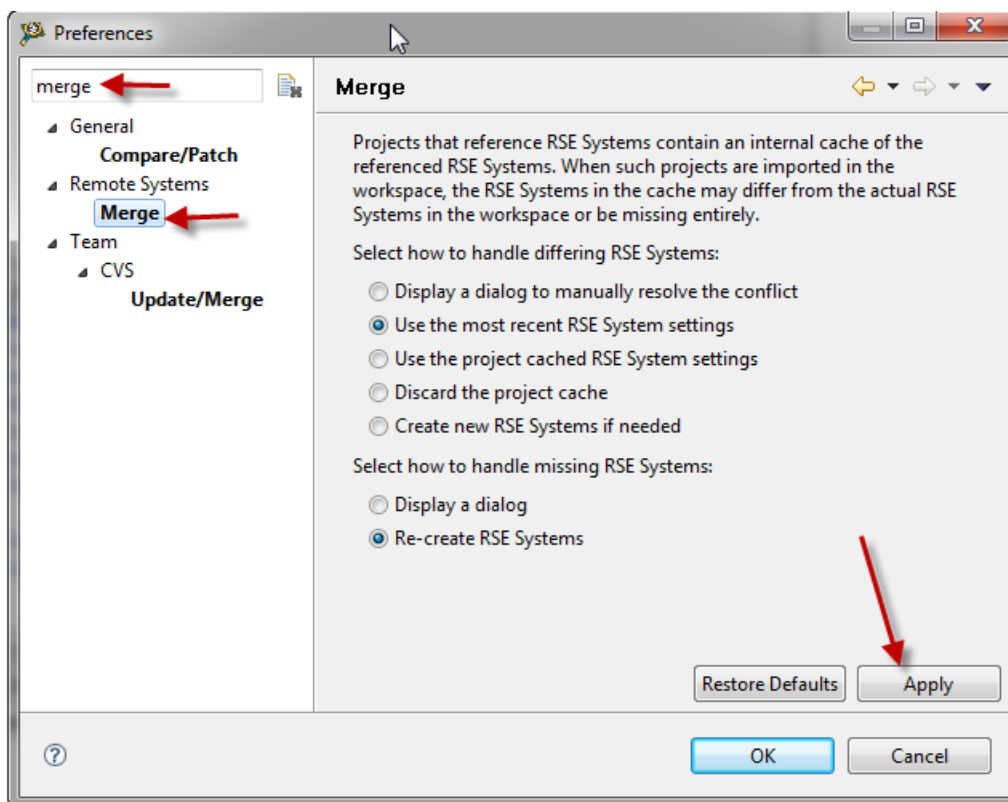
**Figure 4-3. Configure Merge Settings**

Alternatively, select **Window > Preferences** from the CodeWarrior menu bar. In the **Preferences** dialog box, select **s > Merge**.

The **Merge** page appears in the **Preferences** dialog box.

2. Select the appropriate option and click **Apply**.

The merge settings are configured.



**Figure 4-4. Merge Settings | Preferences Dialog Box**

### 4.1.11 Which set of files I need to backup or store in a version control system to completely recover project settings and related debug settings?

The files you need to store in a version control system:

- `.project` and `.cproject` - contains the list of files and the build tool settings.
- `.ttf` - targets the needing target tasks for flash programming (for ColdFire V2).
- `.launch` - required for the debugger launch configuration.
- Debugger configuration files such as `*.tcl` or `*.mem`.
- `rseHostSettingsCache.xml` - contains the cached RSE settings. You need this to have the RSE settings imported.
- `*.c/*.h/etc` source files and linker file, if not using Processor Expert.
- If you are using Processor Expert - `ProcessorExpert.pe` (contains all the settings), `events.c/h`, and the `ProcessorExpert.c` with user modifications.

Any generated folders like `Generated_Code`, `Documentation` or the folder where the object/make files are placed (usually named by the CPU name) can be stripped off to reduce file size.

You can also export/store the workspace settings, like editor/VCS settings/etc, by selecting, **File > Export > Preferences** from the IDE menu bar.

### 4.1.12 I need to share a project without using a version control system. Is there any way to do this?

The simple way of sharing a project without using a version control system, is to export the project. In classic CodeWarrior there was a feature called **Pack&Go** which has a similar counterpart in the eclipse based CodeWarrior IDE, the **Export** wizard.

Before exporting a project, make sure that all the source files of your project are stored. The simplest way to ensure this is to close the project, and then to re-open it again. This ensures that everything is saved, and this is especially recommended for the **Processor Expert** to force it to save the `.pe` file content.

To export a project:

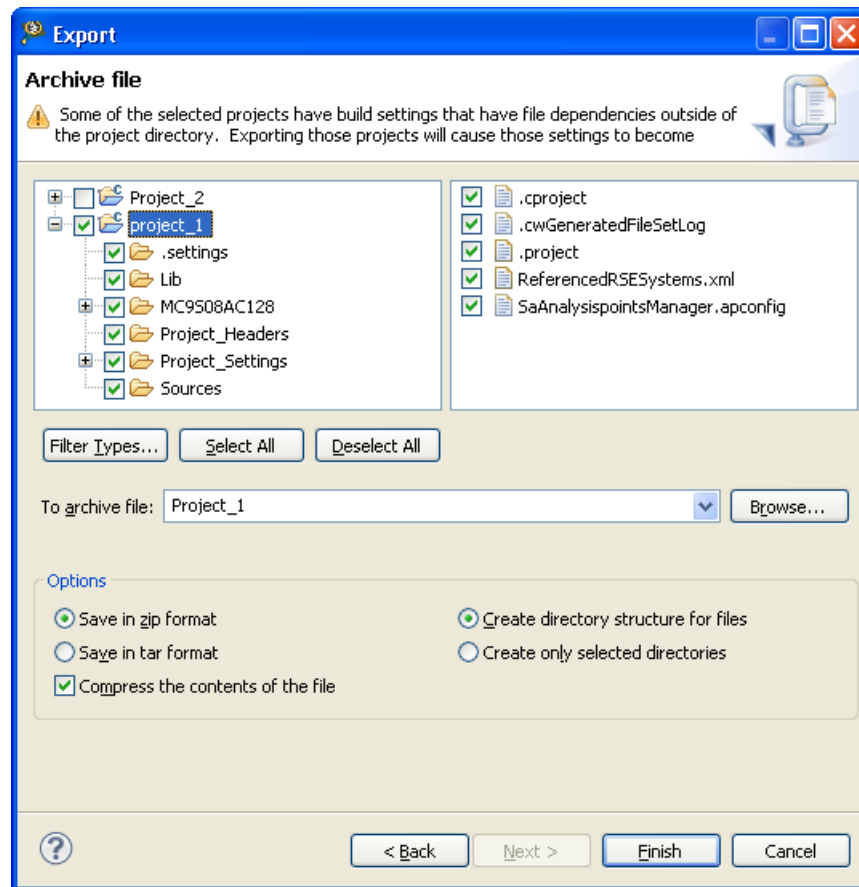
1. Select one or multiple projects and then select **File > Export** from the IDE menu bar.

The **Export** dialog box appears displaying the **Select** page.

2. Expand the **General** tree-control and select **Archive File**.

3. Click **Next**.

The **Archive File** page appears.



**Figure 4-5. Export Dialog Box - Archive File Page**

### NOTE

This dialog box (refer to the image listed above) provides you the options for which file to include (or not to include). You can selectively choose what you want to be exported, into which file and what format shall be used.

4. Click **Finish**.

The selected project(s) and file(s) are exported and compressed to the selected location.

Similarly you can import a saved/shared project. To import a project:

1. Select **File > Import** from the IDE menu bar.

The **Import** dialog box appears displaying the **Select** page.

2. Expand the **General** tree-control and select **Existing Projects into Workspace** from the list.

## NOTE

Now the good question might be, why not **Archive File** option? The answer is, importing the project with **Archive File** option is similar, it imports files into an existing project, but it cannot import a new project. So to import a project, you have to select **Existing Projects into Workspace**.

3. Click **Next**.

The **Import Projects** page appears.

4. Since you are importing an exported archived project into your workspace, select the **Select archive file** option and click **Browse** to select the desired file.
5. Click **Finish**.

The selected project is imported into your workspace.

### 4.1.13 While importing a project, is there any way to find out which file mapping are missing?

Follow the steps listed below:

1. Import an executable by selecting **File > Import** from the IDE menu bar.

The **Import** dialog box appears displaying the **Select** page.

2. Expand the **CodeWarrior** tree-control and select **MCU Executable Importer**.
3. Click **Next**.

The **Import a MCU executable file** page appears.

4. Specify the desired name for the imported project in the **Project name** text box.
5. Click **Next**.

The **Select MCU executable file to be imported** dialog box appears.

6. In the **File to import** option, click **Browse**, navigate to the desired project and select the executable (**.ELF/ .ABS/ .FLT/ .AFX**) file.

## NOTE

Check the **Copy to project folder** checkbox as it allows to set up the debugging paths easier (more later on).

7. Click **Next**.



The **Device and Board** page appears.

8. Select the desired CPU architecture.
9. Click **Next**.

The **Connections** page appears.

10. Select the appropriate connection(s) for your project.
11. Click **Finish**.

The selected project is imported into your workspace.

To set up the search paths (where to search the source files), follow the steps listed below:

1. For this open the launch configuration of your project. To open the launch configuration:
  - a. Select **Run > Debug Configurations** from the IDE menu bar.
  - b. Expand the **CodeWarrior Download** tree-control and select the desired launch configuration
2. Click the **Source** tab.
3. Click **Add**.

The **Add Source** dialog box appears.

4. Select **Path Mapping** and click **OK**.
5. Select the newly created mapping and click **Edit**, which displays the dialog where you can create the mapping.

The **Path Mappings** dialog box appears.

6. Click **Add**.

The **Path Mapping** dialog box appears.

7. Enter the path which was used to compile the binary, and the path where the files can be found on your local machine.
8. Continue to add mappings as needed.

Now to find out which path mappings are missing, right-click on the binary file in the **CodeWarrior Projects** view, and select the **Properties** from the context-menu. This displays the **Paths and Symbols** page. Here you can see which path mappings are still missing, and you can fix them accordingly.



## Chapter 5 Compiler

In this chapter, you find Frequently Asked Questions (FAQs) related to compiler settings in the CodeWarrior IDE.

- [FAQs - Compiler](#)

### 5.1 FAQs - Compiler

In this topic:

- [How do I compile my project?](#)
- [Where do I see the warning and error messages?](#)
- [When I compile my project, where can I see the commands that are being executed?](#)
- [Does CodeWarrior for Microcontrollers v10.x supports the CMSIS 2.0 DSP library?](#)
- [Is there a way to specify a half-precision datatype in MCU 10.x?](#)
- [#pragma pack\(\) does not work in my Kinetis project. What do I do?](#)
- [How do I generate a Cyclic Redundancy Check in Eclipse based CodeWarrior for Microcontrollers v10.x?](#)
- [I am using CodeWarrior for MCU10, Special Edition which allows up to 128 KByte of code to download for my ARM/Kinetis projects. But, when I tried to debug an Example that I have downloaded from the web, I get this dialog: "Download size limit has been exceeded. Please check your license." What do I do?](#)
- [What is PC-lint?](#)
- [What is Linticator?](#)
- [How do I lint my project without using PC-lint plugin?](#)
- [How do I configure the build settings to lint my project without plugin?](#)
- [How do I set up a batch file to call PC-lint?](#)
- [How do I define the list of files to lint my project without plugin?](#)
- [How do I pass the project specific options to lint my project without plugin?](#)
- [How do I define the message format to lint my project without plugin?](#)

### 5.1.1 How do I compile my project?

In order to compile the project, follow steps below:

1. From CodeWarrior Projects, right-click on project name.

A context menu appears.

2. From context menu, select **Clean Project** .

If **Build Automatically** is checked, then when **Clean Project** executes, it creates a **Binaries** folder that contains the \*.abs file that are used for debugging. If **Build Automatically** is unchecked, then **Clean Project** removes any previously-built binaries, including the \*.abs file.

3. Select **Project > Build Project** to compile.

The project compiles.

### 5.1.2 Where do I see the warning and error messages?

The warning and errors messages appear in the **Problems** view. To display the **Problems** views select **Window > Show View > Problems** . The **Problems** view appears docked to the lower part of the screen. Double click on error message in **Problems** view to navigate to the relevant source code. The **Problems** view shows problems for all open projects. If there is more than one open project, the window may show problems not associated with the project you are building. Compiler and linker error message are also shown in the **Problems** view.

### 5.1.3 When I compile my project, where can I see the commands that are being executed?

The commands that are being executed from the compiler or linker can be viewed in the console view. From the IDE menu bar, select **Window > Show View > Console** . The **Console** view appears docked to the lower part of the screen.

## 5.1.4 Does CodeWarrior for Microcontrollers v10.x supports the CMSIS 2.0 DSP library?

You need to get the CMSIS library from ARM. The support for DSP instructions is implemented in an include file and is located in the distribution at the following location:

```
<CWInstallDir>\CW MCU v10.x\MCU\ARM_EABI_Support\ewl\EWL_Runtime\Runtime_ARM\Common_Includes
\cw_cortex_M4_dsp.h
```

*CWInstallDir* is the directory where the CodeWarrior software is installed.

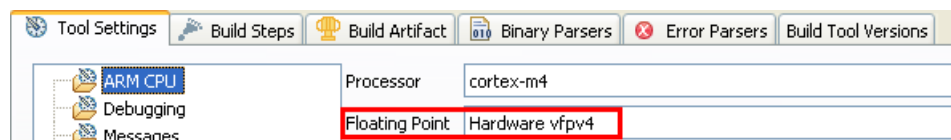
### NOTE

This library provides the standardized intrinsic function interface to the DSP instructions of the M4.

## 5.1.5 Is there a way to specify a half-precision data type in MCU 10.x?

CodeWarrior for Microcontrollers v10.2 have full support for the single precision FPU as it is included in the ARM CortexM4 architecture. However, if you are using the CodeWarrior for Microcontrollers v10.1 with SP for Kinetis P3, you will not have this feature included.

The following image shows the CodeWarrior for Microcontrollers v10.2 option in project build settings which you can use to activate the support



**Figure 5-1. ARM Compiler - Support for the Single Precision FPU**

There is also a similar option in the **Languages and Build Tools Options** page (refer to the image listed below) of the *New Project Wizard* for all derivatives of the Kinetis which support FPU (ones containing *F* in the part name).

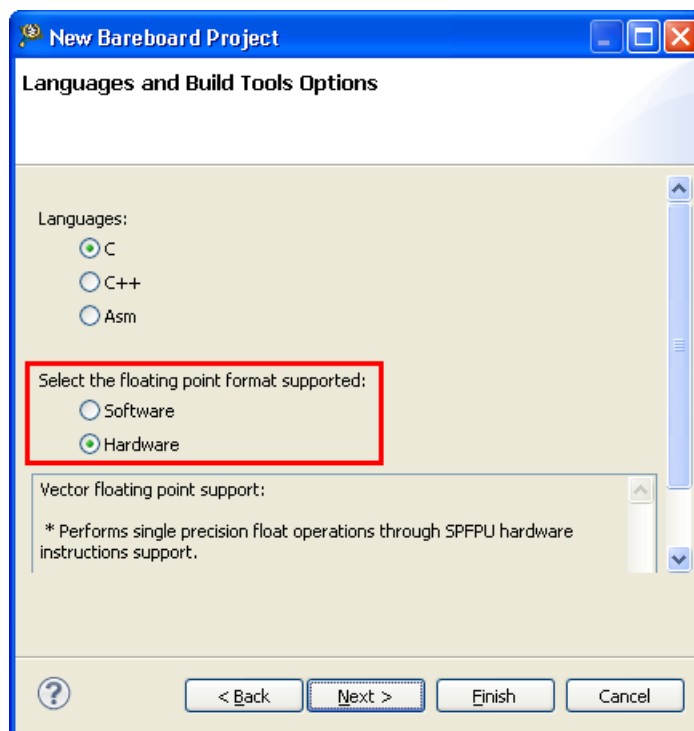


Figure 5-2. Languages and Build Tools Options - Single Precision FPU

### 5.1.6 #pragma pack() does not work in my Kinetis project. What do I do?

This pragma is not supported. You have to use `__attribute__((align))` as a workaround instead ( [Listing: Using \\_\\_attribute\\_\\_\(\(align\)\)](#)). This attribute is documented in the *CodeWarrior Development Studio for Microcontrollers v10.x Kinetis Freescale Build Tools Reference Manual* (CWMCUKINCOMPREF).

#### Listing: Using `__attribute__((align))`

```
typedef struct
{
    unsigned long x __attribute__((aligned (1)));
    unsigned char y __attribute__((aligned (1)));
} new_struct;
```

## 5.1.7 How do I generate a Cyclic Redundancy Check in Eclipse based CodeWarrior for Microcontrollers v10.x?

The CodeWarrior for Microcontrollers v10.x is not installed with the `crcgen.exe`. To generate a Cyclic Redundancy Check (CRC) you have to install the `crcgen.exe` here:

```
<CWinstalldir>\MCU\bin\plugins\Support\crcgen.exe
```

where, `<CWinstalldir>` is the installation directory of the CodeWarrior software.

You have to create a `.crc` file in your project and configure the post-linker in MCUv10.x to use it. To do that:

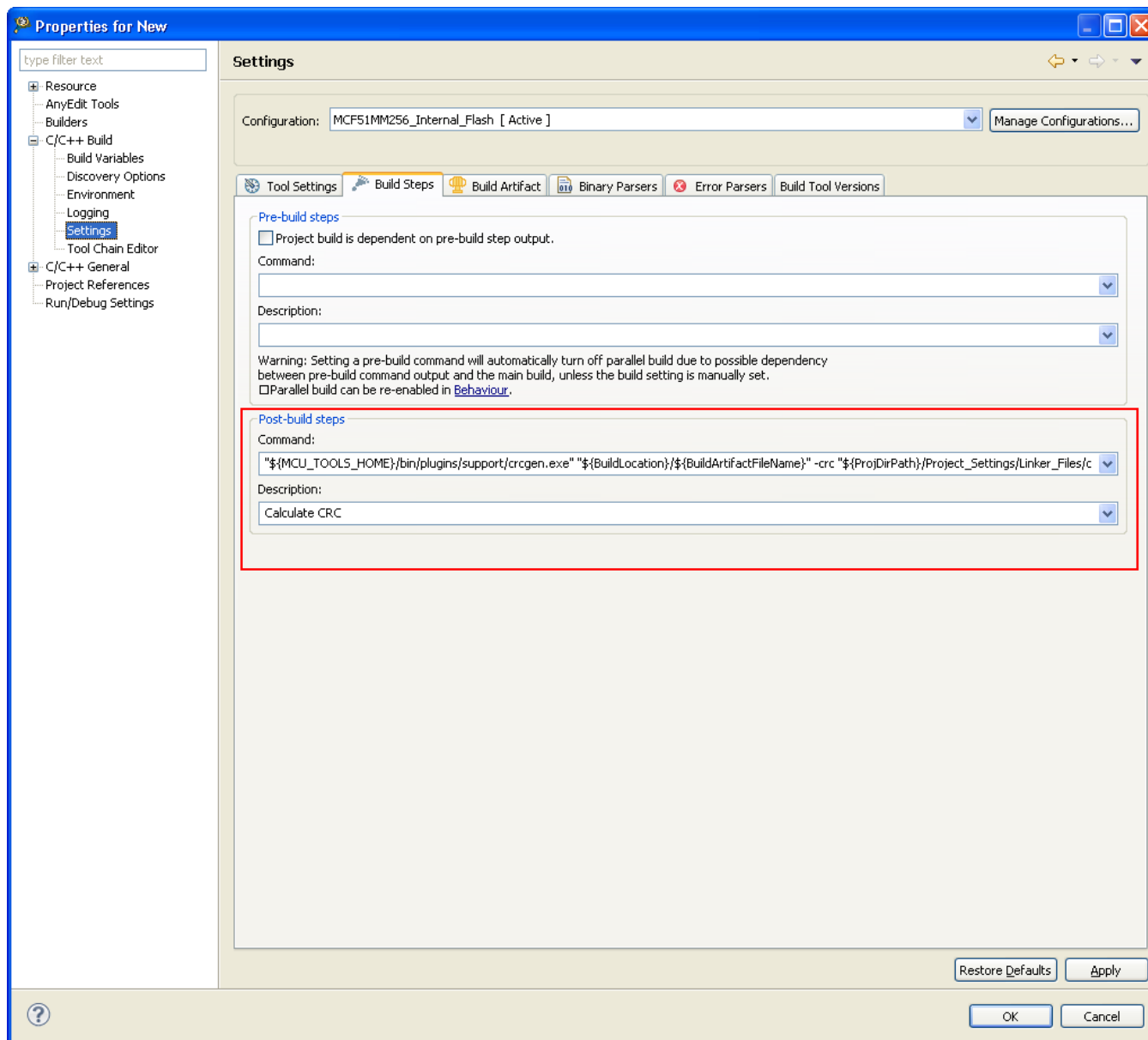
1. Select the desired project in the **Codewarrior Projects** view, and then select **Project > Properties** from the IDE menu bar.

The Properties for `<project_name>` dialog box appears.

2. In the left panel, expand the **C/C++ Build** tree-control and select **Settings**.

The **Settings** page appears in the left panel.

3. Click the **Build Steps** tab.



**Figure 5-3. Properties for <project\_name> - Build Steps Tab**

4. In the **Description** textbox enter the desired description, for example `Calculate CRC`, and in the **Post-build steps** group, enter the following command in the **Commands** textbox:

```
"${MCU_TOOLS_HOME}/bin/plugins/support/crcgen.exe" "${BuildLocation}/${BuildArtifactFileName}" -crc "${ProjDirPath}/Project_Settings/Linker_Files/calc_crc.crc" -o "${BuildLocation}/${BuildArtifactFileName}.crc.elf"
```

where,

- `${MCU_TOOLS_HOME}`: points to the MCU folder inside the CodeWarrior MCU10.x installation
- `${BuildLocation}`: points to the output folder inside your project of your build where the ELF file is located



- `${BuildArtifactFileName}`: this variable contains the file name of your ELF file
- `${ProjDirPath}`: folder name of your project

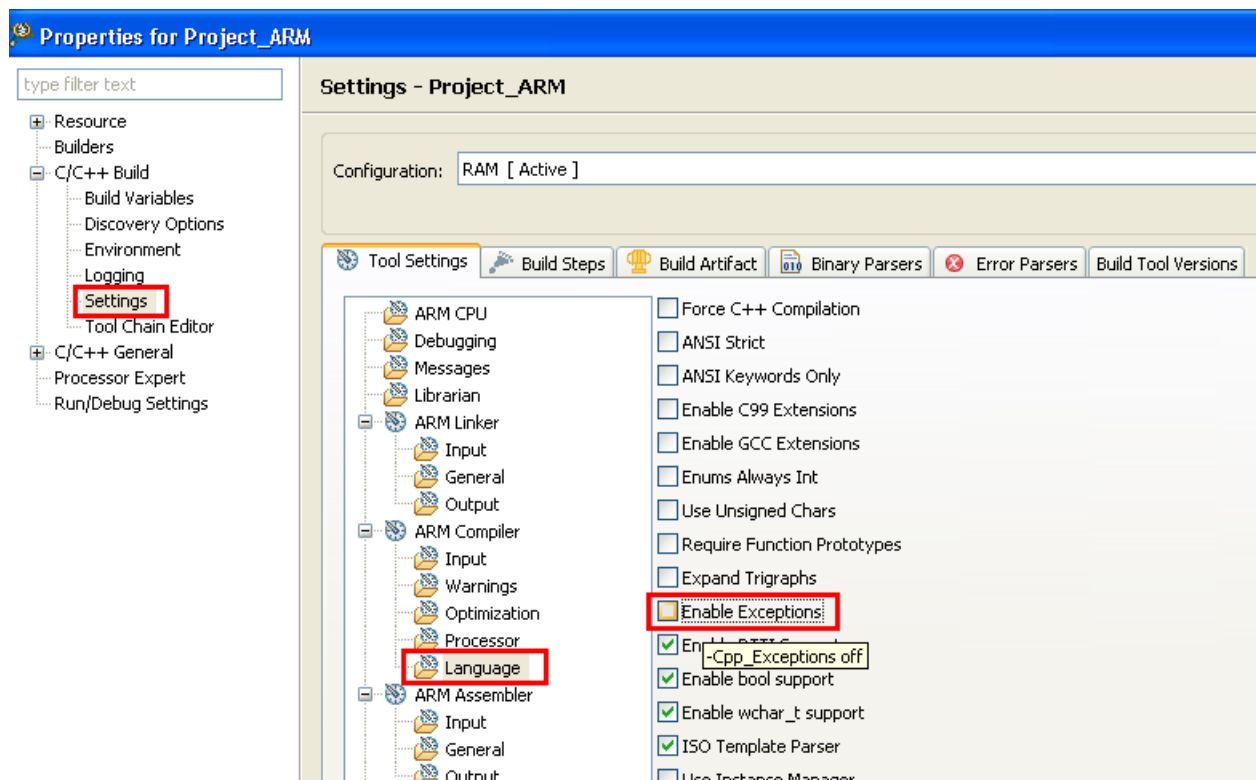
5. Click **OK**.

The changes you made are saved and when you build your project, the post build step will be executed as well to calculate the CRC into your application.

### 5.1.8 I am using CodeWarrior for MCU10, Special Edition which allows up to 128 KByte of code to download for my ARM/Kinetis projects. But, when I tried to debug an Example that I have downloaded from the web, I get this dialog: "Download size limit has been exceeded. Please check your license." What do I do?

Check the code for code size and language as specified for your Special Edition of CodeWarrior software and correct the code if necessary.

In case your code is in specified limits and still resulting in problem, your project might have the **Enable Exceptions** option enabled in the ARM Compiler settings. To solve this problem, you have to disable the **Enable Exceptions** in the compiler settings, as the following figure shows:



**Figure 5-4. ARM Compiler > Language Panel**

## 5.1.9 What is PC-lint?

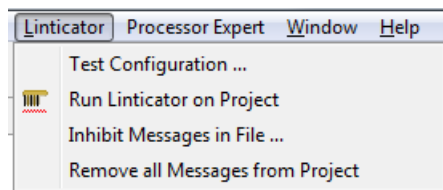
PC-lint is a compiler, it compiles the source files. But, like any other compiler it does not produce object code, instead it produce messages. PC-lint sets the standard for static code analysis: it is used by many companies developing safety critical applications. Additionally PC-lint can check the MISRA rules which are standard for many coding guide lines.

## 5.1.10 What is Linticator?

The Linticator is an eclipse based plugin for PC-lint. The Linticator plugin has been developed at the IFS (Institute for Software) in Rapperswil/Switzerland. You can register at the [www.linticator.com](http://www.linticator.com) website and you get a 30 day evaluation license.

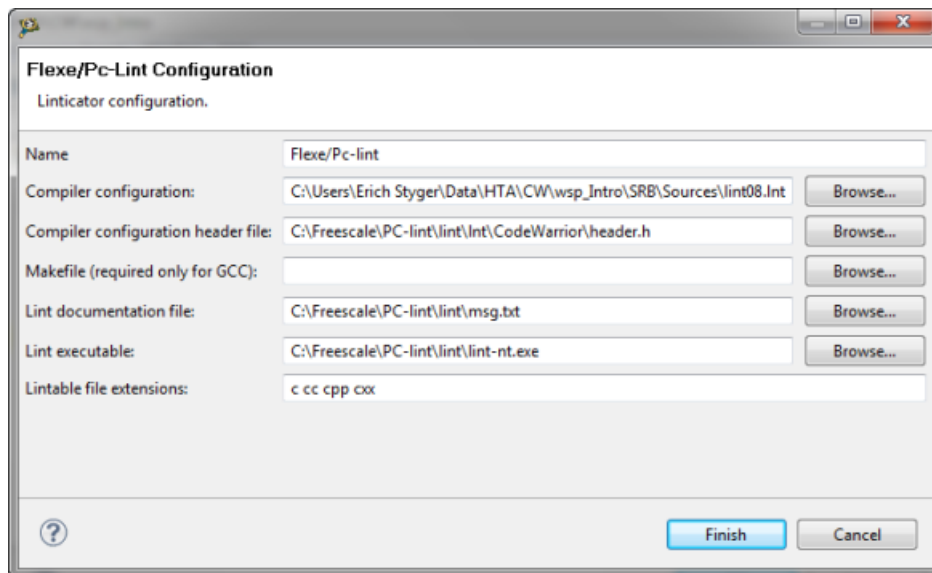
The following are some useful features:

- Linticator nicely integrates into eclipse, and adds a new IDE menu, as the following figure shows:



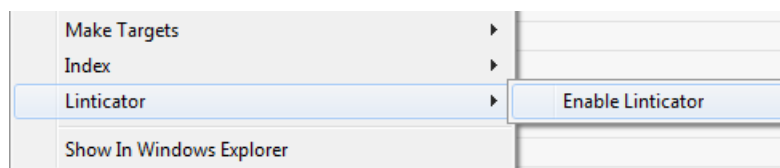
**Figure 5-5. Linticator Menu**

- Using **Flexe/PC-Lint Configuration** dialog box, you can configure the different settings.



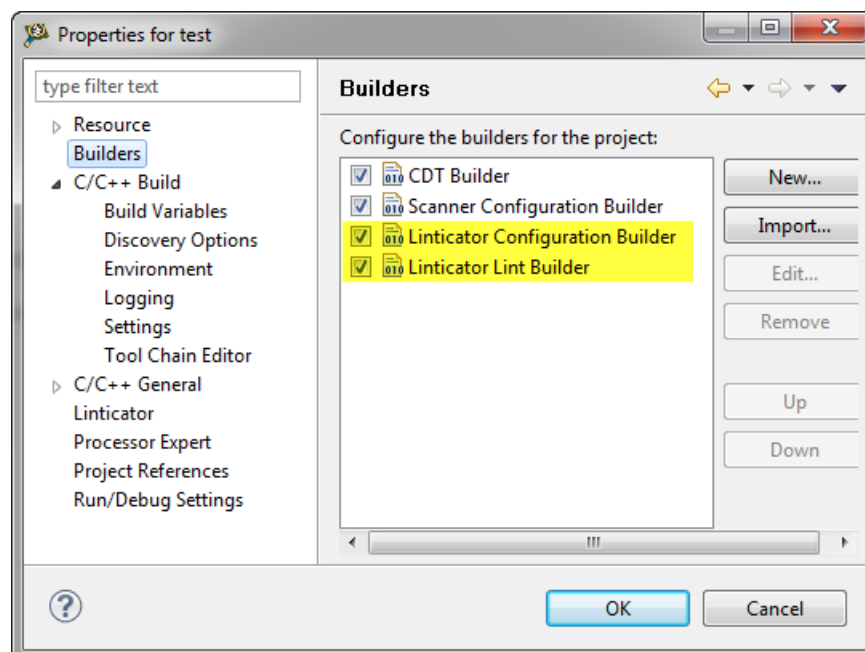
**Figure 5-6. Flexe/PC-Lint Configuration Dialog Box**

- You can enable Linticator for a project through the context menu.



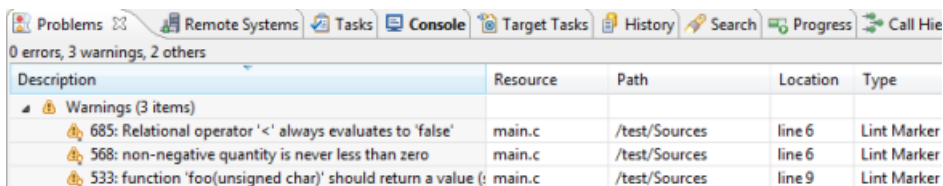
**Figure 5-7. Enabling Linticator for a Project**

- The plugin integrates into eclipse CDT as a builder.



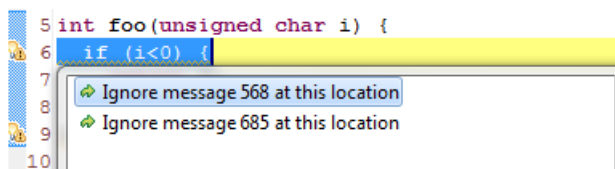
**Figure 5-8. Linticator Builder**

- The Linticator generates the PC-lint messages in the Problems View, as the following figure shows:



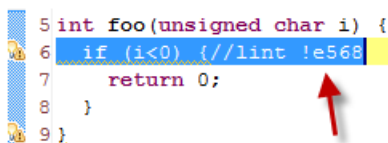
**Figure 5-9. Problems View with Linticator Messages from PC-lint**

- The markers appear in the source file for proper indication. You can simple ignore a message by using the context menu.



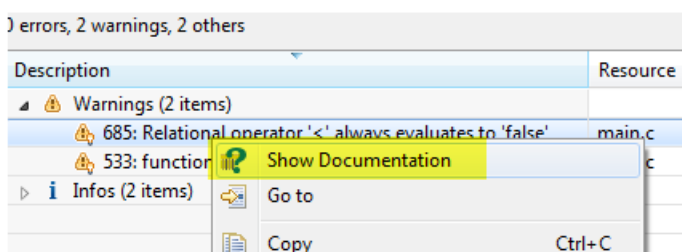
**Figure 5-10. Source with Problem**

This automatically inserts the comment to tell PC-lint to ignore that message.



**Figure 5-11. Quickfix in Source**

- You can get an explanation on a message, by using the **Show Documentation** context menu.



**Figure 5-12. Show Documentation for PC-lint Message**

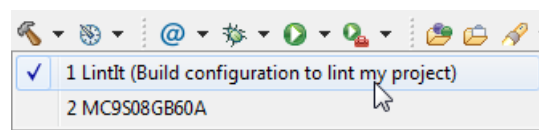
### 5.1.11 How do I lint my project without using PC-lint plugin?

To lint your project, you have to:

1. Configure the build settings (for more information, refer to the topic [How do I configure the build settings to lint my project without plugin?](#))

2. Set up a batch file to call PC-lint (for more information, refer to the topic [How do I set up a batch file to call PC-lint?](#))
3. Define the list of files (for more information, refer to the topic [How do I define the list of files to lint my project without plugin?](#))
4. Specify the options (for more information, refer to the topic [How do I pass the project specific options to lint my project without plugin?](#))
5. Define the message format (for more information, refer to the topic [How do I define the message format to lint my project without plugin?](#))
6. Build the Configuration

You can run it like any other build configuration, click the inverted triangle in the Build button from the IDE toolbar



**Figure 5-13. Building Lint Configuration**

The output messages appears in the **Problems** view.

The figure listed below is an example that shows several lint errors for `main.c`. The messages appears in the **Problems** view, and are listed as well in the **Console** view.

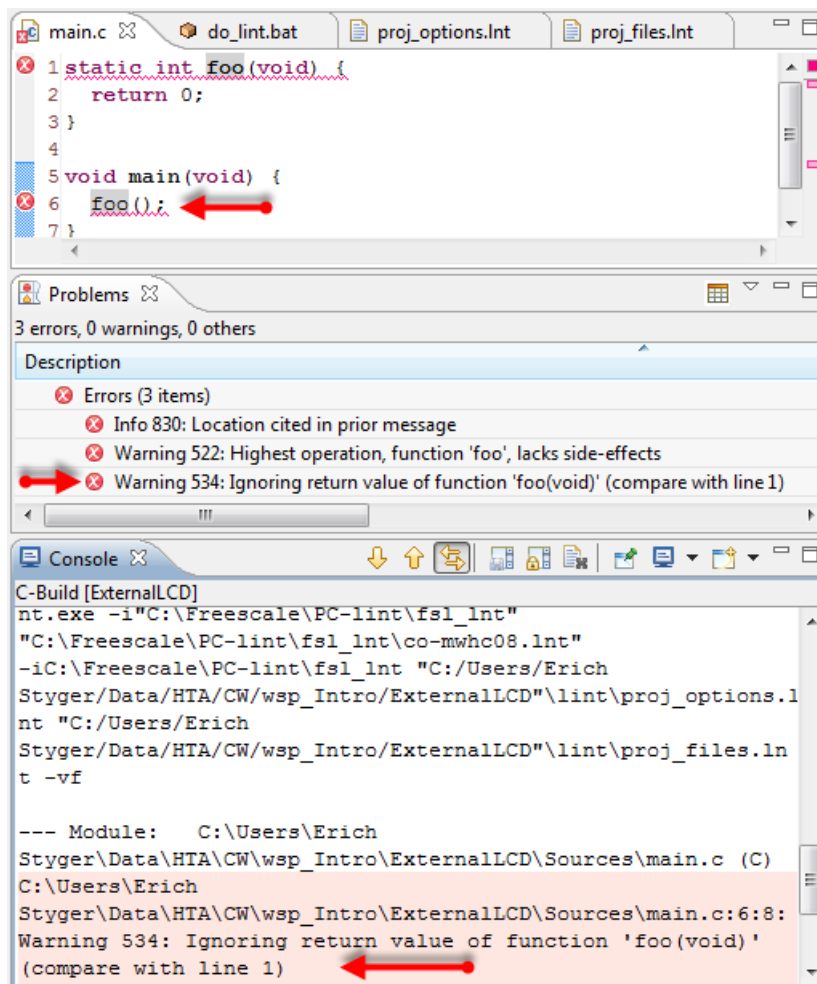
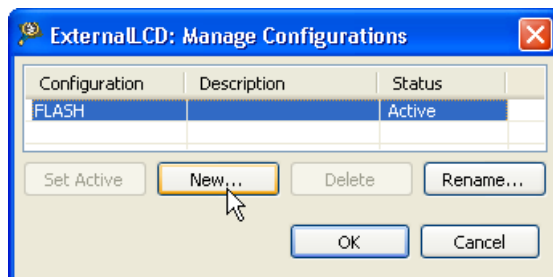


Figure 5-14. Lint Output Example

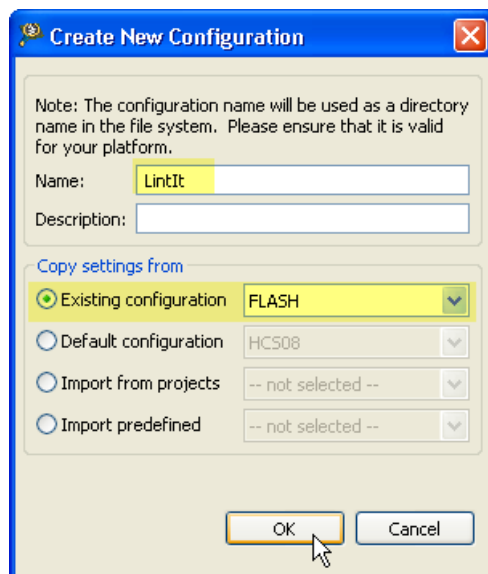
### 5.1.12 How do I configure the build settings to lint my project without plugin?

To your existing project add a managed make build configuration for PC-lint. The build configuration is set up to call a batch file. Select the project in the **CodeWarrior Projects** view, then select **Project > Build Configurations > Manage** from the IDE menu bar and create a new configuration using the **New** button:



**Figure 5-15. Manage Configurations Dialog Box**

Enter a name in the **Name** textbox for the new configuration, and copy the settings from the existing configuration.

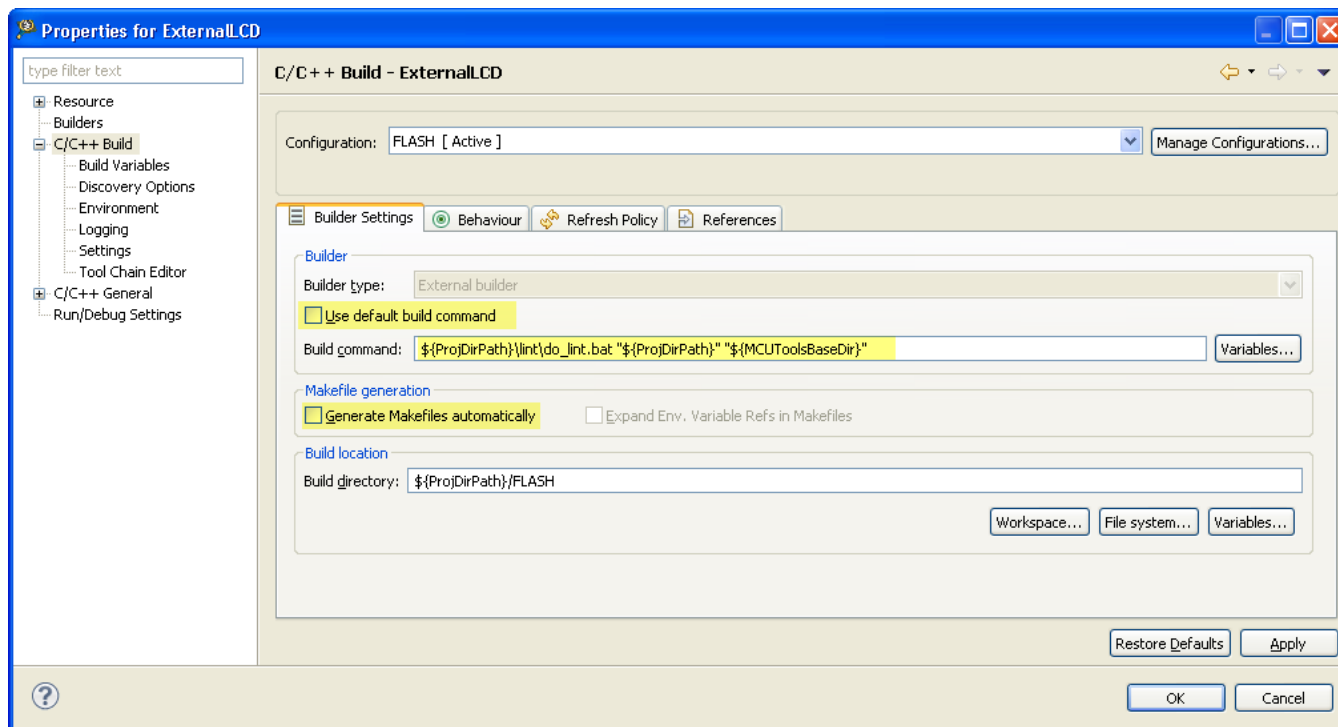


**Figure 5-16. Create New Configuration Dialog Box**

To configure this new configuration to use the lint compiler, select **Project > Properties > C/C++ Build > Builder Settings**, disable **Use default build command** by clearing the checkbox and enter the following command in the **Build command** textbox:

```
${ProjDirPath}\lint\do_lint.bat "${ProjDirPath}"
"${MCUToolsBaseDir}"
```

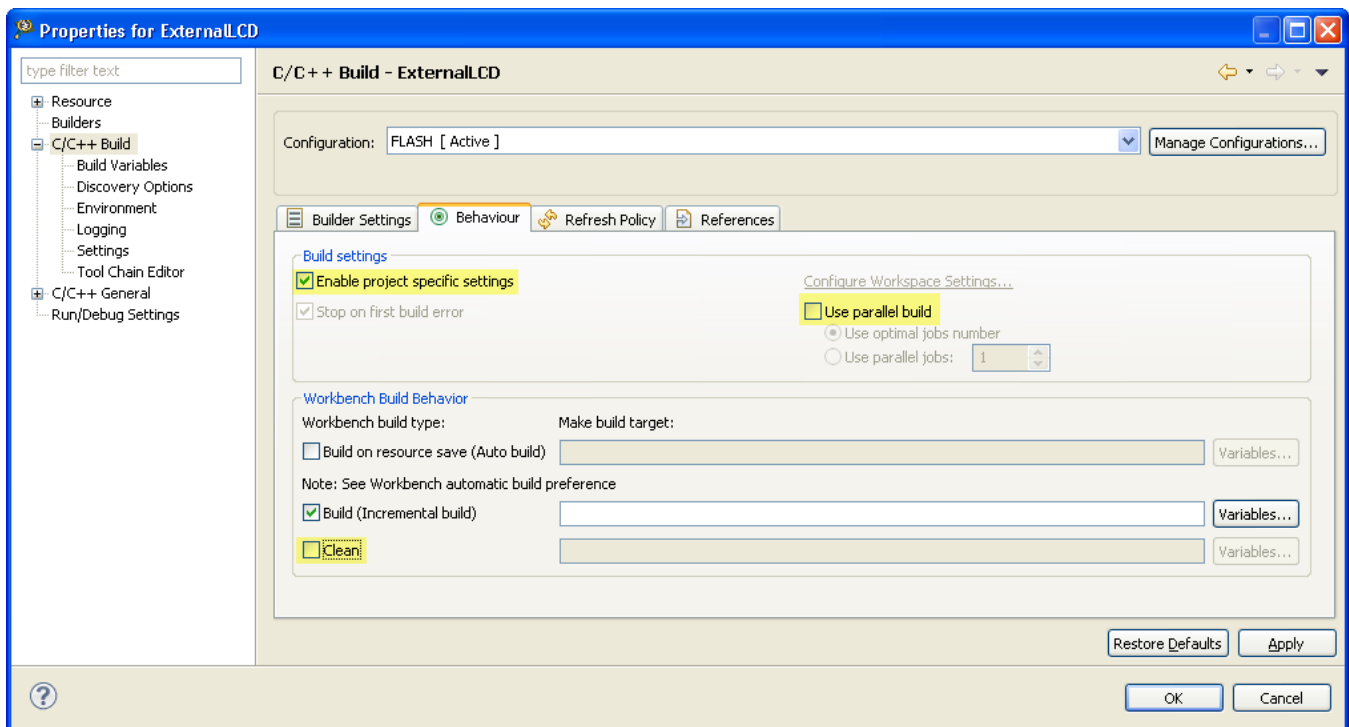
Additionally, disable the **Generate Makefiles automatically** option by clearing the checkbox.



**Figure 5-17. Builder Settings for PC-lint Batch File**

The CodeWarrior for MCU10.x uses parallel builds by default. This adds `-j6` as option to the command line. In order to disable this, select the **Enable project specific settings** option in the **Behaviour** tab, also clear the **Clean** checkbox from the **Workbench Build Behavior** group:

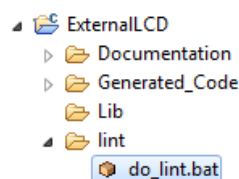




**Figure 5-18. Customizing Build Behaviour**

### 5.1.13 How do I set up a batch file to call PC-lint?

To set up a batch file to call PC-lint, separate the lint files from the rest of your build and project, create a lint sub-folder inside your project root directory with a `do_lint.bat` batch file:



**Figure 5-19. do\_lint Batch File**

The following listing shows the content of `do_lint.bat` file:

#### Listing: do\_lint.bat File

```
@rem The arguments for this batch file:
@rem %1: The path to the project folder

@rem %2: The path to the CodeWarrior installation folder
@rem -----
@rem Path to my project folder
```

## FAQS - Compiler

```

SET PROJ_PATH=%1

@rem Path to CodeWarrior installation folder (which is e.g.
"C:\Freescale\CW MCU v10.2\eclipse\..\MCU")

SET CW_PATH=%2

@rem Path to lint-nt.exe

SET LINT_EXE=C:\lint\lint-nt.exe

@rem Path to my lint configuration files

SET LOCAL_LNT_FILES=C:\Freescale\PC-lint\fsl_lnt

@rem Path to my local lint folder inside the project

SET PROJ_LINT_PATH=%PROJ_PATH%\lint

@rem Lint configuration files and includes

SET LNT_INCLUDES=-i"%LOCAL_LNT_FILES%" "%LOCAL_LNT_FILES%\co-
mwhc08.lnt" -i%LOCAL_LNT_FILES%

@rem ----- Run PC-lint -----

%LINT_EXE% %LNT_INCLUDES% %PROJ_LINT_PATH%\proj_options.lnt
%PROJ_LINT_PATH%\proj_files.lnt -vf

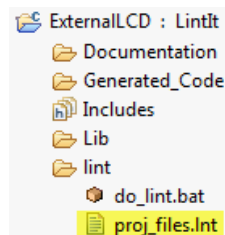
```

The batch file is called from eclipse with two arguments ( %1 and %2): with the path to the project folder and the path to the CodeWarrior installation folder. Assign them to local variables ( PROJ\_PATH and CW\_PATH) so you can use them inside the .lnt files. To know where your lint compiler is, use the LINT\_EXE. If you store your lint configuration files outside of the PC-lint installation folder, then you have to defined a path variable for this:

LOCAL\_LINT\_FILES. The PROJ\_LINT\_PATH contains the project sub-folder with all your batch and lint files for the project. In LNT\_INCLUDES specify your compiler lint configuration file, plus where lint shall search for your lint configuration files.

### 5.1.14 How do I define the list of files to lint my project without plugin?

To define the list of files, create a file `proj_files.lnt` file inside your project directory.



**Figure 5-20. File Listing the Files to Lint**

This file has all your source files listed. And define the environment variables as follows:

```
%PROJ_PATH%\Sources\main.c
```

```
%PROJ_PATH%\Sources\Events.c
```

### 5.1.15 How do I pass the project specific options to lint my project without plugin?

To pass the project specific options add them to the `proj_options.lnt` file.

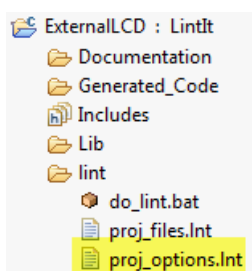


Figure 5-21. Project Options File

In the `proj_options.lnt` file, add all the paths where it can find your project files with the `-i` PC-lint option, as the following listing shows:

#### Listing: Include Paths

```
// Include paths used
-i%PROJ_PATH%

-i%PROJ_PATH%\Sources

-i%PROJ_PATH%\Generated_Code

-i%CW_PATH%\lib\hc08c\include
```

Additionally, specify all the global options, e.g. to inhibit messages:

#### Listing: Example - Inhibit Messages

```
// inhibit messages for Processor Expert libraries
-elib(19, 10)

-e766

+libh(Events.h, Cpu.h)
```

## 5.1.16 How do I define the message format to lint my project without plugin?

To define the message format so they appear properly in the **Problems** view, you have to add them as well to the `proj_options.lnt`, as the following listing shows:

### Listing: `proj_options.lnt` File - Message Format Options

```
// Coerce messages for Eclipse
-hF1

+ffn

// Normally my format is defined as follows:
// -"format=%(\q%f\q %l %C%) %t %n: %m"

// For eclipse-usage, the GCC error format is necessary,
// since we have only the default eclipse error parser available.
-"format=%(%f:%l:%C:%) %t %n: %m"

// Enable warning 831 if you are interested.
-frl

// Do not break lines
-width(0)

// And make sure no foreign includes change the format
+flm
```

## Chapter 6

# Linker

In this chapter, you find Frequently Asked Questions (FAQs) related to the linker settings in the CodeWarrior IDE.

- [FAQs - Linker](#)

### 6.1 FAQs - Linker

In this topic:

- [Can I use a different linker command file than the one being used in my project?](#)

#### 6.1.1 Can I use a different linker command file than the one being used in my project?

Yes, the linker command file can be changed by following the steps given below:

1. In the CodeWarrior Projects window, right-click on project name.

A context menu appears.

2. From context menu, select **Properties**.

The **Properties** window appears.

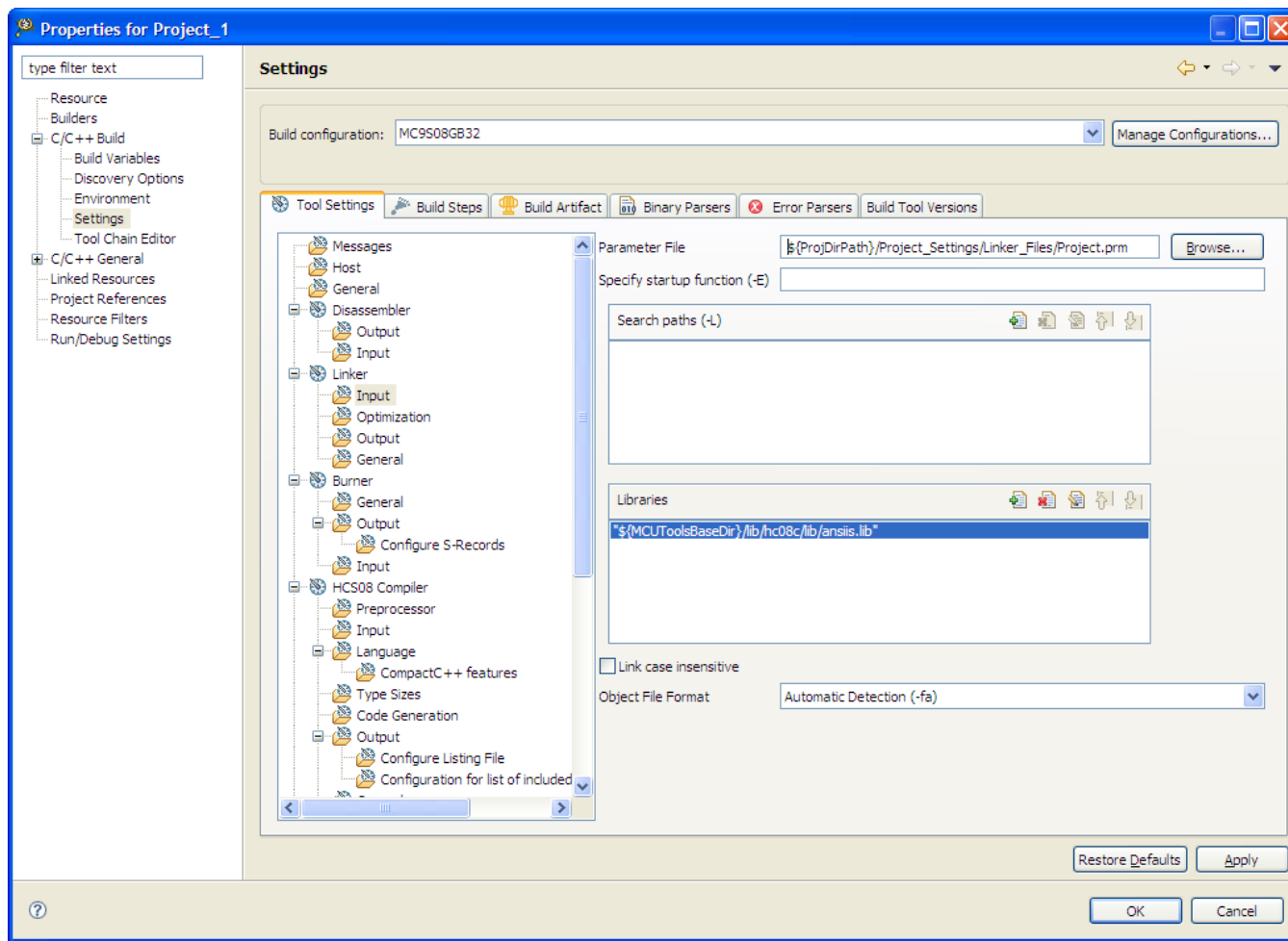
3. Select **C/C++ Build > Settings**.

The **Settings** options appear on right hand side of properties window.

4. Select **Linker > Input**.

The **Input Settings** appears.

5. The **Parameter File** textbox specifies the parameter file that is currently used.
6. Click **Browse** to select a different parameter file (refer to the image listed below).



**Figure 6-1. Properties Dialog Box-Specifying Different Parameter File for Linker**

7. Click **Apply**.
8. Click **OK**.

The linker command file changes, as specified.

# Chapter 7

## Debugger

In this chapter, you find Frequently Asked Questions (FAQs) related to the debugger settings in the CodeWarrior IDE.

The FAQs listed in this chapter are divided into five categories:

- [Breakpoints](#)
- [Variables](#)
- [Build Configuration](#)
- [Flash Programmer](#)
- [Miscellaneous](#)

### 7.1 Breakpoints

In this topic, FAQs related to breakpoints are listed.

- [How do I set breakpoints?](#)
- [How do I view the breakpoints that have been set?](#)
- [How do I disable existing breakpoints?](#)
- [What does the checkmark next to the breakpoint represent?](#)
- [What is the difference between the Toggle Breakpoint and Enable/Disable breakpoint options?](#)
- [Which breakpoint option do I select if I want to add a breakpoint?](#)
- [Which breakpoint option do I select if I want to skip a breakpoint while still keeping it in my breakpoint list?](#)
- [What if I want to delete a breakpoint?](#)
- [How do I place a breakpoint on a given physical address?](#)
- [How do I add hardware breakpoints? Can I add them through the GUI?](#)
- [How can I view the hardware breakpoints that have been set?](#)

- What is the difference between the hardware breakpoint and the software breakpoint? Why would I use one over the other?
- What is the purpose of the Limit New Breakpoint to Active Debug Context button in the Breakpoint window?
- Is there a way to add breakpoints to C and Assembly files that exist in a library?
- How do I set a breakpoint at a particular function?
- What is the difference between a breakpoint and a line breakpoint?
- The warnings symbols in Editor distort visibility of breakpoints. How can I stop these warning symbols from appearing?
- How do I use breakpoints on data?
- Can I use any number of software breakpoints while debugging in RAM?
- How does software breakpoints works while debugging in RAM?
- While debugging in RAM my debugger halts and reports an "illegal breakpoint" error. What do I do?
- Can I use software breakpoints on code running in flash memory?
- What is a hardware breakpoint?
- How do I select the type of breakpoint for my application in the Editor view?
- How do I monitor that how many hardware breakpoints are used in my application?
- If I launch my debug sessions with too many breakpoints, then the debugger shows this error message: "Breakpoint Warning: Unable to set all breakpoints requested. Some breakpoints may be disabled and will not stop the processor." What do I do?
- How do I step over the breakpoints without using Breakpoints view?
- **NEW** What are the points to consider while using Stepping Backwards feature in Debugging?

### 7.1.1 How do I set breakpoints?

To set breakpoints in a source or assembly file, follow one of the four methods given below.

1. Right-click on the left-hand side of source file (the Gray bar)

A context menu appears.

2. Select **Toggle Breakpoint** at the line where you want the breakpoint.

-or-

1. Select **Ctrl+Shift+B** at the line where you want the breakpoint.

-or-



1. Double-click on the line where you want the breakpoint

-or-

1. Click on the line where you want the breakpoint.
2. From the IDE menu bar, select **Run > Toggle Breakpoint** .

A breakpoint appears on the left hand side of source code file.

## 7.1.2 How do I view the breakpoints that have been set?

To view the breakpoints that have been set, select **Window > Show View > Breakpoints** from the IDE menu bar. The **Breakpoints** window appears docked to the upper right-hand side of debugger perspective. The breakpoints window displays the breakpoints that you have enabled.

## 7.1.3 How do I disable existing breakpoints?

To disable existing breakpoints:

1. Click on the line that has the breakpoint enabled.
2. From the IDE menu bar, select **Run > Toggle Breakpoint**.

-or-

- Press **Ctrl+Shift+B** at the source line where you want the breakpoint disabled.

-or-

- Double-click the line where you want the breakpoint disabled.

-or-

1. Right-click on the left-hand side of source file (the Gray area)

A context menu appears.

2. Select **Toggle Breakpoint**.

-or-

1. Select the breakpoint that you want disabled in the **Breakpoints** window.
2. Right-click the breakpoint.

A context menu appears.

3. Select **Disable**.

The Breakpoint gets disabled.

#### 7.1.4 What does the checkmark next to the breakpoint represent?

The checkmark next to a breakpoint indicates that the debugger has resolved the breakpoint and will be hit by the debugger. If the breakpoint does not have a checkmark, the debugger will not hit it.

#### 7.1.5 What is the difference between the Toggle Breakpoint and Enable/Disable breakpoint options?

The **Toggle Breakpoint** option adds or removes breakpoints to the source and breakpoint list. The **Disable/Enable** option makes breakpoints either active or inactive. New breakpoints cannot be added using the **Disable/Enable** breakpoint option.

#### 7.1.6 Which breakpoint option do I select if I want to add a breakpoint?

To add a breakpoint, select **Toggle Breakpoint** .

#### 7.1.7 Which breakpoint option do I select if I want to skip a breakpoint while still keeping it in my breakpoint list?

If you want to skip a breakpoint, but still keeping it in your breakpoint list, then select the **Disable Breakpoint** option.

## 7.1.8 What if I want to delete a breakpoint?

If you want to delete a breakpoint from the source code, select the **Toggle Breakpoint** option.

## 7.1.9 How do I place a breakpoint on a given physical address?

To add a breakpoint on a given physical address, follow steps below.

1. In the CodeWarrior projects window, right-click on project name.

A context menu appears.

2. From context menu, select **Debug As > Debug Configurations** .

The **Debug** window appears.

3. Underneath **CodeWarrior Download** , select project name.

4. Select **Debug** .

The **Debug** perspective appears.

5. In **Debug** perspective, select **Window > Show View > Debugger Shell** .

The **Debugger Shell** window appears.

6. In **Debugger Shell** , type `bp <address>`.

This will set breakpoint at the `<address>` memory location.

## 7.1.10 How do I add hardware breakpoints? Can I add them through the GUI?

You can add hardware breakpoints through the GUI. To add them through the GUI, follow steps below:

1. Right-click on the left hand side of source file (the Gray bar).

A context menu appears.

2. From context menu, select **Set Special Breakpoint > Hardware** .

The Hardware breakpoint appears next to source line.

### 7.1.11 How can I view the hardware breakpoints that have been set?

To view the hardware breakpoints, select **Window > Show View > Breakpoints** from the IDE menu bar. The **Breakpoints** window appears.

### 7.1.12 What is the difference between the hardware breakpoint and the software breakpoint? Why would I use one over the other?

A software breakpoint inserts a debug instruction into the code, so it only works in the writable memory. If you are working in ROM, then you have to use a hardware breakpoint. The hardware breakpoints make use of the hardware debug support using OCE.

### 7.1.13 What is the purpose of the Limit New Breakpoint to Active Debug Context button in the Breakpoint window?

This feature is basically a toggle button in the Breakpoint window that causes a new breakpoint (or a watchpoint) to get set only on the active debug context. This feature lets you set breakpoints at certain locations in different source file instead of same breakpoints being set in the same files. The behavior that happens right now is if you set a breakpoint in one file that is being shared by multiple cores it will set that breakpoint in the same file for all cores. With this feature you will be able to set different breakpoints in different files.

If no debug context exists, then the breakpoint is installed in all contexts as normal. Note that once set, the behavior of the breakpoint is the same as the existing filtering behavior, such that filtering is maintained for the individual context during a restart but is lost after a Terminate. After a terminate, the breakpoint is installed in all debug contexts.

### 7.1.14 Is there a way to add breakpoints to C and Assembly files that exist in a library?

Yes. You need to make sure that you compile the library with debug information. You also need to go to Project Properties and make sure that the project has been imported. Make sure that you put a checkmark on the library project.

### 7.1.15 How do I set a breakpoint at a particular function?

To set a breakpoint at a particular function, you can either do it through the debugger shell or the debugger GUI. If you are going to set it from the debugger shell, you would type in `bp <function_name>`. If you are going to set it from the GUI, then you right-click the function name and select **Toggle Breakpoint**.

### 7.1.16 What is the difference between a breakpoint and a line breakpoint?

They are practically the same thing. Eclipse introduced Method Breakpoints which we don't have in the classic CodeWarrior tools, hence the need for a distinction. The online documentation states that regular breakpoints suspend the execution of a thread before a line of code or method is executed. Regular breakpoints include:

- Line Breakpoint; that suspends thread execution when the line of code it is applied to is executed.
- Method Breakpoint; that suspends thread execution when the method that it was applied to is entered or exited (or both).
- Address Breakpoint; that breaks when the core executes code at the given address.

### 7.1.17 The warnings symbols in Editor distort visibility of breakpoints. How can I stop these warning symbols from appearing?

In order to remove these warning symbols, follow the steps below in the C/C++ perspective:

## breakpoints

1. From the IDE menu bar, select **Window > Preferences** .

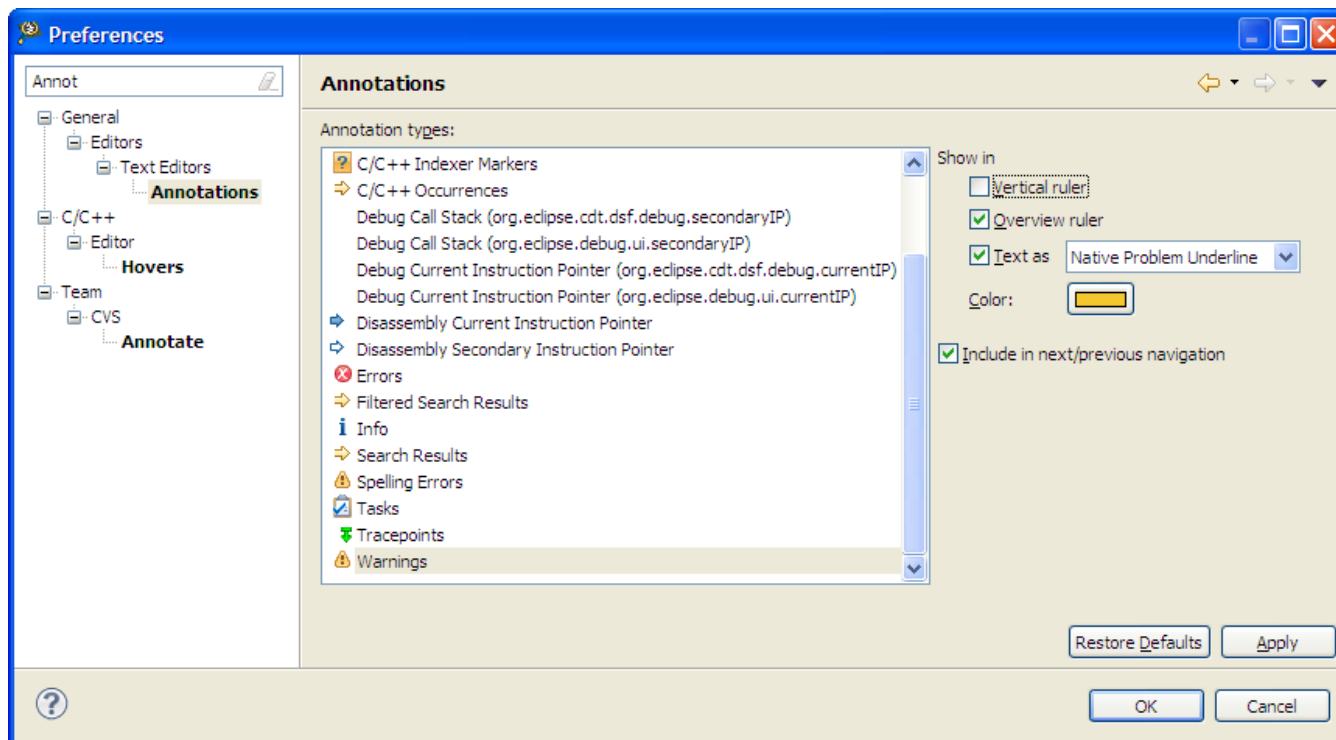
The **Preferences** window appears.

2. Select **General > Editors > Text Editors > Annotations** .

The **Annotations** settings pane appears.

3. Select **Warnings** .

4. In the **Show in** area, clear the **Vertical ruler** checkbox, as shown in the following figure.



**Figure 7-1. Preferences Window - Annotations Page**

5. Click **Apply** .
6. Click **OK** .

The warning symbols are removed.

### 7.1.18 How do I use breakpoints on data?

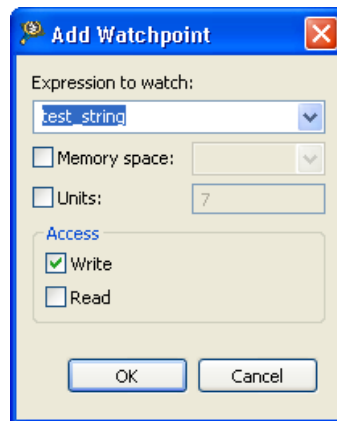
The **Watchpoints** are *data breakpoints*. Instead breaking/stopping the application on an instruction, it allows to stop the CPU on a memory access. The number and capabilities of watchpoints depend on the microcontroller used. Most vendors offer a few watchpoints, and even a single one can save many hours of debugging.

You can set it up to trigger either on read or write access, or both. The hardware implements this typically with an address comparator. If there is a match of the address on the bus, then the CPU gets stopped, or whatever you configure it to do.

To add a watchpoint:

1. Start a debugging session.
2. Right-click in the **Variables** view and select **Add Watchpoint (C/C++)** from the context-menu.

The **Add Watchpoint** dialog box appears.

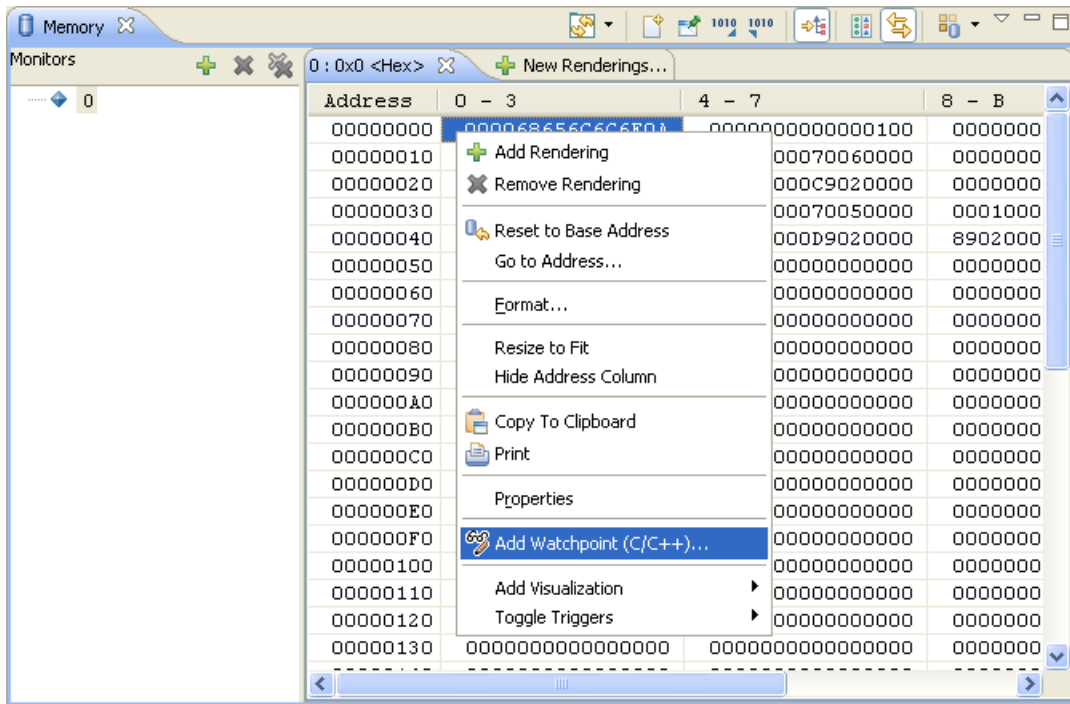


**Figure 7-2. Add Watchpoint Dialog Box**

3. Specify the expression for which you want to add the watchpoint in the **Expression to watch** textbox.
4. Select the Memory space, Units and access type as per requirement.
5. Click **Ok** .

As the watchpoints are the *data breakpoints*, the new watchpoint appears in the **Breakpoints** view.

Similarly you can create a watchpoint on memory in the **Memory View**. The following figure shows the **Memory** view with **Add Watchpoint (C/C++)** option:



**Figure 7-3. Memory View with Add Watchpoint (C/C++) Option**

### 7.1.19 Can I use any number of software breakpoints while debugging in RAM?

Yes, you can use almost unlimited number of software breakpoints while debugging in RAM.

### 7.1.20 How does software breakpoints works while debugging in RAM?

The software breakpoints are generally used for applications in RAM, because changing the instructions in RAM is easy for the debugger.

With software breakpoints, the debugger replaces an instructions with an illegal instruction or with a dedicated breakpoint instruction supported by the instruction set of the microcontroller. To illustrate this, below is some code in RAM starting at address  $0 \times 100$ . The LDA load instruction from the address  $0 \times 80$  is encoded as  $0 \times 3788$ , as the following figure shows:



```

0x100: 37 88 LDA @0x88
0x102: 27 16 MULA #0x16
0x104: 17 90 STA @0x90
    
```

**Figure 7-4. Code in RAM**

If you set a breakpoint on the multiply MULA instruction at address `0x102`. Assuming that this architecture has a breakpoint instruction encoded as `0xffff`, the debugger simply can replace the opcode at address `0x102` with the `0xffff BKPT` opcode:

```

0x100: 37 88 LDA @0x88
0x102: FF FF BKPT
0x104: 17 90 STA @0x90
    
```

**Figure 7-5. Code in RAM with Software Breakpoint**

If the processor runs the code at address `0x102`, it triggers the debug module as defined in the microcontroller architecture.

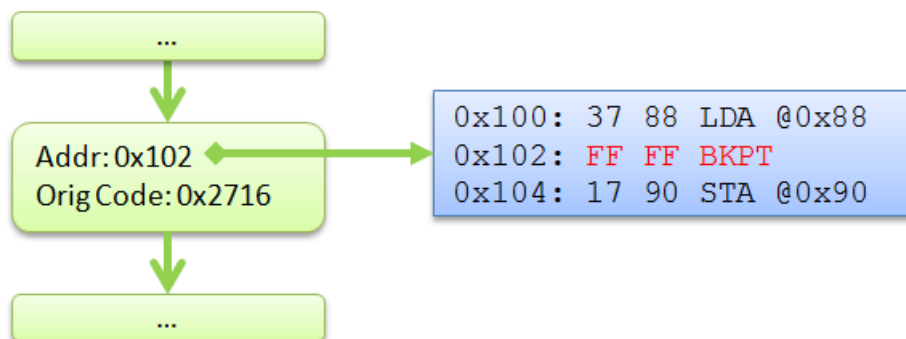
You can use this as well in your application code. For example the ARM Cortex architecture supports the BKPT instruction. You can use this to have present in an interrupt routine to stop your target. The processor will cause an *illegal instruction* exception which the debugger can catch, as the following listing shows:

**Listing: Example Software Breakpoint in RAM Application**

```

1 void __attribute__((interrupt)) ISR_Unhandled(void) {
2     asm("BKPT 255"); /* software breakpoint */
3 }
    
```

The debugger does the replacement just as part of restarting the processor. And it restores the original content after the processor has stopped. The debugger maintains a list of places where it has replaced the memory with the breakpoint instructions. You can see the replaced instruction, as the following image shows:



**Figure 7-6. Software Breakpoint List in Debugger**

### 7.1.21 While debugging in RAM my debugger halts and reports an "illegal breakpoint" error. What do I do?

If the debugger fails to restore the original instruction, then the software breakpoints remains and causes a target halt when it reaches that instruction. The problem might be caused by a board or some hardware problem. In such case, the debugger breakpoint list might gets corrupted too. To solve this problem it is best to reload the application to the target.

### 7.1.22 Can I use software breakpoints on code running in flash memory?

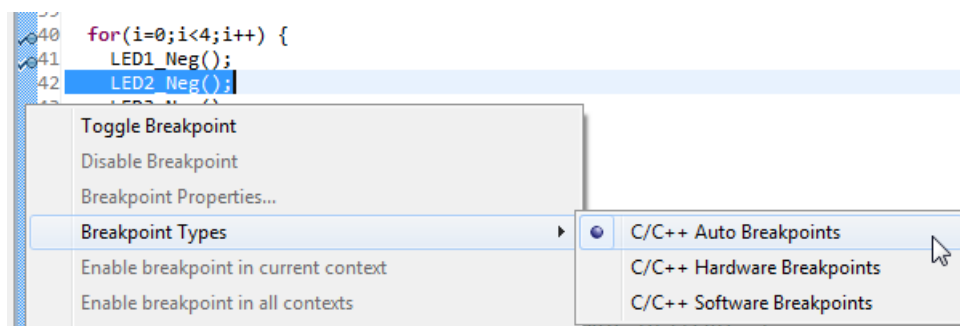
Some debuggers allows software breakpoints on code running in the flash memory. But this usually makes setting and removing breakpoints a time-consuming practice. The debugger has to erase and re-program the flash to set and remove the breakpoints. Other architectures implement a *patch list*, similar to the debugger breakpoint list in RAM. With this it is possible to set software breakpoints with hardware.

### 7.1.23 What is a hardware breakpoint?

A hardware breakpoint watch an internal bus or the program counter, and if it matches a certain condition, it stops the processor, or do whatever the hardware implements for that condition.

### 7.1.24 How do I select the type of breakpoint for my application in the Editor view?

To select the type of breakpoint in the **Editor** view, right-click on the marker bar and select the type of breakpoint from the **Breakpoint Types** option's menu list. The following figure shows the context-menu with **Breakpoint Type** option's menu list:

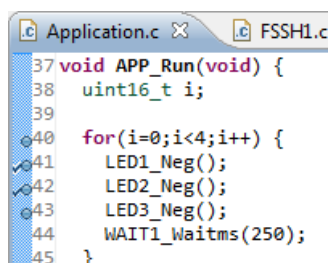


**Figure 7-7. Breakpoint Type Menu**

This is especially useful if you are running an application with parts in RAM and parts in FLASH, or if you want to use hardware breakpoints for code in RAM.

### 7.1.25 How do I monitor that how many hardware breakpoints are used in my application?

The **Breakpoints** view lists and manages all types of breakpoints that you have set in your application. There may be a case where not all the breakpoints that you have set are passed to the debugger. The following figure shows an example:



**Figure 7-8. Breakpoints in Editor View**

In the figure listed above only two breakpoints are appearing with the check mark. You can easily monitor such cases using the **Breakpoints** view.

Name	Context	Address	Type
<input checked="" type="checkbox"/> Application.c [line: 40]			Regular
<input type="checkbox"/> Unresolved breakpoint	ARM Processors, Freedom_PEx.elf		
<input checked="" type="checkbox"/> Application.c [line: 41]			Regular
<input checked="" type="checkbox"/> APP_Run, Freedom_PEx.elf	ARM Processors, Freedom_PEx.elf	Virtual:0x000004fa	Hardware
<input checked="" type="checkbox"/> Application.c [line: 42]			Regular
<input checked="" type="checkbox"/> APP_Run, Freedom_PEx.elf	ARM Processors, Freedom_PEx.elf	Virtual:0x000004fe	Hardware
<input checked="" type="checkbox"/> Application.c [line: 43]			Regular
<input type="checkbox"/> Unresolved breakpoint	ARM Processors, Freedom_PEx.elf		

**Figure 7-9. Breakpoints View with Check Marks**

The check mark shows that the breakpoint is selected for execution. The figure listed above shows the debugger is using hardware breakpoints, as **Type** column shows, and out of four breakpoints only two are set successfully.

Always check if your breakpoints have the check mark appearing with it. Also you can closely monitor how many hardware breakpoints are used using **Breakpoints** view.

### 7.1.26 If I launch my debug sessions with too many breakpoints, then the debugger shows this error message: "Breakpoint Warning: Unable to set all breakpoints requested. Some breakpoints may be disabled and will not stop the processor." What do I do?

If you have launched your debugging sessions with too many breakpoints, your debugger may result in an error. In such case you have to disable the breakpoints from the **Breakpoints** view.

To disable a breakpoint, right-click the desired breakpoint and select **Disable** from the context menu. The following figure shows the **Breakpoints** view with disabled breakpoints:

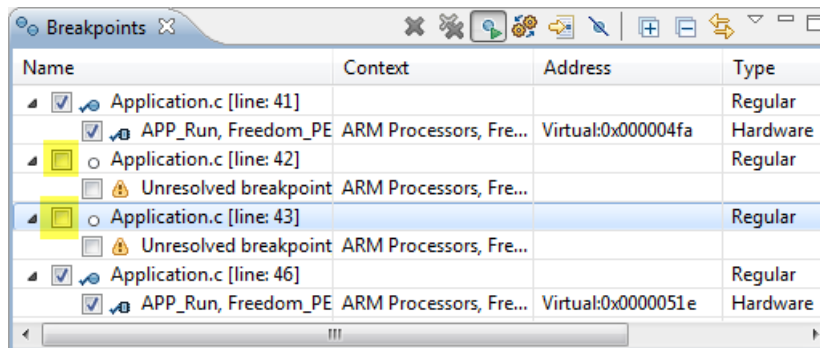


Figure 7-10. Disabled Breakpoints in Breakpoints View

### 7.1.27 How do I step over the breakpoints without using Breakpoints view?

To step over the breakpoints:

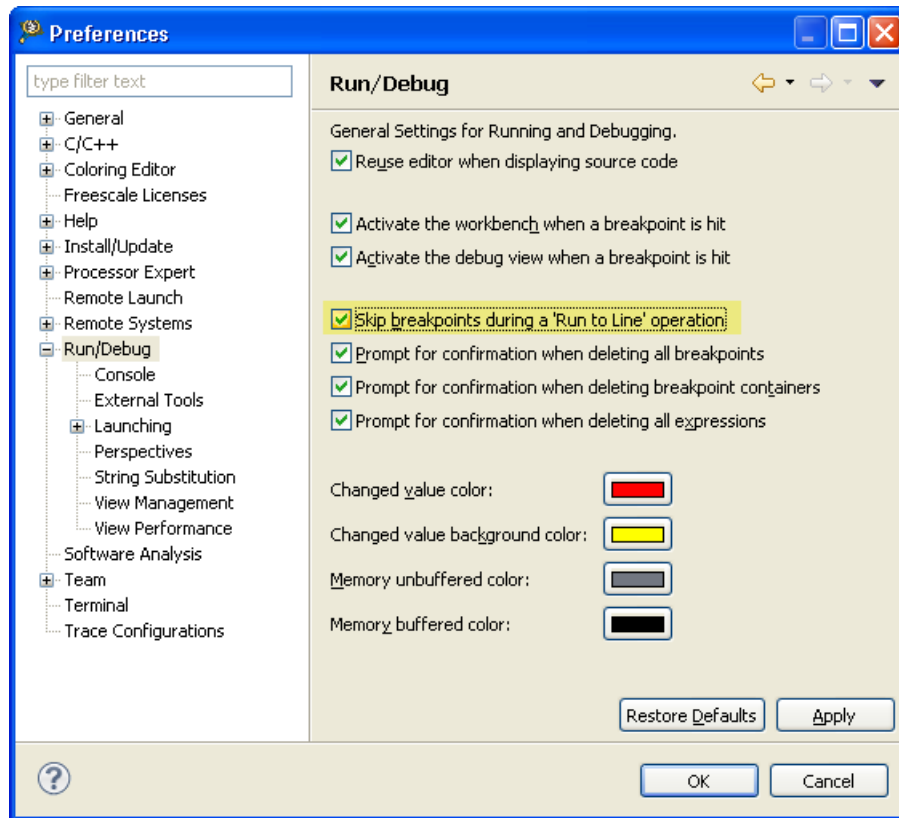
1. Select **Window > Preferences** from the IDE menu bar.

The **Preferences** dialog box appears.

2. Select the **Run/Debug** from the list in the left panel.

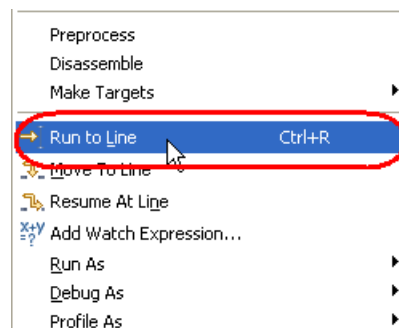
The **Run/Debug** page appears in the right panel.

3. Check the **Skip breakpoints during `Run to Line` operation** checkbox from the **General Settings for Running and Debugging** list in the right panel.



**Figure 7-11. Preference Dialog Box**

4. Start a debugging Session.
5. Select the desired source line in the Editor view.
6. Right-click and select **Run To Line** from the context menu. Alternatively, press **Ctrl +R**.



**Figure 7-12. Run to Line Context Menu**

With this you do not need to disable your hardware breakpoints in the **Breakpoints** view, and still can do full debugger stepping.

## 7.1.28 **NEW** What are the points to consider while using Stepping Backwards feature in Debugging?

The following points need to be considered while using Stepping Backwards, or **Move to Line** feature:

- This only changes the program counter. Any variable changes/etc are not affected or reverted.
- In case of highly optimized code, there might be multiple sequence points per source line. So doing this for highly optimized code might not work correctly.
- It works ok within a function. It is not recommended to use it, for example, to set the PC outside of a function, because the context/stack frame is not set up.

### Tip

You can use the **Move to Line** feature frequently to *advance* the program execution. For example, to bypass some long sequences, or to get out of an *endless* loop.

## 7.2 Variables

In this topic, FAQs related to variables are listed.

- [By default, the debugger shows me the global variables of the currently debugged source file. How can I view my variables from another module?](#)
- [How do I debug my application using global variables?](#)
- [How do I filter variables in files?](#)
- [How do I enable watch on a variable?](#)
- [What is a static local variable?](#)
- [When I enable watch on a static variable, it shows the error, "<error\(s\)\\_during\\_the\\_evaluation>". What do I do?](#)

## 7.2.1 By default, the debugger shows me the global variables of the currently debugged source file. How can I view my variables from another module?

While debugging, the debugger lists the local and global variables of the *currentthread* and stack frame only in the **Variables** view. To view the variables from other modules, you have to debug your application using global variables.

## 7.2.2 How do I debug my application using global variables?

You can debug your application using the variables outside of the current thread or stackframe by adding global variables from the **Variables** view.

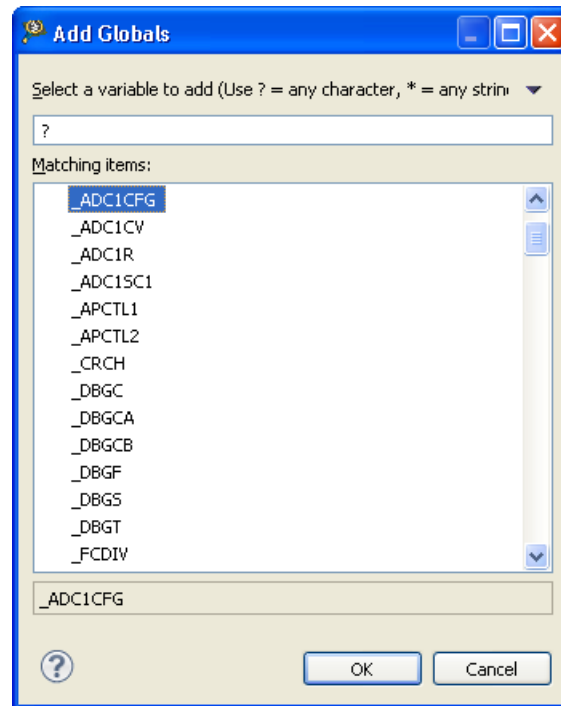
To add a global variable:

1. Start a debugging session.
2. The IDE switches to the **Debug** perspective.
3. Click the **Add Global Variables** button from the **Variables** view toolbar.  
Alternatively, right-click in the **Variables** view and select **Add Global Variables** option from the context menu.

The **Add Globals** dialog box appears listing all global variables present in your application.

### Tip

Select **Window > Show View > Variables** from the IDE menu bar to open the **Variables** view.



**Figure 7-13. Add Globals Dialog Box**

4. Enter the variable name in the text box, or you can enter a filter string and select the desired variable from the **Matching items** list.
5. Click **OK**.

The selected variable appears in the **Variables** view.

### 7.2.3 How do I filter variables in files?

To filter the variables in files:

1. Click the **Add Global Variables** button from the **Variables** view toolbar.  
The **Add Globals** dialog box appears.
2. Use the `<variableName>#<fileName>` format in the text box to filter the file.

The desired variable appears in the **Matching items** list.

### 7.2.4 How do I enable watch on a variable?

To enable **Watch** on a variable:

1. Select the variable for which you want to enable watch in the **Variables** view.
2. Right-click and select **Watch** from the context menu.



The **Expressions** view appears listing the selected variable.

## 7.2.5 What is a static local variable?

The static local variables are like global variables, but with the function scope.

## 7.2.6 When I enable watch on a static variable, it shows the error, "<error(s)\_during\_the\_evaluation>". What do I do?

To enable **Watch** on the *static* variables:

1. Click on the static variable name with error in the **Expressions** view's **Name** column.

The editing mode is enabled for the selected variable.

2. Edit the variable name to this format, "`FileName>::variableName`".

For example:

- For `static_global` variable in the `Application.c` file, enter the expression, `"static_global>::Application.c"`
- For `static_local` variable in the `Application.c` file, enter the expression, `"&static_global>::Application.c"`

The watch is now enabled for the edited static variable.

## 7.3 Build Configuration

In this topic, FAQs related to build configuration are listed.

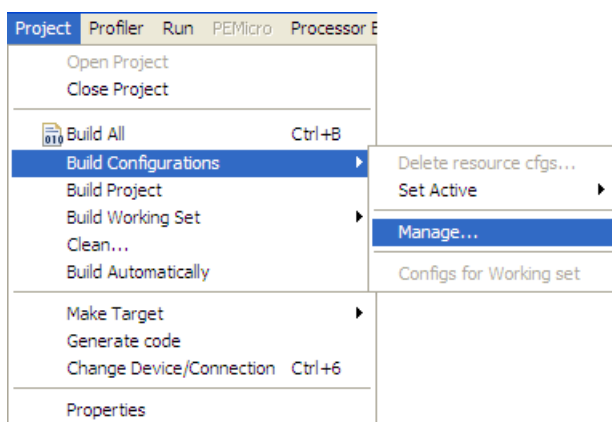
- [How do I create a new build configuration for my project?](#)
- [Is it possible to exclude a single or multiple files from the project while building it?](#)
- [Is it possible to configure build settings for a particular source code file?](#)
- [Is it possible to store output of the build configuration of my project in a separate directory?](#)
- [How can I disable the option of having the CodeWarrior IDE build my project before launching it?](#)

- How can I export the launch configurations settings and then import them back into the project?
- How do I ensure that the launch configurations are a part of the project when it is exported?

### 7.3.1 How do I create a new build configuration for my project?

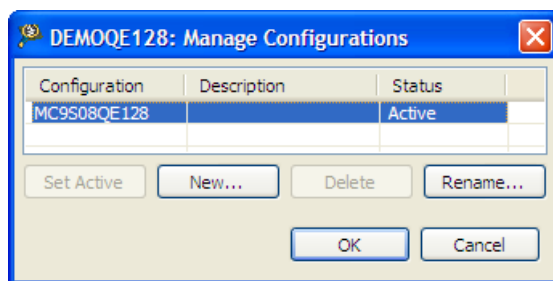
To create a new build configuration:

1. Select the project for which you want to create the build configuration in the **CodeWarrior Projects** view.
2. From the IDE menu bar, select **Project > Build Configurations > Manage** .



**Figure 7-14. Project > Build > Configuration**

3. The **Manage Configurations** dialog box appears.



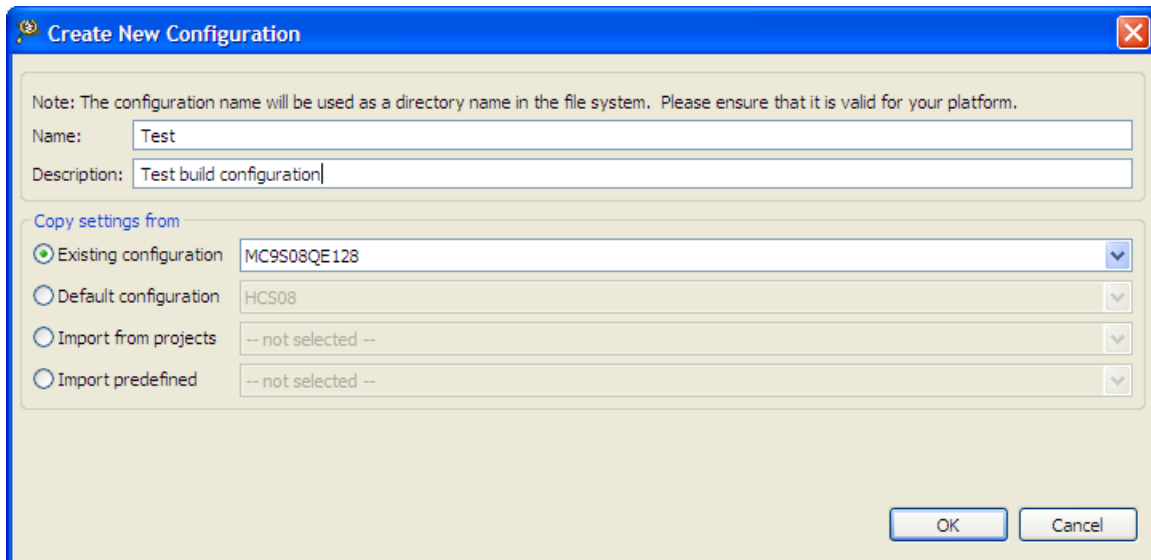
**Figure 7-15. Manage Configurations Dialog Box**

4. Click the **New** button to create a new build configuration.

The **Create New Configuration** dialog box appears.

5. Enter the name and description for the build configuration.

6. Select the appropriate option from the **Copy settings from** section and select the required option. For example, you may choose to copy settings from an existing configuration and select the required configuration.



**Figure 7-16. Create New Configuration Dialog Box**

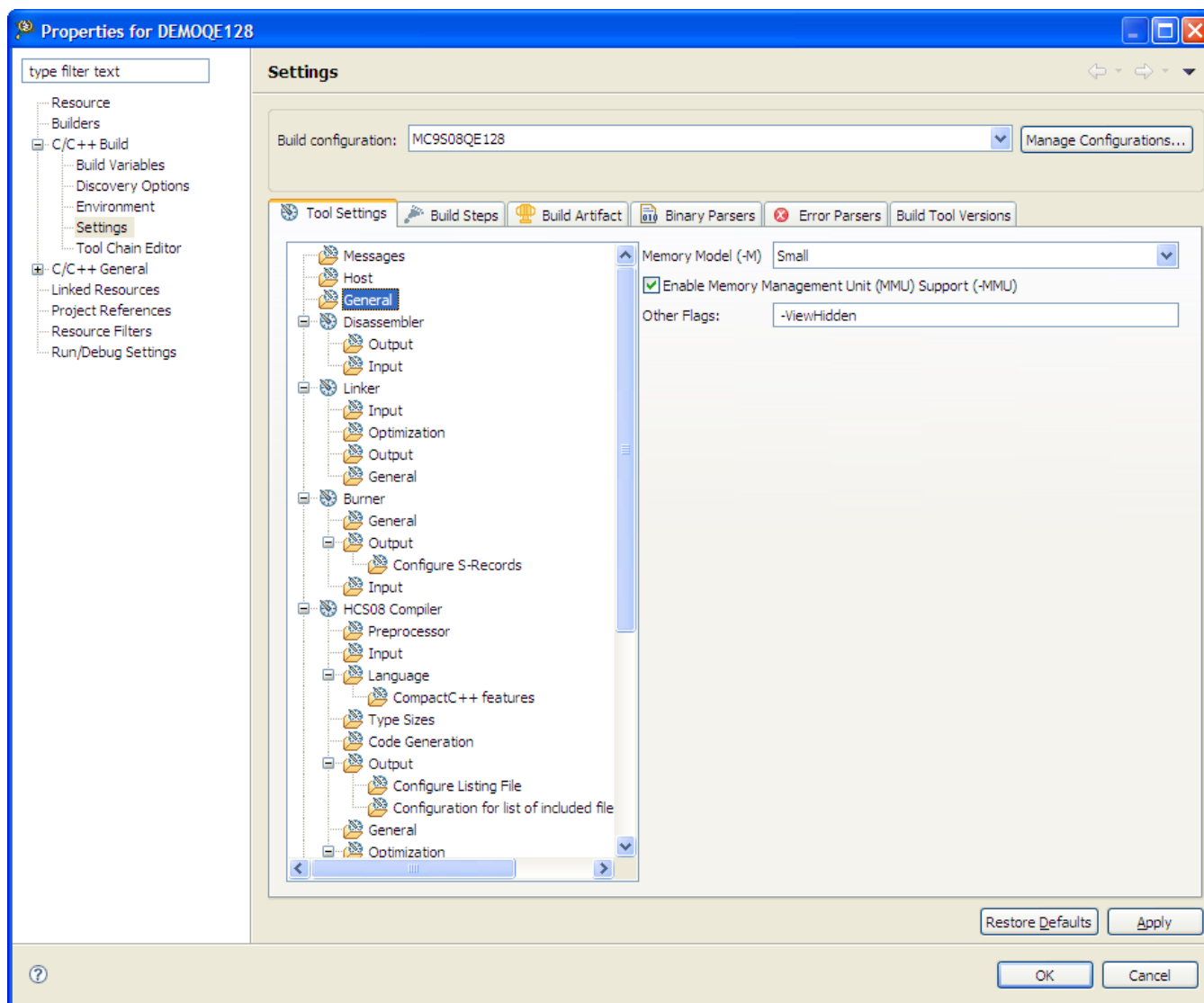
7. Click **OK** to close the **Create New Configuration** dialog box.
8. Click **OK** .

To configure build settings of the newly created build configuration:

1. Select the project with which the build configuration is associated in the **CodeWarrior Projects** view.
2. From the IDE menu bar, select **Project > Properties** .

The **Properties** dialog box appears.

3. Select **C/C++ Build > Settings** .
4. Select the newly created configuration from the **Build Configuration** drop-down list.



**Figure 7-17. Properties Dialog Box**

5. Specify the settings for the selected build configuration, as required. For example, you can define preprocessor directives in the *<target>Compiler > Preprocessor* page.
6. Click **OK**.

The build settings are configured, as specified.

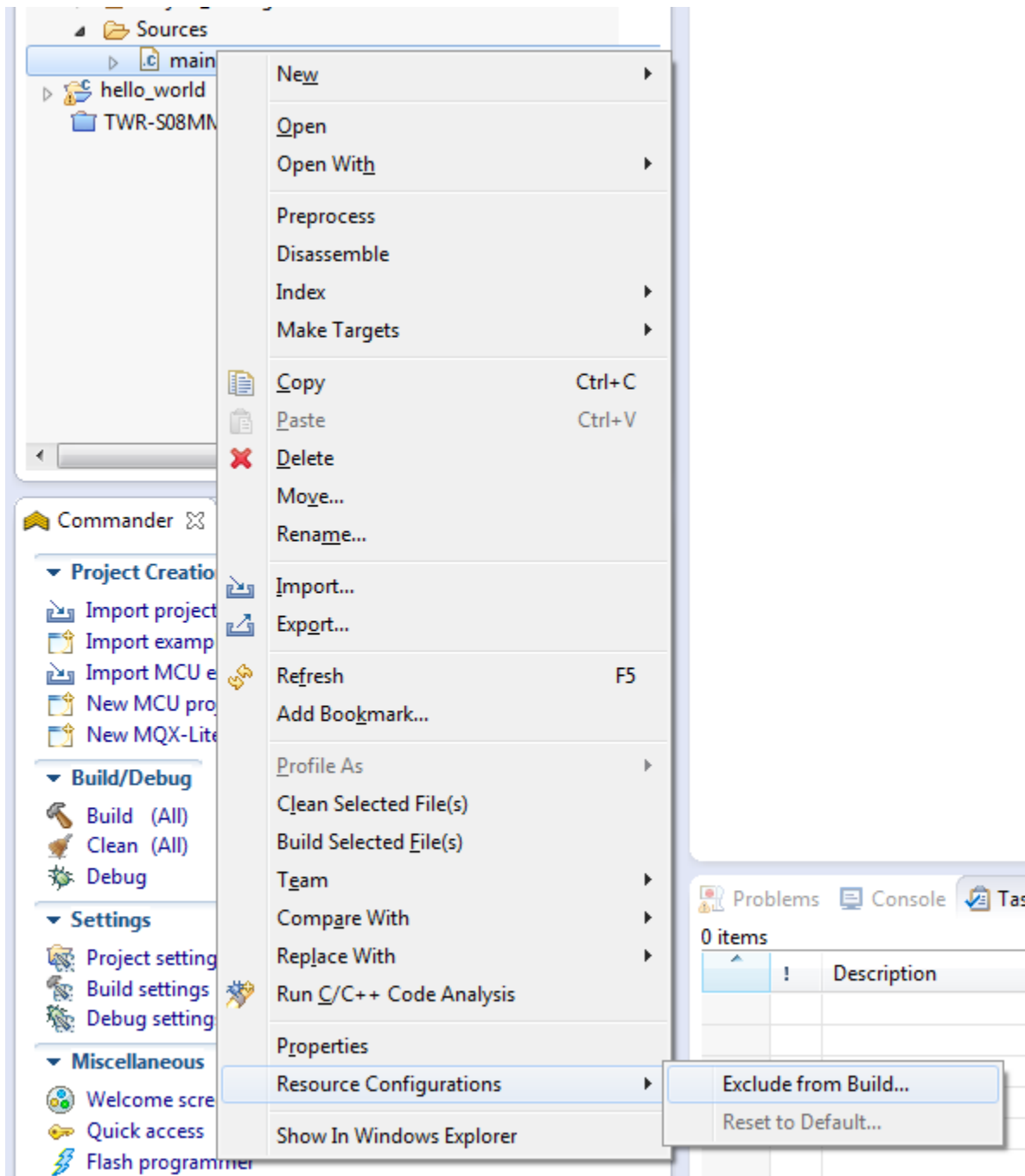
### 7.3.2 Is it possible to exclude a single or multiple files from the project while building it?

Yes, you can exclude single or multiple files from build:

1. Select the file in the **CodeWarrior Projects** view. To select multiple files, press and hold the **Ctrl** key.

- Right-click and select **Resource Configurations > Exclude from build** (for more information, refer to the image listed below).

The **Exclude from build** dialog box appears.



**Figure 7-18. Exclude from build**

- Select the build configuration from which you want to exclude the selected file.

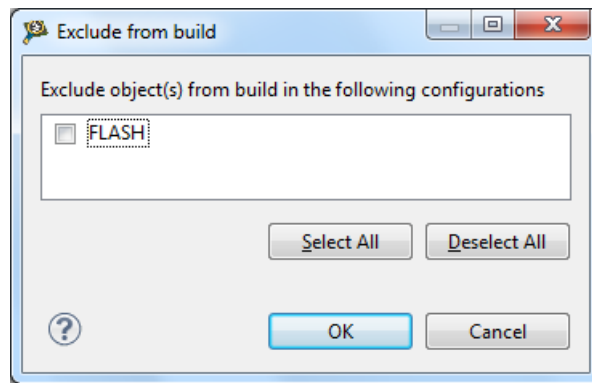


Figure 7-19. Exclude from build Dialog Box

4. Click **OK**.

The selected files are excluded from the build configuration.

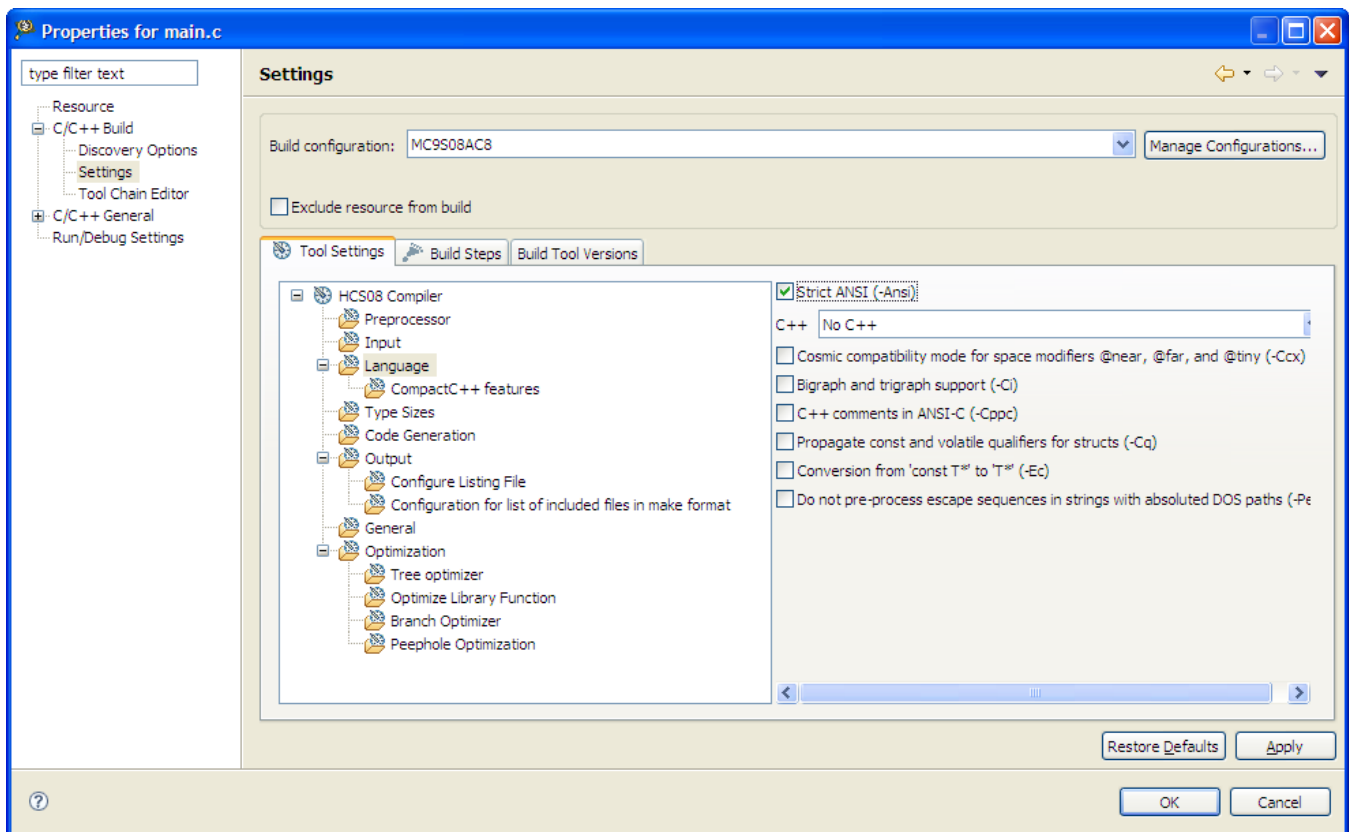
### 7.3.3 Is it possible to configure build settings for a particular source code file?

Yes, to set file specific build options:

1. Right-click the file in the **CodeWarrior Projects** view and select **Properties** .

The **Properties** dialog box for the selected file appears.

2. Specify the required settings. For example, you can select a specific compiler option for compiling the file.



**Figure 7-20. Properties Dialog Box for Selected File**

3. Click **OK** .

The file specific settings are configured, as specified.

### 7.3.4 Is it possible to store output of the build configuration of my project in a separate directory?

Yes, you can specify output directory for the build configuration.

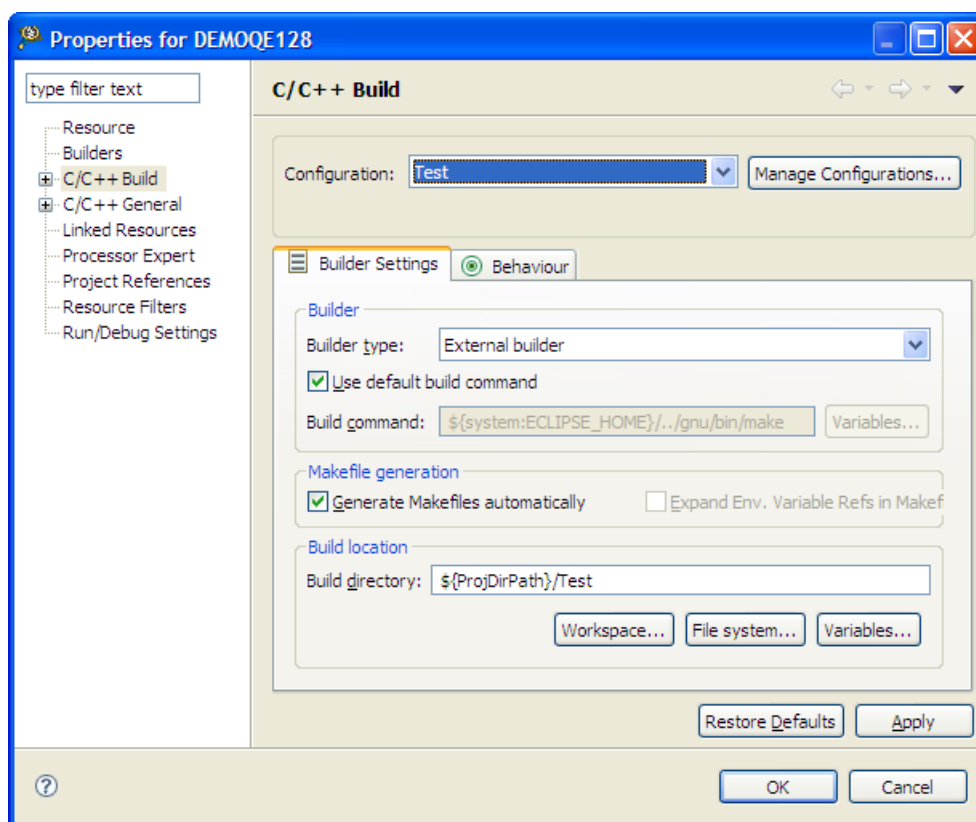
1. Select the project with which the build configuration is associated in the **CodeWarrior Projects** view.
2. From the IDE menu bar, select **Project > Properties** .

The **Properties** dialog box appears.

3. Select **C/C++ Build** .

The **C/C++ Build** page appears in the right panel of the **Properties** dialog box.

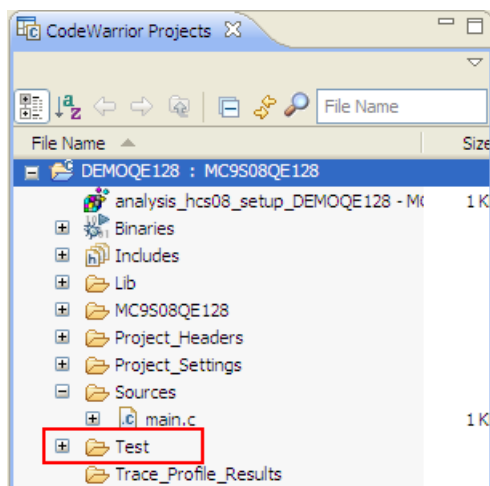
4. Select the build configuration from the **Configuration** drop-down list.
5. Specify the required build directory in the **Build directory** text box.



**Figure 7-21. Specify Build Output Directory**

6. Click **OK** .
7. Right-click the project and select **Build Configurations > Build > Select** .  
The **Build configurations** dialog box appears.
8. Select the build configuration to build and click **OK** .

The project build begins and the output is stored in the specified directory.



**Figure 7-22. Build Output Stored in Specified Directory**



### 7.3.5 How can I disable the option of having the CodeWarrior IDE build my project before launching it?

To disable build before launch:

1. From the IDE menu bar, select **Window > Preferences** .  
The **Preferences** dialog box appears.
2. Expand the tree control to select **Run/Debug > Launching** .  
The **Launching** page appears in the right pane.
3. Uncheck the **Build (if required) before launching** checkbox.
4. Click **Apply** .
5. Click **OK** .

Now, CodeWarrior IDE will not build the project before launching it.

### 7.3.6 How can I export the launch configurations settings and then import them back into the project?

You can export and import the launch configuration settings using the Export and Import wizards.

To export a launch configuration:

1. From the CodeWarrior menu bar, select **File > Export** .  
The **Export** dialog box appears.
2. Expand **Run/Debug** tree control and select **Launch Configurations** .  
The **Export Launch Configurations** wizard appears.
3. Select the launch configuration you want to export.
4. In the **Location** field, specify the location where you want to export the launch configuration.
5. Click **Finish** .

The selected launch configuration exports as `.launch` file.

Similarly, you can import the launch configuration (.launch file) using **File > Import** in the CodeWarrior IDE menu bar.

Alternatively, you can copy the \*.launch files present in the `Project_Settings\Debugger` folder of a project and place it in the project where you want to export it.

### 7.3.7 How do I ensure that the launch configurations are a part of the project when it is exported?

To make a launch configurations be part of the project, follow steps below:

1. From the IDE menu bar, select **Run > Debug Configurations** .

The **Debug** window appears.

2. Under **CodeWarrior Download** , select **Launch Configurations** .
3. Select the **Common** tab.

The **Common** settings pane appears.

4. Under **Save as** , select the shared file radio button. Leave the default folder name as is.
5. Click **Apply** .

Now the launch configuration is physically located in the project folder, so it will be a part of the project when zipped.

## 7.4 Flash Programmer

In this topic:

- [How can I flash or download a binary file to the target?](#)
- [How can I create the flash programmer, hardware diagnostics, or import/export fill memory tasks in the CodeWarrior IDE?](#)
- [I want to download my application directly into a RAM target. How do I do it?](#)
- [I need to flash a large size application \(more than 64 KB\) , and I also need to perform target tasks, such as secure/unsecure the device and additional verify or erase. How do I do it?](#)
- [Is there a way to quickly flash multiple boards with different S-Records?](#)

- Is it possible to debug the target multiple times with the same binary application, without flashing the device?
- I need to enable logging for flash programming. How do I do it?
- How can I program my boards without a project?

### 7.4.1 How can I flash or download a binary file to the target?

You can flash or download a binary file to the target using flash programmer. To flash a binary file, for example a `s19` file, first you need to create a project for the target to which you want to flash the file. For example, to flash a `s19` file to the S08JM60 target using the P&E cable:

1. Create a project for the S08JM60 target using P&E as the target connection.

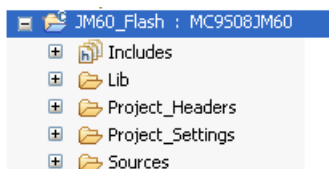



Figure 7-23. S08JM60 Project

2. Click the  icon next to the **Flash Programmer** icon on the CodeWarrior IDE toolbar  
 A drop-down menu appears.
3. Select **Flash File to Target**.
4. Specify the run configuration and flash configuration in the dialog box that appears.
5. Specify the binary file that you want to flash to the target in the **File** text box.
6. Click the **Erase** button to erase the device.
7. Click the **Program with Erase** button to erase and flash the selected `s19` file to the target.
8. Click **Close** to close the dialog box.

The selected `s19` file is downloaded to the target.

### 7.4.2 How can I create the flash programmer, hardware diagnostics, or import/export fill memory tasks in the CodeWarrior IDE?

CodeWarrior IDE provides a framework, **Target Task** view, to create tasks to:

- flash, erase, verify a device,
- perform hardware diagnostic, or
- read, write, or export memory.

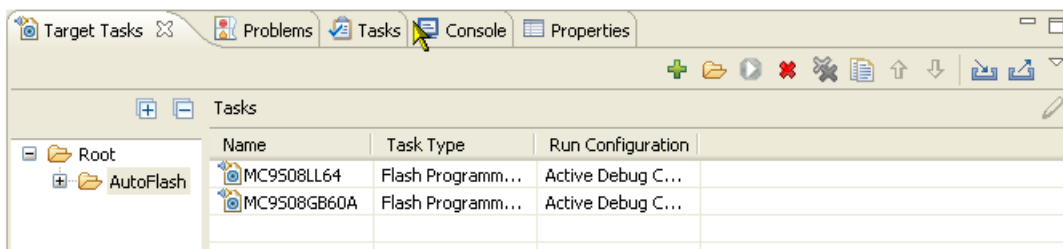
To open the **Target Task** view:

1. Select **Window > Show View > Other**.

The **Show View** dialog box appears.

2. Type "Target Taks" view as the filter text to narrow down the list of the views in the dialog box.
3. Select **Debug > Target Tasks** and click **OK**.

The **Target Tasks** view opens in the Workbench window.



**Figure 7-24. Target Tasks View**

Typically, the target tasks are used in combination with the debugger, as shown in the figure listed above. However, you can also run a target task outside a debugging session, such as to batch program devices.

### 7.4.3 I want to download my application directly into a RAM target. How do I do it?

**CAUTION**

This method is not applicable to the ColdFire V2-V4 targets.

**NOTE**

To download the application to RAM, you need to set up the linker configuration file.

You can let debugger program the flash as a part of the debug action. This method of flashing is especially useful for small size (less than 64 KB) applications.

All P&E connections (Multilink, Cyclone, OSJTAG, and OSBDM) in Microcontrollers can handle flash programming. For this, the P&E cable itself will perform the flash programming directly. The process followed is:

1. The debugger downloads the application to the P&E connection cable.
2. The P&E connection cable directly programs the target.

If you have your target in RAM, this feature of P&E connections lets debugger write the application bytes to the P&E connection, which are written directly to the target. This is because debugger can directly modify the RAM content.

To flash an application to the RAM target:

1. Select **Run > Debug Configurations** .

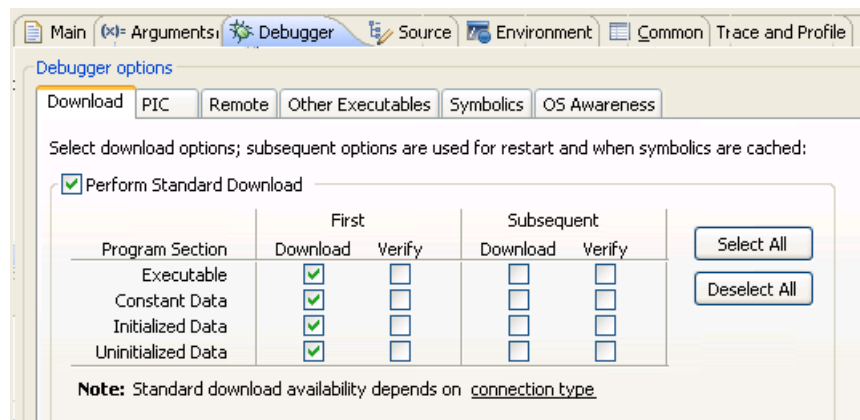
The **Debug Configurations** dialog box appears.

2. Select **Debugger > Download** .

The **Download** page appears in the **Debug Configuration** dialog box.

3. Check the **Perform Standard Download** checkbox.

As the P&E flash programming is performing the verify as well, you do not need to check the **Verify** checkboxes in the **Perform Standard Download** section.



**Figure 7-25. Download Page Configurations**

4. Click **Debug** .

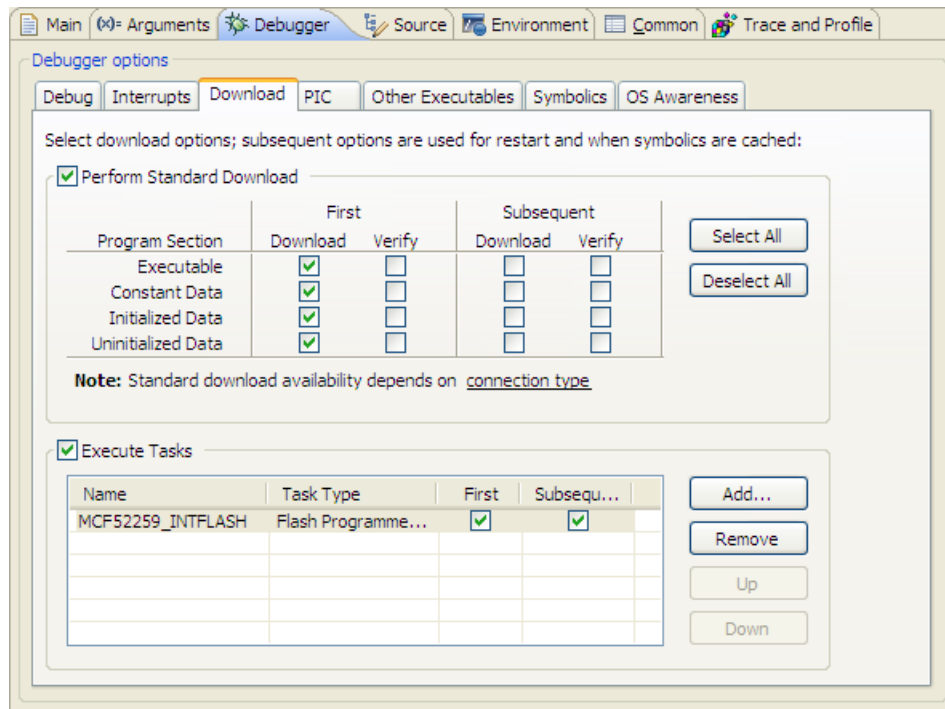
So, just by debugging your project, you can flash your RAM targets.

### 7.4.4 I need to flash a large size application (more than 64 KB) , and I also need to perform target tasks, such as secure/

## unsecure the device and additional verify or erase. How do I do it?

You can have the debugger flash the target using a target task. The flash programming target tasks are small application applets, which are downloaded to the target by the debugger. The debugger gets the application binary from the host and programs the flash.

For all non-P&E connections and for the P&E connection for the ColdFire V2 and PowerPC, a flash programming task is setup in the **Run > Debug Configurations > Debugger > Download** tab.



**Figure 7-26. Flash Programming Target Task Associated with Debugger**

This means that the debugger is using the attached target task to be executed as part of the download.

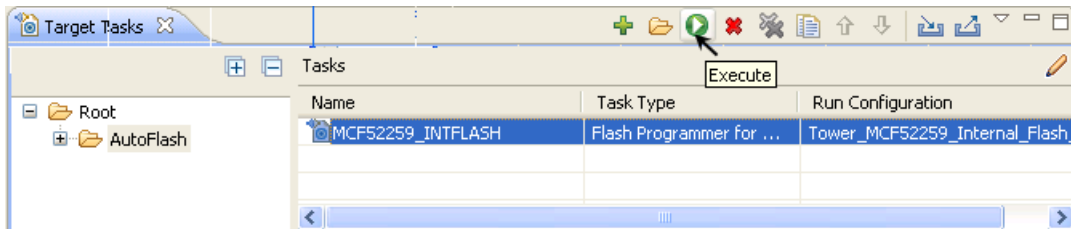
However, you can also perform flash programming as a standalone task, without combining it with the debug task.

1. Select **Flash Programmer > Open Flash Programmer** on the CodeWarrior IDE tool bar.

The **Target Tasks** view appears.

2. To create a new target task, either import a flash configuration file or duplicate an existing flash task.

3. Right-click the new target task and select **Change Run Configuration**.
4. Change the run configuration from **Active Debug Context** to a defined launch configuration.
5. Click the **Execute** button to execute the target task.



**Figure 7-27. Target Tasks View**

When the **MCF52259\_FlashIt** task created in the steps above is executed, the **Console** view shows the following results.

```

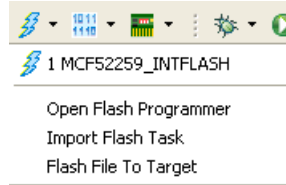
Flash Programmer Console
Program Command Succeeded
Downloading 0x00000018 bytes to be programmed at 0x00000400
Programming ....
Program Command Succeeded
Downloading 0x0000199C bytes to be programmed at 0x00000420
Programming ....
Program Command Succeeded
cmdwin::fl::verify
Beginning Operation ...
-----
Verifying file C:\Users\Erich Styger\Data\HTA\CW\wsp_Intro\Tow
Initializing remote connection ...
Performing target initialization ...
Auto-detection is successful.
    File is of type Elf Format.

Reading 0x00000300 bytes starting from address 0x00000000
Reading 0x00000018 bytes starting from address 0x00000400
Reading 0x0000199C bytes starting from address 0x00000420
Verify Command Succeeded
    
```

**Figure 7-28. Console View**

You can use this information to diagnose issues with the task execution, if there are any.

You can also access the previously executed target task directly from the **Flash Programmer** button on the CodeWarrior IDE toolbar.



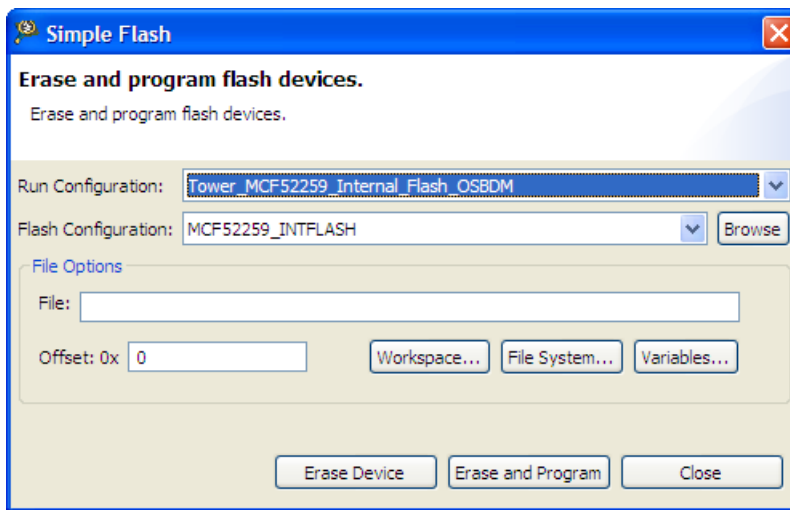
**Figure 7-29. Flash Programmer Button**

### 7.4.5 Is there a way to quickly flash multiple boards with different S-Records?

You can flash the target without the target task.

1. From the CodeWarrior IDE toolbar, select **Flash Programmer > Flash File to Target** .

The **Flash File to Target** dialog box appears.



**Figure 7-30. Flash File to Target**

2. Specify the run configuration that defines the connection, the flash configuration, and the file you want to flash.
3. Click the required button, **Erase Device** or **Erase and Program** .

The diagnostic information is logged to the **Console** View. This helps you view the flash results in detail when flashing large applications.



## 7.4.6 Is it possible to debug the target multiple times with the same binary application, without flashing the device?

Yes, you can debug the target multiple times with the same application.

1. Flash the target once with either using a target task (standalone) or using **Flash Programmer > Flash File to Target** .

### NOTE

For information about flashing using target task and flash file to target, refer [I need to flash a large size application \(more than 64 KB\)](#) , and [I also need to perform target tasks, such as secure/unsecure the device and additional verify or erase. How do I do it?](#) and [Is there a way to quickly flash multiple boards with different S-Records?](#).

2. Select **Run > Debug Configurations** .

The **Debug Configurations** dialog box appears.

3. Select **Debugger > Download** .

The **Download** page appears in the **Debug Configuration** dialog box.

4. Clear the **Perform Standard Download** and **Execute Tasks** checkboxes.

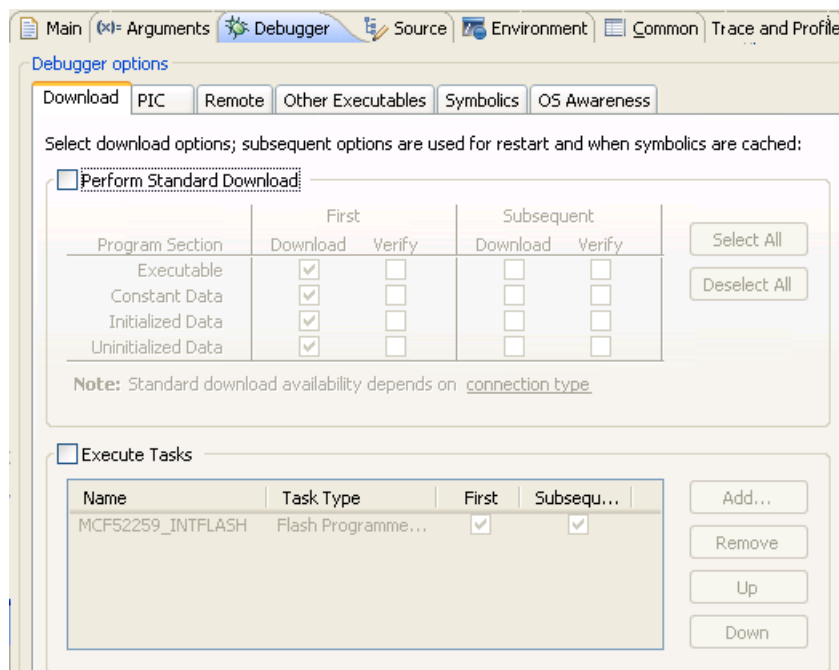


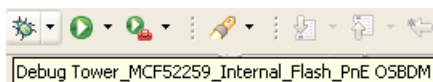
Figure 7-31. Download Page - Debug Configurations Dialog Box

Now as the debugger does not need to flash a potentially large application, you can start debugging within a few seconds, and you can do this multiple times.

5. To flash the target, click the **Flash Programmer** button followed by the **Debug** button on the CodeWarrior IDE toolbar.



**Figure 7-32. Flash Programmer Button**



**Figure 7-33. Debug Button**

Now, you can debug the target multiple times with the same binary application, without flashing the device

### 7.4.7 I need to enable logging for flash programming. How do I do it?

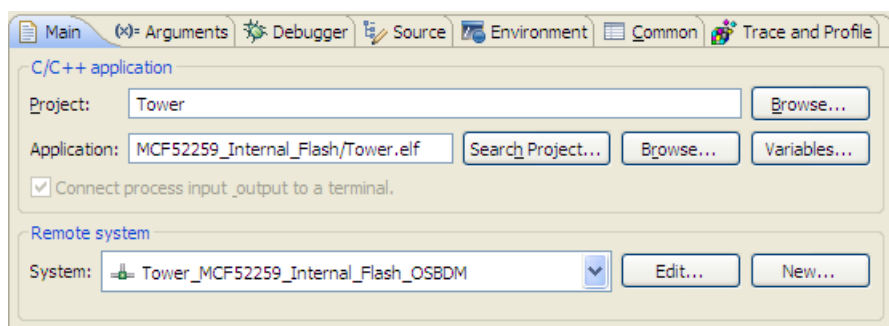
Various issues may occur while flash programming, such as the target is not stable, you might have problems with the target power, or the part might be secured.

To gather more information about such issues, you can enable logging.

1. From the CodeWarrior IDE menu bar, select **Run > Debug Configurations** .

The **Debug Configurations** dialog box appears.

2. Click the **Edit** button in the **Remote system** section.

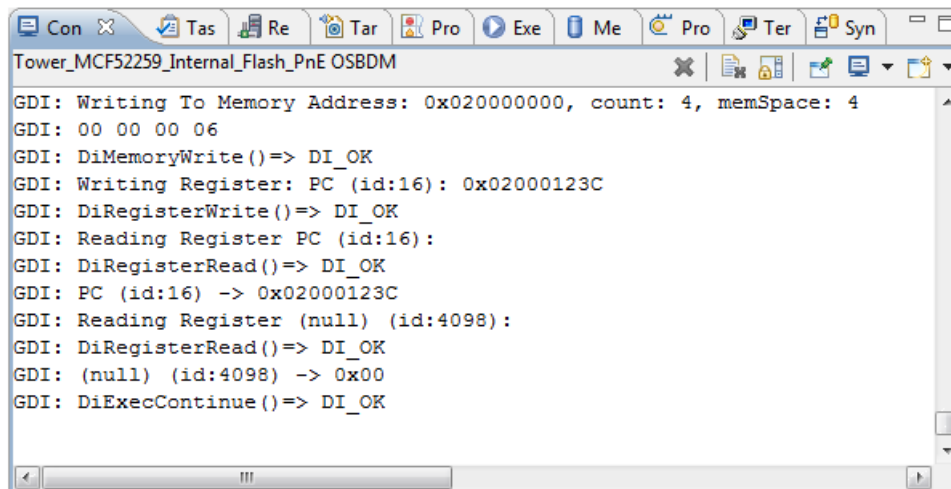


**Figure 7-34. Remote System**

The **Properties for <project\_name>** dialog box appears.

3. Check the **Enable Logging** checkbox in the **Connection** page.
4. Click **OK**.
5. Click **Debug**.

The log results appear in the **Console** view.



```
Tower_MCF52259_Internal_Flash_PnE OSBDM
GDI: Writing To Memory Address: 0x02000000, count: 4, memSpace: 4
GDI: 00 00 00 06
GDI: DiMemoryWrite()=> DI_OK
GDI: Writing Register: PC (id:16): 0x02000123C
GDI: DiRegisterWrite()=> DI_OK
GDI: Reading Register PC (id:16):
GDI: DiRegisterRead()=> DI_OK
GDI: PC (id:16) -> 0x02000123C
GDI: Reading Register (null) (id:4098):
GDI: DiRegisterRead()=> DI_OK
GDI: (null) (id:4098) -> 0x00
GDI: DiExecContinue()=> DI_OK
```

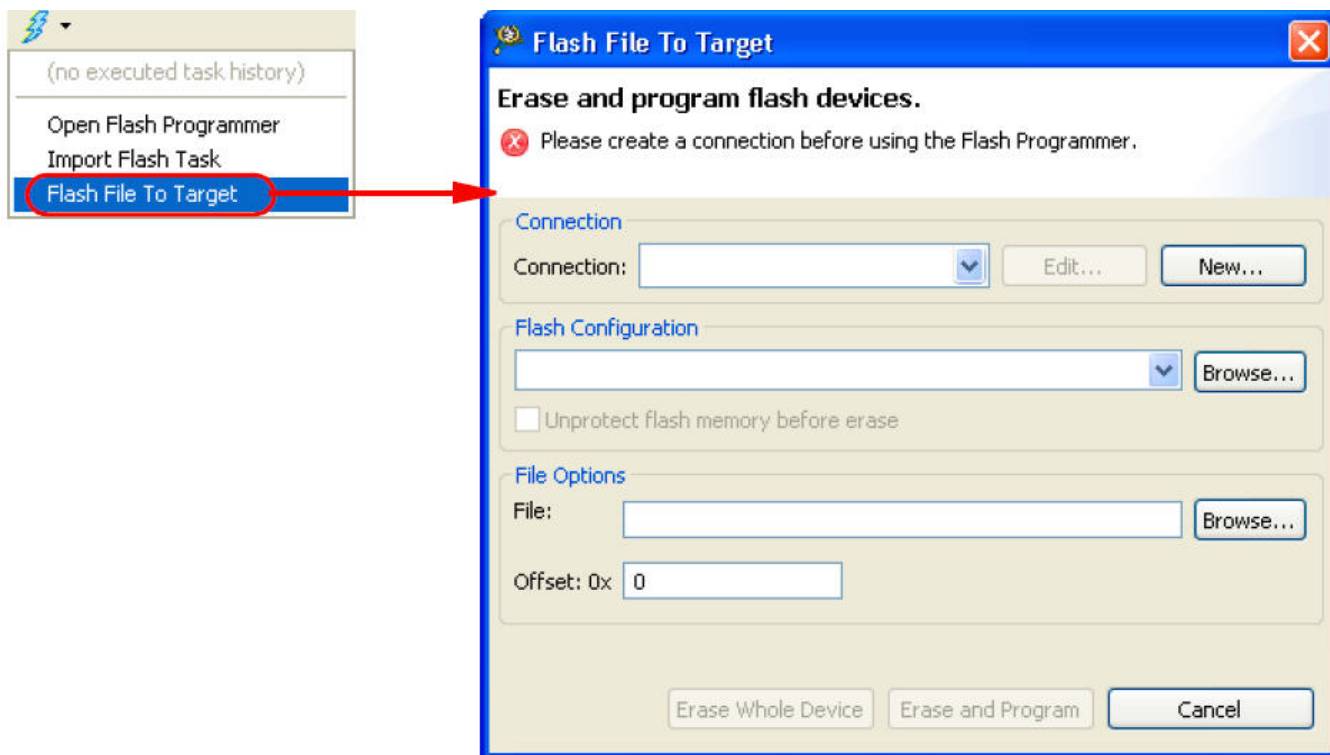
Figure 7-35. Console View

## 7.4.8 How can I program my boards without a project?

You can program your boards without a project by using the **Flash File To Target** feature. To use this feature you need to have the binary file that you want to flash for your device. The following steps lists the procedure to program the boards without a project:

1. Click the *downward arrow key* of the **Flash Programmer** button from the IDE menu bar and select **Flash File To Target** option.

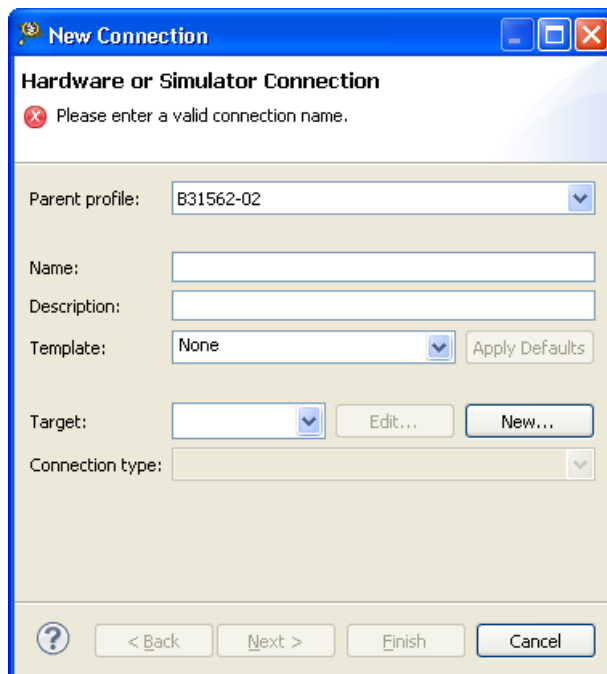
The **Flash File To Target** dialog box appears.



**Figure 7-36. Flash Programmer Button Menu**

2. Click **New** from the **Connection** group.

The **New Connection** dialog box with **Hardware or Simulator Connection** page appears.

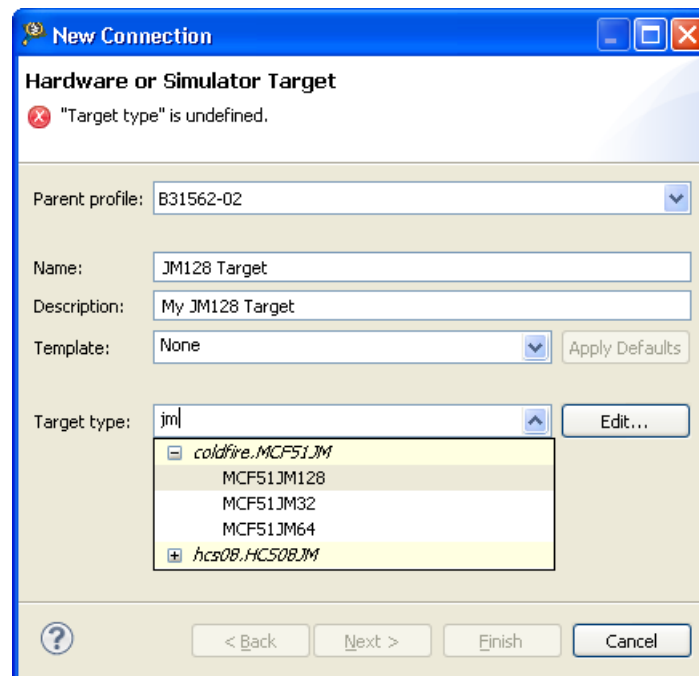


**Figure 7-37. New Connection - Hardware or Simulator Connection Page**

3. Enter the name and description in the **Name** and **Description** textboxes.

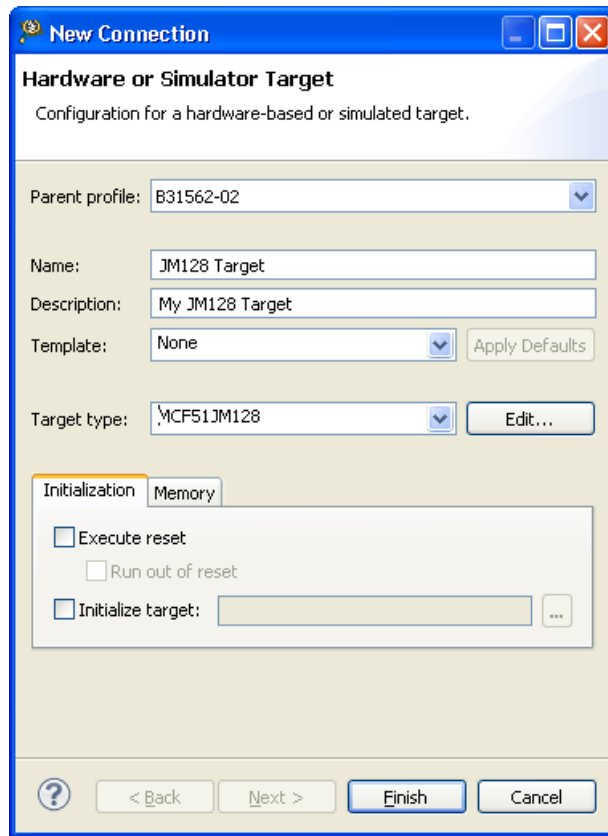
- Click **New** in the **Target** group to add a new target.

The **New Connection** dialog box with **Hardware or Simulator Target** page appears.



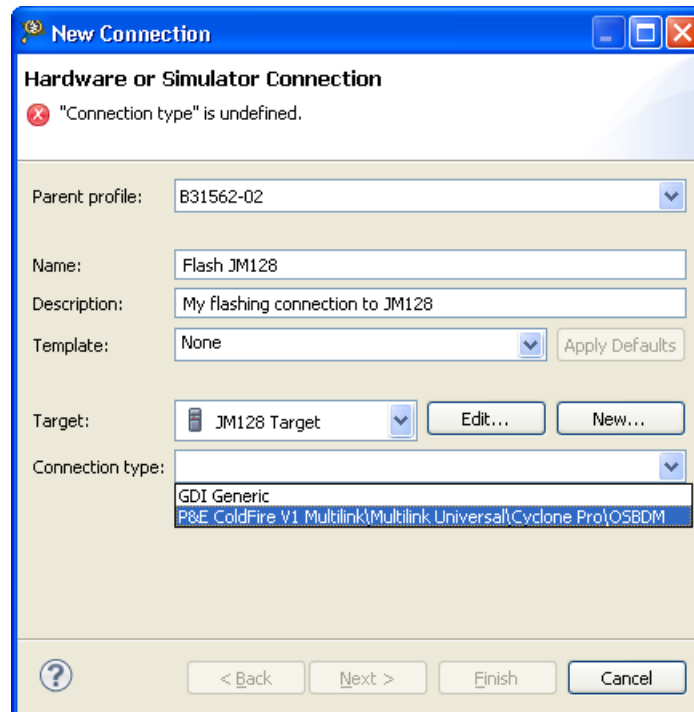
**Figure 7-38. New Connection - Create Hardware or Simulator Target Page**

- Enter the name and description in the **Name** and **Description** textboxes.
- Select the desired target from the **Target type** drop-down list.



**Figure 7-39. New Connection - Create Hardware or Simulator Target Page**

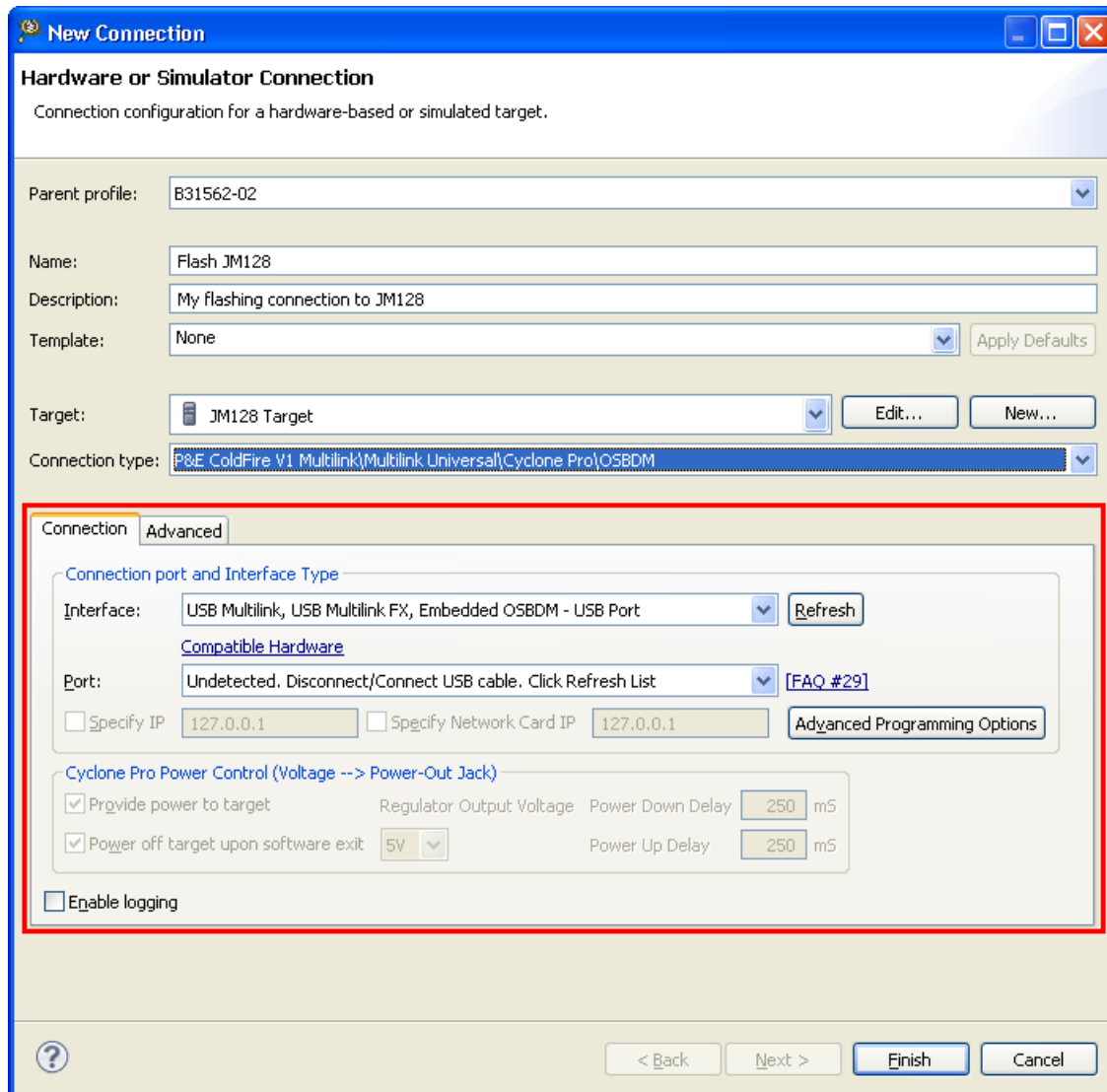
7. Click **Finish** . The **Hardware or Simulator Target** page closes and the **Hardware or Simulator Connection** page appears.



**Figure 7-40. Selecting Connection Type**

8. Select the desired connection from the **Connection type** drop down list.

This expands the dialog for the run control specific settings, as the following figure shows:

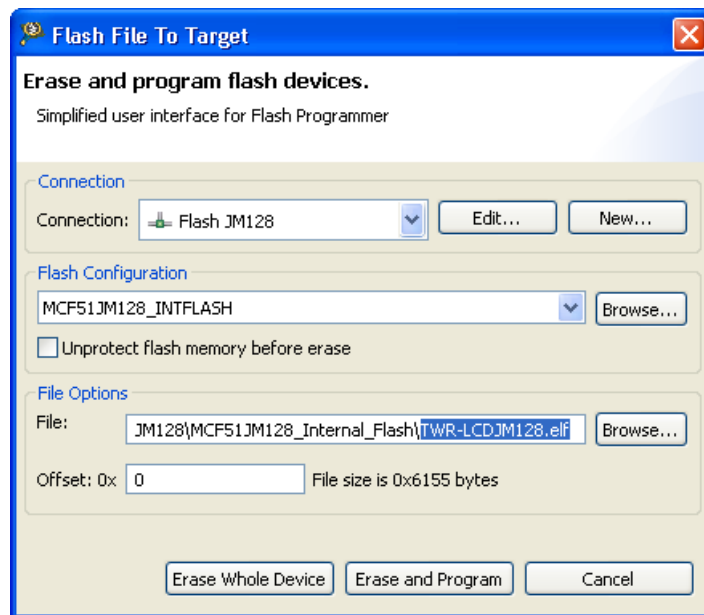


**Figure 7-41. New Connection Dialog Box after Selecting Connection Type**

9. Click **Finish** .

The **New Connection** dialog box closes and the **Flash File To Target** dialog box appears.

10. From the **File Options** group, click **Browse** and select the desired binary file to be programmed on the board.



**Figure 7-42. Flash File To Target - Selecting Binary File**

11. Select the **Erase and Program** button to mass erase your device and program it with the file.

Your board is programmed with the binary file you selected without accessing a project.

## 7.5 Miscellaneous

In this topic, miscellaneous FAQs related to the debugger are listed.

- [How can I view only certain registers?](#)
- [How can I view addresses in memory?](#)
- [How do I open up multiple memory locations in the memory view?](#)
- [How can I view variables that I have set in my project?](#)
- [How can I change the values of variables?](#)
- [How do I view global variables that have been accessed in my project?](#)
- [Is the command-line debugger available in the Eclipse-based CodeWarrior IDE as well?](#)
- [Is it possible to specify a different name for the binary file \(.abs\)?](#)
- [Can I debug the code from the generated assembly file step-by-step?](#)
- [If I have a debugger session running on my application, how can I halt the application?](#)
- [Why do I get the following error message when I attempt to debug my project?](#)
- [Can I change the format of the Memory view?](#)



- Why does the DE.exe appear in the Windows Task Manager even after terminating and closing the debugger perspective?
- Is there a way to display the variables always in the Hexadecimal format?
- Can I change the endianness that is displayed in the Memory view?
- In the Memory view, is it possible to go back to the address that the rendering was created for?
- Is there a way to view disassembly at any location?
- Is it possible to display the expressions always in the Hexadecimal format instead of the default decimal format?
- What does the Reset on Connect option do?
- Can I change the Program Counter (pc) value while in the debugger without having to restart the debugger?
- How do I specify the program entry point?
- How can I run a debug session repeatedly?
- Can I direct console output to a file instead of the console window?
- How do I resolve the following error message that I get when I use Ctrl-Click to open variable declaration?
- How can I configure and launch an external tool or debugger?
- How can I monitor a variable in the debugger while a target is running?
- How can I inspect global variables in the variables view?
- How do I resolve the error encountered on the Kwikstik board?
- At the moment serial data is output in the console in the debugger, which is slow with J-Link. Is there any way to view the output of the debugger in the HyperTerminal?
- Is it possible to flash a target MCU from the command-line?
- Is there an easy way to suppress the target re-programming when there has been no change since the previous session?
- The microcontrollers are usually trimmed in the factory. How can I trim it myself if I need something better?
- How do I get a full dump of my device memory for offline analysis?
- When I build my application I received the message, "Device is secure. Erase to unsecure?" What do I do?
- When I connect to my board, it shows a connection failure. What do I do?
- How do I debug the strings?
- I am debugging the strings using Variables view, but instead of the string, only the address of the first character or the location is shown in the preview panel. What do I do?
- How do I view the declared type of the variables in the Variables view?

## 7.5.1 How can I view only certain registers?

To view only certain registers, follow these steps:

1. Select **Window > Show View > Registers** from the IDE menu bar while in the debugger perspective.

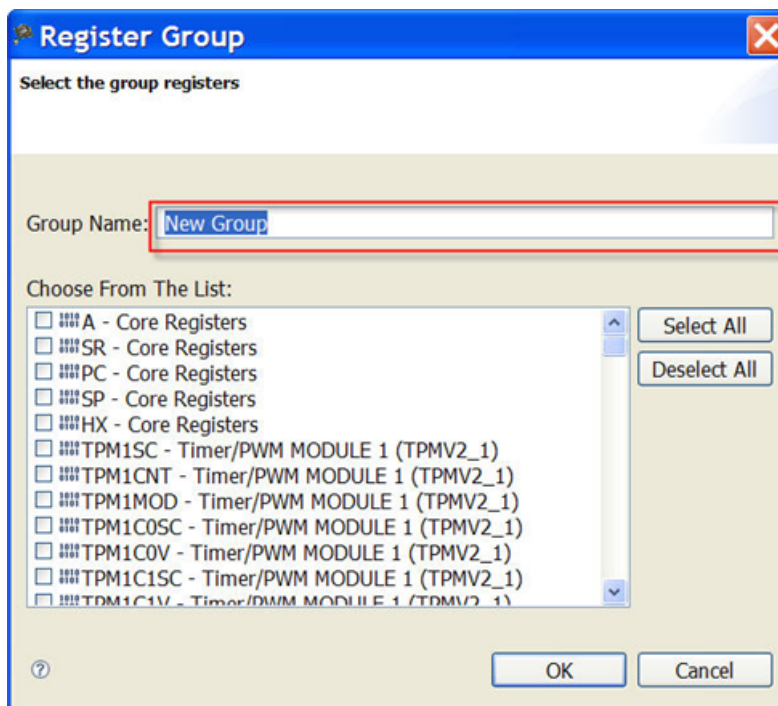
The **Registers** view appears docked to the upper right hand side of debugger perspective.

2. Right-click anywhere in the **Registers** tab.

A context menu appears.

3. Select Add Register Group in the context menu.

The **Register Group** window appears.



**Figure 7-43. Register Group Window**

4. Type the name of new register group in the **Group Name** textbox.
5. Select the checkbox next to registers that you want to be a part of this group.
6. Click **OK** .

The **Register Group** window closes.


Now you can view only the selected registers.

## 7.5.2 How can I view addresses in memory?

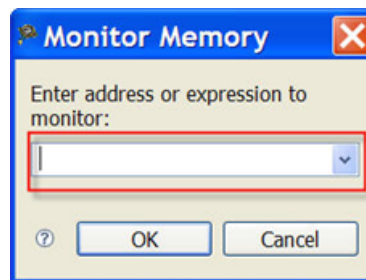
To view addresses in memory, follow these steps:

1. Select **Window > Show View > Memory** in the debug perspective.

The **Memory** window appears docked to the lower part of debugger perspective.

2. Click  sign icon.

The **Monitor Memory** window appears.



**Figure 7-44. Monitor Memory Window**

3. Type memory address in the **Enter address or expression to monitor** textbox.
4. Click **OK**.

The **Monitor Memory** window closes.

The memory location appears in the memory window. The memory contents can be viewed by selecting memory on the left hand side, as shown in the following figure.

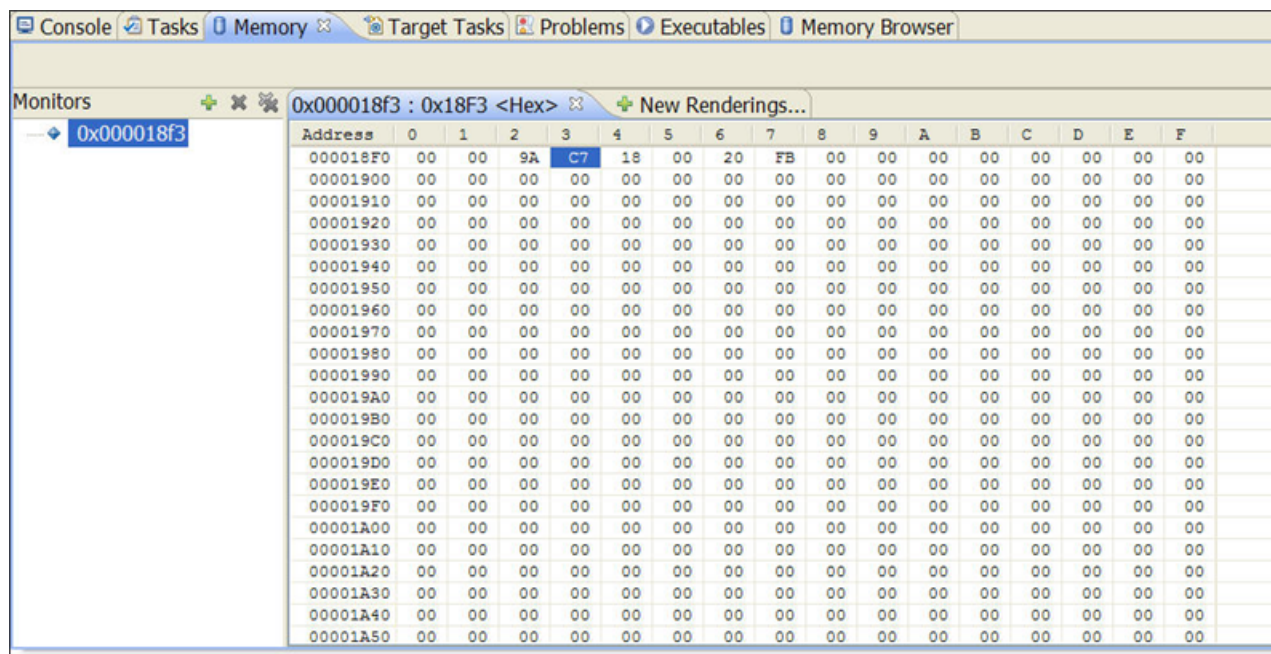


Figure 7-45. Memory View

### 7.5.3 How do I open up multiple memory locations in the memory view?

To open multiple memory locations in the memory view, follow these steps:

1. Select **Window > Show View > Memory** in the debug perspective.

The **Memory** window appears docked to the lower part of debugger perspective.

2. Click  sign icon.

The **Monitor Memory** window appears.

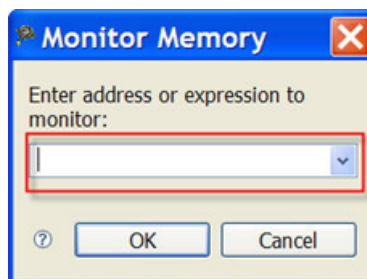



Figure 7-46. Monitor Memory Window

3. Type memory address in the **Enter address or expression to monitor** textbox.
4. Click **OK**.

The new address appears in the memory window.

5. Click  sign icon.
6. The **Monitor Memory** window appears.
7. Type another memory address in the **Enter address or expression to monitor** textbox.
8. Click **OK** .

The new address appears in the memory window. The contents can be viewed by selecting memory on the left hand side, as shown in the following figure.

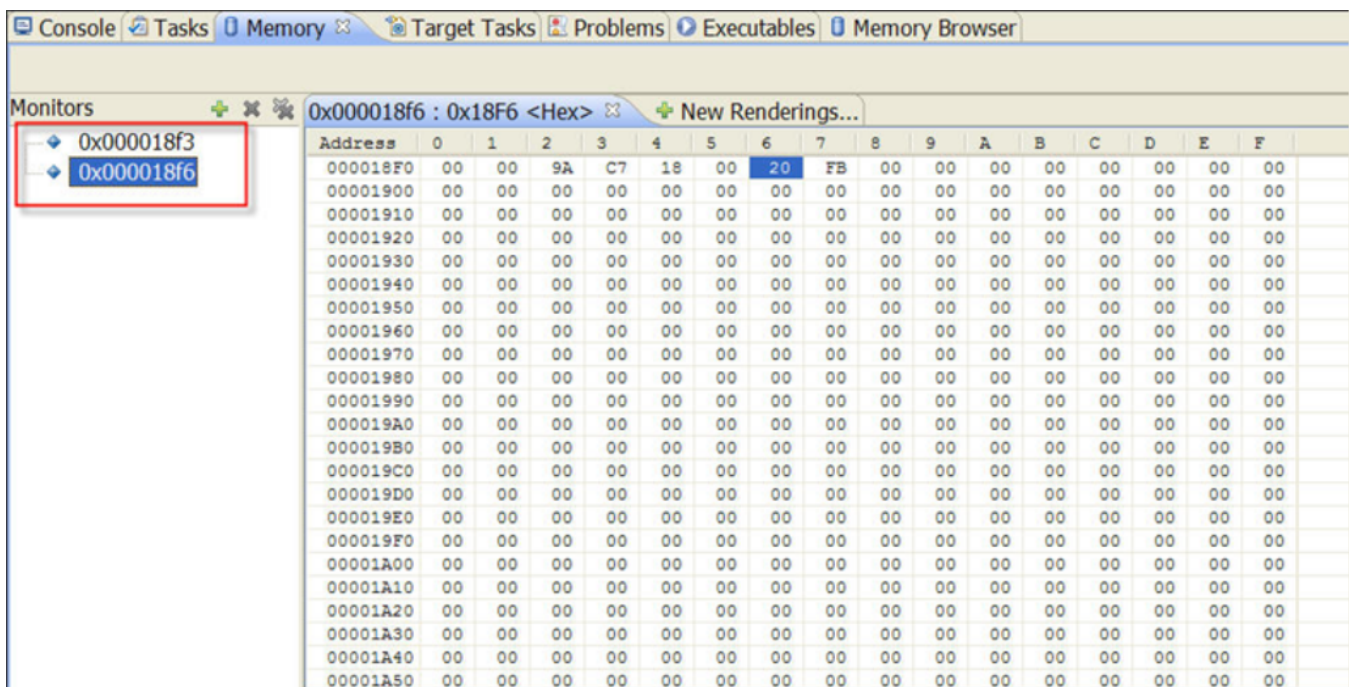


Figure 7-47. Memory View with Multiple Addresses

## 7.5.4 How can I view variables that I have set in my project?

To view variables from the debug perspective toolbar, select **Window > Show View > Variables** . The Variable window appears docked on the upper right hand side of debug perspective.

## 7.5.5 How can I change the values of variables?

To change the value of variables, follow the steps below in the debug perspective:

1. From the toolbar, select **Window > Show View > Variables** . The **Variables** window appears docked on the upper right hand side of debugger perspective.
2. Right-click on the variable.

A context menu appears.

3. From context menu, select **Change Value** .

The **Set Value** window appears.

4. In the **Enter a new value for variables** textbox, type a new variable value.
5. Click **OK** .

The **Set Value** window closes.

The variable value changes in the **Variables** window. The variable being changed is highlighted in yellow.

## 7.5.6 How do I view global variables that have been accessed in my project?

To view global variables that have been accessed in your project, follow these steps:

1. Select **Window > Show View > Variables** from the IDE menu bar while in the debugger perspective.

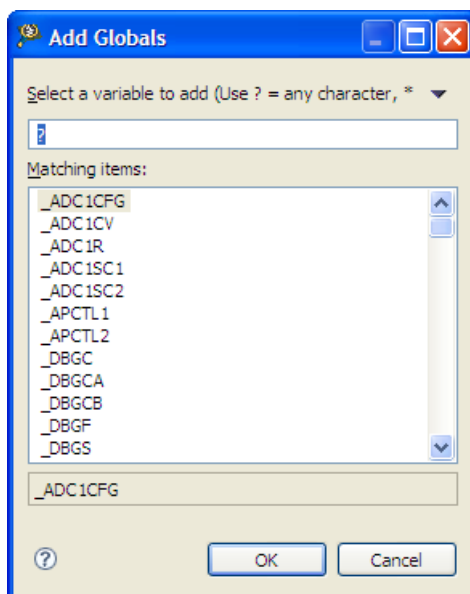
The **Variables** view appears docked to the upper right hand side of debugger perspective.

2. Right-click a variable.

A context menu appears.

3. Select **Add Global Variables** in the context menu.

The **Add Globals** dialog box appears.



**Figure 7-48. Add Globals Dialog Box**

4. Select the global variables that you want to view.
5. Click **OK** .

The **Add Globals** dialog box closes.

The selected variables appear in the **Variables** view with the symbol that identifies them as global variable.

#### **NOTE**

The selected global variables will persist throughout the session and subsequent sessions, until they are removed.

### **7.5.7 Is the command-line debugger available in the Eclipse-based CodeWarrior IDE as well?**

Yes. To use the Command Line Debugger follow steps given below in debug perspective:

1. From the IDE menu bar, select **Window > Show View > Debugger Shell** .
2. Click **OK** .

The Debugger Shell window appears docked on the lower right-hand corner of debug perspective.

## 7.5.8 Is it possible to specify a different name for the binary file (.abs)?

Yes this can be done by following steps below:

1. In the CodeWarrior Projects window, right-click the project name.

A context menu appears.

2. From the context menu, select **Properties**.

The **Properties** window appears.

3. Select **C/C++ Build > Settings**.

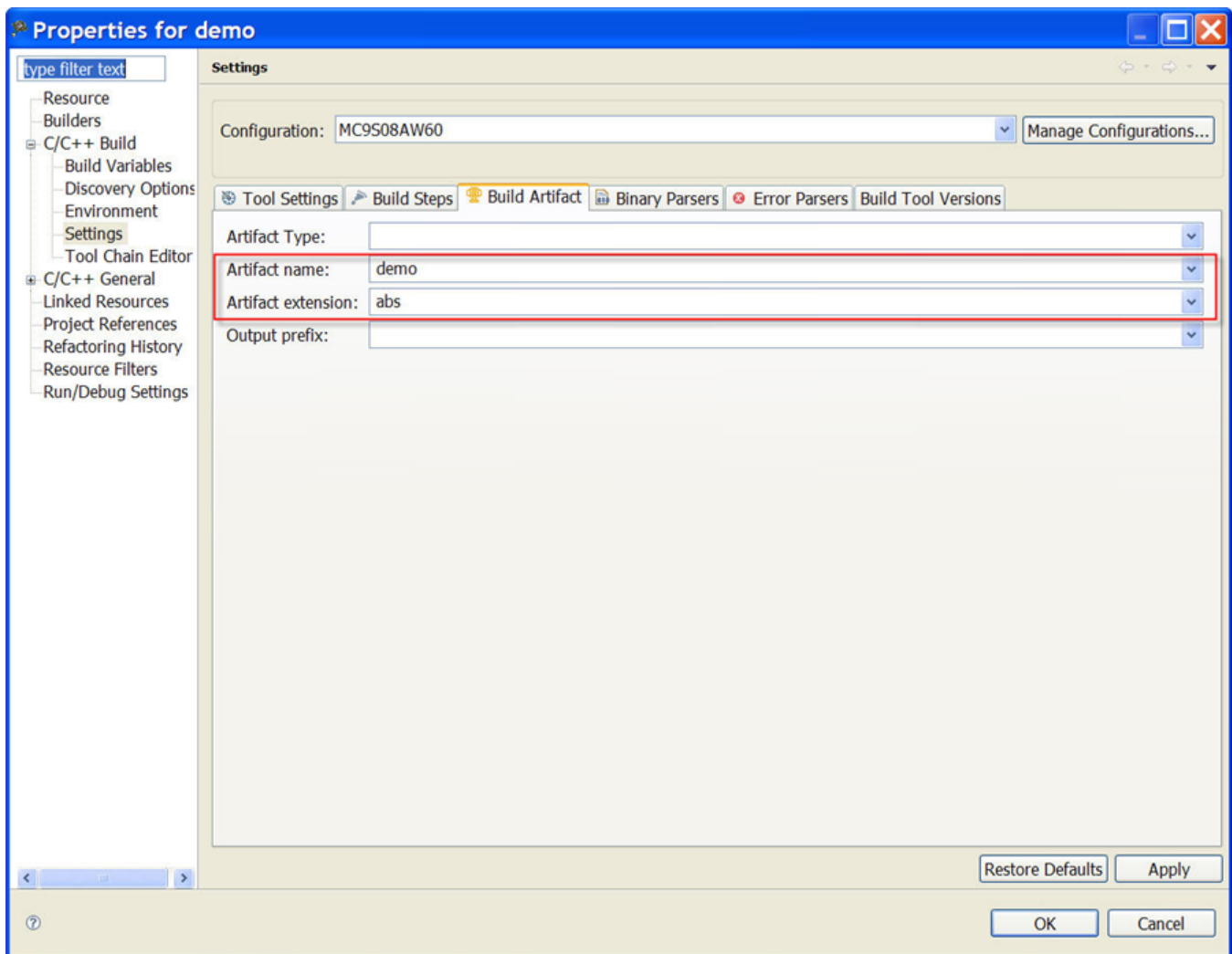
The **Settings** options appear on the right side of properties window.

4. Select **Build Artifact** tab.

5. Type the name of the executable that you want generated in the **Artifact name** textbox.

6. The **Artifact extension** textbox contains the extension of the executable that will be generated. The default extension for binary files is .abs. If another extension is desired, change the extension in the **Artifact extension** textbox (for more information, refer to the image listed below).





**Figure 7-49. Properties Window**

7. Click **Apply**.
8. Click **OK**.

The **Properties** window disappears.

When the project is rebuilt a new binary with the new extension is generated.

### 7.5.9 Can I debug the code from the generated assembly file step-by-step?

Yes this can be done. From the debugger perspective toolbar, select **Window > Show View > Disassembly** . The **Disassembly** window appears displaying the assembly code generated from the C code. Breakpoints can be set by double-clicking on assembly line.

### 7.5.10 If I have a debugger session running on my application, how can I halt the application?

In the **Debug** perspective, click on the .abs being executed. From toolbar, select **Suspend** from the toolbar which halts the application temporarily. To run the application, select **Resume** from toolbar.

### 7.5.11 Why do I get the following error message when I attempt to debug my project?

```
Program does not exist
```

This error occurs because the project you are trying to debug has not been built yet. To build the project, right-click the project name and select the **Build Project** option.

### 7.5.12 Can I change the format of the Memory view?

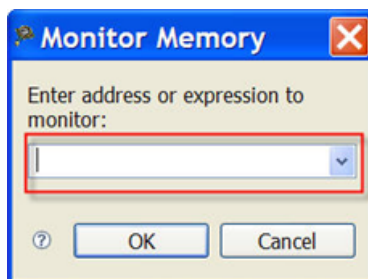
Yes. In order to change the format of the memory window do the following steps:

1. Select **Window > Show View > Memory** in the debug perspective.

The **Memory** window appears docked to the lower part of debugger perspective.

2. Click  sign icon.

The **Monitor Memory** window appears.



**Figure 7-50. Monitor Memory Window**

3. Type memory address in the **Enter address or expression to monitor** textbox.

4. Click **OK**.

The **Monitor Memory** window closes. The memory location appears in the memory window. The memory contents can be viewed by selecting memory on the left hand side, as shown in the following figure.

Address	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
000018F0	00	00	9A	C7	18	00	20	FB	00	00	00	00	00	00	00	00
00001900	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00001910	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00001920	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00001930	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00001940	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00001950	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00001960	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00001970	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00001980	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00001990	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000019A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000019B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000019C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000019D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000019E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000019F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00001A00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00001A10	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00001A20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00001A30	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00001A40	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00001A50	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

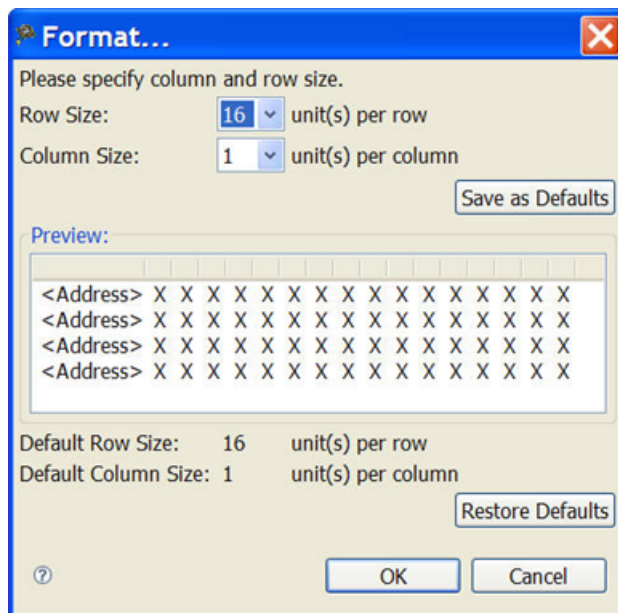
**Figure 7-51. Memory View**

5. Right-click anywhere in the **Memory** view

A context menu appears.

6. From context menu, select **Format**.

The **Format** window appears.



**Figure 7-52. Format Dialog Box**

**Row Size** indicates how the addresses will increase. **Column Size** displays the number of bytes in that address space. For this example, the address will increase by 16 bytes and each row will display 4 bytes in each column. The **Preview** window changes and displays address every 2 bytes by 2 bytes per column.

7. Click **OK**.

The **Format** window closes.

The memory view changes and displays the addresses using new format, as the following image shows.

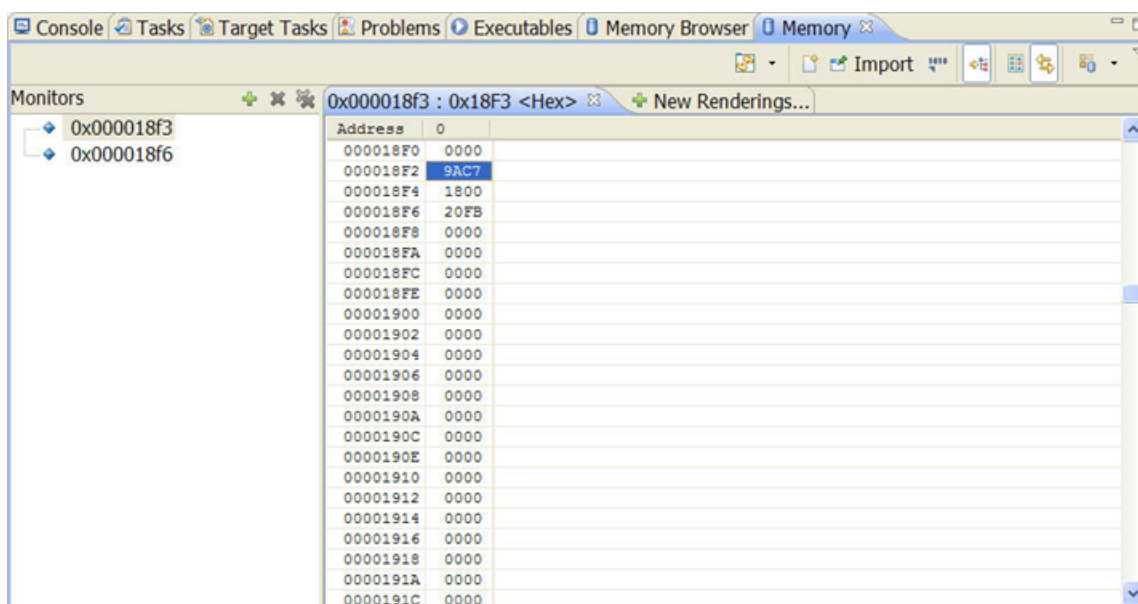


Figure 7-53. Updated Memory View

### 7.5.13 Why does the DE.exe appear in the Windows Task Manager even after terminating and closing the debugger perspective?

The Debugger Engine (DE) is loaded when it is first needed in one of the following situations:

1. Debugger is launched
2. Debugger Shell is opened

The DE.exe is terminated only when CodeWarrior is terminated.

### 7.5.14 Is there a way to display the variables always in the Hexadecimal format?

Yes. There is a way to display the variables always in Hexadecimal instead of the default decimal format. In order to accomplish this, follow steps below.

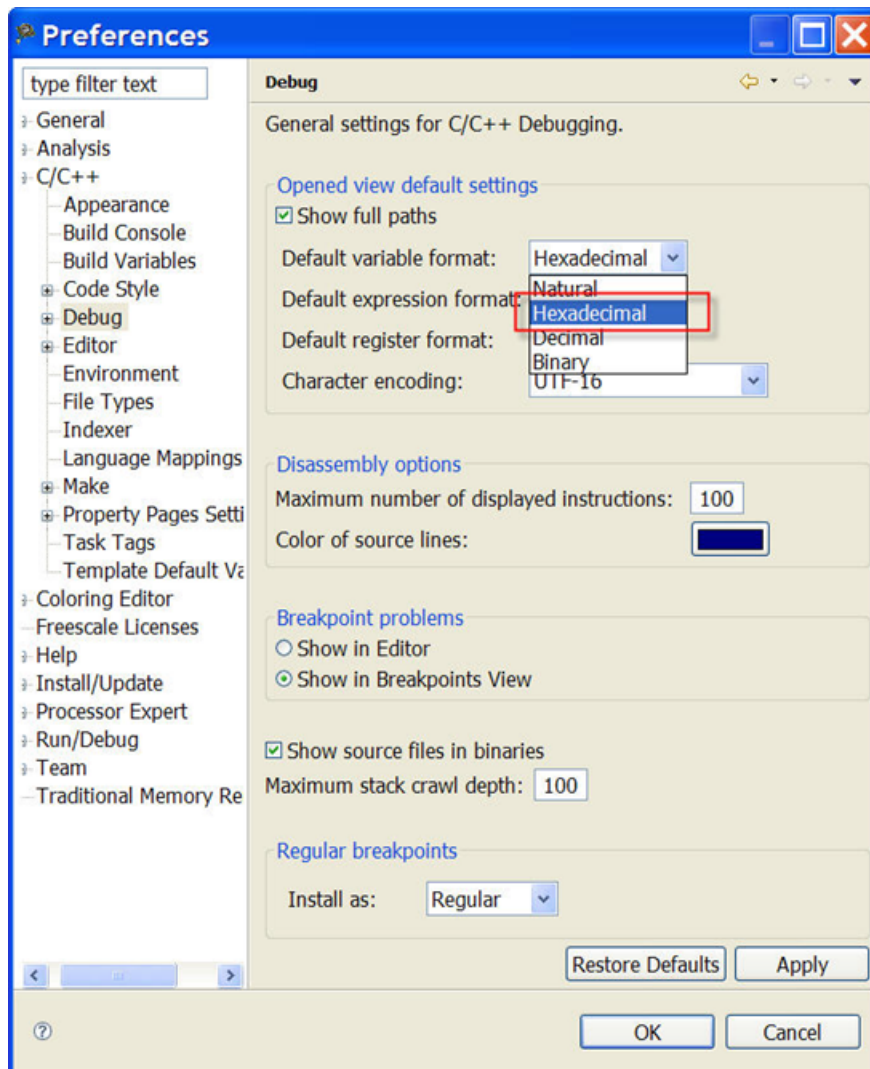
1. Select **Window > Preferences** from C/C++ perspective toolbar.

The **Preferences** window appears.

2. Select **C/C++ > Debug**.

The **Debug** settings appear.

3. Change **Default variable format** to **Hexadecimal**. By default, it is set to **Natural** (for more information, refer to the image listed below).



**Figure 7-54. Preferences Window-Debug Page**

4. Click **Apply**.
5. Click **OK**.
6. Restart CodeWarrior Development Studio.

When **Variables** view appears in the debug perspective, the variables appear in the Hexadecimal format.

**NOTE**

This setting is a global setting.

## 7.5.15 Can I change the endianness that is displayed in the Memory view?

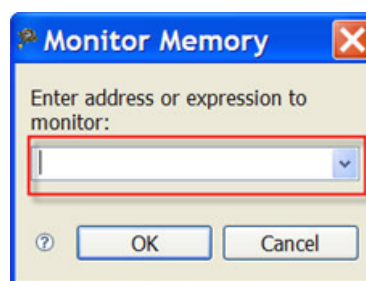
Yes. To change the endianness that is displayed in the memory view, follow these steps:

1. Select **Window > Show View > Memory** in the debug perspective.

The **Memory** window appears docked to the lower part of debugger perspective.

2. Click  sign icon.

The **Monitor Memory** window appears.



**Figure 7-55. Monitor Memory Window**

3. Type memory address in the **Enter address or expression to monitor** textbox.
4. Click **OK**.

The new address appears in the memory window. The contents can be viewed by selecting memory on the left hand side, as the following figure shows.

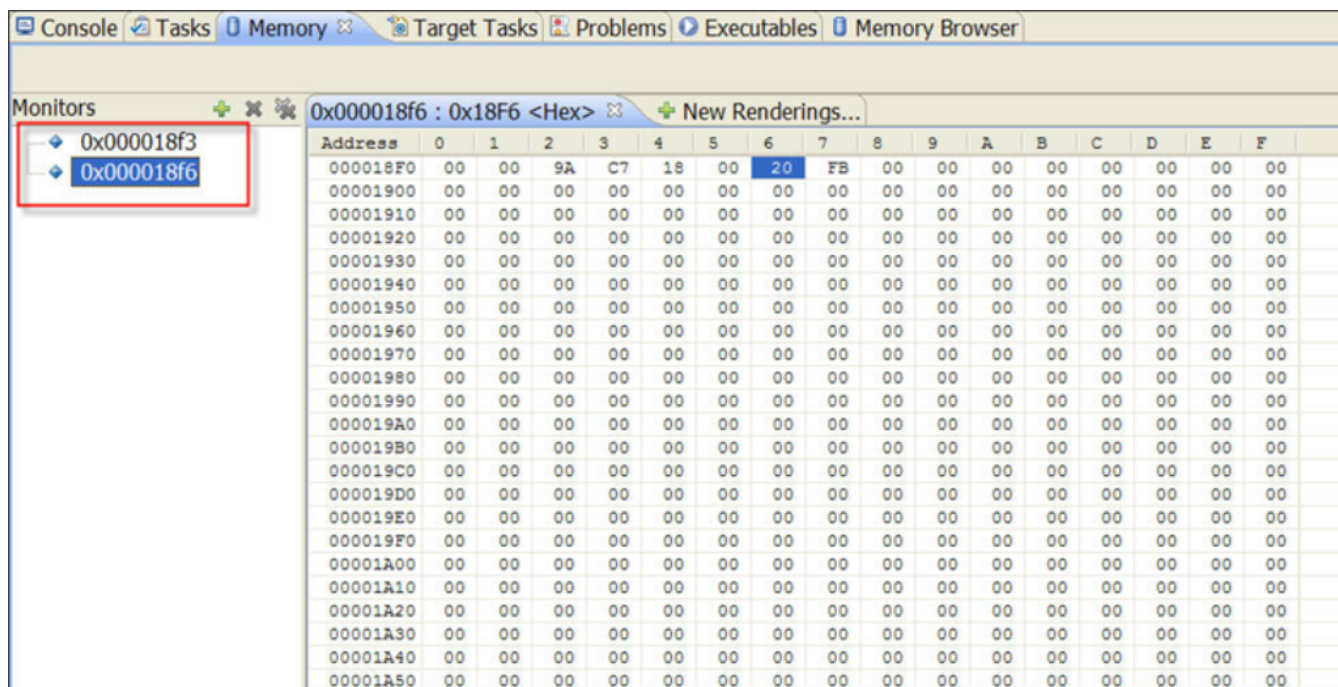


Figure 7-56. Memory View with Multiple Addresses

5. From the **Memory** Toolbar, select **New Renderings...**

The **New Renderings** window appears.

6. Select **Traditional**.

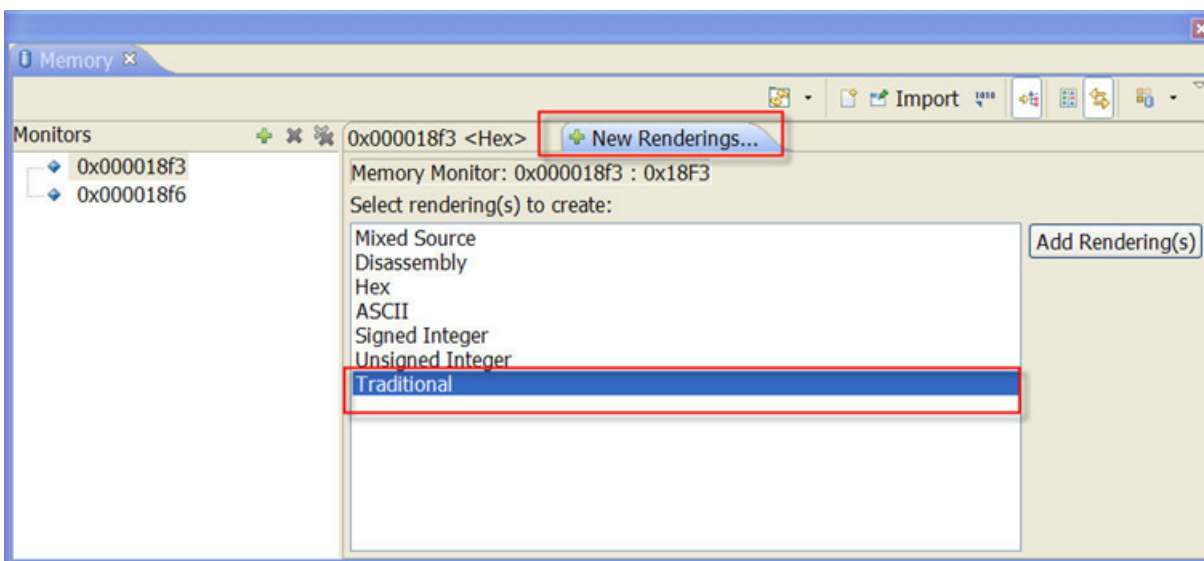


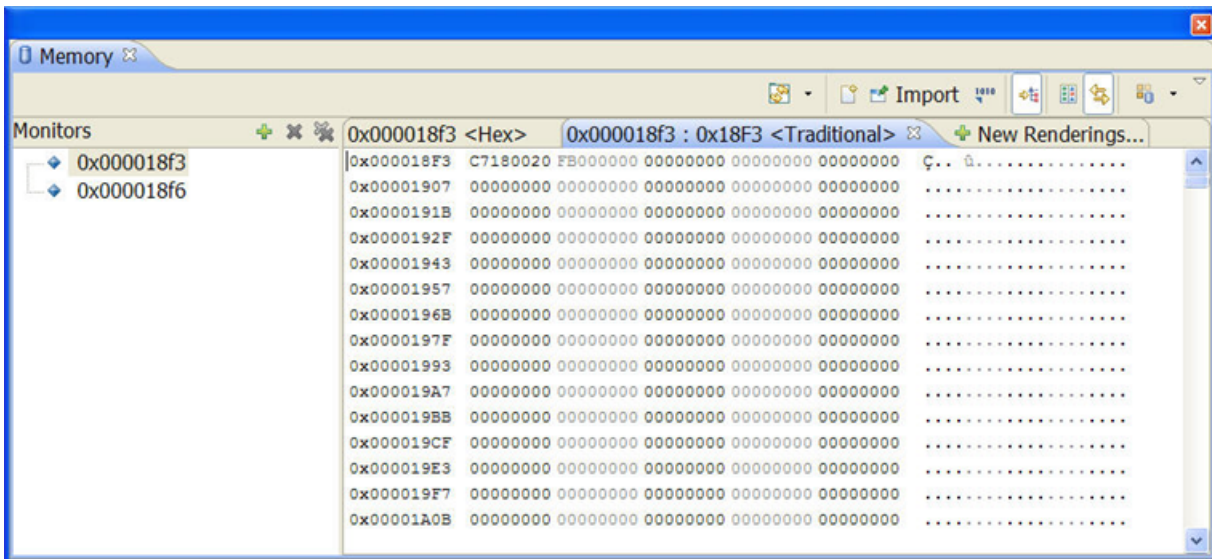
Figure 7-57. Memory View-New Renderings

7. Click **Add Renderings**.

8. Click **OK**.

The **Traditional** view appears in Memory view.





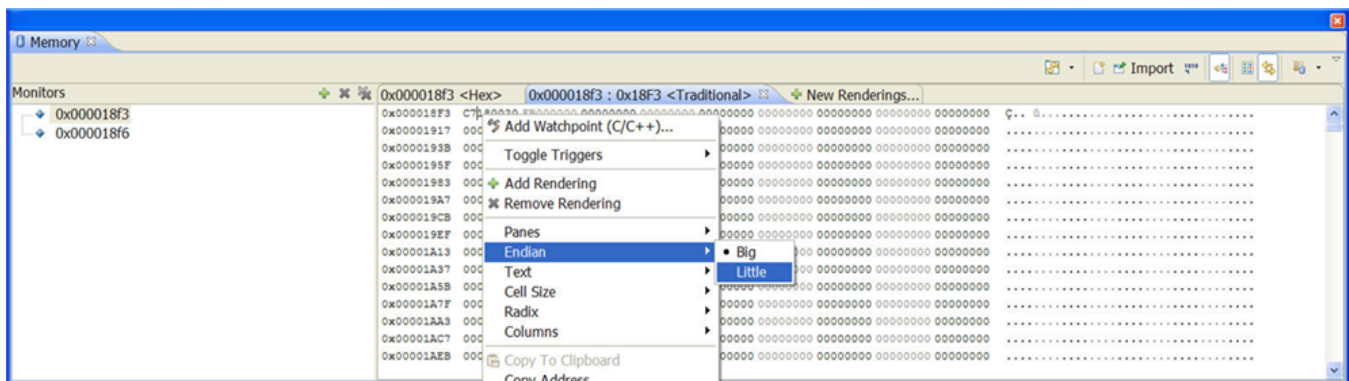
**Figure 7-58. Memory View-Traditional Tab**

9. Right-click anywhere in the **Traditional** tab.

A context menu appears.

10. From the context menu, select **Endian > Little**. By default it is **Big**.

The memory view displays little endianness.



**Figure 7-59. Memory View-Little Endian Addresses**

### NOTE

To change the endianness back to **Big**, right-click anywhere in the **Traditional** tab, and select **Endian > Big** from the context menu.

## 7.5.16 In the Memory view, is it possible to go back to the address that the rendering was created for?

Yes. In order to get back to the address that the rendering was created for, follow steps below:

1. Right-click anywhere in the **Memory** view.  
A context window appears.
2. From context menu, select **Reset to Base Address** .

The view changes to display the base address that the rendering was initially created for.

### 7.5.17 Is there a way to view disassembly at any location?

Yes it is possible to view disassembly at any location. To do so, follow steps below:

1. Right-click anywhere in the **Memory** view.  
A context window appears.
2. From context menu, select **Go To Address**.  
The **Go To Address** textbox appears.
3. Enter the address in the Address textbox. If the **Input as Hex** checkbox is selected, then the address can be entered as either a hexadecimal or a decimal address.
4. Click **OK**.

The assembly instructions located at the specified address appears in the **Memory** view (for more information, refer to the image listed below).

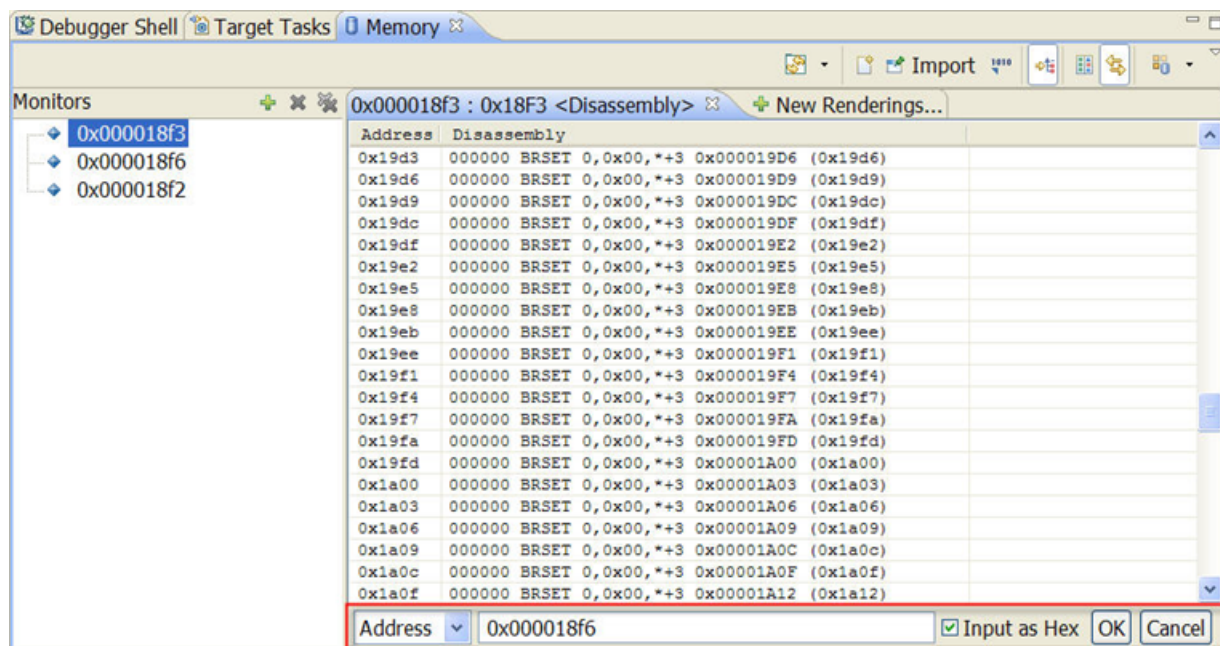


Figure 7-60. Memory View-Assembly Instructions

### 7.5.18 Is it possible to display the expressions always in the Hexadecimal format instead of the default decimal format?

Yes. It is possible to display the expressions always in Hexadecimal format instead of the default decimal format. In order to accomplish this follow steps below:

1. Select **Window > Preferences** from C/C++ perspective toolbar.

The **Preferences** window appears.

2. Select **C/C++ > Debug**.

The **Debug** settings appear.

3. Change **Default expression format** to **Hexadecimal**. By default, it is set to **Natural** (for more information, refer to the image listed below).

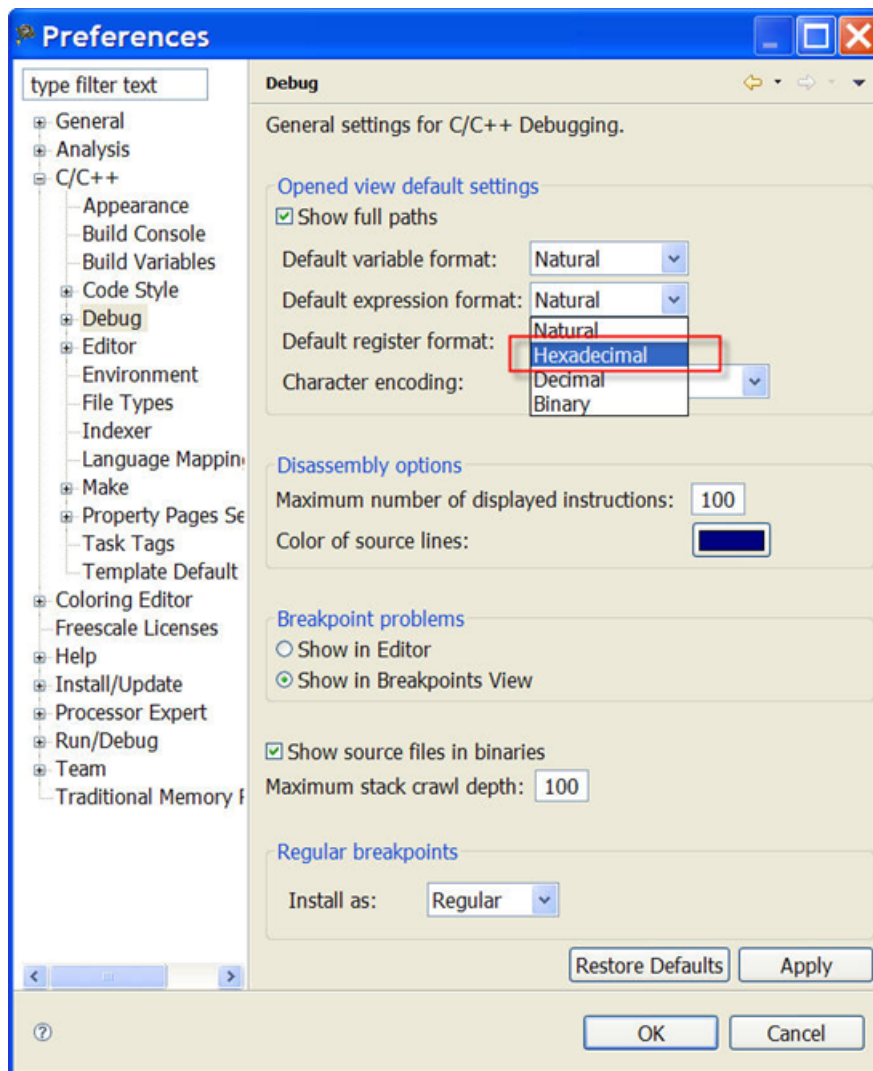


Figure 7-61. Preferences Window-Debug Page

4. Click **Apply**.
5. Click **OK**.
6. Restart CodeWarrior Development Studio.

When **Variables** view appears in the debug perspective, the variables appear in the Hexadecimal format.

#### NOTE

This setting is a global setting.

### 7.5.19 What does the Reset on Connect option do?

The **Reset on Connect** option resets all the cores.

## 7.5.20 Can I change the Program Counter (pc) value while in the debugger without having to restart the debugger?

Yes. In order to change the Program Counter (pc) without restarting the debugger, follow steps below in the debug perspective:

1. Start a debug session.

In the editor view, right-click on the source line where you want the pc to move to.

2. A context menu appears.
3. From context menu, select **Move To Line** .

The pc moves to line. The debugger moves the program counter to the location you specified. The editor view shows the new location.

## 7.5.21 How do I specify the program entry point?

To specify the program entry point, follow steps below in C/C++ perspective:

1. In the CodeWarrior projects window, right-click on project name.

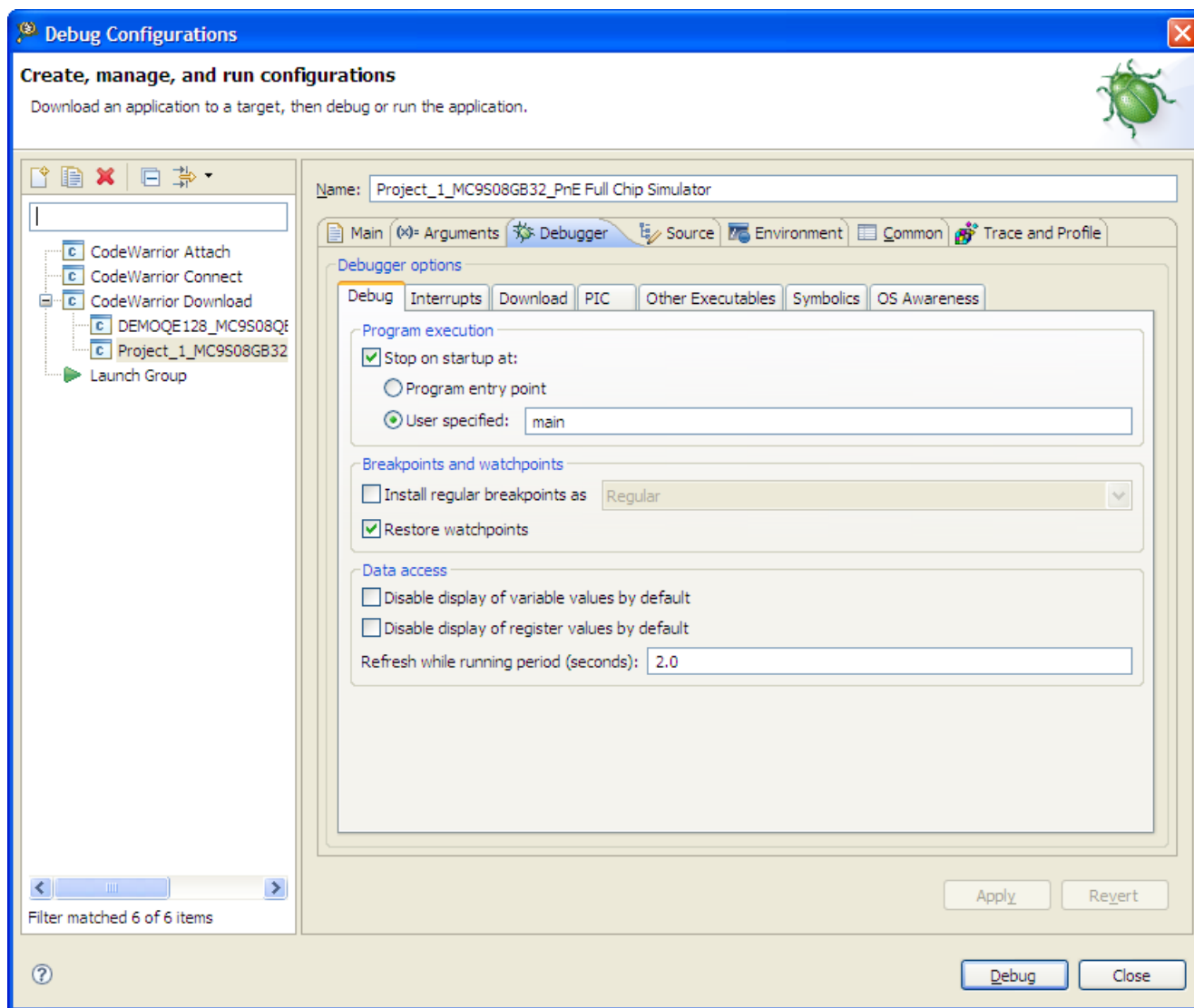
A context menu appears.

2. From context menu, right-click on project name, select **Debug As > Debug Configurations**.

The **Debug** window appears.

3. Under **CodeWarrior Download** , select project name.
4. Select **Debugger** tab.

The **Debugger** page appears (for more information, refer to the image listed below).



**Figure 7-62. Debug Configurations-Debugger Page**

5. Click **Apply**.
6. Click **Debug**.

### 7.5.22 How can I run a debug session repeatedly?

There are several of ways to run a debug session repeatedly. Below is a list of possible ways to run a debug session repeatedly:

- Select the **Debug** button to invoke the last debug session.

-or-

- Right-click the stack window and select **Relaunch** from context menu.

-or-

- If still in debug mode, click **Terminate and Relaunch** .

### 7.5.23 Can I direct console output to a file instead of the console window?

Yes, you can direct the console output to a file instead of the console window. To accomplish this, follow steps given below:

1. In the CodeWarrior projects window, right-click on project name.

A context menu appears.

2. From context menu, right-click on project name, select **Debug As > Debug Configurations**.

The **Debug** window appears.

3. Under **CodeWarrior Download** , select project name.
4. Select **Common** tab.

The **Common** page appears (for more information, refer to the image *Debug Configurations-Common Page* listed below).

5. Select **File** and specify the file to which the console output is directed.

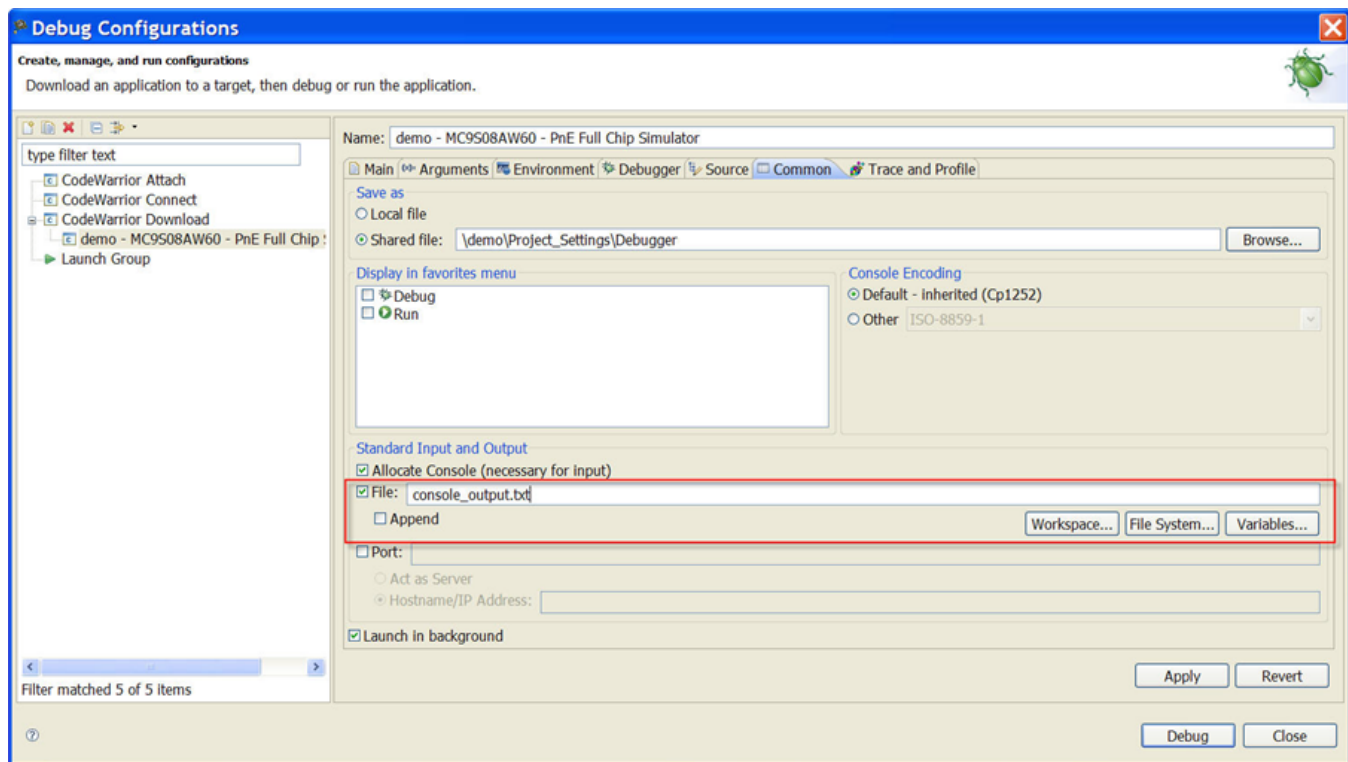


Figure 7-63. Debug Configurations-Common Page

6. Click **Apply**.
7. Click **Debug**.

The console output will be directed to the specified file.

### 7.5.24 How do I resolve the following error message that I get when I use Ctrl-Click to open variable declaration?

```
Could not find symbol xx in index.
```

To resolve this error message, follow the steps below:

1. In the CodeWarrior project window, right-click on project name.

A context menu appears.

2. From context menu, select **Properties**.

The **Properties** window appears.



3. Select **C/C++ General > Indexer** .

The **Indexer Settings** appear.

4. Check the **Enable project specific settings** checkbox.
5. From the **Select Indexer** drop-down list, select **Full C/C++ Indexer (complete parse)** .
6. Click **Apply** .
7. Click **OK** .

The issue will be resolved. Now, you can use Ctrl+click to open the variable declaration.

### 7.5.25 How can I configure and launch an external tool or debugger?

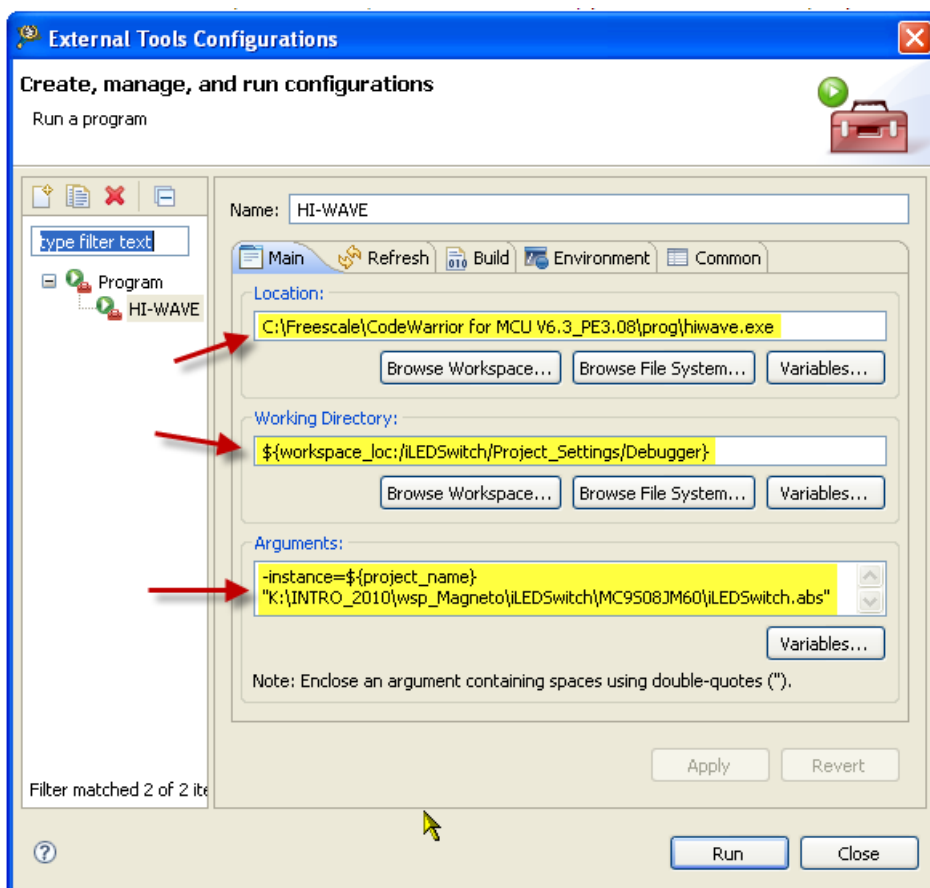
To configure an external tool, follow the steps below:

1. Select **Run > External Tools > External Tools Configurations**.

The **External Tools Configurations** wizard appears.

2. Click the **New Launch Configuration** button in the left pane.
3. Specify the location for the external tool you want to configure.
4. Specify the working directory and arguments.

The following figure shows the configuration settings for the 8/16 bit external debugger that is present in classic CodeWarrior for Microcontrollers.



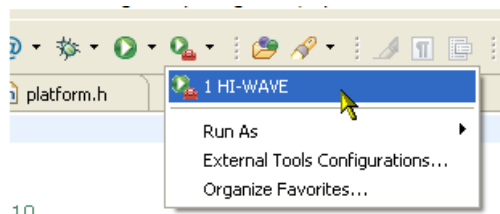
**Figure 7-64. External Tools Configurations Wizard**

-instance option ensures that an already existing/running debugger with the same instance can be used for debugging another project or file.

Also, you can specify multiple variable using the **Variables** button. For example you can use a variable to debug a selected file, or have IDE to prompt for a file.

5. In the additional tabs, such as **Refresh** and **Build** , you can configure if resources need to be refreshed, or if a build should be performed first.

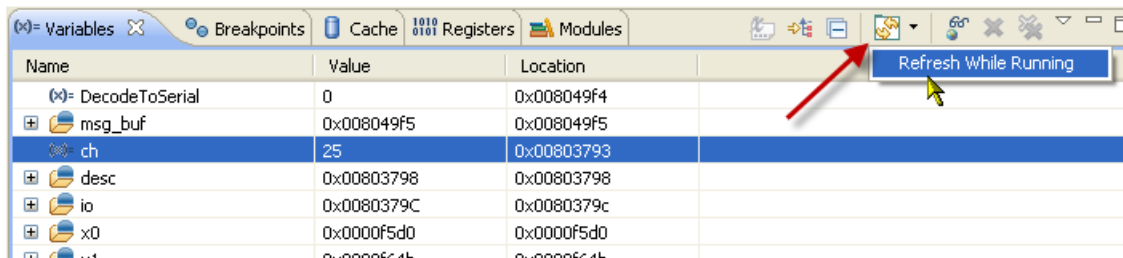
After configuring the external tool, you can launch the tool from the CodeWarrior IDE toolbar.



**Figure 7-65. Launch External Tool**

## 7.5.26 How can I monitor a variable in the debugger while a target is running?

Using the Live View or Refresh While Running feature of the eclipse based CodeWarrior debugger, you can see how a variable is changing while the target is running.

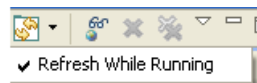


**Figure 7-66. Refresh While Running**

### NOTE

For Background Debug Mode (BDM) targets, such as RS08, S08, and ColdFire, the CPU registers cannot be monitored. However, you can use the Refresh While Running feature for anything that is memory mapped.

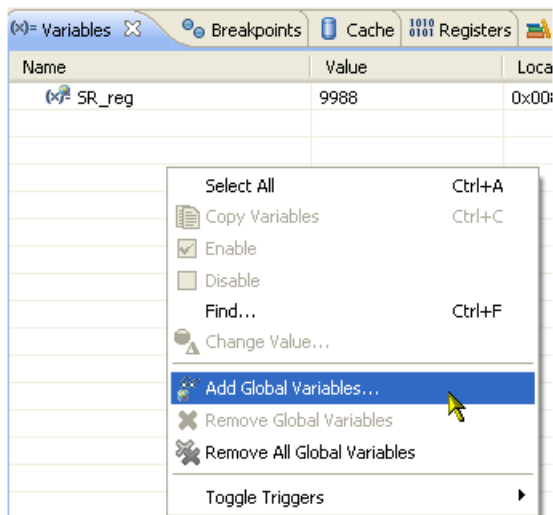
When you select the feature, a checkmark appears indicating the feature is enabled:



**Figure 7-67. Enable Refresh While Running Feature**

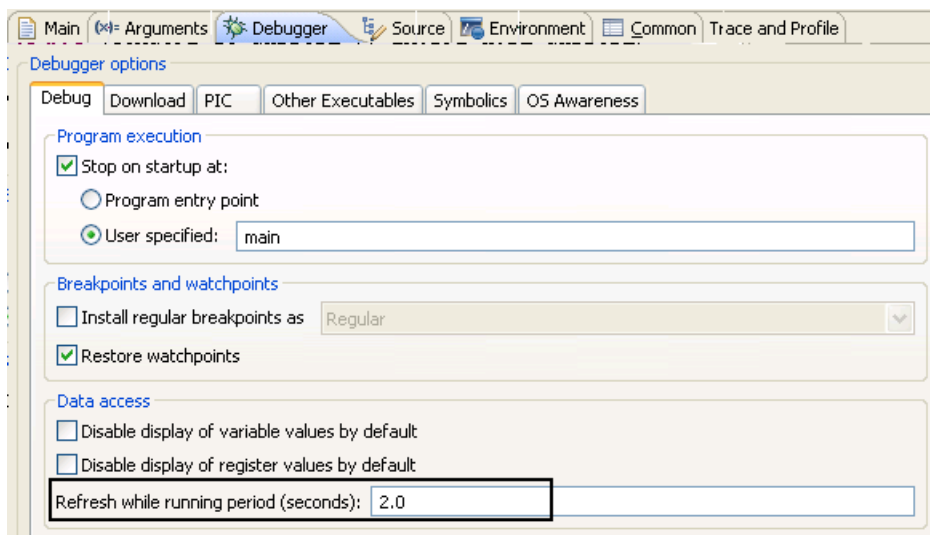
In order for variables to be refreshed, they need to be Global. Global variables are variables with external scope in C. The module variables or static global variables are not considered as global in the eclipse-based CodeWarrior IDE.

In the **Variables** view, add the global variables you want to use as the 'live view' variables, as shown in the following figure.



**Figure 7-68. Add Global Variables**

You can configure the refresh rate in the **Run > Debug Configurations > Debugger** page.



**Figure 7-69. Configure Refresh Rate**

### 7.5.27 How can I inspect global variables in the variables view?

In Microcontrollers V10.1 and onwards, you can add and view global variables.

You can add or view global variables using the **Add Globals** dialog box. To open the **Add Globals** dialog box:

1. Right click the **Variables** view in the Debug perspective.

2. From the context menu, select **Add Global Variables** .

The **Add Globals** dialog box appears.

3. Select the variable you want to add as global variable.

4. Click **OK** .

The following figure shows an example.

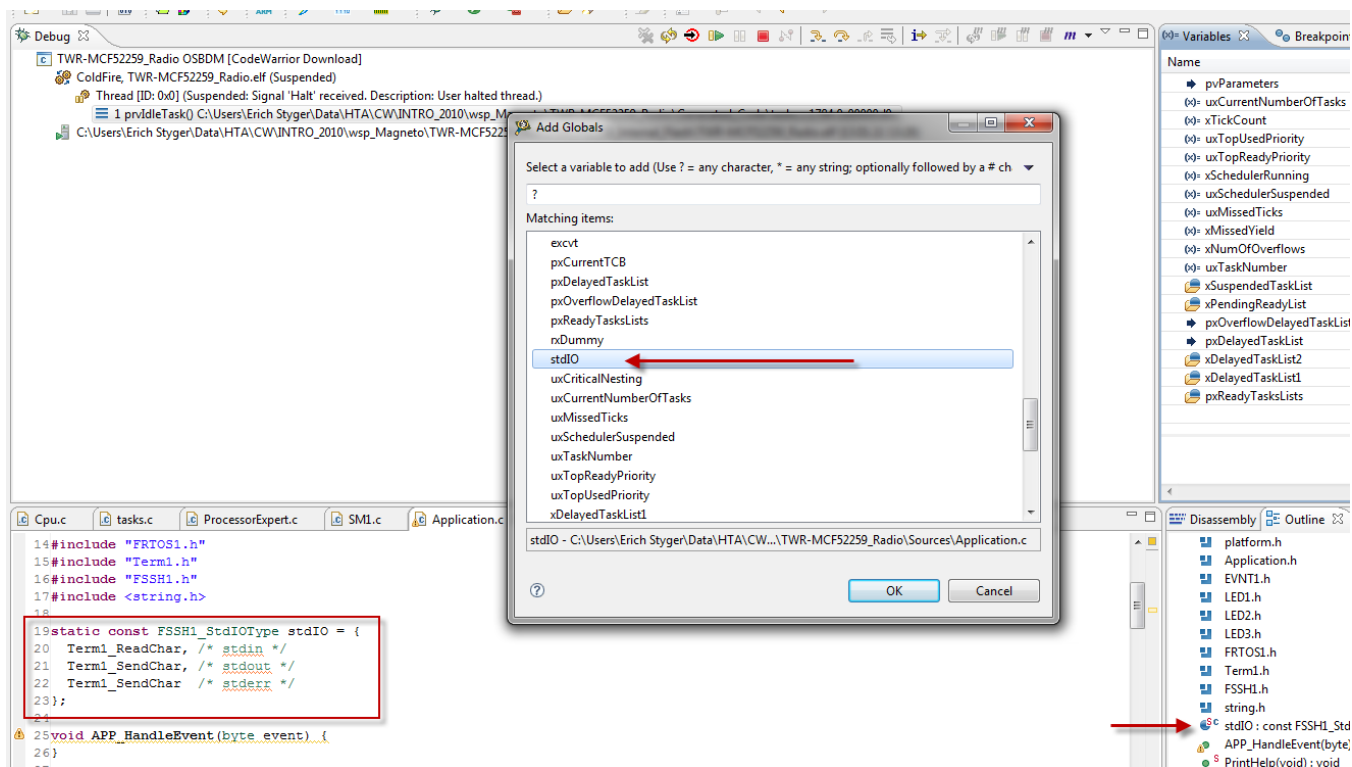
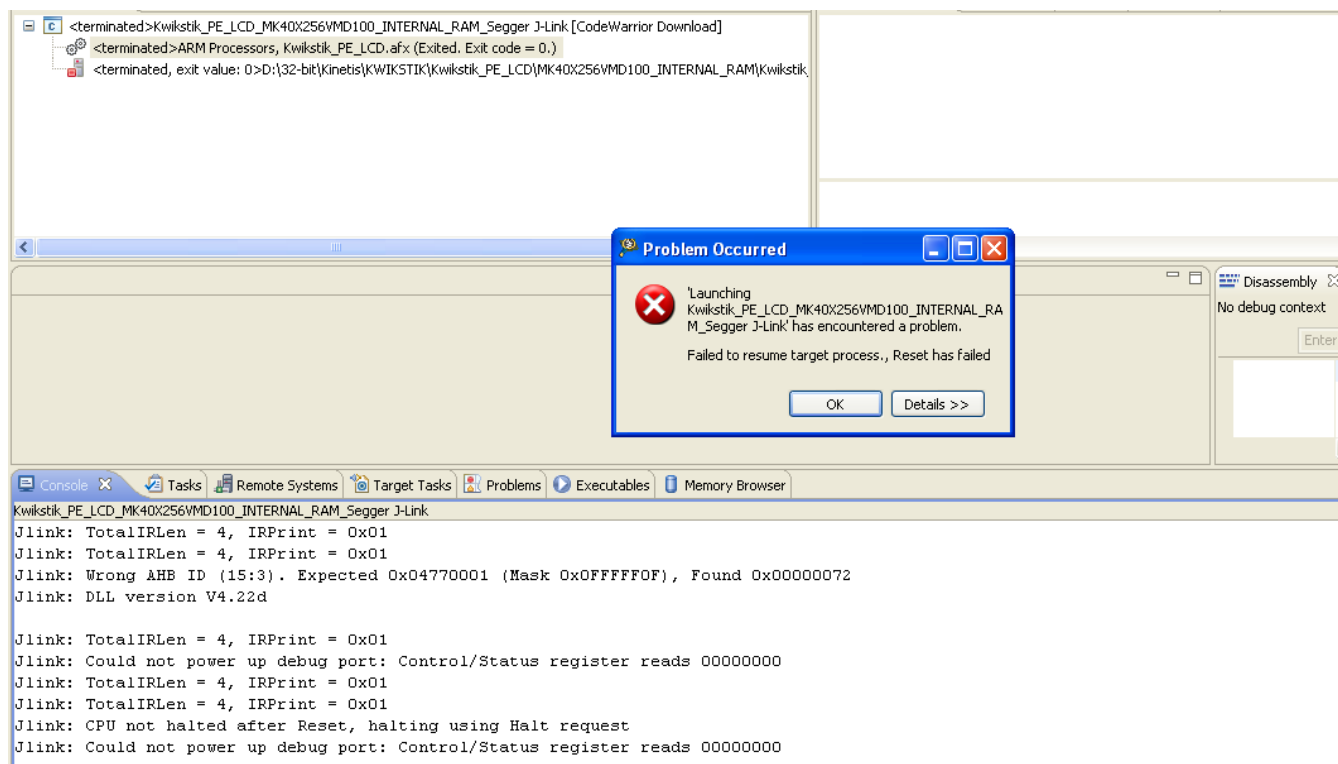


Figure 7-70. Static Global Variables

## 7.5.28 How do I resolve the error encountered on the Kwikstik board?



**Figure 7-71. Launching Error on the Kwikstik Board**

In such case, the chip may be secured or the code in flash might put it in a low power mode and not permit a debug connection. Since you only have the possibility to use the embedded Segger J-Link probe, you have to recover the part by opening the `JLink.exe` application which comes with the Segger package installation and type the, `unlock kinetis` command.

The latest Segger J-Link package can be found here:

<http://www.segger.com/cms/jlink-software.html>

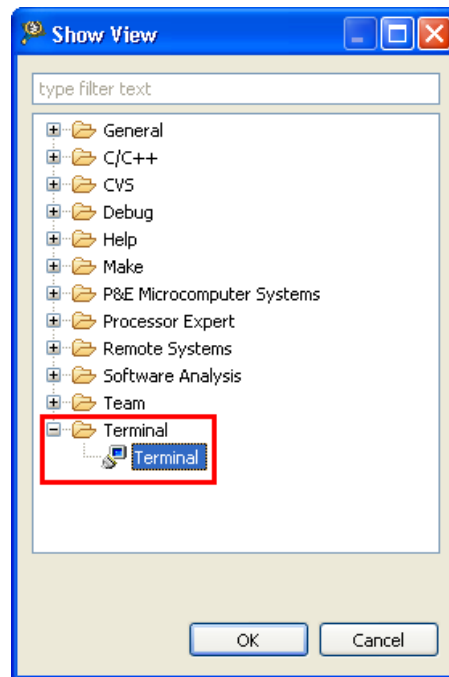
There are several ways how a chip can get secured, including bugs in the silicon, or that you download the wrong image to the target, ending up writing/flashing the security areas of the chip. In such cases the P&E connections, the P&E firmware is able to handle unsecure directly, means you are not required to use an external utility as in the case for J-Link.

### **7.5.29 At the moment serial data is output in the console in the debugger, which is slow with J-Link. Is there any way to view the output of the debugger in the HyperTerminal?**

You can view the output in HyperTerminal by using the **Terminal** view. To open the **Terminal** view, follow the steps listed below:

1. Select **Window > Show View > Other** from the IDE menu bar.

The **Show View** dialog box appears.



**Figure 7-72. Show View Dialog Box**

2. Expand the **Terminal** tree control and select **Terminal**.
3. Click **OK**.

The **Terminal** view appears in the view stack at the bottom of the IDE window.

### 7.5.30 Is it possible to flash a target MCU from the command-line?

It is possible to flash the target using a `.tcl` file. When you pass it to `cwide.exe`, the CodeWarrior opens and execute the commands. If the last command closes the CodeWarrior, the interface becomes transparent.

This can be done with the following parameters:

`-data{workspace}` defines the workspace used

`-vmargs-Dcw.script={script}` defines the script that must be run

It is important to note that, when we use `-vmargs` argument to `cwide.exe` it overrides the parameters defined in `cwide.ini`. Hence, you must pass all parameters to the command-line.

### 7.5.31 Is there an easy way to suppress the target re-programming when there has been no change since the previous session?

Follow the steps listed below:

1. Select **Run > Debug Configuration** from the IDE menu bar.
2. Expand the **CodeWarrior Download** tree-control and select the desired launch configuration.
3. Click the **Debugger** tab in the right panel.
4. Click the **Download** tab and clear the **Perform Standard Download** checkbox.

The target re-programming will now be suppressed if there is no change since the previous session.

#### NOTE

Do not forget to check the **Perform Standard Download** checkbox again when things change in your project. At the moment, there is no automatic way of detecting whether the code on the target is different from the one you are trying to debug, so you will have to do this manually each time.

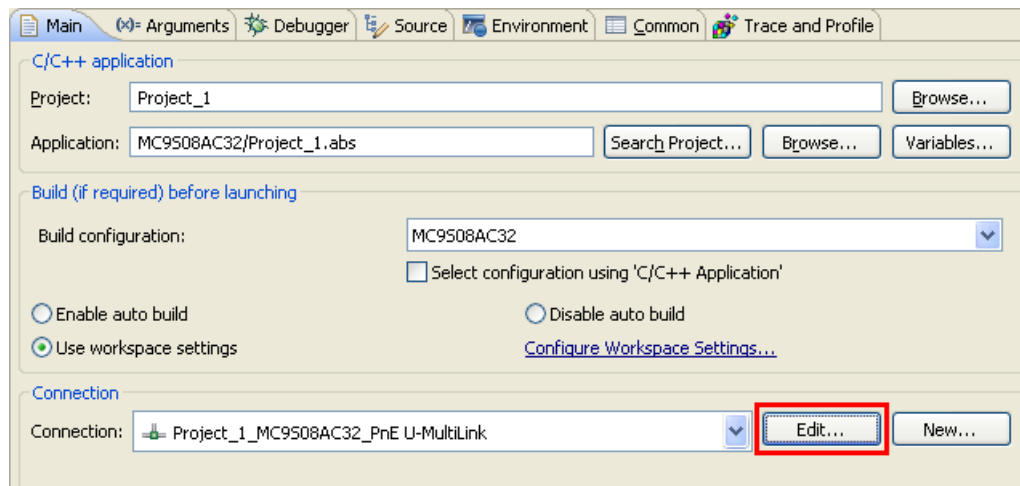
### 7.5.32 The microcontrollers are usually trimmed in the factory. How can I trim it myself if I need something better?

In the CodeWarrior for Microcontrollers v10.1, the trimming functionality is available in the **PEMicro** menu of the IDE menu bar. Whereas, in the CodeWarrior for Microcontrollers v10.2 or higher, you can access this functionality from **Debug/Run Configurations** dialog box. To trim a microcontroller follow the steps listed below:

1. Select **Debug Configurations** from the IDE menu bar.

The **Debug Configurations** dialog box appears.

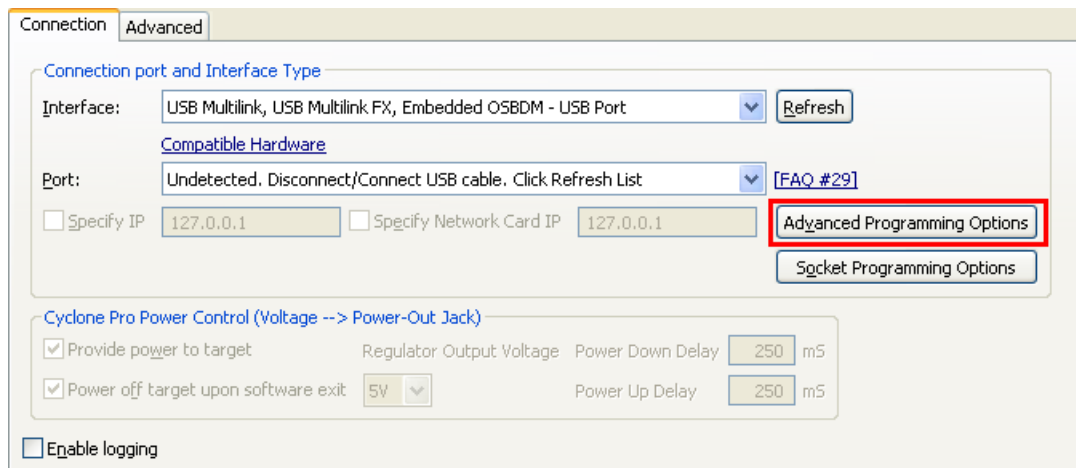




**Figure 7-73. Debug Configurations Dialog Box**

2. Click **Edit** from the **Connection** group.

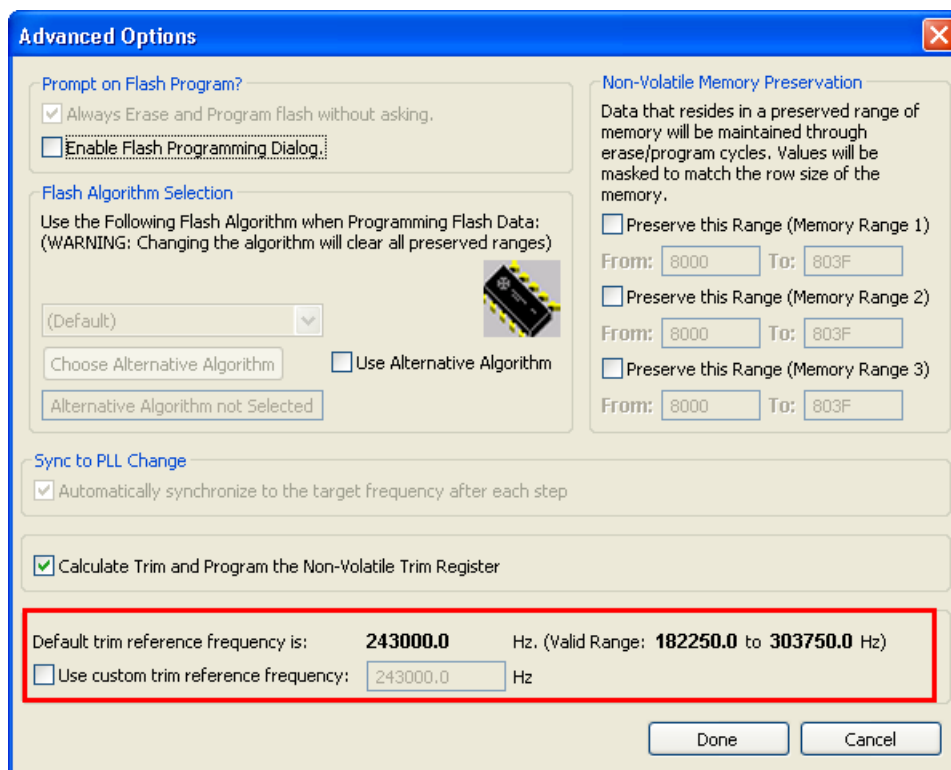
The **Properties for <project\_name>** dialog box appears.



**Figure 7-74. Properties for <project\_name> Dialog Box**

3. Click **Advanced Programming Options** from the **Connection** tab.

The **Advanced Options** dialog box appears.



**Figure 7-75. Advanced Options Dialog Box**

4. Check the **Use custom trim reference frequency** checkbox and specify the desired trim frequency for your microcontroller.

**NOTE**

The trimming feature is not supported for all microcontroller devices.

### 7.5.33 How do I get a full dump of my device memory for offline analysis?

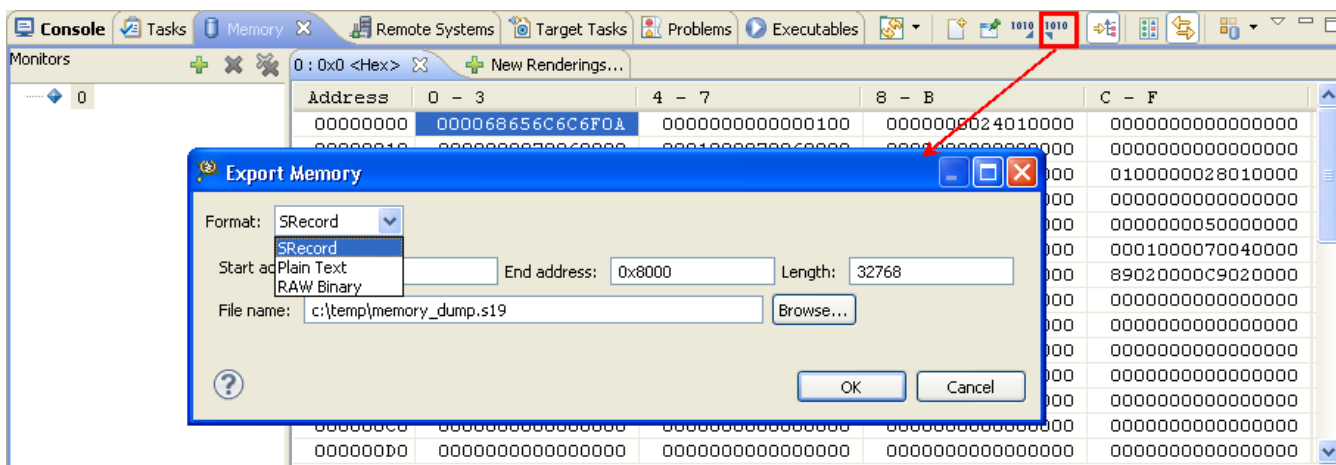
There are multiple ways how to dump the device memory to a file. This is very useful to inspect content offline, to compare dumps or as well to later restore the dumps on the device. You can save the device memory to a file by:

- Using **Memory** view
- Using Target Task
- Using Debugger Shell

**To export memory using Memory view:**

1. Click the **Export** button from the Memory view toolbar.

The **Export Memory** dialog box appears.



**Figure 7-76. Exporting Memory using Memory View**

2. Select **SRecord** from the **Format** drop-down list.
3. Specify the start address, end address and the `s19` file where you want the memory to be saved in the **Start address**, **End address** and the **File name** textboxes respectively.
4. Click **OK**.

The memory is saved in the specified file.

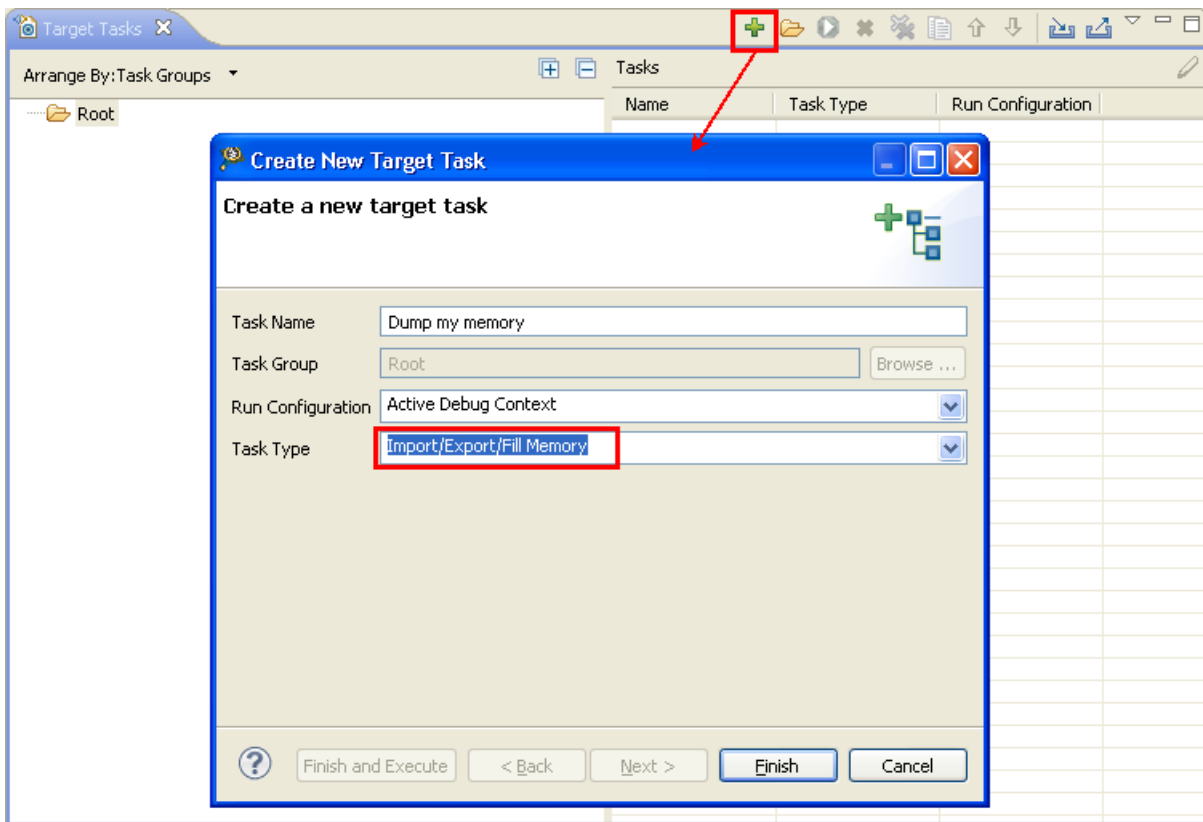
You can import the memory again for later use by using **Import** button.

**To export memory using Target Task view:**

1. Open the **Target Task** view.
2. Click the green plus button from the **Target Task** view toolbar.

The **Create New Target Task** dialog box appears.

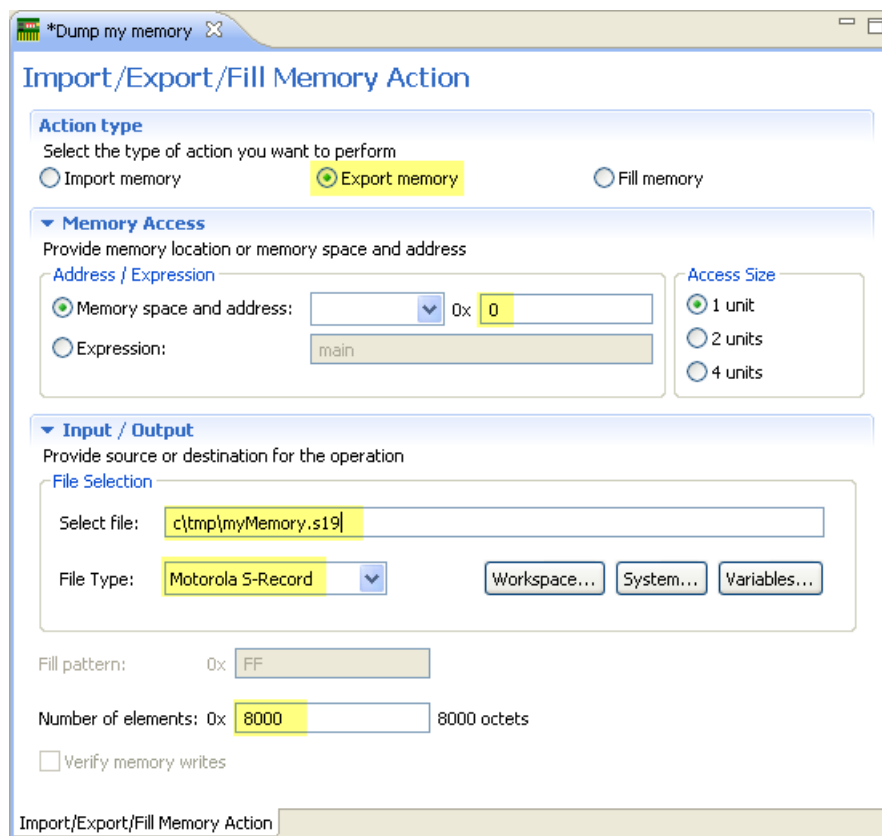
3. Specify a name in the **Task Name** textbox, and select **Import/Export/Fill Memory** from the **Task Type** drop-down list.



**Figure 7-77. Target Task View - Create New Target Task Dialog Box**

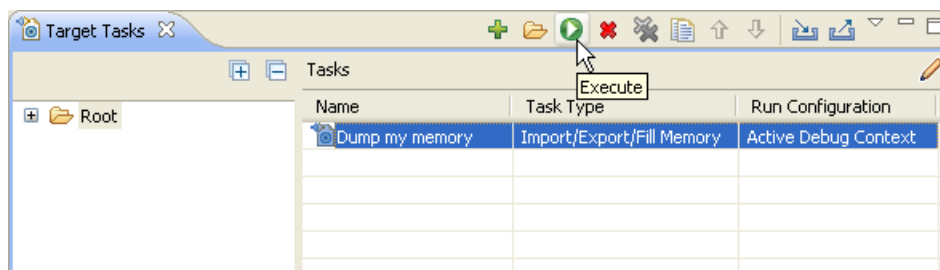
4. Click **Finish**.

The newly created target task appears in the view stack.



**Figure 7-78. Newly Created Target Task**

5. Specify the following details in the newly created target task:
  - a. Select **Export memory** in the **Action Type** Group.
  - b. Select **Memory space and address** option and specify the start address in the **Memory Access** group.
  - c. Specify the file where you want to dump the memory in the Select file textbox, and select Motorola S-Record from the File Type drop-down list in the **Input / Output** group.
  - d. Specify the desired number of elements in the **Number of elements** textbox.
6. Press **Ctrl+S** to save the settings.
7. Click the **Export** button from the **Target Task** view toolbar.

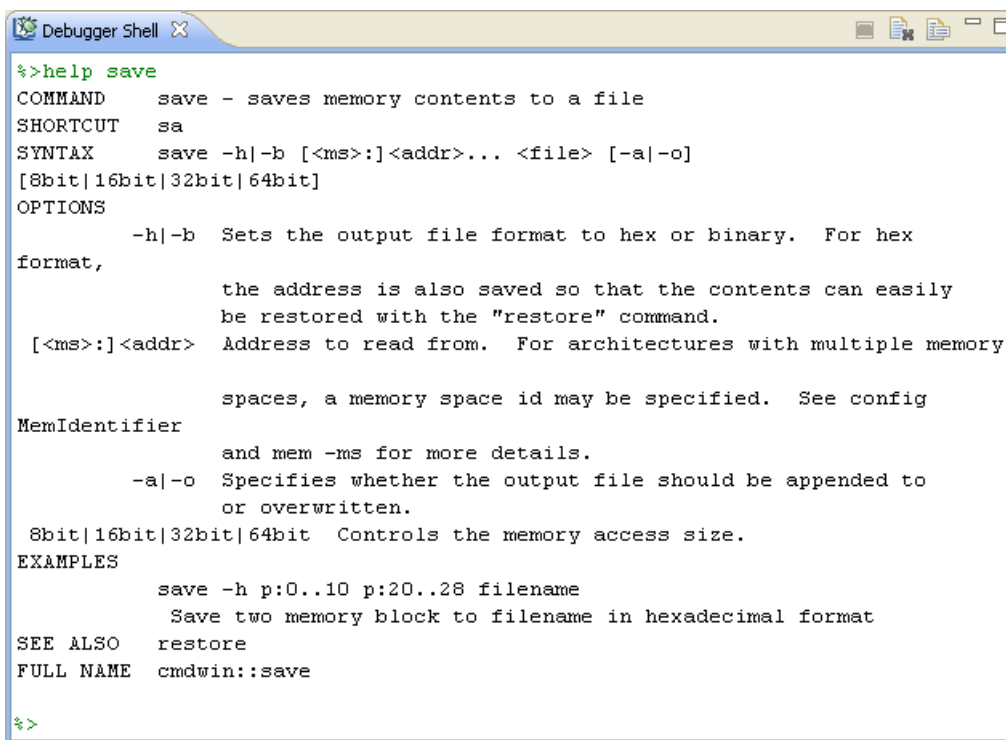


**Figure 7-79. Executing the Target Task to Save the Memory**

The device memory is saved to the file specified. You can import the saved memory format later again by using the **Import memory** feature from the **Action type** group.

### To export memory using Debugger Shell view:

You can use `save` command to dump the device memory to a file for later use. The following image shows the help for `save` command:



```

Debugger Shell
%>help save
COMMAND      save - saves memory contents to a file
SHORTCUT     sa
SYNTAX       save -h|-b [<ms>:]<addr>... <file> [-a|-o]
              [8bit|16bit|32bit|64bit]
OPTIONS
    -h|-b    Sets the output file format to hex or binary.  For hex
format,
              the address is also saved so that the contents can easily
              be restored with the "restore" command.
    [<ms>:]<addr>  Address to read from.  For architectures with multiple memory
MemIdentifier
              spaces, a memory space id may be specified.  See config
              and mem -ms for more details.
    -a|-o    Specifies whether the output file should be appended to
              or overwritten.
    8bit|16bit|32bit|64bit  Controls the memory access size.
EXAMPLES
    save -h p:0..10 p:20..28 filename
              Save two memory block to filename in hexadecimal format
SEE ALSO     restore
FULL NAME    cmdwin::save
%>

```

**Figure 7-80. Debugger Shell - save Command**

For example:

```
save -h 0x0..0x8000 c:\\tmp\\myDump.txt -o
```

The `-h` option specifies the Hex Format.

You can import the saved memory easily with the `restore` command.

## 7.5.34 When I build my application I received the message, "Device is secure. Erase to unsecure?" What do I do?

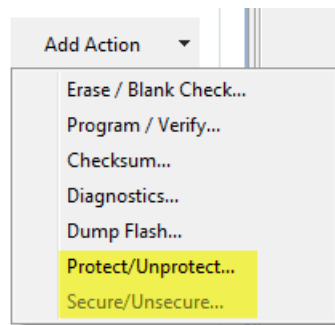
Most microcontroller offer different levels of protection, usually with special configuration registers and settings:

- **Protect:** the device or parts of it are protected against programming the flash memory. This is typically used to protect a bootloader or any parts of the device to be

overwritten either by the debugger or by the application itself. To unprotect usually a configuration register has to be reprogrammed or the flash memory needs a mass erase.

- **Secure:** Access to the flash memory with an external tool or debugger is not possible any more. This used to protect the software on the device to be inspected or to prevent reverse engineering. To recover the device a mass erase typically has to be applied.
- **Disable Mass Erase:** This is probably the most dangerous option a device can offer for a developer. With this configuration set, it will not be possible to do a mass erase of the device. So if you combine *secure* with *disable mass erase*, you will not be able to recover your device any more.

Depending on the device/flash features, CodeWarrior provides **Protect/Unprotect** and **Secure/Unsecure** target task actions of the flash programmer:



**Figure 7-81. Protect/Unprotect, Secure/Unsecure Target Task Actions**

In case during development you wrote/selected wrong settings to your device, and with this accidentally secure it. And if your wrong settings includes *disable mass erase* as well, you will not be able to recover your device anymore. So you have to be very careful not to secure any devices as this might be a permanent thing, unless your settings requires so.

#### NOTE

Connecting to the *wrong* device might trigger a *false* alarm about the device being *secured*. To avoid getting this message accidentally, always carefully check to which board/device you are connecting.

### 7.5.35 When I connect to my board, it shows a connection failure. What do I do?

If you are facing problems connecting to your board, follow these steps:

1. Start the task manager, and check if you have multiple instances of `DE.exe` (or even multiple `cwide.exe` instances). If so, you have to kill the extra instances using the task manager.
2. If you feel that the Debugger Engine somehow still causing the problem, kill the single remaining `DE.exe` as well. That saves you to exit the Eclipse IDE and to restart it.

If following the above listed two steps does not help, then it could be that the problem is in the Eclipse IDE. For this close the IDE and restart it. Generally, after following the step 1 and step 2 listed above you do not need this step.

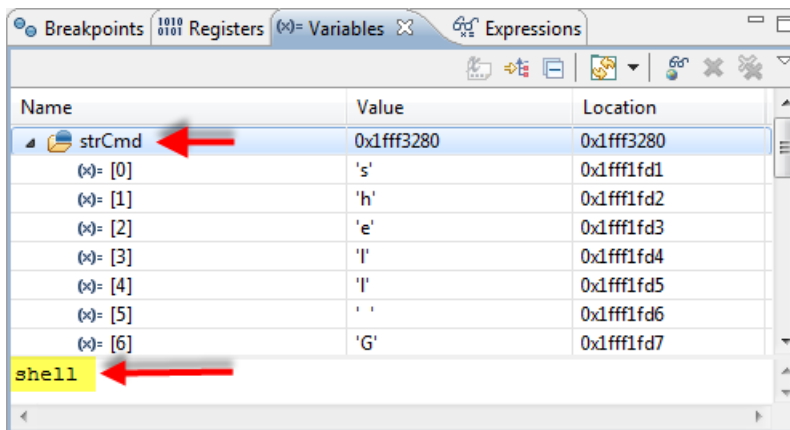
### 7.5.36 How do I debug the strings?

You can debug the strings using, Variable view, or the Editor view.

- Using Variable view:

You can easily debug the strings using **Variables** view. To debug a string, select the desired variable name in the **Variables** view that you want to debug. The preview panel shows the associated string. By using the preview panel:

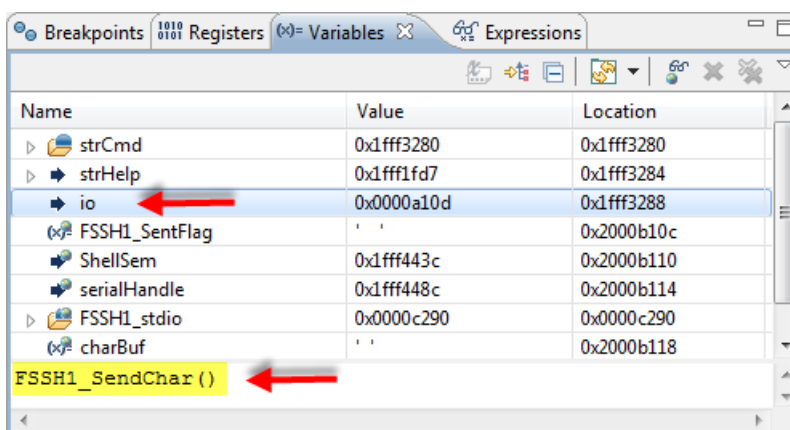
- You can view the associated string. The following figure shows an example:



**Figure 7-82. Details Panel of Variables View**

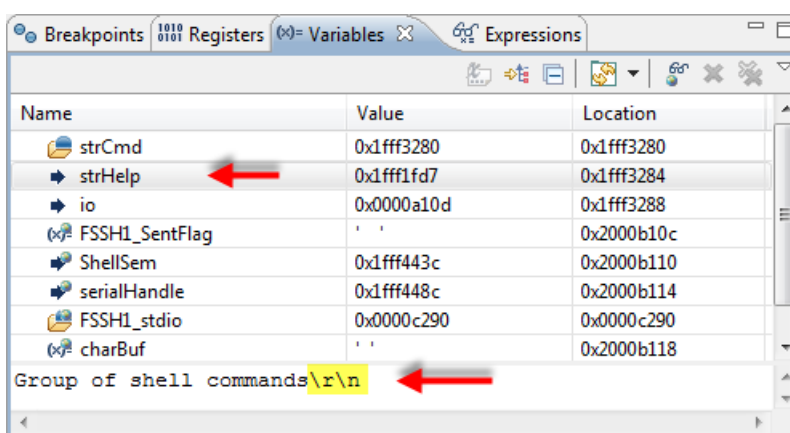
- You can view the function name behind a function pointer. The following figure shows an example:





**Figure 7-83. Details Panel with Function Pointer**

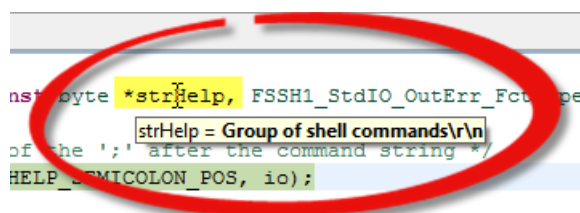
- You can view the non-printable characters as well. The following figure shows an example:



**Figure 7-84. Details Panel with Non-printable Characters**

- Using Editor view:

In the Editor view, move the mouse cursor over the variable in the source code, the tool tip shows the respective string. The following figure shows an example:



**Figure 7-85. Editor View - String Variable**

### 7.5.37 I am debugging the strings using Variables view, but instead of the string, only the address of the first

## character or the location is shown in the preview panel. What do I do?

This is a limitation of Eclipse based CodeWarrior IDE that arrays are not displayed as strings. Further more, the **Variables** view shows the strings only for `unsigned char` types, or if the plain `char` is `unsigned`.

To overcome this problem you can use the Editor view, and move the mouse cursor over the variable in the source code to view the string.

### 7.5.38 How do I view the declared type of the variables in the Variables view?

To view the declared type of the variables in the **Variables** view:

1. Open the View menu by clicking the *inverted triangle*.
2. Select **Layout > Select Columns** from the View menu.

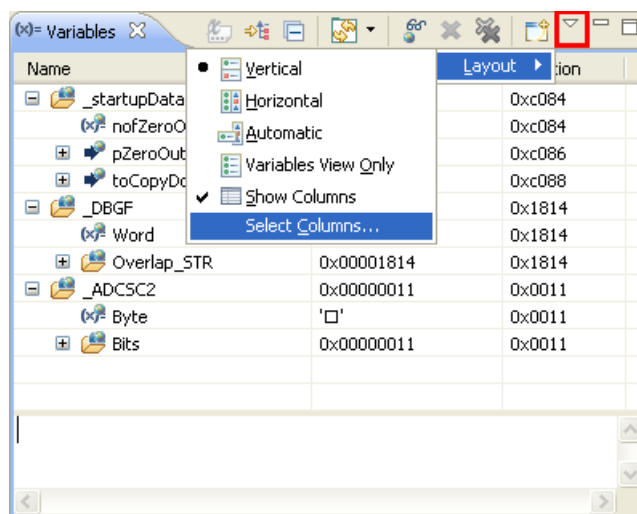
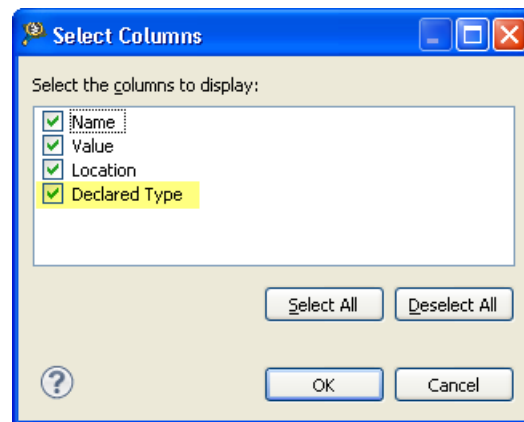


Figure 7-86. Variables View - View Menu

The **Select Columns** dialog box appears.

3. Select the **Declared Type** option, as the following figure shows:



**Figure 7-87. Select Columns Dialog Box**

4. Click **OK** .

The **Declared Type** column appears in the **Variables** view, listing the variable declared types. The following figure shows an example:

The image shows the 'Variables' view in a debugger. It displays a table with four columns: Name, Value, Location, and Declared Type. The 'Declared Type' column is highlighted in yellow. The table contains the following data:

Name	Value	Location	Declared Type
_ADCSC2	0x00000011	0x0011	volatile ADCSC2STR
Byte	'□'	0x0011	unsigned char
Bits	0x00000011	0x0011	
	'□'	0x0011	unsigned char : 1
	'□'	0x0011	unsigned char : 1
	'□'	0x0011	unsigned char : 1
	'□'	0x0011	unsigned char : 1
ACFC	'□'	0x0011	unsigned char : 1
ACFE	'□'	0x0011	unsigned char : 1
ADTF	'□'	0x0011	unsigned char : 1
ADAC	'□'	0x0011	unsigned char : 1

**Figure 7-88. Variables View with Declared Type Column**



## Chapter 8

# Debugger Shell

In this chapter, you find Frequently Asked Questions (FAQs) related to the Debugger Shell.

- [FAQs - Debugger Shell](#)

### 8.1 FAQs - Debugger Shell

In this topic:

- [How can I find the default value of memory width? How can I change the value if I want to?](#)
- [Can I change the Program Counter \(pc\) value?](#)
- [Does the config page off option still available in the Eclipse-based CodeWarrior IDE?](#)
- [Is there a way to accelerate the execution speed and turn off the printing when I use a lot of mem commands in the debugger shell?](#)
- [Is there any printf command in Debugger shell or command line?](#)
- [How can I do test automation in the Debugger Shell?](#)
- [How can I do CodeWarrior Flash programming from DOS Shell?](#)
- [How do I create shell for Kinetis KL25Z freedom board?](#)

#### 8.1.1 How can I find the default value of memory width? How can I change the value if I want to?

To find the default value for memory width, follow steps below in debug perspective:

1. From Debug Perspective toolbar, select **Window > Show View > Debugger Shell** .

The **Debugger Shell** window appears docked to the lower right hand side of debug perspective.

2. In debugger shell, type `config`.

The current configurations settings appear. If you want to change the memory width, type `config MemWidth <value>`. To make sure settings were set, type `config`. The current configuration settings appear.

## 8.1.2 Can I change the Program Counter (pc) value?

Yes. The pc value can be changed through the debugger shell. In order to change the pc value through the debugger shell follow steps below in debug perspective:

1. From the debug perspective, select **Window > Show View > Debugger Shell**.

The **Debugger Shell** window appears docked to the lower right hand side of debug perspective.

2. In debugger shell, type `reg pc`.

The present value of pc is displayed.

3. In debugger shell, type `reg pc=0x10000000`.

This command changes pc value to `0x10000000`.

4. In debugger shell, type `reg pc`.

The current value of pc is displayed; which in this case should be `PC=0x10000000`.

## 8.1.3 Does the config page off option still available in the Eclipse-based CodeWarrior IDE?

No. This option is not implemented. If a tcl script needs to be run and you want to advance the debugger shell display, select **Enabling Page** icon from the toolbar. By selecting this option, the debugger shell automatically advances the display without having to press the space bar.

### 8.1.4 Is there a way to accelerate the execution speed and turn off the printing when I use a lot of mem commands in the debugger shell?

Yes. In order to turn off the printing add the `-np` to the `mem` command in your script file. It should look like the following:

```
mem -np
```

### 8.1.5 Is there any printf command in Debugger shell or command line?

For logging the variable to a file in decimal format, follow the steps listed below:

1. Start a log session in the **Debugger Shell**, for this use the `log` command by specifying the " help log" in the **Debugger Shell**. For example,

```
log s c:\ logfile.log
```

2. Use the `display` command to log the results to that file. For logging in decimal format, use the command

```
display counter %d
```

3. After you finished logging the variable value, use the `log off` command to close the session log file.

You can also add a Debugger Shell Action to a breakpoint. For this, follow the steps listed below:

1. Right-click the desired breakpoint and select **Breakpoint Properties** from the context-menu.

The **Properties for** dialog box appears.

2. Select **Actions** from the list in the left panel.
3. Click the **New** button in the right panel.

The **New Breakpoint Action** dialog box appears.

4. Select the **Debugger Shell Actions** from the **Action type** drop-down list.

The Debugger Shell action will be added to the breakpoint.

## 8.1.6 How can I do test automation in the Debugger Shell?

Debugging is generally done manually, but for testing and automation you can develop scripts to run in an automated fashion. For this you have to use the **Debugger Shell** as the command-line debugger and use the `TCL` as the scripting language. You can perform automation and scripting with the debugger from basic access to memory, to stepping and controlling the execution up to programming the flash memory.

The Debugger shell uses `TCL` as scripting language. And instead typing in commands, you can save them into a file (for example, `myScript.tcl`) and execute it with the `source` command:

```
source myScript.tcl
```

For creating a test automation, you have to perform the following steps:

1. Start the debugging session.
2. Set the breakpoints at specific locations.
3. Run the program.
4. Compare the variables with the expected values.
5. Print the number of errors.
6. Exit the debugging session.

Before creating a `TCL` script file, execute the commands in the **Debugger Shell** view, as listed below:

1. Start the debugger:

```
debug
```

2. Set a breakpoint in the test program. For example, an *automatic* breakpoint on line 9, column 1:

```
bp -auto Test_LED.c 9 1
```

3. Run the program:

```
# run program
```

```
go
```

The debugger stops on line 9, as the following figure shows:



```

01 #include <stdint.h>
02 #include "Test_LED.h"
03 #include "LED1.h"
04
05 static volatile uint32_t val;
06 static volatile uint32_t reg;
07
08 void TestLED(void) {
09     reg = (GPIO1_GetPortValue(NULL)&GPIO1_LED_R_MASK); /* get raw register value */
10     val = LED1_Get(); /* LED shall be off, so val should be zero */
11     LED1_On();
12     val = LED1_Get(); /* LED shall be on, so val should be one */
13     LED1_Off();
14     val = LED1_Get(); /* LED shall be off, so val should be zero */
15 }
    
```

**Figure 8-1. Running Application with Auto Breakpoint**

4. Step over the breakpoint (line 9):

```
step over
```

The debugger jumps to line 10, as the following figure shows:

```

01 #include <stdint.h>
02 #include "Test_LED.h"
03 #include "LED1.h"
04
05 static volatile uint32_t val;
06 static volatile uint32_t reg;
07
08 void TestLED(void) {
09     reg = (GPIO1_GetPortValue(NULL)&GPIO1_LED_R_MASK); /* get raw register value */
10     val = LED1_Get(); /* LED shall be off, so val should be zero */
11     LED1_On();
12     val = LED1_Get(); /* LED shall be on, so val should be one */
13     LED1_Off();
14     val = LED1_Get(); /* LED shall be off, so val should be zero */
15 }
    
```

**Figure 8-2. Running Application - Stepping Over the Breakpoint**

5. Count the number of errors in your test script use an error counter:

```
set test_nofErrors 0
```

6. Check the value of `reg'. For this, set up a scripting variable `test_expected` and initialize it with a value of `0x40000` (the expected LED port value):

```
set test_expected 0x40000
```

7. In a similar way set up a variable to read the `reg` variable of the application and store it into `test_val`:

```
set test_val [evaluate reg]
```

8. The single line code, as listed below compares the actual value with the expected value. If it does not match, it increases the error counter. To reference the TCL variables use the `$` prefix:

```
if {$test_expected != $test_val} {set test_nofErrors [expr {$test_nofErrors + 1}]}
```

9. Clear all breakpoints set:

```
bp all off
```

10. Print the number of errors:

```
puts "Number of errors: $test_nofErrors"
```

## 11. Exit the debugging session:

```
kill
```

You can use the `wait` command to suspend things to follow the flow while the program runs. For example, to wait for one second:

```
wait 1000
```

Now create a script file (`myScript.tcl`), as listed below:

### Listing: Example Automated Test Script - `myScript.tcl`

```
01 # start debug session using current active debug configuration
02 debug;
03
04 # wait for 1 second
05 wait 1000;
06
07 # set auto breakpoint in Test_Led.c at line 9, column 1
08 bp -auto Test_LED.c 9 1;
09
10 # resume execution, run to breakpoint
11 go;
12
13 # wait for the breakpoint to get hit for 1 second
14 wait 1000;
15
16 # remove all breakpoints
17 bp all off;
18
19 # step over line of code
20 step over;
21
22 # define error counter variable
23 set test_nofErrors 0;
24
25 #####
26 # TEST: register value shall be 0x40000
```

```

27 #####
28 # define test expected value variable to the expected value
29 set test_expected 0x40000;
30
31 # evaluate 'reg' variable and define a variable with it
32 set test_val [evaluate reg];
33
34 #compare the expected value with the actual value
35 if {$test_expected != $test_val} {set test_nofErrors [expr
36 {$test_nofErrors + 1]}};
37 #####
38 # TEST: LED shall be off
39 #####
40 step over;
41 set test_expected 0
42 set test_val [evaluate val];
43 if {$test_expected != $test_val} {set test_nofErrors [expr
44 {$test_nofErrors + 1]}};
45 #####
46 # TEST: LED shall be on
47 #####
48 step over;
49 step over;
50 set test_expected 1
51 set test_val [evaluate val];
52 if {$test_expected != $test_val} {set test_nofErrors [expr
53 {$test_nofErrors + 1]}};
54 #####
55 # print number of errors
56 puts "*** Number of errors: $test_nofErrors ***";
57
58 # terminate the debug session
59 kill;
    
```

The following figure shows the output in the **Debugger Shell**:

```

%>source c:\tmp\myScript.tcl
cmdwin::eclipse::debug
Launching {1}: 0% complete
: 0% complete
: 7% complete
Launching application: 7% complete
Creating debug session: 7% complete
Launching executable: 7% complete
Preparing executable: 7% complete
Loading symbolic information: 7% complete
Finished loading symbolic information: 7% complete
Preparing executable: 7% complete
thread break: Stopped, 0x0, 0x0, cpuARMLittle, Freedom_PEx.elf (state, tid, pid, cpu, target)
thread set: Stopped, 0x0, 0x0, cpuARMLittle, Freedom_PEx.elf (state, tid, pid, cpu, target)
Executing Initialization File: C:\tmp\wsp_10.3\Freedom_PEx/Project_Settings/Debugger/init_kinetis.tcl:
7% complete
thread break: Stopped, 0x0, 0x0, cpuARMLittle, Freedom_PEx.elf (state, tid, pid, cpu, target)
source c:\\tmp\\myScript.tcl
Downloading 8300 bytes...: 7% complete
Downloading binary to target...: 7% complete
Downloading binary to target...: 100% complete
thread break: Stopped, 0x0, 0x0, cpuARMLittle, Freedom_PEx.elf (state, tid, pid, cpu, target)
cmdwin::wait 1000
bp -auto Test_LED.c 9 1
  id instance      address      type  enabled?  process  description
  #203          #1 x:0x0000060c -auto  ENABLED   0x0     Test_LED.c, line 9, TestLED [Freedom_PEx.elf]
cmdwin::go
thread break: Stopped, 0x0, 0x0, cpuARMLittle, Freedom_PEx.elf (state, tid, pid, cpu, target)
cmdwin::wait 1000
bp all off
cmdwin::step over
thread break: Stopped, 0x0, 0x0, cpuARMLittle, Freedom_PEx.elf (state, tid, pid, cpu, target)
evaluate reg
262144
cmdwin::step over
thread break: Stopped, 0x0, 0x0, cpuARMLittle, Freedom_PEx.elf (state, tid, pid, cpu, target)
evaluate val
0
cmdwin::step over
thread break: Stopped, 0x0, 0x0, cpuARMLittle, Freedom_PEx.elf (state, tid, pid, cpu, target)
cmdwin::step over
thread break: Stopped, 0x0, 0x0, cpuARMLittle, Freedom_PEx.elf (state, tid, pid, cpu, target)
evaluate val
1
*** Number of errors: 0 ***
cmdwin::kill
thread exit: Stopped, 0x0, 0x0, cpuARMLittle, Freedom_PEx.elf (state, tid, pid, cpu, target)
%>

```

**Figure 8-3. Debugger Shell - myScript.tcl**

### 8.1.7 How can I do CodeWarrior Flash programming from DOS Shell?

To do Flash programming using the DOS shell:

1. Add `quitIDE` to your script to flash a board. The following listing shows an example:

**Listing: Example - Standalone.tcl Script File**

```
01#####
02# Standalone.tcl: a script file to flash my application

03#####

04

05# in any case, disconnect an existing debug connection
06fl::disconnect

07

08# set launch configuration:
09fl::target -lc "LC for Simple Flash"

10

11# set Set the target RAM buffer for downloading image data:
12fl::target -b 0x20000000 0xffff

13

14# switch off verify and logging:
15fl::target -v off -l off

16

17# select flash device, organization and memory range:
18cmdwin::fl::device -d "CFM_MCF5225X_512" -o "256Kx16x1" -a 0x0
0x7ffff

19

20# specify target file, auto detect format, range settings on followed
by the flash range, offset settings off
21cmdwin::fl::image -f "C:\\tmp\\wsp_StandaloneFlsh\\Application.S19"
-t "Auto Detect" -re on -r 0x0 0x7ffff -oe off

22

23# now erase the flash...
24cmdwin::fl::erase image

25

26# ... followed by writing the application to flash:
27cmdwin::fl::write

28

29# disconnect connection
30fl::disconnect

31

32# exit Eclipse IDE

33
quitIDE
```

2. Open the DOS shell.
3. Use the following code at the DOS command prompt,

```
"<CWInstallDir>\eclipse\cwide.exe" -data "c:\tmp\wsp_StandaloneFlash" -vmargsplus -
Dcw.script="c:\tmp\wsp_StandaloneFlash\standalone.tcl"
```

where,

< CWInstallDir> is the installation directory of your CodeWarrior software,

-data argument specifies the workspace, and

-vmargsplus argument executes the Debugger Shell script from the command-line

The DOS shell launches the CodeWarrior IDE and executes the script file to flash the board.

### 8.1.8 How do I create shell for Kinetis KL25Z freedom board?

The Kinetis KL25Z freedom board has no physical UART/RS-232 on-board, but this is not a problem anymore as you can work shell connection through the P&E OpenSDA USB CDC connection. Although the OpenSDA firmware on the board is preliminary, it works very well.

To create the shell for Kinetis KL25Z freedom board, use FSShell with RingBuffer. FSShell is file system shell with command line interface. For developing the application, refactor the FSShell component a bit: the component uses its own buffer management. Having a more universal ring buffer implementation in RingBufferUInt8, use this for the FSShell.

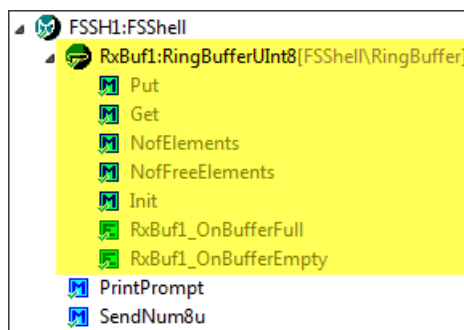


Figure 8-4. FSShell with Ringbuffer

Add a new routine `ReadAndParseLine()` an 'append' buffer management. This is useful if the terminal is sending character by character (and not as a full string). Use the Tera Term to execute the application, as Eclipse Terminal view and Termite does not work well and blocks after few seconds.

## Listing: FSShell Scripting

```

01/*
02** =====
03**      Method      :  FSSH1_ReadAndParseLine (component FSShell)
04**
05**      Description :
06**          Reads characters from the default input channel and
07**          appends
08**          it to the buffer. Once a new line has been detected, the
09**          line will be parsed.
10**      Parameters  :
11**          NAME          - DESCRIPTION
12**          * cmdBuf      - Pointer to buffer provided by the
13**                        caller where to store the command to read
14**                        in. Characters will be appended, so make
15**                        sure string buffer is initialized with a
16**                        zero byte at the beginning.
17**          cmdBufSize   - Size of buffer
18**          * io          - Pointer to I/O channels to be used
19**          parseCallback - callback provided by
20**                        the user application to parse user
21**                        commands.
22**                        If not used, NULL can be passed.
23**      Returns      :
24**          ---          - Error code
25**      =====
26*/
27
28byte FSSH1_ReadAndParseLine(byte *cmdBuf, size_t cmdBufSize,
29FSSH1_ConstStdIOType *io, FSSH1_ParseCommandCallback parseCallback)
30{
31    byte res = ERR_OK;
32    size_t len;
33
34    len = UTIL1_strlen((const char*)cmdBuf);
35    if (FSSH1_ReadLine(cmdBuf+len, cmdBufSize-len, io)) {
36        len = UTIL1_strlen((const char*)cmdBuf); /* length of buffer
37        string */

```

```

33     if (len==0) { /* error case */
34         return ERR_FAILED;
35     } else if (len==1 && (cmdBuf[0]=='\r' || cmdBuf[0]=='\r')) { /*
eat preceding newline characters */
36         cmdBuf[0] = '\0';
37     }
38     if (len>=cmdBufSize-1) { /* buffer overflow? Parse what
we have, will be likely return an error */
39         res = FSSH1_ParseCommand(cmdBuf, io, parseCallback);
40         cmdBuf[0] = '\0'; /* start again */
41         res = ERR_OVERFLOW;
42     } else if (cmdBuf[len-1]=='\n' || cmdBuf[len-1]=='\r') { /* line
end: parse command */
43         cmdBuf[len-1] = '\0'; /* remove line end character
for parser */
44         res = FSSH1_ParseCommand(cmdBuf, io, parseCallback);
45         cmdBuf[0] = '\0'; /* start again */
46     } else {
47         /* continue to append to buffer */
48     }
49 }
50 return res;
51}

```

Then implement a task providing the shell interface, as the following listing shows:

### Listing: Implementing Task

```

01static portTASK_FUNCTION(ShellTask, pvParameters) {
02unsigned char cmd_buf[32];
03
04 (void)pvParameters;
05 cmd_buf[0] = '\0';
06 FSSH1_Init();
07 (void)FSSH1_ParseCommand((const unsigned char*)FSSH1_CMD_HELP,
FSSH1_GetStdio(), ParseCommand);
08 for(;;) {
09     (void)FSSH1_ReadAndParseLine(cmd_buf, sizeof(cmd_buf),
FSSH1_GetStdio(), ParseCommand /* local cmd parser */);
10     FRTOS1_vTaskDelay(50/portTICK_RATE_MS);

```

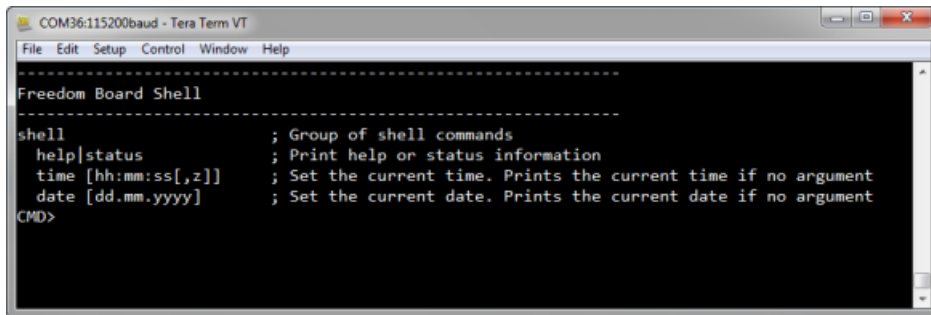


```

11  LED2_Neg();
12  };
13}
14
15void SHELL_Init(void) {
16  if (FRTOS1_xTaskCreate(ShellTask, (signed portCHAR *)"Shell",
configMINIMAL_STACK_SIZE+350, NULL, tskIDLE_PRIORITY+1, NULL) !=
pdPASS) {
17    for(;;){} /* error */
18  }
19}

```

After reset, the board shows the following menu. The same menu is shown if you type in `help` command:



**Figure 8-5. Freedom Board Shell**

Date and Time commands are provided automatically through the FSShell:

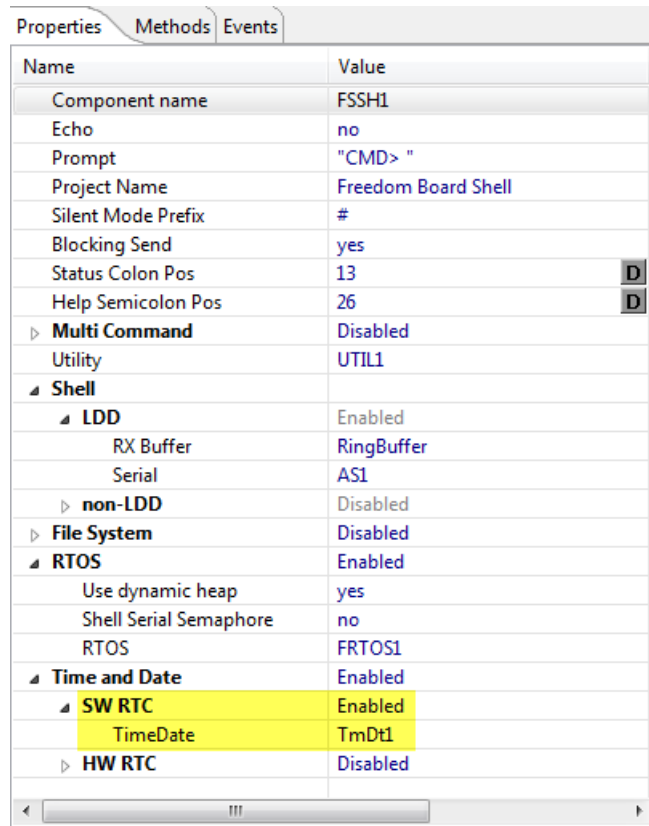


Figure 8-6. FSShell with TimeDate

Use `status` command to view the Date/Time and other information:

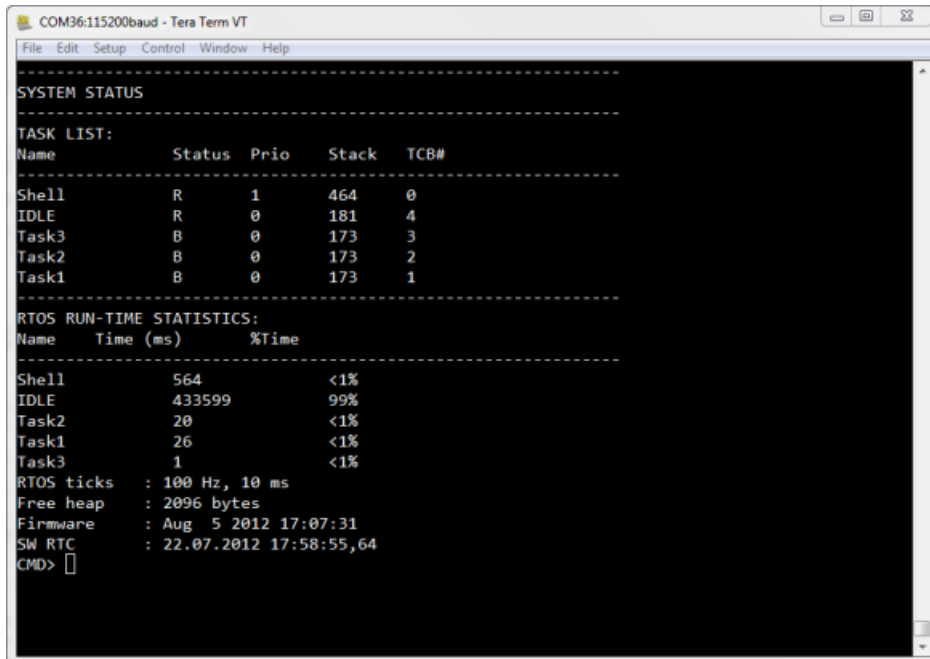


Figure 8-7. System Status

It shows all RTOS tasks with their status, along with performance information about each task.



## Chapter 9

# Profiling and Analysis Tool

In this chapter, you find Frequently Asked Questions (FAQs) related to the CodeWarrior profiling tool for the HCS08, ColdFire V1, and Kinetis targets.

- [FAQs - Profiling](#)

### 9.1 FAQs - Profiling

In this topic:

- [What do we mean by trigger A, trigger B, and trigger C?](#)
- [What are start and stop triggers/tracepoints?](#)
- [How do I set triggers A, B and C?](#)
- [How do I enable trace and profiling?](#)
- [How do I view the triggers/tracepoints that I have set?](#)
- [How do I enable and disable triggers?](#)
- [What is the difference between the Toggle Trace Trigger A/B and Enable/Disable Tracepoint options?](#)
- [Which option do I select if I want to add a trigger?](#)
- [Which option do I select if I want to skip a trigger still keeping it in my trigger list?](#)
- [What if I want to delete a trigger?](#)
- [How do I collect data?](#)
- [How do I view trace, critical code, timeline, performance, and call tree data?](#)
- [What is Timeline?](#)
- [What is Performance data?](#)
- [What is Call Tree data?](#)
- [What does Critical Code data display?](#)
- [What is the use of Selection Mode in TraceTimelineEditor?](#)
- [How do I export critical code and performance data in a CSV file?](#)
- [How can I save my trace results for later use?](#)

- When do we use the Halt the Target when Trace Buffer gets Full option?
- When do we use the Break on Trigger Hit and Break on FIFO Full options?
- When do I use the "Keep Last Buffer Before Trigger" option?
- What are memory access triggers?
- How to set a trigger from the Disassembly view?
- What is LOOP1 Mode?
- What is Profile-Only mode?
- What is the difference between Automatic and Continuous mode?
- What is the purpose of Resume and Suspend buttons in the Profile Results view?
- How do I add a new address tracepoint while debugging a project?
- Are there any limitations to use the debugger and the profiling tool (with tracepoints) simultaneously?
- How are ColdFire V2 - V4 targets different from ColdFire V1 target?
- Is trace feature supported on OSJTAG?
- Which derivatives of the MPC56xx target support tracing?
- What is Data Visualization?
- Can I set analysispoints for data visualization?

Following are the FAQs related to Kinetis target only:

- What are Hardware and Software tracepoints?
- How do I set Hardware/Software tracepoints?
- What is Embedded Trace Macrocell (ETM)?
- What is Instrumentation Trace Macrocell (ITM)?
- What is Embedded Trace Buffer (ETB)?
- What is J-Trace? How do I collect data using J-Trace?
- How can I enable software tracepoints automatically on Kinetis target?

### 9.1.1 What do we mean by trigger A, trigger B, and trigger C?

In the HCS08 target, A and B are two address comparators referred as triggers that make one big trigger. The trace collection starts or ends depending on the *From* or *Until* trigger selected along with a combination of actions involving A and B.

In the ColdFire V1 target, the triggers, A, B, and C are used to start and stop the trace collection. The triggers, A and B are set on a function address and trigger C is set on a variable address.

## 9.1.2 What are start and stop triggers/tracepoints?

A start trigger/tracepoint starts the trace collection from the address where it is set. A stop trigger/tracepoint stops the trace collection at the address where it is set. You can set trigger A as start trigger and trigger B as stop trigger and vice-versa. A trigger is set on the HCS08 or ColdFire V1 target, and a tracepoint is set on the Kinetis target.

## 9.1.3 How do I set triggers A, B and C?

To set triggers A or B in a source or assembly file:

1. Right-click on the marker bar of source file.  
A context menu appears.
2. Select **Trace Triggers > Toggle Trace Trigger A** or **Toggle Trace Trigger B** (as required) at the line where you want the trigger.

To set trigger C (trigger C is set on a variable address):

1. Right-click on the **Name** column of the **Variables** view against the variable on which you want to set trigger C.
2. Select **Toggle Triggers > Set CFv1 Trace Trigger C** option from the context menu.

The **Set Trigger Properties** dialog box appears.

3. Select the **Read/Write** option from the **Access** drop-down list.
4. Click **OK**.

This is how you set triggers A and B in a source or assembly file and trigger C on a variable address.

### NOTE

Before setting triggers, make sure that you have enabled trace and profiling.

## 9.1.4 How do I enable trace and profiling?

To enable trace and profiling:

1. Open the **Debug Configurations** dialog box.

2. Select your project in the tree structure on the left-hand side.
3. Select the **Trace and Profile** tab.
4. Check the **Enable Trace and Profile** checkbox.

This enables the trace and profiling of your project.

### 9.1.5 How do I view the triggers/tracepoints that I have set?

To view triggers:

1. Select **Window -> Show View -> Other -> Software Analysis -> Analysispoints** from the IDE menu bar.

The **Analysispoints** view appears displaying the trigger that is set, the address on which it is set and the memory information.

### 9.1.6 How do I enable and disable triggers?

To enable/disable triggers, follow one of the three methods given below:

1. Right-click on the marker bar where triggers are already set and in enabled/disabled state, select the **Enable/Disable Tracepoint** option from the context menu.

-or-

1. Right-click on the selected attribute in the **Analysispoints** view, and select **Disable/Enable** option. The unchecked attribute indicates the disabled trigger.

-or-

1. Click the **Ignore all** option in the **Analysispoints** view (displayed on the top-right) to disable all the triggers.
2. Click **Ignore All** again to enable the triggers.

This is how you enable/disable triggers in your application.

### 9.1.7 What is the difference between the Toggle Trace Trigger A/B and Enable/Disable Tracepoint options?



The **Toggle Trace Trigger A/B** option adds or removes triggers. The **Enable/Disable Tracepoint** option makes triggers either active or inactive.

### 9.1.8 Which option do I select if I want to add a trigger?

To add a trigger, select **Trace Triggers >Toggle Trace Trigger A/B** from the context menu that appears after you right-click on the marker bar of the source editor.

### 9.1.9 Which option do I select if I want to skip a trigger still keeping it in my trigger list?

If you want to skip a trigger but still want to have it in your trigger list then select the **Trace Triggers > Disable Tracepoint** option.

### 9.1.10 What if I want to delete a trigger?

If you want to delete a trigger from the source code, select the **Trace Triggers >Toggle Trace Trigger A/B** option from the context menu that appears when you right-click on the marker bar.

### 9.1.11 How do I collect data?

To collect trace data:

1. Enable trace and profiling in the **Debug Configurations** dialog box.
2. Debug your application.
3. Click **Resume** to resume execution and begin measurement.
4. Let the application run for several seconds.
5. Click **Suspend** .

The data gets collected in the trace buffer.

## 9.1.12 How do I view trace, critical code, timeline, performance, and call tree data?

To view collected data:

1. From the IDE menu bar, select **Profiler > KinetisTrace and Profile Results** .

The **Profile Results** view appears.

2. Expand the project name.

The data source is listed under the project name along with the hyperlinks to the **Trace** , **Timeline** , **Critical Code** , **Performance** , and **Call Tree** results.

3. Double-click on the hyperlinks to open the corresponding viewer.

The results appear in **Trace Data** , **Timeline** , **Critical Code** , **Performance** , and **Call Tree** viewers.

## 9.1.13 What is Timeline?

Timeline represents the graphical data that appears in the **TraceTimelineEditor** viewer when trace is collected. The timeline data displays the functions that are executed in the application and the number of cycles each function takes when the application is run. Timeline appears as a hyperlink in the **Profile Results** view.

## 9.1.14 What does Critical Code data display?

The critical code data displays the summarized data of a function in a tabular form, such as name of the function that is executed, start address of the function, number of lines executed in the function, and number of clock cycles taken by the function.

The critical code data is displayed in the **Critical Code** viewer that is divided into two parts. The upper part displays the summary of the functions, and the lower part displays the statistical details of all the instructions executed in a function.

## 9.1.15 What is Performance data?

The performance data includes the metric and invocation information for each function that executes in the application. The performance data during measurement enables you to compare the relative efficiencies of various portions of your target program. Both exclusive and inclusive timing measurements are provided in the performance data.

The parent-child calling relationships between your program's functions are also provided. Each function pair consists of a caller and a callee with data provided for each.

### 9.1.16 What is Call Tree data?

The Call Tree data shows the general application flow in a hierarchical tree in which statistics are displayed for each function.

### 9.1.17 What is the use of Selection Mode in TraceTimelineEditor?

The **Selection Mode** is used to set the cursor for reference in the function bars to let you compute the difference between the clock cycles taken by an instruction.

To mark a point in the bar:

1. Click **Selection Mode** in the **TraceTimelineEditor** .
2. Click on the bar where you want to mark the point.

A yellow vertical line appears displaying the number of cycles at that point.

3. Right-click on another point in the bar.

A red vertical line appears displaying the number of cycles at that point along with the difference of cycles between two marked points.

### 9.1.18 How do I export critical code and performance data in a CSV file?

To export critical code and performance data:

1. Click  in the Critical Code/Performance viewer.

2. Select the **Export the statistics above** option to export the details of the top view or the **Export the statistics below** option to export the details of the bottom view respectively.

The **Save** dialog box appears.

3. Specify the name of the file in which you want to export the trace or critical code data.

This exports the critical code and performance data in a CSV file.

### 9.1.19 How can I save my trace results for later use?

To save the trace results, right-click on the **data source** , in the **Profile Results** view and select **Save Current Results**.

### 9.1.20 When do we use the Halt the Target when Trace Buffer gets Full option?

The **Halt the Target when Trace Buffer gets Full** option acts as a breakpoint for stopping the application. You can use this option when you want to stop the application automatically when trace buffer gets full.

### 9.1.21 When do we use the Break on Trigger Hit and Break on FIFO Full options?

The **Break on Trigger Hit** option suspends the application automatically when the trigger condition is met.

The **Break on FIFO Full** option suspends the application automatically when buffer gets full.

### 9.1.22 When do I use the "Keep Last Buffer Before Trigger" option?

You can use the **Keep Last Buffer Before Trigger** option to overwrite the trace buffer during trace collection before the trigger is hit. When trigger is hit, trace starts collecting, gets appended to the existing buffer, and only the last part of the buffer is displayed in the **Trace Data** viewer.

### 9.1.23 What are memory access triggers?

The memory access triggers allow memory access to both variables and instructions. A memory access trigger if set on an instruction fires when the instruction is fetched from the memory. A memory access trigger if set on a variable fires when the variable is fetched from the memory or when the variable is written back to the memory.

### 9.1.24 How to set a trigger from the Disassembly view?

To set a trigger from the **Disassembly** view:

1. Open the **Debug Configurations** dialog box and enable trace and profiling.
2. Specify the trigger conditions as required.
3. Save the settings and debug your project.
4. After the application halts, open the **Disassembly** view and scroll to the address line of the function or instruction where you want to set the trigger.
5. Right-click on the marker bar corresponding to that address line and select **Trace Triggers >Toggle Trace Trigger A/B** .
6. Resume your application to collect trace.

This is how you set a trigger from the **Disassembly** view.

### 9.1.25 What is LOOP1 Mode?

The **LOOP1Mode** writes a register to allow the hardware to use the C comparator and not store duplicate addresses in trace. In **LOOP1** capture mode, the addresses for instructions executed repeatedly, for example, loops with no change of flow instructions and recursive calls, are stored and showed in trace only once.

## 9.1.26 What is Profile-Only mode?

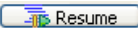
The **Profile-Only** mode does not collect the trace data; it only profiles the data. Trace is empty in this mode; you can only see the profiling information in the **Critical Code Data** viewer.

## 9.1.27 What is the difference between Automatic and Continuous mode?

In the **Automatic** mode, the entries in the buffer start overwriting without interruption when the data reaches at the end of the buffer. If there is more trace data than the size of the buffer, the old entries are overwritten.

The **Continuous** mode collects the trace data continuously. The trace buffer is read, processed, and emptied periodically, so that the **Trace Data** viewer can collect all the trace records generated by the application.


## 9.1.28 What is the purpose of Resume and Suspend buttons in the Profile Results view?

You can control the generation of trace from the **Profile Results** view using the **Resume/Suspend** toggle button. The same button is used to start or stop the trace. This toggle button appears on launching the debug session of an application. The default status of trace collection is ON. Therefore, when the application is debugged, the **Suspend** toggle button appears next to the data source. When clicked, the button toggles to  .

The toggle button disappears when you click **Resume** in the **Debug** view or terminate the debug session. After clicking **Suspend** in the **Debug** view, it is visible again with the last selected status.

## 9.1.29 How do I add a new address tracepoint while debugging a project?

You can use the **Analysispoints** view to add a tracepoint on the address of an instruction while the debug session is running. To add an address tracepoint:

1. Open the **Analysispoints** view.
2. Click the **Add new tracepoint** icon  to display the **Add new tracepoint** dialog box.
3. Select the project, type, and action of the tracepoint from the respective text boxes.
4. Enter the address where you want to set the tracepoint in the **Address** text box.
5. Click **OK**.

The tracepoint is set and appears in the **Analysispoints** view.

### 9.1.30 Are there any limitations to use the debugger and the profiling tool (with tracepoints) simultaneously?

Yes, both debugger and profiling need hardware resources to work. This might lead to shortage of hardware resources and the debugger might not work fully when profiling is enabled. Therefore, it is recommended not to use debugger breakpoints and watchpoints while using the profiling tool.

However, you can work upon this limitation by following the rules given below:

- If you are profiling without tracepoints, you can use all debugger features, that is four breakpoints and one watchpoint.
- If you are profiling with tracepoints, only two breakpoints can be set; and no watchpoint.
- If you are profiling in **Expert** mode, no debugger breakpoints and watchpoint can be used.

### 9.1.31 How are ColdFire V2 - V4 targets different from ColdFire V1 target?

The ColdFire V2 - V4 targets collect the profiling information by using the profiling system. These targets do not have the hardware capability to collect trace. The ColdFire V1 target perform tracing and profiling using the target hardware. The profiling system consists of three main components:

- The statically-linked code library of compiled code containing the profiler

- An Application Programming Interface (API) to control the profiler
- The Simple Profiler Viewer to view and analyze the profile results

### 9.1.32 Is trace feature supported on OSJTAG?

It is recommended to not use OSJTAG for trace capability, as it slows down your system. You should use Mutlilink Universal instead.

### 9.1.33 Which derivatives of the MPC56xx target support tracing?

The MPC5668G and MPC5668E derivatives of the MPC56xx target have two e200 cores, e200z6 (Core 0) and e200z0 (Core 1) in which only e200z6 core provides tracing capability.

### 9.1.34 What is Data Visualization?

Data visualization allows you to visualize during runtime the evolution of certain application data, such as variables, registers, memory, which represent parameters defining the functionality of certain equipment. Data Visualization samples the values of registers, variables and/or raw memory data as the application is running and displays the collected data in the form of a chart. Data Visualization is available for DSC architectures only.

### 9.1.35 Can I set analysispoints for data visualization?

Yes, you can set analysispoints in the application to select the data to be visualized. You can set two types of analysispoints, Data analysispoints (on **Memory** or **Variables** view) and Register analysispoints (on the **Registers** view), which appear in the **Analysispoints** view.

### 9.1.36 What are Hardware and Software tracepoints?



The Kinetis target supports hardware and software tracepoints for trace collection. Hardware tracepoints use hardware resources to start and stop trace. Hardware tracepoints allow only four comparators to be set for trace collection because they use DWT comparators to start or stop the trace collection. Software tracepoints on the other hand do not use hardware resources and generate interrupts from software to start and stop trace. They allow you to install infinite number of comparators for trace collection and are more intrusive.

### 9.1.37 How do I set Hardware/Software tracepoints?

To set a start hardware/software tracepoint:

1. Right-click on the marker bar of the source file at the line where you want the tracepoint.

A context menu appears.

2. Select **Toggle Trace Start Point > Software/Hardware Trace Point** .

To set a stop hardware/software tracepoint:

1. Right-click on the marker bar of the source file at the line where you want the tracepoint.

A context menu appears.

2. Select **Toggle Trace Stop Point > Software/Hardware Trace Point** .

This is how you set a start and a stop hardware/software tracepoint in your application.

### 9.1.38 What is Embedded Trace Macrocell (ETM)?

An ETM is a debug component that enables reconstruction of program execution and helps in minimizing area and reducing gate count. The main features of ETM are trace generation and triggering and filtering.

### 9.1.39 What is Instrumentation Trace Macrocell (ITM)?

ITM provides a memory-mapped register interface to allow applications to write logging/event words to the optional external Trace Port Interface Unit (TPIU). ITM supports synchronization and generation of timestamp information packets.

### 9.1.40 What is Embedded Trace Buffer (ETB)?

ETB stores data that ETM produces. ETB provides on-chip storage of trace data using a configurable sized RAM. This reduces the clock rate and removes the requirement of high-speed for collecting trace data.

### 9.1.41 What is J-Trace? How do I collect data using J-Trace?

J-Trace is a connection used for collecting data on the Kinetis target. The J-Trace probe has an internal memory buffer of 4MB where it can store trace data. It supports two modes of trace collection, TPIU (or Rawtrace) and Serial Wire Output (SWO), depending on the configuration of the processor. TPIU can output complex trace, that is both ETM and ITM. SWO is a lightweight standard and can only output ITM trace. The benefits of using J-Trace probe is the bigger memory size and the better collection speed. The drawback is that you cannot collect continuous trace, only the last 4MB of trace is stored.

To collect trace using the J-Trace probe:

1. Create a Kinetis project with **Segger J-Link** connection selected in the **Connections** screen.
2. Build the project.
3. Open the **Debug Configurations** dialog box.
4. Select your project in the tree structure on the left.
5. In the **Main** tab, click **Edit** . The **Properties** dialog box appears.
6. In the **Connection** tab, select **SWD** from the **Debug port** interface drop-down list.
7. Click **OK** .
8. Select the **Trace and Profile** tab and enable tracing and profiling.
9. Select the **JTrace** option and then select the **SWO** option.

#### NOTE

If you choose **TPIU** in the **Trace and Profile** tab for JTrace, you can select either **JTAG** or **SWD** (Serial Wire Debug) as debug port interface.

10. Click **Apply** .

11. Debug the application and collect trace.

This is how you collect trace data using the J-Trace probe.

### 9.1.42 How can I enable software tracepoints automatically on Kinetis target?

To enable software tracepoints automatically, you can use the **Software Tracepoints Support** dialog box, which appears when you add first start software tracepoint on a project in the source code. You can also invoke the dialog box by selecting the project in **CodeWarrior Projects** view, right-click on it, and select the **Profiler > Add software tracepoint support** option. For details, refer the *Profiling and Analysis Tools User Guide*.



# Index

.abs [214](#), [304](#)  
.lib [214](#)  
.metadata [125](#)  
.project [132](#)  
.tcl [52](#)  
'Initialize RSE' has encountered a problem [29](#)  
#pragma pack() [238](#)

64-bit. [24](#)  
8.04, [24](#)

## A

absolute [135](#)  
access [124](#)  
accessing [26](#)  
accidentally [107](#)  
Action [90](#)  
Active [260](#)  
add [85](#)  
adding/removing [108](#)  
address [366](#)  
addresses [299](#)  
all [136](#)  
All Items are installed [29](#)  
already [30](#)  
alternate [37](#)  
Analysis [357](#)  
another [63](#), [99](#), [126](#)  
any [31](#), [34](#)  
appear [119](#)  
application [157](#), [170](#), [284](#), [286](#)  
are [24](#), [30](#), [117](#), [135](#)  
assembly file [305](#)  
Assembly files [261](#)  
assist [41](#)  
associated [117](#)  
asterisk [52](#)  
attempt [27](#)  
Automatic mode [366](#)

## B

back [99](#)  
backup [100](#)  
bbl [63](#)  
beans [210](#)  
better visibility [79](#)  
binary [304](#)  
Binary [57](#)  
binary application [289](#)

binary file [283](#)  
binding? [112](#)  
bindings [120](#)  
block [65](#)  
bootload [157](#)  
bootloader [167](#), [169](#), [170](#), [181](#), [190](#), [195](#), [196](#), [198](#)  
bootloader sequence [197](#)  
Bootloader Shell commands [195](#)  
Break [364](#)  
breakpoint [258](#), [259](#), [261](#)  
Breakpoint [258](#), [260](#)  
breakpoints [256](#), [257](#), [261](#)  
Breakpoints [255](#)  
bricked [152](#), [161](#)  
Buffer [364](#)  
Buffer Trigger [364](#)  
build [63](#), [123](#), [281](#)  
Build [273](#)  
build configuration [211](#), [274](#), [279](#)  
building [31](#)  
building project [205](#)  
build settings [278](#)  
burner [63](#)  
Burner [61](#)  
button [132](#)

## C

C/C++ [132](#)  
call [118](#)  
Call Tree data [363](#)  
certain [130](#)  
change [38](#), [43](#), [49](#), [93](#), [112](#), [120](#), [123](#)  
change history [114](#)  
changes [48](#)  
check [24](#), [31](#)  
checkmark [258](#)  
Classic [99](#), [120](#)  
Classic CodeWarrior IDE [209](#)  
CMSIS 2.0 [237](#)  
CMSIS-DAP [171](#)  
code? [41](#), [52](#), [118](#)  
code data [362](#)  
CodeWarrior IDE [106](#), [227](#)  
CodeWarrior project [224](#)  
CodeWarrior Projects view [119](#)  
ColdFire targets [367](#)  
collect data [361](#)  
color [38](#), [49](#)  
command file [253](#)  
command line [343](#)  
command-line [303](#), [327](#)  
commands [236](#)

comment/uncomment [65](#)  
 comments [49](#)  
 compare [69](#), [71](#), [206](#), [207](#), [226](#)  
 compile [236](#)  
 compiler [31](#)  
 Compiler [235](#)  
 computer, [37](#)  
 condense [55](#)  
 config page off [342](#)  
 configuration [120](#), [124](#)  
 Configuration [273](#)  
 configuration? [63](#)  
 configurations [63](#)  
 Configurations [132](#)  
 configure [47](#), [278](#), [321](#)  
 configured bean [210](#)  
 Connection page [191](#)  
 console [123](#), [319](#)  
 contain [130](#)  
 contains [124](#), [132](#)  
 contents [51](#), [225](#)  
 Continuous mode [366](#)  
 control [83](#)  
 convert [51](#)  
 correctly, [31](#)  
 created [63](#), [119](#)  
 Critical [362](#)  
 critical code [363](#)  
 CSV file [363](#)  
 Ctrl + - [209](#)  
 Ctrl+Tab [120](#)  
 Ctrl-Click [320](#)  
 Ctrl Shift [99](#)  
 current [38](#)  
 currently [64](#)  
 current workspace [204](#), [226](#)  
 CVS [82](#), [85](#)  
 CW10.1SE [136](#)

## D

data type [237](#)  
 DE.exe [309](#)  
 debug [63](#), [130](#), [211](#), [305](#)  
 debug application [166](#)  
 debug configuration [133](#)  
 Debug Context [260](#)  
 debugger [120](#), [303](#), [317](#), [321](#), [323](#), [326](#), [367](#)  
 Debugger [255](#)  
 debugger shell [343](#)  
 Debugger shell [343](#)  
 Debugger Shell [341](#)  
 debugging [186](#), [366](#)  
 Debugging [220](#), [270](#)  
 debug session [318](#)  
 debug settings [100](#), [230](#)  
 declaration [34](#)

Declaration [99](#)  
 declarations? [37](#)  
 declared [52](#)  
 decode [159](#)  
 default [49](#)  
 default format [45](#)  
 defined [135](#)  
 definition [36](#), [40](#)  
 definitions [135](#)  
 delete [259](#)  
 Delimiters [51](#)  
 derivatives? [61](#)  
 determine [64](#), [117](#)  
 device [289](#)  
 different [126](#)  
 differentiate [46](#)  
 directly? [130](#)  
 directories [207](#)  
 directory [132](#)  
 disable [257](#), [360](#)  
 disabled [46](#)  
 disadvantages [193](#)  
 disassemble [125](#)  
 disassemble file [125](#)  
 disassembly [314](#)  
 Disassembly [365](#)  
 display [44](#)  
 distort [261](#)  
 does [46](#), [119](#)  
 Does [83](#)  
 Dot [116](#)  
 Doxygen? [116](#)  
 drivers [24](#)  
 DSP library? [237](#)  
 duplicate [120](#)  
 duplicating [120](#)

## E

each [63](#)  
 easily [97](#)  
 edit [51](#)  
 Editor [33](#), [37](#), [44](#), [52](#), [99](#), [261](#)  
 editor? [36](#), [65](#)  
 Editor? [34](#)  
 elf [214](#)  
 ELF file [213](#)  
 embedded [193](#)  
 Embedded Trace Buffer [370](#)  
 Embedded Trace Macrocell [369](#)  
 enable [359](#), [360](#), [371](#)  
 enabled [46](#)  
 encoding [134](#)  
 encountered [325](#)  
 endianness [311](#)  
 equivalent [99](#)  
 error [236](#)

- ETB [370](#)
- ETM [369](#)
- evaluated expansion [47](#)
- example, [99](#)
- example projects [210](#)
- executable [123](#)
- executed [236](#)
- existing [131](#)
- Explorer [119](#)
- export [205](#), [281](#)
- exported [282](#)
- expressions [315](#)
- extension [52](#)
- external tool [321](#)

## F

- facing [26](#)
- fails [27](#)
- FIFO [364](#)
- file? [38](#), [46](#), [117](#), [125](#), [132](#)
- file mapping [232](#)
- files [52](#), [57](#), [69](#), [71](#), [95](#), [117](#), [130](#), [134](#)
- filter [111](#)
- find [52](#), [107](#), [130](#)
- Firmware [135](#)
- flash [283](#), [286](#), [288](#), [327](#)
- FLASH [167](#)
- flashing [289](#)
- flash pages [190](#)
- flash programmer [283](#)
- Flash Programmer [282](#)
- flash programming [290](#)
- floating point format [218](#)
- folder [125](#), [225](#)
- Folder [202](#)
- folding/unfolding [55](#)
- format [306](#)
- format? [51](#)
- found. [24](#)
- FRDM-K20D50M [166](#), [193](#), [199](#)
- FRDM-KL05Z [186](#)
- FRDM-KL25Z [178](#), [198](#)
- freedom board [152](#), [163](#), [166](#), [172](#), [193](#), [199](#)
- frequently, [108](#)
- function [34](#), [36](#), [52](#), [93](#)

## G

- GBK [134](#)
- generate [57](#)
- generated [123](#)
- generates [31](#)
- get [130](#), [131](#)
- global variables [302](#), [324](#)
- graphical [116](#)
- grayed [120](#)

- GUI [259](#)

## H

- half-precision [237](#)
- Halt [364](#)
- handles [124](#)
- Hardware [368](#), [369](#)
- hardware breakpoint [260](#)
- hardware breakpoints [259](#), [260](#)
- hardware diagnostics [283](#)
- has [108](#)
- have [24](#), [31](#), [63](#), [120](#), [126](#), [132](#)
- header [34](#), [36](#), [46](#), [99](#), [117](#), [224](#)
- Help > Install New Software [29](#)
- Hex, [57](#)
- Hexadecimal [309](#)
- Hexadecimal format [315](#)
- hierarchy [118](#)
- highlights [38](#)
- history [109](#)
- However, [24](#)
- HyperTerminal [326](#)

## I

- identical [132](#)
- import [281](#)
- import/export [283](#)
- Importer [124](#)
- importing [232](#)
- improve [106](#), [108](#)
- information? [130](#)
- initial [74](#)
- initialize [153](#)
- inspect [324](#)
- install [24](#), [27](#)
- Install [29](#)
- installation [64](#)
- Installation [23](#)
- installed [24](#), [31](#)
- installed? [30](#), [31](#)
- Installing [23](#)
- instruct [135](#), [205](#)
- Instrumentation [369](#)
- Intel [57](#)
- issues [26](#)
- ITM [369](#)

## J

- J-Link [326](#)
- J-Trace [370](#)

## K

- key [37](#), [112](#), [120](#)

[keys](#) [120](#)  
[Kinetis](#) [61](#)  
[Kinetis project](#) [238](#)  
[Kinetis target](#) [371](#)  
[KL25Z Freedom Board](#) [171](#)  
[know](#) [30](#)  
[Kwikstik board](#) [325](#)

## L

[launch](#) [124](#), [321](#)  
[launch configurations](#) [133](#), [281](#), [282](#)  
[launching](#) [281](#)  
[layout view](#) [74](#)  
[libraries.](#) [31](#)  
[library](#) [214](#), [261](#)  
[linker](#) [253](#)  
[Linker](#) [253](#)  
[Linticator](#) [242](#)  
[Linux](#) [24](#)  
[local](#) [109](#), [133](#)  
[local configurations](#) [133](#)  
[local file](#) [133](#)  
[local history](#) [207](#)  
[location](#) [99](#), [135](#)  
[locations](#) [126](#)  
[logging](#) [290](#)  
[long](#) [108](#)  
[LOOP1 Mode](#) [365](#)

## M

[Macintosh](#) [37](#)  
[macro](#) [40](#), [47](#)  
[macro](#), [34](#)  
[Macrocell](#) [369](#)  
[macros](#) [47](#)  
[macros?](#) [46](#)  
[Manage](#) [132](#)  
[Management](#) [223](#)  
[mem commands](#) [343](#)  
[memory](#) [299](#)  
[memory access triggers](#) [365](#)  
[memory tasks](#) [283](#)  
[memory view](#) [300](#)  
[Memory view](#) [306](#), [311](#), [313](#)  
[memory width](#) [341](#)  
[messages](#) [123](#), [236](#)  
[Microcontrollers](#) [24](#)  
[minimize](#) [76](#)  
[Miscellaneous](#) [102](#), [296](#)  
[mismatch](#) [135](#)  
[modifications](#) [227](#)  
[modify](#) [130](#), [134](#)  
[monitor](#) [323](#)  
[MPC56xx](#) [368](#)  
[MQX3.7](#) [136](#)

[Multilink](#) [24](#)  
[multiple](#) [64](#)  
[multiple boards](#) [288](#)  
[multiple lines](#) [49](#)  
[multiple memory locations](#) [300](#)  
[multiple times](#) [289](#)

## N

[name](#) [123](#), [132](#)  
[name](#), [34](#)  
[navigate](#) [120](#)  
[necessary](#) [132](#)  
[new](#) [119](#)  
[not](#) [24](#), [119](#)  
[numbers](#) [44](#)

## O

[object code](#) [209](#)  
[one](#) [63](#)  
[ones](#) [120](#)  
[OpenSDA](#) [152](#), [161](#), [191](#)  
[option?](#) [99](#)  
[original](#) [51](#)  
[OSJTAG](#) [135](#)  
[OSJTAG?](#) [368](#)  
[out](#) [52](#), [130](#)  
[Outline](#) [130](#)  
[Outline view](#) [129](#)  
[output](#) [123](#)  
[overview](#) [130](#)

## P

[P&E](#) [135](#)  
[PARENT-COUNT-MyVariable](#) [135](#)  
[particular](#) [111](#)  
[particular function](#) [261](#)  
[party](#) [27](#)  
[patches](#) [31](#), [136](#)  
[paths](#) [135](#)  
[paths?](#) [124](#)  
[PC-lint](#) [242](#)  
[perform](#) [91](#)  
[performance](#) [106](#)  
[performance?](#) [108](#)  
[performance data](#) [363](#)  
[Performance data](#) [362](#)  
[perspective](#) [107](#), [120](#), [132](#)  
[Perspective](#) [226](#)  
[physical address](#) [259](#)  
[plugins?](#) [27](#)  
[Port](#) [135](#)  
[portions](#) [55](#)  
[port pin](#) [153](#)  
[post-build](#) [132](#)



- predefined [47](#)
- Preferences [111](#)
- preprocessor macros [208](#)
- preview [91](#)
- previous [99](#)
- printf [123](#), [343](#)
- printf() [178](#), [193](#)
- probable [135](#)
- problem? [136](#)
- Processor Expert [178](#), [210](#)
- product [27](#)
- Profile-Only mode [366](#)
- Profile Results view [366](#)
- profiling [359](#), [367](#)
- Profiling [357](#)
- Program Counter [317](#), [342](#)
- program entry point [317](#)
- project [71](#), [93](#), [107](#), [133](#), [202](#), [224](#), [225](#), [236](#), [274](#), [301](#), [366](#)
- Project [223](#)
- project? [57](#), [123](#)
- project properties [227](#)
- projects [226](#)
- Properties view [129](#)
- Property Settings [226](#)
- purpose [128](#), [129](#)

## Q

- quickly [34](#), [46](#)

## R

- RAM [284](#)
- recover [100](#)
- recursive [124](#)
- registers [298](#)
- related [224](#)
- relative [135](#)
- release. [63](#)
- Remote System [227](#)
- Rename option [120](#)
- replace [97](#)
- Replace [90](#)
- Replace action [91](#)
- representation [116](#)
- re-programming [328](#)
- Reset on Connect [316](#)
- resize [226](#)
- resolve [26](#), [31](#), [126](#), [131](#), [133](#)
- resources [74](#)
- restrict [74](#)
- retrieve [107](#)
- revert [51](#)
- roll back [48](#)
- RTSCTC [154](#)
- run [318](#)

- running [323](#)

## S

- S08 Projects [218](#)
- S19, [57](#)
- S19 Files [159](#)
- same [64](#)
- search [91](#), [95](#)
- Search [90](#)
- search window [99](#)
- secure/unsecure [286](#)
- see [123](#)
- select [99](#)
- separate directory [279](#)
- Serial [135](#)
- serial data [326](#)
- setting [111](#)
- settings [45](#), [111](#), [124](#)
- settings? [109](#)
- share [64](#), [230](#)
- shell commands [195](#)
- shortcut [99](#)
- shortcut key [99](#), [209](#)
- simple [91](#)
- sites. [26](#)
- slowed [108](#)
- software [31](#), [64](#)
- Software [23](#), [368](#), [369](#)
- software. [24](#), [108](#)
- software breakpoint [260](#)
- software flow control [154](#)
- solve [135](#), [136](#)
- source [38](#), [41](#), [46](#), [52](#), [99](#), [116–118](#), [125](#), [130](#), [224](#)
- Source [202](#)
- source code [48](#)
- source file [40](#), [51](#), [114](#), [278](#)
- source files [205–207](#), [224](#), [225](#)
- span [49](#)
- specific [95](#)
- specify [63](#)
- S-Records? [288](#)
- start [106](#), [131](#), [132](#)
- startup [155](#), [170](#)
- Stepping Backwards [220](#), [270](#)
- steps [132](#)
- still [120](#)
- store [100](#), [135](#)
- successfully [31](#)
- support [83](#)
- SWD programmer [152](#), [163](#), [172](#)
- switch [46](#), [126](#)
- systems? [83](#)

## T

- tab [43](#)

- takes [99](#)
- target [283](#), [323](#)
- target tasks [286](#)
- Tasks view [128](#)
- terminal program [154](#)
- that [38](#), [107](#), [120](#), [123](#), [131](#), [132](#)
- there [31](#)
- third [27](#)
- this [108](#), [136](#)
- this? [26](#), [31](#)
- time [108](#)
- Timeline [362](#)
- title bar [52](#)
- Toggle [258](#), [360](#)
- tool [27](#)
- toolbar [132](#)
- trace [359](#), [368](#)
- Trace [360](#), [364](#), [369](#)
- tracepoint [366](#)
- Tracepoint [360](#)
- tracepoints [359](#), [360](#), [367–369](#), [371](#)
- trace results [364](#)
- TraceTimelineEditor [363](#)
- tracing [368](#)
- trigger [358](#), [361](#), [365](#)
- Trigger [360](#), [364](#)
- trigger list [361](#)
- triggers [359](#), [360](#)
- trim [328](#)
- two [63](#)

## U

- Ubuntu [24](#)
- uninstall [27](#)
- Unix [51](#)
- update [109](#)
- updater [26](#), [31](#)
- updates [24](#)
- updates, [24](#)
- updates/patches [30](#)
- updates. [136](#)
- Updating [23](#)
- USBDM [186](#)
- use [61](#), [120](#), [130](#)
- used [120](#)
- users? [64](#)
- using? [107](#)
- Utility? [135](#)

## V

- values [301](#)
- variable [40](#), [93](#), [97](#), [323](#)
- variable, [34](#)
- variable declaration [320](#)
- variables [301](#), [309](#)

- Variables [270](#)
- variables view [324](#)
- version [27](#), [83](#), [107](#)
- version control [230](#)
- version control system [100](#), [230](#)
- view [116](#), [118](#)
- View [130](#)
- view. [119](#)
- views [226](#)
- Virtual [135](#)
- visibility [261](#)

## W

- warning [123](#), [236](#)
- warnings symbols [261](#)
- way [31](#)
- what [30](#), [37](#)
- What [24](#)
- when [24](#), [123](#), [131](#)
- when, [99](#)
- where [52](#)
- while [31](#)
- Why? [119](#)
- width/size? [43](#)
- window [111](#)
- window? [52](#), [76](#), [123](#)
- Windows [31](#), [119](#)
- windows? [120](#)
- with? [106](#)
- Workbench [76](#)
- Workbench window [74](#)
- Workbench Window [74](#)
- working [64](#), [106](#)
- working. [24](#)
- workspace [64](#), [108](#), [125](#), [126](#), [135](#), [202](#), [225](#), [226](#)
- workspace? [95](#), [126](#)
- workspaces [126](#)
- writing [41](#)



**How to Reach Us:**

**Home Page:**

[freescale.com](http://freescale.com)

**Web Support:**

[freescale.com/support](http://freescale.com/support)

Information in this document is provided solely to enable system and software implementers to use Freescale products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. Freescale reserves the right to make changes without further notice to any products herein.

Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. Freescale does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address: [freescale.com/SalesTermsandConditions](http://freescale.com/SalesTermsandConditions).

Freescale, the Freescale logo, CodeWarrior, ColdFire, ColdFire+, Kinetis, Processor Expert are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. All other product or service names are the property of their respective owners.

© 2010 - 2014 Freescale Semiconductor, Inc.