Sourcery CodeBench Lite

ColdFire ELF

Sourcery CodeBench Lite 2011.09-21

Getting Started

Sourcery CodeBench Lite: ColdFire ELF: Sourcery CodeBench Lite 2011.09-21: Getting Started

CodeSourcery, Inc.

Abstract

This guide explains how to install and build applications with Sourcery CodeBench Lite, CodeSourcery's customized and validated version of the GNU Toolchain. Sourcery CodeBench Lite includes everything you need for application development, including C and C++ compilers, assemblers, linkers, and libraries.

When you have finished reading this guide, you will know how to use Sourcery CodeBench from the command line.

Table of Contents

Preface

This preface introduces the Sourcery CodeBench Lite Getting Started guide. It explains the structure of this guide and describes the documentation conventions used.

1. Intended Audience

This guide is written for people who will install and/or use Sourcery CodeBench Lite. This guide provides a step-by-step guide to installing Sourcery CodeBench Lite and to building simple applications. Parts of this document assume that you have some familiarity with using the command-line interface.

2. Organization

This document is organized into the following chapters and appendices:

Chapter 1, “Quick Start”	This chapter includes a brief checklist to follow when installing and using Sourcery CodeBench Lite for the first time. You may use this chapter as an abbreviated guide to the rest of this manual.
Chapter 2, “Installation and Configuration”	This chapter describes how to download, install and configure Sourcery CodeBench Lite. This section describes the available installation options and explains how to set up your environment so that you can build applications.
Chapter 3, “Sourcery CodeBench Lite for ColdFire ELF”	This chapter contains information about using Sourcery CodeBench Lite that is specific to ColdFire ELF targets. You should read this chapter to learn how to best use Sourcery CodeBench Lite on your target system.
Chapter 4, “Using Sourcery CodeBench from the Command Line”	This chapter explains how to build applications with Sourcery CodeBench Lite using the command line. In the process of reading this chapter, you will build a simple application that you can use as a model for your own programs.
Chapter 5, “CS3™: The CodeSourcery Common Startup Code Sequence”	CS3 is CodeSourcery's low-level board support library. This chapter documents the boards supported by Sourcery CodeBench Lite and the compiler and linker options you need to use with them. It also explains how you can use and modify CS3-provided definitions for memory maps, system startup code and interrupt vectors in your own code.
Chapter 6, “Sourcery CodeBench Debug Sprite”	This chapter describes the use of the Sourcery CodeBench Debug Sprite for remote debugging. The Sprite allows you to debug programs running on a bare board without an operating system. This chapter includes information about the debugging devices and boards supported by the Sprite for ColdFire ELF.
Chapter 7, “Next Steps with Sourcery CodeBench”	This chapter describes where you can find additional documentation and information about using Sourcery CodeBench Lite and its components. It also provides information about Sourcery CodeBench subscriptions. CodeSourcery customers with Sourcery CodeBench subscriptions receive comprehensive support for Sourcery CodeBench.
Appendix A, “Sourcery CodeBench Lite Release Notes”	This appendix contains information about changes in this release of Sourcery CodeBench Lite for ColdFire ELF. You

	should read through these notes to learn about new features and bug fixes.
Appendix B, “Sourcery CodeBench Lite Licenses”	This appendix provides information about the software licenses that apply to Sourcery CodeBench Lite. Read this appendix to understand your legal rights and obligations as a user of Sourcery CodeBench Lite.

3. Typographical Conventions

The following typographical conventions are used in this guide:

`> command arg ...`	A command, typed by the user, and its output. The “>” character is the command prompt.
`command`	The name of a program, when used in a sentence, rather than in literal input or output.
`literal`	Text provided to or received from a computer program.
`placeholder`	Text that should be replaced with an appropriate value when typing a command.
`\`	At the end of a line in command or program examples, indicates that a long line of literal input or output continues onto the next line in the document.

Chapter 1
Quick Start

This chapter includes a brief checklist to follow when installing and using Sourcery CodeBench Lite for the first time. You may use this chapter as an abbreviated guide to the rest of this manual.

Sourcery CodeBench Lite for ColdFire ELF is intended for developers working on embedded applications or firmware for boards without an operating system, or that run an RTOS or boot loader. This Sourcery CodeBench configuration is not intended for Linux or uClinux kernel or application development.

Follow the steps given in this chapter to install Sourcery CodeBench Lite and build and run your first application program. The checklist given here is not a tutorial and does not include detailed instructions for each step; however, it will help guide you to find the instructions and reference information you need to accomplish each step.

You can find additional details about the components, libraries, and other features included in this version of Sourcery CodeBench Lite in Chapter 3, “Sourcery CodeBench Lite for ColdFire ELF”.

1.1. Installation and Set-Up

Install Sourcery CodeBench Lite on your host computer. You may download an installer package from the Sourcery CodeBench web site[1], or you may have received an installer on CD. The installer is an executable program that pops up a window on your computer and leads you through a series of dialogs to configure your installation. When the installation is complete, it offers to launch the Getting Started guide. For more information about installing Sourcery CodeBench Lite, including host system requirements and tips to set up your environment after installation, refer to Chapter 2, “Installation and Configuration”.

Install drivers for your debug device. If you plan to use the Sourcery CodeBench Debug Sprite, you may need to install drivers, libraries, or other software on your host system. Refer to Chapter 6, “Sourcery CodeBench Debug Sprite” for a list of supported devices and information about installing drivers and other device set-up. Sourcery CodeBench Lite also supports third-party debug devices that communicate via the GDB remote serial protocol. If you plan to use one of these devices, follow the manufacturer's directions to connect the device and install any required drivers or software.

1.2. Configuring Sourcery CodeBench Lite for the Target System

Identify your target board. On bare-metal targets, you must explicitly specify a linker script for your target board on your link command line. Supported boards are listed in Chapter 5, “CS3™: The CodeSourcery Common Startup Code Sequence”.

1.3. Building Your Program

Build your program with Sourcery CodeBench command-line tools. Create a simple test program, and follow the directions in Chapter 4, “Using Sourcery CodeBench from the Command Line” to compile and link it using Sourcery CodeBench Lite. On bare-metal targets, you must specify a linker script using the `-T` option on your link command line. Supported boards and linker scripts are listed in Chapter 5, “CS3™: The CodeSourcery Common Startup Code Sequence”.

1.4. Running and Debugging Your Program

The steps to run or debug your program depend on your target system and how it is configured. Choose the appropriate method for your target.

[1] http://www.codesourcery.com/gnu_toolchains/

Debug your program on the target using the Debug Sprite. You can use the Sourcery CodeBench Debug Sprite to load and execute your program on the target from the debugger. Refer to Section 4.3, “Running Applications from GDB” for instructions on using the Sprite from the GDB command line. Detailed reference material for the Sourcery CodeBench Debug Sprite, including information about supported debug devices, can be found in Chapter 6, “Sourcery CodeBench Debug Sprite”.

Debug your program on the target using a third-party debug device. Sourcery CodeBench supports debugging programs on the remote target using third-party debug devices that can communicate via the GDB remote serial protocol. For command-line GDB instructions, see Section 4.3, “Running Applications from GDB”.

Chapter 2
Installation and Configuration

This chapter explains how to install Sourcery CodeBench Lite. You will learn how to:

1. Verify that you can install Sourcery CodeBench Lite on your system.
2. Download the appropriate Sourcery CodeBench Lite installer.
3. Install Sourcery CodeBench Lite.
4. Configure your environment so that you can use Sourcery CodeBench Lite.

2.1. Terminology

Throughout this document, the term *host system* refers to the system on which you run Sourcery CodeBench while the term *target system* refers to the system on which the code produced by Sourcery CodeBench runs. The target system for this version of Sourcery CodeBench is `m68k-elf`.

If you are developing a workstation or server application to run on the same system that you are using to run Sourcery CodeBench, then the host and target systems are the same. On the other hand, if you are developing an application for an embedded system, then the host and target systems are probably different.

2.2. System Requirements

2.2.1. Host Operating System Requirements

This version of Sourcery CodeBench supports the following host operating systems and architectures:

- Microsoft Windows XP (SP1), Windows Vista, and Windows 7 systems using IA32, AMD64, and Intel 64 processors.
- GNU/Linux systems using IA32, AMD64, or Intel 64 processors, including Debian 3.1 (and later), Red Hat Enterprise Linux 3 (and later), SuSE Enterprise Linux 8 (and later), and Ubuntu 8.04 (and later).

Sourcery CodeBench is built as a 32-bit application. Therefore, even when running on a 64-bit host system, Sourcery CodeBench requires 32-bit host libraries. If these libraries are not already installed on your system, you must install them before installing and using Sourcery CodeBench Lite. Consult your operating system documentation for more information about obtaining these libraries.

Installing on Ubuntu and Debian GNU/Linux Hosts

The Sourcery CodeBench graphical installer is incompatible with the `dash` shell, which is the default `/bin/sh` for recent releases of the Ubuntu and Debian GNU/Linux distributions. To install Sourcery CodeBench Lite on these systems, you must make `/bin/sh` a symbolic link to one of the supported shells: `bash`, `csh`, `tcsh`, `zsh`, or `ksh`.

For example, on Ubuntu systems, the recommended way to do this is:

```
> sudo dpkg-reconfigure -plow dash
Install as /bin/sh? No
```

This is a limitation of the installer and uninstaller only, not of the installed Sourcery CodeBench Lite toolchain.

2.2.2. Host Hardware Requirements

In order to install and use Sourcery CodeBench Lite, you must have at least 512MB of available memory.

The amount of disk space required for a complete Sourcery CodeBench Lite installation directory depends on the host operating system and the number of target libraries included. When you start the graphical installer, it checks whether there is sufficient disk space before beginning to install. Note that the graphical installer also requires additional temporary disk space during the installation process. On Microsoft Windows hosts, the installer uses the location specified by the `TEMP` environ-

ment variable for these temporary files. If there is not enough free space on that volume, the installer prompts for an alternate location. On Linux hosts, the installer puts temporary files in the directory specified by the `IATEMPDIR` environment variable, or `/tmp` if that is not set.

2.2.3. Target System Requirements

See Chapter 3, “Sourcery CodeBench Lite for ColdFire ELF” for requirements that apply to the target system.

2.3. Downloading an Installer

If you have received Sourcery CodeBench Lite on a CD, or other physical media, then you do not need to download an installer. You may skip ahead to Section 2.4, “Installing Sourcery CodeBench Lite”.

You can download Sourcery CodeBench Lite from the Sourcery CodeBench web site[1]. This free version of Sourcery CodeBench, which is made available to the general public, does not include all the functionality of CodeSourcery's product releases. If you prefer, you may instead purchase or register for an evaluation of CodeSourcery's product toolchains at the Sourcery CodeBench Portal[2].

Once you have navigated to the appropriate web site, download the installer that corresponds to your host operating system. For Microsoft Windows systems, the Sourcery CodeBench installer is provided as an executable with the `.exe` extension. For GNU/Linux systems Sourcery CodeBench Lite is provided as an executable installer package with the `.bin` extension. You may also install from a compressed archive with the `.tar.bz2` extension.

On Microsoft Windows systems, save the installer to the desktop. On GNU/Linux systems, save the download package in your home directory.

2.4. Installing Sourcery CodeBench Lite

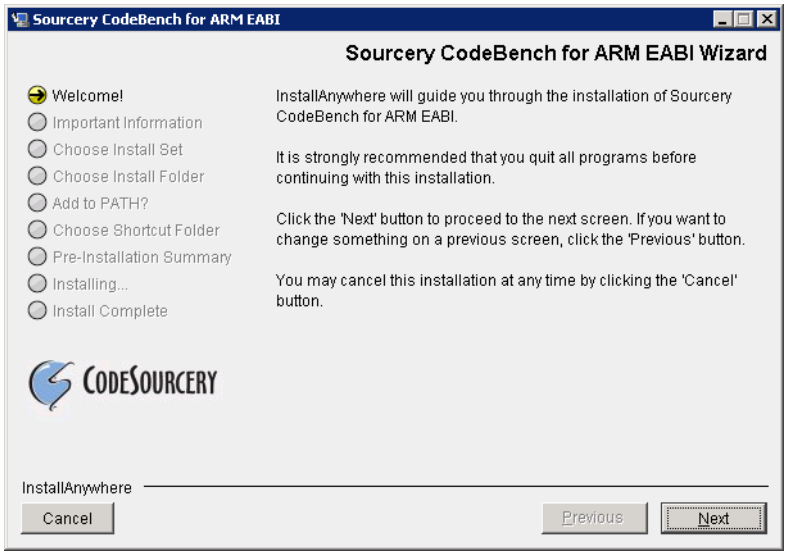
The method used to install Sourcery CodeBench Lite depends on your host system and the kind of installation package you have downloaded.

2.4.1. Using the Sourcery CodeBench Lite Installer on Microsoft Windows

If you have received Sourcery CodeBench Lite on CD, insert the CD in your computer. On most computers, the installer then starts automatically. If your computer has been configured not to automatically run CDs, open `My Computer`, and double click on the CD. If you downloaded Sourcery CodeBench Lite, double-click on the installer.

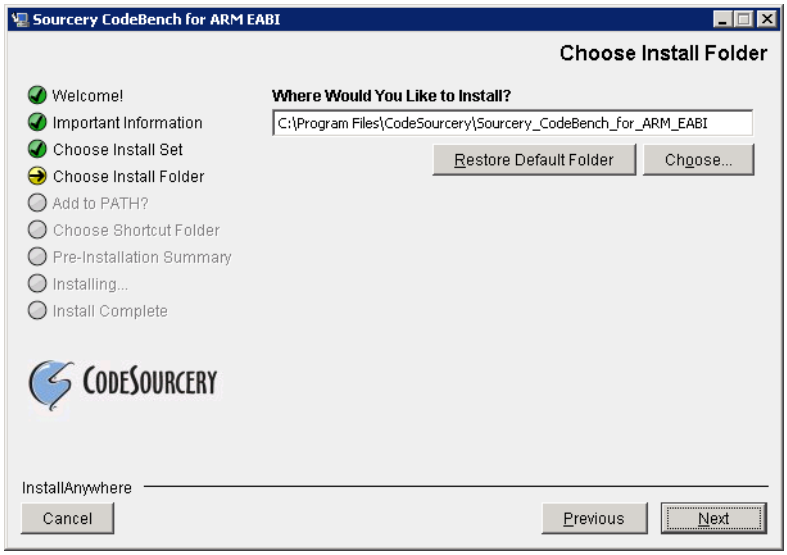
After the installer starts, follow the on-screen dialogs to install Sourcery CodeBench Lite. The installer is intended to be self-explanatory and on most pages the defaults are appropriate.

[1] http://www.codesourcery.com/gnu_toolchains/
[2] https://support.codesourcery.com/GNUToolchain/

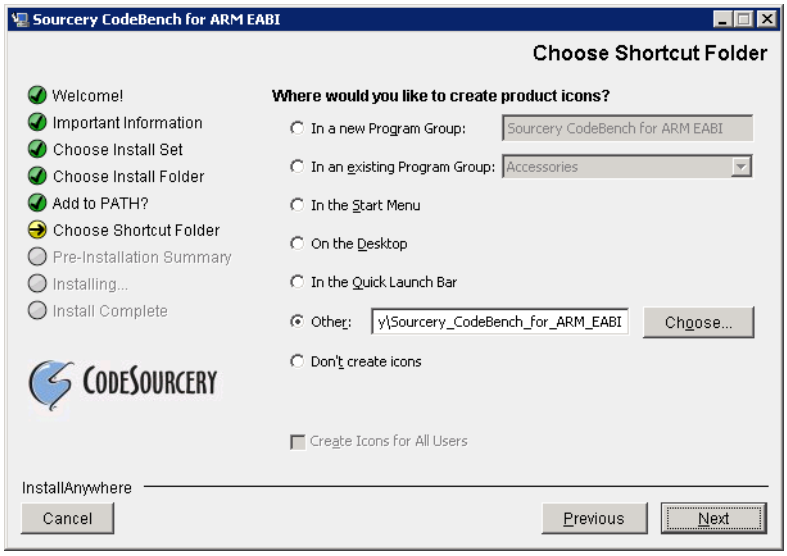


Running the Installer. The graphical installer guides you through the steps to install Sourcery CodeBench Lite.

You may want to change the install directory pathname and customize the shortcut installation.

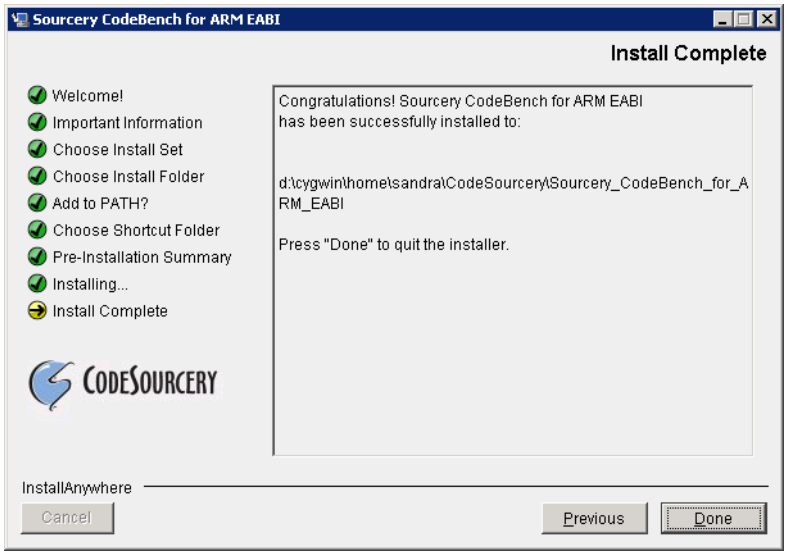


Choose Install Folder. Select the pathname to your install directory.



Choose Shortcut Folder. You can customize where the installer creates shortcuts for quick access to Sourcery CodeBench Lite.

When the installer has finished, it asks if you want to launch a viewer for the Getting Started guide. Finally, the installer displays a summary screen to confirm a successful install before it exits.



Install Complete. You should see a screen similar to this after a successful install.

If you prefer, you can run the installer in console mode rather than using the graphical interface. To do this, invoke the installer with the `-i console` command-line option. For example:

```
> /path/to/package.exe -i console
```

2.4.2. Using the Sourcery CodeBench Lite Installer on GNU/Linux Hosts

Start the graphical installer by invoking the executable shell script:

```
> /bin/sh ./path/to/package.bin
```

After the installer starts, follow the on-screen dialogs to install Sourcery CodeBench Lite. For additional details on running the installer, see the discussion and screen shots in the Microsoft Windows section above.

If you prefer, or if your host system does not run the X Window System, you can run the installer in console mode rather than using the graphical interface. To do this, invoke the installer with the `-i console` command-line option. For example:

```
> /bin/sh ./path/to/package.bin -i console
```

2.4.3. Installing Sourcery CodeBench Lite from a Compressed Archive

You do not need to be a system administrator to install Sourcery CodeBench Lite from a compressed archive. You may install Sourcery CodeBench Lite using any user account and in any directory to which you have write access. This guide assumes that you have decided to install Sourcery CodeBench Lite in the `$HOME/CodeSourcery` subdirectory of your home directory and that the filename of the package you have downloaded is `/path/to/package.tar.bz2`. After installation the toolchain will be in `$HOME/CodeSourcery/sourceryg++-2011.09`.

First, uncompress the package file:

```
> bunzip2 /path/to/package.tar.bz2
```

Next, create the directory in which you wish to install the package:

```
> mkdir -p $HOME/CodeSourcery
```

Change to the installation directory:

```
> cd $HOME/CodeSourcery
```

Unpack the package:

```
> tar xf /path/to/package.tar
```

2.5. Installing Sourcery CodeBench Lite Updates

If you have already installed an earlier version of Sourcery CodeBench Lite for ColdFire ELF on your system, it is not necessary to uninstall it before using the installer to unpack a new version in the same location. The installer detects that it is performing an update in that case.

If you are installing an update from a compressed archive, it is recommended that you remove any previous installation in the same location, or install in a different directory.

Note that the names of the Sourcery CodeBench commands for the ColdFire ELF target all begin with `m68k-elf`. This means that you can install Sourcery CodeBench for multiple target systems in the same directory without conflicts.

2.6. Setting up the Environment

As with the installation process itself, the steps required to set up your environment depend on your host operating system.

2.6.1. Setting up the Environment on Microsoft Windows Hosts

2.6.1.1. Setting the PATH

If you installed Sourcery CodeBench Lite using the graphical installer then you may skip this step. The installer does this setup for you.

In order to use the Sourcery CodeBench tools from the command line, you should add them to your PATH. In the instructions that follow, replace *installdir* with the full pathname of your Sourcery CodeBench Lite installation directory, including the drive letter.

To set the PATH on a Microsoft Windows Vista system, use the following command in a cmd.exe shell:

```
> setx PATH "%PATH%;installdir\bin"
```

To set the PATH on a system running Microsoft Windows 7, from the desktop bring up the Start menu and right click on Computer. Select Properties and click on Advanced system settings. Go to the Advanced tab, then click on the Environment Variables button. Select the PATH variable and click Edit. Add the string ;*installdir*\bin to the end, and click OK.

To set the PATH on older versions of Microsoft Windows, from the desktop bring up the Start menu and right click on My Computer. Select Properties, go to the Advanced tab, then click on the Environment Variables button. Select the PATH variable and click the Edit. Add the string ;*installdir*\bin to the end, and click OK.

You can verify that your PATH is set up correctly by starting a new cmd.exe shell and running:

```
> m68k-elf-g++ -v
```

Verify that the last line of the output contains: Sourcery CodeBench Lite 2011.09-21.

2.6.1.2. Working with Cygwin

Sourcery CodeBench Lite does not require Cygwin or any other UNIX emulation environment. You can use Sourcery CodeBench directly from the Windows command shell. You can also use Sourcery CodeBench from within the Cygwin environment, if you prefer.

The Cygwin emulation environment translates Windows path names into UNIX path names. For example, the Cygwin path /home/user/hello.c corresponds to the Windows path c:\cygwin\home\user\hello.c. Because Sourcery CodeBench is not a Cygwin application, it does not, by default, recognize Cygwin paths.

If you are using Sourcery CodeBench from Cygwin, you should set the CYGPATH environment variable. If this environment variable is set, Sourcery CodeBench Lite automatically translates Cygwin path names into Windows path names. To set this environment variable, type the following command in a Cygwin shell:

```
> export CYGPATH=cygpath
```

To resolve Cygwin path names, Sourcery CodeBench relies on the cygpath utility provided with Cygwin. You must provide Sourcery CodeBench with the full path to cygpath if cygpath is not in your PATH. For example:

```
> export CYGPATH=c:/cygwin/bin/cygpath
```

directs Sourcery CodeBench Lite to use `c:/cygwin/bin/cygpath` as the path conversion utility. The value of `CYGPATH` must be an ordinary Windows path, not a Cygwin path.

2.6.2. Setting up the Environment on GNU/Linux Hosts

If you installed Sourcery CodeBench Lite using the graphical installer then you may skip this step. The installer does this setup for you.

Before using Sourcery CodeBench Lite you should add it to your `PATH`. The command you must use varies with the particular command shell that you are using. If you are using the C Shell (`csh` or `tcsh`), use the command:

```
> setenv PATH installdir/bin:$PATH
```

If you are using Bourne Shell (`sh`), the Korn Shell (`ksh`), or another shell, use:

```
> PATH=installdir/bin:$PATH
> export PATH
```

If you are not sure which shell you are using, try both commands. In both cases, replace *`installdir`* with the full pathname of your Sourcery CodeBench Lite installation directory.

You may also wish to set the `MANPATH` environment variable so that you can access the Sourcery CodeBench manual pages, which provide additional information about using Sourcery CodeBench. To set the `MANPATH` environment variable, follow the same steps shown above, replacing `PATH` with `MANPATH`, and `bin` with `share/doc/sourceryg++-m68k-elf/man`.

You can test that your `PATH` is set up correctly by running the following command:

```
> m68k-elf-g++ -v
```

Verify that the last line of the output contains: `Sourcery CodeBench Lite 2011.09-21`.

2.7. Uninstalling Sourcery CodeBench Lite

The method used to uninstall Sourcery CodeBench Lite depends on the method you originally used to install it. If you have modified any files in the installation it is recommended that you back up these changes. The uninstall procedure may remove the files you have altered. In particular, the `m68k-elf` directory located in the install directory will be removed entirely by the uninstaller.

2.7.1. Using the Sourcery CodeBench Lite Uninstaller on Microsoft Windows

You should use the provided uninstaller to remove a Sourcery CodeBench Lite installation originally created by the graphical installer. Start the graphical uninstaller by invoking the Uninstall executable located in your installation directory, or use the uninstall shortcut created during installation. After the uninstaller starts, follow the on-screen dialogs to uninstall Sourcery CodeBench Lite.

You can run the uninstaller in console mode, rather than using the graphical interface, by invoking the Uninstall executable found in your Sourcery CodeBench Lite installation directory with the `-i console` command-line option.

To uninstall third-party drivers bundled with Sourcery CodeBench Lite, first disconnect the associated hardware device. Then use `Uninstall a program` (Vista and newer) or `Add or Remove`

`Programs` (older versions of Windows) to remove the drivers separately. Depending on the device, you may need to reboot your computer to complete the driver uninstall.

2.7.2. Using the Sourcery CodeBench Lite Uninstaller on GNU/Linux

You should use the provided uninstaller to remove a Sourcery CodeBench Lite installation originally created by the executable installer script. Start the graphical uninstaller by invoking the executable Uninstall shell script located in your installation directory. After the uninstaller starts, follow the on-screen dialogs to uninstall Sourcery CodeBench Lite.

You can run the uninstaller in console mode, rather than using the graphical interface, by invoking the Uninstall script with the `-i console` command-line option.

2.7.3. Uninstalling a Compressed Archive Installation

If you installed Sourcery CodeBench Lite from a `.tar.bz2` file, you can uninstall it by manually deleting the installation directory created in the install procedure.

Chapter 3
Sourcery CodeBench Lite for ColdFire ELF

This chapter contains information about features of Sourcery CodeBench Lite that are specific to ColdFire ELF targets. You should read this chapter to learn how to best use Sourcery CodeBench Lite on your target system.

3.1. Included Components and Features

This section briefly lists the important components and features included in Sourcery CodeBench Lite for ColdFire ELF, and tells you where you may find further information about these features.

Component	Version	Notes
GNU programming tools		
GNU Compiler Collection	4.6.1	Separate manual included.
GNU Binary Utilities	2.21.53	Includes assembler, linker, and other utilities. Separate manuals included.
Debugging support and simulators		
GNU Debugger	7.2.50	Separate manual included.
Sourcery CodeBench Debug Sprite for ColdFire	2011.09-21	See Chapter 6, “Sourcery CodeBench Debug Sprite”.
CCS Server	N/A	Included with the Sourcery CodeBench Debug Sprite. See Section 6.6, “Command Converter Server Devices”.
Target libraries		
CodeSourcery Common Startup Code Sequence	2011.09-21	See Chapter 5, “CS3™: The CodeSourcery Common Startup Code Sequence”.
Newlib C Library	1.18.0	Separate manuals included.
Other utilities		
GNU Make	N/A	Build support on Windows hosts.
GNU Core Utilities	N/A	Build support on Windows hosts.

3.2. Library Configurations

Sourcery CodeBench Lite for ColdFire ELF includes the following library configuration.

MCF5206	
Command-line option(s):	default
Library subdirectory:	`./`
Notes:	This library is built for the base ISA_A ColdFire architecture and can run on all ColdFire processors.

MCF51QE	
Command-line option(s):	`-mcpu=51qe`
Library subdirectory:	`m51qe/`
Notes:	This multilib is also compatible with Freescale MCF51AC, MCF51CN, MCF51JM, and MCF51EM cores.

MCF5206E	
Command-line option(s):	`-mcpu=5206e`
Library subdirectory:	`m5206e/`
Notes:	This multilib is also compatible with Freescale MCF524X, MCF525X, and SCF5250 cores.

MCF5208	
Command-line option(s):	`-mcpu=5208`
Library subdirectory:	`m5208/`
Notes:	This multilib supports Freescale MCF520X, MCF521X, MCF521XX, MCF5221X, MCF5222X, MCF5223X, MCF5225X, MCF5227X, MCF523X, MCF527X, and MCF528X cores.

MCF5307	
Command-line option(s):	`-mcpu=5307`
Library subdirectory:	`m5307/`

MCF532X/7X	
Command-line option(s):	`-mcpu=5329`
Library subdirectory:	`m5329/`
Notes:	This multilib supports Freescale MCF5301x, MCF532X, and MCF537X cores.

MCF5407	
Command-line option(s):	`-mcpu=5407`
Library subdirectory:	`m5407/`

MCF54455	
Command-line option(s):	`-mcpu=54455`
Library subdirectory:	`m54455/`
Notes:	This multilib supports Freescale MCF5441x and MCF5445x cores.

MCF547X/8X	
Command-line option(s):	`-mcpu=5475`
Library subdirectory:	`m5475/`
Notes:	This multilib supports Freescale MCF547X and MCF548X cores.

MCF547X/8X - Soft-Float	
Command-line option(s):	`-mcpu=5475 -msoft-float`
Library subdirectory:	`m5475/softfp/`
Notes:	This multilib supports Freescale MCF547X and MCF548X cores.

Sourcery CodeBench includes copies of run-time libraries that have been built with optimizations for different target architecture variants or other sets of build options. Each such set of libraries is

referred to as a *multilib*. When you link a target application, Sourcery CodeBench selects the multilib matching the build options you have selected.

Sourcery CodeBench Lite's library support includes linker scripts that pull in appropriate CS3 startup code, as well as the libraries themselves. You can find these linker scripts in multilib-specific subdirectories of the `m68k-elf/lib` directory of your Sourcery CodeBench install.

3.3. Using Flash Memory

Sourcery CodeBench Lite supports development and debugging of applications loaded into flash memory on ColdFire ELF targets. There are three steps involved:

1. You must use an appropriate linker script that identifies the ROM memory region on your target board, and locates the program text within that region. Refer to Chapter 5, "CS3™: The CodeSourcery Common Startup Code Sequence" for information about the boards supported by Sourcery CodeBench.

2. Next, load your program into the flash memory on your target board. You must use third-party tools to program the flash memory.

3. Finally, when debugging a program in flash memory, GDB must be told about the ROM region so that it knows where it must use hardware breakpoints to control program execution. If you are using the Sourcery CodeBench Debug Sprite to debug your program, the Sprite does this automatically, using the memory map provided in the board configuration file. Otherwise, you must provide this information explicitly.

   When using GDB from the command line, you can mark the flash memory as read-only by using the command:

   ```
   (gdb) mem start end ro
   ```

   where *`start`* and *`end`* define the address range of the read-only memory region.

In addition to GDB's automatic use of hardware breakpoints in the ROM region, you can also set hardware breakpoints explicitly from the debugger. However, on many targets the number of available hardware breakpoints is very small. Furthermore, GDB also uses hardware breakpoints internally to implement commands such as `step`, `next`, and `finish`. Thus the number of breakpoints you can explicitly set in ROM may be fewer than the number of hardware breakpoints supported by the target system.

For example, ColdFire ISA_A and ISA_B cores support only a single hardware breakpoint. Other ColdFire cores typically support 4 hardware breakpoints.

Chapter 4
Using Sourcery CodeBench from the Command Line

This chapter demonstrates the use of Sourcery CodeBench Lite from the command line.

4.1. Building an Application

This chapter explains how to build an application with Sourcery CodeBench Lite using the command line. As elsewhere in this manual, this section assumes that your target system is m68k-elf, as indicated by the `m68k-elf` command prefix.

Using an editor (such as `notepad` on Microsoft Windows or `vi` on UNIX-like systems), create a file named `main.c` containing the following simple factorial program:

```
#include <stdio.h>

int factorial(int n) {
  if (n == 0)
    return 1;
  return n * factorial (n - 1);
}

int main () {
  int i;
  int n;
  for (i = 0; i < 10; ++i) {
    n = factorial (i);
    printf ("factorial(%d) = %d\n", i, n);
  }
  return 0;
}
```

Compile and link this program using the command:

```
> m68k-elf-gcc -o factorial main.c -T script
```

Sourcery CodeBench requires that you specify a linker script with the `-T` option to build applications for bare-board targets. Linker errors like `undefined reference to `read'` are a symptom of failing to use an appropriate linker script. Default linker scripts are provided in `m68k-elf/lib`. Refer to Chapter 5, “CS3™: The CodeSourcery Common Startup Code Sequence” for information about the boards and linker scripts supported by Sourcery CodeBench Lite. You must also add the processor options for your board, as documented in that chapter, to your compile and link command lines.

There should be no output from the compiler. (If you are building a C++ application, instead of a C application, replace `m68k-elf-gcc` with `m68k-elf-g++`.)

4.2. Running Applications on the Target System

Consult your target board documentation for instructions on loading programs onto the target, and running them. Alternatively, you can use the Sourcery CodeBench Debug Sprite from within GDB to download and run programs on the target via a supported hardware debugging device.

4.3. Running Applications from GDB

You can run GDB, the GNU Debugger, on your host system to debug programs running remotely on a target board or system.

When starting GDB, give it the pathname to the program you want to debug as a command-line argument. For example, if you have built the factorial program as described in Section 4.1, “Building an Application”, enter:

```
> m68k-elf-gdb factorial
```

While this section explains the alternatives for using GDB to run and debug application programs, explaining the use of the GDB command-line interface is beyond the scope of this document. Please refer to the GDB manual for further instructions.

4.3.1. Connecting to the Sourcery CodeBench Debug Sprite

The Sourcery CodeBench Debug Sprite is a program that runs on the host system to support hardware debugging devices. You can use the Debug Sprite to run and debug programs on a target board without an operating system, or to debug an operating system kernel. See Chapter 6, “Sourcery CodeBench Debug Sprite” for detailed information about the supported devices.

You can start the Sprite directly from within GDB:

```
(gdb) target remote | m68k-elf-sprite arguments
```

Refer to Section 6.3, “Invoking Sourcery CodeBench Debug Sprite” for a full description of the Sprite arguments.

4.3.2. Connecting to an External GDB Server

From within GDB, you can connect to a running `gdbserver` or other debugging stub that uses the GDB remote protocol using:

```
(gdb) target remote host:port
```

where *`host`* is the host name or IP address of the machine the stub is running on, and *`port`* is the port number it is listening on for TCP connections.

4.3.3. Loading and Running Applications

Connecting to a bare-metal target or simulator from GDB does not cause your program to be loaded into target memory. You must do this explicitly from GDB after you connect:

```
(gdb) load
```

Alternatively, you can use third-party tools to load your application into flash memory before starting GDB.

To begin execution of your application, you should generally use the `continue` command:

```
(gdb) continue
```

Chapter 5
CS3™: The CodeSourcery Common Startup Code Sequence

CS3 is CodeSourcery's low-level board support library. This chapter documents the boards supported by Sourcery CodeBench Lite and the compiler and linker options you need to use with them. It also explains how you can use and modify CS3-provided definitions for memory maps, system startup code and interrupt vectors in your own code.

Many developers turn to the GNU toolchain for its cross-platform consistency: having a single system support so many different processors and boards helps to limit risk and keep learning curves gentle. Historically, however, the GNU toolchain has lacked a consistent set of conventions for processor- and board-level initialization, language run-time setup, and interrupt and trap handler definition.

The CodeSourcery Common Startup Code Sequence (CS3) addresses this problem. For each supported system, CS3 provides a set of linker scripts describing the system's memory map, and a board support library providing generic reset, startup, and interrupt handlers. These scripts and libraries all follow a standard set of conventions across a range of processors and boards.

In addition to providing linker support, CS3's functionality is fully integrated with the Sourcery CodeBench Debug Sprite. For each supported board, CS3 provides the board file containing the memory map and initialization sequence required for debugging applications on the board via the Sprite, as documented in Section 6.11, “Supported Board Files”.

This chapter is organized in two parts. The first part explains CS3 concepts:

- Section 5.1, “Linker Scripts” provides basic information you need to know in order to select an appropriate CS3-provided linker script for your ColdFire ELF board.
- CS3's program startup and termination model is discussed in Section 5.2, “Program Startup and Termination”.
- Section 5.3, “Memory Layout” discusses the mapping from program sections to memory regions. It also explains how you can refer to memory regions using CS3-provided symbolic names from C, assembly language, or the linker script, and customize placement of code or data in your program.
- Section 5.4, “Interrupt Vectors and Handlers” covers CS3's interrupt handling model, and discusses how you can customize the CS3-provided interrupt vector tables.

The second part provides details about the CS3 implementation for ColdFire ELF:

- Section 5.5, “Supported Boards for ColdFire ELF” lists the boards supported by CS3 for ColdFire ELF, and the available linker scripts for them.
- Section 5.6, “Interrupt Vector Tables” documents the details of the provided interrupt vectors for CS3-supported devices.

5.1. Linker Scripts

When you build programs for ColdFire ELF targets, you must use a linker script. The linker script serves several purposes:

- It determines the memory addresses for placement of code and data sections.
- It defines symbolic names for memory regions present on the board, which you can use programmatically within your code.
- It provides appropriate program startup and termination code, and causes the linker to pull in any low-level board support libraries that are required to run code on the target.
- It optionally provides a *hosting* library for basic I/O functionality.
- It provides a default interrupt vector appropriate for the target processor.

When invoking the Sourcery CodeBench linker from the command line, you must explicitly supply a linker script using the `-T` option; otherwise a link error results.

CS3 may provide multiple linker scripts for different configurations using the same board. For example, on some boards CS3 may support running the program from either RAM or ROM (flash). Some CS3 link configurations are also designed to co-exist with, or be run from, a boot monitor on the target board. Simulator targets typically require different startup code configurations than hardware targets. In CS3 terminology, each of these different configurations is referred to as a *profile*.

The remainder of this section discusses profile and hosting selection considerations in more detail. You can find the full list of supported boards and linker scripts included in this release of Sourcery CodeBench Lite in Section 5.5, “Supported Boards for ColdFire ELF”.

5.1.1. Program and Data Placement

Many boards have both RAM and ROM (flash) memory devices. CS3 provides distinct linker scripts to place the application either entirely in RAM, or to place code and read-only data in ROM.

Some boards have very small amounts of RAM memory. If you use large library functions (such as `printf` and `malloc`), you may overflow the available memory. You may need to use the ROM-based profile for such programs, so that the program itself is stored in ROM. You may be able to reduce the total amount of memory used by your program by replacing portions of the Sourcery CodeBench runtime library and/or startup code.

5.1.2. Hosting and Semihosting

CS3 is designed to support boards without an operating system. To allow functions like `open` and `write` to work without operating system support, a *semihosting* feature is supported, in conjunction with the debugger.

With semihosting enabled, these system calls are translated into equivalent function calls on your host system. You can only use these function calls while connected to the debugger; if you try to use them when disconnected from the debugger, you will get a hardware exception.

Semihosting requires support from the remote GDB debugging stub or agent, as well as the debugger itself. The Sourcery CodeBench Debug Sprite implements semihosting for all supported devices. However, semihosting may not be supported by debugging stubs provided by third parties. If you are using a debug device that communicates with GDB using the GDB remote protocol, check the documentation for your device to see whether semihosting is supported.

A good use of semihosting is to display debugging messages. For example, this program prints a message on the debugger console on the host:

```
#include <unistd.h>

int main () {
  write (STDERR_FILENO, "Hello, world!\n", 14);
  return 0;
}
```

The hosted CS3 linker scripts provide the semihosting support, and as such programs linked with them may only be run with the debugger. For production code, or programs where memory usage is tightly constrained, use the unhosted CS3 linker scripts instead. These scripts provide stub versions of the system calls, which return an appropriate error value in `errno`. If such a stub system call is

required in the executable, the linker also produces a warning. Such a warning may indicate that you have left debugging code active, or that your program contains unused code.

As an alternative to semihosting via the debugger, some targets supported by CS3 can run a boot monitor that provides console I/O services and other basic system calls. CS3 can also provide hosting via these facilities; where a boot monitor is supported, this is noted in the board tables below. Unlike semihosting, hosting via the boot monitor can be used when running programs outside of the debugger.

5.1.3. Specifying a Linker Script

When using Sourcery CodeBench from the command line or from a `Makefile`, you must add `-T` *`script`* to your linking command, where *`script`* is the appropriate linker script. For example, to target DEMOEM boards, you could link with `-T demoem-ram-hosted.ld`.

5.2. Program Startup and Termination

This section documents CS3's model for target initialization prior to invoking the `main` function of your program, and aspects of program termination that are left unspecified in the C and C++ standards. It explains how you can customize or override the default behavior for your application.

CS3 divides the startup sequence into three phases:

- The *hard reset phase* (`__cs3_reset`) includes actions such as initializing the memory controller and setting up the memory map.
- The *assembly initialization phase* (`__cs3_start_asm`) prepares the stack to run C code, and jumps to the C initialization function.
- The *C initialization phase* (`__cs3_start_c`) is responsible for initializing the data areas, running constructors for statically-allocated objects, and calling `main`.

The hard reset and assembly initialization phases are necessarily written in assembly language; at reset, there may not yet be stack to hold compiler temporaries, or perhaps even any RAM accessible to hold the stack. These phases do the minimum necessary to prepare the environment for running simple C code. Then, the code for the final phase may be written in C; CS3 leaves as much as possible to be done at this point.

The CodeSourcery board support library provides default code for all three phases. The hard reset phase is implemented by board- and profile-specific code. The assembly initialization phase is implemented by profile-specific code. The C initialization phase is implemented by generic code.

5.2.1. The Hard Reset Phase

This phase, which begins at `__cs3_reset`, is responsible for initializing board-specific registers, such as memory base registers and DRAM controllers, or scanning memory to check the available size. It is written in assembler and ends with a jump to `__cs3_start_asm`, which is where the assembly initialization phase begins.

The hard reset code is in a section named `.cs3.reset`. CS3 linker scripts define `__cs3_reset` as an alias for a board- and profile-specific entry point. You may override the CS3-provided reset code by defining your own `__cs3_reset` entry point in the `.cs3.reset` section.

Program execution always begins at `__cs3_reset`, whether the program is started from the reset vector, the debugger, or a boot monitor. However, the `__cs3_reset` code linked into the application

is typically non-empty only for ROM-based profiles. For example, in a RAM-based profile, resetting the memory controllers would overwrite the code being executed.

When using the Sourcery CodeBench Debug Sprite, the Sprite is responsible for carrying out the hard reset actions before the program is loaded onto the target. This is performed prior to execution of both RAM- and ROM-profile applications from the debugger. Thus, when debugging a ROM-profile application, hard reset is actually performed twice — once by the Sprite, and once by the application itself.

5.2.2. The Assembly Initialization Phase

This phase is responsible for initializing the stack pointer and creating an initial stack frame. The symbol `__cs3_start_asm` marks the entry point of the assembly initialization code. The assembly initialization phase ends with a call or jump to `__cs3_start_c`.

The assembly initialization phase is profile-specific. For example, while bare-board applications typically must initialize the stack themselves, CS3 also supports boot-monitor profiles where the stack is initialized by the boot monitor before it launches the application. Likewise, some simulators automatically initialize the stack pointer and initial stack frame on startup, while others require a supervisory operation on startup to determine the amount of available memory. Each of these scenarios requires different assembly initialization behavior.

Note that on bare-board targets setting the stack pointer explicitly in the assembly initialization phase is required even if the processor itself initializes the stack pointer automatically on reset. This is to support running programs from the debugger as well as from processor reset.

For backwards compatibility with previous versions of CS3, on RAM and ROM profiles the symbol `__cs3_start_asm` is actually an alias for a symbol named `_start`. However, referencing or defining `_start` directly is now deprecated.

The value of the symbol `__cs3_stack` provides the initial value of the stack pointer for profiles that must set it explicitly. The CodeSourcery linker scripts provide a default value for this symbol, which you may override by defining `__cs3_stack` yourself. See Section 5.3.3, “Heap and Stack Placement” for an example of a custom stack.

The initial stack frame is created for the use of ordinary C and C++ calling conventions. The stack should be initialized so that backtraces stop cleanly at this point; this might entail zeroing a dynamic link pointer, or providing hand-written DWARF call frame information.

The last action of the assembly initialization phase is to call the C function `__cs3_start_c`. This function never returns, and `__cs3_start_asm` need not be prepared to handle a return from it.

As with the hard reset code, the CodeSourcery board support library provides reasonable default assembly initialization code. However, you may provide your own code by providing a definition for `__cs3_start_asm`, either in an object file or a library.

5.2.3. The C Initialization Phase

Finally, C code can be executed. The C startup function is declared as follows:

```
void __cs3_start_c (void) __attribute__ ((noreturn));
```

This function performs the following steps:

- Initialize all `.data`-like sections by copying their contents. For example, ROM-profile linker scripts use this mechanism to initialize writable data in RAM from the read-only data program image.

- Clear all `.bss`-like sections.

- Run constructors for statically-allocated objects, recorded using whatever conventions are usual for C++ on the target architecture.

  CS3 reserves priorities from 0 to 100 for use by initialization code. You can handle tasks like enabling interrupts, initializing coprocessors, pointing control registers at interrupt vectors, and so on by defining constructors with appropriate priorities.

- Call `main` as appropriate.

- Call `exit`, if it is available.

As with the hard reset and assembly initialization code, the CodeSourcery board support library provides a reasonable definition for the `__cs3_start_c` function. You may override this by providing a definition for `__cs3_start_c`, either in an object file or in a library.

5.2.4. Arguments to `main`

The CodeSourcery-provided definition of `__cs3_start_c` can pass command-line arguments to `main` using the normal C `argc` and `argv` mechanism if the board support package provides corresponding definitions for `__cs3_argc` and `__cs3_argv`. For example:

```
int __cs3_argc;
char **__cs3_argv;
```

These variables should be initialized using a constructor function, which is run by `__cs3_start_c` after it initializes the data segment. Use the `constructor` attribute on the function definition:

```
__attribute__((constructor))
static void __cs3_init_args (void) {
   __cs3_argc = ...;
   __cs3_argv = ...;
}
```

The constructor function may have an arbitrary name; `__cs3_init_args` is used only for illustrative purposes here.

If definitions of `__cs3_argc` and `__cs3_argv` are not provided, then the default `__cs3_start_c` function invokes `main` with zero as the `argc` argument and a null pointer as `argv`.

5.2.5. Program Termination

A program running on an embedded system is usually designed never to exit — it runs until the system is powered down. The C and C++ standards leave it unspecified as to whether `exit` is called at program termination. If the program never exits, then there is no reason to include `exit`, facilities to run functions registered with `atexit`, or global destructors. This code would never be run and would therefore just waste space in the application.

The CS3 startup code, by itself, does not cause `exit` to be present in the application. It dynamically checks whether `exit` is present, and only calls it if it is. If you require `exit` to be present, either refer to it within your application, or add `-Wl,-u,exit` to the linking command line.

Similarly, code to register global destructors is only invoked when `atexit` is already in the executable; CS3, by itself, does not cause `atexit` to be present. If you require `atexit`, either refer to it within your application, or add `-Wl,-u,atexit` to the linking command line.

5.3. Memory Layout

Boards supported by CS3 can have multiple banks or regions of memory with different characteristics. This section describes how program sections are mapped onto memory regions, and how you can use these CS3 features to customize placement of your program's code or data in memory. CS3 also provides a uniform set of symbolic names for each region, allowing you to programmatically refer to each region's address range from C or assembly language as well as from the linker script.

5.3.1. Memory Regions and Program Sections

The regions that are available on a particular board are listed in the table for that board in Section 5.5, “Supported Boards for ColdFire ELF”, below. There are two kinds of regions: those documented as "Memory regions", which are general-purpose memory banks that can be used for program or data storage; and those documented as "Other regions", which typically correspond to memory-mapped control registers or other special-purpose storage.

CS3 supports boards that include both `ram` and `rom` memory regions. The `ram` region holds the `.data` and `.bss` sections, and the `.text` section in RAM profiles. In ROM profiles, the `rom` region holds the `.text` section and initialization values for the writable data sections.

In addition, all regions documented as "Memory regions" correspond to similarly-named program sections. For example, the linker script assigns the `.ram` section to the `ram` region.

More generally, for a memory region named *R*, CS3 linker scripts define a section named `.`*R*, which may contain initialized data or code. There is also a section named `.bss.`*R* for zero-initialized data (BSS), which is placed after the initialized data section for this region.

You can explicitly locate data or code in a section corresponding to a particular memory region using section attributes in your source C or C++ code. Section attributes are especially useful on code compiled for boards that include special memory banks, such as a fast on-chip cache memory, in addition to the default `ram` and/or `rom` regions. CS3's start-up code arranges for additional data-like sections to be initialized in the same way as the default `.data` section.

As an example to illustrate the attribute syntax, you can put a variable `v` in the `.ram` section using:

```
int v __attribute__ ((section (".ram")));
```

To declare a function `f` in this section, use:

```
int f (void) __attribute__ ((section (".ram"))) {...}
```

For more information about attribute syntax, see the GCC manual.

In addition to the `.`*R* and `.bss.`*R* sections, CS3 places a `.cs3.region-head.`*R* section at the beginning of each region *R*. Explicitly placing data in `.cs3.region-head.`*R* sections is discouraged, because CS3 itself may want to place items (like interrupt vector tables) at these locations. If there is a conflict, CS3 raises an error at link time.

Regions documented as "Other regions" in the tables in Section 5.5, “Supported Boards for ColdFire ELF” do not have corresponding program sections. Typically, these regions contain memory-mapped control and I/O registers and cannot be used for general data or program storage. If your program

needs to manipulate data in these regions, you can use the CS3 memory map access interface declared in `cs3.h`, as described in Section 5.3.2, “Programmatic Access to the CS3 Memory Map”.

Memory maps for boards supported by Sourcery CodeBench Lite for ColdFire ELF are documented in XML files in the `m68k-elf/lib/boards/` subdirectory of your Sourcery CodeBench installation directory. Note that the memory maps are consistent with those used by Freescale's CodeWarrior tools.

5.3.2. Programmatic Access to the CS3 Memory Map

CS3 makes C declarations describing the memory regions on the target board available to your program via the header file `cs3.h`, which you can find in the `m68k-elf/include` directory within your install.

For each region named *R*, `cs3.h` declares a byte array variable `__cs3_region_start_`*R* at the region's start address, and a `size_t` variable `__cs3_region_size_`*R* to represent the total size of the region. These symbols are defined by the linker script and so may also be referenced from assembly language. Note that all regions are aligned on eight-byte boundaries and sizes are also multiples of eight bytes.

For memory regions that can correspond to program sections (as described in Section 5.3.1, “Memory Regions and Program Sections”), there are additional symbols `__cs3_region_init_`*R* and `__cs3_region_init_size_`*R* that describe constant data used to initialize the region. During the C initialization phase (Section 5.2, “Program Startup and Termination”), this data is copied into the lower part of the memory region. The symbol `__cs3_region_zero_size_`*R* represents the size of the zero-initialized `.bss.`*R* section following the initialized data. Any of these identifiers may actually be defined as a preprocessor macro that expands to an expression of the appropriate type and value.

To perform the memory region initializations during startup, CS3 internally uses the array variable `__cs3_regions`, which contains descriptors for all of the writable (RAM) memory regions. These descriptors are also exposed in `cs3.h`; refer to the header file for details.

5.3.3. Heap and Stack Placement

CS3 linker scripts provide default placement of the heap and stack in the RAM region. However, you can override the defaults by providing your own definitions of the associated CS3 variables. For example, you may put the heap and/or stack in some other memory region.

Heap placement is controlled by defining the symbol `__cs3_heap_start` at the beginning of the heap, and either the symbol `__cs3_heap_end` or the pointer variable `__cs3_heap_limit` to mark the end of the heap. For example, this fragment of C code places the heap in a region named `extsram`:

```
#define HEAPSIZE ...  /* However big you want to make it. */
unsigned char __cs3_heap_start[HEAPSIZE]
    __attribute__ ((section (".bss.extsram"), aligned(8)));
unsigned char *__cs3_heap_limit = __cs3_heap_start + HEAPSIZE;
```

The default initial stack pointer for bare-metal profiles is given by the symbol `__cs3_stack`, and the stack grows downward from this address. Stack initialization is discussed in more detail in Section 5.2.2, “The Assembly Initialization Phase”.

You can find C declarations for the CS3 heap and stack symbols in the header file `cs3.h`.

The `cs3.h` header file also defines a macro for creating a custom stack. The custom stack is created as a block of RAM in the zero-initialized data section (BSS). The specified size must be a compile-time constant. To account for alignment, the final size of the stack may be a few bytes less than the requested size. The symbol `__cs3_stack` is initialized to point to the last extent of the stack block, and is 16-byte aligned. For example, the following fragment of C code creates a stack of 8192 bytes:

```
#include <cs3.h>

CS3_STACK(2 * 4096);
```

As indicated in Section 5.2.2, "The Assembly Initialization Phase", there are cases where a boot monitor or simulator overrides a custom stack.

5.4. Interrupt Vectors and Handlers

CS3 provides standard handlers for interrupts, exceptions and traps, but also allows you to define your own handlers as needed. In this section, we use the term *interrupt* as a generic term for this entire class of events.

Different processors handle interrupts in various ways, but there are two general approaches:

- Some processors fetch an address from an array indexed by the interrupt number, and jump to that address. We call these *address vector* processors.
- Others multiply the interrupt number by some constant factor, add a base address, and jump directly to that address. Here, the interrupt vector consists of blocks of code, so we call these *code vector* processors.
- Still other processors use a more complicated descriptor mechanism for the interrupt table.

ColdFire processors use the address vector model. The remainder of this section assumes that you have some understanding of the specific requirements for your target; refer to the architecture manuals if necessary.

5.4.1. ColdFire ELF Interrupt Vector Implementation

On address vector processors, the CS3 library provides an array of pointers to interrupt handlers named `__cs3_interrupt_vector_`*`form`*, where *`form`* identifies the particular processor variant the vector is appropriate for. Each entry in the vector holds a reference to a symbol named `__cs3_isr_`*`name`*, where *`name`* is the customary name of that interrupt on the processor, or a number if there is no consistently used name. You can find the interrupt vector details in Section 5.6, "Interrupt Vector Tables". The particular vector used by a given CS3-supported board is documented in the tables in Section 5.5, "Supported Boards for ColdFire ELF".

CS3 provides a reasonable default definition for each `__cs3_isr_`*`name`* handler. Many of these symbols are aliased to a common handler routine. If your program stops at a default interrupt handler, its name as shown in backtraces may therefore not correctly reflect which interrupt occurred.

To override an individual handler, provide your own definition for the appropriate `__cs3_isr_`*`name`* symbol. The definition need not be placed in any particular object file section.

To override the entire interrupt vector, you can define `__cs3_interrupt_vector_`*`form`*. You must place this definition in a section named `.cs3.interrupt_vector`. The linker script reports an error if the `.cs3.interrupt_vector` section is empty, to ensure that the definition of `__cs3_interrupt_vector_`*`form`* occupies the proper section.

You may define the vector in C with an array of pointers using the `section` attribute to place it in the appropriate section. For example, to override the interrupt vector on DEMOEM boards, make the following definition:

```
typedef void handler(void);
handler *__attribute__((section (".cs3.interrupt_vector")))
  __cs3_interrupt_vector_coldfire[] =
{ ... };
```

5.4.2. Writing Interrupt Handlers

Interrupt handlers typically require special call/return and register usage conventions that are target-specific and beyond the scope of this document. In many cases, normal C functions cannot be used as interrupt handlers.

As an alternative to writing interrupt handlers in assembly language, on ColdFire targets they may be written in C using the `interrupt` attribute. This tells the compiler to generate appropriate function entry and exit sequences for an interrupt handler. For example, to override the `__cs3_isr_access_error` handler, use the following definition:

```
void __attribute__ ((interrupt)) __cs3_isr_access_error (void)
{
  ... custom handler code ...
}
```

Refer to the GCC manual for more details about attribute syntax and usage.

5.5. Supported Boards for ColdFire ELF

CS3 provides support for the following boards on ColdFire ELF targets.

DEMOEM		
Processor name:	Freescale MCF51EM (Nucleus)	
Processor options:	`-mcpu=51em`	
Memory regions:	`rom` (128KBytes Flash Block 1), `rom2` (128KBytes Flash Block 2), `ram` (16KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`demoem-ram-hosted.ld`
	RAM Unhosted	`demoem-ram.ld`
	ROM Hosted	`demoem-rom-hosted.ld`
	ROM Unhosted	`demoem-rom.ld`

Freescale M5206eC3 Eval		
Processor name:	Freescale MCF5206e	
Processor options:	`-mcpu=5206e`	
Memory regions:	`rom` (1 MByte External Flash (two AM29LV004DT chips)), `ram` (4 MBytes ADRAM), `rambar` (8 KBytes Internal SRAM)	
Other regions:	`mbar` (Internal memory mapped IO)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m5206ec3-ram-hosted.ld`
	RAM Unhosted	`m5206ec3-ram.ld`
	ROM Hosted	`m5206ec3-rom-hosted.ld`
	ROM Unhosted	`m5206ec3-rom.ld`

Freescale M5208EVB		
Processor name:	Freescale MCF5208 (MiniMe)	
Processor options:	`-mcpu=5208`	
Memory regions:	`rom` (2 MBytes External Flash Am19BDD160G, 16bit wide), `ram` (32 MBytes DDR SDRAM), `rambar` (16 KBytes Internal SRAM)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m5208evb-ram-hosted.ld`
	RAM Unhosted	`m5208evb-ram.ld`
	ROM Hosted	`m5208evb-rom-hosted.ld`
	ROM Unhosted	`m5208evb-rom.ld`

Freescale M5208EVB RevE		
Processor name:	Freescale MCF5208 (MiniMe)	
Processor options:	`-mcpu=5208`	
Memory regions:	`rom` (8 MBytes External Flash), `ram` (32 MBytes DDR SDRAM), `rambar` (16 KBytes Internal SRAM)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m5208evbe-ram-hosted.ld`
	RAM Unhosted	`m5208evbe-ram.ld`
	ROM Hosted	`m5208evbe-rom-hosted.ld`
	ROM Unhosted	`m5208evbe-rom.ld`

Freescale M5213EVB (66MHz)		
Processor name:	Freescale MCF5213 (Kirin)	
Processor options:	`-mcpu=5213`	
Memory regions:	`rom` (256 KBytes Internal Flash), `ram` (32 KBytes Internal SRAM)	
Other regions:	`ipsbar` (Internal memory mapped IO)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m5213evb-66-ram-hosted.ld`
	RAM Unhosted	`m5213evb-66-ram.ld`
	ROM Hosted	`m5213evb-66-rom-hosted.ld`
	ROM Unhosted	`m5213evb-66-rom.ld`

Freescale M5213EVB (80MHz)		
Processor name:	Freescale MCF5213 (Kirin)	
Processor options:	`-mcpu=5213`	
Memory regions:	`rom` (256 KBytes Internal Flash), `ram` (32 KBytes Internal SRAM)	
Other regions:	`ipsbar` (Internal memory mapped IO)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m5213evb-80-ram-hosted.ld`
	RAM Unhosted	`m5213evb-80-ram.ld`
	ROM Hosted	`m5213evb-80-rom-hosted.ld`
	ROM Unhosted	`m5213evb-80-rom.ld`

Freescale M52223EVB (66MHz)		
Processor name:	Freescale MCF52223	
Processor options:	`-mcpu=52223`	
Memory regions:	`rom` (256 KBytes Internal Flash), `ram` (32 KBytes Internal SRAM)	
Other regions:	`ipsbar` (Internal memory mapped IO)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m52223evb-66-ram-hosted.ld`
	RAM Unhosted	`m52223evb-66-ram.ld`
	ROM Hosted	`m52223evb-66-rom-hosted.ld`
	ROM Unhosted	`m52223evb-66-rom.ld`

Freescale M52223EVB (80MHz)		
Processor name:	Freescale MCF52223	
Processor options:	`-mcpu=52223`	
Memory regions:	`rom` (256 KBytes Internal Flash), `ram` (32 KBytes Internal SRAM)	
Other regions:	`ipsbar` (Internal memory mapped IO)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m52223evb-80-ram-hosted.ld`
	RAM Unhosted	`m52223evb-80-ram.ld`
	ROM Hosted	`m52223evb-80-rom-hosted.ld`
	ROM Unhosted	`m52223evb-80-rom.ld`

Freescale M52235EVB (20MHz)		
Processor name:	Freescale MCF52235 (Kirin2e)	
Processor options:	`-mcpu=52235`	
Memory regions:	`rom` (256 KBytes Internal Flash), `ram` (32 KBytes Internal SRAM)	
Other regions:	`ipsbar` (Internal memory mapped IO)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m52235evb-ram-hosted.ld`
	RAM Unhosted	`m52235evb-ram.ld`
	ROM Hosted	`m52235evb-rom-hosted.ld`
	ROM Unhosted	`m52235evb-rom.ld`

Freescale M52259EVB (48MHz)		
Processor name:	Freescale MCF5225x (Kirin3)	
Processor options:	`-mcpu=52259`	
Memory regions:	`rom` (512 KBytes Internal Flash), `ram` (64 KBytes Internal SRAM)	
Other regions:	`ipsbar` (Internal memory mapped IO)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m52259evb-ram-hosted.ld`
	RAM Unhosted	`m52259evb-ram.ld`
	ROM Hosted	`m52259evb-rom-hosted.ld`
	ROM Unhosted	`m52259evb-rom.ld`

Freescale M52277EVB		
Processor name:	Freescale MCF5227x	
Processor options:	`-mcpu=52277`	
Memory regions:	`rom` (16 MBytes External Flash), `ram` (64 MBytes DDR SDRAM), `rambar` (128 KBytes Internal SRAM)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m52277evb-ram-hosted.ld`
	RAM Unhosted	`m52277evb-ram.ld`
	ROM Hosted	`m52277evb-rom-hosted.ld`
	ROM Unhosted	`m52277evb-rom.ld`

Freescale M5235EVB		
Processor name:	Freescale MCF5235	
Processor options:	`-mcpu=5235`	
Memory regions:	`ram` (16 MBytes SDRAM), `rom` (2 MBytes External Flash)	
Other regions:	`rambar` (64 KBytes Internal SRAM), `ipsbar` (Internal memory mapped IO)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m5235evb-ram-hosted.ld`
	RAM Unhosted	`m5235evb-ram.ld`
	ROM Hosted	`m5235evb-rom-hosted.ld`
	ROM Unhosted	`m5235evb-rom.ld`

Freescale M5249C3 Eval		
Processor name:	Freescale MCF5249	
Processor options:	`-mcpu=5249`	
Memory regions:	`ram` (4 MBytes SDRAM), `rambar0` (64 KBytes Internal SRAM), `rambar1` (32 KBytes Internal SRAM), `rom` (2 MBytes External Flash)	
Other regions:	`mbar` (Internal memory mapped IO), `mbar2` (Internal memory mapped IO)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m5249c3-ram-hosted.ld`
	RAM Unhosted	`m5249c3-ram.ld`
	ROM Hosted	`m5249c3-rom-hosted.ld`
	ROM Unhosted	`m5249c3-rom.ld`

Freescale M5253EVB		
Processor name:	Freescale MCF5253 (Amadeus)	
Processor options:	`-mcpu=5253`	
Memory regions:	`ram` (16 MBytes SDRAM), `rambar0` (64 KBytes Internal SRAM), `rambar1` (64 KBytes Internal SRAM), `rom` (2 MBytes External Flash)	
Other regions:	`mbar` (Internal memory mapped IO), `mbar2` (Internal memory mapped IO)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m5253evb-ram-hosted.ld`
	RAM Unhosted	`m5253evb-ram.ld`
	ROM Hosted	`m5253evb-rom-hosted.ld`
	ROM Unhosted	`m5253evb-rom.ld`

Freescale M5272C3 Eval		
Processor name:	Freescale MCF5272	
Processor options:	`-mcpu=5272`	
Memory regions:	`ram` (4 MBytes SRAM), `rambar` (4 KBytes Internal SRAM), `rom` (2 MBytes External Flash)	
Other regions:	`mbar` (Internal memory mapped IO)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m5272c3-ram-hosted.ld`
	RAM Unhosted	`m5272c3-ram.ld`
	ROM Hosted	`m5272c3-rom-hosted.ld`
	ROM Unhosted	`m5272c3-rom.ld`

Freescale M5275EVB		
Processor name:	Freescale MCF5275	
Processor options:	`-mcpu=5275`	
Memory regions:	`ram` (16 MBytes SRAM), `rambar` (64 KBytes Internal SRAM), `rom` (2 MBytes External Flash)	
Other regions:	`ipsbar` (Internal memory mapped IO)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m5275evb-ram-hosted.ld`
	RAM Unhosted	`m5275evb-ram.ld`
	ROM Hosted	`m5275evb-rom-hosted.ld`
	ROM Unhosted	`m5275evb-rom.ld`

Freescale M5282EVB		
Processor name:	Freescale MCF5282	
Processor options:	`-mcpu=5282`	
Memory regions:	`ram` (16 MBytes SRAM), `rambar` (64 KBytes Internal SRAM), `rombar` (512 KBytes Internal Flash), `rom` (2 MBytes External Flash)	
Other regions:	`ipsbar` (Internal memory mapped IO)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m5282evb-ram-hosted.ld`
	RAM Unhosted	`m5282evb-ram.ld`
	ROM Hosted	`m5282evb-rom-hosted.ld`
	ROM Unhosted	`m5282evb-rom.ld`

Freescale M53015EVB		
Processor name:	Freescale MCF5301x (Longjing)	
Processor options:	`-mcpu=53017`	
Memory regions:	`rom` (16 MBytes External Flash), `ram` (64 MBytes DDR SDRAM), `rambar` (128 KBytes Internal SRAM), `mram` (512 KBytes External MRAM)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m53015evb-ram-hosted.ld`
	RAM Unhosted	`m53015evb-ram.ld`
	ROM Hosted	`m53015evb-rom-hosted.ld`
	ROM Unhosted	`m53015evb-rom.ld`

Freescale M5307C3 Eval		
Processor name:	Freescale MCF5307	
Processor options:	`-mcpu=5307`	
Memory regions:	`rom` (1 MBytes External Flash), `ram` (16 MBytes SDRAM), `sram` (512 KBytes SRAM), `rambar` (4 KBytes Internal SRAM)	
Other regions:	`mbar` (Internal memory mapped IO)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m5307c3-ram-hosted.ld`
	RAM Unhosted	`m5307c3-ram.ld`
	ROM Hosted	`m5307c3-rom-hosted.ld`
	ROM Unhosted	`m5307c3-rom.ld`

Freescale M5329EVB		
Processor name:	Freescale MCF5329 (DragonFire)	
Processor options:	`-mcpu=5329`	
Memory regions:	`rom` (2 MBytes External Flash), `ram` (32 MBytes DDR SDRAM), `rambar` (32 KBytes Internal SRAM)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m5329evb-ram-hosted.ld`
	RAM Unhosted	`m5329evb-ram.ld`
	ROM Hosted	`m5329evb-rom-hosted.ld`
	ROM Unhosted	`m5329evb-rom.ld`

Freescale M54455EVB		
Processor name:	Freescale MCF54455 (RedStripe)	
Processor options:	`-mcpu=54455`	
Memory regions:	`ram` (256MBytes DDR SDRAM), `rambar1` (32KBytes Internal SRAM), `rom` (16MBytes External Flash), `bootrom` (512KBytes External Flash)	
Other regions:	`mbar` (Internal memory mapped IO)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m54455evb-ram-hosted.ld`
	RAM Unhosted	`m54455evb-ram.ld`
	ROM Hosted	`m54455evb-rom-hosted.ld`
	ROM Unhosted	`m54455evb-rom.ld`

Freescale M54455EVB (Intel flash at CS0)		
Processor name:	Freescale MCF54455 (RedStripe)	
Processor options:	`-mcpu=54455`	
Memory regions:	`ram` (256MBytes DDR SDRAM), `rambar1` (32KBytes Internal SRAM), `rom` (16MBytes External Flash), `bootrom` (512KBytes External Flash)	
Other regions:	`mbar` (Internal memory mapped IO)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m54455evb-intel-ram-hosted.ld`
	RAM Unhosted	`m54455evb-intel-ram.ld`
	ROM Hosted	`m54455evb-intel-rom-hosted.ld`
	ROM Unhosted	`m54455evb-intel-rom.ld`

Freescale M5475EVB		
Processor name:	Freescale MCF5475 (Rigoletto)	
Processor options:	`-mcpu=5475`	
Memory regions:	`ram` (64 MBytes DDR SDRAM), `rambar0` (4 KBytes Internal SRAM), `rambar1` (4 KBytes Internal SRAM), `rom` (16 MBytes External Flash), `bootrom` (2 MBytes External BootFlash)	
Other regions:	`mbar` (Internal memory mapped IO)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m5475evb-ram-hosted.ld`
	RAM Unhosted	`m5475evb-ram.ld`
	ROM Hosted	`m5475evb-rom-hosted.ld`
	ROM Unhosted	`m5475evb-rom.ld`

Freescale M5485EVB		
Processor name:	Freescale MCF5485 (Rigoletto)	
Processor options:	`-mcpu=5485`	
Memory regions:	`ram` (64 MBytes DDR SDRAM), `rambar0` (4 KBytes Internal SRAM), `rambar1` (4 KBytes Internal SRAM), `rom` (16 MBytes External StrataFlash, 16bit wide), `bootrom` (2 MBytes External BootFlash, 16bit wide)	
Other regions:	`mbar` (Internal memory mapped IO)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m5485evb-ram-hosted.ld`
	RAM Unhosted	`m5485evb-ram.ld`
	ROM Hosted	`m5485evb-rom-hosted.ld`
	ROM Unhosted	`m5485evb-rom.ld`

Freescale MCF51AC128A		
Processor name:	Freescale MCF51AC (Celis)	
Processor options:	`-mcpu=51ac`	
Memory regions:	`rom` (128KBytes Flash), `ram` (32KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m51ac128Aevb-ram-hosted.ld`
	RAM Unhosted	`m51ac128Aevb-ram.ld`
	ROM Hosted	`m51ac128Aevb-rom-hosted.ld`
	ROM Unhosted	`m51ac128Aevb-rom.ld`

Freescale MCF51AC128A (33MHz)		
Processor name:	Freescale MCF51AC (Celis)	
Processor options:	`-mcpu=51ac`	
Memory regions:	`rom` (128KBytes Flash), `ram` (32KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m51ac128Aevb-33-ram-hosted.ld`
	RAM Unhosted	`m51ac128Aevb-33-ram.ld`
	ROM Hosted	`m51ac128Aevb-33-rom-hosted.ld`
	ROM Unhosted	`m51ac128Aevb-33-rom.ld`

Freescale MCF51AC128A (50MHz)		
Processor name:	Freescale MCF51AC (Celis)	
Processor options:	`-mcpu=51ac`	
Memory regions:	`rom` (128KBytes Flash), `ram` (32KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m51ac128Aevb-50-ram-hosted.ld`
	RAM Unhosted	`m51ac128Aevb-50-ram.ld`
	ROM Hosted	`m51ac128Aevb-50-rom-hosted.ld`
	ROM Unhosted	`m51ac128Aevb-50-rom.ld`

Freescale MCF51AC128C		
Processor name:	Freescale MCF51AC (Celis)	
Processor options:	`-mcpu=51ac`	
Memory regions:	`rom` (128KBytes Flash), `ram` (16KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m51ac128Cevb-ram-hosted.ld`
	RAM Unhosted	`m51ac128Cevb-ram.ld`
	ROM Hosted	`m51ac128Cevb-rom-hosted.ld`
	ROM Unhosted	`m51ac128Cevb-rom.ld`

Freescale MCF51AC128C (33MHz)		
Processor name:	Freescale MCF51AC (Celis)	
Processor options:	`-mcpu=51ac`	
Memory regions:	`rom` (128KBytes Flash), `ram` (16KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m51ac128Cevb-33-ram-hosted.ld`
	RAM Unhosted	`m51ac128Cevb-33-ram.ld`
	ROM Hosted	`m51ac128Cevb-33-rom-hosted.ld`
	ROM Unhosted	`m51ac128Cevb-33-rom.ld`

Freescale MCF51AC128C (50MHz)		
Processor name:	Freescale MCF51AC (Celis)	
Processor options:	`-mcpu=51ac`	
Memory regions:	`rom` (128KBytes Flash), `ram` (16KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m51ac128Cevb-50-ram-hosted.ld`
	RAM Unhosted	`m51ac128Cevb-50-ram.ld`
	ROM Hosted	`m51ac128Cevb-50-rom-hosted.ld`
	ROM Unhosted	`m51ac128Cevb-50-rom.ld`

Freescale MCF51AC256		
Processor name:	Freescale MCF51AC (Celis)	
Processor options:	`-mcpu=51ac`	
Memory regions:	`rom` (256KBytes Flash), `ram` (32KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m51ac256evb-ram-hosted.ld`
	RAM Unhosted	`m51ac256evb-ram.ld`
	ROM Hosted	`m51ac256evb-rom-hosted.ld`
	ROM Unhosted	`m51ac256evb-rom.ld`

Freescale MCF51AC256 (33MHz)		
Processor name:	Freescale MCF51AC (Celis)	
Processor options:	`-mcpu=51ac`	
Memory regions:	`rom` (256KBytes Flash), `ram` (32KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m51ac256evb-33-ram-hosted.ld`
	RAM Unhosted	`m51ac256evb-33-ram.ld`
	ROM Hosted	`m51ac256evb-33-rom-hosted.ld`
	ROM Unhosted	`m51ac256evb-33-rom.ld`

Freescale MCF51AC256 (50MHz)		
Processor name:	Freescale MCF51AC (Celis)	
Processor options:	`-mcpu=51ac`	
Memory regions:	`rom` (256KBytes Flash), `ram` (32KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m51ac256evb-50-ram-hosted.ld`
	RAM Unhosted	`m51ac256evb-50-ram.ld`
	ROM Hosted	`m51ac256evb-50-rom-hosted.ld`
	ROM Unhosted	`m51ac256evb-50-rom.ld`

Freescale MCF51AG128		
Processor name:	Freescale MCF51AG	
Processor options:	`-mcpu=51ag`	
Memory regions:	`rom` (128KBytes Flash), `ram` (16KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`mcf51ag128-ram-hosted.ld`
	RAM Unhosted	`mcf51ag128-ram.ld`
	ROM Hosted	`mcf51ag128-rom-hosted.ld`
	ROM Unhosted	`mcf51ag128-rom.ld`

Freescale MCF51AG96		
Processor name:	Freescale MCF51AG	
Processor options:	`-mcpu=51ag`	
Memory regions:	`rom` (96KBytes Flash), `ram` (16KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`mcf51ag96-ram-hosted.ld`
	RAM Unhosted	`mcf51ag96-ram.ld`
	ROM Hosted	`mcf51ag96-rom-hosted.ld`
	ROM Unhosted	`mcf51ag96-rom.ld`

Freescale MCF51CN128 (Lasko)		
Processor name:	Freescale MCF51CN (Lasko)	
Processor options:	`-mcpu=51cn`	
Memory regions:	`rom` (128KBytes Flash), `ram` (24KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`mcf51cn128-ram-hosted.ld`
	RAM Unhosted	`mcf51cn128-ram.ld`
	ROM Hosted	`mcf51cn128-rom-hosted.ld`
	ROM Unhosted	`mcf51cn128-rom.ld`

Freescale MCF51CN96 (Lasko)		
Processor name:	Freescale MCF51CN (Lasko)	
Processor options:	`-mcpu=51cn`	
Memory regions:	`rom` (96KBytes Flash), `ram` (24KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`mcf51cn96-ram-hosted.ld`
	RAM Unhosted	`mcf51cn96-ram.ld`
	ROM Hosted	`mcf51cn96-rom-hosted.ld`
	ROM Unhosted	`mcf51cn96-rom.ld`

Freescale MCF51EM128 (Nucleus)		
Processor name:	Freescale MCF51EM (Nucleus)	
Processor options:	`-mcpu=51em`	
Memory regions:	`rom` (64KBytes Flash Block 1), `rom2` (64KBytes Flash Block 2), `ram` (8KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`mcf51em128-ram-hosted.ld`
	RAM Unhosted	`mcf51em128-ram.ld`
	ROM Hosted	`mcf51em128-rom-hosted.ld`
	ROM Unhosted	`mcf51em128-rom.ld`

Freescale MCF51EM256 (Nucleus)		
Processor name:	Freescale MCF51EM (Nucleus)	
Processor options:	`-mcpu=51em`	
Memory regions:	`rom` (128KBytes Flash Block 1), `rom2` (128KBytes Flash Block 2), `ram` (16KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`mcf51em256-ram-hosted.ld`
	RAM Unhosted	`mcf51em256-ram.ld`
	ROM Hosted	`mcf51em256-rom-hosted.ld`
	ROM Unhosted	`mcf51em256-rom.ld`

Freescale MCF51JE128		
Processor name:	Freescale MCF51JE	
Processor options:	`-mcpu=51je`	
Memory regions:	`rom` (64KBytes Flash Block 1), `rom2` (64KBytes Flash Block 2), `bootrom` (8KBytes Boot ROM), `ram` (32KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`mcf51je128-ram-hosted.ld`
	RAM Unhosted	`mcf51je128-ram.ld`
	ROM Hosted	`mcf51je128-rom-hosted.ld`
	ROM Unhosted	`mcf51je128-rom.ld`

Freescale MCF51JE256		
Processor name:	Freescale MCF51JE	
Processor options:	`-mcpu=51je`	
Memory regions:	`rom` (128KBytes Flash Block 1), `rom2` (128KBytes Flash Block 2), `bootrom` (8KBytes Boot ROM), `ram` (32KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`mcf51je256-ram-hosted.ld`
	RAM Unhosted	`mcf51je256-ram.ld`
	ROM Hosted	`mcf51je256-rom-hosted.ld`
	ROM Unhosted	`mcf51je256-rom.ld`

Freescale MCF51JF128		
Processor name:	Freescale MCF51JF	
Processor options:	`-mcpu=51jf`	
Memory regions:	`rom` (128 KBytes Flash), `flexnvm` (32 KBytes FlexNVM), `flexram` (2 KBytes FlexRAM), `ram` (32 KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`mcf51jf128-ram-hosted.ld`
	RAM Unhosted	`mcf51jf128-ram.ld`
	ROM Hosted	`mcf51jf128-rom-hosted.ld`
	ROM Unhosted	`mcf51jf128-rom.ld`

Freescale MCF51JF32		
Processor name:	Freescale MCF51JF	
Processor options:	`-mcpu=51jf`	
Memory regions:	`rom` (32 KBytes Flash), `flexnvm` (16 KBytes FlexNVM), `flexram` (1 KByte FlexRAM), `ram` (8 KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`mcf51jf32-ram-hosted.ld`
	RAM Unhosted	`mcf51jf32-ram.ld`
	ROM Hosted	`mcf51jf32-rom-hosted.ld`
	ROM Unhosted	`mcf51jf32-rom.ld`

Freescale MCF51JF64		
Processor name:	Freescale MCF51JF	
Processor options:	`-mcpu=51jf`	
Memory regions:	`rom` (64 KBytes Flash), `flexnvm` (32 KBytes FlexNVM), `flexram` (2 KBytes FlexRAM), `ram` (16 KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`mcf51jf64-ram-hosted.ld`
	RAM Unhosted	`mcf51jf64-ram.ld`
	ROM Hosted	`mcf51jf64-rom-hosted.ld`
	ROM Unhosted	`mcf51jf64-rom.ld`

Freescale MCF51JG128		
Processor name:	Freescale MCF51JG	
Processor options:	`-mcpu=51jg`	
Memory regions:	`rom` (128 KBytes Flash), `flexnvm` (32 KBytes FlexNVM), `flexram` (2 KBytes FlexRAM), `ram` (32 KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`mcf51jg128-ram-hosted.ld`
	RAM Unhosted	`mcf51jg128-ram.ld`
	ROM Hosted	`mcf51jg128-rom-hosted.ld`
	ROM Unhosted	`mcf51jg128-rom.ld`

Freescale MCF51JG256		
Processor name:	Freescale MCF51JG	
Processor options:	`-mcpu=51jg`	
Memory regions:	`rom` (256 KBytes Flash), `flexnvm` (32 KBytes FlexNVM), `flexram` (2 KBytes FlexRAM), `ram` (64 KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`mcf51jg256-ram-hosted.ld`
	RAM Unhosted	`mcf51jg256-ram.ld`
	ROM Hosted	`mcf51jg256-rom-hosted.ld`
	ROM Unhosted	`mcf51jg256-rom.ld`

Freescale MCF51JG64		
Processor name:	Freescale MCF51JG	
Processor options:	`-mcpu=51jg`	
Memory regions:	`rom` (64 KBytes Flash), `flexnvm` (32 KBytes FlexNVM), `flexram` (2 KBytes FlexRAM), `ram` (16 KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`mcf51jg64-ram-hosted.ld`
	RAM Unhosted	`mcf51jg64-ram.ld`
	ROM Hosted	`mcf51jg64-rom-hosted.ld`
	ROM Unhosted	`mcf51jg64-rom.ld`

Freescale MCF51MM128		
Processor name:	Freescale MCF51MM	
Processor options:	`-mcpu=51mm`	
Memory regions:	`rom` (64KBytes Flash Block 1), `rom2` (64KBytes Flash Block 2), `bootrom` (8KBytes Boot ROM), `ram` (32KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`mcf51mm128-ram-hosted.ld`
	RAM Unhosted	`mcf51mm128-ram.ld`
	ROM Hosted	`mcf51mm128-rom-hosted.ld`
	ROM Unhosted	`mcf51mm128-rom.ld`

Freescale MCF51MM256		
Processor name:	Freescale MCF51MM	
Processor options:	`-mcpu=51mm`	
Memory regions:	`rom` (128KBytes Flash Block 1), `rom2` (128KBytes Flash Block 2), `bootrom` (8KBytes Boot ROM), `ram` (32KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`mcf51mm256-ram-hosted.ld`
	RAM Unhosted	`mcf51mm256-ram.ld`
	ROM Hosted	`mcf51mm256-rom-hosted.ld`
	ROM Unhosted	`mcf51mm256-rom.ld`

Freescale MCF51QE128 (17MHz)		
Processor name:	Freescale MCF51QE	
Processor options:	`-mcpu=51qe`	
Memory regions:	`rom` (128KBytes Flash), `ram` (8KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m51qe128evb-20-ram-hosted.ld`
	RAM Unhosted	`m51qe128evb-20-ram.ld`
	ROM Hosted	`m51qe128evb-20-rom-hosted.ld`
	ROM Unhosted	`m51qe128evb-20-rom.ld`

Freescale MCF51QE128 (33MHz)		
Processor name:	Freescale MCF51QE	
Processor options:	`-mcpu=51qe`	
Memory regions:	`rom` (128KBytes Flash), `ram` (8KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m51qe128evb-40-ram-hosted.ld`
	RAM Unhosted	`m51qe128evb-40-ram.ld`
	ROM Hosted	`m51qe128evb-40-rom-hosted.ld`
	ROM Unhosted	`m51qe128evb-40-rom.ld`

Freescale MCF51QE128 (50MHz)		
Processor name:	Freescale MCF51QE	
Processor options:	`-mcpu=51qe`	
Memory regions:	`rom` (128KBytes Flash), `ram` (8KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m51qe128evb-50-ram-hosted.ld`
	RAM Unhosted	`m51qe128evb-50-ram.ld`
	ROM Hosted	`m51qe128evb-50-rom-hosted.ld`
	ROM Unhosted	`m51qe128evb-50-rom.ld`

Freescale MCF51QE32 (17MHz)		
Processor name:	Freescale MCF51QE	
Processor options:	`-mcpu=51qe`	
Memory regions:	`rom` (32KBytes Flash), `ram` (8KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m51qe32evb-20-ram-hosted.ld`
	RAM Unhosted	`m51qe32evb-20-ram.ld`
	ROM Hosted	`m51qe32evb-20-rom-hosted.ld`
	ROM Unhosted	`m51qe32evb-20-rom.ld`

Freescale MCF51QE32 (33MHz)		
Processor name:	Freescale MCF51QE	
Processor options:	`-mcpu=51qe`	
Memory regions:	`rom` (32KBytes Flash), `ram` (8KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m51qe32evb-40-ram-hosted.ld`
	RAM Unhosted	`m51qe32evb-40-ram.ld`
	ROM Hosted	`m51qe32evb-40-rom-hosted.ld`
	ROM Unhosted	`m51qe32evb-40-rom.ld`

Freescale MCF51QE32 (50MHz)		
Processor name:	Freescale MCF51QE	
Processor options:	`-mcpu=51qe`	
Memory regions:	`rom` (32KBytes Flash), `ram` (8KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m51qe32evb-50-ram-hosted.ld`
	RAM Unhosted	`m51qe32evb-50-ram.ld`
	ROM Hosted	`m51qe32evb-50-rom-hosted.ld`
	ROM Unhosted	`m51qe32evb-50-rom.ld`

Freescale MCF51QE64 (17MHz)		
Processor name:	Freescale MCF51QE	
Processor options:	`-mcpu=51qe`	
Memory regions:	`rom` (64KBytes Flash), `ram` (8KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m51qe64evb-20-ram-hosted.ld`
	RAM Unhosted	`m51qe64evb-20-ram.ld`
	ROM Hosted	`m51qe64evb-20-rom-hosted.ld`
	ROM Unhosted	`m51qe64evb-20-rom.ld`

Freescale MCF51QE64 (33MHz)		
Processor name:	Freescale MCF51QE	
Processor options:	`-mcpu=51qe`	
Memory regions:	`rom` (64KBytes Flash), `ram` (8KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m51qe64evb-40-ram-hosted.ld`
	RAM Unhosted	`m51qe64evb-40-ram.ld`
	ROM Hosted	`m51qe64evb-40-rom-hosted.ld`
	ROM Unhosted	`m51qe64evb-40-rom.ld`

Freescale MCF51QE64 (50MHz)		
Processor name:	Freescale MCF51QE	
Processor options:	`-mcpu=51qe`	
Memory regions:	`rom` (64KBytes Flash), `ram` (8KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`m51qe64evb-50-ram-hosted.ld`
	RAM Unhosted	`m51qe64evb-50-ram.ld`
	ROM Hosted	`m51qe64evb-50-rom-hosted.ld`
	ROM Unhosted	`m51qe64evb-50-rom.ld`

Freescale MCF51QM128		
Processor name:	Freescale MCF51QM	
Processor options:	`-mcpu=51qm`	
Memory regions:	`rom` (128 KBytes Flash), `flexnvm` (32 KBytes FlexNVM), `flexram` (2 KBytes FlexRAM), `ram` (32 KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`mcf51qm128-ram-hosted.ld`
	RAM Unhosted	`mcf51qm128-ram.ld`
	ROM Hosted	`mcf51qm128-rom-hosted.ld`
	ROM Unhosted	`mcf51qm128-rom.ld`

Freescale MCF51QM32		
Processor name:	Freescale MCF51QM	
Processor options:	`-mcpu=51qm`	
Memory regions:	`rom` (32 KBytes Flash), `flexnvm` (16 KBytes FlexNVM), `flexram` (1 KByte FlexRAM), `ram` (8 KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`mcf51qm32-ram-hosted.ld`
	RAM Unhosted	`mcf51qm32-ram.ld`
	ROM Hosted	`mcf51qm32-rom-hosted.ld`
	ROM Unhosted	`mcf51qm32-rom.ld`

Freescale MCF51QM64		
Processor name:	Freescale MCF51QM	
Processor options:	`-mcpu=51qm`	
Memory regions:	`rom` (64 KBytes Flash), `flexnvm` (32 KBytes FlexNVM), `flexram` (2 KBytes FlexRAM), `ram` (16 KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`mcf51qm64-ram-hosted.ld`
	RAM Unhosted	`mcf51qm64-ram.ld`
	ROM Hosted	`mcf51qm64-rom-hosted.ld`
	ROM Unhosted	`mcf51qm64-rom.ld`

Freescale TWR-MCF51CN		
Processor name:	Freescale MCF51CN (Lasko)	
Processor options:	`-mcpu=51cn`	
Memory regions:	`rom` (128KBytes Flash), `ram` (24KBytes Internal RAM)	
Other regions:	`gpio` (IO space), `io` (IO space)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`twr-mcf51cn-ram-hosted.ld`
	RAM Unhosted	`twr-mcf51cn-ram.ld`
	ROM Hosted	`twr-mcf51cn-rom-hosted.ld`
	ROM Unhosted	`twr-mcf51cn-rom.ld`

Freescale TWR-MCF52259 (48MHz)		
Processor name:	Freescale MCF5225x (Kirin3)	
Processor options:	`-mcpu=52259`	
Memory regions:	`rom` (512 KBytes Internal Flash), `ram` (64 KBytes Internal SRAM)	
Other regions:	`ipsbar` (Internal memory mapped IO)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`twr-mcf52259-ram-hosted.ld`
	RAM Unhosted	`twr-mcf52259-ram.ld`
	ROM Hosted	`twr-mcf52259-rom-hosted.ld`
	ROM Unhosted	`twr-mcf52259-rom.ld`

Freescale TWR-MCF5441x		
Processor name:	Freescale MCF54418 (Modelo)	
Processor options:	`-mcpu=54418`	
Memory regions:	`ram` (128MBytes DDR SDRAM), `rambar1` (64KBytes Internal SRAM), `rom` (256MBytes External Flash)	
Other regions:	`flexbus` (FlexBus memory mapped IO), `io1` (Peripheral bus controller 1 memory mapped IO), `io0` (Peripheral bus controller 0 memory mapped IO)	
Interrupt vector:	`__cs3_interrupt_vector_coldfire`	
Linker scripts:	RAM Hosted	`twr-mcf5441x-ram-hosted.ld`
	RAM Unhosted	`twr-mcf5441x-ram.ld`
	ROM Hosted	`twr-mcf5441x-rom-hosted.ld`
	ROM Unhosted	`twr-mcf5441x-rom.ld`

5.6. Interrupt Vector Tables

5.6.1. `__cs3_interrupt_vector_coldfire`

The ColdFire interrupt vector table (`__cs3_interrupt_vector_coldfire`) contents are:

Number	**Name**	**Meaning**
0	`__cs3_stack`	Initial stack pointer
1	`__cs3_reset`	Reset entry point
2	`__cs3_isr_access_error`	Access error
3	`__cs3_isr_address_error`	Address error
4	`__cs3_isr_illegal_instruction`	
5	`__cs3_isr_divide_by_zero`	Divide by zero
6..7	`__cs3_isr_interrupt_6..7`	Unspecified
8	`__cs3_isr_privilege_violation`	Privilege violation
9	`__cs3_isr_trace`	Trace
10	`__cs3_isr_unimplemented_line_a_opcode`	Unimplemented instruction
11	`__cs3_isr_unimplemented_line_f_opcode`	Unimplemented instruction
12	`__cs3_isr_non_pc_breakpoint_debug_interrupt`	Breakpoint
13	`__cs3_isr_pc_breakpoint_debug_interrupt`	Breakpoint
14	`__cs3_isr_format_error`	
15..23	`__cs3_isr_interrupt_15..23`	Unspecified
24	`__cs3_isr_spurious_interrupt`	
25..31	`__cs3_isr_interrupt_25..31`	Unspecified
32..47	`__cs3_isr_trap0..trap15`	User trap
48	`__cs3_isr_fp_branch_unordered`	Floating point error

Number	Name	Meaning
49	`__cs3_isr_fp_inexact_result`	Floating point error
50	`__cs3_isr_fp_divide_by_zero`	Floating point error
51	`__cs3_isr_fp_underflow`	Floating point error
52	`__cs3_isr_fp_operand_error`	Floating point error
53	`__cs3_isr_fp_overflow`	Floating point error
54	`__cs3_isr_fp_input_not_a_number`	Floating point error
55	`__cs3_isr_fp_input_denormalized_number`	Floating point error
56..60	`__cs3_isr_interrupt_56..60`	Unspecified
61	`__cs3_isr_unsupported_instruction`	
62..255	`__cs3_isr_interrupt_62..255`	Unspecified

Chapter 6
Sourcery CodeBench Debug Sprite

This chapter describes the use of the Sourcery CodeBench Debug Sprite for remote debugging. The Sprite allows you to debug programs running on a bare board without an operating system. This chapter includes information about the debugging devices and boards supported by the Sprite for ColdFire ELF.

Sourcery CodeBench Lite contains the Sourcery CodeBench Debug Sprite for ColdFire ELF. This Sprite is provided to allow debugging of programs running on a bare board. You can use the Sprite to debug a program when there is no operating system on the board, or for debugging the operating system itself. If the board is running an operating system, and you wish to debug a program running on that OS, you should use the facilities provided by the OS itself (for instance, using `gdbserver`).

The Sprite acts as an interface between GDB and external debug devices and libraries. Refer to Section 6.3, “Invoking Sourcery CodeBench Debug Sprite” for information about the specific devices supported by this version of Sourcery CodeBench Lite.

Important

The Sourcery CodeBench Debug Sprite is not part of the GNU Debugger and is not free or open-source software. You may use the Sourcery CodeBench Debug Sprite only with the GNU Debugger. You may not distribute the Sourcery CodeBench Debug Sprite to any third party.

6.1. Probing for Debug Devices

Before running the Sourcery CodeBench Debug Sprite for the first time, or when attaching new debug devices to your host system, it is helpful to verify that the Sourcery CodeBench Debug Sprite recognizes your debug hardware. From the command line, invoke the Sprite with the `-i` option:

```
> m68k-elf-sprite -i
```

This prints out a list of supported device types. For devices that can be autodetected, it additionally probes for and prints out a list of attached devices. For instance:

```
Sourcery CodeBench Debug Sprite for ColdFire
    (Sourcery CodeBench Lite 2011.09-21)
pe: [speed=<n:0-31>&memory-timeout=<n:0-99>] P&E Adaptor
  pe://USBMultilink/PE6011970 - USB1 : USB-ML-CF Rev C (PE6011970)
  pe://CycloneProMaxEthernet/10.0.0.85 - 10.0.0.85  :  cyclone1
ccs: [timeout=<n>&speed=<n>] CCS Protocol
  ccs://$Host:$Port/$Chain_position - CCS address
tblcf: TBLCF Interface
  tblcf://:0/ - TBLCF device
osbdm: Open Source BDM
  osbdm://0/ - OSBDM device
```

This shows that P&E, Command Converter Server (CCS), Turbo BDM Light ColdFire (TBLCF), and Open Source BDM (OSBDM) devices are supported. Two P&E devices are detected, one TBLCF device, and one OSBDM device. Although CCS devices are supported, they cannot be autodetected.

Note that it may take several seconds for the Debug Sprite to probe for all types of supported devices.

6.2. Debug Sprite Example

Start by compiling and linking a simple test program for your target board, following the instructions in Chapter 4, “Using Sourcery CodeBench from the Command Line”. Use the `-g` option to tell the compiler to generate debugging information.

For example, to build the `factorial` program to run from RAM on an m5208evb Eval board use:

```
> m68k-elf-gcc -g -mcpu=5208 -Tm5208evb-ram-hosted.ld \
  main.c -o factorial
```

Next start the debugger on your host system:

```
> m68k-elf-gdb factorial
```

The command for connecting GDB to the board depends on the debug device you are using; this is described in more detail in Section 6.3, “Invoking Sourcery CodeBench Debug Sprite”. If you are using a P&E debug device with an m5208evb Eval board, use:

```
(gdb) target remote | m68k-elf-sprite pe: m5208evb
```

The Sprite prints some status messages as it connects to your debug device and target board. If the connection is successful, you should see output similar to:

```
m68k-elf-sprite:Target reset
0x00008936 in ?? ()
(gdb)
```

Next, use GDB to load your program onto the target board.

```
(gdb) load
```

At this point you can use GDB to control the execution of your program as required. For example:

```
(gdb) break main
(gdb) continue
```

6.3. Invoking Sourcery CodeBench Debug Sprite

The Debug Sprite is invoked as follows:

```
> m68k-elf-sprite [options] device-url board-file
```

The *device-url* specifies the debug device to use to communicate with the board. It follows the standard format:

```
scheme:scheme-specific-part[?device-options]
```

Most device URL schemes also follow the regular format:

```
scheme:[//hostname:[port]]/path[?device-options]
```

The meanings of *hostname*, *port*, *path* and *device-options* parts depend on the *scheme* and are described below. The following schemes are supported in Sourcery CodeBench Lite for ColdFire ELF:

`pe` Use a P&E Microcomputer Systems debugging device. Refer to Section 6.5, “P&E Devices”.

`ccs` Use a debugging device controlled by the Command Converter Server (CCS) utility, such as a CodeWarrior Ethernet TAP or USB TAP. Refer to Section 6.6, “Command Converter Server Devices”.

`tblcf` Use a Turbo BDM Light ColdFire (e.g. Axiom AxBDM) debugging device. Refer to Section 6.7, “Turbo BDM Light ColdFire Devices”.

`osbdm` Use an Open Source BDM debugging device. Refer to Section 6.8, “Open Source BDM Devices”.

The optional `?`*`device-options`* portion is allowed in all schemes. These allow additional device-specific options of the form *`name=value`*. Multiple options are concatenated using `&`.

The *`board-file`* specifies an XML file that describes how to initialize the target board, as well as other properties of the board used by the debugger. If *`board-file`* refers to a file (via a relative or absolute pathname), it is read. Otherwise, *`board-file`* can be a board name, and the toolchain's board directory is searched for a matching file. See Section 6.11, “Supported Board Files” for the list of supported boards, or invoke the Sprite with the `-b` option to list the available board files. You can also write a custom board file; see Section 6.12, “Board File Syntax” for more information about the file format.

Both the *`device-url`* and *`board-file`* command-line arguments are required to correctly connect the Sprite to a target board.

6.4. Sourcery CodeBench Debug Sprite Options

The following command-line options are supported by the Sourcery CodeBench Debug Sprite:

`-b` Print a list of *`board-file`* files in the board config directory.

`-h` Print a list of options and their meanings. A list of *`device-url`* syntaxes is also shown.

`-i` Print a list of the accessible devices. If a *`device-url`* is also specified, only devices for that device type are scanned. Each supported device type is listed along with the options that can be appended to the *`device-url`*. For each discovered device, the *`device-url`* is printed along with a description of that device.

`-l` *`[host]:port`* Specify the host address and port number to listen for a GDB connection. If this option is not given, the Debug Sprite communicates with GDB using stdin and stdout. If you start the Sprite from within GDB using the `target remote | m68k-elf-sprite ...` command, you do not need this option.

`-m` Listen for multiple sequential connections. Normally the Debug Sprite terminates after the first connection from GDB terminates. This option instead makes it listen for a subsequent connection. To terminate the Sprite, open a connection and send the string `END\n`.

`-q` Do not print any messages.

`-v` Print additional messages.

If any of `-b`, `-i` or `-h` are given, the Debug Sprite terminates after providing the information rather than waiting for a debugger connection.

6.5. P&E Devices

P&E debug devices are supported. The P&E device partitions the *device-url* as follows:

```
pe:[//type[:number]][/key][?device-options]
```

The various parts are:

type Specify the debug device type. The following debug device types are supported

- USBMultilink
- CycloneProMaxUSB
- CycloneProMaxSerial
- CycloneProMaxEthernet
- ParallelPortCable
- PCIBDMLightning

number Specify the debug device number. Be aware that a device's number depends on whether other devices are concurrently accessed (this is a feature of the underlying P&E library).

key Some P&E devices report unique device keys. This option allows you to select a device by its key, independently of USB device numbering.

Not all the separate parts of the *device-url* are required to uniquely define a particular device. If you specify more than required, the URL must be self-consistent. If you specify fewer components than required, the Sprite uses the first P&E device found that satisfies the specified components.

The *key* is the most robust mechanism for specifying a device, as it uses the unique ID of a particular P&E device. It is immune from renumbering issues, should boards be unplugged or inserted.

The following *device-options* are permitted:

speed=*speed* Specify the speed of the connection. This is a clock divider value, so higher values are slower connection speeds. Refer to the P&E documentation for valid speed settings for your board.

memory-timeout=*timeout* Some boards report memory errors for every access within a certain time of a genuine memory error. This option instructs the Sprite to compensate for this and retry a memory access that reports an error within the specified time of a prior error. If you need to use this option you need to increase GDB's protocol timeout by specifying `set remotetimeout N` at the GDB prompt.

debug=*file* Write P&E debug information to *file*.

6.5.1. Connection Problems

If you get a message “Cannot load P&E library 'UNIT_CFZ.DLL'” or “Cannot load P&E library 'libUnit_cfz.so'”, you probably have not installed the P&E device software. This software is included

with Sourcery CodeBench Lite; see Section 6.5.2, “Installing P&E Drivers” for installation instructions.

The message “Cannot find a matching debug device” means that no P&E device could be found matching the *device-url* that you used. Use the `-i` option to enumerate the devices available.

The message “Cannot force background mode” can occur if you connect at too high a speed. Try slowing the connection by increasing the `speed=` option in the device URL.

6.5.2. Installing P&E Drivers

Drivers provided by P&E Microcomputer Systems are bundled with Sourcery CodeBench Lite for your convenience. You must run the driver installer manually before using your P&E device.

To install or update the drivers on a Windows host, follow these steps:

1. Complete the Sourcery CodeBench Lite installation.
2. If you have previously used a P&E device on your computer, turn off your system and disconnect all P&E devices. Then reboot the system and use `Add/Remove Software`, available through the Windows control panel, to check for and remove any previously-installed P&E drivers.
3. Run the P&E driver installer executable located in the `libexec/m68k-elf-post-install/sprite-drivers/` subdirectory of your Sourcery CodeBench Lite installation.
4. Turn off your system and connect all P&E devices.
5. Reboot the system and start using Sourcery CodeBench Lite.

On Linux, the P&E driver is a loadable kernel module that has to be compiled for your system. You need kernel headers and a native C compiler for your system. The package is provided as a `.tar.gz` file in the `libexec/m68k-elf-post-install/sprite-drivers` subdirectory of your Sourcery CodeBench Lite installation. You should unpack that file, and use the `setup.sh` script to build and install it. You should manually remove all files of a previous install before building this module.

6.6. Command Converter Server Devices

The Sourcery CodeBench Debug Sprite supports devices such as the CodeWarrior Ethernet TAP and USB TAP that are controlled by the Command Converter Server (CCS) utility. You need to start CCS separately before connecting to the debug device from GDB; see Section 6.6.1, “Starting CCS”.

The Sprite partitions the CCS *device-url* as follows:

```
ccs:[//host[:port]][/chainpos][?device-options]
```

The *host* and *port* indicate the location of the CCS port to connect to. The *chainpos* (a number) indicates where the ColdFire debug device is in the CCS chain.

The following *device-options* are permitted:

`speed=`*speed* — Specify the speed used to connect to the target. This is specified in KHz by default. You can use `MHz` and `KHz` suffixes.

`timeout=`*timeout* — This specifies the timeout, in seconds, used for communication with the Command Converter Server.

As an example, if CCS is listening on port `localhost:41475`, connect GDB to the board with:

```
(gdb) target remote | \
  m68k-elf-sprite ccs://localhost:41475 demoem
```

6.6.1. Starting CCS

CCS is included with Sourcery CodeBench Lite; you do not need to have the CodeWarrior tools installed. You can find the CCS executable in the `m68k-elf/ccs/bin` subdirectory of your Sourcery CodeBench Lite installation.

The server can be started by clicking on the CCS icon, or by entering `ccs` on the command line. You can use the `-nogfx` option to use its command-line interface rather than having it create a GUI window.

Use the following commands to initialize the server:

```
% delete all
% config port port
% config cc device
% config client all
```

The *`port`* number is the TCP/IP port the server listens on, and is what you should use in the Sprite's URI. The *`device`* indicates what target device should be used. For USB devices use `utap` for COP/OnCE and `utap_dpi` for BDM or DPI. For Ethernet devices use `powertap` for COP/OnCE and `powertap_dpi` for BDM or DPI. If you have multiple devices of you can append a `:`*`serial-number`* to the USB *`device`* name. The eight-digit *`serial-number`* is located on the underside of the TAP device just after the revision information. For Ethernet devices append the device's IP address.

In summary, to connect to a COP/OnCE target using an Ethernet TAP:

```
% config cc powertap:1.2.3.4
```

To connect to a BDM or DPI target using an Ethernet TAP:

```
% config cc powertap_dpi:1.2.3.4
```

To connect to a COP/OnCE target using a USB TAP:

```
% config cc utap
```

To connect to a BDM or DPI target using a USB TAP:

```
% config cc utap_dpi
```

You can use the `config save` command to save the configuration for later use. The `show cc` command shows you the current configuration. The `show port` command shows you the port number CCS is serving.

6.6.2. Common CCS Errors

Here are some common error messages and their causes:

Cable disconnected
: The target board is not powered up, the board hardware is faulty or in a bad state or the jumper settings are incorrect.

CC not present	The required Command Converter is not present. You did not use `utap_bdm` or `utap_dpi` to connect CCS to a BDM or DPI TAP device connection.
Core not responding	CCS is no longer has control of the target system. The board hardware is faulty or in a bad state, the board initialization settings are incorrect or there is another debugger configuration problem.
USB open failure	For a Windows host, the USB driver on the host computer is hung. Unplug/replug the USB tap, or reboot the host PC if the problem persists. This might also happen if the USB drivers were not installed. You may install USB drivers manually from `m68k-elf\ccs\drivers` subdirectory of your Sourcery CodeBench Lite installation. For a Linux host this can occur if the permissions are not set correctly. Try running CCS as root, and if this resolves the problem, review the instructions in `m68k-elf/ccs/drivers/usb` subdirectory of your Sourcery CodeBench Lite installation for setting up USB permissions.
Maximum number of Command Converters reached	You have tried to reconfigure without first deleting the current configuration.
Cannot reset to debug mode	This can indicate that the clock speed is too high. Try a lower clock speed with the `speed=` option in the device URL.

6.7. Turbo BDM Light ColdFire Devices

Turbo BDM Light ColdFire (TBLCF) devices, such as the Axiom AxBDM device, are supported. The TBLCF device type partitions the *`device-url`* as follows:

```
tblcf:[//:number/]
```

The *`number`* indicates the number of the TBLCF interface to connect to, counting from zero upwards. If the number is omitted, the default is to connect to the zeroth interface, which works well if you have only one TBLCF device connected to your computer.

There are no further options for the TBLCF device.

If you are connecting via TBLCF from Windows, you may see a message like:

```
m68k-elf-sprite:error: Couldn't load libusb DLL
```

If this happens, you must install the driver for the TBLCF device, included with Sourcery CodeBench Lite. See Section 6.7.1, “Installing TBLCF (AxBDM) Windows Drivers” for installation instructions.

If you are connecting via TBLCF from Linux, you may see a message like:

```
m68k-elf-sprite:error: Error claiming interface \
(-1, permission denied)
```

If you see this message, consult Section 6.7.2, “Configuring TBLCF (AxBDM) Devices on Linux” for configuration instructions.

6.7.1. Installing TBLCF (AxBDM) Windows Drivers

Before using a TBLCF device, you must install a driver. To install the TBLCF (AxBDM) driver on Windows, follow these steps:

1. Complete the Sourcery CodeBench Lite installation.
2. Run the `Add Hardware` Control Panel. Click `Yes, I have already connected the hardware`.
3. Select `Add a new hardware device`.
4. Select `Install the hardware that I manually select from a list`.
5. Select `Show all devices`.
6. Click `Have Disk`. Browse to `libexec/m68k-elf-post-install/axbdm-drivers/axbdm.inf`, then select `AxBDM` from the list on the following pane.
7. You will get warnings about the driver not being signed by Microsoft. This is expected.
8. Reboot the system when prompted and start using Sourcery CodeBench Lite.

Windows may auto-detect the TBLCF device when it is connected, and invoke the driver installation dialog automatically. If you have already installed Sourcery CodeBench Lite, you may continue with the dialog using steps similar to those outlined above. Otherwise, close the dialog, install Sourcery CodeBench Lite first, and then follow the above steps to install the driver.

6.7.2. Configuring TBLCF (AxBDM) Devices on Linux

The method you should use for configuring the TBLCF device on Linux depends on whether your machine is using udev or hotplug to manage USB device permissions. To determine which of these your distribution uses, find out your kernel and udev version numbers as follows:

```
> uname -r
2.6.20
> udevinfo -V
udevinfo, version 108
```

A rule of thumb is that if your kernel version is less than 2.6.13 (*`2.6.20`* in the example) or your udev version is less than 059 (*`108`* in the example), your machine uses hotplug to control USB device permissions, else it uses udev. If this rule of thumb doesn't work for you, consult your operating system vendor to determine which method your distribution uses.

Performing the following steps allows any user to access the TBLCF device, rather than just the superuser (root). Running the Debug Sprite as root is technically possible, but is *strongly* discouraged.

6.7.2.1. Configuring TBLCF with udev

To configure udev to handle TBLCF permissions, first locate your udev rule configuration directory (e.g. `/etc/udev/rules.d/`). As root, create a file in that directory called `25-tblcf.rules` with the following contents:

```
BUS=="usb", SYSFS{idVendor}=="0425", SYSFS{idProduct}=="1001", \
MODE="0666"
```

Note that this should be entered on one line, without the backslash. Once this file is created, plug in the TBLCF device (if it is not already plugged in) then reboot your machine to make sure your changes take effect.

6.7.2.2. Configuring TBLCF with hotplug

To configure hotplug to handle TBLCF permissions, you must create two files in your hotplug USB configuration directory (e.g. `/etc/hotplug/usb/`) as root. The first file is named `tblcf` and contains:

```
#!/bin/bash
# /etc/hotplug/usb/tblcf
#
if [ "${ACTION}" = "add" ] && [ -f "${DEVICE}" ]
then
    case "$PRODUCT" in
        425/1001/*)
            chmod 0666 "${DEVICE}"
            ;;
    esac
fi
```

The second file (in the same directory) is named `tblcf.usermap` and contains:

```
tblcf 0x0003 0x0425 0x1001 0x0000 0x0000 0x00 0x00 0x00 0x00 \
0x00 0x00 0x00000000
```

Note that the above must be entered on one line, without the backslash. Create these files and plug in your TBLCF device, if it is not already plugged in. Reboot your machine to make sure your changes take effect.

6.7.2.3. Troubleshooting TBLCF Device Permissions

If you are having difficulties using the Debug Sprite as a non-root user, check that your udev or hotplug configuration is working properly by ensuring that the TBLCF device has the right file permissions. To do this, first run the following command:

```
> lsusb -d 0x0425:0x1001
Bus 004 Device 002: ID 0425:1001 Motorola Semiconductors HK, Ltd
```

Note the bus and device number (*`004`* and *`002`* above). Now, examine the permissions of the corresponding device file as follows:

```
> ls -l /proc/bus/usb/004/002
-rw-rw-rw-  1 root root 50 2007-11-02 12:12 /proc/bus/usb/004/002
```

If the file has permissions as shown, you should be able run the Debug Sprite as any user, and the problem lies elsewhere. If the permissions are different, or there was no output from the `lsusb` command above, your configuration is not working properly. Ask CodeSourcery for further guidance.

6.8. Open Source BDM Devices

Open Source BDM (OSBDM) devices are supported. The OSBDM device type partitions the *`device-url`* as follows:

```
osbdm:[//number/]
```

The *number* indicates the number of the OSBDM interface to connect to, counting from zero upwards. If the number is omitted, the default is to connect to the zeroth interface, which works well if you have only one OSBDM device connected to your computer.

There are no further options for the OSBDM device.

If you are connecting via OSBDM from Windows, you may see a message like:

```
m68k-elf-sprite:error: Cannot load OSBDM library 'OSBDM-JM60.DLL'
```

If this happens, you must install the driver for the OSBDM device. You can obtain the driver from the vendor of your OSBDM device.

As of this writing, there is not yet an OSBDM driver available for Linux hosts.

6.9. Debugging a Remote Board

You can run the Sourcery CodeBench Debug Sprite on a different machine from the one on which GDB is running. For example, if your board is connected to a machine in your lab, you can run the debugger on your laptop and connect to the remote board. The Sourcery CodeBench Debug Sprite must run on the machine that is connected to the target board. You must have Sourcery CodeBench installed on both machines.

To use this mode, you must start the Sprite with the `-l` option and specify the port on which you want it to listen. For example:

```
> m68k-elf-sprite -l :10000 device-url board-file
```

starts the Sprite listening on port 10000.

When running GDB from the command line, use the following command to connect GDB to the remote Sprite:

```
(gdb) target remote host:10000
```

where *host* is the name of the remote machine. After this, debugging is just as if you are debugging a target board connected to your host machine.

For more detailed instructions on using the Sourcery CodeBench Debug Sprite in this way, please refer to the Sourcery CodeBench Knowledge Base[1].

6.10. Implementation Details

The Sourcery CodeBench Debug Sprite uses Background Debug Mode, which is supported by all ColdFire cores. In most cases this is completely non-intrusive to the program being debugged. However, if you are using the Sourcery CodeBench Debug Sprite to debug an operating system kernel (or program with kernel-like features), some of the debugging operations can interact with the program being debugged.

[1] https://support.codesourcery.com/GNUToolchain/kbentry132

6.10.1. Software Breakpoints

The Debug Sprite uses `HALT` instructions to implement software breakpoints and semihosting. On execution of a `HALT` instruction, the Debug Sprite gains control. If the `HALT` instruction is one that the Debug Sprite inserted itself, it reports a breakpoint to the host's GDB. Semihosting breakpoints are detected by checking for the bit pattern `0x4e7bf000`, which corresponds to an unrealistic `movec %sp,0` instruction. The semihosting operation will be performed and the program counter adjusted to skip the ill-formed instruction. For all other `HALT` instructions GDB will report a `SIGTRAP`.

If the program being debugged uses `HALT` instructions in an idle loop, each iteration of the idle loop will cause such a `SIGTRAP` to be reported by GDB. If you want GDB to ignore these signals, enter the following GDB command:

```
handle SIGTRAP nostop noprint nopass
```

As `HALT` is a privileged instruction, the Debug Sprite sets the `UHE` bit in the `CSR` so that user mode programs do not raise a privilege violation exception on `HALT` execution.

6.10.2. Hardware Watchpoints

A single hardware watchpoint is implemented using ColdFire's `TDR`, `AATR`, `ABLR` & `ABHR` debug registers (Trigger Definition Register, Address Attribute Trigger Register, Address Bus Low Register and Address Bus High Register respectively). A range of addresses can watch for data read, write or access.

Because of the way ColdFire implements the address range check, it is possible for an access to an address just before the range, but whose final byte is within the watched range to be undetected. For instance watching a single byte at address `4N+3` fails to trigger on 32 bit writes to address `4N` or on 16 bit writes to address `4N+2`.

6.10.3. Single Stepping

Single stepping uses the ColdFire single step feature. This is performed with the `IPI` (Ignore Pending Interrupts) bit set in the `CSR`. Without this bit set, single stepping an instruction when an interrupt is pending stops at the first instruction of the ISR, which is undesirable. Thus single stepping a sequence of instructions does not process any interrupts. During continuous execution, interrupts are not so inhibited, and ISRs are executed, if the remainder of the processor state allows them. GDB commands that perform single stepping are `step` and `stepi`. Commands that perform continuous execution are `continue`, `jump` and `finish`. The `next` and `nexti` commands perform single stepping, except when a function is called, in which case they perform a sequence of single steps to enter the called function, followed by continuous execution for the bulk of the called function.

6.11. Supported Board Files

The Sourcery CodeBench Debug Sprite for ColdFire ELF includes support for the following target boards. Specify the appropriate *`board-file`* as an argument when invoking the Sprite from the command line.

Board	**Config**
DEMOEM	`demoem`
Freescale M5206eC3 Eval	`m5206ec3`
Freescale M5208EVB	`m5208evb`

Board	**Config**
Freescale M5208EVB RevE	`m5208evbe`
Freescale M5213EVB (66MHz)	`m5213evb-66`
Freescale M5213EVB (80MHz)	`m5213evb-80`
Freescale M52223EVB (66MHz)	`m52223evb-66`
Freescale M52223EVB (80MHz)	`m52223evb-80`
Freescale M52235EVB (20MHz)	`m52235evb`
Freescale M52259EVB (48MHz)	`m52259evb`
Freescale M52277EVB	`m52277evb`
Freescale M5235EVB	`m5235evb`
Freescale M5249C3 Eval	`m5249c3`
Freescale M5253EVB	`m5253evb`
Freescale M5272C3 Eval	`m5272c3`
Freescale M5275EVB	`m5275evb`
Freescale M5282EVB	`m5282evb`
Freescale M53015EVB	`m53015evb`
Freescale M5307C3 Eval	`m5307c3`
Freescale M5329EVB	`m5329evb`
Freescale M54455EVB	`m54455evb`
Freescale M54455EVB (Intel flash at CS0)	`m54455evb-intel`
Freescale M5475EVB	`m5475evb`
Freescale M5485EVB	`m5485evb`
Freescale MCF51AC128A	`m51ac128Aevb`
Freescale MCF51AC128A (33MHz)	`m51ac128Aevb-33`
Freescale MCF51AC128A (50MHz)	`m51ac128Aevb-50`
Freescale MCF51AC128C	`m51ac128Cevb`
Freescale MCF51AC128C (33MHz)	`m51ac128Cevb-33`
Freescale MCF51AC128C (50MHz)	`m51ac128Cevb-50`
Freescale MCF51AC256	`m51ac256evb`
Freescale MCF51AC256 (33MHz)	`m51ac256evb-33`
Freescale MCF51AC256 (50MHz)	`m51ac256evb-50`
Freescale MCF51AG128	`mcf51ag128`
Freescale MCF51AG96	`mcf51ag96`
Freescale MCF51CN128 (Lasko)	`mcf51cn128`
Freescale MCF51CN96 (Lasko)	`mcf51cn96`
Freescale MCF51EM128 (Nucleus)	`mcf51em128`
Freescale MCF51EM256 (Nucleus)	`mcf51em256`
Freescale MCF51JE128	`mcf51je128`
Freescale MCF51JE256	`mcf51je256`

Board	Config
Freescale MCF51JF128	`mcf51jf128`
Freescale MCF51JF32	`mcf51jf32`
Freescale MCF51JF64	`mcf51jf64`
Freescale MCF51JG128	`mcf51jg128`
Freescale MCF51JG256	`mcf51jg256`
Freescale MCF51JG64	`mcf51jg64`
Freescale MCF51MM128	`mcf51mm128`
Freescale MCF51MM256	`mcf51mm256`
Freescale MCF51QE128 (17MHz)	`m51qe128evb-20`
Freescale MCF51QE128 (33MHz)	`m51qe128evb-40`
Freescale MCF51QE128 (50MHz)	`m51qe128evb-50`
Freescale MCF51QE32 (17MHz)	`m51qe32evb-20`
Freescale MCF51QE32 (33MHz)	`m51qe32evb-40`
Freescale MCF51QE32 (50MHz)	`m51qe32evb-50`
Freescale MCF51QE64 (17MHz)	`m51qe64evb-20`
Freescale MCF51QE64 (33MHz)	`m51qe64evb-40`
Freescale MCF51QE64 (50MHz)	`m51qe64evb-50`
Freescale MCF51QM128	`mcf51qm128`
Freescale MCF51QM32	`mcf51qm32`
Freescale MCF51QM64	`mcf51qm64`
Freescale TWR-MCF51CN	`twr-mcf51cn`
Freescale TWR-MCF52259 (48MHz)	`twr-mcf52259`
Freescale TWR-MCF5441x	`twr-mcf5441x`

6.12. Board File Syntax

The *`board-file`* can be a user-written XML file to describe a non-standard board. The Sourcery CodeBench Debug Sprite searches for board files in the `m68k-elf/lib/boards` directory in the installation. Refer to the files in that directory for examples.

The file's DTD is:

```
<!-- Board description files

     Copyright (c) 2007-2009 CodeSourcery, Inc.

     THIS FILE CONTAINS PROPRIETARY, CONFIDENTIAL, AND TRADE
     SECRET INFORMATION OF CODESOURCERY AND/OR ITS LICENSORS.

     You may not use or distribute this file without the express
     written permission of CodeSourcery or its authorized
     distributor.  This file is licensed only for use with
     Sourcery CodeBench.  No other use is permitted.
     -->
```

```
<!ELEMENT board
 (category?, properties?, feature?, initialize?, memory-map?, \
debuggerDefaults?)>

<!-- Board category to group boards list into the tree -->
<!ELEMENT category (#PCDATA)>

<!ELEMENT properties
 (description?, property*)>

<!ELEMENT initialize
 (write-register | write-memory | delay
  | wait-until-memory-equal | wait-until-memory-not-equal)* >
<!ELEMENT write-register EMPTY>
<!ATTLIST write-register
          address CDATA   #REQUIRED
                        value   CDATA   #REQUIRED
                        bits    CDATA   #IMPLIED>
<!ELEMENT write-memory EMPTY>
<!ATTLIST write-memory
          address CDATA   #REQUIRED
                        value   CDATA   #REQUIRED
                        bits    CDATA   #IMPLIED>
<!ELEMENT delay EMPTY>
<!ATTLIST delay
          time CDATA   #REQUIRED>
<!ELEMENT wait-until-memory-equal EMPTY>
<!ATTLIST wait-until-memory-equal
          address CDATA   #REQUIRED
                        value   CDATA   #REQUIRED
                        timeout CDATA   #IMPLIED
                        bits    CDATA   #IMPLIED>
<!ELEMENT wait-until-memory-not-equal EMPTY>
<!ATTLIST wait-until-memory-not-equal
          address CDATA   #REQUIRED
                        value   CDATA   #REQUIRED
                        timeout CDATA   #IMPLIED
                        bits    CDATA   #IMPLIED>

<!ELEMENT memory-map (memory-device)*>
<!ELEMENT memory-device (property*, description?, sectors*)>
<!ATTLIST memory-device
                        address CDATA   #REQUIRED
          size    CDATA   #REQUIRED
          type    CDATA   #REQUIRED
                        device  CDATA   #IMPLIED>

<!ELEMENT description (#PCDATA)>
<!ELEMENT property (#PCDATA)>
<!ATTLIST property name CDATA #REQUIRED>
<!ELEMENT sectors EMPTY>
<!ATTLIST sectors
 size CDATA #REQUIRED
```

```
 count CDATA #REQUIRED>

<!-- Definition of default option values for each debug interface -->
<!ELEMENT debuggerDefaults (debugInterface*)>
<!ELEMENT debugInterface (option*)>
<!ATTLIST debugInterface
 name CDATA #REQUIRED
>
<!ELEMENT option EMPTY>
<!ATTLIST option
 name CDATA #REQUIRED
 defaultValue CDATA #REQUIRED
>

<!ENTITY % gdbtarget SYSTEM "gdb-target.dtd">
%gdbtarget;
```

All values can be provided in decimal, hex (with a `0x` prefix) or octal (with a `0` prefix). Addresses and memory sizes can use a `K`, `KB`, `M`, `MB`, `G` or `GB` suffix to denote a unit of memory. Times must use a `ms` or `us` suffix.

The following elements are available:

`<board>` This top-level element encapsulates the entire description of the board. It can contain `<properties>`, `<feature>`, `<initialize>` and `<memory-map>` elements.

`<properties>` The `<properties>` element specifies specific properties of the target system. This element can occur at most once. It can contain a `<description>` element.

It can also contain `<property>` elements with the following names:

`system-clock` This property specifies the target clock frequency (in Hertz) after reset. It is used to configure flash programming algorithms.

It can also contain `<property>` elements with the following names:

`cache` This boolean property is used to indicate that the target has a cache. This knowledge is necessary to correctly write to a program's instruction stream.

`floating-point` This boolean property indicates whether floating point registers are provided on the target.

`<initialize>` The `<initialize>` element defines an initialization sequence for the board, which the Sprite performs before downloading a program. It can contain `<write-register>`, `<write-memory>` and `<delay>` elements.

`<feature>` This element is used to inform GDB about additional registers and peripherals available on the board. It is passed directly to GDB; see the GDB manual for further details.

`<memory-map>` This element describes the memory map of the target board. It is used by GDB to determine where software breakpoints may be used and when flash programming sequences must be used. This element can occur at most once. It can contain `<memory-device>` elements.

`<memory-device>` This element specifies a region of memory. It has four attributes: `address`, `size`, `type` and `device`. The `address` and `size` attributes specify the location of the memory device. The `type` attribute specifies that device as `ram`, `rom` or `flash`. The `device` attribute is required for `flash` regions; it specifies the flash device type. The `<memory-device>` element can contain a `<description>` element.

It can also contain the following named `<property>` element for additional flash-specific information:

`page-size` This numeric property is used for `cfm` flash devices. It specifies the flash logical page size. When not specified, the page size is set to 1K for the ColdFire V1 devices and to 2K for the ColdFire V2+ devices.

`<write-register>` This element writes a value to a control register. It has three attributes: `address`, `value` and `bits`. The `bits` attribute, specifying the bit width of the write operation, is optional; it defaults to 32. The address may be specified as a number, or as a name. The following registers are available: `ASID`, `ACR0`, `ACR1`, `ACR1`, `ACR1`, `MMUBAR`, `VBR`, `ROMBAR0`, `ROMBAR1`, `FLASHBAR`, `RAMBAR0`, `RAMBAR1`, `MPCR`, `EDRAMBAR`, `SECMBAR`, `MBAR2`, `MBAR`.

`<write-memory>` This element writes a value to a memory location. It has three attributes: `address`, `value` and `bits`. The `bits` attribute is optional and defaults to 32. Bit widths of 8, 16 and 32 bits are supported. The address written to must be naturally aligned for the size of the write being done.

`<delay>` This element introduces a delay. It has one attribute, `time`, which specifies the number of milliseconds, or microseconds to delay by.

`<description>` This element encapsulates a human-readable description of its enclosing element.

`<property>` The `<property>` element allows additional name/value pairs to be specified. The property name is specified in a `name` attribute. The property value is the body of the `<property>` element.

Chapter 7
Next Steps with Sourcery CodeBench

This chapter describes where you can find additional documentation and information about using Sourcery CodeBench Lite and its components.

7.1. Sourcery CodeBench Knowledge Base

The Sourcery CodeBench Knowledge Base is available to registered users at the Sourcery CodeBench Portal[1]. Here you can find solutions to common problems including installing Sourcery CodeBench, making it work with specific targets, and interoperability with third-party libraries. There are also additional example programs and tips for making the most effective use of the toolchain and for solving problems commonly encountered during debugging. The Knowledge Base is updated frequently with additional entries based on inquiries and feedback from customers.

7.2. Example Programs

Sourcery CodeBench Lite includes some bundled example programs. You can find the source code for these examples in the `share/sourceryg++-m68k-elf-examples` directory of your Sourcery CodeBench installation.

7.2.1. Other Examples

The subdirectories contain a number of small, target-independent test programs. You may find these programs useful as self-contained test cases when experimenting with configuring the correct compiler and debugger settings for your target, or when learning how to use the debugger or other features of the Sourcery CodeBench toolchain.

7.3. Manuals for GNU Toolchain Components

Sourcery CodeBench Lite includes the full user manuals for each of the GNU toolchain components, such as the compiler, linker, assembler, and debugger. Most of the manuals include tutorial material for new users as well as serving as a complete reference for command-line options, supported extensions, and the like.

When you install Sourcery CodeBench Lite, links to both the PDF and HTML versions of the manuals are created in the shortcuts folder you select. If you elected not to create shortcuts when installing Sourcery CodeBench Lite, the documentation can be found in the `share/doc/sourceryg++-m68k-elf/` subdirectory of your installation directory.

In addition to the detailed reference manuals, Sourcery CodeBench Lite includes a Unix-style manual page for each toolchain component. You can view these by invoking the `man` command with the pathname of the file you want to view. For example, you can first go to the directory containing the man pages:

```
> cd $INSTALL/share/doc/sourceryg++-m68k-elf/man/man1
```

Then you can invoke `man` as:

```
> man ./m68k-elf-gcc.1
```

Alternatively, if you use `man` regularly, you'll probably find it more convenient to add the directory containing the Sourcery CodeBench man pages to your `MANPATH` environment variable. This should go in your `.profile` or equivalent shell startup file; see Section 2.6, “Setting up the Environment” for instructions. Then you can invoke `man` with just the command name rather than a pathname.

[1] https://support.codesourcery.com/GNUToolchain/

Finally, note that every command-line utility program included with Sourcery CodeBench Lite can be invoked with a `--help` option. This prints a brief description of the arguments and options to the program and exits without doing further processing.

Appendix A
Sourcery CodeBench Lite Release Notes

This appendix contains information about changes in this release of Sourcery CodeBench Lite for ColdFire ELF. You should read through these notes to learn about new features and bug fixes.

A.1. Changes in Sourcery CodeBench Lite for ColdFire ELF

This section documents Sourcery CodeBench Lite changes for each released revision.

A.1.1. Changes in Sourcery CodeBench Lite 2011.09-21

Support for the MCF51JG ColdFire+ processor. Sourcery CodeBench now supports the FreeScale MCF51JG family of ColdFire+ microcontrollers. Use the `-mcpu=51jg` option to generate code for these devices. CS3 board support for MCF51JG64, MCF51JG128 and MCF51JG256 microcontrollers has also been added.

Binutils version 2.21. Sourcery CodeBench Lite for ColdFire ELF is now based on binutils version 2.21.

A crash in GDB `maint print arch` has been fixed . A bug in the GDB command `maint print arch` that sometimes caused GDB to crash has been fixed.

A.1.2. Changes in Sourcery CodeBench Lite 2011.09-17

New Sourcery CodeBench Lite branding. Sourcery G++ has been renamed to Sourcery CodeBench. This change affects the names of the default installation directory and installer-created shortcuts, but no internal pathnames or tool names within the installation directory have been changed.

GCC version 4.6. Sourcery CodeBench Lite for ColdFire ELF is now based on GCC version 4.6. For more information about changes from GCC version 4.5 that was included in previous releases, see `http://gcc.gnu.org/gcc-4.6/changes.html`.

ColdFire comparison fix. A compiler bug that caused floating-point comparisons with zero to be miscompiled in some circumstances has been fixed. This only affected code using the hardware floating-point unit, not soft-float code.

Support for MCF51JF and MCF51QM ColdFire+ processors. Sourcery CodeBench now supports the FreeScale MCF51JF and MCF51QM families of ColdFire+ microcontrollers. Use the `-mcpu=51jf` and `-mcpu=51qm` options to generate code for these devices. CS3 board support for MCF51JF32, MCF51JF64, MCF51JF128, MCF51QM32, MCF51QM64 and MCF51QM128 microcontrollers has also been added.

Map file name demangling bug fix. GCC now properly passes the `--demangle` and `--no-demangle` options to the linker to control map file output. The default behavior on all hosts is now to demangle C++ names.

Assembler crash. The assembler now warns when there is line information for the `*ABS*` section, rather than crash. This can occur when the `.offset` directive is used incorrectly.

Support for TWR-MCF5441x board. CS3 board support for programs running from SDRAM on the TWR-MCF5441x board has been added.

GDB interrupt handling bug fix. A bug in GDB has been fixed that caused it to sometimes fail to interrupt lengthy single-step operations (as by a **Ctrl**+**C** when using GDB from the command line).

Fix GDB crash during connection to debug agent. A bug has been fixed that caused GDB to crash while connecting to any debug agent through standard IO where the debug agent had detected an early error and terminated the communication.

Improved disassembler performance in the debugger. GDB's disassembler has been improved to use more efficient memory access on remote targets.

P&E driver updates. The bundled P&E drivers and libraries for both Windows and Linux hosts have been updated. The update to version 10 drivers fixes compatibility problems with the previously-distributed drivers on Windows 7 hosts. In addition, the P&E support library has been updated to version 3.66-1002. Refer to Section 6.5.2, "Installing P&E Drivers" for P&E driver installation instructions; note that the install procedure on Windows hosts has changed from previous releases.

Debug Sprite option defaults. The Sourcery CodeBench Debug Sprite now uses default option values specified in board configuration files. Options included in the device URL override the default values.

Changes to host operating system requirements. The minimum required Microsoft Windows OS needed to run Sourcery CodeBench Lite is now Windows XP (SP1).

A.1.3. Changes in Sourcery CodeBench Lite 2011.03-96

New Sourcery CodeBench Lite branding. Sourcery G++ has been renamed to Sourcery CodeBench. This change affects the names of the default installation directory and installer-created shortcuts, but no internal pathnames or tool names within the installation directory have been changed.

Interprocedural register optimization. The compiler has a new experimental optimization that generates better code for functions that only call functions in the same object. To switch this optimization on, use `-fuse-caller-save`.

Support for MCF51AG, MCF51JE and MCF51MM ColdFire processors. Sourcery CodeBench now supports the FreeScale MCF51AG, MCF51JE and MCF51MM families of ColdFire microcontrollers. Use the `-mcpu=51ag`, `-mcpu=51je` and `-mcpu=51mm` options to generate code for these devices. CS3 board support for MCF51AG96, MCF51AG128, MCF51JE128, MCF51JE256, MCF51MM128 and MCF51MM256 microcontrollers has also been added.

Improved function call profiling. The compiler now supports the `-finstrument-function-calls` option. Please see the GCC documentation for more details.

Fix GDB crash during connection to debug agent. A bug has been fixed that caused GDB to crash while connecting to any debug agent through standard IO where the debug agent had detected an early error and terminated the communication.

Improved disassembler performance in the debugger. GDB's disassembler has been improved to use more efficient memory access on remote targets.

P&E driver updates. The bundled P&E drivers and libraries for both Windows and Linux hosts have been updated. The update to version 10 drivers fixes compatibility problems with the previously-distributed drivers on Windows 7 hosts. In addition, the P&E support library has been updated to version 3.66-1002. Refer to Section 6.5.2, "Installing P&E Drivers" for P&E driver installation instructions; note that the install procedure on Windows hosts has changed from previous releases.

A.1.4. Changes in Sourcery G++ Lite 2011.03-49

C++ constructor bug fix. A compiler bug has been fixed that caused incorrect code for C++ constructors for some class hierarchies that use virtual inheritance and include empty classes. At runtime, the incorrect constructors resulted in memory corruption or other errors.

A.1.5. Changes in Sourcery G++ Lite 2011.03-15

Compiler optimization improvements. The compiler has been enhanced with a number of optimization improvements, including:

- Smaller and faster code for compound conditionals.
- Removal of superfluous sign and zero extensions.

GCC version 4.5.2. Sourcery G++ Lite for ColdFire ELF is now based on GCC version 4.5.2.

New `-fstrict-volatile-bitfields` option. The compiler has a new option, `-fstrict-volatile-bitfields`, which forces access to a volatile structure member using the width that conforms to its type. Refer to the GCC manual for details.

GCC code generation bug for casts to `volatile` types. A compiler bug has been fixed that sometimes caused incorrect code for references to pointers to types with `volatile` casts.

Incorrect optimization fix. An optimizer bug that in rare cases caused incorrect code to be generated for complex AND and OR expressions containing redundant subexpressions has been fixed.

Incorrect C++ warning fixed. A bug in GCC has been fixed that caused spurious warnings about lambda expressions in C++ code that does not use them.

GCC bug where accesses to volatile structure fields are optimized away. A bug has been fixed where accesses to volatile fields of a structure were sometimes incorrectly optimized away if the structure instance was defined as non-volatile.

A.1.6. Changes in Sourcery G++ Lite 2010.09-39

Linker debug information fix. A bug in linker processing of debug information has been fixed. The bug sometimes prevented the Sourcery G++ debugger from displaying source code if the executable was linked with the `--gc-sections` option.

Debugger warnings quieted. GDB no longer prints `RMT ERROR` diagnostics on connection to the Sourcery G++ Debug Sprite. In spite of the alarming appearance of the messages, they were not actually indicative of a serious problem.

Debug Sprite abnormal termination bug fix. The Sourcery G++ Debug Sprite no longer terminates abnormally if GDB is killed while the target is waiting for semihosted I/O to complete. The bug was only triggered when running GDB on a Windows host.

A.1.7. Changes in Sourcery G++ Lite 2010.09-1

Changes to Sourcery G++ version numbering. Sourcery G++ product and Lite toolchains now uniformly use a version numbering scheme of the form 2011.09-21. The major and minor parts of the version number, in this case 2011.09, identify the release branch, while the final component is a build number within the branch. There are also new preprocessor macros defined by the compiler

for the version number components so that you may conditionalize code for Sourcery G++ or particular Sourcery G++ versions. Details are available in the Sourcery G++ Knowledge Base[1].

GCC fix for reference to undefined label. A bug in the optimizer that caused GCC to emit references to undefined labels has been fixed.

Alignment attributes. A bug has been fixed that caused the compiler to ignore alignment attributes of C++ static member variables where the attribute was present on the definition, but not the declaration.

Compiler optimization improvements. The compiler has been enhanced with a number of optimization improvements, including:

- More efficient assignment for structures containing bitfields.
- Better code for initializing C++ arrays with explicit element initializers.
- Improved logic for eliminating/combining redundant comparisons in code with nested conditionals.
- Better selection of loop variables, resulting in fewer temporaries and more efficient register usage.
- Better code when constant addresses are used as arguments to inline assembly statements.
- Better code for copying small constant strings.

GCC version 4.5.1. Sourcery G++ Lite for ColdFire ELF is now based on GCC version 4.5.1. For more information about changes from GCC version 4.4 that was included in previous releases, see `http://gcc.gnu.org/gcc-4.5/changes.html`.

Smaller C++ programs with `-g`. An assembler bug has been fixed that caused unnecessary references to exception-handling routines from C++ programs when debug information is enabled. For programs that do not otherwise use exceptions, this change results in smaller code size.

Additional validation in the assembler. The assembler now diagnoses an error, instead of producing an invalid object file, when directives such as `.hidden` are missing operands.

Strip bug fix. A bug in the `strip` and `objcopy` utilities, which resulted in stripped object files that the linker could not recognize, has been fixed.

Binutils update. The binutils package has been updated to version 2.20.51.20100809 from the FSF trunk. This update includes numerous bug fixes.

Object file flags fix. The `objdump` and `readelf` utilities have been updated to decode ISA-C and EMAC-B ELF header flags.

Additional alignment in CS3-defined linker scripts. Sourcery G++ now ensures 8-byte alignment at additional points in CS3-defined linker scripts. Previously, placing a symbol in certain sections broke the initialization of the `.data` and/or `.bss` sections.

Support for TWR-MCF52259 board. Sourcery G++ now includes support for Freescale TWR-MCF52259 boards.

Newlib update. The Newlib package has been updated to version 1.18.0, with additions from the community CVS trunk as of 2010-08-12. This update provides additional wide-character functions, along with other bug fixes and enhancements.

[1] https://support.codesourcery.com/GNUToolchain/kbentry1

`malloc` fix. A bug that sometimes caused `free` to dereference an invalid address has been fixed. The bug was caused by incorrect handling within `malloc` of calls to `sbrk` from outside of `malloc`.

GDB update. The included version of GDB has been updated to 7.2.50.20100908. This update adds numerous bug fixes and new features, including improved C++ language support, a new command to save breakpoints to a file, a new convenience variable `$_thread` that holds the number of the current thread, among many other improvements.

GDB crash fix. A bug has been fixed that caused GDB to crash on launch if the environment variable `CYGPATH` is set to a program that does not exist or cannot be executed.

ColdFire V2-V4 OSBDM support. The Sourcery G++ Debug Sprite has been fixed to work around OSBDM driver issues for ColdFire V2-V4 boards. The issues fixed include failure to perform semihosting operations.

A.1.8. Changes in Older Releases

For information about changes in older releases of Sourcery G++ Lite for ColdFire ELF, please refer to the Getting Started guide packaged with those releases.

Appendix B
Sourcery CodeBench Lite Licenses

Sourcery CodeBench Lite contains software provided under a variety of licenses. Some components are “free” or “open source” software, while other components are proprietary. This appendix explains what licenses apply to your use of Sourcery CodeBench Lite. You should read this appendix to understand your legal rights and obligations as a user of Sourcery CodeBench Lite.

B.1. Licenses for Sourcery CodeBench Lite Components

The table below lists the major components of Sourcery CodeBench Lite for ColdFire ELF and the license terms which apply to each of these components.

Some free or open-source components provide documentation or other files under terms different from those shown below. For definitive information about the license that applies to each component, consult the source package corresponding to this release of Sourcery CodeBench Lite. Sourcery CodeBench Lite may contain free or open-source components not included in the list below; for a definitive list, consult the source package corresponding to this release of Sourcery CodeBench Lite.

Component	**License**
GNU Compiler Collection	GNU General Public License 3.0 http://www.gnu.org/licenses/gpl.html
GNU Binary Utilities	GNU General Public License 3.0 http://www.gnu.org/licenses/gpl.html
GNU Debugger	GNU General Public License 3.0 http://www.gnu.org/licenses/gpl.html
Sourcery CodeBench Debug Sprite for ColdFire	CodeSourcery License
CCS Server	CCS Server License
CodeSourcery Common Startup Code Sequence	CodeSourcery License
Newlib C Library	BSD License. For the text of the license and a complete list of copyright holders, see Section B.3.2, “Newlib”.
GNU Make	GNU General Public License 2.0 http://www.gnu.org/licenses/old-licenses/gpl-2.0.html
GNU Core Utilities	GNU General Public License 2.0 http://www.gnu.org/licenses/old-licenses/gpl-2.0.html

The CodeSourcery License is available in Section B.2, “Sourcery CodeBench Software License Agreement”.

Important

Although some of the licenses that apply to Sourcery CodeBench Lite are “free software” or “open source software” licenses, none of these licenses impose any obligation on you to reveal the source code of applications you build with Sourcery CodeBench Lite. You can develop proprietary applications and libraries with Sourcery CodeBench Lite.

Sourcery CodeBench Lite may include some third party example programs and libraries in the `share/sourceryg++-m68k-elf-examples` subdirectory. These examples are not covered by the Sourcery CodeBench Software License Agreement. To the extent permitted by law, these examples are provided by CodeSourcery as is with no warranty of any kind, including implied warranties of merchantability or fitness for a particular purpose. Your use of each example is governed by the license notice (if any) it contains.

B.2. Sourcery CodeBench™ Software License Agreement

1. **Parties.** The parties to this Agreement are you, the licensee ("You" or "Licensee") and Mentor Graphics. If You are not acting on behalf of Yourself as an individual, then "You" means Your company or organization.

2. **The Software.** The Software licensed under this Agreement consists of computer programs and documentation referred to as Sourcery CodeBench™ Lite Edition (the "Software").

3. **Definitions.**

    3.1. **Mentor Graphics Proprietary Components.** The components of the Software that are owned and/or licensed by Mentor Graphics and are not subject to a "free software" or "open source" license, such as the GNU Public License. The Mentor Graphics Proprietary Components of the Software include, without limitation, the Sourcery CodeBench Installer, any Sourcery CodeBench Eclipse plug-ins, the CodeSourcery C Library (CSLIBC), and any Sourcery CodeBench Debug Sprite. For a complete list, refer to the *Getting Started Guide* included with the distribution.

    3.2. **Open Source Software Components.** The components of the Software that are subject to a "free software" or "open source" license, such as the GNU Public License.

    3.3. **Proprietary Rights.** All rights in and to copyrights, rights to register copyrights, trade secrets, inventions, patents, patent rights, trademarks, trademark rights, confidential and proprietary information protected under contract or otherwise under law, and other similar rights or interests in intellectual or industrial property.

    3.4. **Redistributable Components.** The Mentor Graphics Proprietary Components that are intended to be incorporated or linked into Licensee object code developed with the Software. The Redistributable Components of the Software include, without limitation, CSLIBC and the CodeSourcery Common Startup Code Sequence (CS3). For a complete list, refer to the *Getting Started Guide* included with the distribution.

4. **License Grant to Proprietary Components of the Software.** You are granted a non-exclusive, royalty-free license (a) to install and use the Mentor Graphics Proprietary Components of the Software, (b) to transmit the Mentor Graphics Proprietary Components over an internal computer network, (c) to copy the Mentor Graphics Proprietary Components for Your internal use only, and (d) to distribute the Redistributable Component(s) in binary form only and only as part of Licensee object code developed with the Software that provides substantially different functionality than the Redistributable Component(s).

5. **Restrictions.** You may not: (i) copy or permit others to use the Mentor Graphics Proprietary Components of the Software, except as expressly provided above; (ii) distribute the Mentor Graphics Proprietary Components of the Software to any third party, except as expressly provided above; or (iii) reverse engineer, decompile, or disassemble the Mentor Graphics Proprietary Components of the Software, except to the extent this restriction is expressly prohibited by applicable law.

    5.1. **Sourcery CodeBench Debug Sprite for P&E Devices.** You may use the Sourcery CodeBench Debug Sprite for P&E only in conjunction with ColdFire microprocessors and with debugging devices produced by P&E Microcomputer Systems.

5.2. **Sourcery CodeBench Debug Sprite for CCS Debugging Devices.** The Sourcery CodeBench Debug Sprite for CCS includes the CodeWarrior Connection Server Dynamic Linked Library ("CCS DLL") from Freescale Semiconductor, Inc. You may use the CCS DLL only in conjunction with Sourcery CodeBench on a Windows or Linux-hosted platform. You may not translate, reverse engineer, decompile, or disassemble the CCS DLL, except to the extent applicable law specifically prohibits such restriction. If You are a U.S. Government end user, the CCS DLL is "restricted computer software" and is subject to FAR 52.227-19(c)(1) and (c)(2).

6. **"Free Software" or "Open Source" License to Certain Components of the Software.** This Agreement does not limit Your rights under, or grant You rights that supersede, the license terms of any Open Source Software Component delivered to You by Mentor Graphics. Sourcery CodeBench includes components provided under various different licenses. The *Getting Started Guide* provides an overview of which license applies to different components, and, for components subject to the Eclipse Public License, contains information on how to obtain the source code. Definitive licensing information for each "free software" or "open source" component is available in the relevant source file.

7. **Mentor Graphics Trademarks.** Notwithstanding any provision in a "free software" or "open source" license agreement applicable to a component of the Software that permits You to distribute such component to a third party in source or binary form, You may not use any Mentor Graphics trademark, whether registered or unregistered, including without limitation, CodeSourcery™, Sourcery CodeBench™, the CodeSourcery crystal ball logo, or the Sourcery CodeBench splash screen, or any confusingly similar mark, in connection with such distribution, and You may not recompile the Open Source Software Components with the `--with-pkgversion` or `--with-bugurl` configuration options that embed Mentor Graphics trademarks in the resulting binary.

8. **Term and Termination.** This Agreement shall remain in effect unless terminated pursuant to this provision. Mentor Graphics may terminate this Agreement upon seven (7) days written notice of a material breach of this Agreement if such breach is not cured; provided that the unauthorized use, copying, or distribution of the Mentor Graphics Proprietary Components of the Software will be deemed a material breach that cannot be cured.

9. **Transfers.** You may not transfer any rights under this Agreement without the prior written consent of Mentor Graphics, which consent shall not be unreasonably withheld. A condition to any transfer or assignment shall be that the recipient agrees to the terms of this Agreement. Any attempted transfer or assignment in violation of this provision shall be null and void.

10. **Ownership.** Mentor Graphics owns and/or has licensed the Mentor Graphics Proprietary Components of the Software and all intellectual property rights embodied therein, including copyrights and valuable trade secrets embodied in its design and coding methodology. The Mentor Graphics Proprietary Components of the Software are protected by United States copyright laws and international treaty provisions. Mentor Graphics also owns all rights, title and interest in and with respect to its trade names, domain names, trade dress, logos, trademarks, service marks, and other similar rights or interests in intellectual property. This Agreement provides You only a limited use license, and no ownership of any intellectual property.

11. **Warranty Disclaimer; Limitation of Liability.** MENTOR GRAPHICS AND ITS LICENSORS PROVIDE THE SOFTWARE "AS-IS" AND PROVIDED WITH ALL FAULTS. MENTOR GRAPHICS DOES NOT MAKE ANY WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. MENTOR GRAPHICS SPECIFICALLY DISCLAIMS THE IMPLIED WARRANTIES OF TITLE, NON-INFRINGEMENT, MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, SYSTEM INTEGRATION, AND DATA ACCURACY. THERE IS NO WARRANTY OR GUARANTEE THAT THE OPERATION OF THE SOFTWARE

WILL BE UNINTERRUPTED, ERROR-FREE, OR VIRUS-FREE, OR THAT THE SOFTWARE WILL MEET ANY PARTICULAR CRITERIA OF PERFORMANCE, QUALITY, ACCURACY, PURPOSE, OR NEED. YOU ASSUME THE ENTIRE RISK OF SELECTION, INSTALLATION, AND USE OF THE SOFTWARE. THIS DISCLAIMER OF WARRANTY CONSTITUTES AN ESSENTIAL PART OF THIS AGREEMENT. NO USE OF THE SOFTWARE IS AUTHORIZED HEREUNDER EXCEPT UNDER THIS DISCLAIMER.

12. **Local Law.** If implied warranties may not be disclaimed under applicable law, then ANY IMPLIED WARRANTIES ARE LIMITED IN DURATION TO THE PERIOD REQUIRED BY APPLICABLE LAW.

13. **Limitation of Liability.** INDEPENDENT OF THE FORGOING PROVISIONS, IN NO EVENT AND UNDER NO LEGAL THEORY, INCLUDING WITHOUT LIMITATION, TORT, CONTRACT, OR STRICT PRODUCTS LIABILITY, SHALL MENTOR GRAPHICS BE LIABLE TO YOU OR ANY OTHER PERSON FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY KIND, INCLUDING WITHOUT LIMITATION, DAMAGES FOR LOSS OF GOODWILL, WORK STOPPAGE, COMPUTER MALFUNCTION, OR ANY OTHER KIND OF COMMERCIAL DAMAGE, EVEN IF MENTOR GRAPHICS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THIS LIMITATION SHALL NOT APPLY TO LIABILITY FOR DEATH OR PERSONAL INJURY TO THE EXTENT PROHIBITED BY APPLICABLE LAW. IN NO EVENT SHALL MENTOR GRAPHICS' LIABILITY FOR ACTUAL DAMAGES FOR ANY CAUSE WHATSOEVER, AND REGARDLESS OF THE FORM OF ACTION, EXCEED THE AMOUNT PAID BY YOU IN FEES UNDER THIS AGREEMENT DURING THE PREVIOUS ONE YEAR PERIOD.

14. **Export Controls.** You agree to comply with all export laws and restrictions and regulations of the United States or foreign agencies or authorities, and not to export or re-export the Software or any direct product thereof in violation of any such restrictions, laws or regulations, or without all necessary approvals. As applicable, each party shall obtain and bear all expenses relating to any necessary licenses and/or exemptions with respect to its own export of the Software from the U.S. Neither the Software nor the underlying information or technology may be electronically transmitted or otherwise exported or re-exported (i) into Cuba, Iran, Iraq, Libya, North Korea, Sudan, Syria or any other country subject to U.S. trade sanctions covering the Software, to individuals or entities controlled by such countries, or to nationals or residents of such countries other than nationals who are lawfully admitted permanent residents of countries not subject to such sanctions; or (ii) to anyone on the U.S. Treasury Department's list of Specially Designated Nationals and Blocked Persons or the U.S. Commerce Department's Table of Denial Orders. By downloading or using the Software, Licensee agrees to the foregoing and represents and warrants that it complies with these conditions.

15. **U.S. Government End-Users.** The Software is a “commercial item,” as that term is defined in 48 C.F.R. 2.101 (Oct. 1995), consisting of “commercial computer software” and “commercial computer software documentation,” as such terms are used in 48 C.F.R. 12.212 (Sept. 1995). Consistent with 48 C.F.R. 12.212 and 48 C.F.R. 227.7202-1 through 227.7202-4 (June 1995), all U.S. Government End Users acquire the Software with only those rights set forth herein.

16. **Licensee Outside The U.S.** If You are located outside the U.S., then the following provisions shall apply: (i) Les parties aux presentes confirment leur volonte que cette convention de meme que tous les documents y compris tout avis qui siy rattache, soient rediges en langue anglaise (translation: “The parties confirm that this Agreement and all related documentation is and will be in the English language.”); and (ii) You are responsible for complying with any local laws in your jurisdiction which might impact your right to import, export or use the Software, and

You represent that You have complied with any regulations or registration procedures required by applicable law to make this license enforceable.

17. **Severability.** If any provision of this Agreement is declared invalid or unenforceable, such provision shall be deemed modified to the extent necessary and possible to render it valid and enforceable. In any event, the unenforceability or invalidity of any provision shall not affect any other provision of this Agreement, and this Agreement shall continue in full force and effect, and be construed and enforced, as if such provision had not been included, or had been modified as above provided, as the case may be.

18. **Arbitration.** Except for actions to protect intellectual property rights and to enforce an arbitrator's decision hereunder, all disputes, controversies, or claims arising out of or relating to this Agreement or a breach thereof shall be submitted to and finally resolved by arbitration under the rules of the American Arbitration Association ("AAA") then in effect. There shall be one arbitrator, and such arbitrator shall be chosen by mutual agreement of the parties in accordance with AAA rules. The arbitration shall take place in Granite Bay, California, and may be conducted by telephone or online. The arbitrator shall apply the laws of the State of California, USA to all issues in dispute. The controversy or claim shall be arbitrated on an individual basis, and shall not be consolidated in any arbitration with any claim or controversy of any other party. The findings of the arbitrator shall be final and binding on the parties, and may be entered in any court of competent jurisdiction for enforcement. Enforcements of any award or judgment shall be governed by the United Nations Convention on the Recognition and Enforcement of Foreign Arbitral Awards. Should either party file an action contrary to this provision, the other party may recover attorney's fees and costs up to $1000.00.

19. **Jurisdiction And Venue.** The courts of Placer County in the State of California, USA and the nearest U.S. District Court shall be the exclusive jurisdiction and venue for all legal proceedings that are not arbitrated under this Agreement.

20. **Independent Contractors.** The relationship of the parties is that of independent contractor, and nothing herein shall be construed to create a partnership, joint venture, franchise, employment, or agency relationship between the parties. Licensee shall have no authority to enter into agreements of any kind on behalf of Mentor Graphics and shall not have the power or authority to bind or obligate Mentor Graphics in any manner to any third party.

21. **Force Majeure.** Neither Mentor Graphics nor Licensee shall be liable for damages for any delay or failure of delivery arising out of causes beyond their reasonable control and without their fault or negligence, including, but not limited to, Acts of God, acts of civil or military authority, fires, riots, wars, embargoes, or communications failure.

22. **Miscellaneous.** This Agreement constitutes the entire understanding of the parties with respect to the subject matter of this Agreement and merges all prior communications, representations, and agreements. This Agreement may be modified only by a written agreement signed by the parties. If any provision of this Agreement is held to be unenforceable for any reason, such provision shall be reformed only to the extent necessary to make it enforceable. This Agreement shall be construed under the laws of the State of California, USA, excluding rules regarding conflicts of law. The application of the United Nations Convention of Contracts for the International Sale of Goods is expressly excluded. This license is written in English, and English is its controlling language.

B.3. Attribution

This version of Sourcery CodeBench Lite may include code based on work under the following copyright and permission notices:

B.3.1. Android Open Source Project

```
/*
 * Copyright (C) 2008 The Android Open Source Project
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *  * Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 *  * Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in
 *    the documentation and/or other materials provided with the
 *    distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 * "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
 * LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS
 * FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE
 * COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT,
 * INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING,
 * BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS
 * OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
 * AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
 * OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT
 * OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */
```

B.3.2. Newlib

```
The newlib subdirectory is a collection of software from several sources.

Each file may have its own copyright/license that is embedded in the source
file.  Unless otherwise noted in the body of the source file(s), the following copyright
notices will apply to the contents of the newlib subdirectory:

(1) Red Hat Incorporated

Copyright (c) 1994-2007  Red Hat, Inc. All rights reserved.

This copyrighted material is made available to anyone wishing to use,
modify, copy, or redistribute it subject to the terms and conditions
of the BSD License.   This program is distributed in the hope that
it will be useful, but WITHOUT ANY WARRANTY expressed or implied,
including the implied warranties of MERCHANTABILITY or FITNESS FOR
A PARTICULAR PURPOSE.  A copy of this license is available at
http://www.opensource.org/licenses. Any Red Hat trademarks that are
incorporated in the source code or documentation are not subject to
the BSD License and may only be used or replicated with the express
permission of Red Hat, Inc.

(2) University of California, Berkeley

Copyright (c) 1981-2000 The Regents of the University of California.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification,
are permitted provided that the following conditions are met:

    * Redistributions of source code must retain the above copyright notice,
      this list of conditions and the following disclaimer.
    * Redistributions in binary form must reproduce the above copyright notice,
      this list of conditions and the following disclaimer in the documentation
      and/or other materials provided with the distribution.
    * Neither the name of the University nor the names of its contributors
      may be used to endorse or promote products derived from this software
      without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
```

AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.
IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT,
INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY
OF SUCH DAMAGE.

(3) David M. Gay (AT&T 1991, Lucent 1998)

The author of this software is David M. Gay.

Copyright (c) 1991 by AT&T.

Permission to use, copy, modify, and distribute this software for any
purpose without fee is hereby granted, provided that this entire notice
is included in all copies of any software which is or includes a copy
or modification of this software and in all copies of the supporting
documentation for such software.

THIS SOFTWARE IS BEING PROVIDED "AS IS", WITHOUT ANY EXPRESS OR IMPLIED
WARRANTY. IN PARTICULAR, NEITHER THE AUTHOR NOR AT&T MAKES ANY
REPRESENTATION OR WARRANTY OF ANY KIND CONCERNING THE MERCHANTABILITY
OF THIS SOFTWARE OR ITS FITNESS FOR ANY PARTICULAR PURPOSE.

The author of this software is David M. Gay.

Copyright (C) 1998-2001 by Lucent Technologies
All Rights Reserved

Permission to use, copy, modify, and distribute this software and
its documentation for any purpose and without fee is hereby
granted, provided that the above copyright notice appear in all
copies and that both that the copyright notice and this
permission notice and warranty disclaimer appear in supporting
documentation, and that the name of Lucent or any of its entities
not be used in advertising or publicity pertaining to
distribution of the software without specific, written prior
permission.

LUCENT DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE,
INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS.
IN NO EVENT SHALL LUCENT OR ANY OF ITS ENTITIES BE LIABLE FOR ANY
SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER
IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION,
ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF
THIS SOFTWARE.

(4) Advanced Micro Devices

Copyright 1989, 1990 Advanced Micro Devices, Inc.

This software is the property of Advanced Micro Devices, Inc (AMD) which
specifically grants the user the right to modify, use and distribute this
software provided this notice is not removed or altered. All other rights
are reserved by AMD.

AMD MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS
SOFTWARE. IN NO EVENT SHALL AMD BE LIABLE FOR INCIDENTAL OR CONSEQUENTIAL
DAMAGES IN CONNECTION WITH OR ARISING FROM THE FURNISHING, PERFORMANCE, OR
USE OF THIS SOFTWARE.

So that all may benefit from your experience, please report any problems
or suggestions about this software to the 29K Technical Support Center at
800-29-29-AMD (800-292-9263) in the USA, or 0800-89-1131 in the UK, or
0031-11-1129 in Japan, toll free. The direct dial number is 512-462-4118.

```
Advanced Micro Devices, Inc.
29K Support Products
Mail Stop 573
5900 E. Ben White Blvd.
Austin, TX 78741
800-292-9263

(5) C.W. Sandmann

Copyright (C) 1993 C.W. Sandmann

This file may be freely distributed as long as the author's name remains.

(6) Eric Backus

(C) Copyright 1992 Eric Backus

This software may be used freely so long as this copyright notice is
left intact.  There is no warrantee on this software.

(7) Sun Microsystems

Copyright (C) 1993 by Sun Microsystems, Inc. All rights reserved.

Developed at SunPro, a Sun Microsystems, Inc. business.
Permission to use, copy, modify, and distribute this
software is freely granted, provided that this notice is preserved.

(8) Hewlett Packard

(c) Copyright 1986 HEWLETT-PACKARD COMPANY

To anyone who acknowledges that this file is provided "AS IS"
without any express or implied warranty:
    permission to use, copy, modify, and distribute this file
for any purpose is hereby granted without fee, provided that
the above copyright notice and this notice appears in all
copies, and that the name of Hewlett-Packard Company not be
used in advertising or publicity pertaining to distribution
of the software without specific, written prior permission.
Hewlett-Packard Company makes no representations about the
suitability of this software for any purpose.

(9) Hans-Peter Nilsson

Copyright (C) 2001 Hans-Peter Nilsson

Permission to use, copy, modify, and distribute this software is
freely granted, provided that the above copyright notice, this notice
and the following disclaimer are preserved with no changes.

THIS SOFTWARE IS PROVIDED ``AS IS'' AND WITHOUT ANY EXPRESS OR
IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
PURPOSE.

(10) Stephane Carrez (m68hc11-elf/m68hc12-elf targets only)

Copyright (C) 1999, 2000, 2001, 2002 Stephane Carrez (stcarrez@nerim.fr)

The authors hereby grant permission to use, copy, modify, distribute,
and license this software and its documentation for any purpose, provided
that existing copyright notices are retained in all copies and that this
notice is included verbatim in any distributions. No written agreement,
license, or royalty fee is required for any of the authorized uses.
Modifications to this software may be copyrighted by their authors
and need not follow the licensing terms described here, provided that
the new terms are clearly indicated on the first page of each file where
they apply.

(11) Christopher G. Demetriou
```

(12) SuperH, Inc.

So that all may benefit from your experience, please report any problems
or suggestions about this software to the SuperH Support Center via
e-mail at softwaresupport@superh.com .

SuperH, Inc.
405 River Oaks Parkway
San Jose
CA 95134
USA

(13) Royal Institute of Technology

```
EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL KTH OR ITS CONTRIBUTORS BE
LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR
BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF
ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

(14) Alexey Zelkin

Copyright (c) 2000, 2001 Alexey Zelkin <phantom@FreeBSD.org>
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

(15) Andrey A. Chernov

Copyright (C) 1997 by Andrey A. Chernov, Moscow, Russia.
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED.  IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

(16) FreeBSD

Copyright (c) 1997-2002 FreeBSD Project.
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
```

```
   notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

(17) S. L. Moshier

Author:  S. L. Moshier.

Copyright (c) 1984,2000 S.L. Moshier

Permission to use, copy, modify, and distribute this software for any
purpose without fee is hereby granted, provided that this entire notice
is included in all copies of any software which is or includes a copy
or modification of this software and in all copies of the supporting
documentation for such software.

THIS SOFTWARE IS BEING PROVIDED "AS IS", WITHOUT ANY EXPRESS OR IMPLIED
WARRANTY.  IN PARTICULAR,  THE AUTHOR MAKES NO REPRESENTATION
OR WARRANTY OF ANY KIND CONCERNING THE MERCHANTABILITY OF THIS
SOFTWARE OR ITS FITNESS FOR ANY PARTICULAR PURPOSE.

(18) Citrus Project

Copyright (c)1999 Citrus Project,
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

(19) Todd C. Miller

Copyright (c) 1998 Todd C. Miller <Todd.Miller@courtesan.com>
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
```

```
   documentation and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote products
   derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES,
INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY
AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL
THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS;
OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF
ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

(20) DJ Delorie (i386)
Copyright (C) 1991 DJ Delorie
All rights reserved.

Redistribution and use in source and binary forms is permitted
provided that the above copyright notice and following paragraph are
duplicated in all such forms.

This file is distributed WITHOUT ANY WARRANTY; without even the implied
warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

(21) Free Software Foundation LGPL License (*-linux* targets only)

   Copyright (C) 1990-1999, 2000, 2001    Free Software Foundation, Inc.
   This file is part of the GNU C Library.
   Contributed by Mark Kettenis <kettenis@phys.uva.nl>, 1997.

   The GNU C Library is free software; you can redistribute it and/or
   modify it under the terms of the GNU Lesser General Public
   License as published by the Free Software Foundation; either
   version 2.1 of the License, or (at your option) any later version.

   The GNU C Library is distributed in the hope that it will be useful,
   but WITHOUT ANY WARRANTY; without even the implied warranty of
   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
   Lesser General Public License for more details.

   You should have received a copy of the GNU Lesser General Public
   License along with the GNU C Library; if not, write to the Free
   Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA
   02110-1301 USA.

(22) Xavier Leroy LGPL License (i[3456]86-*-linux* targets only)

Copyright (C) 1996 Xavier Leroy (Xavier.Leroy@inria.fr)

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU Library General Public License
as published by the Free Software Foundation; either version 2
of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU Library General Public License for more details.

(23) Intel (i960)

Copyright (c) 1993 Intel Corporation

Intel hereby grants you permission to copy, modify, and distribute this
software and its documentation.  Intel grants this permission provided
that the above copyright notice appears in all copies and that both the
copyright notice and this permission notice appear in supporting
documentation.  In addition, Intel grants this permission provided that
you prominently mark as "not part of the original" any modifications
made to this software or documentation, and that the name of Intel
```

```
Corporation not be used in advertising or publicity pertaining to
distribution of the software or the documentation without specific,
written prior permission.

Intel Corporation provides this AS IS, WITHOUT ANY WARRANTY, EXPRESS OR
IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY
OR FITNESS FOR A PARTICULAR PURPOSE.  Intel makes no guarantee or
representations regarding the use of, or the results of the use of,
the software and documentation in terms of correctness, accuracy,
reliability, currentness, or otherwise; and you rely on the software,
documentation and results solely at your own risk.

IN NO EVENT SHALL INTEL BE LIABLE FOR ANY LOSS OF USE, LOSS OF BUSINESS,
LOSS OF PROFITS, INDIRECT, INCIDENTAL, SPECIAL OR CONSEQUENTIAL DAMAGES
OF ANY KIND.  IN NO EVENT SHALL INTEL'S TOTAL LIABILITY EXCEED THE SUM
PAID TO INTEL FOR THE PRODUCT LICENSED HEREUNDER.

(24) Hewlett-Packard  (hppa targets only)

(c) Copyright 1986 HEWLETT-PACKARD COMPANY

To anyone who acknowledges that this file is provided "AS IS"
without any express or implied warranty:
    permission to use, copy, modify, and distribute this file
for any purpose is hereby granted without fee, provided that
the above copyright notice and this notice appears in all
copies, and that the name of Hewlett-Packard Company not be
used in advertising or publicity pertaining to distribution
of the software without specific, written prior permission.
Hewlett-Packard Company makes no representations about the
suitability of this software for any purpose.

(25) Henry Spencer (only *-linux targets)

Copyright 1992, 1993, 1994 Henry Spencer.  All rights reserved.
This software is not subject to any license of the American Telephone
and Telegraph Company or of the Regents of the University of California.

Permission is granted to anyone to use this software for any purpose on
any computer system, and to alter it and redistribute it, subject
to the following restrictions:

1. The author is not responsible for the consequences of use of this
   software, no matter how awful, even if they arise from flaws in it.

2. The origin of this software must not be misrepresented, either by
   explicit claim or by omission.  Since few users ever read sources,
   credits must appear in the documentation.

3. Altered versions must be plainly marked as such, and must not be
   misrepresented as being the original software.  Since few users
   ever read sources, credits must appear in the documentation.

4. This notice may not be removed or altered.

(26) Mike Barcroft

Copyright (c) 2001 Mike Barcroft <mike@FreeBSD.org>
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
```

```
ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

(27) Konstantin Chuguev (--enable-newlib-iconv)

Copyright (c) 1999, 2000
   Konstantin Chuguev.  All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

   iconv (Charset Conversion Library) v2.0

(28) Artem Bityuckiy (--enable-newlib-iconv)

Copyright (c) 2003, Artem B. Bityuckiy, SoftMine Corporation.
Rights transferred to Franklin Electronic Publishers.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

(29) IBM, Sony, Toshiba (only spu-* targets)

  (C) Copyright 2001,2006,
  International Business Machines Corporation,
  Sony Computer Entertainment, Incorporated,
  Toshiba Corporation,

  All rights reserved.
```

```
  Redistribution and use in source and binary forms, with or without
  modification, are permitted provided that the following conditions are met:

    * Redistributions of source code must retain the above copyright notice,
      this list of conditions and the following disclaimer.
    * Redistributions in binary form must reproduce the above copyright
      notice, this list of conditions and the following disclaimer in the
      documentation and/or other materials provided with the distribution.
    * Neither the names of the copyright holders nor the names of their
      contributors may be used to endorse or promote products derived from this
      software without specific prior written permission.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
  AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
  IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
  ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
  LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
  CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
  SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
  INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
  CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
  ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
  POSSIBILITY OF SUCH DAMAGE.

(30) - Alex Tatmanjants (targets using libc/posix)

  Copyright (c) 1995 Alex Tatmanjants <alex@elvisti.kiev.ua>
   at Electronni Visti IA, Kiev, Ukraine.
    All rights reserved.

  Redistribution and use in source and binary forms, with or without
  modification, are permitted provided that the following conditions
  are met:
  1. Redistributions of source code must retain the above copyright
     notice, this list of conditions and the following disclaimer.
  2. Redistributions in binary form must reproduce the above copyright
     notice, this list of conditions and the following disclaimer in the
     documentation and/or other materials provided with the distribution.

  THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND
  ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
  IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
  ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR BE LIABLE
  FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
  DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
  OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
  HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
  LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
  OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
  SUCH DAMAGE.

(31) - M. Warner Losh (targets using libc/posix)

  Copyright (c) 1998, M. Warner Losh <imp@freebsd.org>
  All rights reserved.

  Redistribution and use in source and binary forms, with or without
  modification, are permitted provided that the following conditions
  are met:
  1. Redistributions of source code must retain the above copyright
     notice, this list of conditions and the following disclaimer.
  2. Redistributions in binary form must reproduce the above copyright
     notice, this list of conditions and the following disclaimer in the
     documentation and/or other materials provided with the distribution.

  THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
  ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
  IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
  ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
  FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
  DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
  OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
```

```
  HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
  LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
  OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
  SUCH DAMAGE.

(32) - Andrey A. Chernov (targets using libc/posix)

  Copyright (C) 1996 by Andrey A. Chernov, Moscow, Russia.
  All rights reserved.

  Redistribution and use in source and binary forms, with or without
  modification, are permitted provided that the following conditions
  are met:
  1. Redistributions of source code must retain the above copyright
     notice, this list of conditions and the following disclaimer.
  2. Redistributions in binary form must reproduce the above copyright
     notice, this list of conditions and the following disclaimer in the
     documentation and/or other materials provided with the distribution.

  THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND
  ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
  IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
  ARE DISCLAIMED.  IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
  FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
  DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
  OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
  HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
  LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
  OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
  SUCH DAMAGE.

(33) - Daniel Eischen (targets using libc/posix)

  Copyright (c) 2001 Daniel Eischen <deischen@FreeBSD.org>.
  All rights reserved.

  Redistribution and use in source and binary forms, with or without
  modification, are permitted provided that the following conditions
  are met:
  1. Redistributions of source code must retain the above copyright
     notice, this list of conditions and the following disclaimer.
  2. Redistributions in binary form must reproduce the above copyright
     notice, this list of conditions and the following disclaimer in the
     documentation and/or other materials provided with the distribution.

  THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
  ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
  IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
  ARE DISCLAIMED.  IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
  FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
  DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
  OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
  HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
  LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
  OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
  SUCH DAMAGE.


(34) - Jon Beniston (only lm32-* targets)

 Contributed by Jon Beniston <jon@beniston.com>

 Redistribution and use in source and binary forms, with or without
 modification, are permitted provided that the following conditions
 are met:
 1. Redistributions of source code must retain the above copyright
 notice, this list of conditions and the following disclaimer.
 2. Redistributions in binary form must reproduce the above copyright
 notice, this list of conditions and the following disclaimer in the
 documentation and/or other materials provided with the distribution.

 THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
```

```
 ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
 FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 SUCH DAMAGE.


(35) - ARM Ltd (arm and thumb variant targets only)

 Copyright (c) 2009 ARM Ltd
 All rights reserved.

 Redistribution and use in source and binary forms, with or without
 modification, are permitted provided that the following conditions
 are met:
 1. Redistributions of source code must retain the above copyright
    notice, this list of conditions and the following disclaimer.
 2. Redistributions in binary form must reproduce the above copyright
    notice, this list of conditions and the following disclaimer in the
    documentation and/or other materials provided with the distribution.
 3. The name of the company may not be used to endorse or promote
    products derived from this software without specific prior written
    permission.

 THIS SOFTWARE IS PROVIDED BY ARM LTD ``AS IS'' AND ANY EXPRESS OR IMPLIED
 WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF
 MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.
 IN NO EVENT SHALL ARM LTD BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED
 TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
 PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
 LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
 NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
 SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.


(36) - CodeSourcery, Inc.

Copyright (c) 2009 CodeSourcery, Inc.
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:
    * Redistributions of source code must retain the above copyright
      notice, this list of conditions and the following disclaimer.
    * Redistributions in binary form must reproduce the above copyright
      notice, this list of conditions and the following disclaimer in the
      documentation and/or other materials provided with the distribution.
    * Neither the name of CodeSourcery nor the
      names of its contributors may be used to endorse or promote products
      derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY CODESOURCERY, INC. ``AS IS'' AND ANY
EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL CODESOURCERY BE LIABLE FOR ANY
DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

(37) MIPS Technologies, Inc
/*
 * Copyright (c) 2009 MIPS Technologies, Inc.
 *
```

```
* All rights reserved.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
*
*      * Redistributions of source code must retain the above copyright
*        notice, this list of conditions and the following disclaimer.
*      * Redistributions in binary form must reproduce the above
*      copyright
*        notice, this list of conditions and the following disclaimer
*        in the documentation and/or other materials provided with
*        the distribution.
*      * Neither the name of MIPS Technologies Inc. nor the names of its
*        contributors may be used to endorse or promote products derived
*        from this software without specific prior written permission.
*
* THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
* "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
* LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
* A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
* OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
* SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
* LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
* DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
* THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
* (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
* OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
*/
```