Sourcery CodeBench Lite 5.1

Hexagon ELF

Sourcery CodeBench Lite 5.1 2012.03-150

Getting Started

Sourcery CodeBench Lite 5.1: Hexagon ELF: Sourcery CodeBench Lite 5.1 2012.03-150: Getting Started

CodeSourcery, Inc.

Abstract

This guide explains how to install and build applications with Sourcery CodeBench Lite 5.1, CodeSourcery's customized and validated version of the GNU Toolchain. Sourcery CodeBench Lite 5.1 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 5.1 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 5.1. This guide provides a step-by-step guide to installing Sourcery CodeBench Lite 5.1 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 5.1 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 5.1. 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 5.1 for Hexagon ELF”

This chapter contains information about using Sourcery CodeBench Lite 5.1 that is specific to Hexagon ELF targets. You should read this chapter to learn how to best use Sourcery CodeBench Lite 5.1 on your target system.

Chapter 4, “Using Sourcery CodeBench from the Command Line”

This chapter explains how to build applications with Sourcery CodeBench Lite 5.1 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, “Next Steps with Sourcery CodeBench”

This chapter describes where you can find additional documentation and information about using Sourcery CodeBench Lite 5.1 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 5.1 Release Notes”

This appendix contains information about changes in this release of Sourcery CodeBench Lite 5.1 for Hexagon ELF. You should read through these notes to learn about new features and bug fixes.

Appendix B, “Sourcery CodeBench Lite 5.1 Licenses”

This appendix provides information about the software licenses that apply to Sourcery CodeBench Lite 5.1. Read this appendix to understand your legal rights and obligations as a user of Sourcery CodeBench Lite 5.1.

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 5.1 for the first time. You may use this chapter as an abbreviated guide to the rest of this manual.

Sourcery CodeBench Lite 5.1 for Hexagon 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 5.1 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 5.1 in Chapter 3, “Sourcery CodeBench Lite 5.1 for Hexagon ELF”.

1.1. Installation and Set-Up

Install Sourcery CodeBench Lite 5.1 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 5.1, 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. Sourcery CodeBench Lite 5.1 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 5.1 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.

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 5.1. On bare-metal targets, you must specify a linker script using the `-T` option on your link command line.

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.

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”.

[1] http://go.mentor.com/codebench/

Chapter 2
Installation and Configuration

This chapter explains how to install Sourcery CodeBench Lite 5.1. You will learn how to:

1. Verify that you can install Sourcery CodeBench Lite 5.1 on your system.
2. Download the appropriate Sourcery CodeBench Lite 5.1 installer.
3. Install Sourcery CodeBench Lite 5.1.
4. Configure your environment so that you can use Sourcery CodeBench Lite 5.1.

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 `hexagon`.

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 5 (and later), Red Hat Enterprise Linux 5 (and later), SuSE Enterprise Linux 10 (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 5.1. 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 5.1 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 5.1 toolchain.

2.2.2. Host Hardware Requirements

In order to install and use Sourcery CodeBench Lite 5.1, you must have at least 512MB of available memory.

The amount of disk space required for a complete Sourcery CodeBench Lite 5.1 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` environment 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 5.1 for Hexagon ELF” for requirements that apply to the target system.

2.3. Downloading an Installer

If you have received Sourcery CodeBench Lite 5.1 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 5.1”.

You can download Sourcery CodeBench Lite 5.1 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 5.1 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 5.1

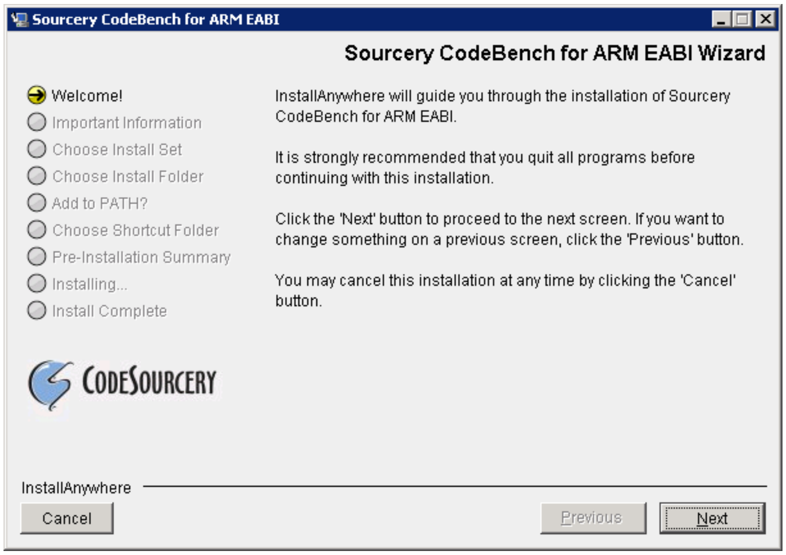
The method used to install Sourcery CodeBench Lite 5.1 depends on your host system and the kind of installation package you have downloaded.

2.4.1. Using the Sourcery CodeBench Lite 5.1 Installer on Microsoft Windows

If you have received Sourcery CodeBench Lite 5.1 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 5.1, double-click on the installer.

After the installer starts, follow the on-screen dialogs to install Sourcery CodeBench Lite 5.1. The installer is intended to be self-explanatory and on most pages the defaults are appropriate.

[1] http://go.mentor.com/codebench/
[2] https://sourcery.mentor.com/GNUToolchain/



InstallAnywhere will guide you through the installation of Sourcery CodeBench for ARM EABI.

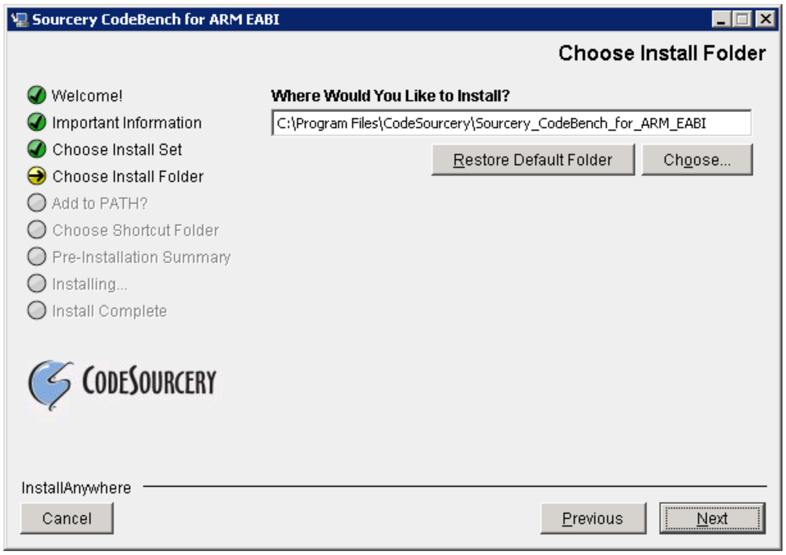
It is strongly recommended that you quit all programs before continuing with this installation.

Click the 'Next' button to proceed to the next screen. If you want to change something on a previous screen, click the 'Previous' button.

You may cancel this installation at any time by clicking the 'Cancel' button.

Running the Installer. The graphical installer guides you through the steps to install Sourcery CodeBench Lite 5.1.

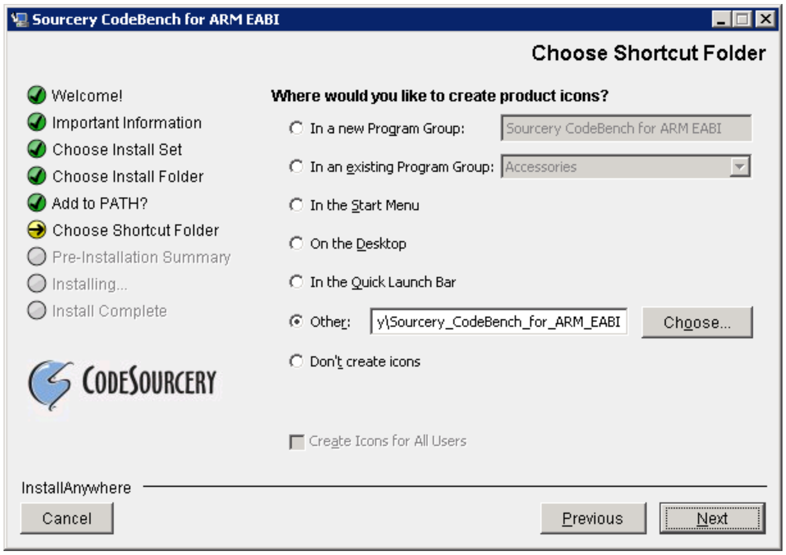
You may want to change the install directory pathname and customize the shortcut installation.



Where Would You Like to Install?

C:\Program Files\CodeSourcery\Sourcery_CodeBench_for_ARM_EABI

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



Where would you like to create product icons?

○ In a new Program Group: Sourcery CodeBench for ARM EABI

○ In an existing Program Group: Accessories

○ In the Start Menu

○ On the Desktop

○ In the Quick Launch Bar

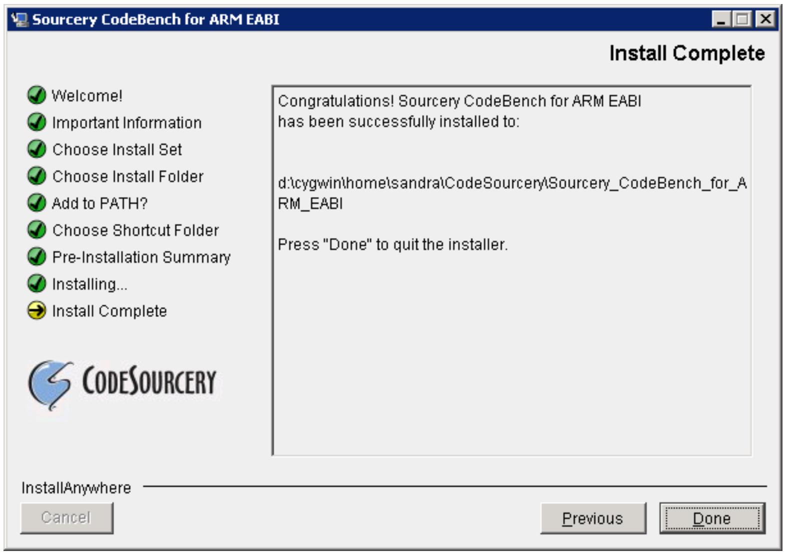
● Other: y\Sourcery_CodeBench_for_ARM_EABI

○ Don't create icons

□ Create Icons for All Users

Choose Shortcut Folder. You can customize where the installer creates shortcuts for quick access to Sourcery CodeBench Lite 5.1.

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.



Congratulations! Sourcery CodeBench for ARM EABI has been successfully installed to:

d:\cygwin\home\sandra\CodeSourcery\Sourcery_CodeBench_for_ARM_EABI

Press "Done" to quit the installer.

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 5.1 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 5.1. 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 5.1 from a Compressed Archive

You do not need to be a system administrator to install Sourcery CodeBench Lite 5.1 from a compressed archive. You may install Sourcery CodeBench Lite 5.1 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 5.1 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++-2012.03`.

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 5.1 Updates

If you have already installed an earlier version of Sourcery CodeBench Lite 5.1 for Hexagon 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 Hexagon ELF target all begin with `hexagon`. 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 5.1 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 5.1 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:

```
> hexagon-g++ -v
```

Verify that the last line of the output contains: `Sourcery CodeBench Lite 5.1 2012.03-150`.

2.6.1.2. Working with Cygwin

Sourcery CodeBench Lite 5.1 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 5.1 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 5.1 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 5.1 using the graphical installer then you may skip this step. The installer does this setup for you.

Before using Sourcery CodeBench Lite 5.1 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 5.1 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++-hexagon/man.

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

```
> hexagon-g++ -v
```

Verify that the last line of the output contains: Sourcery CodeBench Lite 5.1 2012.03-150.

2.7. Uninstalling Sourcery CodeBench Lite 5.1

The method used to uninstall Sourcery CodeBench Lite 5.1 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 hexagon directory located in the install directory will be removed entirely by the uninstaller.

2.7.1. Using the Sourcery CodeBench Lite 5.1 Uninstaller on Microsoft Windows

You should use the provided uninstaller to remove a Sourcery CodeBench Lite 5.1 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 5.1.

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 5.1 installation directory with the `-i console` command-line option.

To uninstall third-party drivers bundled with Sourcery CodeBench Lite 5.1, 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 5.1 Uninstaller on GNU/Linux

You should use the provided uninstaller to remove a Sourcery CodeBench Lite 5.1 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 5.1.

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 5.1 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 5.1 for Hexagon ELF

This chapter contains information about features of Sourcery CodeBench Lite 5.1 that are specific to Hexagon ELF targets. You should read this chapter to learn how to best use Sourcery CodeBench Lite 5.1 on your target system.

3.1. Included Components and Features

This section briefly lists the important components and features included in Sourcery CodeBench Lite 5.1 for Hexagon 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	7.0	Separate manual included.
Target libraries		
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 5.1 for Hexagon ELF includes the following library configuration.

Hexagon V4	
Command-line option(s):	default

Hexagon V5	
Command-line option(s):	`-march=hexagonv5`

Hexagon V55	
Command-line option(s):	`-march=hexagonv55`

Hexagon V4 -G0	
Command-line option(s):	`-G0`

Hexagon V5 -G0	
Command-line option(s):	`-march=hexagonv5 -G0`

Hexagon V55 -G0	
Command-line option(s):	`-march=hexagonv55 -G0`

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.

3.3. Use with Dinkumware Libraries

Sourcery CodeBench Lite 5.1 for Hexagon ELF does not include C or C++ libraries or header files. This toolchain is intended to be used only with the Dinkumware libraries, which you must obtain and install separately.

Because the Dinkumware libraries and header files may be installed in an arbitrary location on your host computer system, you must explicitly specify their pathnames on your compile and link commands. For example, if you set the environment variable `LIB` to point at the root of your Dinkumware install tree, use:

```
> hexagon-gcc options files \
  -I $LIB/include/common -I $LIB/include/common/c++ \
  -B $LIB/gnu/lib/gcc/hexagon/qdsp6/4.4.0 -B $LIB/dinkumware/lib
```

3.4. Compatibility with QuIC Toolchains

3.4.1. Bitfield Signedness

The ABI for Hexagon processors states that plain `int` bitfields are unsigned by default. This default is maintained in non-ANSI compatibility mode. With the `-fsigned-bitfields` or `-ansi` options, plain `int` bitfields are signed. See the GCC web site[1] for additional discussion of bitfield signedness.

3.4.2. Hardware Loop Support

The compiler features improved support for hardware looping instructions relative to legacy QuIC toolchains, including full support for nested loop constructs. Note that, due to limitations of the architecture, compiling to hardware loops may result in incorrect runtime behaviour if the iteration count exceeds 2^{32}-1. If this causes issues for your software, use the `-mno-hardware-loops` compiler option.

3.4.3. Linker Trampolines

The linker command-line option to disable trampolines has changed from `--trampolines=no` to `--no-trampolines` in order to bring the option syntax into conformance with the usual GNU conventions.

3.4.4. Alternate "Fast Math" Functions

The GCC command-line option `-ffast-math2` has been removed. You can use linker options instead to use the alternate math functions in your applications.

3.4.5. Small Data and Multilib Flags

The GCC command-line options `-mG0lib` and `-mbuilding-multilib` have been removed. Use `-G0` to disable small data sections. Using `-G 0` will trigger an error. Note that linking object files compiled with different small data options is not supported.

[1] http://gcc.gnu.org/onlinedocs/gcc/Non_002dbugs.html

Chapter 4
Using Sourcery CodeBench from the Command Line

This chapter demonstrates the use of Sourcery CodeBench Lite 5.1 from the command line.

4.1. Building an Application

This chapter explains how to build an application with Sourcery CodeBench Lite 5.1 using the command line. As elsewhere in this manual, this section assumes that your target system is hexagon, as indicated by the `hexagon` 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:

```
> hexagon-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 `hexagon/lib`.

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

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:

```
> hexagon-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 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.

4.3.2. 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
Next Steps with Sourcery CodeBench

This chapter describes where you can find additional documentation and information about using Sourcery CodeBench Lite 5.1 and its components.

5.1. 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.

5.2. Example Programs

Sourcery CodeBench Lite 5.1 includes some bundled example programs. You can find the source code for these examples in the `share/sourceryg++-hexagon-examples` directory of your Sourcery CodeBench installation.

5.2.1. 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.

5.3. Manuals for GNU Toolchain Components

Sourcery CodeBench Lite 5.1 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 5.1, 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 5.1, the documentation can be found in the `share/doc/sourceryg++-hexagon/` subdirectory of your installation directory.

In addition to the detailed reference manuals, Sourcery CodeBench Lite 5.1 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++-hexagon/man/man1
```

Then you can invoke `man` as:

```
> man ./hexagon-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://sourcery.mentor.com/GNUToolchain/

Finally, note that every command-line utility program included with Sourcery CodeBench Lite 5.1 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 5.1 Release Notes

This appendix contains information about changes in this release of Sourcery CodeBench Lite 5.1 for Hexagon ELF. You should read through these notes to learn about new features and bug fixes.

A.1. Changes in Sourcery CodeBench Lite 5.1 for Hexagon ELF

This section documents Sourcery CodeBench Lite 5.1 changes for each released revision.

A.1.1. Changes in Sourcery CodeBench Lite 5.1 2012.03-150

Fix to linker-generated trampolines. A bug that caused some re-used trampolines to jump to the wrong location has been fixed.

A.1.2. Changes in Sourcery CodeBench Lite 5.1 2012.03-143

No significant changes. There are no significant changes for Hexagon ELF in this release.

A.1.3. Changes in Sourcery CodeBench Lite 5.1 2012.03-139

Compiler fix for invalid packets. The compiler no longer produces packets using the result of a floating point instruction with `.new`.

Hardware loop bug fix. A compiler optimization bug has been fixed that caused some hardware loops to use an incorrect iteration count.

Fix for linker internal error. A linker bug has been fixed that caused it to abort with an internal error instead of producing a diagnostic for certain invalid symbol relocations.

Increase trampoline re-use. The linker has been enhanced to re-use trampolines up to 256 times where possible.

A.1.4. Changes in Sourcery CodeBench Lite 5.1 2012.03-138

Fix to packet optimization. A compiler optimization bug has been fixed that incorrectly changed the order of related write instructions.

Trampolines for local symbols. The linker now ensures that all trampolines for local symbols jump to their corresponding targets.

A.1.5. Changes in Sourcery CodeBench Lite 5.1 2012.03-133

Restriction on new-value compare jumps. New-value compare jump instructions cannot use the result of a floating-point instruction created in the same packet. The compiler no longer produces such packets.

Fix for internal compiler error. An internal compiler error (ICE) in `insert_value_copy_on_edge` has been fixed.

Fix for debug info bug. A compiler bug has been fixed that was causing incomplete DWARF-2 debug information for function prologues in some situations.

Define additional guest registers. The assembler now recognizes registers `g16` to `g19` and `gpmucnt4` to `gpmucnt7`.

Restriction on new-value compare jumps. New-value compare jump instructions cannot use the result of a floating-point instruction created in the same packet. The assembler now recognizes this combination and reports an error.

A.1.6. Changes in Sourcery CodeBench Lite 5.1 2012.03-130

Optimization fix for hardware loops. A GCC optimization bug has been fixed that sometimes caused incorrect code to be generated for hardware loops.

Fix for hardware loop compile error. An internal compiler error (ICE) in hwloop_optimize has been fixed.

Fix for internal compiler error. An internal compiler error (ICE) in `change_queue_index` has been fixed.

Fix for internal compiler error. An internal compiler error (ICE) in `merge_if_block` has been fixed.

GDB `info tlb` and `info pagetable` command enhancements.. GDB's `info tlb` and `info pagetable` commands have been modified to use the `qXfer:osdata` protocol.

Fix for GDB segmentation fault. A bug has been fixed that caused GDB to crash when reading a binary with incorrect debug information.

A.1.7. Changes in Sourcery CodeBench Lite 5.1 2012.03-129

Fix for `hexagon-gcc -mno-dealloc_return`. A code generation bug for `-mno-dealloc_return` has been fixed that sometimes caused an infinite loop.

A.1.8. Changes in Sourcery CodeBench Lite 5.1 2012.03-128

New compiler option. Support was added for `hexagon-gcc -mno-dealloc_return`.

New assembler option. Support was added for `hexagon-as -mno-dealloc_return`.

A.1.9. Changes in Sourcery CodeBench Lite 5.1 2012.03-126

No significant changes. There are no significant changes for Hexagon ELF in this release.

A.1.10. Changes in Sourcery CodeBench Lite 5.1 2012.03-125

`ld --dynamic-list` fixed for undefined symbols. The option `--dynamic-list` has been fixed so that creating an executable includes and export definitions for symbols not already defined.

Bug fix in `objdump --disassemble`. A disassembler bug is fixed that caused a crash for an unknown instruction.

A.1.11. Changes in Sourcery CodeBench Lite 5.1 2012.03-123

Optimization fix for packed structs. A GCC optimization bug has been fixed that could cause an unaligned access to a member of a packed struct.

A.1.12. Changes in Sourcery CodeBench Lite 5.1 2012.03-122

Optimization fix for packed structs. A GCC optimization bug has been fixed that caused an exception for some accesses to unaligned members of packed structures.

Fix `objcopy --change-section-lma` output. An objcopy bug was fixed that caused a bad header for the DYNAMIC segment when modifying section load addresses.

A.1.13. Changes in Sourcery CodeBench Lite 5.1 2012.03-121

Switch statement optimization bug fix. A GCC optimization bug has been fixed that caused incorrect code for some switch statements.

Support for Hexagon V55. The binutils tools now assemble/disassemble V55 instructions and recognize V55 register names.

GDB Hexagon v55 global registers support. GDB now supports displaying and setting Hexagon v55 global registers with the `info globalregs` and `set globalregs` commands.

A.1.14. Changes in Sourcery CodeBench Lite 5.1 2012.03-119

Update Python to 2.7.5. The version of Python used by GDB has been updated to 2.7.5.

A.1.15. Changes in Sourcery CodeBench Lite 5.1 2012.03-118

Missing debug info. A gcc bug has been fixed that prevented the debug option (`-g`) from being passed to the assembler.

Linker section merging. A linker bug that incorrectly placed sections with different write attributes into the same segment has been fixed.

Linker trampoline bug fixed. A linker bug that could cause incorrect sharing of a trampoline for long-range calls and jumps to local symbols has been fixed.

Linker trampoline in overlay sections. A linker bug that allowed invalid use of trampolines in overlapping overlay sections has been fixed.

Disassembly packets. A bug in objdump that caused disassembly to have incorrect packet boundaries has been fixed.

Linker trampoline bug fixed. A linker bug that could allow trampolines to become unreachable has been fixed.

GDB segmentation fault fix. A bug has been fixed that caused GDB to crash with a segmentation fault when the command `info globalregs` was used.

A.1.16. Changes in Sourcery CodeBench Lite 5.1 2012.03-114

Loop optimization fix. A bug that caused a compiler crash when optimizing certain patterns of loop code has been fixed.

Fix for linker script location counter assignments. A bug in the linker has been fixed that caused it to miscalculate the value for some assignments to the location counter.

A.1.17. Changes in Sourcery CodeBench Lite 5.1 2012.03-113

Linker trampoline bug fixed. A bug in the linker has been fixed that caused it to generate incorrect offsets to some trampolines for long-range calls and jumps.

A.1.18. Changes in Sourcery CodeBench Lite 5.1 2012.03-112

Linker output is often smaller. The linker now re-uses trampolines for long-range branches and calls when possible, instead of creating a new one for each use. This change results in smaller executables.

Installer `PATH` bug fix. An installer bug that caused it to set the `PATH` variable incorrectly has been fixed.

`cs-rm -f` bug fix. A bug that caused `cs-rm -f` on Windows hosts to incorrectly issue an error when passed a glob pattern that matched nothing has been fixed.

A.1.19. Changes in Sourcery CodeBench Lite 5.1 2012.03-111

Improved linker message. The linker now gives a better error message when it cannot emit a trampoline for a long-range branch.

Small data fix. An assembler bug has been fixed where small data sections were not marked with the `SHF_GPREL` flag.

A.1.20. Changes in Sourcery CodeBench Lite 5.1 2012.03-110

GDB breakpoint fix. A GDB bug has been fixed that caused it to set breakpoints on some optimized functions at an incorrect location.

Some title. Some text.

GDB support for Hexagon v55. Support for Hexagon v55 has been added to GDB.

A.1.21. Changes in Sourcery CodeBench Lite 5.1 2012.03-106

Optimizer bug fix. A bug that caused the compiler to generate incorrect code for read-modify-write operations at optimization level `-O2` and above has been fixed.

Fixed 64-bit `asrrnd`. The assembler gave an `invalid instruction` error for the 64-bit version of the `asrrnd` instruction. Now the assembler handles this instruction correctly.

Dynamic object generation for Hexagon V4. The linker no longer generates an incorrect PLT for Hexagon V4 or later.

File open errors in Windows-host toolchain. A bug that caused the binary utilities programs to fail with file open errors on Windows hosts has been fixed.

Diagnostic for invalid packet. The assembler now emits a `register `rN' modified more than once` error when it encounters a packet containing multiple writes to the same register predicated on the same condition.

Relocation range checking. A linker bug has been fixed that caused errors such as `ld: can't relax section: File format not recognized`.

Installer warnings fixed. A bug that caused Gtk warnings relating to `libappmenu.so` when running the installer on 64-bit Ubuntu GNU/Linux hosts has been fixed.

Backtrace bug fix. A bug has been fixed that caused GDB to display incorrect backtraces for optimized binaries.

Breakpoint trigger behavior fix. GDB now assumes the Hexagon ELF RTOS debug stub reports the PC of the next instruction and the Hexagon ELF simulator reports the PC of the current instruction when a software breakpoint is hit. This fixes a bug with incorrect reporting of the PC when stopped at a breakpoint.

Support for Hexagon ELF debug stub. GDB now supports debugging via a stub, available from Qualcomm. Previously, only the simulator (`hexagon-sim`) was supported.

Invalid memory read fix. A bug has been fixed that caused GDB to attempt to read target memory at address 0, thus resulting in an internal error.

Sourcery CodeBench Lite 5.1 naming. This edition of Sourcery CodeBench has been renamed to Sourcery CodeBench Lite 5.1 to match Qualcomm's version numbering. The included components and component versions have not changed since previous releases.

A.1.22. Changes in Sourcery CodeBench Lite 5.1 2012.03-101

Hexagon V55 support in QDSP6 compatibility mode. The compiler no longer encounters an ICE in `c_cpp_builtins` when invoked with both `-mv55` and `-mqdsp6-compat`.

Loop instruction restriction for Hexagon V4. A restriction in the assembler that prevented a packet containing jump instructions from also containing a loop set-up instruction has now been lifted for Hexagon V4 and above.

A.1.23. Changes in Sourcery CodeBench Lite 5.1 2012.03-100

Support for Hexagon architectures. Generating code for Hexagon V4 architecture is now the default. Support for older architectures is removed.

Fix for internal compiler error. A GCC bug has been fixed that caused it to report an internal compiler error in `add_var_loc_to_decl` when compiling some files with `-g`.

Hexagon V55 option handling. The linker no longer emits an `unrecognised emulation mode: v55` error when invoked with `-mv55`.

Support for Hexagon architectures. Generating code for Hexagon V4 architecture is now the default. Support for older architectures is removed.

Loop instruction restriction. For Hexagon V5 and above, a restriction in the assembler has been lifted that prevented a packet containing jump instructions from also containing a loop set-up instruction.

GDB duplex instruction bug. A bug has been fixed that prevented GDB from properly stepping through a group of instructions in a packet containing a duplex instruction. Another bug has been fixed that caused GDB to sometimes insert breakpoints at incorrect addresses for packets containing duplex instructions.

GDB segfault. A bug has been fixed that caused GDB to crash whenever the `maint print architecture` command was used.

A.1.24. Changes in Sourcery CodeBench Lite 5.1 2012.03-97

Hexagon V55 preprocessor macros. The behavior of preprocessor macros indicating Hexagon V55 has been modified. When compiling for that architecture version, the `__HEXAGON_ARCH__` macro is now set to `55`, and `__HEXAGON_V5__` is no longer defined.

Braces in disassembled code. Code disassembled with the `objdump` command now shows braces around packets containing only a single instruction (or duplex/compound instruction).

Hexagon simulator environment variables. A bug has been fixed that caused the Hexagon simulator not to inherit GDB's environment variables when it is launched via the GDB `start` or `run` commands on Windows hosts.

A.1.25. Changes in Sourcery CodeBench Lite 5.1 2012.03-96

Hexagon V55 support. The toolchain now contains nominal support for Hexagon V55.

Internal compiler error with intrinsics. A bug has been fixed that caused internal compiler errors when using certain intrinsics from the `hexagon_protos.h` header file.

Corrected loop initialization. A GCC bug has been fixed that caused incorrect code for some loop initializations.

Packet restriction for Hexagon V5. The assembler has been corrected so that a MEMOP-class instruction is no longer permitted in the same packet as a store instruction for Hexagon V5.

Hexagon V5 double-precision float instructions. Support for several double-precision floating point instructions which are no longer part of the Hexagon V5 architecture has been removed from the assembler.

Bitfield signedness. A defect has been fixed that caused plain `int` bitfields to default to signed behaviour, contrary to the Hexagon psABI.

A.1.26. Changes in Sourcery CodeBench Lite 5.1 2012.03-92

Stack-frame debug info fix. A bug in the stack-frame debug information emitted by the compiler has been fixed that caused `Could not find the frame base` errors when examining function arguments and local variables in the debugger.

Internal compiler error with virtual functions. A compiler bug has been fixed that caused internal compiler errors when compiling a virtual function in some cases of multiple or virtual inheritance.

Volatile memory accesses. For Hexagon V4 and above the compiler may now generate instruction packets containing 2 volatile memory accesses by default. This functionality was previously controlled by the command-line option `-mpacketize-volatiles`, which has been removed.

Missing register names fixed. The compiler now recognizes the register names `gp`, `cs0`, and `cs1` in inline assembly specifications. These registers are also now available in the debugger.

Duplicate `.debug_line` sections fixed. A GCC bug has been fixed that caused assembler errors when compiling with both the `-g` and `-Wa,-gdwarf2` options.

Double-precision float compatibility. The compiler no longer generates several double-precision floating-point instructions which have been removed from the Hexagon V5 architecture.

Section address adjustment bug fix. A bug has been fixed that caused `objcopy` to emit errors of the form *`section_name`* `can't be allocated in segment` *`segment_number`* when adjusting section addresses.

Profiler support for constant-propagated functions. A bug has been fixed in the `gprof` utility that caused it to report incorrect function names for cloned functions, generated by the compiler to take advantage of constant propagation optimization opportunities.

Linker bug fix for mixed architecture versions. A bug has been fixed that caused the linker to ignore the Hexagon architecture version of objects read from archives in determining how to mark the version of the output file.

Windows shortcuts. The linker is is now able to follow Windows shortcuts to locate needed files, such as libraries.

Install to empty directory failure fixed. A bug that prevented installation of Sourcery CodeBench Lite 5.1 into an existing empty directory has been fixed.

Debugging shared objects. The debugger is now capable of detecting shared library events. Previously, any breakpoints set pending a shared library load would never trigger, as the load event was never detected by the debugger.

GDB internal error bug fix. A bug has been fixed that caused the debugger to fail with an `Assertion `get_frame_type (frame) == DUMMY_FRAME' failed` error message. The bug was triggered by manually calling functions in the debugged program.

Updated system requirements. The host operating system requirements for Sourcery CodeBench Lite 5.1 have been updated. The minimum versions of GNU/Linux now supported are Red Hat Enterprise Linux 5, SuSE Enterprise Linux 10, Fedora Core 6, Ubuntu 8.04, and Debian 5, or later versions of these distributions running on 32-bit or 64-bit Intel or AMD CPUs.

`hexagon-gcov`. The code-coverage utility `hexagon-gcov` is no longer included in Sourcery CodeBench Lite 5.1.

A.1.27. Changes in Sourcery CodeBench Lite 5.1 2012.03-50

Compiler optimization improvements. The compiler now generates faster and/or smaller code in many cases due to optimization improvements. In addition, several performance regressions of the previous release relative to legacy QuIC toolchains have been addressed. Specific improvements include:

- The compiler now makes more efficient use of constant extenders in generated code for Hexagon V4 and above. For example, it now avoids duplicating the same absolute base address in multiple loads and stores.
- The compiler can now generate the floating-point multiply-accumulate instructions supported by Hexagon V5.
- Support for comparisons has been improved in the compiler, allowing (for example) better code to be generated for comparisons between `short` variables on Hexagon V4 and above.
- Hardware loop instructions are now generated more effectively for loops with `short` or `char` index variables.
- The compiler is now able to generate the conditional forms of the sign-extend and shift-halfword instructions, supported by Hexagon V4 and above.
- The compiler now uses post-increment and post-decrement addressing modes more effectively on all architecture versions, and generates conditional loads and stores on Hexagon V4 and above.
- The compiler now provides improved support for read-modify-write (`MEMOP` class) instructions on Hexagon V4 and above.

- The compiler now optimizes calls to `pow` with a constant base 2 into equivalent calls to `exp2` or `ldexp`.

Bitfield signedness. The signedness for plain bitfields defined with `int` defaults to unsigned.

Shared linking. The compiler driver now supports shared linking with the `-shared` option.

Packet constraint violations with `dcfetch`. The assembler now uniformly issues an error rather than attempting to remove excess `dcfetch` instructions to satisfy packet formation constraints. The compiler bug that caused such invalid packets to be generated has been fixed.

Big-endian ELF support. Support for big-endian ELF objects has been removed from the Hexagon ELF toolchain, as the architecture is little-endian only. Both the `-EB` and `-EL` assembler command-line options have been removed, since the default is always correct for Hexagon.

Assembler fix for local common. A bug has been fixed that caused the assembler to erroneously generate an `attempt to move .org backwards` error for some uses of the `.lcomm` directive.

Out-of-range branch veneers. The linker now generates smaller branch veneers for Hexagon V5 object files.

Disassembler output bug fix. The disassembler now correctly displays instructions using 32-bit absolute addressing modes.

Installer failure during upgrade. Some recent releases were affected by an installer bug on Windows hosts that caused installing a newer version of Sourcery CodeBench Lite 5.1 into the same directory to fail with the error `Sourcery CodeBench Lite 5.1 for Hexagon ELF upgrade failed`. This bug has now been fixed, but the affected releases cannot be upgraded. As a workaround, uninstall the older release before installing the new version.

A.1.28. Changes in Sourcery CodeBench Lite 5.1 2011.09-385

Incorrect compilation bug fix. A bug has been fixed which sometimes caused the compiler to incorrectly reorder writes to memory.

New-value jumps. The compiler now makes better use of new-value comparison jumps.

Position-independent code. A bug has been fixed in the compiler that caused it to emit code that is not position-independent when compiling with `-fPIC`.

Target prefix changed. The compiler and other executables for the bare-metal Hexagon ELF toolchain now use `hexagon-` as the target prefix instead of `hexagon-elf-`. For example, the C compiler is now named `hexagon-gcc`.

Constant-extended conditional loads/stores. A bug has been fixed that prevented the compiler from generating constant-extended conditional loads/stores; these are more efficient than conditional jumps.

Long calls. Code generation in the compiler has been improved by making better use of long calls.

Some target preprocessor defines removed. The preprocessor no longer defines `hexagon` or `__hexagon` by default. Similarly, `qdsp6` and `__qdsp6` are no longer defined when using `-mqdsp6-compat`.

Hexagon V5 intrinsics. Incorrect type information for Hexagon V5 intrinsics has been corrected in the compiler.

Hardware loops. The compiler now uses hardware loop instructions more aggressively. While this results in faster code, it can also result in incorrect program behavior if loop counters overflow. For more information see Section 3.4.2, "Hardware Loop Support". Additionally, hardware loop initialization code has been improved in many cases.

Improved support for out-of-range branches. Linker support for out-of-range branches has been improved for V4 compatible architectures.

Improved assembler support for Hexagon V5. Support for additional Hexagon V5 instructions has been added to the assembler.

Smaller executables. A bug in the linker has been fixed that cause it to create executables containing up to 512K of unnecessary padding bytes. Run-time memory usage was not affected by the bug, only the size of executable files.

Constant-extended `mpyi` instructions. A bug has been fixed that caused the assembler to reject `mpyi` instructions with a constant extender as their immediate value.

Assembler register parsing fix. A bug has been fixed that caused the assembler to incorrectly reject register pair syntax such as `lr:fp`.

A.1.29. Changes in Sourcery CodeBench Lite 5.1 2011.09-378

GCC comparison improvements. The compiler now generates better code for comparison operations, in particular when the result of a comparison is assigned to a variable. Better code is also generated in some cases for logical operations on Boolean values.

A.1.30. Changes in Sourcery CodeBench Lite 5.1 2011.09-375

Initial release. This is the initial release for Hexagon ELF.

Appendix B
Sourcery CodeBench Lite 5.1 Licenses

Sourcery CodeBench Lite 5.1 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 5.1. You should read this appendix to understand your legal rights and obligations as a user of Sourcery CodeBench Lite 5.1.

B.1. Licenses for Sourcery CodeBench Lite 5.1 Components

The table below lists the major components of Sourcery CodeBench Lite 5.1 for Hexagon 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 5.1. Sourcery CodeBench Lite 5.1 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 5.1.

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
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 5.1 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 5.1. You can develop proprietary applications and libraries with Sourcery CodeBench Lite 5.1.

Sourcery CodeBench Lite 5.1 may include some third party example programs and libraries in the `share/sourceryg++-hexagon-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.

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 5.1 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.
 */
```